Wesleyan ◇ University

A Bunched Homotopy Type Theory for Synthetic Stable Homotopy Theory

by

Mitchell Riley

Faculty Advisor: Dr. Daniel R. Licata

A dissertation submitted to the Faculty of Wesleyan University
in partial fulfilment of the requirements for the degree of Doctor of Philosophy

Middletown, Connecticut May 2022

Abstract

Homotopy type theory allows for a synthetic formulation of homotopy theory, where arguments can be checked by computer and automatically apply in many semantic settings. Modern homotopy theory makes essential use of the category of *spectra*, the natural setting in which to investigate 'stable' phenomena: the suspension and loop space operations become inverses. One can define a version of spectra internally to type theory, but this definition can be quite difficult to work with. In particular, there is not presently a convenient way to construct and manipulate the smash product and internal hom of such spectra. This thesis describes an extension of Martin-Löf Type Theory that is suitable for working with these constructions *synthetically*.

There is an ∞-topos of *parameterised spectra*, whose objects are an index space with a family of spectra over it, so standard homotopy type theory can be interpreted in this setting. To isolate the spaces (as objects with the trivial family of spectra) and the spectra (as objects with trivial indexing space), we extend type theory with a novel modality $\natural$ that is simultaneously a monad and a comonad. Intuitively, this modality keeps the base of an object the same but replaces the spectrum over each point with a trivial one.

The system is further extended with a monoidal tensor $\otimes$, unit $\mathbb{S}$ and internal hom $\multimap$, which capture abstractly the constructions on spectra mentioned above. We are lead naturally to consider a 'bunched' type theory, where the contexts have a tree-like structure. The modality is crucial for making dependency in these linear type formers work correctly: dependency between $\otimes$ 'bunches' is mediated by $\natural$.

To demonstrate that this type theory is usable in practice, we prove some basic synthetic results in this new system. For example, externally, any map of spaces induces a 'six-functor formalism' between the categories of parameterised spectra over those spaces, and this structure can be reconstructed internal to the type theory. We additionally investigate an axiom asserting that the internal category of spectra is semiadditive; we show that in the presence of univalence this in fact implies that the category of spectra is stable.

For my parents Mark and Leah,
to whom I owe everything

Acknowledgements

This thesis could not have been written without the limitless type-theoretic wisdom and equally limitless patience and kindness of my advisor Dan Licata. His support made it possible to persist past the many syntactic dead ends that I walked myself down, and he has my deepest gratitude.

I am grateful to the members of the HoTT community for countless enlightening conversations, particularly Eric Finster, Mike Shulman, Mathieu Anel, David Jaz Myers, and all of the HoTT MURI Team. I owe a particular debt to Eric, for putting forward what has proven to be a fruitful avenue for research.

Thank you also to my academic siblings Matthew Weaver, Joomy Korkut and Joseph Cutler, and to my compatriots at Wesleyan; Ed Morehouse, Alex Kavvos, Robert Rose, Will, Wengang, Justin, Dan, Zonia and Mostafa, and to Ana, for her wonderful support and companionship.

Finally, thank you to my friends in TYH, who without question diverted me to a happier life.

Contents

Introduction

Homotopy type theory [HoTTBook], the extension of Martin-Löf Type Theory with higher inductive types and Voevodsky's univalence axiom, has proven a useful tool for formalising homotopy-theoretic arguments in a manner that can be verified by computers. Rather than working with concrete topological spaces or simplicial sets, one works with types and the universal constructions provided by the type formers of the theory. Working this way sometimes requires clever new arguments for standard results, but the payoff is that the same proof applies in any model of the theory, not just in the homotopy theory of spaces. The basics of synthetic homotopy theory are presented in [HoTTBook, Chapter 8], and a significant number of results have been developed and formalized since [nLaba].

It is conjectured that all of homotopy type theory (as in [HoTTBook]) can be interpreted in any ∞-topos — the interpretation of the basic Martin-Löf type theory [AW09; LW15; AK11], Voevodsky's univalence axiom [KL21; GK17; Shu15; Shu19a] and a class of higher inductive types [LS20] have been worked out, though closure of the universes under higher inductive types is in progress.

One of the main advantages of HoTT is that all constructions performed in it are necessarily homotopy invariant. However, this is a double-edged sword, as it rules out some of the first definitions from algebraic topology one might hope to make.

For example, a key invariant of topological spaces is their homology and cohomology groups, which are often easier to calculate than their homotopy groups. Externally, the ordinary homology and cohomology of a space X are defined using maps into X from simplices of various dimensions. But these simplices are all contractible, so internal to type theory they are indistinguishable from the point. Instead, homology and cohomology can be defined in type theory [Cav15; Gra18] via the *spectra* that represent them.

Classically, the study of spectra was motivated by the Freudenthal suspension theorem (see [HoTTBook, Chapter 8] for a proof in type theory), which implies that, for nice spaces X and Y, the sequence obtained by repeated suspension

$$[X, Y] \to [\Sigma X, \Sigma Y] \to [\Sigma^2 X, \Sigma^2 Y] \to \dots$$

eventually *stabilises*. *Stable homotopy theory* studies the phenomena that survive after arbitrarily many suspensions, and it is an important tool for obtaining results in unstable homotopy theory as well — for example, the homotopy groups of spheres in a certain range coincide with certain stable homotopy groups. One can form a category out of the above observation, where the objects are nice pointed spaces and the hom-sets are given by the colimit over the above diagram, but the resulting category does not have very nice properties. This motivates passing to a category of *spectra* that includes the previous as a full subcategory, but is much better behaved categorically.

Using ∞-categorical technology (by which we mean $(\infty, 1)$-categorical), the ∞-category of spectra can be defined by starting with the ∞-category of pointed spaces $\mathcal{S}_*$, and inverting the loop space functor Ω in a universal way:

$$\mathrm{Spec} = \varprojlim \left(\cdots \xrightarrow{\Omega} \mathcal{S}_* \xrightarrow{\Omega} \mathcal{S}_* \xrightarrow{\Omega} \mathcal{S}_* \right)$$

In the ∞-category of spectra, the suspension Σ and loop space Ω functors are equivalences, in contrast to ordinary spaces where Σ and Ω are far from being equivalences. This makes Spec the right place to do stable homotopy theory: *everything* is stable under suspension.

Unwinding the above definition, a spectrum can be presented concretely as a sequence of pointed spaces $X_0, X_1, X_2, \ldots$, together with a pointed equivalence between each space and the loop space of the next, $X_0 \simeq_\star \Omega X_1$, $X_1 \simeq_\star \Omega X_2, \ldots$. This definition of spectra is readily internalised in type theory, replacing 'pointed spaces' with 'pointed types', and has been formalised [SDRPS19]. However, most basic constructions on spectra are still works in progress, as working with such spectra can be difficult. Instead of rebuilding the theory of spectra from scratch within type theory, we could instead use a type theory that provides some of the structure built-in.

It would therefore be nice to have a type theory that allows us to manipulate spectra in an ergonomic way. The issue is that spectra behave much like objects from algebra, such as modules over a ring or chain complexes, rather than like topological spaces or sets. The category of spectra is not cartesian closed, rather it is monoidal closed for an operation on spectra called the smash product. Because of this inherent 'linear' nature, we cannot model MLTT directly in the category of spectra. Instead, our strategy is to consider a model in *parameterised spectra*: space-indexed families of spectra.

The ∞-category $P\mathrm{Spec}$ of parameterised spectra can be pleasantly described as follows [ABG18]: Given a space X, one can think of it as an ∞-groupoid and consider the functor ∞-category $\mathrm{Fun}(X, \mathrm{Spec})$. The objects of this are our parameterised spectra over X. This assignment from spaces to functor categories assembles into a functor $\mathcal{S}^{\mathrm{op}} \to \mathrm{Cat}_\infty$, with Cat_∞ the ∞-category of ∞-categories. An ∞-categorical version of the Grothendieck construction yields the ∞-category $P\mathrm{Spec}$ of parameterised spectra.

These objects have both linear and nonlinear aspects: the indexing space behaves nonlinearly and the spectra associated to each index behave linearly. The ∞-category of parameterised spectra was shown to be an ∞-topos by Joyal and Biedermann [Joy08], and so admits a model of MLTT with the univalence axiom. Thus, we can begin by interpreting HoTT into this ∞-topos. There are important practical advantages over designing a new type theory from scratch for just the category of spectra, not the families: we can reuse all of the synthetic homotopy theory that has been developed for an arbitrary ∞-topos, and we can reuse existing proof assistants to work in this setting.

The 'Underlying Space' Modality

When working in HoTT with the model in parameterised spectra in mind, it will be necessary to have some syntactic way to say when a type denotes a single spectrum, rather than a family of them — for example, the fact that suspension and loop space are an equivalence holds only for individual spectra, not families thereof. Moreover, it will be necessary to say when a type (in $P\mathrm{Spec}$)

denotes an ordinary space; for example, some constructions that we will perform apply only to families of spectra over a space, not families of spectra over a spectrum, so in certain places we will want to restrict dependency to types that denote spaces. It turns out that we can access both the full subcategory of spaces and the full subcategory of spectra by way of a *modality* [HoTTBook; RSS20].

A basic operation on PSpec is the functor that, given a parameterised spectrum, extracts its underlying space. In the other direction, we can assign to each space X the parameterised spectrum (X, const_0), where $\mathsf{const}_0 : X \to \mathrm{Spec}$ denotes the constant functor at the zero spectrum (the spectrum with the one-point space at each level). Because 0 is both initial and terminal in Spec, this functor is both left and right adjoint to the forgetful functor from that extracts the index space:

$$\begin{tikzcd} P\mathrm{Spec} \arrow[r] \arrow[r, phantom, shift left=2.5, "\top"] \arrow[r, phantom, shift right=2.5, "\top"'] & \mathcal{S} \arrow[l, bend right=50, "0"'] \arrow[l, bend left=50, "0"] \end{tikzcd}$$

This diagram satisfies the additional coherence of a bireflective subcategory [FOPTST99]. This means that the roundtrip on PSpec, which we write as $\natural$, has a number of special properties: is a left-exact idempotent monad and comonad, it is adjoint to itself, and the counit followed by the unit $\natural E \to E \to \natural E$ is the identity. Then, we can isolate the spaces within the parameterised spectra as the $\natural$-modal types, roughly the types A such that $A \simeq \natural A$ — i.e., an object is a space iff it is equivalent to the object consisting of the same base space with the 0-spectrum over every point, because that means the object has no interesting spectra to begin with. Dually, we can isolate the spectra as the $\natural$-connected types, the types such that $\natural A$ is contractible — i.e., an object is a single spectrum iff it is a family of spectra over the point.

Monadic modalities are well-studied in axiomatic HoTT [RSS20], but to describe non-trivial comonadic modalities, one must modify the judgemental structure of the context. The adjunction between $\mathcal{S}$ and PSpec described above is a degenerate example of *axiomatic cohesion* [Law07]. An ∞-topos $\mathcal{T}$ is cohesive (over $\mathcal{S}$) if there is a string of adjoint functors relating $\mathcal{T}$ to $\mathcal{S}$, satisfying certain conditions. These adjunctions induce a string of adjoint functors $\int \dashv \flat \dashv \sharp$ on $\mathcal{T}$ so that $\int$ and $\sharp$ are monads and $\flat$ is a comonad. In our setting, we have $\int \equiv \flat \equiv \sharp$, hence the name $\natural$ for our roundtrip operation.

A 'cohesive type theory' capturing the structure of cohesion was introduced by Shulman [Shu18], where the three functors also appear as unary type formers. The part of the theory we are especially interested in is the $\flat/\sharp$ fragment, which on its own is called 'spatial type theory'. In spatial type theory, the context is divided into two zones following [Bar96; PD01]: a 'modal' context zone, where the types of all variables are thought of as invisibly prefixed with a $\flat$, followed by an ordinary context zone with ordinary variables. The rules for the $\flat$ and $\sharp$ modalities move variables between the zones to enforce the correct relationships between the modalities and the judgemental version of $\flat$ represented by the modal context zone.

In principle one could use spatial type theory to reason about PSpec by adding axioms asserting that the modalities are all equivalent. While we have no objection in principle to adding axioms to type theories for synthetic homotopy theory (we extend HoTT, which already adds univalence and higher inductive types as axioms), the ergonomics of such axioms would be poor — the equivalence between $\flat$ and $\sharp$ in this setting is so fundamental that transport across these equivalences would be pervasive in every construction.

In Chapter 1 of this thesis, we describe an extension of the judgements and rules of MLTT which give the $\natural$ modality the correct properties without the need for any axioms. Relative to existing modal dependent type theories, the primary difficulty of the $\natural$ modality is that the bireflection induces a non-trivial roundtrip on every type[1]: the unit map $A \to \natural A$ followed by the counit map $\natural A \to A$. The syntax has a pair of novel features to handle this. Firstly, each variable may be used in two ways: as normal, or "marked", which is written $\underline{x}$ and means using it via the roundtrip described above. Secondly, in contrast to spatial type theory, there is no separation of the context into two zones, one modal and one non-modal. The presence of both the unit and counit mean that modal and ordinary variables can be mixed together without restriction.

Linear Operations

Parameterised spectra also inherit a monoidal closed structure extending the monoidal closed structure on spectra, but HoTT on its own, even extended with $\natural$, is still missing is a way to access this structure. For this we would like to add linear type-formers $\otimes$ and $\multimap$ corresponding to the monoidal closure. Combining linearity and dependent types is a difficult problem, and there have been many different attempts and strategies.

The overriding principle of this work is that the theory ought to be an extension of ordinary dependent type theory. There is a large body of work in synthetic homotopy theory that we do not want to have to reconstruct in an entirely new system. This rules out all the previous work on linear dependent type theories (as far as we are aware), where either the type theory ends up incompatible with MLTT [Vák14; IP98; FKS20], or linear and nonlinear types inhabit different sorts [Isa21; KPB15; Lun18]. In the latter case, the existing synthetic homotopy theory results would only apply to the nonlinear part of the theory, and we would have to begin from scratch in the linear part. By contrast, in an extension of ordinary dependent type theory, the usual constructions of synthetic homotopy theory would be applicable directly to types containing linear information. For example, the 'suspension' of a spectrum should be defined using the same higher-inductive type as usual [HoTTBook, §6.5], rather as a new type-former just for linear types.

To achieve this, the new linear type-formers should be to be *freely* added to MLTT, so that they produce types that can be intermingled with ordinary types as one pleases. Freely combining a nonlinear and linear type theory has been studied in the non-dependent setting in the form of *bunched implication* [OP99; OHe03]. In bunched implication there are two binary context-forming operations, an ordinary cartesian comma and a linear monoidal product. These products can be combined in arbitrary ways, giving contexts an inherently tree-like structure.

After introducing $\natural$, we present a dependent version of bunched implication that additionally solves some of the metatheoretic issues of simply-typed bunched implication. Our theory follows a similar idea to the simply-typed version: we will have structured contexts that allow us to combine bunches of the context with a monoidal product, in addition to ordinary context extension. The central question then is how dependency works between assumptions that lie in different 'monoidally combined' bunches. The missing ingredient here is exactly the $\natural$ modality.

Importantly, the dependency permitted when using the ordinary MLTT type-formers is just as usual. For example, given two terms $a, a' : A$ we will always be able to form the identity type

[1]In full detail, the round-trip only applies to types that depend on a space, not a spectrum.

$a =_A a' : \mathcal{U}$, regardless of how A, a and a' use the linear features of the theory. Because of this, the new judgemental structure does not interfere with existing work in HoTT, which can be imported essentially unchanged.

The challenge in designing this type theory is in making it usable on pen and paper in a similar informal style to the HoTT Book and the synthetic work that follows it. The main trick is that we separate the bunched shape of the context from the typing information of the variables in it. A generic context $\Phi \mid \Gamma$ ctx has two pieces, a *palette* Φ describing the bunched structure of the context, and then a more typical context Γ of variables with types. Each variable is labelled with a *colour* from the palette, which places the data of that variable at the corresponding node of the tree. This allows us to refer to sections of the context using labels in the palette, rather than using variable names.

Synthetic Stable Homotopy Theory

To demonstrate that the resulting type theory is quite ergonomic to use, even without a proof assistant, we spend Chapter 2 developing some synthetic homotopy theory informally in the style of [HoTTBook].

While introducing the type formers we prove various basic properties that are essentially 'getting out what we put in'; reifying the structure of contexts in the types. The emergent behaviour is more interesting, and some of the results are unexpected.

As mentioned above, the types that correspond to spaces can be identified as those that are $\natural$-modal: types X for which the canonical map $X \to \natural X$ is an equivalence. We can further identify the "parameterised spectra over X" as types A equipped with an equivalence between $\natural A$ and X. We show that for any internal map of spaces $X \to Y$, there is an induced "six functor formalism" between the spectra over X and the spectra over Y.

Our "synthetic spectra" thus far do not have all of the properties of actual spectra. To bring the synthetic spectra closer to actual spectra, we identify a pair of axioms. The first is a 'stability' axiom that forces the internal category of synthetic spectra to be *stable*, particular making $\Sigma \dashv \Omega$ an adjoint equivalence for these types. It turns out to be sufficient to assert that coproducts and products of synthetic spectra coincide, an apparently weaker property (Theorem 2.2.18).

The type theory augmented with this axiom (conjecturally) has models in parameterised families in *any* stable ∞-category. This includes the trivial stable ∞ category, whose families are just ordinary spaces, so this axiom does not preclude the trivial model in spaces! To home in on parameterised spectra, we add a second axiom which connects synthetic spectra to 'analytic' spectra, defined concretely as sequences of pointed types with connecting maps. We show that these axioms are strong enough to fix the homotopy groups (appropriately defined) of the synthetic sphere spectrum to be equivalent to the stable homotopy groups of the ordinary higher inductive spheres.

Chapter 1

Foundations

We begin with Homotopy Type Theory as in [HoTTBook], meaning Martin-Löf Type Theory (0, 1, 2, Π, Σ, Id, $\mathcal{U}$) with (higher) inductive types and the univalence axiom. In the next few sections we extend this theory, gradually adding additional judgemental structure to the contexts and corresponding type formers.

- Section 1.1 introduces the concept of 'marked' variables and variable uses, which are used to provide rules for the $\natural$ modality. The basic properties of this type former are studied in Section 1.1.3.
- Section 1.2 further extends the context structure with a 'palette', in preparation for the linear type formers. All variables are now additionally assigned a colour from this palette.
- Section 1.3 describes the $\otimes$-type former that internalises this new judgement structure as a type. We allow $\otimes$-types to be dependent in a certain sense: dependency between the two sides of a $\otimes$ is mediated by the $\natural$ modality.
- Section 1.4 describes $\mathbb{S}$, the monoidal unit type.
- Section 1.5 describes $\multimap$-types, the dependent right adjoint to $\otimes$, and shows that it interacts with the other type formers in various interesting ways. In Section 1.5.5, we find that univalence implies 'hom extensionality', much as univalence implies ordinary function extensionality.
- Section 1.6 shows that some of the induction principles for $\otimes$-types used in previous sections are derivable via the $\multimap$-type. The induction principles derivable this away are still not as strong as we might like, so we add a primitive notion of pattern matching to fill this gap.
- Section 1.7 describes the relationship between the present work and previous work on linear dependent type theories.

We introduce the rules of this theory piecemeal so to not overwhelm the reader. The complete collection of rules is listed in Section A.

1.1 The Modality

This section contains joint work with Dan Licata and Eric Finster [RFL21].

We begin with our new *modality*, a unary type constructor $\natural$. Unlike the monadic modalities studied in [RSS20] which can be described by axioms, our $\natural$ modality also has some comonadic aspects. By the no-go theorem of [Shu18, Theorem 4.1] for comonadic modalities, adding our $\natural$ modality will necessarily require changes to the judgements of the type theory. We first describe a judgemental version of the modality that applies to contexts and that has the desired unit and counit maps, and then give a type constructor that internalises this judgemental operation as a type.

1.1.1 Judgemental Rules

The new pieces of syntax and admissible operations appear in Figure 1.1, which we now discuss in more detail.

Putting aside the new context extension CTX-EXT-MARKED for a moment, we first have the ordinary VAR rule which does not interact with the new context structure at all. So let us inspect the first new variable rule VAR-ROUNDTRIP. At a high level, the reason we require some new judgemental structure for presenting the $\natural$ modality is that it has both a unit $A \to \natural A$ and a counit $\natural A \to A$ and the *roundtrip* $A \to \natural A \to A$ gives a non-trivial map for any type A. So to β-reduce $\natural$-INTRO followed by $\natural$-ELIM (once we have those rules), we need a judgemental version of this roundtrip map $A \to A$ to reduce to.

To achieve this, we add post-composition with the roundtrip as a new way of using variables in the rule VAR-ROUNDTRIP, in addition to the standard VAR rule. For a variable $x : A$, we write $\underline{x}$ for the roundtrip on A applied to x, and say that the variable usage is a *marked* variable usage. In our intended models, this keeps the base of x the same but replaces the fibres of x with the default sections of A. In concrete syntax, it is best to think of the variable usage $\underline{x}$ as a term constructor $\mathsf{underline}(x)$, rather than considering the underline as part of the variable name.

For a general term a, we define an admissible rule ROUNDTRIP $\underline{a}$ that "underlines all of the free variables in a", and denotes precomposing with the roundtrip on the context. There is an interaction with dependency, because applying the roundtrip to a term must also apply it to the term's type, so if $x : A$ is a variable then an instance of VAR-ROUNDTRIP gives $\underline{x} : \underline{A}$. On the left of : we have $\underline{x}$ as an indivisible piece of syntax, but on the right we have $\underline{A}$ which is the admissible ROUNDTRIP rule applied to the type A. We admit that this may be confusing initially, but is quite natural to use in practice.

For certain rules, we will need a judgement classifying terms whose free variables are *only* used marked, which we call *dull terms*. This is signalled by asking for terms in context '$\underline{\Gamma}$'. To make the type theory easier to use, we will make $\underline{\Gamma}$ an *admissible* operation (defined by induction on syntax), rather than a *derivable* one (a new piece of formal syntax). The new piece of formal syntax is instead a new context *extension* $\Gamma, \underline{x} : A$ (CTX-EXT-MARKED), which should be thought of as extending the context with $\natural A$ instead of A itself.

The restriction imposed by a marked context extension is that a variable that is *declared* marked in the context $\Gamma, \underline{x} : A$ can only be *used* marked as $\underline{x}$ in a term, as indicated in the VAR-MARKED rule.

$$\textsc{ctx-ext-marked}\ \dfrac{\Gamma\ \mathsf{ctx} \qquad \underline{\Gamma} \vdash A : \mathcal{U}}{\Gamma, \underline{x} : A\ \mathsf{ctx}}$$

$$\textsc{var}\ \dfrac{}{\Gamma, a : A, \Gamma' \vdash x : A} \qquad \textsc{var-roundtrip}\ \dfrac{}{\Gamma, x : A, \Gamma' \vdash \underline{x} : \underline{A}}$$

$$\textsc{var-marked}\ \dfrac{}{\Gamma, \underline{x} : A, \Gamma' \vdash \underline{x} : A}$$

'Natural' on contexts:

$$\textsc{ctx-mark}\ \begin{array}{c} \Gamma\ \mathsf{ctx} \\ \hdashline \underline{\Gamma}\ \mathsf{ctx} \end{array} \qquad \begin{array}{c} \underline{\cdot} :\equiv \cdot \\ \underline{\Gamma, x : A} :\equiv \underline{\Gamma}, \underline{x} : \underline{A} \\ \underline{\Gamma, \underline{x} : A} :\equiv \underline{\Gamma}, \underline{x} : A \end{array} \qquad \begin{array}{c} \Gamma\ \mathsf{ctx} \\ \hdashline \underline{\underline{\Gamma}} \equiv \underline{\Gamma} \end{array}$$

Precomposition with the counit:

$$\textsc{mark}\ \begin{array}{c} \Gamma \vdash a : A \\ \hdashline \underline{\Gamma} \vdash \underline{a} : \underline{A} \end{array} \qquad \begin{array}{c} \underline{\Gamma} \vdash a : A \\ \hdashline \underline{\Gamma} \vdash \underline{a} \equiv a : A \end{array} \qquad \begin{array}{c} \Gamma \vdash a : A \\ \hdashline \underline{\Gamma} \vdash \underline{\underline{a}} \equiv \underline{a} : \underline{A} \end{array}$$

$$\begin{array}{c} \Gamma \vdash \Delta\ \mathsf{tele} \\ \hdashline \underline{\Gamma} \vdash \Delta^{\mathsf{m}\Gamma}\ \mathsf{tele} \end{array} \qquad \begin{array}{c} (\cdot)^{\mathsf{m}\Gamma} :\equiv \cdot \\ (\Delta, x : A)^{\mathsf{m}\Gamma} :\equiv \Delta^{\mathsf{m}\Gamma}, x : A^{\mathsf{m}\Gamma} \\ (\Delta, \underline{x} : A)^{\mathsf{m}\Gamma} :\equiv \Delta^{\mathsf{m}\Gamma}, \underline{x} : A \end{array} \qquad \textsc{mark}\ \begin{array}{c} \Gamma, \Delta \vdash a : A \\ \hdashline \underline{\Gamma}, \Delta^{\mathsf{m}\Gamma} \vdash a^{\mathsf{m}\Gamma} : A^{\mathsf{m}\Gamma} \end{array}$$

Precomposition with the unit/roundtrip:

$$\textsc{markwk}\ \begin{array}{c} \Psi, \underline{\Gamma}, \Delta \vdash a : A \\ \hdashline \Psi, \Gamma, \Delta \vdash a : A \end{array} \qquad \textsc{roundtrip}\ \begin{array}{c} \Gamma \vdash a : A \\ \hdashline \Gamma \vdash \underline{a} : \underline{A} \end{array} \qquad \textsc{roundtrip}\ \begin{array}{c} \Gamma, \Delta \vdash a : A \\ \hdashline \Gamma, \Delta^{\mathsf{m}\Gamma} \vdash a^{\mathsf{m}\Gamma} : A^{\mathsf{m}\Gamma} \end{array}$$

Figure 1.1: New Context Structure.

Semantically, this is using the counit $\natural A \to A$, because $\underline{x} : A$ in the context is semantically $\natural A$, but A as a type on the right is semantically just A. We intentionally use the same raw syntax for these two distinct typing rules VAR-MARKED and VAR-ROUNDTRIP, one of which uses a variable that is marked in the context (via the counit), and the other uses a variable that is unmarked in the context (via the roundtrip). This allows precomposition with the unit $\Gamma \to \underline{\Gamma}$ to be a "silent" operation that leaves the raw syntax unchanged.

Finally, we have admissible structural rules corresponding to precomposing a judgement with the unit $\Gamma \to \underline{\Gamma}$ and counit $\underline{\Gamma} \to \Gamma$. We now discuss in more detail the admissible rules in play.

'Natural' on Contexts Marking a context $\underline{\Gamma}$ is defined inductively by marking all of the variables in a context. Semantically, these equations build into the theory (roughly — there are some subtleties with dependency that we discuss below) that $\natural 1 = 1$, $\natural(\Gamma.A) = \natural\Gamma.\natural A$ (which are the equations of a strict CwF morphism [Dyb96, Definition 2]) and that $\natural(\Gamma.\natural A) = \natural\Gamma.\natural A$ (which is reasonable because we intend for $\natural$ to be idempotent). The equation states that the $\underline{\Gamma}$ operation on contexts is idempotent—syntactically, $\underline{\Gamma}$ has only marked variable declarations which $\underline{\underline{\Gamma}}$ leaves unchanged (and the $\underline{A}$ operation on terms discussed next is also idempotent).

Putting together CTX-EXT-MARKED and VAR-MARKED and the definition of $\underline{\Gamma}$, the types of later marked variables can mention earlier unmarked ones, but can only use them marked. For example, in a context $x : A, \underline{y} : B$, the type B is in context $\underline{x : A} \equiv \underline{x} : \underline{A}$, so may refer to $\underline{x}$ (but not x).

Marking: Precomposition with the Counit. As mentioned above, we will often be interested in types and terms where *every* use of a free variable is marked. We call such types and terms *dull*. A dull term in context Γ is equivalently a term in the context $\underline{\Gamma}$—because all the variables in $\underline{\Gamma}$ are marked, any term $\underline{\Gamma} \vdash a : A$ necessarily uses these variables marked. We can turn any $\Gamma \vdash a : A$ into a dull term $\underline{\Gamma} \vdash \underline{a} : \underline{A}$ by marking all the free variable uses in $\underline{a}$ with an underscore (MARK). (Recall that we overload the notation and write the $\underline{a}$ operation on general terms using the same syntax as for marked/roundtripped variables.) Semantically, $\underline{a}$ is precomposing a with the counit $\underline{\Gamma} \to \Gamma$. Because the type A also depends on Γ, substitution by the counit will also mark all its free variables, giving $\underline{A}$. We think of types as elements of a universe, so $\Gamma \vdash A : \mathcal{U}$ implies $\underline{\Gamma} \vdash \underline{A} : \mathcal{U}$ is another instance of this rule. This means that when a term is marked i.e. only varies over the underlying space of the context, its type will also only vary over the underlying space of the context.

We delay a formal definition of $\underline{a}$ on raw syntax until we have introduced the further extensions of this type theory in later sections. The operation is given by recursion over the term syntax (like substitution), turning each variable usage x into $\underline{x}$, leaving marked variable uses unchanged, and proceeding recursively otherwise (e.g. $\underline{f(a)} = \underline{f}(\underline{a})$). In particular, marking commutes with the type former for Id-types: $\underline{(x = y)} \equiv (\underline{x} = \underline{y})$, which is part of what makes our modality left-exact.

One subtlety is that, because $\underline{a}$ is semantically substitution/precomposition with the counit $\underline{\Gamma} \to \Gamma$, it marks the *free* variables of a term, but leaves the bound variables the same. For example, $\underline{\lambda x.fx} \equiv \lambda x.\underline{f}x$. This leads to the full form of the operation in MARK, which marks the variables in Γ in the context and changes all occurrences of the Γ-variables in the term into marked ones, but does not change the occurrences of Δ-variables in the term. Formally, Δ is a telescope (context in context), but we omit the rules for $\Gamma \vdash \Delta$ tele, with formation rules analogous to those for contexts, and the operation of concatenating a context and a telescope Γ, Δ. For example, with f in Γ and x in

Δ we would calculate $\underline{f\,x} = \underline{f}\,x$. However, when the variables declared in Γ occur in the types in Δ, those occurrences must be marked, which we notate with $\Delta^{\mathsf{m}\Gamma}$. The telescope $\Delta^{\mathsf{m}\Gamma}$ is defined by sending $x : A$ to $x : A^{\mathsf{m}\Gamma}$ (*not* $\underline{x} : A^{\mathsf{m}\Gamma}$, differing from $\underline{\Gamma}$) and $\underline{x} : A$ to $\underline{x} : A$ (since all variables are already marked) for each variable in Δ. Officially, we should be annotating the underscores like $\underline{a}_\Gamma$ to indicate which variables in a are to be marked, but when we use this operation informally we always start with Γ being all free variables and Δ empty, so we adopt a convention that $\underline{a}$ means to mark all free variables of a.

The equations for precomposition with the counit state that if a term starts out in $\underline{\Gamma}$, then marking has no effect and $\underline{a} \equiv a$. Syntactically, this is because a term in context $\underline{\Gamma}$ cannot have any unmarked free variable uses, which are the only parts of a term changed by the marking operation. Note that this equation needs $\underline{\underline{\Gamma}} \equiv \underline{\Gamma}$ to typecheck, and semantically corresponds to the counit $\underline{\underline{\Gamma}} \to \underline{\Gamma}$ being the identity. Consequently, marking is idempotent: $\underline{\underline{a}} \equiv \underline{a}$.

Mark-weakening: Precomposition with the Unit. There is an analogous operation of precomposition with the unit $\Gamma \to \underline{\Gamma}$. Following [GSB19], we make this a "silent" operation, i.e. it does not change the raw syntax of the term or the type, only the typing derivation, as stated in MARKWK. We refer to as use of the unit as "mark-weakening" a piece of the context, because variables that are marked in the context $\underline{\Gamma}$ in the premise become unmarked in the conclusion. The unit does *not* unmark the *uses* of the variables in a term or type — a use $\underline{x}$ of a marked variable $\underline{x} : A$ from $\underline{\Gamma}$ (typed by VAR-MARKED, which is the counit $\natural A \to A$) becomes a use $\underline{x}$ of $x : A$ from Γ (typed by VAR-ROUNDTRIP, which is the roundtrip $A \to \natural A \to A$, the counit precomposed with the unit). Thus, the unit can be silent because we use the same syntax for the counit on marked variables as for the roundtrip on unmarked variables.

To work up to the rule in the figure, the most basic form, where Ψ and Δ are empty, says that any $\underline{\Gamma} \vdash a : A$ is also $\Gamma \vdash a : A$. For the same reasons as for the counit, we will need a tail telescope Δ that is not "mark-weakened" by the operation (i.e. the marks in Δ in the premise are still there in the conclusion), for inductively pushing this operation under bound variables, which are not mark-weakened (and indeed, might not even be marked in the premise). We will also sometimes find it useful to mark-weaken a variable in the middle of the context, without mark-weakening its prefix, e.g. going from $\Gamma, \underline{x} : A \vdash \mathcal{J}$ to $\Gamma, x : A \vdash \mathcal{J}$. Semantically, this is precomposition with the unit $A \to \natural A$ paired with the identity substitution on Γ. It is an implicit requirement for the judgement in the conclusion to be well-formed that Ψ, Γ, Δ is a well-formed context

Precomposition with the Roundtrip. Composing MARKWK and MARK, we have a rule ROUNDTRIP representing precomposition with the non-trivial roundtrip $\Gamma \to \natural\Gamma \to \Gamma$. (We have not seen a use for a counit rule with a prefix Ψ as in the unit rule, so we do not include one, and consequently restrict the roundtrip to the setting where both the unit and counit exist, when Ψ is empty for the unit.) The section-retraction property of a bireflection states that composing MARKWK and MARK in the other direction, i.e. going from $\underline{\Gamma} \vdash a : A$ to $\Gamma \vdash a : A$ to $\underline{\Gamma} \vdash \underline{a} : \underline{A}$ should be the identity; because the unit is silent, this is the same as the counit equation $\underline{a} \equiv a$.

Returning to the rule VAR-ROUNDTRIP, the type $\underline{A}$ in the conclusion is typed by ROUNDTRIP, because, as for the counit, precomposing/substituting by the roundtrip on Γ substitutes into the type A as well. Because variable uses $x : A$ and $\underline{x} : \underline{A}$ (in general) have different types, the marked-

$$\natural\text{-FORM}\ \frac{\underline{\Gamma} \vdash A : \mathcal{U}}{\Gamma \vdash \natural A : \mathcal{U}} \qquad \natural\text{-INTRO}\ \frac{\underline{\Gamma} \vdash a : A}{\Gamma \vdash a^{\natural} : \natural A} \qquad \natural\text{-ELIM}\ \frac{\Gamma \vdash b : \natural A}{\Gamma \vdash b_{\natural} : A}$$

$$\natural\text{-BETA}\ \frac{\underline{\Gamma} \vdash a : A}{\Gamma \vdash {a^{\natural}}_{\natural} \equiv a : A} \qquad \natural\text{-ETA}\ \frac{\Gamma \vdash b : \natural A}{\Gamma \vdash b \equiv \underline{b_{\natural}}^{\natural} : \natural A}$$

Figure 1.2: Rules for $\natural$

ness of a variable usage cannot be naïvely flipped at will in a term. For example, if $x : A$ then $\underline{x} = x$ may not be well-formed, as $\underline{A}$ is not in general the same type as A. The only reason that we do not need to analogously mark the type A in the conclusion of VAR-MARKED as $\underline{A}$ is that the marked context extension CTX-MARKED 'pre-marks' the type — the type A is in context $\underline{\Gamma}$, so must already use only marked variables.

Well-formedness of the Conclusions. Whenever we can form a term $\Gamma \vdash a : A$, we of course want that Γ ctx and $\Gamma \vdash A : \mathcal{U}$. (Depending on precisely how the type theory is set up, these are sometimes presuppositions of the term judgement; later when checking that MARK and MARKWK are admissible in Section 3.2, we follow [Str91] in having them be consequences of the term judgement.) There are few spots in the above rules where it is a little subtle why these invariants are maintained. First, in VAR-MARKED, we have by CTX-EXT-MARKED that $\underline{\Gamma} \vdash A : \mathcal{U}$, but for the use of A on the right, we need $\Gamma, x : A, \Gamma' \vdash A : \mathcal{U}$. In addition to the usual weakening with $x : A, \Gamma'$, this uses MARKWK. In VAR-ROUNDTRIP, we have $\Gamma \vdash A : \mathcal{U}$, so by another application of ROUNDTRIP, we also have $\Gamma \vdash \underline{A} : \mathcal{U}$, so all that is needed is the usual weakening. In the definition of $\underline{\Gamma}$ for unmarked variables, we begin with $\Gamma \vdash A : \mathcal{U}$, and need $\underline{\Gamma} \vdash \underline{A} : \mathcal{U}$, which we have by MARK (with Δ empty). In the definition for marked variables, we start with $\underline{\Gamma} \vdash A : \mathcal{U}$, and need $\underline{\underline{\Gamma}} \vdash A : \mathcal{U}$ to apply CTX-EXT-MARKED, which holds by idempotence. In the equation $\underline{a} \equiv a$, we need the same equation on types to see that $\underline{A} \equiv A$, and similarly for the $\underline{\underline{a}} \equiv \underline{a}$ equation.

Substitution for Marked Variables. There is a new case of standard substitution for substituting into VAR-ROUNDTRIP, which is defined by

$$\underline{x}[a/x] :\equiv \underline{a}$$

That is, when we substitute a term $a : A$ for a marked variable usage $\underline{x}$, the result is the marking of a. This type checks for $\Gamma, x : A \vdash \underline{x} : \underline{A}$ and $\Gamma \vdash a : A$ because the ROUNDTRIP rule gives $\Gamma \vdash \underline{a} : \underline{A}$. Semantically, $\underline{x}$ is the roundtrip $A \to \natural A \to A$, and the substitution post-composes this roundtrip with a; but $\underline{a}$ is a pre-composed with the roundtrip $\Gamma \to \natural\Gamma \to \Gamma$, and these are equal by naturality of the unit and counit.

1.1.2 Type Former

Using this judgement structure, it is now simple to describe the $\natural$ type using the rules in Figure 1.2.

Recall that we refer to a term/type in context $\underline{\Gamma}$, i.e. a term/type all of whose free variables are marked, as *dull*. The formation rule says that for any dull type A there is a type $\natural A$. Formally, this formation rule is analogous to $\sharp$ in spatial type theory or dependent right adjoints [BCMEPS20], in that it asks for a type under the left adjoint of $\natural A$, which in this case is also $\natural$, represented by $\underline{\Gamma}$. One alternate rule that one could imagine is like $\flat$ in spatial type theory, $\underline{\Gamma} \vdash A : \mathcal{U}$ implies $\underline{\Gamma} \vdash \natural A : \mathcal{U}$. However, this rule breaks admissibility of precomposition with the unit, because it forces variables in the conclusion's context to be marked.

The introduction rule says that for any dull term of a dull type $a : A$ there is a term $a^\natural : \natural A$, again transposing $\natural$ on the right to $\natural$ on the left (roughly, $\natural\Gamma \to A$ implies $\Gamma \to \natural A$). This is the same as the introduction rule for $\sharp$ and dependent right adjoints. Note that the type A must be assumed to be dull for the type $\natural A$ in the conclusion to be well-formed.

The elimination rule says that for any (not necessarily dull) term $b : \natural A$, there is a term $b_\natural : A$. Semantically, this is the counit $\natural A \to A$ precomposed with b. Note that the type A must be assumed to be dull for $\natural A$ in the premise to be well-formed — for a non-dull type A, we have a counit $\natural\underline{A} \to \underline{A}$, but in general we do not have a map $\natural\underline{A} \to A$.

The computation or β-reduction rule says that $a^\natural{}_\natural \equiv a$. Whenever the left-hand side is well-typed, the right-hand side is too, because of the silent unit rule MARKWK. Note that a is necessarily dull for the $\natural$-INTRO rule to have been applied, and all of its free variables are still marked on the right.

The uniqueness or η-rule says that $b \equiv \underline{b_\natural}^\natural$ for any term $b : \natural\underline{A}$. Since b is not necessarily dull, it must be marked (precomposed with the counit) before being used in the introduction rule $-^\natural$. One must be cautious in applying this rule from right to left, as not every possible 'unmarking' of a term $\underline{b}$ will be well-typed.

For a non-dull type $\Gamma \vdash A : \mathcal{U}$, note that $\underline{A}$ (given by applying MARK) and $\natural\underline{A}$ are very different. In parameterised spectra, $\underline{A}$ is A with its dependency on the fibres of the context Γ replaced by the sections of Γ. On the other hand, $\natural\underline{A}$ also replaces the fibres of A itself with the trivial spectrum. For example, if A is a closed type then $\underline{A} \equiv A \not\simeq \natural A$. From this point of view, our notation $\underline{\Gamma}$ for marking a context is confusing, because it semantically is $\natural\Gamma$; however, we use this notation to emphasise that it is implemented by "underlining all of the variables in Γ".

We have not proved canonicity or normalisation for the $\natural$ type, as our intended applications rely on many axioms, but we conjecture they are true: the equations for $\underline{\Gamma}$ and $\underline{a}$ are proved rather than asserted, and the $\natural$ type has a β rule for weak head reduction and a type-directed η rule.

Remark 1.1.1. When working informally, we will often assume a dull/marked variable $\underline{x} : A$, which has the same meaning as assuming a variable $x : \natural A$ and then working with $x_\natural : A$. Using a dull variable by writing $\underline{x}$ is a bit terser than writing $x_\natural$, and substitution into $\underline{x}$ with some term $\underline{a}$ does not need to go through the β-reduction $\underline{a}^\natural{}_\natural \equiv \underline{a}$.

Remark 1.1.2. Because $\natural$ is adjoint to itself, there was a choice here about whether to axiomatise it as left adjoint or right adjoint to the judgemental marking operation. We choose the right adjoint, as this permits us to easily have a judgemental uniqueness rule as well as a judgemental computation rule. It is not difficult to show that the rules for $\natural$ as a left adjoint are derivable; we do so in Proposition 1.1.23.

1.1.3 Properties of the Modality

First, we now develop the basic structure of the $\natural$ type internally, proving that $\natural$ behaves like both the $\flat$ and $\sharp$ modalities of spatial type theory [Shu18].

Definition 1.1.3 (Unit and counit for $\natural$)**.** The introduction and elimination rules immediately give, for any type A, unit and counit maps

$$\eta_A :\equiv (\lambda x.\underline{x}^\natural) : A \to \natural\underline{A}$$
$$\varepsilon_A :\equiv (\lambda n.n_\natural) : \natural\underline{A} \to \underline{A}$$

The fact that we only have a counit for *dull* types is what defeats the 'no-go theorem' for comonadic modalities [Shu18, Theorem 4.1]. In general, there is no way to go from a term of $\natural\underline{A}$ or $\underline{A}$ to a term of the non-marked type A.

Proposition 1.1.4. *The counit and unit are a section-retraction pair, i.e. the roundtrip $\eta_A \circ \varepsilon_A : \natural\underline{A} \to \underline{A} \to \natural\underline{A}$ is the identity. The composite $\varepsilon_A \circ \eta_A : A \to \natural\underline{A} \to \underline{A}$ is equal to $\lambda x.\underline{x}$.*

Proof. For $n : \natural\underline{A}$ we have

$$\eta(\varepsilon(n)) \equiv \underline{n_\natural}^\natural \equiv \underline{n}_\natural{}^\natural \equiv n$$

by the definition of and the η-law. For the composite on $x : \underline{A}$, we get

$$\varepsilon(\eta(x)) \equiv \underline{x}^\natural{}_\natural \equiv \underline{x}$$

by the β-law. □

Definition 1.1.5. We can define the functorial action of $\natural$ on a map, in any ambient context: given $f : A \to B$ we define $\natural\underline{f} : \natural\underline{A} \to \natural\underline{B}$ by:

$$\natural f(x) :\equiv [\underline{f}(\underline{x}_\natural)]^\natural$$

yielding a map $(A \to B) \to (\natural\underline{A} \to \natural\underline{B})$. When f is $\lambda y.y$ we get

$$\natural(\lambda y.y)(x) \equiv [\underline{(\lambda y.y)}(\underline{x}_\natural)]^\natural \equiv [(\lambda y.y)(\underline{x}_\natural)]^\natural \equiv (\underline{x}_\natural)^\natural \equiv x$$

When f is $f_2 \circ f_1$ we first have

$$\natural(f_2 \circ f_1)(x) \equiv [\underline{(f_2 \circ f_1)}(\underline{x}_\natural)]^\natural \equiv [(\underline{f_2} \circ \underline{f_1})(\underline{x}_\natural)]^\natural \equiv [\underline{f_2}(\underline{f_1}(\underline{x}_\natural))]^\natural$$

But we also have

$$(\natural(f_2) \circ \natural(f_1))(x) \equiv \natural(f_2)([\underline{f_1}(\underline{x}_\natural)]^\natural) \equiv [\underline{f_2}(\underline{([\underline{f_1}(\underline{x}_\natural)]^\natural)}_\natural)]^\natural \equiv [\underline{f_2}(([\underline{f_1}(\underline{x}_\natural)]^\natural)_\natural)]^\natural \equiv [\underline{f_2}(\underline{f_1}(\underline{x}_\natural))]^\natural$$

So $\natural f$ preserves identity and composition definitionally.

Remark 1.1.6. Note that, in contrast with $\flat$ of spatial type theory, we do not need the function f to be 'crisp', i.e., only use modal variables. Here, we can turn any function $f : A \to B$ into a 'crisp' one $\underline{f} : \underline{A} \to \underline{B}$ by marking, allowing us to apply $\natural$-INTRO to $\underline{f}(\underline{x}_\natural) : \underline{B}$.

The η-rule for $\natural\underline{A}$ implies that any term of natural type is equal to the marked version of it:

Proposition 1.1.7. *For any dull type $\underline{\Gamma} \vdash A : \mathcal{U}$ and not necessarily dull term $\Gamma \vdash a : \natural A$ there is a definitional equality $\Gamma \vdash a \equiv \underline{a} : \natural A$*

Proof. Suppose a term $\Gamma \vdash a : \natural A$. By the η-rule, we have $a \equiv \underline{a_\natural}^\natural$. But applying the admissible rule ROUNDTRIP, we have $\Gamma \vdash \underline{a} : \natural A$ (using the fact that $\underline{A} \equiv A$ because A is dull). Applying the η-rule to that gives $\underline{a} \equiv \underline{\underline{a}_\natural}^\natural$. But $\underline{\underline{n}} \equiv \underline{n}$.

Semantically, this is because, for any $f : \Gamma \to \natural A$, the composite with the roundtrip $\Gamma \to \natural\Gamma \to \Gamma \to \natural A$ is still equal to f — first, use naturality of the unit/counit to see this is equal to $\Gamma \to \natural A \to \natural\natural A \to \natural A$ and then the latter two maps are inverse by idempotence of $\natural$. ☐

Proposition 1.1.8. *The unit and counit are natural, so for any $f : A \to B$ the diagrams*

$$\begin{tikzcd} A \arrow[r, "f"] \arrow[d, "\eta_A"'] & B \arrow[d, "\eta_B"] \\ \natural\underline{A} \arrow[r, "\natural f"'] & \natural\underline{B} \end{tikzcd} \qquad \begin{tikzcd} \natural\underline{A} \arrow[r, "\natural f"] \arrow[d, "\varepsilon_A"'] & \natural\underline{B} \arrow[d, "\varepsilon_B"] \\ \underline{A} \arrow[r, "\underline{f}"'] & \underline{B} \end{tikzcd}$$

commute.

Proof. On the left:

$$\underline{f}((\underline{a}^\natural)_\natural)^\natural \equiv \underline{f}(\underline{a})^\natural \equiv \underline{f(a)}^\natural$$

On the right:

$$(\underline{f}(\underline{x}_\natural)^\natural)_\natural \equiv \underline{f}(\underline{x}_\natural) \equiv \underline{f}(x_\natural)$$

☐

We now consider types that are equivalent to their 'underlying space'.

Definition 1.1.9. A type A is *modal* if the unit $\eta_A : A \to \natural\underline{A}$ is an equivalence. We define

$$\mathsf{Space} :\equiv \textstyle\sum_{(X:\mathcal{U})} \mathsf{isEquiv}(\lambda x.\underline{x}^\natural)$$

for the type of modal types.

This definition is studied in detail in [HoTTBook, Definition 7.7.5],[RSS20, §1]. In the intended model, the modal types are the spaces, embedded in PSpec as a space equipped with the constant zero family of spectra. We will sometimes need to restrict statements to such spaces, so it is important that we can carve out a subuniverse of spaces using the modality.

Proposition 1.1.10. *If A is modal then $\underline{A}$ is also modal.*

Proof. Given a witness $w : \mathsf{isModal}(A)$, we have $\underline{w} : \mathsf{isModal}(\underline{A})$, showing $\underline{A} \simeq \natural\underline{A}$. ☐

Proposition 1.1.11. *For any A, the type $\natural\underline{A}$ is modal.*

Proof. The counit $(\lambda z.z_\natural) : \natural\natural\underline{A} \to \natural\underline{A}$ is an inverse to the unit $(\lambda v.\underline{v}^\natural) : \natural\underline{A} \to \natural\natural\underline{A}$. Using Proposition 1.1.7, for the roundtrip on $\natural\underline{A}$ we have

$$\underline{v}^{\natural}{}_{\natural} \equiv \underline{v} \equiv v$$

and for the roundtrip on $\natural\natural\underline{A}$ we have

$$\underline{z_\natural}^\natural \equiv \underline{z}_\natural{}^\natural \equiv z.$$

□

Note that the unit and counit are *not* in general an equivalence between $\natural\underline{A}$ and $\underline{A}$ — the use of the η-law to prove $\underline{v} \equiv v$ does not apply in $\underline{A}$. Intuitively, $\natural\underline{A}$ is the base space of $\underline{A}$, while $\underline{A}$ itself only zeroes out the dependence of A on the ambient context. However, we do have:

Proposition 1.1.12. *A type A is modal iff $(\lambda x.\underline{x}) : A \to \underline{A}$ is an equivalence.*

Proof. Suppose $\lambda x.\underline{x} : A \to \underline{A}$ is an equivalence, with inverse $g : \underline{A} \to A$. We show that η_A is a quasi-equivalence, which can be improved to an equivalence. The inverse is $g \circ \varepsilon_A : \natural\underline{A} \to A$. For $x : A$, we have

$$g[(\underline{x}^\natural)_\natural] \equiv g(\underline{x}) = x$$

using the inverse law for $g \circ (\lambda x.\underline{x})$.

For the other composite, let $(\lambda x.\underline{x}, g, w) : A \simeq \underline{A}$, so that w is the witness that $(\lambda x.\underline{x})$ and g are inverses. Now observe that $\underline{(\lambda x.\underline{x}, g, w)} : \underline{A} \simeq \underline{A}$ and by definition of $\underline{\ }$, the maps are $\lambda x.\underline{x}$ and $\underline{g}$, so we also have $\underline{g} \circ (\lambda x.\underline{x}) = \mathsf{id}_{\underline{A}}$. Then, for $y : \natural\underline{A}$, the composite is

$$\underline{g(y_\natural)}^\natural \equiv [\underline{g}(\underline{y}_\natural)]^\natural \equiv ((\lambda x.\underline{x}^\natural) \circ \underline{g} \circ (\lambda x : \underline{A}.\underline{x}))(y_\natural) = (\lambda x.\underline{x}^\natural)(y_\natural) \equiv (\underline{y}_\natural)^\natural \equiv y$$

Conversely, if A is modal then $\underline{A}$ is also modal by Proposition 1.1.10. By the β-law, the map $(\lambda x.\underline{x}) : A \to \underline{A}$ is equal to the composite $\varepsilon_A \circ \eta_A : A \to \natural\underline{A} \to \underline{A}$, which is the composite of two equivalences. First, η_A is an equivalence because A is modal. Second, ε_A is an equivalence, because it is left-inverse to $\eta_{\underline{A}}$, which is an equivalence because $\underline{A}$ is modal, and a left-inverse of a map that is an equivalence is its inverse. □

Corollary 1.1.13. *A dull type $\underline{A}$ is modal iff $x = \underline{x}$ for any $x : \underline{A}$.*

Proof. By function extensionality, $(\lambda x.x) = (\lambda x.\underline{x})$, and transporting the fact that the identity function is an equivalence along this allows us to use Proposition 1.1.12. □

In the remainder of this section we show that $\natural$ has all the properties of both $\flat$ and $\sharp$ from spatial type theory.

1.1.4 Monadic Properties

We begin with the monadic properties; those shared with $\sharp$.

Proposition 1.1.14 (Monadic $\natural$-induction, cf. [Shu18, Theorem 3.4])**.** *Suppose $P : \natural\underline{A} \to \mathcal{U}$ is a type family such that each $P(v)$ is modal. Given a dependent function $f : \prod_{x:A} P(\underline{x}^\natural)$, there is $g : \prod_{v:\natural\underline{A}} P(v)$ such that $g(\underline{x}^\natural) = f(x)$ for all $x : A$.*

Proof. Because each $P(v)$ is modal, we have inverses $r_v : \natural\underline{P}(\underline{v}) \to P(v)$. So it is enough to produce a function $g' : \prod_{v:\natural\underline{A}} \natural\underline{P}(\underline{v})$. Marking f gives a function $\underline{f} : \prod_{x:\underline{A}} \underline{P}(\underline{x}^\natural)$, which we can use to define

$$g'(v) :\equiv \underline{f}(\underline{v_\natural})^\natural$$

and $g'(v)$ has type $\natural\underline{P}(\underline{v_\natural}^\natural) \equiv \natural\underline{P}(\underline{v})$ as required. To get the goal function $g : \prod_{v:\natural\underline{A}} P(v)$ we then post-compose with r_v:

$$g(v) :\equiv r_v(\underline{f}(\underline{v_\natural})^\natural)$$

This has the correct computation property:

$$g(\underline{x}^\natural) \equiv r_v(\underline{f}(\underline{x}^\natural{}_\natural)^\natural) \equiv r_v(\underline{f}(\underline{x})^\natural) \equiv r_v(\underline{f(x)}^\natural) = f(x)$$

as r_v is an inverse of $(\lambda x.\underline{x}^\natural)$. □

Theorem 1.1.15 (Monadic universal property, cf. [Shu18, Theorem 3.6])**.** *Suppose $B : \natural\underline{A} \to \mathcal{U}$ is a type family with each $B(v)$ modal. Then precomposition with $\eta_A : A \to \natural\underline{A}$ is an equivalence*

$$\prod_{(v:\natural\underline{A})} B(v) \simeq \prod_{(x:A)} B(\underline{x}^\natural)$$

Proof. The inverse is given by monadic $\natural$-induction (Proposition 1.1.14), and the roundtrip on $f : \prod_{(x:A)} B(\underline{x}^\natural)$ is exactly the $g(\underline{x}^\natural) = f(x)$ equation given above.

For the other composite, suppose $h : \prod_{(v:\natural\underline{A})} B(v)$, and $y : \natural\underline{A}$, and we need to show that $h(y)$ is equal to the $g(y)$, for the g determined by monadic $\natural$-induction on h-precomposed-with-η_A, $\lambda x.h(\underline{x}^\natural)$. However, by the η-law, $y \equiv (\underline{y_\natural})^\natural$, so

$$g(y) \equiv g((\underline{y_\natural})^\natural) = h((\underline{y_\natural})^\natural) = h(y)$$

(This proof is morally doing another monadic $\natural$-induction to reduce y to something of the form $\underline{x}^\natural$, but we have not yet proved that $b =_{B(y)} b'$ is modal when $B(y)$ is modal, so we cannot use Proposition 1.1.14 directly, but instead η-expand explicitly.)

□

Corollary 1.1.16. *$(\lambda A.\natural\underline{A}) : \mathcal{U} \to \mathcal{U}$ with unit $\lambda x.\underline{x}^\natural : A \to \natural\underline{A}$ is a monadic modality in the sense of [RSS20].*

Proof. The precomposition equivalence

$$\prod_{(v:\natural\underline{A})} \natural\underline{B}(\underline{v}) \to \prod_{(x:A)} \natural\underline{B}(\underline{x}^\natural)$$

of the previous proposition, where $\natural\underline{B}$ is modal by Proposition 1.1.11, is precisely the definition of a 'uniquely eliminating modality' [RSS20, Definition 1.2], one of the several equivalent definitions of a monadic modality. □

Being a monadic modality has many formal consequences. In particular:

Lemma 1.1.17 (Properties of a modality, [RSS20])**.** *The following all hold for* $\natural$*:*

1. *A is modal iff $(\lambda x.\underline{x}^{\natural}) : A \to \natural \underline{A}$ admits a retraction.*
2. *If the input types are modal then all the following are modal:*

$$1 \qquad A \times B \qquad x =_A y \qquad \mathsf{fib}_f(x) \qquad B \times_A C \qquad A \simeq B$$

3. *If A is any type and $P : A \to \mathcal{U}$ is such that every $P(x)$ is modal, then $\prod_{(x:A)} P(x)$ is modal. If additionally A is modal, then $\sum_{(x:A)} P(x)$ is modal.*
4. *For any types A and B, the canonical map $\natural(\underline{A} \times \underline{B}) \to \natural \underline{A} \times \natural \underline{B}$ is an equivalence.*
5. *If A is a proposition, then so is $\natural \underline{A}$.*

Also like $\sharp$, the $\natural$ modality preserves Σ-types and is left-exact.

Proposition 1.1.18 ($\natural$ preserves Σ)**.** *For types $A : \mathcal{U}$ and $B : A \to \mathcal{U}$, we have*

$$\natural\left(\textstyle\sum_{(x:\underline{A})} \underline{B}(x)\right) \simeq \textstyle\sum_{(u:\natural \underline{A})} \natural \underline{B}(\underline{u}_{\natural})$$

Proof. We know that the right-hand side is modal, so to define a map from left-to-right it is sufficient to provide $f : \sum_{(x:A)} B(x) \to \sum_{(u:\natural \underline{A})} \natural \underline{B}(\underline{u}_{\natural})$ and then apply $\natural$-induction. For this we have $f(a,b) :\equiv (\underline{a}^{\natural}, \underline{b}^{\natural})$.

The other way, we are provided $u : \natural \underline{A}$ and $v : \natural \underline{B}(\underline{u}_{\natural})$, with which we can produce

$$(\underline{u}_{\natural}, \underline{v}_{\natural})^{\natural} : \natural\left(\textstyle\sum_{(x:\underline{A})} \underline{B}(x)\right).$$

To show the roundtrip on the left is the identity, suppose $v : \natural\left(\sum_{(x:\underline{A})} \underline{B}(x)\right)$. By $\natural$-induction ($=_{\natural}$ is modal by Lemma 1.1.17) we assume this is of the form $\underline{(a,b)}^{\natural}$, and then the round trip is just

$$(\underline{a}^{\natural}{}_{\natural}, \underline{b}^{\natural}{}_{\natural})^{\natural} \equiv (\underline{a}, \underline{b})^{\natural} \equiv \underline{(a,b)}^{\natural}$$

by the β-rule for $\natural$.

For the other roundtrip, starting with $(u,v) : \sum_{(u:\natural \underline{A})} \natural \underline{B}(\underline{u}_{\natural})$, the computation rule for $\natural$-induction gives that the roundtrip is equal to

$$f(\underline{u}_{\natural}, \underline{v}_{\natural}) \equiv (\underline{u_{\natural}}^{\natural}, \underline{v_{\natural}}^{\natural}) \equiv (u,v)$$

$\square$

Proposition 1.1.19 ($\natural$ is left-exact, cf. [Shu18, Theorem 3.7])**.** *For $x, y : A$, there is an equivalence $(\underline{x}^\natural = \underline{y}^\natural) \simeq \natural(\underline{x} = \underline{y})$ such that*

$$\begin{tikzcd} & (\underline{x}^\natural = \underline{y}^\natural) \arrow[dd] \\ (x = y) \arrow[ur, "\mathsf{ap}_{(\underline{-})^\natural}"] \arrow[dr, "(\underline{-})^\natural"'] & \\ & \natural(\underline{x} = \underline{y}) \end{tikzcd}$$

commutes

Proof. We can define maps both ways by:

$$c \mapsto \mathsf{ap}_{(\underline{-})^\natural}(c_\natural) : \natural(\underline{x} = \underline{y}) \to (\underline{x}^\natural = \underline{y}^\natural)$$
$$p \mapsto (\mathsf{ap}_{(-)_\natural}(\underline{p}))^\natural : (\underline{x}^\natural = \underline{y}^\natural) \to \natural(\underline{x} = \underline{y})$$

For showing the roundtrip on $c : \natural(\underline{x} = \underline{y})$ is the identity, first, for any type A with $x, y : A$ and $p : x =_A y$, we have $\mathsf{ap}_{(\lambda x.\underline{x})}(\underline{p}) = \underline{p}$ as paths in $\underline{x} =_{\underline{A}} \underline{y}$ (which type checks because $\underline{\underline{x}} = \underline{x}$). By path induction, it suffices to show

$$\mathsf{ap}_{(\lambda x.\underline{x})}(\underline{\mathsf{refl}_x}) \equiv \mathsf{ap}_{(\lambda x.\underline{x})}(\mathsf{refl}_{\underline{x}}) = \mathsf{refl}_{\underline{x}} \equiv \underline{\mathsf{refl}_x}$$

Then we have

$$[\mathsf{ap}_{(-)_\natural}(\underline{\mathsf{ap}_{(\underline{-})^\natural}(c_\natural)})]^\natural \equiv [\mathsf{ap}_{(-)_\natural}(\mathsf{ap}_{(\underline{-})^\natural}(\underline{c_\natural}))]^\natural = [\mathsf{ap}_{((\underline{-})^\natural)_\natural}(\underline{c_\natural})]^\natural \equiv [\mathsf{ap}_{\underline{-}}(\underline{c_\natural})]^\natural = [\underline{\underline{c_\natural}}]^\natural \equiv [\underline{c_\natural}]^\natural \equiv c$$

The other direction is easier: the roundtrip on p is:

$$\begin{aligned} & \mathsf{ap}_{(\underline{-})^\natural}(\mathsf{ap}_{(-)_\natural}(\underline{p})^\natural{}_\natural) \\ & \equiv \mathsf{ap}_{(\underline{-})^\natural}(\mathsf{ap}_{(-)_\natural}(\underline{p})) \\ & = \mathsf{ap}_{(\underline{-})_\natural{}^\natural}(\underline{p}) \\ & \equiv \mathsf{ap}_{\mathsf{id}}(\underline{p}) \\ & = \underline{p} \\ & = p \end{aligned}$$

The last equality is by Lemma 1.1.13 and that $p : (\underline{x}^\natural = \underline{y}^\natural)$ is a term of a dull modal type.

The triangle commutes by path-induction:

$$\begin{aligned} \mathsf{ap}_{(-)_\natural}(\mathsf{ap}_{(\underline{-})^\natural}(\mathsf{refl}_{\underline{x}}))^\natural &= \mathsf{ap}_{(\underline{-})^\natural{}_\natural}(\mathsf{refl}_{\underline{x}})^\natural \\ &= (\mathsf{refl}_{\underline{x}})^\natural \\ &\equiv \underline{(\mathsf{refl}_x)}^\natural \end{aligned}$$

□

Left-exactness has some additional formal consequences outlined by [RSS20].

Proposition 1.1.20. *The following hold for $\natural$:*

1. *$\natural$ preserves pullbacks, [RSS20, Theorem 3.1].*
2. *$\natural$ preserves n-types and more generally n-truncated maps for all n, [RSS20, Corollary 3.9].*

1.1.5 Comonadic Properties

Now we turn to the comonadic properties of $\natural$; the properties it shares with the $\flat$ modality of spatial type theory. First, as remarked in Section 1.1, we can derive a substitution principle of dull terms for dull variables:

Definition 1.1.21 (Dull substitution)**.** The dull substitution principle is

$$\textsc{subst-dull}\ \frac{\Gamma, \underline{x} : A \vdash c : C \qquad \underline{\Gamma} \vdash a : A}{\Gamma \vdash c[a/\underline{x}] :\equiv c[a/x] : C[a/\underline{x}]}$$

To see that this type checks, precompose with the unit on A to get $\Gamma, x : A \vdash c : C$ and the unit on Γ to get $\Gamma \vdash a : A$, and then a normal substitution $c[a/x]$ has type $C[a/x] \equiv C[a/\underline{x}]$.

Remark 1.1.22. Definition 1.1.21 corresponds to the substitution principle for crisp variables in spatial type theory, where a crisp variable can be substituted by a term containing only crisp variables. In our setting, given a term a in a general context, we can mark it and then substitute it for a dull variable:

$$\frac{\Gamma, \underline{x} : A \vdash c : C \qquad \Gamma \vdash a : A}{\Gamma \vdash c[\underline{a}/\underline{x}] \equiv c[a/x] : C[\underline{a}/\underline{x}]}$$

Because $\underline{\Gamma} \vdash \underline{a} : A$ is a dull term, we can use SUBST-DULL. This is equal to the ordinary substitution $c[a/x]$ because all of the uses of x in c and C must be marked, so a will be marked during substitution; we can prove inductively that

$$\frac{\Gamma, \underline{x} : A \vdash c : C \qquad \Gamma \vdash a : A}{\Gamma \vdash c[\underline{a}/\underline{x}] \equiv c[a/x] : C[\underline{a}/\underline{x}]}$$

Proposition 1.1.23 (Comonadic $\natural$-induction)**.** *A $\flat$-style eliminator is derivable for $\natural$:*

$$\text{“}\flat\text{”-}\textsc{elim}\ \frac{\Gamma, x : \natural\underline{A} \vdash C : \mathcal{U} \qquad \Gamma \vdash v : \natural\underline{A} \qquad \Gamma, \underline{u} : \underline{A} \vdash c : C[\underline{u}^{\natural}/x]}{\Gamma \vdash (\mathsf{let}\ \underline{u}^{\natural} = v\ \mathsf{in}\ c) : C[v/x]}$$

$$\text{“}\flat\text{”-}\textsc{beta}\ \frac{\Gamma, x : \natural\underline{A} \vdash C : \mathcal{U} \qquad \underline{\Gamma} \vdash v : \underline{A} \qquad \Gamma, \underline{u} : \underline{A} \vdash c : C[\underline{u}^{\natural}/x]}{\Gamma \vdash (\mathsf{let}\ \underline{u}^{\natural} = v^{\natural}\ \mathsf{in}\ c) \equiv c[v/\underline{u}] : C[v^{\natural}/x]}$$

Proof. The eliminator can be derived by dull substitution:

$$(\mathsf{let}\ \underline{u}^{\natural} = v\ \mathsf{in}\ c) :\equiv c[\underline{v}_{\natural}/\underline{u}]$$

which has type $C[\underline{v}_{\natural}{}^{\natural}/x] \equiv C[v/x]$ as required. The β-rule follows immediately from the β-rule for $\natural$, using the fact that $\underline{v} \equiv v$ for $\underline{\Gamma} \vdash v : A$, as all variable uses in v are already marked. □

This induction principle can be rephrased as a characterisation of maps out of $\natural A$.

Theorem 1.1.24 (Comonadic universal property, cf. [Shu18, Theorem 6.16])**.** *For any $A, B : \mathcal{U}$, post-composition (and functoriality of $\natural$) with $(-)_{\natural} : \natural\underline{B} \to \underline{B}$ induces an equivalence*

$$\natural(\natural\underline{A} \to \natural\underline{B}) \simeq \natural(\natural\underline{A} \to \underline{B})$$

or more dependently, for $A : \mathcal{U}$ and $B : \natural\underline{A} \to \mathcal{U}$, fibrewise post-composition (and functoriality of $\natural$) with $(-)_{\natural} : \natural\underline{B}(\underline{v}) \to \underline{B}(\underline{v})$ yields an equivalence

$$\natural\left(\textstyle\prod_{(v:\natural\underline{A})} \natural\underline{B}(\underline{v})\right) \simeq \natural\left(\textstyle\prod_{(v:\natural\underline{A})} \underline{B}(\underline{v})\right)$$

Proof. The counit map $\natural\underline{B}(\underline{v}) \to \underline{B}(\underline{v})$ is always a section of the unit map, so the post-composition map in the statement of the Theorem is also a section.

We just have to check that the roundtrip on $\natural\left(\prod_{(v:\natural\underline{A})} \underline{B}(\underline{v})\right)$ is the identity. Suppose we have an $f : \natural\left(\prod_{(v:\natural\underline{A})} \underline{B}(\underline{v})\right)$. Unfolding the definition of post-composition and functoriality of $\natural$ with both the unit and the counit is:

$$\begin{aligned}(\lambda x.(\underline{f}_{\natural}(\underline{x}))^{\natural}{}_{\natural})^{\natural} &\equiv (\lambda x.(\underline{f}_{\natural}(\underline{x})))^{\natural} \\ &\equiv (\lambda x.(\underline{f}_{\natural}(x)))^{\natural} \\ &\equiv (\underline{f}_{\natural})^{\natural} \\ &\equiv f\end{aligned}$$

where $\underline{x} \equiv x : \natural\underline{A}$ by Proposition 1.1.7. Note that $(\lambda x.(\underline{f}_{\natural}(x)))^{\natural}$ is well-typed, as x is not free below the $\natural$-INTRO, so does not need to be marked. □

Because the rules for $\flat$ are derivable, we could instead have repeated the proof for $\flat$ verbatim, but the above is more direct.

Remark 1.1.25. One may wonder why the applications of $\natural$ are required around the two sides. Thinking syntactically, they are necessary to block access to 'spectral' information from the context. Without $\natural$ on the right, one could use a $b : \underline{B}$ in the context to form constant functions $\mathsf{const}_b : \natural\underline{A} \to \underline{B}$ which have no corresponding maps $\natural\underline{A} \to \natural\underline{B}$ on the left.

Another way to see that they are necessary is to consider the model in families of pointed types, looking ahead to Section 3.3. There, it is clear that the map is not an equivalence without the $\natural$ present: $\mathsf{E}(\natural\underline{A} \to \natural\underline{B})$ is equivalent to the point in every fibre, but $\mathsf{E}(\natural\underline{A} \to \underline{B})$ may be non-trivial.

Corollary 1.1.26 (Dull Self-adjointness)**.** *For any $A, B : \mathcal{U}$ there is an equivalence*

$$\natural(\underline{A} \to \natural\underline{B}) \simeq \natural(\natural\underline{A} \to \underline{B})$$

Proof. Combining the monadic and comonadic universal properties we get

$$\natural(\underline{A} \to \natural\underline{B}) \simeq \natural(\natural\underline{A} \to \natural\underline{B}) \simeq \natural(\natural\underline{A} \to \underline{B})$$

□

Proposition 1.1.27 (cf. [Shu18, Theorem 3.11])**.** *In the presence of univalence, the type* Space *of modal types is modal.*

Proof. Recall that Space $:\equiv \sum_{(A:\mathcal{U})} \mathsf{isModal}(A)$. We can use a simpler proof than the one used for $\sharp$. By Corollary 1.1.13, we just have to show that for any $A : \mathcal{U}$ and $w : \mathsf{isModal}(A)$, there is an equality $(A, w) = (\underline{A}, \underline{w})$. And there is: by assumption A is modal so $A \simeq \underline{A}$, again by Corollary 1.1.13. In the second component, $\mathsf{isModal}(A)$ is a proposition, so we are done.

□

Remark 1.1.28. When working with inductive types in a theory with more structured contexts such as ours, one has to make sure that the induction principles are strong enough. In spatial type theory, one needs to assert or prove 'crisp induction principles', for when the motive of an elimination rule depends on a *crisp* variable of type being eliminated. The crisp induction principle for a type constructor is a judgemental way of saying that the modality preserves the type constructor. For example, crisp coproduct case analysis is a judgemental way of saying that $\flat(A + B) \simeq \flat A + \flat B$, because the crisp induction principle gives crisp variables of type $A + B$ the same universal property as an ordinary variable of type $\flat A + \flat B$. In our setting, a dull induction principle for coproducts looks like:

$$\textsc{dull-+-elim}\ \dfrac{\begin{array}{c}\Gamma, \underline{z} : \underline{A} + \underline{B} \vdash C : \mathcal{U} \\ \Gamma, \underline{x} : \underline{A} \vdash p : C[\mathsf{inl}(\underline{x})/\underline{z}] \qquad \Gamma, \underline{y} : \underline{B} \vdash q : C[\mathsf{inr}(\underline{y})/\underline{z}] \\ \underline{\Gamma} \vdash s : \underline{A} + \underline{B}\end{array}}{\Gamma \vdash \mathsf{dullcase}(\underline{z}.C, \underline{x}.p, \underline{y}.q, s) : C[s/\underline{z}]}$$

In spatial type theory, crisp induction principles are proven using the adjointness of $\flat$ and $\sharp$, and because $\natural$ is self-adjoint, we could repeat the proof almost verbatim. But we can show they are valid more directly, using the MARK and MARKWK rules.

With the above inputs, we can apply MARKWK to obtain

$$\begin{array}{c}\Gamma, z : \underline{A} + \underline{B} \vdash C : \mathcal{U} \\ \Gamma, x : \underline{A} \vdash p : C[\mathsf{inl}(\underline{x})/z] \\ \Gamma, y : \underline{B} \vdash q : C[\mathsf{inr}(\underline{y})/z] \\ \Gamma \vdash s : \underline{A} + \underline{B}\end{array}$$

Now note that $C[\mathsf{inl}(\underline{x})/z] \equiv C[\mathsf{inl}(x)/z]$ and $C[\mathsf{inr}(\underline{x})/z] \equiv C[\mathsf{inr}(x)/z]$, because z is only used marked in C, as in Remark 1.1.22. These inputs are now of the right shape to apply the ordinary +-ELIM rule.

Using an analogous construction, we can derive dull induction principles for Id-types, pushouts, etc.

Proposition 1.1.29 ($\natural$ preserves pushouts)**.** *Suppose $\underline{f} : \underline{C} \to \underline{A}$ and $\underline{g} : \underline{C} \to \underline{B}$ are dull functions between dull types. Then*

$$\natural(\underline{A} +_{\underline{C}} \underline{B}) \simeq \natural\underline{A} +_{\natural\underline{C}} \natural\underline{B},$$

the pushout of $\natural\underline{f} : \natural\underline{C} \to \natural\underline{A}$ and $\natural\underline{g} : \natural\underline{C} \to \natural\underline{B}$.

Proof. Morally, this follows because we just proved that $\natural$ is a left adjoint, but we can also write out the maps explicitly as follows.

From left-to-right, we extract a term of $p : \underline{A} +_{\underline{C}} \underline{B}$, and then do case analysis. On $a : \underline{A}$, we have $\mathsf{inl}(\underline{a}^{\natural}) : \natural\underline{A} +_{\natural\underline{C}} \natural\underline{B}$. Similarly, on $b : \underline{B}$ we have $\mathsf{inr}(\underline{b}^{\natural}) : \natural\underline{A} +_{\natural\underline{C}} \natural\underline{B}$. To complete the cocone we have to provide for any $c : \underline{C}$, a path $\mathsf{inl}(\underline{f}(\underline{c})^{\natural}) = \mathsf{inr}(\underline{g}(\underline{c})^{\natural})$ in $\natural\underline{A} +_{\natural\underline{C}} \natural\underline{B}$. The glue constructor for the $\natural\underline{A} +_{\natural\underline{C}} \natural\underline{B}$ pushout gives us a path

$$\mathsf{glue}(\underline{c}^{\natural}) : \mathsf{inl}(\natural\underline{f}(\underline{c}^{\natural})) = \mathsf{inr}(\natural\underline{g}(\underline{c}^{\natural}))$$

And this type is equal to $\mathsf{inl}(\underline{f}(\underline{c})^{\natural}) = \mathsf{inr}(\underline{g}(\underline{c})^{\natural})$ by the definition of the functorial action of $\natural$, and the β-rule.

The right-to-left direction is similar. We begin with case analysis on $\underline{z}$. On $n : \natural\underline{A}$ and $m : \natural\underline{B}$ we have $\mathsf{inl}(\underline{n}_{\natural}) : \underline{A} +_{\underline{C}} \underline{B}$ and $\mathsf{inr}(\underline{m}_{\natural}) : \underline{A} +_{\underline{C}} \underline{B}$ respectively. For any $o : \natural\underline{C}$, we need a path $\mathsf{inl}(\natural\underline{f}(\underline{o})_{\natural}) = \mathsf{inr}(\natural\underline{g}(\underline{o})_{\natural})$. Expanding the definition of functoriality and applying the β-rule, this is a path

$$\mathsf{inl}(\underline{f}(\underline{o}_{\natural})) = \mathsf{inr}(\underline{g}(\underline{o}_{\natural}))$$

and $\mathsf{glue}(\underline{o}_{\natural})$ is such a path in $\underline{A} +_{\underline{C}} \underline{B}$. All together this produces a dull term of $\underline{A} +_{\underline{C}} \underline{B}$, so applying $\natural$-INTRO we are done.

Checking the roundtrips are the identity is straightforward, using comonadic $\natural$-induction and dull pushout induction, which can be derived as in Remark 1.1.28.

□

The sequential colimit (see e.g. [SDR20]) of a sequence

$$A(0) \xrightarrow{a(0)} A(1) \xrightarrow{a(1)} A(2) \xrightarrow{a(2)} \dots$$

is given by the higher inductive type colim A_n with point and path constructors

$$\iota : \textstyle\prod_{(n:\mathbb{N})} A(n) \to \mathrm{colim}_n\, A_n$$
$$\kappa : \textstyle\prod_{(n:\mathbb{N})} \prod_{(x:A(n))} \iota(n+1, a(n,x)) = \iota(n,x)$$

Proposition 1.1.30 ($\natural$ preserves sequential colimits)**.** *Suppose we have a diagram*

$$\underline{A}(0) \xrightarrow{\underline{a}(0)} \underline{A}(1) \xrightarrow{\underline{a}(1)} \underline{A}(2) \xrightarrow{\underline{a}(2)} \dots$$

of dull types and dull functions between them. Then

$$\natural(\mathrm{colim}_n\, \underline{A}(n)) \simeq \mathrm{colim}_n\, \natural\underline{A}(n)$$

where the sequential colimit on the right is over the diagram

$$\natural\underline{A}(0) \xrightarrow{\natural\underline{a}(0)} \natural A(1) \xrightarrow{\natural\underline{a}(1)} \natural\underline{A}(2) \xrightarrow{\natural\underline{a}(2)} \ldots$$

Proof. The proof is analogous to that for pushouts. □

1.1.6 Discussion

Remark 1.1.31. In spatial type theory [Shu18], there is also a special context extension $x :: A$, which is a judgemental version of extending the context with $\flat A$. Such variables are called *crisp*. A usage of a crisp variable x in a term corresponds to a use of the counit $\flat A \to A$, like our VAR-MARKED rule. Because there is only a counit and not also a unit, the type A of a crisp variable $x :: A$ is only permitted to depend on other crisp variables. A loose way to think about this is as follows. Before we have access to any structural rules, dependency forces us to apply modalities to an entire context at once. Given a type-in-context $\Gamma \vdash A : \mathcal{U}$ presented as a fibration $p : A \to \Gamma$, ordinary context extension corresponds to considering the object A as a context. If we want to make A discrete, we have to apply $\flat$ to everything, giving $\flat p : \flat A \to \flat\Gamma$. So $\flat A$ can only depend on a discrete context, and the judgemental version $x : A$ has the same restriction. Therefore, all crisp variables must occur before regular ones, and the context naturally divides into two zones. In our system, however, the presence of the unit map $A \to \natural A$ means that we can no longer neatly divide the context in this way. For example, if we have a context $\underline{x} : A, \underline{y} : B$, then we can precompose with the unit substitution just on $\underline{x}$, giving $x : A, \underline{y} : B$. This breaks the invariant that crisp variables all occur before ordinary ones.

Remark 1.1.32. A substitution principle for marked variables that is typical from other comonadic type theories following [PD01] is

$$\frac{\Gamma, \underline{x} : A, \Gamma' \vdash b : B \qquad \underline{\Gamma} \vdash a : A}{\Gamma, \Gamma'[a/\underline{x}] \vdash b[a/\underline{x}] : B[a/\underline{x}]}$$

Here, the term being substituted must already have all of its variables marked, as indicated by the premise $\underline{\Gamma} \vdash a : A$ of the rule, and this substitution principle is implemented by a syntactic substitution, replacing $\underline{x}$ with a everywhere. Given the admissible rules in Figure 1.1, we can in fact define this by first mark-weakening the variable $\underline{x}$ to get $\Gamma, x : A, \Gamma' \vdash b : B$ and then doing an ordinary substitution $b[a/x]$. Since all uses of x will be marked $\underline{x}$ in b and B, this will replace $\underline{x}$ with $\underline{a}$ everywhere—but since $\underline{\Gamma} \vdash a : A$, we have $\underline{a} \equiv a$, so we get the same result as the more specialised principle would have given.

We prefer this style of presenting substitution, where the substitution for marked variables is given by unmarking and then ordinary substitution, because it corresponds more closely to what we will do when working informally in this type theory. When performing substitutions $b[a/x]$ by hand, we can simply look in b for each instance of x and $\underline{x}$ and replace them with either a or

$\underline{a}$ accordingly, without having to mentally keep track of the context through each subterm to see whether x is marked or not (variables that are not marked become marked in the premises of some rules), as we would have to do if substitution for a marked variable required pre-marking the term. This is another benefit of having VAR-MARKED and VAR-ROUNDTRIP rules be identical raw syntax. In our experience trying different systems for this setting, this choice seems critical for the usability of the system for informal type theory.

1.2 Palettes and Contexts

Supporting $\otimes$- and $\multimap$-types will require a more radical change to the way contexts are structured. In this section we describe this new structure. The rules of the type theory described in the previous section will turn out all be derivable in this larger theory, and so we refer to the theory of the previous section as the '$\natural$-fragment'.

We extend our theory in a manner similar to the way the $\alpha\lambda$-calculus extends the ordinary λ-calculus, as part of work on bunched logics [OP99; OHe03]. Bunched logics capture semantic situations where there are two independent symmetric monoidal products on a category. One product is typically assumed to be the ordinary cartesian product, but little is assumed about the other monoidal product besides symmetry. The $\alpha\lambda$-calculus [OHe03; Pym02] is a term calculus for bunched logic, with cartesian product $\times$, monoidal product $\otimes$ and corresponding function types $\to$ and $\multimap$.

Contexts in ordinary type theory can be written as flat lists because any finite product of objects can be reassociated in a canonical way. Once there are two monoidal products, neither of which distributes over the other, combinations of objects using these operations cannot in general be flattened to a list. We must therefore consider contexts with a tree structure.

We present a fragment of the $\alpha\lambda$-calculus in Figure 1.3. There are two binary context formers: the comma denotes combining two contexts in the ordinary cartesian way, and the $\otimes$ combining in a linear way[1]. Weakening and contraction of the ordinary cartesian comma are *explicit* rules used in the derivation of a term, and contexts are considered up to an equivalence relation containing symmetry, associativity and unitality for both products. The judgement $\Gamma\{\Delta\}$ used in WK and CONTR indicate that the context Δ appears *somewhere* as a subtree of the context Γ. The explicit structural rules are required, because the conclusions of the 'splitting' rules $\to$-ELIM and $\multimap$-ELIM are not fully general: instead, the conclusions are exactly the (judgemental) cartesian product or monoidal product of the contexts in the premises. If the user wishes to use function application of either kind while building a derivation for a term, they may have to invoke the explicit structural rules in complicated ways to massage the context into the correct form.

These context manipulations are *not* recorded in the terms, so typechecking a term can be difficult. First, it is not always obvious when and how the contraction rule has to be applied, and secondly, it is not always obvious how the context has to be split into two pieces to apply $\multimap$-ELIM. Besides the challenge of typechecking, the combination of the weakening rule and the rules for the monoidal unit also breaks soundness for the intended semantics (we discuss this in Remark 1.4.1).

[1]We have replaced some symbols of the $\alpha\lambda$-calculus to be more consistent with MLTT and the system are about to present. The type formers $\times$, $\otimes$, $\to$ and $\multimap$ are typically called $\wedge$, $*$, $\to$ and $-\!\!*$ respectively. For forming contexts, typically ; denotes cartesian combination and , linear combination.

$$\frac{A \text{ type}}{x : A \text{ ctx}} \qquad \frac{\Gamma \text{ ctx} \qquad \Delta \text{ ctx}}{\Gamma, \Delta \text{ ctx}} \qquad \frac{\Gamma \text{ ctx} \qquad \Delta \text{ ctx}}{\Gamma \otimes \Delta \text{ ctx}}$$

$$\textsc{var}\ \frac{}{x : A \vdash x : A}$$

$$\textsc{wk}\ \frac{\Gamma\{\Delta\} \vdash a : A}{\Gamma\{\Delta, \Delta'\} \vdash a : A} \qquad \textsc{contr}\ \frac{\Gamma\{\Delta, \Delta'\} \vdash a : A \qquad \Delta \cong \Delta'}{\Gamma\{\Delta\} \vdash a[\Delta/\Delta'] : A[\Delta/\Delta']}$$

$$\to\text{-}\textsc{intro}\ \frac{\Gamma, x : A \vdash b : B}{\Gamma \vdash \lambda x.b : A \to B} \qquad \to\text{-}\textsc{elim}\ \frac{\Gamma \vdash f : A \to B \qquad \Delta \vdash a : A}{\Gamma, \Delta \vdash f(a) : B}$$

$$\multimap\text{-}\textsc{intro}\ \frac{\Gamma \otimes (x : A) \vdash b : B}{\Gamma \vdash \partial x.b : A \multimap B} \qquad \multimap\text{-}\textsc{elim}\ \frac{\Gamma \vdash f : A \multimap B \qquad \Delta \vdash a : A}{\Gamma \otimes \Delta \vdash f\langle a \rangle : B}$$

Figure 1.3: Selected Rules of the $\alpha\lambda$-calculus.

The various extensions of the $\alpha\lambda$-calculus that have since appeared [BO06; CPR08; SS04; Atk04] do not resolve these issues. It is also not so clear how to add dependent types to this system and how dependency should behave between bunches.

We have some desiderata for an extension of MLTT with such bunched contexts, if we are aiming for a type theory that is usable by hand:

- Weakening and symmetry of the context should be silent on raw terms,
- Contraction (of the cartesian context former) should be admissible, including contraction of entire pieces of context and not just pairs of variables,
- Substitution should traverse the term 'to the leaves', in particular there should be no stuck explicit substitutions. It should also be easily computable by hand, and as far as possible be defined by syntactic substitution of raw terms.
- Determining which variables are permitted to be used should be easy at any position in a term.
- Any label bound in a term should be named, so nameless techniques like de Bruijn indices should be avoided.
- Typechecking should be syntax-directed, so no proof search or similar is necessary to type-check terms.

These properties are all taken for granted when working in ordinary MLTT, but only some of these are maintained in the $\alpha\lambda$-calculus. A new strategy is necessary!

In the $\alpha\lambda$-calculus, the shape of a context and the types appearing in the context are described simultaneously. The new idea is to track the shape of the context separately from the types using a book-keeping tool that we call a *palette*.

Linearity requirements are tracked by labelling each variable with a 'colour', and the palette describes how these colours relate to each other. Context splits that are required in rules like $\multimap$-ELIM can be specified just by referring to the palette, not the variables. This separation between the context and the bunched structure has a couple of advantages. The primary benefit is that weakening, contraction, and the various monoidal symmetry/associativity rules can be made admissible. Like ordinary type theory, weakening and symmetry are invisible operations on the level of raw terms. Having individual labels to refer to large pieces of the context also makes the syntax more compact.

Before introducing the new judgemental structure and types formally, we give a few examples of how our theory feels to use informally:

Proposition 1.2.1. *Given two closed types A and B, there is a function $\mathsf{sym}_{A,B} : A \otimes B \to B \otimes A$.*

(When we revisit this map in Proposition 1.3.5 we will relax the requirement that A and B are closed types.)

Proof. Just as in ordinary type theory, we introduce an assumption $p : A \otimes B$ by λ-abstraction and now have to prove $B \otimes A$. To break apart the assumption $p : A \otimes B$ into its two pieces, we use $\otimes$-*induction*. This produces two new assumptions $x^{\mathfrak{r}} : A$ and $y^{\mathfrak{b}} : B$, where x and y have been labelled with two new colours $\mathfrak{r}$ and $\mathfrak{b}$. The colours $\mathfrak{r}$ and $\mathfrak{b}$ give names to the sides of a tensor product $\mathfrak{r} \otimes \mathfrak{b}$. The fact that x and y are labelled by $\mathfrak{r}$ and $\mathfrak{b}$ is placing them on the corresponding sides of this tensor.

These new assumptions are not immediately available to be used in a term: right now we cannot conclude A from x or B from y. What 'unlocks' these new assumptions is a use of $\otimes$-*introduction*. To form a term of $B \otimes A$, we must linearly split the palette into two disjoint pieces, one of which we use to form an element of B and the other to form an element of A. Now assigning $\mathfrak{b}$ to the B side we have $y : B$, and assigning $\mathfrak{r}$ to the A side we have $x : A$, so we can form $y \,{}_{\mathfrak{b}}{\otimes}_{\mathfrak{r}}\, x : B \otimes A$.

In all, we have described the term

$$\mathsf{sym}_{A,B} :\equiv \lambda p.(\mathsf{let}\ x \,{}_{\mathfrak{r}}{\otimes}_{\mathfrak{b}}\, y = p\ \mathsf{in}\ y \,{}_{\mathfrak{b}}{\otimes}_{\mathfrak{r}}\, x) : A \otimes B \to B \otimes A$$

□

In Proposition 1.3.5 we will show that this is an equivalence. Note that the arrow $\to$ used here is the ordinary function type, and so the introduced p is an ordinary variable. The linear type formers create ordinary types that can be used just like any other; there is no separation between linear and ordinary types.

This $\otimes$ is a positive type former, and like other positive type formers in dependent type theory, a propositional uniqueness principle is provable from the computation rule.

Proposition 1.2.2. *Suppose A and B are closed types, $\mathsf{C} : A \otimes B \to \mathcal{U}$ is a type family and $f : \prod_{(p:A\otimes B)} \mathsf{C}(p)$. For any $p : A \otimes B$ we have*

$$(\mathsf{let}\ x \,{}_{\mathfrak{r}}{\otimes}_{\mathfrak{b}}\, y = p\ \mathsf{in}\ f(x \,{}_{\mathfrak{r}}{\otimes}_{\mathfrak{b}}\, y)) = f(p)$$

Proof. Let $P : A \otimes B \to \mathcal{U}$ denote the type family

$$P(p) :\equiv (\mathsf{let}\ x\ {}_{\mathfrak{r}}\!\otimes_{\mathfrak{b}}\ y = p\ \mathsf{in}\ f(x\ {}_{\mathfrak{r}}\!\otimes_{\mathfrak{b}}\ y)) = f(p)$$

We wish to find an element $\prod_{(p:A\otimes B)} P(p)$, so by $\otimes$-induction it is enough to assume $p \equiv x'\ {}_{\mathfrak{r}'}\!\otimes_{\mathfrak{b}'}\ y'$ where $\mathfrak{r}' \otimes \mathfrak{b}'$ is a new split of our top colour. Here we choose *different* colour labels to the ones we mention in the statement of the Proposition: those labels are bound in the definition of P.

Our new goal is

$$(\mathsf{let}\ x\ {}_{\mathfrak{r}}\!\otimes_{\mathfrak{b}}\ y = x'\ {}_{\mathfrak{r}'}\!\otimes_{\mathfrak{b}'}\ y'\ \mathsf{in}\ f(x\ {}_{\mathfrak{r}}\!\otimes_{\mathfrak{b}}\ y)) = f(x'\ {}_{\mathfrak{r}'}\!\otimes_{\mathfrak{b}'}\ y')$$

In the body of the let, "$f(x\ {}_{\mathfrak{r}}\!\otimes_{\mathfrak{b}}\ y)$", there are *two* different tensors of colours in scope: the $\mathfrak{r}' \otimes \mathfrak{b}'$ from the outer induction, and the $\mathfrak{r} \otimes \mathfrak{b}$ from the inner induction. These are completely independent, and in particular are not combined with each other using a further $\otimes$.

The *computation rule* for $\otimes$-induction lets us reduce the left endpoint of the path by syntactically substituting the variables x' and y' for x and y as normal, and also substituting the labels $\mathfrak{r}'$ and $\mathfrak{b}'$ for $\mathfrak{r}$ and $\mathfrak{b}$. This gives the goal

$$f(x'\ {}_{\mathfrak{r}'}\!\otimes_{\mathfrak{b}'}\ y') = f(x'\ {}_{\mathfrak{r}'}\!\otimes_{\mathfrak{b}'}\ y')$$

for which we have $\mathsf{refl}_{f(x'\ {}_{\mathfrak{r}'}\otimes_{\mathfrak{b}'}\ y')}$. □

There is no need for 'linear identity types': ordinary identity types are sufficient.

As a final example, we show that the syntax builds in a way to use the nonlinear information of any variable. Recall that $\natural A$ is supposed to represent the nonlinear aspect of A.

Proposition 1.2.3. *Given two closed types A and B, there is a function* $\mathsf{base}_{A,B} : A \otimes B \to \natural A \times \natural B$*.*

Later a similar map appears when we show that $\natural$ is monoidal in that it sends $\otimes$ to $\times$ (Proposition 1.3.21).

Proof. As in the definition of the symmetry map, we start by introducing $p : A \otimes B$ and applying $\otimes$-induction to give $x^{\mathfrak{r}} : A$ and $y^{\mathfrak{b}} : B$. Now, to form a term of A, we use the assumption x in a special way, that we call a *marked* usage, written $\underline{x} : A$. We can then use $\natural$*-introduction* to form $\underline{x}^{\natural} : \natural A$. Doing the for y lets us form

$$\mathsf{base}_{A,B} :\equiv \lambda p.(\mathsf{let}\ x\ {}_{\mathfrak{r}}\!\otimes_{\mathfrak{b}}\ y = p\ \mathsf{in}\ (\underline{x}^{\natural}, \underline{y}^{\natural})) : A \otimes B \to \natural A \times \natural B$$

□

1.2.1 Rules for Palettes

We now describe the syntactic gadget used to keep track of these colour labels. On its own, a palette Φ describes a particular kind of binary tree, where each node is tagged with either a cartesian product $\times$ or a tensor product $\otimes$. Palettes have no semantic interpretation on their own; they describe a shape that is later filled by the variables of the context. (In particular, although they will be written as a context 'zone', they are not really a piece of context in the sense of spatial type theory, or the special interval variables in cubical type theory.)

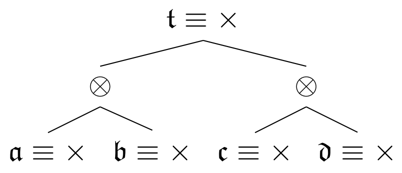
In a palette, children of a $\times$-node are separated by a comma and children of a $\otimes$-node by the symbol $\otimes$.

Any $\times$-node can be given a label. We require that these labels are all distinct across the entire tree. We call these label *colours*, and write $\Phi \vdash \mathfrak{c}\ \mathsf{colour}$ for the judgement picking out a single label[2]. Each variable in the context will be assigned one of these labels, which corresponds to adding that variable as a child of the corresponding $\times$ node.

A typical palette may look like as follows:

$$\mathfrak{t} \prec (\mathfrak{a} \otimes \mathfrak{b}, \mathfrak{c} \otimes \mathfrak{d})\ \mathsf{palette}$$

This corresponds to the following tree:

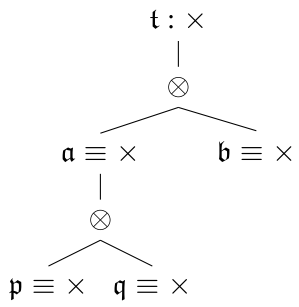

$\mathfrak{t} \equiv \times$
$\otimes$
$\otimes$
$\mathfrak{a} \equiv \times$
$\mathfrak{b} \equiv \times$
$\mathfrak{c} \equiv \times$
$\mathfrak{d} \equiv \times$

The $\prec$ symbol is supposed to invoke something splitting into pieces, and doesn't denote an order on colours.

Palettes can be nested in arbitrary ways. For example, starting with the palette $\mathfrak{t} \prec (\mathfrak{a} \otimes \mathfrak{b})\ \mathsf{palette}$, we might like for the $\mathfrak{a}$ labelled $\times$-node to itself contain a $\otimes$-node. We write this as

$$\mathfrak{t} \prec ((\mathfrak{a} \prec \mathfrak{p} \otimes \mathfrak{q}) \otimes \mathfrak{b})\ \mathsf{palette},$$

which corresponds to the tree


$\mathfrak{t} : \times$
$\otimes$
$\mathfrak{a} \equiv \times$
$\mathfrak{b} \equiv \times$
$\otimes$
$\mathfrak{p} \equiv \times$
$\mathfrak{q} \equiv \times$

The $\mathfrak{a}$ labelled $\times$-node could have a second $\otimes$-node for a child $\mathfrak{r} \otimes \mathfrak{s}$. This is written using a comma, as it was at the top level:

$$\mathfrak{t} \prec ((\mathfrak{a} \prec \mathfrak{p} \otimes \mathfrak{q}, \mathfrak{r} \otimes \mathfrak{s}) \otimes \mathfrak{b})\ \mathsf{palette}$$

giving the tree

[2]A reference for the more peculiar Fraktur letters: $w \equiv \mathfrak{w}$, $x \equiv \mathfrak{x}$, $y \equiv \mathfrak{y}$, $z \equiv \mathfrak{z}$.

$$\textsc{pal-empty}\ \frac{}{1\ \mathsf{palette}} \qquad \textsc{pal-}\times\ \frac{\Phi_1\ \mathsf{palette} \qquad \Phi_2\ \mathsf{palette}}{\Phi_1, \Phi_2\ \mathsf{palette}}$$

$$\textsc{pal-unit}\ \frac{}{\varnothing\ \mathsf{palette}} \qquad \textsc{pal-}\otimes\ \frac{\Phi_1\ \mathsf{palette} \qquad \Phi_2\ \mathsf{palette}}{\Phi_1 \otimes \Phi_2\ \mathsf{palette}} \qquad \textsc{pal-col}\ \frac{\Phi\ \mathsf{palette}}{\mathfrak{c} \prec \Phi\ \mathsf{palette}}$$

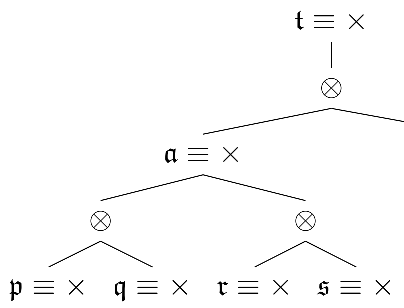
Figure 1.4: Rules for Palette Formation

$$\textsc{col-here}\ \frac{}{\mathfrak{c} \prec \Phi \vdash \mathfrak{c}\ \mathsf{colour}} \qquad \textsc{col-sub}\ \frac{\Phi \vdash \mathfrak{c}\ \mathsf{colour}}{\mathfrak{t} \prec \Phi \vdash \mathfrak{c}\ \mathsf{colour}}$$

$$\textsc{col-}\times\textsc{-left}\ \frac{\Phi_1 \vdash \mathfrak{c}\ \mathsf{colour}}{\Phi_1, \Phi_2 \vdash \mathfrak{c}\ \mathsf{colour}} \qquad \textsc{col-}\times\textsc{-right}\ \frac{\Phi_2 \vdash \mathfrak{c}\ \mathsf{colour}}{\Phi_1, \Phi_2 \vdash \mathfrak{c}\ \mathsf{colour}}$$

$$\textsc{col-}\otimes\textsc{-left}\ \frac{\Phi_1 \vdash \mathfrak{c}\ \mathsf{colour}}{\Phi_1 \otimes \Phi_2 \vdash \mathfrak{c}\ \mathsf{colour}} \qquad \textsc{col-}\otimes\textsc{-right}\ \frac{\Phi_2 \vdash \mathfrak{c}\ \mathsf{colour}}{\Phi_1 \otimes \Phi_2 \vdash \mathfrak{c}\ \mathsf{colour}}$$

Figure 1.5: Rules for Colours


$\mathfrak{t} \equiv \times$
$\otimes$
$\mathfrak{a} \equiv \times$
$\mathfrak{b} \equiv \times$
$\otimes$
$\otimes$
$\mathfrak{p} \equiv \times$
$\mathfrak{q} \equiv \times$
$\mathfrak{r} \equiv \times$
$\mathfrak{s} \equiv \times$

As well as labels and the binary palette formers, there are two special unary symbols: 1, representing the terminal object, and $\varnothing$, representing the monoidal unit. The 1 palette will be used when 'marking' a term as part of the rules for $\natural$: pieces of the palette will be replaced with 1.

The rules for constructing palettes are given in Figure 1.4. To cut down on notation, we write $\mathfrak{r}$ as a shorthand for $\mathfrak{r} \prec 1$, we have used this already in the preceding examples.

Colours. As mentioned above, a colour $\Phi \vdash \mathfrak{c}\ \mathsf{colour}$ is a single label in the palette, formally given by the rules of Figure 1.5. Ordinary variables in the type theory will each be labelled by a colour.

$$\textsc{ctx-empty}\ \frac{\Phi\ \mathsf{palette}}{\mathfrak{t} \prec \Phi \mid \cdot\ \mathsf{ctx}} \qquad \textsc{ctx-ext}\ \frac{\Phi \vdash \mathfrak{c}\ \mathsf{colour} \quad \Phi^{\mathfrak{c}} \mid \Gamma^{\mathfrak{c}} \vdash A\ \mathsf{type}}{\Phi \mid \Gamma, x^{\mathfrak{c}} : A\ \mathsf{ctx}} \qquad \textsc{ctx-ext-marked}\ \frac{\mathfrak{r} \mid \underline{\Gamma} \vdash A\ \mathsf{type}}{\mathfrak{t} \prec \Phi \mid \Gamma, \underline{x}^{\mathfrak{r}} : A\ \mathsf{ctx}}$$

$$\textsc{var}\ \frac{}{\mathfrak{t} \prec \Phi \mid \Gamma, x^{\mathfrak{t}} : A, \Gamma' \vdash x : A} \qquad \textsc{var-roundtrip}\ \frac{\mathfrak{t} \prec \Phi \vdash \mathfrak{c}\ \mathsf{colour}}{\mathfrak{t} \prec \Phi \mid \Gamma, x^{\mathfrak{c}} : A, \Gamma' \vdash \underline{x} : \underline{A}^{\mathfrak{t} \leftrightarrow \mathfrak{c}}}$$

$$\textsc{var-marked}\ \frac{}{\mathfrak{t} \prec \Phi \mid \Gamma, \underline{x}^{\mathfrak{c}} : A, \Gamma' \vdash \underline{x} : A^{\mathfrak{t} \leftrightarrow \mathfrak{c}}}$$

Figure 1.6: Rules for Context Formation and Variable Usage.

1.2.2 Rules for Contexts and Variables

Contexts in our theory have the form $\Phi \mid \Gamma$ ctx where Φ palette. We require that the palette associated with a term *always* has a label at the top level, so $\mathfrak{t} \prec \Phi \mid \Gamma \vdash a : A$. The rules for context formation and variable usage are given in Figure 1.6.

Context Extension. There are two ways to extend a context with a variable: either 'colourful' or 'marked'. These are analogous to the ordinary and marked context extensions of the $\natural$-fragment. The difference is that the variables are now also labelled with a colour which further restricts how they can be used.

- **Colourful Context Extension.** A 'colourful' context extension $\mathfrak{t} \prec \Phi \mid \Gamma, x^{\mathfrak{c}} : A$ ctx denotes an assumption whose linear data *is* accessible. The $\Phi \vdash \mathfrak{c}$ colour label describes where this linear data is positioned relative to the rest of the context: the data is placed in the $\times$-node labelled $\mathfrak{c}$.

  The superscript on $x^{\mathfrak{c}}$ should be thought of as a part of the syntax of context extension, not as a part of the variable name. The superscript does not appear on the variable x when it is later used in a term. (We might write $x :^{\mathfrak{c}} A$ if it weren't so ugly!)

  The type A assigned to x must be well-formed in the palette $(\mathfrak{t} \prec \Phi)^{\mathfrak{c}}$, which is the subpalette rooted at the label $\mathfrak{c}$, with the variables Γ marked to match: we describe this process below in the section on 'filtering'.

  As a special case of colourful context extension, we have ordinary context extension when we use the colour that is at the top of the palette, as in

  $$\mathfrak{t} \prec \Phi \mid \Gamma, x^{\mathfrak{t}} : A.$$

  Here, A is a type in the context $\mathfrak{t} \prec \Phi \mid \Gamma$ without any changes, as we will see that $(\mathfrak{t} \prec \Phi)^{\mathfrak{t}} \equiv \mathfrak{t} \prec \Phi$ and $\Gamma^{\mathfrak{t}} \equiv \Gamma$.

- **Marked Context Extension.** A 'marked' context extension $\mathfrak{t} \prec \Phi \mid \Gamma, \underline{x}^{\mathfrak{c}} : A$ ctx denotes an assumption whose linear data is not accessible. It corresponds semantically to extending

Γ with the underlying space of A, i.e. $\natural A$. The $\mathfrak{c}$ label here does not come from the palette: instead it is bound in this context extension. When a variable becomes marked, its connection to the ambient palette is completely severed.

The type A in such a context extension must be 'dull', i.e. all uses of variables from Γ in A must be marked uses, and nor can it use any colours form Φ: the palette is cleared to $\mathfrak{c}$, using the top colour label that the extension is annotated with.

As in the $\natural$-fragment of the theory, there are three variable rules.

- **Ordinary Usage.** We can use a colourful variable $x^{\mathfrak{t}} : A$ in the ordinary way via the VAR rule if the colour $\mathfrak{t}$ of the variable is precisely the label at the top of the palette.
- **Roundtrip Usage.** Any colourful variable $x^{\mathfrak{c}} : A$ can be used *marked* via VAR-ROUNDTRIP, regardless of what colour that variable is labelled with. As in the previous section, the result is a variable usage whose type has had the marking operation applied to it.

  By the form of colourful context extension, the type A is in a context that has $\mathfrak{c}$ as the top colour, so the marked type $\underline{A}$ also has $\mathfrak{c}$ as the top colour. We a further admissible rule $(-)^{t\leftrightarrow c}$ to obtain a type in the correct context $\mathfrak{t} \prec \Phi \mid \Gamma$ ctx.
- **Marked Usage.** Any marked variable $\underline{x}^{\mathfrak{c}} : A$ can be used, regardless of what colour the marked variable is labelled with.

  Here, A is already a marked type, so all that is necessary is to apply the same operation $(-)^{t\leftrightarrow c}$ to obtain a type in $\mathfrak{t} \prec \Phi \mid \Gamma$ ctx.

The raw syntax for VAR-ROUNDTRIP and VAR-MARKED continues to be identical, so that the 'mark-weakening' rule MARKWK acts invisibly on terms.

We have used various admissible rules in these context and variable rules, and we display these rules in Figure 1.7.

Marking a Context. As in the $\natural$ fragment of the theory, there is an operation on contexts $\underline{\Gamma}$ that turns every context extension with a marked context extension, also clearing the palette other than the top colour. Semantically, this is still applying $\natural$ to the context Γ.

Filtering a Context. A new ingredient in the present system is a more refined marking of the context that picks out all the variables whose colour lies in a particular part of the palette. Whenever we have a colour $\Phi \vdash \mathfrak{c}$ colour, we can *filter* the palette to that colour, extracting the 'subpalette' headed by that colour as a new palette $\Phi^{\mathfrak{c}}$ palette. For example, using the palette from earlier we have

$$\begin{aligned}
(\mathfrak{t} \prec (\mathfrak{a} \prec \mathfrak{p} \otimes \mathfrak{q}, \mathfrak{r} \otimes \mathfrak{s}) \otimes \mathfrak{b})^{\mathfrak{t}} &\equiv \mathfrak{t} \prec (\mathfrak{a} \prec \mathfrak{p} \otimes \mathfrak{q}, \mathfrak{r} \otimes \mathfrak{s}) \otimes \mathfrak{b} \\
(\mathfrak{t} \prec (\mathfrak{a} \prec \mathfrak{p} \otimes \mathfrak{q}, \mathfrak{r} \otimes \mathfrak{s}) \otimes \mathfrak{b})^{\mathfrak{a}} &\equiv \mathfrak{a} \prec \mathfrak{p} \otimes \mathfrak{q}, \mathfrak{r} \otimes \mathfrak{s} \\
(\mathfrak{t} \prec (\mathfrak{a} \prec \mathfrak{p} \otimes \mathfrak{q}, \mathfrak{r} \otimes \mathfrak{s}) \otimes \mathfrak{b})^{\mathfrak{p}} &\equiv \mathfrak{p} \\
(\mathfrak{t} \prec (\mathfrak{a} \prec \mathfrak{p} \otimes \mathfrak{q}, \mathfrak{r} \otimes \mathfrak{s}) \otimes \mathfrak{b})^{\mathfrak{b}} &\equiv \mathfrak{b}
\end{aligned}$$

Marking contexts:

$$\textsc{ctx-mark}\ \frac{\mathfrak{t} \prec \Phi \mid \Gamma\ \mathsf{ctx}}{\mathfrak{t} \mid \underline{\Gamma}\ \mathsf{ctx}} \qquad \begin{aligned} \underline{\cdot} &:\equiv \cdot \\ \underline{\Gamma, x^{\mathfrak{c}} : A} &:\equiv \underline{\Gamma}, \underline{x}^{\mathfrak{c}} : \underline{A} \\ \underline{\Gamma, \underline{x}^{\mathfrak{c}} : A} &:\equiv \underline{\Gamma}, \underline{x}^{\mathfrak{c}} : A \end{aligned} \qquad \frac{\mathfrak{t} \prec \Phi \mid \Gamma\ \mathsf{ctx}}{\underline{\underline{\Gamma}} \equiv \underline{\Gamma}}$$

Filtering contexts:

$$\textsc{pal-filter}\ \frac{\Phi \vdash \mathfrak{c}\ \mathsf{colour}}{\Phi^{\mathfrak{c}}\ \mathsf{palette}} \qquad \frac{\Phi \vdash \mathfrak{c}\ \mathsf{colour} \qquad \Phi^{\mathfrak{c}} \vdash \mathfrak{d}\ \mathsf{colour}}{(\Phi^{\mathfrak{c}})^{\mathfrak{d}} \equiv \Phi^{\mathfrak{d}}\ \mathsf{palette}}$$

$$\textsc{ctx-filter}\ \frac{\Phi \vdash \mathfrak{c}\ \mathsf{colour} \qquad \Phi \mid \Gamma\ \mathsf{ctx}}{\Phi^{\mathfrak{c}} \mid \Gamma^{\mathfrak{c}}\ \mathsf{ctx}} \qquad \begin{aligned} (\cdot)^{\mathfrak{c}} &:\equiv \cdot \\ (\Gamma, x^{\mathfrak{d}} : A)^{\mathfrak{c}} &:\equiv \Gamma^{\mathfrak{c}}, x^{\mathfrak{d}} : \underline{A} && \text{if } \mathfrak{d} \in \Phi^{\mathfrak{c}} \\ (\Gamma, x^{\mathfrak{d}} : A)^{\mathfrak{c}} &:\equiv \Gamma^{\mathfrak{c}}, \underline{x}^{\mathfrak{d}} : A && \text{if } \mathfrak{d} \notin \Phi^{\mathfrak{c}} \\ (\Gamma, \underline{x}^{\mathfrak{d}} : A)^{\mathfrak{c}} &:\equiv \Gamma^{\mathfrak{c}}, \underline{x}^{\mathfrak{d}} : A \end{aligned}$$

$$\frac{\mathfrak{t} \prec \Phi \mid \Gamma\ \mathsf{ctx} \qquad \mathfrak{t} \prec \Phi \vdash \mathfrak{c}\ \mathsf{colour}}{\underline{\Gamma^{\mathfrak{c}}} \equiv \underline{\Gamma}} \qquad \frac{\mathfrak{t} \prec \Phi \mid \Gamma\ \mathsf{ctx} \qquad \mathfrak{t} \prec \Phi \vdash \mathfrak{c}\ \mathsf{colour} \qquad (\mathfrak{t} \prec \Phi)^{\mathfrak{c}} \vdash \mathfrak{d}\ \mathsf{colour}}{(\Gamma^{\mathfrak{c}})^{\mathfrak{d}} \equiv \Gamma^{\mathfrak{d}}}$$

Marking terms:

$$\textsc{mark}\ \frac{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \mathcal{J}}{\mathfrak{t} \mid \underline{\Gamma} \vdash \underline{\mathcal{J}}}$$

Palette-weakening and mark-weakening:

$$\textsc{palwk}\ \frac{\begin{array}{c}\mathfrak{t} \mid \Gamma\ \mathsf{ctx} \\ \mathfrak{t} \mid \Gamma \vdash \mathcal{J}\end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \mathcal{J}} \qquad \textsc{markwk}\ \frac{\begin{array}{c}\mathfrak{t} \prec \Phi \mid \Gamma\ \mathsf{ctx} \\ \mathfrak{t} \mid \underline{\Gamma} \vdash \mathcal{J}\end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \mathcal{J}}$$

Colour renaming:

$$\textsc{recolour}\ \frac{\mathfrak{r} \prec \Phi \mid \Gamma \vdash \mathcal{J}}{\mathfrak{t} \prec \Phi \mid \Gamma^{\mathfrak{t} \leftrightarrow \mathfrak{r}} \vdash \mathcal{J}^{\mathfrak{t} \leftrightarrow \mathfrak{r}}}$$

Figure 1.7: Admissible Rules for Marking Contexts and Terms.

The associated operation on contexts $\Phi^{\mathfrak{c}} \mid \Gamma^{\mathfrak{c}}$ ctx, which we also call filtering, marks all variables (and their associated types) whose labels are not in $\Phi^{\mathfrak{c}}$. The variables whose labels *are* in $\Phi^{\mathfrak{c}}$ are left unmarked, and their types untouched.

For example, letting

$$\begin{aligned} \Phi &:\equiv \mathfrak{t} \prec (\mathfrak{a} \prec (\mathfrak{p} \otimes \mathfrak{q}, \mathfrak{r} \otimes \mathfrak{s})) \otimes \mathfrak{b} \\ \Gamma &:\equiv x^{\mathfrak{t}} : X, y^{\mathfrak{a}} : Y, z^{\mathfrak{p}} : Z, u^{\mathfrak{r}} : U, v^{\mathfrak{b}} : V \end{aligned}$$

we can calculate

$$\begin{aligned} \Phi^{\mathfrak{t}} \mid \Gamma^{\mathfrak{t}} &\equiv \mathfrak{t} \prec (\mathfrak{a} \prec \mathfrak{p} \otimes \mathfrak{q}, \mathfrak{r} \otimes \mathfrak{s}) \otimes \mathfrak{b} \mid x^{\mathfrak{t}} : X, y^{\mathfrak{a}} : Y, z^{\mathfrak{p}} : Z, u^{\mathfrak{r}} : U, v^{\mathfrak{b}} : V \\ \Phi^{\mathfrak{a}} \mid \Gamma^{\mathfrak{a}} &\equiv \mathfrak{a} \prec \mathfrak{p} \otimes \mathfrak{q}, \mathfrak{r} \otimes \mathfrak{s} \mid \underline{x}^{\mathfrak{t}} : \underline{X}, y^{\mathfrak{a}} : Y, z^{\mathfrak{p}} : Z, u^{\mathfrak{r}} : U, \underline{v}^{\mathfrak{b}} : \underline{V} \\ \Phi^{\mathfrak{p}} \mid \Gamma^{\mathfrak{p}} &\equiv \mathfrak{p} \mid \underline{x}^{\mathfrak{t}} : \underline{X}, \underline{y}^{\mathfrak{a}} : Y, z^{\mathfrak{p}} : Z, \underline{u}^{\mathfrak{r}} : \underline{U}, \underline{v}^{\mathfrak{b}} : \underline{V} \\ \Phi^{\mathfrak{b}} \mid \Gamma^{\mathfrak{b}} &\equiv \mathfrak{b} \mid \underline{x}^{\mathfrak{t}} : \underline{X}, \underline{y}^{\mathfrak{a}} : \underline{Y}, \underline{z}^{\mathfrak{p}} : \underline{Z}, \underline{u}^{\mathfrak{r}} : \underline{U}, v^{\mathfrak{b}} : V \end{aligned}$$

Semantically, this filtering operation should correspond to extracting the $\otimes$ factor of Γ corresponding to the colour $\mathfrak{r}$. So, if we have $\mathfrak{r} \otimes \mathfrak{b} \mid \Gamma$, then it should be the case that $\Gamma \cong \Gamma^{\mathfrak{r}} \otimes \Gamma^{\mathfrak{b}}$.

Marking a Term. As in the $\natural$-fragment, an important class of terms is those where every free variable is used via a marked variable usage, and we continue to call these 'dull' terms. In the present system this means that the palette is only used in a trivial way.

A dull term in $\mathfrak{t} \prec \Phi \mid \Gamma$ ctx is equivalently a term in the marked context $\mathfrak{t} \mid \underline{\Gamma}$ ctx. The shape of the variable rules means that any use of the variables in Γ must be via a use of VAR-MARKED, but the term also may not use any colours in the palette Φ: the palette is completely cleared other than the top colour $\mathfrak{t}$. This accords with dull terms as denoting points in the base space: such points do not depend on any linear information from the context whatsoever. Bound variables may still be used in an ordinary way: for example, the identity function $(\lambda x.x)$ is dull.

The MARK rule allows us to take any term $\mathfrak{t} \prec \Phi \mid \Gamma \vdash a : A$ and produce a dull term $\mathfrak{t} \prec \Phi \mid \Gamma \vdash \underline{a} : \underline{A}$ by replacing every variable usage in a with a marked variable usage, and replacing all colour labels from Φ mentioned in a with the special symbol 1. Semantically, this is precomposition with the counit $\natural(\Phi \mid \Gamma) \to (\Phi \mid \Gamma)$.

In the marking operation, it is not enough to pick a palette Φ and mark all variables whose colours lie in that palette. The ordinary cartesian type formers bind new variables of the current top colour, and any uses of these new bound variables should not become marked.

There is no equivalent of filtering for terms: this operation is all-or-nothing on the variable uses in a term. Semantically, there is typically no morphism $\Phi^{\mathfrak{s}} \to \Phi$ to pre-compose by.

Mark-weakening a Term. We can also take a term defined in a marked context and perform 'mark-weakening', adding resources back to the context that are not used in the term, corresponding semantically to precomposition with the unit $(\Phi \mid \Gamma) \to \natural(\Phi \mid \Gamma)$. This is done by the admissible MARKWK rule and, as in the $\natural$-fragment, this is a silent operation on raw syntax.

Well-formedness of the Conclusions. We briefly argue that the conclusions of the above rules are well-formed.

- CTX-MARK: For colourful context extensions $x^{\mathfrak{c}} : A$, the type is well-formed in context $\Phi^{\mathfrak{c}} \mid \Gamma^{\mathfrak{c}}$ ctx. The marking operation gives $\mathfrak{c} \mid \underline{\Gamma^{\mathfrak{c}}} \vdash A$ type, and

  $$\underline{\Gamma^{\mathfrak{c}}} \equiv \underline{\Gamma} \equiv \underline{\underline{\Gamma}}$$

  by combining the two equations in the figure. This makes the marked context extension $\mathfrak{t} \mid \underline{\Gamma}, \underline{x}^{\mathfrak{c}} : \underline{A}$ ctx well-formed.

  For already marked context extensions $\underline{x}^{\mathfrak{c}} : A$, the type A is well-formed in context $\mathfrak{c} \mid \underline{\Gamma}$ ctx and $\underline{\Gamma} \equiv \underline{\underline{\Gamma}}$ by the same equation, so $\mathfrak{t} \mid \underline{\Gamma}, \underline{x}^{\mathfrak{c}} : \underline{A}$ ctx is well-formed.

- CTX-FILTER: Similar to the previous. The new case is when a colourful context extension $x^{\mathfrak{d}} : A$ is in the palette $\Phi^{\mathfrak{c}}$ being filtered to, and so remains unmarked in the conclusion. This time $\Phi^{\mathfrak{d}} \mid \Gamma^{\mathfrak{d}} \vdash A$ type, but the two displayed equations give $(\Phi^{\mathfrak{c}})^{\mathfrak{d}} \equiv \Phi^{\mathfrak{d}}$ and $(\Gamma^{\mathfrak{c}})^{\mathfrak{d}} \equiv \Gamma^{\mathfrak{d}}$. These are exactly what is required for $\Phi^{\mathfrak{c}}, x^{\mathfrak{d}} : A$ ctx to be well-formed.

- VAR: The type A is well-formed in context $(\mathfrak{t} \prec \Phi)^{\mathfrak{t}} \mid \Gamma^{\mathfrak{t}}$ ctx. By the definition of filtering, $(\mathfrak{t} \prec \Phi)^{\mathfrak{t}} \equiv \mathfrak{t} \prec \Phi$ and $\Gamma^{\mathfrak{t}} \equiv \Gamma$, because every colour used in Γ is certainly in $\mathfrak{t} \prec \Phi$. The conclusion of VAR then weakens this type to include the further context extensions $x^{\mathfrak{t}} : A, \Gamma'$.

- VAR-ROUNDTRIP: The type A is well-formed in context $\Phi^{\mathfrak{c}} \mid \Gamma^{\mathfrak{c}}$ ctx, which typically is not equal to the context in the conclusion. The marking operation gives us a type

  $$\mathfrak{c} \mid \underline{\Gamma^{\mathfrak{c}}} \vdash \underline{A} \text{ type.}$$

  The equation in the figure gives that $\underline{\Gamma^{\mathfrak{c}}} \equiv \underline{\Gamma}$, and applying colour renaming,

  $$\mathfrak{t} \mid \underline{\Gamma} \vdash \underline{A}^{\mathfrak{t}\leftrightarrow\mathfrak{c}} \text{ type}$$

  Finally, MARKWK followed by ordinary weakening gives

  $$\mathfrak{t} \vdash \Phi \mid \Gamma, x^{\mathfrak{c}} : A, \Gamma' \vdash \underline{A}^{\mathfrak{t}\leftrightarrow\mathfrak{c}} \text{ type}$$

  and so the conclusion is well-formed.

- VAR-MARKED: This is similar, but we instead start with

  $$\mathfrak{c} \mid \underline{\Gamma} \vdash A \text{ type}$$

  and so no marking of A is required. Applying the same mark-weakening and variable weakening as above gives

  $$\mathfrak{t} \vdash \Phi \mid \Gamma, \underline{x}^{\mathfrak{c}} : A, \Gamma' \vdash A^{\mathfrak{t}\leftrightarrow\mathfrak{c}} \text{ type}$$

$$\Pi\text{-}\textsc{form}\ \frac{\begin{array}{c}\mathfrak{t} \prec \Phi \mid \Gamma \vdash A\ \mathsf{type} \\ \mathfrak{t} \prec \Phi \mid \Gamma, x^{\mathfrak{t}} : A \vdash B\ \mathsf{type}\end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \prod_{(x:A)} B\ \mathsf{type}}$$

$$\Pi\text{-}\textsc{intro}\ \frac{\mathfrak{t} \prec \Phi \mid \Gamma, x^{\mathfrak{t}} : A \vdash b : B}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \lambda x.b : \prod_{(x:A)} B} \qquad \Pi\text{-}\textsc{elim}\ \frac{\begin{array}{c}\mathfrak{t} \prec \Phi \mid \Gamma \vdash f : \prod_{(x:A)} B\ \mathsf{type} \\ \mathfrak{t} \prec \Phi \mid \Gamma \vdash a : A\end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash f(a) : B[a/x]}$$

$$\textsc{subst}\ \frac{\mathfrak{t} \prec \Phi \mid \Gamma \vdash a : A \qquad \mathfrak{t} \prec \Phi \mid \Gamma, x^{\mathfrak{t}} : A, \Gamma' \vdash \mathcal{J}}{\mathfrak{t} \prec \Phi \mid \Gamma, \Gamma'[a/x] \vdash \mathcal{J}[a/x]}$$

Figure 1.8: Rules for Π-types

1.2.3 Ordinary Type Formers

The type formers of MLTT have their rules imported almost unchanged. A generic context Γ is replaced with a generic palette/context pair $\mathfrak{t} \prec \Phi \mid \Gamma$, and any variables bound in the rules are bound with the top colour $\mathfrak{t}$ as their label. Figure 1.8 gives the example of Π-types.

Our theory contains all of MLTT as a fragment (in any fixed palette), and so all the results provable in ordinary HoTT will also be available to us. The only possible concern is that the admissible weakening and substitution rules used in the ordinary MLTT rules somehow behave differently here. In the fragment of the theory without any of the new type formers, single variable substitution $a/x^{\mathfrak{t}}$ is defined on raw syntax exactly as it is usually: we scan the term for each occurrence of the variable x and syntactically replace it with a. This continues to hold when the judgement being substituted into includes the new term and type formers of our theory: the structure of the rules ensures that the result is well-formed.

Once the term a itself involves the new type formers, there is a caveat involving terms that contain the top colour label. For the moment, we add the side-condition that the top colour label $\mathfrak{t}$ does not appear in the term a.

Remark 1.2.4. These ordinary cartesian type formers are the reason that we always need a name for the top colour available, as at any point in a term we might go under a cartesian variable binder. There are design trade-offs for how this top colour is handled, and this choice touches almost every aspect of the judgemental structure. We discuss these trade-offs in Section 3.1.2.

1.2.4 The Modality

The rules for $\natural$ are given in Figure 1.9, and are essentially the same as in the $\natural$-fragment. We may apply $\natural$-FORM to any 'dull' type, and similarly for $\natural$-INTRO and dull terms. Recall however that marking a term is slightly more complicated in the present theory: beyond marking all the free variables, all the free colours in a term must also be replaced with 1.

We import all the results of Section 1.1 into the present type theory.

$$\natural\text{-FORM}\ \frac{\mathfrak{t} \mid \underline{\Gamma} \vdash A \text{ type}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \natural A \text{ type}}$$

$$\natural\text{-INTRO}\ \frac{\mathfrak{t} \mid \underline{\Gamma} \vdash a : A}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash a^{\natural} : \natural A} \qquad \natural\text{-ELIM}\ \frac{\mathfrak{t} \prec \Phi \mid \Gamma \vdash a : \natural A}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash a_{\natural} : A}$$

$$\natural\text{-BETA}\ \frac{\mathfrak{t} \mid \underline{\Gamma} \vdash a : A}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash {a^{\natural}}_{\natural} \equiv a : A} \qquad \natural\text{-ETA}\ \frac{\mathfrak{t} \prec \Phi \mid \Gamma \vdash v : \natural A}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash v \equiv {\underline{v}_{\natural}}^{\natural} : \natural A}$$

Figure 1.9: Rules for the Natural Modality.

1.2.5 Working Informally

When working informally, we propose to take the palette and colour metaphor literally, assigning an actual colour to each 'colour' in the palette and painting each variable in that colour. We already demonstrated this in our initial examples for the ⊗-type. Ignoring the types of the variables for a moment, we might have a context

$$\mathfrak{p} \prec \mathfrak{r} \otimes \mathfrak{b} \mid x^{\mathfrak{r}} : A, y^{\mathfrak{b}} : B, z^{\mathfrak{p}} : C \text{ ctx}$$

Variables are also painted in their associated colour at each point they are used. It will be easy to see that the colour annotation on each variable is never changed as we move up a derivation, so different usages of the same variable will always be written in the same colour.

In the above context, only the variable z is usable, because the top colour of the palette is purple:

$$\mathfrak{p} \prec \mathfrak{r} \otimes \mathfrak{b} \mid x^{\mathfrak{r}} : A, y^{\mathfrak{b}} : B, z^{\mathfrak{p}} : C \vdash z : C$$

When using the theory, it is important to keep track of what the current top colour is to know which variables are usable. We think of a term in the above context as a 'purple term': its immediate subterms are either also purple or produced by rules that allow combining other colours to form purple.

Writing the variable uses in colour at their points of use has no syntactic meaning, because whenever we can use a variable via the ordinary VAR rule, there is only one way to do so. Using colours in this way has a couple of advantages, however. Firstly, it makes it easier to keep track of which colour each variable is labelled with, which is important when working informally as the current state of the context is not immediately visible in the middle of a term.

Secondly, it allows us to tell at a glance whether a term is well-formed. Working in the above context, if we see a 'colour clash' $(y, z) : B \times C$ as a subterm, then we know that the term is not well-typed and we have made a mistake somewhere. This becomes more interesting when typechecking an instance of ⊗-INTRO or ⊸-ELIM. We will discuss how to use these informally once the rules have been presented.

Finally, using colours is fun! It is satisfying to see the colours line up nicely when completing a proof.

Marked Variables and Uses. Using colours works nicely with marked variables and marked variable usages. Marked variables do not interact with the colours in the palette, so we write them in black in the context and at all usage points. Ordinary colourful variables are also written in black when they are *used* marked. Recall that these marked usages are permitted no matter what the current top colour is, for example, we have purple terms $(\underline{x}, z) : A \times C$ and $(\underline{y}, z) : B \times C$.

When using some rules, like the formation rule for the modality ♮-FORM, the entire context becomes marked in the premises. Working informally, this means that when we go under ♮-FORM, we lose colourful access to all the variables in the context, so any use of these variables must be marked. And so the provenance of terminology 'dull' is revealed: a dull term is one where all free variables have no colour.

The top colour of the context is unchanged when going under ♮-FORM, which is relevant when later going under a λ-binder, for example. The bound variable is still given the top colour of the context, and so is permitted to be used immediately. For example, we import the following version of Theorem 1.1.15, arbitrarily choosing the top colour to be purple.

Theorem 1.2.5 (Monadic universal property)**.** *Suppose $B : \natural\underline{A} \to \mathcal{U}$ is a type family with each $B(v)$ a space. Then precomposition with $\eta_A : A \to \natural\underline{A}$ is an equivalence*

$$\prod_{(v:\natural\underline{A})} B(v) \simeq \prod_{(x:A)} B(\underline{x}^{\natural})$$

Filtered Contexts. The informal interpretation of context filtering is that terms in a filtered context must be 'in the colour' that the context was filtered to. As we saw in the introductory examples, a use of $\otimes$-INTRO will allow us to form the term

$$\mathfrak{p} \prec \mathfrak{r} \otimes \mathfrak{b} \mid x^{\mathfrak{r}} : A, y^{\mathfrak{b}} : B, z^{\mathfrak{p}} : C \vdash x \;{}_{\mathfrak{r}}\!\otimes_{\mathfrak{b}}\; y : A \otimes B$$

The annotations on ${}_{\mathfrak{r}}\!\otimes_{\mathfrak{b}}$ specify the colour that the term on each side should be, so that the context is filtered to $\mathfrak{r}$ on the left and to $\mathfrak{b}$ on the right. Working informally, this means that the term on the left must 'be red', with all non-red variables used marked, and the term on the right must 'be blue', with all non-blue variables used marked. The result is the 'purple' term $x \;{}_{\mathfrak{r}}\!\otimes_{\mathfrak{b}}\; y$. The colour theory learned finger-painting as a toddler comes in useful: Here, the top colour is purple, which can also be produced by combining red and blue. Throughout this work we make an effort to align the colours in this way, so purple $\mathfrak{p}$ is created by combining red $\mathfrak{r}$ and blue $\mathfrak{b}$, orange $\mathfrak{o}$ is given by combining yellow $\mathfrak{y}$ and red $\mathfrak{r}$, et cetera.

When filtering the context to a colour, we are permitted to keep any 'subcolours'. Consider the more complicated palette

$$\mathfrak{w} \prec (\mathfrak{o} \prec \mathfrak{y} \otimes \mathfrak{r}) \otimes \mathfrak{b} \ \ \textsf{palette}$$

(taken from Proposition 1.5.1). Then when producing an orange $\mathfrak{o}$ term, we are still permitted to use yellow $\mathfrak{y}$ and red $\mathfrak{r}$ variables.

Reviewing the rule for colourful context extension, the type of a $\mathfrak{r}$-coloured variable in the context is required to be a type in context $\Phi^{\mathfrak{r}} \mid \Gamma^{\mathfrak{r}}$ ctx. Another way to say this is that the type of a red variable $x^{\mathfrak{r}}$ should be a red type A in $\Phi \mid \Gamma$ ctx.

There are a couple of downsides. If the goal is to stick to additive colour mixing, then one only has three primary colours available, and any use of three or more colours becomes an unpleasant shade of brown. In such situations we are better off sticking to explicit labels. It is also sometimes unclear at the start of a proof which colours will need to be split into subcolours, causing some retroactive 'palette swapping' once the best arrangement is determined.

Remark 1.2.6. Colours are often used when presenting a type theory to make symbols in different syntactic classes more visually distinct, but the labelling of variables with 'colours' (and literally colouring them accordingly) has also appeared previously in the Core Calculus of Coloured Constructions [BG13]. The theory is designed to give the user control over erasure: different parts of an expression can be assigned different colours and then erasing a colour deletes any part of the expression 'tainted' with that colour, the rules arranged so that typing is preserved.

Interestingly, the filtering metaphor in their system works the opposite way to ours. When combining two colour labels, they use additive rather than subtractive colour mixing, so blue + red becomes magenta rather than purple. In CCCC, when erasing magenta one thinks of looking at the term through a magenta filter: all the magenta subterms fade into the background and so disappear, as do the red and blue subterms. In our system, we focus instead on what happens to all the *other* colours: the filter makes them appear black.

1.3 Tensor

Our $\otimes$-type is a restricted version of the ordinary Σ-type so that to form a $\otimes$-pair, the two components must divide the resources of the palette in a linear way. We call such a division a *split*. For the first part of this section we will work with a simpler notion of split than the one we eventually use, because we can make a surprising amount of progress with it.

For a palette Φ with two colours $\Phi \vdash \mathfrak{r}$ colour and $\Phi \vdash \mathfrak{b}$ colour we use a judgement $\Phi \vdash \mathfrak{r} \boxtimes \mathfrak{b}$ split to mean that Φ has $\mathfrak{r}$ tensored with $\mathfrak{b}$ at the top level (possibly with some subpalettes Ψ_1 and Ψ_2 below them):

$$\frac{}{\mathfrak{p} \prec \Phi, (\mathfrak{r} \prec \Psi_1) \otimes (\mathfrak{b} \prec \Psi_2), \Phi' \vdash \mathfrak{l} \boxtimes \mathfrak{r} \text{ split}}$$

A split can therefore use a tensor verbatim: $\mathfrak{a} \otimes \mathfrak{b} \vdash \mathfrak{a} \boxtimes \mathfrak{b}$ split, and also builds in cartesian projection: $\mathfrak{a} \otimes \mathfrak{b}, \mathfrak{c} \otimes \mathfrak{d} \vdash \mathfrak{a} \boxtimes \mathfrak{b}$ split. But there is no way to use both $\mathfrak{a}$ on the left and $\mathfrak{c}$ on the right of a single split: a split must choose a single side of a comma to use.

Unfortunately, in general it will not be enough to refer to pieces of a palette described by a single colour label. Already when proving associativity of $\otimes$, it will be necessary to have a collection of colours from the palette on either side of a split. We defer the use of these 'slices' until Section 1.3.1.

The type $A \otimes B$ is well-formed when the types A and B only depend on the *nonlinear* resources of the context: syntactically A and B are required to be dull in the same sense as the previous section. In fact, we generalise further and allow our tensor type to be dependent between the two sides: the dull type B may depend on a marked variable $\underline{x}$ of type A. We write the dependent tensor as $\textcircled{\scriptsize$\Sigma$}_{(\underline{x}:A)} B$, as a pun on the relationship between $\otimes$ and $\times$.

The rules for $\otimes$-types are given in Figure 1.10.

- **Formation:** Whenever we have a dull type A and a dull type B depending on $\underline{x} : A$, we can form the type $\textcircled{\scriptsize$\Sigma$}_{(\underline{x}:A)} B$. We write $\underline{x} : A$ rather than $x : A$ in the syntax of the type to remind

$$\otimes\text{-FORM}\ \frac{\mathfrak{t} \mid \underline{\Gamma} \vdash A\ \mathsf{type} \qquad \mathfrak{t} \mid \underline{\Gamma}, \underline{x}^{\mathfrak{t}} : \underline{A} \vdash B\ \mathsf{type}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \textcircled{Σ}_{(\underline{x}:A)}\, B\ \mathsf{type}}$$

$$\otimes\text{-INTRO}\ \frac{\begin{array}{c}\mathfrak{t} \prec \Phi \vdash \mathfrak{l} \boxtimes \mathfrak{r}\ \mathsf{split} \\ \Phi^{\mathfrak{l}} \mid \Gamma^{\mathfrak{l}} \vdash a : A^{\mathfrak{l} \leftrightarrow \mathfrak{t}} \qquad \Phi^{\mathfrak{r}} \mid \Gamma^{\mathfrak{r}} \vdash b : B[\underline{a}/\underline{x}]^{\mathfrak{r} \leftrightarrow \mathfrak{t}}\end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash a\ {}_{\mathfrak{l}}{\otimes}_{\mathfrak{r}}\ b : \textcircled{Σ}_{(\underline{x}:A)}\, B}$$

$$\otimes\text{-ELIM}\ \frac{\begin{array}{c}\mathfrak{t} \prec \Phi \mid \Gamma, z^{\mathfrak{t}} : \textcircled{Σ}_{(\underline{x}:A)}\, B \vdash C\ \mathsf{type} \\ \mathfrak{t} \prec \Phi, \mathfrak{l} \otimes \mathfrak{r} \mid \Gamma, x^{\mathfrak{l}} : A^{\mathfrak{l} \leftrightarrow \mathfrak{t}}, y^{\mathfrak{r}} : B^{\mathfrak{r} \leftrightarrow \mathfrak{t}} \vdash c : C[x\ {}_{\mathfrak{l}}{\otimes}_{\mathfrak{r}}\ y/z] \\ \mathfrak{t} \prec \Phi \mid \Gamma \vdash s : \textcircled{Σ}_{(\underline{x}:A)}\, B\end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \mathsf{let}\ x\ {}_{\mathfrak{l}}{\otimes}_{\mathfrak{r}}\ y = s\ \mathsf{in}\ c : C[s/z]}$$

$$\otimes\text{-BETA}\ \frac{\begin{array}{c}\mathfrak{t} \prec \Phi \mid \Gamma, z^{\mathfrak{t}} : \textcircled{Σ}_{(\underline{x}:A)}\, B \vdash C\ \mathsf{type} \\ \mathfrak{t} \prec \Phi, \mathfrak{l} \otimes \mathfrak{r} \mid \Gamma, x^{\mathfrak{l}} : A^{\mathfrak{l} \leftrightarrow \mathfrak{t}}, y^{\mathfrak{r}} : B^{\mathfrak{r} \leftrightarrow \mathfrak{t}} \vdash c : C[x\ {}_{\mathfrak{l}}{\otimes}_{\mathfrak{r}}\ y/z] \\ \mathfrak{t} \prec \Phi \vdash \mathfrak{l}' \boxtimes \mathfrak{r}'\ \mathsf{split} \\ \Phi^{\mathfrak{l}'} \mid \Gamma^{\mathfrak{l}'} \vdash a : A^{\mathfrak{l}' \leftrightarrow \mathfrak{t}} \qquad \Phi^{\mathfrak{r}'} \mid \Gamma^{\mathfrak{r}'} \vdash b : B[\underline{a}/\underline{x}]^{\mathfrak{r}' \leftrightarrow \mathfrak{t}}\end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash (\mathsf{let}\ x\ {}_{\mathfrak{l}}{\otimes}_{\mathfrak{r}}\ y = a\ {}_{\mathfrak{l}'}{\otimes}_{\mathfrak{r}'}\ b\ \mathsf{in}\ c) \equiv c[\mathfrak{l}'/\mathfrak{l} \otimes \mathfrak{r}'/\mathfrak{r} \mid a/x, b/y] : C[a \otimes b/z]}$$

$$\text{SUBST-}\otimes\ \begin{array}{c}\mathfrak{t} \prec \Phi \vdash \mathfrak{l}' \boxtimes \mathfrak{r}'\ \mathsf{split} \\ \Phi^{\mathfrak{l}'} \mid \Gamma^{\mathfrak{l}'} \vdash a : A^{\mathfrak{l}' \leftrightarrow \mathfrak{t}} \qquad \Phi^{\mathfrak{r}'} \mid \Gamma^{\mathfrak{r}'} \vdash b : B[\underline{a}/\underline{x}]^{\mathfrak{r}' \leftrightarrow \mathfrak{t}} \\ \mathfrak{t} \prec \Phi, \mathfrak{l} \otimes \mathfrak{r} \mid \Gamma, x^{\mathfrak{l}} : A^{\mathfrak{l} \leftrightarrow \mathfrak{t}}, y^{\mathfrak{r}} : B^{\mathfrak{r} \leftrightarrow \mathfrak{t}} \vdash \mathcal{J} \\ \hdashline \mathfrak{t} \prec \Phi \mid \Gamma \vdash \mathcal{J}[\mathfrak{l}'/\mathfrak{l} \otimes \mathfrak{r}'/\mathfrak{r} \mid a/x, b/y]\end{array}$$

Figure 1.10: Rules for the Tensor Type (with Simple Splits)

us that all uses of x in B must be marked. Similarly to ordinary Σ-types, in the case that B does not depend on A we write $A \otimes B$.

- **Introduction:** If the ambient palette can be split into two colours $\mathfrak{l}$ and $\mathfrak{r}$, with $\mathfrak{l}$ proving $a : A^{\mathfrak{l} \leftrightarrow \mathfrak{t}}$ and $\mathfrak{r}$ proving $b : B[\underline{a}/\underline{x}]^{\mathfrak{r} \leftrightarrow \mathfrak{t}}$, then there is a term $a \;{}_{\mathfrak{l}}\!\otimes_{\mathfrak{r}}\; b$ of type $\bigotimes_{(\underline{x}:A)} B$.

  It is typically obvious which split of the resources has been used in an occurrence of $\otimes$-INTRO, just by inspecting how the variables appear marked or unmarked on both sides. When working informally we will often leave these splits off, unless there is some ambiguity in which split has been used. (Later we will come across examples where this happens.)

  In the terms $a : A^{\mathfrak{l} \leftrightarrow \mathfrak{t}}$ and $b : B[\underline{a}/\underline{x}]^{\mathfrak{r} \leftrightarrow \mathfrak{t}}$, the types are originally in palette $\mathfrak{t}$, and so have to have their top colours renamed to the colour on the appropriate side of the split, so $\mathfrak{l}$ or $\mathfrak{r}$ respectively. The types are then silently weakened via the MARKWK rule, giving types with palette $\Phi^{\mathfrak{l}}$ and $\Phi^{\mathfrak{r}}$ respectively. It is for this reason that $a : A^{\mathfrak{t} \leftrightarrow \mathfrak{l}}$ and $b : B[\underline{a}/\underline{x}]^{\mathfrak{t} \leftrightarrow \mathfrak{r}}$ are well formed, even though the terms use linear resources from the context and the types do not.

  Marking a term that contains a use of $\otimes$-intro is a situation which will require the more general 'slices' of Section 1.3.1, so we will simply postpone any examples that require this until after that section.

- **Elimination:** Any term p of $\bigotimes_{(\underline{x}:A)} B$ may be assumed to be of the form $x \;{}_{\mathfrak{r}}\!\otimes_{\mathfrak{b}}\; y$, where x and y are assigned fresh colours $\mathfrak{r}$ and $\mathfrak{b}$ that are tensored together as $\mathfrak{r} \otimes \mathfrak{b}$ at the top level in the ambient palette. We call this *$\otimes$-induction*.

  Syntactically, if we have a target type C that depends on a variable $z^{\mathfrak{t}} : \bigotimes_{(\underline{x}:A)} B$ and $c : C[x \;{}_{\mathfrak{l}}\!\otimes_{\mathfrak{r}}\; y/z]$ is a term that uses variables x and y as above, then for any $p : \bigotimes_{(\underline{x}:A)} B$ we have an induced term
$$\mathsf{let}\; x \;{}_{\mathfrak{l}}\!\otimes_{\mathfrak{r}}\; y = p \;\mathsf{in}\; c : C[p/z].$$

- **Computation:** If we perform $\otimes$-induction on a term that is actually already of the form $a \;{}_{\mathfrak{l}'}\!\otimes_{\mathfrak{r}'}\; b$, then the result is the term c with a and b substituted for x and y, and the colours $\mathfrak{l}'$ and $\mathfrak{r}'$ substituted for the colours $\mathfrak{l}$ and $\mathfrak{r}$, via the admissible SUBST-$\otimes$:
$$(\mathsf{let}\; x \;{}_{\mathfrak{l}}\!\otimes_{\mathfrak{r}}\; y = a \;{}_{\mathfrak{l}'}\!\otimes_{\mathfrak{r}'}\; b \;\mathsf{in}\; c) \equiv c[\mathfrak{l}'/\mathfrak{l} \otimes \mathfrak{r}'/\mathfrak{r} \mid a/x^{\mathfrak{l}}, b/y^{\mathfrak{r}}] : C[a \;{}_{\mathfrak{l}'}\!\otimes_{\mathfrak{r}'}\; b/z].$$

  On raw syntax, the operation on is implemented by plugging in $\mathfrak{l}$, $\mathfrak{r}$, a and b for the appropriate variables in the term c, just as in ordinary substitution.

Remark 1.3.1. The rule for forming a split forbids us from forming a diagonal map $\underline{A} \to \underline{A} \otimes \underline{A}$. Given an $x : \underline{A}$, there is no split $\mathfrak{l} \boxtimes \mathfrak{r}$ split such that $x \;{}_{\mathfrak{l}}\!\otimes_{\mathfrak{r}}\; x$ is well-formed, as any split assigns $\mathfrak{p}$ to exactly one side.

Similarly, our variable rule forbids us from forming projection maps $\underline{A} \otimes \underline{A} \to \underline{A}$. Given $p : \underline{A} \otimes \underline{A}$, the term $(\mathsf{let}\; x \;{}_{\mathfrak{r}}\!\otimes_{\mathfrak{b}}\; y = p \;\mathsf{in}\; x)$ is not well-formed, because $x : \underline{A}$ is not a well-formed term with colour $\mathfrak{p}$.

Remark 1.3.2. We use the same symbol $\otimes$ to form the type as to introduce a term. This should be familiar from working with the tensor product in ordinary algebra: there is not usually any confusion between the module $M \otimes N$ and an element $x \otimes y$. There is a possible ambiguity if we leave off the slices in the syntax of $\otimes$-INTRO: if A and B are types, then $A \otimes B$ could mean either $A \otimes B : \mathcal{U}$ via $\otimes$-FORM or $A \otimes B : \mathcal{U} \otimes \mathcal{U}$ via $\otimes$-INTRO. In this thesis there are no examples of the latter meaning.

Remark 1.3.3. In an axiomatic style, our introduction rule would be:

$$\frac{}{\mathfrak{t} \prec \Phi, \mathfrak{r} \otimes \mathfrak{b} \mid \Gamma, x^{\mathfrak{r}} : A, y^{\mathfrak{b}} : B \vdash \mathsf{axtensor}_{x,y} : \textcircled{\Sigma}_{(\underline{x}:A)} B}$$

Our actual rule is obtained from this by building in an arbitrary substitution for the 'telescope' $\mathfrak{r} \otimes \mathfrak{b} \mid x^{\mathfrak{r}} : A, y^{\mathfrak{b}} : B \ \mathsf{tele}$.

Remark 1.3.4. We can describe the shape of the $\otimes$-type former as a function into the universe. The left type must be marked with respect to the ambient context, we can capture this internally as asking for a term $A : \natural\mathcal{U}$. The right type also must be marked with respect to the context, and can also depend on A but only marked; this a type family $A : \natural\underline{A}_{\natural} \to \natural\mathcal{U}$. Given these inputs we can form the dependent tensor, so the type former itself has type

$$\text{“}\textcircled{\Sigma}\text{”} : \prod_{(A:\natural\mathcal{U})} (\natural\underline{A}_{\natural} \to \natural\mathcal{U}) \to \mathcal{U}$$

When positing a $\otimes$-type, it is a little cumbersome to suppose $A : \natural\mathcal{U}$ and $B : \natural\underline{A}_{\natural} \to \natural\mathcal{U}$, and then work with the type $\textcircled{\Sigma}_{(\underline{x}:\underline{A}_{\natural})} (\underline{B}(\underline{x}^{\natural}))_{\natural}$. Instead, we will usually assume $\underline{A} : \mathcal{U}$ and $\underline{B} : \underline{A} \to \mathcal{U}$, and work with $\textcircled{\Sigma}_{(\underline{x}:\underline{A})} \underline{B}(\underline{x})$. The type of $\underline{B}$ is not the most precise possible, because we could instantiate $\underline{B}$ with a function that uses its argument $x' : \underline{A}$ unmarked, and then apply $\underline{B}$ to $\underline{x}$ when forming the $\otimes$-type anyway. We accept this trade-off, because statement written this way are a lot neater.

With the $\otimes$-type now properly introduced, we prove some of its basic properties internally. First, we can revisit the symmetry equivalence given in Proposition 1.2.1.

Proposition 1.3.5. *For any dull types $\underline{A}$ and $\underline{B}$, there is a symmetry equivalence $\mathsf{sym}_{\underline{A},\underline{B}} : \underline{A} \otimes \underline{B} \to \underline{B} \otimes \underline{A}$ whose inverse is $\mathsf{sym}_{\underline{B},\underline{A}}$.*

The map in Proposition 1.2.1 was defined for A and B closed types, but exactly the same definition can be used when they are instead only dull:

$$\mathsf{sym}_{\underline{A},\underline{B}} :\equiv \lambda p.(\mathsf{let}\ x^{\mathfrak{r}} \otimes y^{\mathfrak{b}} = p \ \mathsf{in}\ y \otimes x).$$

To show that this is an equivalence, we will need the uniqueness principle for $\otimes$ which we showed in Proposition 1.2.2. We also generalise this proposition to dull types, and allow $\underline{B}$ to depend on $\underline{x} : \underline{A}$.

Proposition 1.3.6 (Uniqueness principle for $\otimes$)**.** *Suppose $\underline{A}$ and $\underline{B} : \underline{A} \to \mathcal{U}$ are types. For any type family $C : \textcircled{\Sigma}_{(\underline{x}:\underline{A})} \underline{B}(\underline{x}) \to \mathcal{U}$, section $f : \prod_{(p:\textcircled{\Sigma}_{(\underline{x}:\underline{A})} \underline{B}(\underline{x}))} C(p)$ and term $p : \textcircled{\Sigma}_{(\underline{x}:\underline{A})} \underline{B}(\underline{x})$ we have*

$$(\mathsf{let}\ x\ {}_{\mathfrak{r}}\!\otimes_{\mathfrak{b}}\ y = p \ \mathsf{in}\ f(x\ {}_{\mathfrak{r}}\!\otimes_{\mathfrak{b}}\ y)) = f(p)$$

Proof. Let $P : A \otimes B \to \mathcal{U}$ denote the type family

$$P(p) :\equiv (\text{let } x\ {}_{\mathfrak{r}}\!\otimes_{\mathfrak{b}} y = p \text{ in } f(x\ {}_{\mathfrak{r}}\!\otimes_{\mathfrak{b}} y)) = f(p)$$

We wish to find an element $\prod_{(p:A\otimes B)} P(p)$, so by $\otimes$-induction it is enough to assume $p \equiv x'\ {}_{\mathfrak{r}'}\!\otimes_{\mathfrak{b}'} y'$ where $\mathfrak{r}' \otimes \mathfrak{b}'$ is a new split of our top colour.

Our new goal is

$$(\text{let } x\ {}_{\mathfrak{r}}\!\otimes_{\mathfrak{b}} y = x'\ {}_{\mathfrak{r}'}\!\otimes_{\mathfrak{b}'} y' \text{ in } f(x\ {}_{\mathfrak{r}}\!\otimes_{\mathfrak{b}} y)) = f(x'\ {}_{\mathfrak{r}'}\!\otimes_{\mathfrak{b}'} y')$$

and reducing the left-hand side gives

$$f(x'\ {}_{\mathfrak{r}'}\!\otimes_{\mathfrak{b}'} y') = f(x'\ {}_{\mathfrak{r}'}\!\otimes_{\mathfrak{b}'} y')$$

for which we have $\text{refl}_{f(x'\ {}_{\mathfrak{r}'}\otimes_{\mathfrak{b}'} y')}$. □

The $\otimes$-induction that provides the colours $\mathfrak{r}' \otimes \mathfrak{b}'$ is placing the new split *in a* $\times$*-bunch at the top of the palette*, so the new state of the palette is $\mathfrak{p} \prec (\mathfrak{r} \otimes \mathfrak{b}, \mathfrak{r}' \otimes \mathfrak{b}')$. This means that, for example, the variables x and x' cannot interact directly, and the type $x =_{\underline{A}} x'$ is not well formed, even though x and x' are both of type $\underline{A}$.

Proof of Proposition 1.3.5. Using the symmetry map twice gives:

$$\text{sym}_{\underline{B},\underline{A}}(\text{sym}_{\underline{A},\underline{B}}(p)) \equiv \left(\text{let } y'^{\mathfrak{b}'} \otimes x'^{\mathfrak{r}'} = (\text{let } x^{\mathfrak{r}} \otimes y^{\mathfrak{b}} = p \text{ in } y \otimes x) \text{ in } x' \otimes y'\right)$$

(The variables used in $\text{sym}_{\underline{B},\underline{A}}$ have been α-renamed to line up better with the result of $\text{sym}_{\underline{A},\underline{B}}$.)

This expression does not judgmentally reduce, but it would if p were a term constructed by $\otimes$-INTRO. To make this so, we use the uniqueness principle, which exposes a couple of opportunities to apply the computation rule.

$$\begin{aligned}
\text{sym}_{\underline{B},\underline{A}}(\text{sym}_{\underline{A},\underline{B}}(p)) &= (\text{let } x''^{\mathfrak{r}''} \otimes y''^{\mathfrak{b}''} = p \text{ in } \text{sym}_{\underline{B},\underline{A}}(\text{sym}_{\underline{A},\underline{B}}(x'' \otimes y''))) \\
&\equiv (\text{let } x''^{\mathfrak{r}''} \otimes y''^{\mathfrak{b}''} = p \text{ in } \text{sym}_{\underline{B},\underline{A}}(\text{let } x^{\mathfrak{r}} \otimes y^{\mathfrak{b}} = x'' \otimes y'' \text{ in } y \otimes x)) \\
&\equiv (\text{let } x''^{\mathfrak{r}''} \otimes y''^{\mathfrak{b}''} = p \text{ in } \text{sym}_{\underline{B},\underline{A}}(y'' \otimes x'')) \\
&\equiv (\text{let } x''^{\mathfrak{r}''} \otimes y''^{\mathfrak{b}''} = p \text{ in } (\text{let } y'^{\mathfrak{b}'} \otimes x'^{\mathfrak{r}'} = y'' \otimes x'' \text{ in } x' \otimes y')) \\
&\equiv (\text{let } x''^{\mathfrak{r}''} \otimes y''^{\mathfrak{b}''} = p \text{ in } x'' \otimes y'') \\
&= p
\end{aligned}$$

where we have ended by using the uniqueness principle in the other direction. □

Definition 1.3.7 (Action of $\otimes$ on functions)**.** Given dull $\underline{f} : \underline{A} \to \underline{A}'$ and $\underline{g} : \underline{B} \to \underline{B}'$, there is a function

$$\text{func}(\underline{f}, \underline{g}) : \underline{A} \otimes \underline{B} \to \underline{A}' \otimes \underline{B}'$$

defined by

$$\text{func}(\underline{f}, \underline{g}) :\equiv \lambda p.\text{let } x^{\mathfrak{r}} \otimes y^{\mathfrak{b}} = p \text{ in } \underline{f}(x)\ {}_{\mathfrak{r}}\!\otimes_{\mathfrak{b}} \underline{g}(y).$$

or dependently, if $\underline{f} : \underline{A} \to \underline{A}'$ and $\underline{g} : \prod_{(x:\underline{A})} \underline{B}(\underline{x}) \to \underline{B}'(\underline{f}(\underline{x}))$, there is

$$\mathsf{func}(\underline{f}, \underline{g}) : \textcircled{Σ}_{(\underline{x}:\underline{A})}\underline{B}(\underline{x}) \to \textcircled{Σ}_{(\underline{x}:\underline{A}')}\underline{B}'(\underline{x})$$

defined by

$$\mathsf{func}(\underline{f}, \underline{g}) :\equiv \lambda p.\mathsf{let}\ x^{\mathfrak{r}} \otimes y^{\mathfrak{b}} = p \ \mathsf{in}\ \underline{f}(x)\ {}_{\mathfrak{r}}\!\otimes_{\mathfrak{b}} \underline{g}(\underline{x})(y).$$

Proposition 1.3.8 (⊗ is functorial)**.** *If $\underline{f} : \underline{A} \to \underline{A}'$, $\underline{f}' : \underline{A}' \to \underline{A}''$ and $\underline{g} : \underline{B} \to \underline{B}'$, $\underline{g}' : \underline{B}' \to \underline{B}''$, we have*

$$\begin{aligned}\mathsf{func}(\underline{f}' \circ \underline{f}, \underline{g}' \circ \underline{g}) &= \mathsf{func}(\underline{f}', \underline{g}') \circ \mathsf{func}(\underline{f}, \underline{g}) \\ \mathsf{func}(\mathsf{id}_{\underline{A}}, \mathsf{id}_{\underline{B}}) &= \mathsf{id}_{\underline{A} \otimes \underline{B}}\end{aligned}$$

Proof. By function extensionality, we have to show

$$\mathsf{func}(\underline{f}' \circ \underline{f}, \underline{g}' \circ \underline{g})(p) = (\mathsf{func}(\underline{f}', \underline{g}') \circ \mathsf{func}(\underline{f}, \underline{g}))(p)$$

for any $p : \underline{A} \otimes \underline{B}$. Using ⊗-induction, assume $p \equiv x\ {}_{\mathfrak{r}}\!\otimes_{\mathfrak{b}} y$, and then the left side reduces to

$$\begin{aligned}(\mathsf{func}(\underline{f}' \circ \underline{f}, \underline{g}' \circ \underline{g})(x\ {}_{\mathfrak{r}}\!\otimes_{\mathfrak{b}} y) &\equiv (\underline{f}' \circ \underline{f})(x)\ {}_{\mathfrak{r}}\!\otimes_{\mathfrak{b}} (\underline{g}' \circ \underline{g})(y) \\ &\equiv \underline{f}'(\underline{f}(x))\ {}_{\mathfrak{r}}\!\otimes_{\mathfrak{b}} \underline{g}'(\underline{g}(y))\end{aligned}$$

and the right side reduces to

$$\begin{aligned}(\mathsf{func}(\underline{f}', \underline{g}') \circ \mathsf{func}(\underline{f}, \underline{g}))(x\ {}_{\mathfrak{r}}\!\otimes_{\mathfrak{b}} y) &\equiv \mathsf{func}(\underline{f}', \underline{g}')(\mathsf{func}(\underline{f}, \underline{g})(x\ {}_{\mathfrak{r}}\!\otimes_{\mathfrak{b}} y)) \\ &\equiv \mathsf{func}(\underline{f}', \underline{g}')(\underline{f}(x)\ {}_{\mathfrak{r}}\!\otimes_{\mathfrak{b}} \underline{g}(y)) \\ &\equiv \underline{f}'(\underline{f}(x))\ {}_{\mathfrak{r}}\!\otimes_{\mathfrak{b}} \underline{g}'(\underline{g}(y))\end{aligned}$$

so we have $\mathsf{refl}_{\underline{f}'(\underline{f}(x))\,{}_{\mathfrak{r}}\otimes_{\mathfrak{b}}\underline{g}'(\underline{g}(y))}$ as the required path. The path for $(\mathsf{id}_{\underline{A}} \otimes \mathsf{id}_{\underline{B}}) = \mathsf{id}_{\underline{A} \otimes \underline{B}}$ is constructed similarly. □

Remark 1.3.9. There does not appear to be a useful version of functoriality where the functions are used unmarked, although one can define a function

$$((\underline{A} \to \underline{A}') \times \underline{A}) \otimes ((\underline{B} \to \underline{B}') \times \underline{B}) \to (\underline{A}' \otimes \underline{B}')$$

by functoriality.

For convenience, we define two maps that project the components of the base from any tensor.

Definition 1.3.10. For any $\underline{A}$ and $\underline{B} : \underline{A} \to \mathcal{U}$, define

$$\begin{aligned}&\underline{\mathsf{pr}_1} : \textcircled{Σ}_{(\underline{x}:\underline{A})}\underline{B}(\underline{a}) \to \underline{A} \\ &\underline{\mathsf{pr}_2} : \textstyle\prod_{(p:\textcircled{Σ}_{(\underline{x}:\underline{A})}\underline{B}(\underline{a}))}\underline{B}(\underline{\mathsf{pr}_1}(p))\end{aligned}$$

by

$$\begin{aligned}\underline{\mathsf{pr}_1}(p) &:\equiv \mathsf{let}\ x \otimes y = p\ \mathsf{in}\ \underline{x} \\ \underline{\mathsf{pr}_2}(p) &:\equiv \mathsf{let}\ x \otimes y = p\ \mathsf{in}\ \underline{y}\end{aligned}$$

These functions clearly satisfy the computation rules $\underline{\mathsf{pr}}_1(x \otimes y) \equiv \underline{x}$ and $\underline{\mathsf{pr}}_2(x \otimes y) \equiv \underline{y}$ by the computation rule for $\otimes$-induction. Put together, these give a canonical map to the base spaces.

Definition 1.3.11.

$$\begin{aligned} \mathsf{mon} : \natural\left(\text{\textcircled{Σ}}_{(\underline{x}:\underline{A})}\underline{B}(\underline{a})\right) &\to \Sigma_{(x:\natural A)}\natural B(\underline{x}_\natural) \\ \mathsf{mon}(p) &:\equiv ((\underline{\mathsf{pr}}_1 \underline{p}_\natural)^\natural, (\underline{\mathsf{pr}}_2 \underline{p}_\natural)^\natural) \end{aligned}$$

The $\otimes$ type interacts with $\natural$ types in a convenient way.

Proposition 1.3.12 ($\natural$ types float through $\otimes$)**.** *For dull types $\underline{A}, \underline{B}, \underline{C}$, there are equivalences:*

$$\begin{aligned} \natural\underline{A} \times (\underline{B} \otimes \underline{C}) &\simeq (\natural\underline{A} \times \underline{B}) \otimes \underline{C} \\ \underline{A} \otimes (\natural\underline{B} \times \underline{C}) &\simeq (\underline{A} \times \natural\underline{B}) \otimes C \\ \underline{A} \otimes (\underline{B} \times \natural\underline{C}) &\simeq (\underline{A} \otimes \underline{B}) \times \natural\underline{C} \end{aligned}$$

More dependently,

$$\begin{aligned} \Sigma_{(x:\natural\underline{A})}\text{\textcircled{Σ}}_{(\underline{y}:\underline{B}(\underline{x}))}\underline{C}(\underline{x},\underline{y}) &\simeq \text{\textcircled{Σ}}_{((x,\underline{y}):\Sigma_{(x:\natural\underline{A})}\underline{B}(\underline{x}))}\underline{C}(\underline{x},\underline{y}) \\ \text{\textcircled{Σ}}_{(\underline{x}:\underline{A})}\Sigma_{(y:\natural\underline{B}(\underline{x}))}\underline{C}(\underline{x},\underline{y}) &\simeq \text{\textcircled{Σ}}_{((\underline{x},\underline{y}):\Sigma_{(x:\underline{A})}\natural\underline{B}(\underline{x}))}\underline{C}(\underline{x},\underline{y}) \\ \text{\textcircled{Σ}}_{(\underline{x}:\underline{A})}\Sigma_{(y:\underline{B}(\underline{x}))}\natural\underline{C}(\underline{x},\underline{y}) &\simeq \Sigma_{(w:\text{\textcircled{Σ}}_{(\underline{x}:\underline{A})}\underline{B}(\underline{x}))}\natural\underline{C}(\underline{\mathsf{pr}}_1\underline{w}, \underline{\mathsf{pr}}_2\underline{w}) \end{aligned}$$

Of course, in the simply typed case, the first equivalence immediately implies the other two by symmetry of $\otimes$ and $\times$.

The intuition here is that if A is a space, the spectrum over each point of $\natural A$ is trivial. So considering the first equivalence, the spectrum over a point $(\underline{a},(\underline{b},\underline{c})) : \natural\underline{A} \times (\natural\underline{B} \times \natural\underline{C})$ on the left is given by $1 \times (\underline{B}_{\underline{b}} \otimes \underline{C}_{\underline{c}})$, and over a point $((\underline{a},\underline{b}),\underline{c}) : (\natural\underline{A} \times \natural\underline{B}) \times \natural\underline{C}$ on the right is given by $(1 \times \underline{B}_{\underline{b}}) \otimes \underline{C}_{\underline{c}}$, and these are obviously equivalent.

The colour of the variable introduced by the Σ-type is different on each side of the dependent equivalences. This is a hint that the $\natural$ is necessary for the proof to go through: terms of $\natural$ type are always judgmentally marked via Proposition 1.1.7, and are therefore 'colourless'.

Proof. The maps are defined in the obvious ways using $\otimes$-induction. The critical step comes at the end when checking the round-trips: we use that all terms of $\natural$ type are definitionally marked (Proposition 1.1.7). For the first equivalence, this is

$$\begin{aligned} (\underline{x}, y) \otimes z \equiv (x, y) \otimes z &: \text{\textcircled{Σ}}_{((\underline{x},\underline{y}):\Sigma_{(x:\natural\underline{A})}\underline{B}(\underline{x}))}\underline{C}(\underline{x},\underline{y}) \\ (\underline{x}, y \otimes z) \equiv (x, y \otimes z) &: \Sigma_{(x:\natural\underline{A})}\text{\textcircled{Σ}}_{(\underline{y}:\underline{B}(\underline{x}))}\underline{C}(\underline{x},\underline{y}) \end{aligned}$$

□

This interaction of the $\natural$ and $\otimes$ type formers also allows us to reconstruct the dependent $\otimes$-type from a non-dependent $\otimes$-type.

Proposition 1.3.13. *For any $\underline{A} : \mathcal{U}$ and $\underline{B} : \underline{A} \to \mathcal{U}$, there is an equivalence*

$$\text{\textcircled{Σ}}_{(\underline{x}:\underline{A})}\underline{B}(\underline{x}) \simeq \Sigma_{(n:\natural\underline{A})}\Sigma_{(p:\underline{A}\otimes\underline{B}(\underline{n}_\natural))}(\underline{n} = \underline{\mathsf{pr}}_1(\underline{p})^\natural)$$

This might feel strange, but there is an analogous (and not very useful) equivalence for Σ-types

$$\sum_{(x:A)} B(x) \simeq \sum_{(x':A)} \sum_{((x,y):A \times B(x'))} (x' = x)$$

which follows quickly from singleton contractibility.

Proof. For any $\underline{A}$, there is an equivalence

$$\underline{A} \simeq \sum_{(n:\natural\underline{A})} \sum_{(x:\underline{A})} (\underline{n} = \underline{x}^{\natural})$$

given by singleton contractibility, and that $n \equiv \underline{n}$. The equivalence then follows by pulling spaces out of $\otimes$ (Proposition 1.3.12), because both $\natural A$ and $(\underline{n} = \underline{x}^{\natural})$ are spaces:

$$\begin{aligned}
\textcircled{Σ}_{(\underline{x}:\underline{A})} \underline{B}(\underline{x}) &\simeq \textcircled{Σ}_{((\underline{n},\underline{x},\underline{e}):\sum_{(n:\natural\underline{A})} \sum_{(x:\underline{A})} (\underline{n}=\underline{x}^{\natural}))} \underline{B}(\underline{n}_{\natural}) \\
&\simeq \sum_{(n:\natural\underline{A})} \textcircled{Σ}_{((\underline{x},\underline{e}):\sum_{(x:\underline{A})} (\underline{n}=\underline{x}^{\natural}))} \underline{B}(\underline{n}_{\natural}) && \text{(Proposition 1.3.12)} \\
&\simeq \sum_{(n:\natural\underline{A})} \textcircled{Σ}_{(\underline{x}:\underline{A})} (\underline{n} = \underline{x}^{\natural}) \times \underline{B}(\underline{n}_{\natural}) && \text{(Proposition 1.3.12)} \\
&\simeq \sum_{(n:\natural\underline{A})} \textcircled{Σ}_{(\underline{x}:\underline{A})} \underline{B}(\underline{n}_{\natural}) \times (\underline{n} = \underline{x}^{\natural}) \\
&\simeq \sum_{(n:\natural\underline{A})} \sum_{(p:\underline{A}\otimes\underline{B}(\underline{n}_{\natural}))} \mathsf{let}\ x \otimes y = p\ \mathsf{in}\ (\underline{n} = \underline{x}^{\natural}) && \text{(Proposition 1.3.12)} \\
&\simeq \sum_{(n:\natural\underline{A})} \sum_{(p:\underline{A}\otimes\underline{B}(\underline{n}_{\natural}))} (\underline{n} = (\mathsf{let}\ x \otimes y = p\ \mathsf{in}\ \underline{x})^{\natural}) && \text{(Proposition 1.3.6)} \\
&\equiv \sum_{(n:\natural\underline{A})} \sum_{(p:\underline{A}\otimes\underline{B}(\underline{n}_{\natural}))} (\underline{n} = \underline{\mathsf{pr}}_1(\underline{p})^{\natural})
\end{aligned}$$

□

Remark 1.3.14. We might hope that $\otimes$-types are 'bilinear', in the sense that

$$(x \,{}_{\mathfrak{r}}{\otimes}_{\mathfrak{b}}\, \underline{y}) = (\underline{x} \,{}_{\mathfrak{r}}{\otimes}_{\mathfrak{b}}\, \underline{y}) = (\underline{x} \,{}_{\mathfrak{r}}{\otimes}_{\mathfrak{b}}\, y),$$

but this does not follow from our rules, even including the hom-type later, as there are models where it does not hold. Consider some non-trivial model of the $\natural$ fragment of our theory, and then interpret $\textcircled{$\Sigma$}$ as ordinary Σ: ap of first projection on $(x \otimes \underline{y}) = (\underline{x} \otimes \underline{y})$ would then give $x = \underline{x}$ for any term $x : A$. Typically such a path does not exist. We will later impose an axiom that does imply the above 'bilinearity', see Section 1.5.4.

Remark 1.3.15. The reader may be wondering why we do not instead devise a type theory appropriate for working in any category that is both locally cartesian closed and monoidal, and why the $\natural$ modality needs to be made so tightly connected with $\otimes$. Unfortunately, a version of this type theory without the modality is not very expressive.

Suppose we instead have rules along the lines of

$$\frac{\Gamma \vdash A \ \mathsf{type} \qquad \Gamma' \vdash B \ \mathsf{type}}{\Gamma \otimes \Gamma' \vdash A \otimes B \ \mathsf{type}} \qquad\qquad \frac{\Gamma \vdash a : A \qquad \Gamma' \vdash b : B}{\Gamma \otimes \Gamma' \vdash a \otimes b : A \otimes B}$$

With these rules, we cannot even posit a monoid object internally. In an ambient context Γ, if we suppose A type then we can only form $A \otimes A$ type in context $\Gamma \otimes \Gamma$, and so there is no hope for forming the type of a multiplication operation $\cdot : A \otimes A \to A$.

(Of course, in an actual type theory, we would build a substitution for the context $\Gamma \otimes \Gamma'$ into the conclusion of these rules, but this does not solve the issue.)

The above rules without the modality do allow a different kind of dependency to the one in our rules: the types A and B can use the variables of Γ and Γ' in an unrestricted way. But our theory can simulate this apparently stronger rule. In the above rule, Γ and Γ' have no dependence between them and so we can consider the type $\Gamma \vdash A$ type as a (closed) map $\mathsf{pr}_1 : \sum_{(g:\underline{G})} \underline{A}(g) \to \underline{G}$, where $\underline{G}$ represents the type Γ packed into a single type (which is closed and therefore dull). Similarly consider B as a closed map $\mathsf{pr}_1 : \sum_{(g':\underline{G'})} \underline{B}(g') \to \underline{G'}$. The above version of $A \otimes B$ can be defined from ours by first using functoriality to form

$$\mathsf{pr}_1 \otimes \mathsf{pr}_1 : \left(\textstyle\sum_{(g:\underline{G})} \underline{A}(g)\right) \otimes \left(\textstyle\sum_{(g':\underline{G'})} \underline{B}(g')\right) \to \underline{G} \otimes \underline{G'}$$

and then calculating the fibres of this map:

$$\mathsf{fib}_{\mathsf{pr}_1 \otimes \mathsf{pr}_1} : \underline{G} \otimes \underline{G'} \to \mathcal{U}$$

1.3.1 Palette Slices

We now introduce the more general notion of split that is actually used in the type theory. The two sides of a split will be allowed to be more general collections of colours from the palette, rather than just a single colour. We call these collections *slices*, and use a judgement $\Phi \vdash s$ slice. A split will then divide the palette into two slices, as in $\Phi \vdash s_L \boxtimes s_R$ split.

As a piece of raw syntax, a slice is one of the following:

- $\mathfrak{r}$, a single colour from Φ;
- $\mathfrak{t}. \prec \mathfrak{c}_1 \otimes \mathfrak{c}_2 \otimes \cdots \otimes \mathfrak{c}_n (\otimes 1)$, a fresh colour $\mathfrak{t}$ followed by a list of labels from Φ optionally including the special symbol 1;

The periods in the latter option indicate that such a slice binds a fresh colour $\mathfrak{t}$.

Slices are typed using the rules in Figure 1.11. We have a 'preslice' judgement, representing a valid list of colours from the palette, a 'preslice$_\epsilon$' judgement that optionally adds $- \otimes 1$, and finally an actual 'slice' judgement, adding the fresh $\mathfrak{t}. \prec -$ label if necessary.

Not all lists of labels form a valid slice: the choices must be compatible with the shape of the palette. For example, when we reach a comma node we must choose exactly one branch to be included in the slice. A slice also cannot contain a colour together with any colours in the subpalette under that colour; choosing a colour already specifies that we are selecting the entire subpalette.

Remark 1.3.16. One way to identify valid preslices is to begin with the entire palette and delete pieces until we are left with a list of labels tensored together. Consider the palette

$$\mathfrak{t} \prec (\mathfrak{a} \prec (\mathfrak{p} \otimes \mathfrak{q}, \mathfrak{r} \otimes \mathfrak{s})) \otimes \mathfrak{b} \text{ palette}$$

from Section 1.2.2. If we delete the $\mathfrak{t}$ label, we have $(\mathfrak{a} \prec (\mathfrak{p} \otimes \mathfrak{q}, \mathfrak{r} \otimes \mathfrak{s})) \otimes \mathfrak{b}$, giving $\mathfrak{a} \otimes \mathfrak{b}$ as a valid preslice. We could continue by deleting $\mathfrak{a}$, $\mathfrak{q}$, $\mathfrak{r}$ and $\mathfrak{s}$ giving $\mathfrak{p} \otimes \mathfrak{b}$, another valid preslice. But starting from the same context, deleting everything but $\mathfrak{p}$ and $\mathfrak{r}$ leaves them separated by a comma, and so $\mathfrak{p} \otimes \mathfrak{r}$ is not a valid preslice in this palette.

$$\frac{}{\Phi \vdash \varnothing \ \mathsf{preslice}} \qquad \frac{}{\mathfrak{c} \prec \Phi \vdash \mathfrak{c} \ \mathsf{preslice}} \qquad \frac{\Phi \vdash s \ \mathsf{preslice}}{\mathfrak{c} \prec \Phi \vdash s \ \mathsf{preslice}}$$

$$\frac{\Phi_1 \vdash s \ \mathsf{preslice}}{\Phi_1, \Phi_2 \vdash s \ \mathsf{preslice}} \qquad \frac{\Phi_2 \vdash s \ \mathsf{preslice}}{\Phi_1, \Phi_2 \vdash s \ \mathsf{preslice}}$$

$$\frac{\Phi_L \vdash s_L \ \mathsf{preslice} \qquad \Phi_R \vdash s_R \ \mathsf{preslice}}{\Phi_L \otimes \Phi_R \vdash s_L \otimes s_R \ \mathsf{preslice}}$$

$$\frac{\Phi \vdash s \ \mathsf{preslice}}{\Phi \vdash s \ \mathsf{preslice}_\epsilon} \qquad \frac{\Phi \vdash s \ \mathsf{preslice}}{\Phi \vdash s \otimes 1 \ \mathsf{preslice}_\epsilon}$$

$$\frac{\Phi \vdash \mathfrak{c} \ \mathsf{colour}}{\Phi \vdash \mathfrak{c} \ \mathsf{slice}} \qquad \frac{\Phi \vdash s \ \mathsf{preslice}_\epsilon}{\Phi \vdash (\mathfrak{t}. \prec s) \ \mathsf{slice}}$$

Figure 1.11: Rules for Slices.

$$\textsc{pal-filter}\ \frac{\Phi \vdash s \ \mathsf{slice}}{\Phi^s \ \mathsf{palette}} \qquad \textsc{ctx-filter}\ \frac{\Phi \vdash s \ \mathsf{slice} \qquad \Phi \mid \Gamma \ \mathsf{ctx}}{\Phi^s \mid \Gamma^s \ \mathsf{ctx}}$$

Figure 1.12: Admissible Slice Rules.

Definition 1.3.17. For any slice $\Phi \vdash s \ \mathsf{slice}$, the *top colour* of s, written $\ulcorner s \urcorner$, is defined by

$$\begin{aligned} \ulcorner \mathfrak{c} \urcorner &:\equiv \mathfrak{c} \\ \ulcorner \mathfrak{c}. \prec s \urcorner &:\equiv \mathfrak{c} \end{aligned}$$

In other words, the top colour of a slice s is the fresh label that it binds, unless s is already a single, in which case the top colour is that single colour.

Definition 1.3.18. For any slice s, the *underlying preslice* of s, written $\mathsf{u}(s)$, is defined by

$$\begin{aligned} \mathsf{u}(\mathfrak{c}) &:\equiv \mathfrak{c} \\ \mathsf{u}(\mathfrak{t}. \prec s) &:\equiv s \end{aligned}$$

Filtering. The notion of filtering from Section 1.2 needs to be strengthened to work on a general slice of a palette rather than just a single colour, as shown in Figure 1.12.

If the slice s is a single colour $\mathfrak{c}$, then filtering a palette and context works exactly as before.

For a general slice $s \equiv \mathfrak{t}. \prec \mathfrak{c}_1 \otimes \mathfrak{c}_2 \otimes \cdots \otimes \mathfrak{c}_n$, filtering to Φ^s is defined by filtering to each colour $\Phi^{\mathfrak{c}_i}$, combining them via $\otimes$, and finally prepending $\mathfrak{t}$ as the new top colour. For example,

$$\begin{aligned}
(\mathfrak{t} \prec (\mathfrak{a} \prec \mathfrak{p} \otimes \mathfrak{q}, \mathfrak{r} \otimes \mathfrak{s}) \otimes (\mathfrak{b} \prec \mathfrak{u} \otimes \mathfrak{v}))^{\mathfrak{t}'.\prec \mathfrak{p} \otimes \mathfrak{b}} &:\equiv \mathfrak{t}' \prec (\mathfrak{p} \otimes (\mathfrak{b} \prec \mathfrak{u} \otimes \mathfrak{v}))\\
(\mathfrak{t} \prec (\mathfrak{a} \prec \mathfrak{p} \otimes \mathfrak{q}, \mathfrak{r} \otimes \mathfrak{s}) \otimes (\mathfrak{b} \prec \mathfrak{u} \otimes \mathfrak{v}))^{\mathfrak{t}'.\prec \mathfrak{a} \otimes \mathfrak{b}} &:\equiv \mathfrak{t}' \prec ((\mathfrak{a} \prec \mathfrak{p} \otimes \mathfrak{q}, \mathfrak{r} \otimes \mathfrak{s}) \otimes (\mathfrak{b} \prec \mathfrak{u} \otimes \mathfrak{v}))\\
(\mathfrak{t} \prec (\mathfrak{a} \prec \mathfrak{p} \otimes \mathfrak{q}, \mathfrak{r} \otimes \mathfrak{s}) \otimes (\mathfrak{b} \prec \mathfrak{u} \otimes \mathfrak{v}))^{\mathfrak{t}'.\prec \mathfrak{q} \otimes \mathfrak{u}} &:\equiv \mathfrak{t}' \prec (\mathfrak{q} \otimes \mathfrak{u})
\end{aligned}$$

If the slice contains a 1, we add the palette 1 combined with $\otimes$, even if 1 does not occur in the original palette:

$$(\mathfrak{t} \prec (\mathfrak{a} \prec \mathfrak{p} \otimes \mathfrak{q}, \mathfrak{r} \otimes \mathfrak{s}) \otimes (\mathfrak{b} \prec \mathfrak{u} \otimes \mathfrak{v}))^{\mathfrak{t}'.\prec \mathfrak{p} \otimes \mathfrak{b} \otimes 1} \equiv \mathfrak{t}' \prec ((\mathfrak{p} \otimes (\mathfrak{b} \prec \mathfrak{u} \otimes \mathfrak{v})) \otimes 1)$$

In all cases, the colour $\ulcorner s \urcorner$ becomes the top colour of the palette Φ^s.

Substitution into a Slice. There is a operation that substitutes one slice for a colour name in another.

First we define what it means to substitute one preslice into another. Preslices are lists, so we can simply define:

$$\begin{aligned}
(s \otimes \mathfrak{c} \otimes s')[\![t/\mathfrak{c}]\!] &:\equiv s \otimes t \otimes s' \\
s[\![t/\mathfrak{c}]\!] &:\equiv s \qquad\qquad\qquad\qquad \text{otherwise}
\end{aligned}$$

On actual slices, there are two cases depending on whether the target of the substitution binds its own top colour. If it does, then this top colour is maintained:

$$\begin{aligned}
\mathfrak{c}[\![t/\mathfrak{c}]\!] &:\equiv t \\
(\mathfrak{s}. \prec s)[\![t/\mathfrak{c}]\!] &:\equiv \mathfrak{s}. \prec s[\![\mathsf{u}(t)/\mathfrak{c}]\!]
\end{aligned}$$

1.3.2 Palette Splits

A split is specified by a judgement $\Phi \vdash s_L \boxtimes s_R \ \mathsf{split}$ where s_L and s_R are slices of Φ. A split has no associated term, it is merely a predicate on pairs of slices.

A split corresponds roughly to a morphism $\Phi \to \Phi^{s_L} \otimes \Phi^{s_R}$, building in all the associativity and cartesian weakening necessary for this to make sense. We write $\boxtimes$ as the separator for the two sides to distinguish it from the symbol $\otimes$ that may appear in the slices s_L and s_R.

We still have the examples we have been using where the slices are two colours that appear at the top level of the context, but there are new possibilities:

- The rules build in associativity: we will have

$$\mathfrak{t} \prec (\mathfrak{p} \prec \mathfrak{a} \otimes \mathfrak{b}) \otimes \mathfrak{c} \vdash \mathfrak{a} \boxtimes (\mathfrak{t}'. \prec \mathfrak{b} \otimes \mathfrak{c}) \ \mathsf{split}$$

  allowing us to use the resources $\mathfrak{b} \otimes \mathfrak{c}$, even though that combination does not appear verbatim in the palette.

- Similarly for unitality: one of the base rules gives

$$\mathfrak{a} \vdash \mathfrak{a} \boxtimes \varnothing \ \mathsf{split}$$

where the preslice $\varnothing$ represents the unit.

The formal rules for splits are given in Figure 1.13. Like slices, these are built in two stages. First a 'preslice' $\Phi \vdash s_L \boxtimes s_R$ presplit where s_L and s_R are preslices. A 'split' $\Phi \vdash s_L \boxtimes s_R$ split where s_L and s_R are slices, is then a presplit on the underlying preslices of s_L and s_R

We have compressed the rules for presplits using $(- \otimes \epsilon)$ in the first few rules to represent the possible addition of $- \otimes 1$ to the preslice, so that $(\mathfrak{c} \otimes \epsilon)$ denotes either $(\mathfrak{c} \otimes 1)$ or simply $\mathfrak{c}$. To be concrete, expanding all the possibilities gives

$$\begin{aligned}
\varnothing &\vdash \varnothing \boxtimes \varnothing \ \mathsf{presplit} \\
\mathfrak{c} \prec \Phi &\vdash \mathfrak{c} \boxtimes \varnothing \ \mathsf{presplit} \\
\mathfrak{c} \prec \Phi &\vdash \varnothing \boxtimes \mathfrak{c} \ \mathsf{presplit} \\
\\
\varnothing &\vdash 1 \boxtimes \varnothing \ \mathsf{presplit} \\
\varnothing &\vdash \varnothing \boxtimes 1 \ \mathsf{presplit} \\
\varnothing &\vdash 1 \boxtimes 1 \ \mathsf{presplit} \\
\Phi &\vdash 1 \boxtimes \varnothing \ \mathsf{presplit} \\
\Phi &\vdash \varnothing \boxtimes 1 \ \mathsf{presplit} \\
\Phi &\vdash 1 \boxtimes 1 \ \mathsf{presplit} \\
\mathfrak{c} \prec \Phi &\vdash (\mathfrak{c} \otimes 1) \boxtimes \varnothing \ \mathsf{presplit} \\
\mathfrak{c} \prec \Phi &\vdash \mathfrak{c} \boxtimes 1 \ \mathsf{presplit} \\
\mathfrak{c} \prec \Phi &\vdash (\mathfrak{c} \otimes 1) \boxtimes 1 \ \mathsf{presplit} \\
\mathfrak{c} \prec \Phi &\vdash \varnothing \boxtimes (\mathfrak{c} \otimes 1) \ \mathsf{presplit} \\
\mathfrak{c} \prec \Phi &\vdash 1 \boxtimes \mathfrak{c} \ \mathsf{presplit} \\
\mathfrak{c} \prec \Phi &\vdash 1 \boxtimes (\mathfrak{c} \otimes 1) \ \mathsf{presplit}
\end{aligned}$$

The idea is that the presence of a 1 in a split allows arbitrary pieces of the palette to be ignored via the unique map $\Phi \to 1$. The presence of 1 on both sides of a split is also sound, because $1 \otimes 1 \simeq 1$ follows from the fact that 1 is both initial and terminal in the category of 'linear' objects.

1.3.3 Tensor Type Revisited

Figure 1.14 now builds this more general notion of splits into the rules for $\otimes$.

- **Formation:** Unchanged.
- **Introduction:** If the ambient palette can be split into two slices $s_L \boxtimes s_R$ split, with s_L proving $a : A^{\mathsf{t} \leftrightarrow \ulcorner s_L \urcorner}$ and s_R proving $b : B[\underline{a}/\underline{x}]^{\mathsf{t} \leftrightarrow \ulcorner s_R \urcorner}$, then there is a term $a \ {}_{s_L}\!\otimes_{s_R} b$ of type $\textcircled{\scriptsize$\Sigma$}_{(\underline{x}:A)} B$.

$$\frac{}{\varnothing \vdash (\varnothing \otimes \epsilon_L) \boxtimes (\varnothing \otimes \epsilon_R) \ \textsf{presplit}} \qquad \frac{\epsilon_L \equiv \top \text{ or } \epsilon_R \equiv \top}{\Phi \vdash (\varnothing \otimes \epsilon_L) \boxtimes (\varnothing \otimes \epsilon_R) \ \textsf{presplit}}$$

$$\frac{}{\mathfrak{c} \prec \Phi \vdash (\mathfrak{c} \otimes \epsilon_L) \boxtimes (\varnothing \otimes \epsilon_R) \ \textsf{presplit}} \qquad \frac{}{\mathfrak{c} \prec \Phi \vdash (\varnothing \otimes \epsilon_L) \boxtimes (\mathfrak{c} \otimes \epsilon_R) \ \textsf{presplit}}$$

$$\frac{\Phi \vdash s_L \boxtimes s_R \ \textsf{presplit}}{\mathfrak{c} \prec \Phi \vdash s_L \boxtimes s_R \ \textsf{presplit}}$$

$$\frac{\Phi_1 \vdash s_{1L} \boxtimes s_{1R} \ \textsf{presplit}}{\Phi_1, \Phi_2 \vdash s_{1L} \boxtimes s_{1R} \ \textsf{presplit}} \qquad \frac{\Phi_2 \vdash s_{2L} \boxtimes s_{2R} \ \textsf{presplit}}{\Phi_1, \Phi_2 \vdash s_{2L} \boxtimes s_{2R} \ \textsf{presplit}}$$

$$\frac{\begin{array}{c}\Phi_1 \vdash s_{1L} \boxtimes s_{1R} \ \textsf{presplit} \\ \Phi_2 \vdash s_{2L} \boxtimes s_{2R} \ \textsf{presplit}\end{array}}{\Phi_1 \otimes \Phi_2 \vdash (s_{1L} \otimes s_{2L}) \boxtimes (s_{1R} \otimes s_{2R}) \ \textsf{presplit}}$$

$$\frac{\Phi \vdash \mathsf{u}(s_L) \boxtimes \mathsf{u}(s_R) \ \textsf{presplit}}{\Phi \vdash s_L \boxtimes s_R \ \textsf{split}}$$

Figure 1.13: Rules for Splits.

$$\otimes\text{-FORM}\ \frac{\mathfrak{t} \mid \underline{\Gamma} \vdash A \text{ type} \qquad \mathfrak{t} \mid \underline{\Gamma}, \underline{x}^{\mathfrak{t}} : \underline{A} \vdash B \text{ type}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \textcircled{Σ}_{(\underline{x}:A)} B \text{ type}}$$

$$\otimes\text{-INTRO}\ \frac{\begin{array}{c}\mathfrak{t} \prec \Phi \vdash s_L \boxtimes s_R \text{ split} \\ \Phi^{s_L} \mid \Gamma^{s_L} \vdash a : A^{\ulcorner s_L \urcorner \leftrightarrow \mathfrak{t}} \qquad \Phi^{s_R} \mid \Gamma^{s_R} \vdash b : B[\underline{a}/\underline{x}]^{\ulcorner s_R \urcorner \leftrightarrow \mathfrak{t}}\end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash a \; {}_{s_L}\!\otimes_{s_R} b : \textcircled{Σ}_{(\underline{x}:A)} B}$$

$$\otimes\text{-ELIM}\ \frac{\begin{array}{c}\mathfrak{t} \prec \Phi \mid \Gamma, z^{\mathfrak{t}} : \textcircled{Σ}_{(\underline{x}:A)} B \vdash C \text{ type} \\ \mathfrak{t} \prec \Phi, \mathfrak{l} \otimes \mathfrak{r} \mid \Gamma, x^{\mathfrak{l}} : A^{\mathfrak{l} \leftrightarrow \mathfrak{t}}, y^{\mathfrak{r}} : B^{\mathfrak{r} \leftrightarrow \mathfrak{t}} \vdash c : C[x \; {}_{\mathfrak{l}}\!\otimes_{\mathfrak{r}} y/z] \\ \mathfrak{t} \prec \Phi \mid \Gamma \vdash s : \textcircled{Σ}_{(\underline{x}:A)} B\end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \text{let } x \; {}_{\mathfrak{l}}\!\otimes_{\mathfrak{r}} y = s \text{ in } c : C[s/z]}$$

$$\otimes\text{-BETA}\ \frac{\begin{array}{c}\mathfrak{t} \prec \Phi \mid \Gamma, z^{\mathfrak{t}} : \textcircled{Σ}_{(\underline{x}:A)} B \vdash C \text{ type} \\ \mathfrak{t} \prec \Phi, \mathfrak{l} \otimes \mathfrak{r} \mid \Gamma, x^{\mathfrak{l}} : A^{\mathfrak{l} \leftrightarrow \mathfrak{t}}, y^{\mathfrak{r}} : B^{\mathfrak{r} \leftrightarrow \mathfrak{t}} \vdash c : C[x \; {}_{\mathfrak{l}}\!\otimes_{\mathfrak{r}} y/z] \\ \mathfrak{t} \prec \Phi \vdash s_L \boxtimes s_R \text{ split} \\ \Phi^{s_L} \mid \Gamma^{s_L} \vdash a : A^{\ulcorner s_L \urcorner \leftrightarrow \mathfrak{t}} \qquad \Phi^{s_R} \mid \Gamma^{s_R} \vdash b : B[\underline{a}/\underline{x}]^{\ulcorner s_R \urcorner \leftrightarrow \mathfrak{t}}\end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash (\text{let } x \; {}_{\mathfrak{l}}\!\otimes_{\mathfrak{r}} y = a \; {}_{s_L}\!\otimes_{s_R} b \text{ in } c) \equiv c[s_L/\mathfrak{l} \otimes s_R/\mathfrak{r} \mid a/x, b/y] : C[a \otimes b/z]}$$

$$\text{SUBST-}\otimes\ \frac{\begin{array}{c}\mathfrak{t} \prec \Phi \vdash s_L \boxtimes s_R \text{ split} \\ \Phi^{s_L} \mid \Gamma^{s_L} \vdash a : A^{\ulcorner s_L \urcorner \leftrightarrow \mathfrak{t}} \qquad \Phi^{s_R} \mid \Gamma^{s_R} \vdash b : B[\underline{a}/\underline{x}]^{\ulcorner s_R \urcorner \leftrightarrow \mathfrak{t}} \\ \mathfrak{t} \prec \Phi, \mathfrak{l} \otimes \mathfrak{r} \mid \Gamma, x^{\mathfrak{l}} : A^{\mathfrak{l} \leftrightarrow \mathfrak{t}}, y^{\mathfrak{r}} : B^{\mathfrak{r} \leftrightarrow \mathfrak{t}} \vdash \mathcal{J}\end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \mathcal{J}[s_L/\mathfrak{l} \otimes s_R/\mathfrak{r} \mid a/x, b/y]}$$

Figure 1.14: Rules for the Tensor Type

- **Elimination:** Unchanged.
- **Computation:** If we perform $\otimes$-induction on a term that is actually already of the form $a \;{}_{s_L}\!\otimes_{s_R}\, b$, then the result is the term c with a and b substituted for x and y, and the slices s_L and s_R substituted for the colours $\mathfrak{l}$ and $\mathfrak{r}$, via the admissible SUBST-$\otimes$:

$$(\mathsf{let}\ x \;{}_{\mathfrak{l}}\!\otimes_{\mathfrak{r}}\, y = a \;{}_{s_L}\!\otimes_{s_R}\, b\ \mathsf{in}\ c) \equiv c[s_L/\mathfrak{l} \otimes s_R/\mathfrak{r} \mid a/x^{\mathfrak{l}}, b/y^{\mathfrak{r}}] : C[a \;{}_{s_L}\!\otimes_{s_R}\, b/z].$$

  On raw syntax, the operation on is implemented by plugging in s_L, s_R, a and b for the appropriate variables in c, just as in ordinary substitution.

  For the moment, we make a simplifying assumption about the terms and slices s_L, s_R, a and b. We assume that the top colour of the slice s_L does not appear in a and the top colour of the slice s_R does not appear in b. This can only happen in a term involving the hom type or a unitor, neither of which will appear in examples until later. If the top colour *does* appear then the substitution operation is very slightly more complicated, but we defer discussion of this until it can actually affect us, see Remark 1.4.3 and Remark 1.5.8.

Remark 1.3.19. The rules for splits still forbid us from forming a diagonal map $\underline{A} \to \underline{A} \otimes \underline{A}$, as given an $x : \underline{A}$, there is no split $s_L \boxtimes s_R$ split such that $x \;{}_{s_L}\!\otimes_{s_R}\, x$ is well-formed as any split assigns $\mathfrak{p}$ to exactly one side.

The exact choice of the rules for splits is a parameter to the type theory: if we allow a split $\mathfrak{p} \boxtimes \mathfrak{p}$ split, the above term would be well-formed. It is likely that our theory would then have semantics in 'relevance categories', where all objects have such diagonal morphisms. Unfortunately, not every property of $\otimes$ can be controlled this way. To devise a type theory with semantics in semicartesian monoidal categories (where the monoidal unit is equal to the terminal object), we would likely need a more permissive variable rule allowing us to use variables of any colour, something that cannot be controlled via the rules for splits.

Remark 1.3.20. When functoriality is applied directly to a $\otimes$-pair, the slices are preserved: we can calculate:

$$\begin{aligned}
\mathsf{func}(\underline{f} \otimes \underline{g})(x' \;{}_{s_L}\!\otimes_{s_R}\, y') &\equiv \mathsf{let}\ x^{\mathfrak{r}} \otimes y^{\mathfrak{b}} = x' \;{}_{s_L}\!\otimes_{s_R}\, y'\ \mathsf{in}\ \underline{f}(x) \;{}_{\mathfrak{r}}\!\otimes_{\mathfrak{b}}\, \underline{g}(y) \\
&\equiv \underline{f}(x) \;{}_{\mathfrak{r}}\!\otimes_{\mathfrak{b}}\, \underline{g}(y)[s_L/\mathfrak{r}, s_R/\mathfrak{b} \mid x'/x, y'/y] \\
&\equiv \underline{f}(x') \;{}_{s_L}\!\otimes_{s_R}\, \underline{g}(y')
\end{aligned}$$

We can now use the interaction between $\otimes$-INTRO and marking to show that $\natural$ is monoidal:

Proposition 1.3.21 ($\natural$ is monoidal)**.** *The canonical map*

$$\mathsf{mon} : \natural(\underline{A} \otimes \underline{B}) \to \natural\underline{A} \times \natural\underline{B}$$

is an equivalence, and more dependently, the canonical map

$$\mathsf{mon} : \natural \textstyle\bigcirc\!\!\!\!\Sigma_{(\underline{x}:\underline{A})} B(\underline{x}) \to \sum_{(u:\natural\underline{A})} \natural\underline{B}(\underline{u}_{\natural})$$

is an equivalence.

Proof. The canonical map mon was defined in Definition 1.3.11. To define an inverse, from $z : \sum_{(x:\natural\underline{A})} \natural\underline{B}(\underline{x}_\natural)$ we get $(\mathsf{pr}_1\underline{z})_\natural : \underline{A}$ and $(\mathsf{pr}_2\underline{z})_\natural : \underline{B}(\mathsf{pr}_1(\underline{z})_\natural)$. These terms are both marked, so we can use the split $1 \boxtimes 1$ to form the $\otimes$-pair

$$((\mathsf{pr}_1\underline{z})_\natural \;{}_1\!\otimes_1 (\mathsf{pr}_2\underline{z})_\natural)^\natural : \natural \mathop{\bigcirc\kern-0.75em\Sigma}_{(\underline{x}:\underline{A})}\underline{B}(\underline{x})$$

We now check that the round trips are the identity. Starting with $z : \sum_{(x:\natural\underline{A})} \natural\underline{B}(\underline{x}_\natural)$:

$$\begin{aligned}
&((\underline{\mathsf{pr}}_1((\mathsf{pr}_1\underline{z})_\natural \;{}_1\!\otimes_1 (\mathsf{pr}_2\underline{z})_\natural)^\natural{}_\natural)^\natural, (\underline{\mathsf{pr}}_2((\mathsf{pr}_1\underline{z})_\natural \;{}_1\!\otimes_1 (\mathsf{pr}_2\underline{z})_\natural)^\natural{}_\natural)^\natural)\\
&\equiv ((\underline{\mathsf{pr}}_1((\mathsf{pr}_1\underline{z})_\natural \;{}_1\!\otimes_1 (\mathsf{pr}_2\underline{z})_\natural)^\natural, (\underline{\mathsf{pr}}_2((\mathsf{pr}_1\underline{z})_\natural \;{}_1\!\otimes_1 (\mathsf{pr}_2\underline{z})_\natural)^\natural)\\
&\equiv ((\mathsf{pr}_1\underline{z})_\natural{}^\natural, (\mathsf{pr}_2\underline{z})_\natural{}^\natural)\\
&\equiv (\mathsf{pr}_1\underline{z}, \mathsf{pr}_2\underline{z})\\
&\equiv \underline{z}\\
&\equiv z
\end{aligned}$$

For the other direction, the composite is:

$$\begin{aligned}
&((\mathsf{pr}_1((\underline{\mathsf{pr}}_1\underline{w}_\natural)^\natural, (\underline{\mathsf{pr}}_2\underline{w}_\natural)^\natural))_\natural \;{}_1\!\otimes_1 (\mathsf{pr}_2((\underline{\mathsf{pr}}_1\underline{w}_\natural)^\natural, (\underline{\mathsf{pr}}_2\underline{w}_\natural)^\natural))_\natural)^\natural\\
&\equiv ((\underline{\mathsf{pr}}_1\underline{w}_\natural)^\natural{}_\natural \;{}_1\!\otimes_1 (\underline{\mathsf{pr}}_2\underline{w}_\natural)^\natural{}_\natural)^\natural\\
&\equiv (\underline{\mathsf{pr}}_1\underline{w}_\natural \;{}_1\!\otimes_1 \underline{\mathsf{pr}}_2\underline{w}_\natural)^\natural
\end{aligned}$$

By $\otimes$-induction we may assume $\underline{w}_\natural \equiv x \otimes y$, and then

$$(\underline{\mathsf{pr}}_1(x \otimes y) \;{}_1\!\otimes_1 \underline{\mathsf{pr}}_2(x \otimes y))^\natural \equiv (\underline{x} \;{}_1\!\otimes_1 \underline{y})^\natural \equiv \underline{x \otimes y}^\natural \equiv \underline{w}_\natural{}^\natural \equiv w$$

□

There is a subtle point that we have managed to avoid so far. Consider the context

$$\mathfrak{p} \prec \mathfrak{r} \otimes \mathfrak{b} \mid x^{\mathfrak{r}} : \underline{A}, x^{\mathfrak{b}} : \underline{B}.$$

The term $\underline{x} \otimes \underline{y}$ with the slices suppressed is ambiguous: it could mean either of

$$\underline{x} \;{}_{\mathfrak{r}'.}\!\otimes_{\mathfrak{b}'.} \underline{y} \qquad \text{or} \qquad \underline{x} \;{}_{\mathfrak{r}}\!\otimes_{\mathfrak{b}} \underline{y}$$

and these are not definitionally equal.

This issue appears when we try to show that $\natural(\underline{A} \otimes \underline{B}) \simeq \natural\underline{A} \otimes \natural\underline{B}$. There are obvious maps in both directions, using $\otimes$-induction:

$$\begin{aligned}
(\underline{x} \;{}_1\!\otimes_1 \underline{y})^\natural &\mapsto \underline{x}^\natural \;{}_1\!\otimes_1 \underline{y}^\natural\\
\underline{x}^\natural \;{}_1\!\otimes_1 \underline{y}^\natural &\mapsto (\underline{x} \;{}_1\!\otimes_1 \underline{y})^\natural
\end{aligned}$$

The round-trip on $\natural(\underline{A} \otimes \underline{B})$ is the identity, because any space is definitionally marked. But the round-trip the other way yields $\underline{x} \;{}_1\!\otimes_1 \underline{y} : \natural\underline{A} \otimes \natural\underline{B}$, and we need this to equal $\underline{x} \;{}_{\mathfrak{r}}\!\otimes_{\mathfrak{b}} \underline{y}$. The equality follows from the apparently simpler equivalence $1 \otimes 1 \simeq 1$:

Proposition 1.3.22. *The canonical map* $\natural(\underline{A} \otimes \underline{B}) \to \natural\underline{A} \otimes \natural\underline{B}$ *is an equivalence for every* $\underline{A}$ *and* $\underline{B}$ *if and only if* $1 \otimes 1 \simeq 1$.

Proof. The former clearly implies the latter, taking $\underline{A} \equiv \underline{B} \equiv 1$: we see

$$(1 \otimes 1) \simeq (\natural 1 \otimes \natural 1) \simeq \natural(1 \otimes 1) \simeq (\natural 1 \times \natural 1) \simeq (1 \times 1) \simeq 1.$$

In the other direction,

$$\begin{aligned} \natural\underline{A} \otimes \natural\underline{B} &\simeq (\natural\underline{A} \times 1) \otimes (\natural\underline{B} \times 1) \\ &\simeq \natural\underline{A} \times \natural\underline{B} \times (1 \otimes 1) && \text{(Proposition 1.3.12)} \\ &\simeq \natural\underline{A} \times \natural\underline{B} \\ &\simeq \natural(\underline{A} \otimes \underline{B}) && \text{(Proposition 1.3.21)} \end{aligned}$$

□

To show that $1 \otimes 1 \simeq 1$, we need to use the monoidal unit. We discuss the monoidal in Section 1.4, but all we need now is that there is a pointed type $\mathbb{S}$ and an equivalence $1 \otimes \mathbb{S} \simeq 1$.

Proposition 1.3.23. $1 \otimes 1 \simeq 1$

Proof. Functoriality of $1 \otimes -$ on the composite

$$\begin{tikzcd} 1 \arrow[r] \arrow[rr, bend left, "\mathsf{id}"] & \mathbb{S} \arrow[r] & 1 \end{tikzcd}$$

gives

$$\begin{tikzcd} 1 \otimes 1 \arrow[r] \arrow[rr, bend left, "\mathsf{id}"] & 1 \otimes \mathbb{S} \arrow[r] & 1 \otimes 1 \end{tikzcd}$$

with the middle object equivalent to 1. The composite the other way is also the identity, as all maps $(1 \otimes \mathbb{S}) \to (1 \otimes \mathbb{S})$ are equal to the identity. □

As a consequence, we have the following meta-principle:

Meta-Lemma 1.3.24. *For dull* $\underline{x} : \underline{A}$ *and* $\underline{y} : \underline{B}$ *and splits* $s_L \boxtimes s_R$ split *and* $t_L \boxtimes t_R$ split, *there is an equality*

$$\underline{x} \;{}_{s_L}\!\otimes_{s_R} \underline{y} = \underline{x} \;{}_{t_L}\!\otimes_{t_R} \underline{y}$$

This is only a meta-principle, because we cannot quantify over slices or splits internally.

Proof. Follows because $\underline{x}^\natural \;{}_{s_L}\!\otimes_{s_R} \underline{y}^\natural$ and $\underline{x}^\natural \;{}_{t_L}\!\otimes_{t_R} \underline{y}^\natural$ have the same image under the equivalence $\natural\underline{A} \otimes \natural\underline{B} \to \natural(\underline{A} \otimes \underline{B})$, and so there is a path

$$\underline{x}^\natural \;{}_{s_L}\!\otimes_{s_R} \underline{y}^\natural = \underline{x}^\natural \;{}_{t_L}\!\otimes_{t_R} \underline{y}^\natural.$$

By functoriality we can apply $\natural$-ELIM to each factor to get the desired path, as functoriality on a tensor-pair preserves the slices that are used (Remark 1.3.20). □

This trick only applies to dull terms, because we have to pass through the $\natural$-types.

Remark 1.3.25. A possible improvement to our type theory would be to make the splits proof-irrelevant, so that two terms that differ only in the choice of splits are definitionally equal. This would be convenient, but would make the metatheory more complicated: all constructions would need to be invariant under the choice of split.

We have seen that in some cases there is a path between terms that only differ in the choice of split. There is then a concern that declaring the two endpoints definitionally equal might not make that path equal to refl, investigating whether this can ever happen is left to future work.

1.3.4 Stronger Induction Principles

Unfortunately, we have more-or-less exhausted the facts about $\otimes$-types that are provable directly from the binary eliminator given above. The issue is that, with the rules we have so far, we can only apply the induction principle to terms of the top colour. We run into this limitation quickly when dealing with nested tensors. Suppose we are trying to define an associativity map, and we have a term $p : \underline{A} \otimes (\underline{B} \otimes \underline{C})$. Using $\otimes$-induction on p gives $x^{\mathfrak{r}} : \underline{A}$ and $y^{\mathfrak{b}} : \underline{B} \otimes \underline{C}$, but now we are stuck! We would like to apply $\otimes$-induction again to y, but we cannot. The term y on its own is not well-formed in the body of this induction on p: it is not of colour $\mathfrak{p}$.

Later we will introduce hom types, the right adjoint to tensor types. It should not surprise us that, in the presence of these hom types, more general induction principles are derivable. For associativity, we need the 'ternary' eliminators given in Figure 1.15, and we will show later in Section 1.6 that these eliminators are indeed derivable using the hom type.

Unfortunately, the presence of this hom type is *still* not sufficient to derive all the induction principles that we would expect. The issue is the following: in the typical arguments for deriving these induction principles, the right adjoint types (here Π- and $\multimap$-types) are used to move the variables to the right of the target 'out of the way', by placing them into the motive of the induction. With a bunched structure however, this cannot always be done. To try and derive an induction principle for $A \otimes (B \times (C \otimes D))$, say, we cannot move D into the motive on its own, as $- \otimes (- \times (- \otimes D)$ does not have a right adjoint.[3]

Rather than define a custom eliminator for every new situation we encounter, in Section 1.6 we will describe an inductive collection of 'patterns', and give a general rule for matching against arbitrary pattern. We will wait until then to properly define patterns—several new judgement forms are needed.

Remark 1.3.26. A similar issue is encountered in Spatial Type Theory [Shu18, §5], where $\flat$-induction can only be applied when the target is an ordinary variable, not when the target is a 'crisp' variable. A 'crisp' $\flat$-induction principle is derivable, using the fact that $\flat$ has a right adjoint. This trick works very generally: such 'crisp' induction principles are derivable in Multimodal Type Theory [GKNB20, §6.2] for any modality with a right adjoint.

We can now clear up associativity. As promised earlier, we will begin to leave the splits off the syntax of $\otimes$-INTRO when the split used is unambiguous.

[3] In this particular case we can use a hom type to move A into the motive, but this forces the ambient context to be only used marked, and so is not fully general.

$$\otimes\text{-ELIM-TRIPLE-LEFT}\;\dfrac{\begin{array}{c}\mathfrak{t} \mid \underline{\Gamma}, w^{\mathfrak{t}} : \underline{\textcircled{Σ}}_{(\underline{p}:\underline{\textcircled{Σ}}_{(\underline{x}:A)}\, B)}(\mathsf{let}\ x \otimes y = \underline{p}\ \mathsf{in}\ \mathsf{C}) \vdash D\ \mathsf{type} \\ \mathfrak{t} \prec (\mathfrak{l} \prec \mathfrak{l}' \otimes \mathfrak{r}') \otimes \mathfrak{r} \mid \underline{\Gamma}, x^{\mathfrak{l}'} : A, y^{\mathfrak{r}'} : B, z^{\mathfrak{r}} : \mathsf{C} \vdash d : D[(x\ {}_{\mathfrak{l}'}{\otimes}_{\mathfrak{r}'}\ y)\ {}_{\mathfrak{l}}{\otimes}_{\mathfrak{r}}\ z/w] \\ \mathfrak{t} \prec \Phi \mid \Gamma \vdash s : \underline{\textcircled{Σ}}_{(\underline{p}:\underline{\textcircled{Σ}}_{(\underline{x}:A)}\, B)}(\mathsf{let}\ x \otimes y = \underline{p}\ \mathsf{in}\ \mathsf{C})\end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \mathsf{let}\ (x\ {}_{\mathfrak{l}'}{\otimes}_{\mathfrak{r}'}\ y)\ {}_{\mathfrak{l}}{\otimes}_{\mathfrak{r}}\ z = s\ \mathsf{in}\ d : D[s/w]}$$

$$\begin{aligned}&(\mathsf{let}\ (x\ {}_{\mathfrak{l}'}{\otimes}_{\mathfrak{r}'}\ y)\ {}_{\mathfrak{l}}{\otimes}_{\mathfrak{r}}\ z = (a\ {}_{s_{LL}}{\otimes}_{s_{LR}}\ b)\ {}_{s_L}{\otimes}_{s_R}\ c\ \mathsf{in}\ d)\\ &\equiv d[(s_L/\mathfrak{l} \prec s_{LL}/\mathfrak{l}' \otimes s_{LR}/\mathfrak{r}') \otimes s_R/\mathfrak{r} \mid a/x^{\mathfrak{l}'}, b/y^{\mathfrak{r}'}, c/z^{\mathfrak{r}}]\end{aligned}$$

$$\otimes\text{-ELIM-TRIPLE-RIGHT}\;\dfrac{\begin{array}{c}\mathfrak{t} \mid \underline{\Gamma}, w^{\mathfrak{t}} : \underline{\textcircled{Σ}}_{(\underline{x}:\underline{A})}\underline{\textcircled{Σ}}_{(\underline{y}:\underline{B})}\underline{\mathsf{C}} \vdash D\ \mathsf{type} \\ \mathfrak{t} \prec \mathfrak{l} \otimes (\mathfrak{r} \prec \mathfrak{l}' \otimes \mathfrak{r}') \mid \underline{\Gamma}, x^{\mathfrak{l}} : A, y^{\mathfrak{l}'} : B, z^{\mathfrak{r}'} : \mathsf{C} \vdash d : D[x\ {}_{\mathfrak{l}}{\otimes}_{\mathfrak{r}}\ (y\ {}_{\mathfrak{l}'}{\otimes}_{\mathfrak{r}'}\ z)/w] \\ \mathfrak{t} \prec \Phi \mid \Gamma \vdash s : \underline{\textcircled{Σ}}_{(\underline{x}:\underline{A})}\underline{\textcircled{Σ}}_{(\underline{y}:\underline{B})}\underline{\mathsf{C}}\end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \mathsf{let}\ x\ {}_{\mathfrak{l}}{\otimes}_{\mathfrak{r}}\ (y\ {}_{\mathfrak{l}'}{\otimes}_{\mathfrak{r}'}\ z) = s\ \mathsf{in}\ d : D[s/w]}$$

$$\begin{aligned}&(\mathsf{let}\ x\ {}_{\mathfrak{l}}{\otimes}_{\mathfrak{r}}\ (y\ {}_{\mathfrak{l}'}{\otimes}_{\mathfrak{r}'}\ z) = a\ {}_{s_L}{\otimes}_{s_R}\ (b\ {}_{s_{RL}}{\otimes}_{s_{RR}}\ c)\ \mathsf{in}\ d)\\ &\equiv d[s_L/\mathfrak{l} \otimes (s_R/\mathfrak{r} \prec s_{RL}/\mathfrak{l}' \otimes s_{RR}/\mathfrak{r}') \mid a/x^{\mathfrak{r}}, b/y^{\mathfrak{l}'}, c/z^{\mathfrak{r}'}]\end{aligned}$$

$$\otimes\text{-Id-ELIM}\;\dfrac{\begin{array}{c}\mathfrak{t} \mid \underline{\Gamma}, w^{\mathfrak{t}} : \left(\sum_{(x:A)}\sum_{(x':A)} x = x'\right) \otimes \left(\sum_{(y:B)}\sum_{(y':B)} y = y'\right) \vdash D\ \mathsf{type} \\ \mathfrak{t} \prec (\mathfrak{l} \otimes \mathfrak{r}) \mid \underline{\Gamma}, x^{\mathfrak{l}} : A, y^{\mathfrak{r}} : B \vdash d : D[(x, x, \mathsf{refl}_x)\ {}_{\mathfrak{l}}{\otimes}_{\mathfrak{r}}\ (y, y, \mathsf{refl}_y)/w] \\ \mathfrak{t} \prec \Phi \mid \Gamma \vdash s : \left(\sum_{(x:A)}\sum_{(x':A)} x = x'\right) \otimes \left(\sum_{(y:B)}\sum_{(y':B)} y = y'\right)\end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \mathsf{let}\ (x, x, \mathsf{refl}_x)\ {}_{\mathfrak{l}}{\otimes}_{\mathfrak{r}}\ (y, y, \mathsf{refl}_y) = s\ \mathsf{in}\ d : D[s/w]}$$

$$\begin{aligned}&(\mathsf{let}\ (x, x, \mathsf{refl}_x)\ {}_{\mathfrak{l}}{\otimes}_{\mathfrak{r}}\ (y, y, \mathsf{refl}_y) = (a, a, \mathsf{refl}_a)\ {}_{s_L}{\otimes}_{s_R}\ (b, b, \mathsf{refl}_b)\ \mathsf{in}\ d)\\ &\equiv d[s_L/\mathfrak{l} \otimes s_R/\mathfrak{r} \mid a/x, b/y]\end{aligned}$$

Figure 1.15: Derivable Eliminators

Proposition 1.3.27 (Associativity of $\otimes$)**.** *For dull types $\underline{A}$, $\underline{B}$ and $\underline{C}$,*

$$\underline{A} \otimes (\underline{B} \otimes \underline{C}) \simeq (\underline{A} \otimes \underline{B}) \otimes \underline{C}.$$

More generally, if $\underline{A} : \mathcal{U}$, $\underline{B} : \underline{A} \to \mathcal{U}$ and $\underline{C} : \prod_{(\underline{x}:\underline{A})} \underline{B}(\underline{x}) \to \mathcal{U}$,

$$\textstyle\bigotimes_{(\underline{x}:\underline{A})} \bigotimes_{(\underline{y}:\underline{B}(\underline{x}))} \underline{C}(\underline{x})(\underline{y}) \simeq \bigotimes_{(\underline{v}:\bigotimes_{(\underline{x}:\underline{A})} \underline{B}(\underline{x}))} \mathsf{let}\ x \otimes y = \underline{v}\ \mathsf{in}\ \underline{C}(\underline{x})(\underline{y}).$$

Proof. We prove the simple version, the maps for the dependent version are defined the same way. Suppose the total colour is brown, which comprises red, blue and yellow.

The maps are defined in the expected way using the eliminators of Figure 1.15.

$$\mathsf{assoc}_{\underline{A},\underline{B},\underline{C}}(x \otimes (y \otimes z)) :\equiv (x \otimes y) \otimes z$$
$$\mathsf{associnv}_{\underline{A},\underline{B},\underline{C}}((x \otimes y) \otimes z) :\equiv x \otimes (y \otimes z)$$

The proof that they are inverses uses the same eliminators. By function extensionality we have to give a function

$$\textstyle\prod_{(t:\underline{A}\otimes(\underline{B}\otimes\underline{C}))} \mathsf{associnv}_{\underline{A},\underline{B},\underline{C}}(\mathsf{assoc}_{\underline{A},\underline{B},\underline{C}}(t)) = t$$

If we apply the triple-induction on t as $x \otimes (y \otimes z)$, then we can calculate

$$\mathsf{associnv}_{\underline{A},\underline{B},\underline{C}}(\mathsf{assoc}_{\underline{A},\underline{B},\underline{C}}(x \otimes (y \otimes z))) \equiv \mathsf{associnv}_{\underline{A},\underline{B},\underline{C}}((x \otimes y) \otimes z) \equiv x \otimes (y \otimes z)$$

so we can take $\mathsf{refl}_{x\otimes(y\otimes z)}$ as our proof term:

$$(\lambda.(x \otimes (y \otimes z)).\mathsf{refl}_{x\otimes(y\otimes z)}) : \textstyle\prod_{(t:\underline{A}\otimes(\underline{B}\otimes\underline{C}))} \mathsf{associnv}_{\underline{A},\underline{B},\underline{C}}(\mathsf{assoc}_{\underline{A},\underline{B},\underline{C}}(t)) = t$$

The other composite is similar, instead using the eliminator for $(x \otimes y) \otimes z$. $\square$

Paths in Σ types are characterised by a pair of paths in the two components, but the same is not true for paths in $\otimes$-types, the type of such paths is not equal to the 'tensor of paths in the two components': there is no way to make sense of that statement in a way that satisfies the linearity restrictions. Given two points $p, p' : A \otimes B$, performing $\otimes$-induction on p gives fresh colours and variables $\mathfrak{r} \otimes \mathfrak{b} \mid x^{\mathfrak{r}} : A, y^{\mathfrak{b}} : B$. Performing a second $\otimes$-induction on p' gives fresh colours and variables $\mathfrak{r}' \otimes \mathfrak{b}' \mid x'^{\mathfrak{r}'} : A, y'^{\mathfrak{b}'} : B$. But now we can't form the type $x = x'$, because the colours of the two terms do not match. There is therefore no hope of a characterisation of the type $p = p'$ for generic $p, p' : A \otimes B$.

We can however show that, like everything else in type theory, $\otimes$-INTRO respects equality. This will require another specialised induction principle, also displayed in Figure 1.15, because path-induction may only be applied to a term of the top colour. We discuss this restriction further in Remark 1.6.10.

Proposition 1.3.28 ($\otimes$ respects equality)**.** *Suppose $a, a' : \underline{A}$ with $p : a = a'$, and $b, b' : \underline{B}$ with $q : b = b'$. Then there is a path $(a \otimes b) = (a' \otimes b')$.*

In syntax, we are inhabiting the type

$$\prod_{(w:\underline{T})} \mathsf{let}\ (a, a', p) \otimes (b, b', q) = w\ \mathsf{in}\ (a \otimes b) = (a' \otimes b')$$

where $\underline{T}$ is the type

$$\underline{T} :\equiv \left(\Sigma_{(a:\underline{A})}\Sigma_{(a':\underline{A})} a = a'\right) \otimes \left(\Sigma_{(b:\underline{B})}\Sigma_{(b':\underline{B})} b = b'\right).$$

Proof. Follows by applying the above eliminator, which reduces the goal type to $(a \otimes b) = (a \otimes b)$.

$$f((a, a, \mathsf{refl}_a) \otimes (b, b, \mathsf{refl}_b)) :\equiv \mathsf{refl}_{a \otimes b}$$

□

Proposition 1.3.29. *If $\underline{f} : \underline{A} \to \underline{A}'$ and $\underline{g} : \underline{B} \to \underline{B}'$ are equivalences, then $\underline{f} \otimes \underline{g} : \underline{A} \otimes \underline{B} \to \underline{A}' \otimes \underline{B}'$ is an equivalence.*

Proof. Suppose we have left inverses $i : \underline{A}' \to \underline{A}$ and $j : \underline{B}' \to \underline{B}$, so that there are homotopies $p : i \circ \underline{f} \sim \mathsf{id}_{\underline{A}}$ and $q : j \circ \underline{g} \sim \mathsf{id}_{\underline{B}}$. Then also $\underline{p} : \underline{i} \circ \underline{f} \sim \mathsf{id}_{\underline{A}}$ and $\underline{q} : \underline{j} \circ \underline{g} \sim \mathsf{id}_{\underline{B}}$, so $\underline{i}$ and $\underline{j}$ are also left inverses to $\underline{f}$ and $\underline{g}$.

Given a $p : \underline{A} \otimes \underline{B}$ our goal is to prove $(\underline{i} \otimes \underline{j}) \circ (\underline{f} \otimes \underline{g})(p) = p$. By $\otimes$-induction, it is enough to prove $(\underline{i} \otimes \underline{j}) \circ (\underline{f} \otimes \underline{g})(x \otimes y) = x \otimes y$, and by functoriality this type is $\underline{i}(\underline{f}(x)) \otimes \underline{j}(\underline{g}(y)) = x \otimes y$. We have such a path by the previous Proposition applied to the paths $\underline{i}(\underline{f}(x)) = x$ and $\underline{j}(\underline{g}(y)) = y$. □

1.4 Unit

The monoidal unit type $\mathbb{S}$ is, roughly, a nullary version of the $\otimes$-type. We already have a palette constructor for a nullary monoidal product, $\varnothing$ palette, but this is not yet reified as a type.

We can in fact already use the unitors in one direction via the split judgement: if the palette is $\mathfrak{p}$, then we can form a term $a\ {}_{\mathfrak{p}}\!\otimes_{\mathfrak{t}.\prec\varnothing}\ b$, where b is a term in palette $\mathfrak{t} \prec \varnothing$.

The natural thing to try here is a simple 'nullary' split as we did at the beginning of Section 1.3 for binary splits:

$$\frac{}{\mathfrak{t} \prec \Phi, \varnothing, \Phi' \vdash \mathsf{unit}} \qquad \textsc{$\mathbb{S}$-intro?}\ \frac{\mathfrak{t} \prec \Phi \vdash \mathsf{unit}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \mathord{\maltese} : \mathbb{S}}$$

Unfortunately this is not enough, because of the following issue. In the presence of a cartesian weakening rule, the term

$$\textsc{pal-wk}\ \frac{\textsc{$\mathbb{S}$-intro?}\ \dfrac{\mathfrak{t} \prec \varnothing \vdash \mathsf{unit}}{\mathfrak{t} \prec \varnothing \vdash \mathord{\maltese} : \mathbb{S}}}{\mathfrak{t} \prec \varnothing, \varnothing \vdash \mathord{\maltese} : \mathbb{S}}$$

yields a term that is ambiguous in a material way: there should be two non-equal projections $\mathbb{S} \times \mathbb{S} \to \mathbb{S}$.

$$\textsc{pal-sphere-named}\ \frac{}{\varnothing_i\ \mathsf{palette}}$$

$$\frac{}{\Phi \vdash (\mathfrak{t}. \prec \varnothing_{i.})\ \mathsf{slice}} \qquad \begin{aligned} \Phi^{\mathfrak{t}.\prec\varnothing_{i.}} &:\equiv \mathfrak{t} \prec \varnothing_i \\ \mathsf{u}(\mathfrak{t}. \prec \varnothing_{i.}) &:\equiv \varnothing \\ (\mathfrak{t}. \prec \varnothing_{i.})[\![s/\mathfrak{c}]\!] &:\equiv \mathfrak{t}. \prec \varnothing_{i.} \end{aligned} \qquad \textsc{unit-zero}\ \frac{}{\Phi \vdash 0\ \mathsf{unit}}$$

$$\textsc{unit-here}\ \frac{}{\varnothing_i \vdash i\ \mathsf{unit}} \qquad \textsc{unit-sub}\ \frac{\Phi \vdash i\ \mathsf{unit}}{(\mathfrak{c} \prec \Phi) \vdash i\ \mathsf{unit}}$$

$$\textsc{unit-left}\ \frac{\Phi_1 \vdash i\ \mathsf{unit}}{\Phi_1, \Phi_2 \vdash i\ \mathsf{unit}} \qquad \textsc{unit-right}\ \frac{\Phi_2 \vdash i\ \mathsf{unit}}{\Phi_1, \Phi_2 \vdash i\ \mathsf{unit}}$$

Figure 1.16: Rules for Palette Units

Remark 1.4.1. The $\alpha\lambda$-calculus [Pym02, §2.3] uses a rule of this kind, and so suffers from this issue. This contradicts soundness for the $\alpha\lambda$-calculus in 'cartesian doubly-closed categories' [Pym02, Proposition 3.8]. Often $\otimes$ is assumed to be affine when working in the $\alpha\lambda$-calculus [OHe03, §3.3], meaning $\mathbb{S} \equiv 1$, and then there is no issue: $\mathbb{S}$ is then also the unit of the context comma and all maps into $\mathbb{S}$ are equal.

We therefore need some way to refer to different instances of the monoidal unit that are combined with a $\times$. We introduce a second, labelled version of the unit palette constructor $\varnothing_i$ palette, where i is a 'unit label'.

1.4.1 Palette Units

The new rules are show in Figure 1.16, also introducing a new judgement $\Phi \vdash i$ unit that picks out a unit label or the special label 0. The latter case corresponds to the morphism $1 \to \mathbb{S}$ picking out the basepoint: such a map into the sphere is always available regardless of the state of the palette.

We add a new kind of palette slice that creates this labelled unit palette. We already had a slice $\Phi \vdash \mathfrak{t}. \prec \varnothing$ slice for any palette Φ, so we add the option to name this new judgemental unit: $\Phi \vdash \mathfrak{t}. \prec \varnothing_{i.}$ slice. The various operations that use slices have to be extended to include this new case, as shown in the figure. Substitution into the slice $\mathfrak{t}. \prec \varnothing_{i.}$ has no effect, because both $\mathfrak{t}$ and i are freshly bound names.

We do not allow an instance of $\Phi \vdash i$ unit to combine multiple $\varnothing_i$ together, and we do not allow the chosen unit to appear below an instance of the $\otimes$ palette former; it must appear at the top level.

1.4.2 Type Former

The rules for the monoidal unit $\mathbb{S}$, shown in Figure 1.17, now internalise the judgemental unit $\varnothing_i$.

- **Formation:** There is a type $\mathbb{S}$, well-formed in any context.

$$\textsc{$\mathbb{S}$-form}\ \frac{}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \mathbb{S}\ \mathsf{type}}$$

$$\textsc{$\mathbb{S}$-intro}\ \frac{\mathfrak{t} \prec \Phi \vdash i\ \mathsf{unit}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash ⌑_i : \mathbb{S}} \qquad \textsc{$\mathbb{S}$-elim}\ \frac{\begin{array}{c}\mathfrak{t} \prec \Phi \mid \Gamma, z^{\mathfrak{t}} : \mathbb{S} \vdash C\ \mathsf{type} \\ \mathfrak{t} \prec \Phi, \varnothing_i \mid \Gamma \vdash c : C[⌑_i/z] \\ \mathfrak{t} \prec \Phi \mid \Gamma \vdash s : \mathbb{S}\end{array}}{\Phi \mid \Gamma \vdash \mathsf{let}\ ⌑_i = s\ \mathsf{in}\ c : C[s/z]}$$

$$\textsc{$\mathbb{S}$-beta}\ \frac{\begin{array}{c}\mathfrak{t} \prec \Phi \mid \Gamma, z^{\mathfrak{t}} : \mathbb{S} \vdash C\ \mathsf{type} \\ \mathfrak{t} \prec \Phi, \varnothing_i \mid \Gamma \vdash c : C[⌑_i/z] \\ \mathfrak{t} \prec \Phi \vdash j\ \mathsf{unit}\end{array}}{\Phi \mid \Gamma \vdash (\mathsf{let}\ ⌑_i = ⌑_j\ \mathsf{in}\ c) \equiv c[i/j] : C[⌑_j/z]}$$

Figure 1.17: Rules for the Monoidal Unit Type

- **Introduction:** For any judgemental unit i, there is a term $⌑_i$. If $i \equiv 0$ then we write $\underline{⌑}$, to align with the syntax for marked variables.
- **Elimination:** Whenever we have a term $s : \mathbb{S}$, we may assume it is of the form $⌑_i$ for a fresh judgemental unit label $\varnothing_i$ in the palette. We call this $\mathbb{S}$-*induction*. Syntactically, if a target type C depends on a variable $z^{\mathfrak{t}} : \mathbb{S}$, and $c : C[⌑_i/z]$ is a term that uses a unit label i, then there is an induced term

  $$\mathsf{let}\ ⌑_i = s\ \mathsf{in}\ c : C[s/z]$$

- **Computation:** If we perform $\mathbb{S}$-induction on a term that is of the form $\mathbb{S}_j$, then the result is c with j substituted in accordingly:

  $$(\mathsf{let}\ ⌑_i = ⌑_j\ \mathsf{in}\ c) \equiv c[j/i] : C[⌑_j/z]$$

  This $c[j/i]$ operation is yet another form of substitution, that simply replaces the label j for i everywhere that it appears.

To define maps *into* $\underline{A} \otimes \mathbb{S}$, we can use the new labelled unit palette slice. If the top colour is $\mathfrak{p}$, then we have the split $\mathfrak{p} \boxtimes \mathfrak{y}. \prec \varnothing_{i.}$. We call such splits unitor splits.

Definition 1.4.2. For any dull type $\underline{A}$, define $\mathsf{unitlinv}_{\underline{A}} : \underline{A} \to \mathbb{S} \otimes \underline{A}$ and $\mathsf{unitrinv}_{\underline{A}} : \underline{A} \to \underline{A} \otimes \mathbb{S}$ by

$$\begin{aligned}\mathsf{unitlinv}_{\underline{A}}(a) &:\equiv ⌑_i\ {}_{\mathfrak{y}.\prec\varnothing_{i.}}\!\otimes_{\mathfrak{p}}\ a \\ \mathsf{unitrinv}_{\underline{A}}(a) &:\equiv a\ {}_{\mathfrak{p}}\!\otimes_{\mathfrak{y}.\prec\varnothing_{i.}}\ ⌑_i\end{aligned}$$

On the left side (right side resp.) of $\otimes$-INTRO, we immediately apply $\mathbb{S}$-INTRO, but as usual, we have the option of putting any term in palette $\mathfrak{y} \prec \varnothing_i$ there.

Remark 1.4.3. A use of a unitor split is the second way that the name of the top colour, in this case $\mathfrak{p}$, can appear in a term. If we have to substitute the above term $a\ {}_{\mathfrak{p}}\!\otimes_{\mathfrak{y}.\prec\varnothing_{i}.}\ ¤_{\mathfrak{y}}$ into a marked variable where the top colour is $\mathfrak{g}$, say, the result would be $\underline{a}\ {}_{\mathfrak{g}}\!\otimes_{\mathfrak{y}.\prec\varnothing_{i}.}\ ¤_{\mathfrak{i}}$: the top colour $\mathfrak{p}$ has been replaced with $\mathfrak{g}$.

As with normal splits, when working informally we will omit the colour labels from a unitor split if they are not used in the term, writing simply $\mathsf{unitlinv}_{\underline{A}}(a) :\equiv ¤_{i}\ {}_{\varnothing_i}\!\otimes\ a$, for example.

1.4.3 Unitors

Similarly to the situation for the $\otimes$-type, the $\mathbb{S}$-ELIM rule is not strong enough to derive all the induction principles for $\mathbb{S}$ that we might like. In particular, we are not aware of any way to derive the unitors $\mathsf{unitl}_{\underline{A}} : \mathbb{S} \otimes \underline{A} \to \underline{A}$ and $\mathsf{unitr}_{\underline{A}} : \underline{A} \otimes \mathbb{S} \to \underline{A}$ from what we have outlined above. We add these missing induction principles to our pattern matching construct Section 1.6, and preview them in Figure 1.18.

Proposition 1.4.4. $\natural\mathbb{S} \simeq 1$

Proof. We choose $\underline{¤}^{\natural} : \natural\mathbb{S}$ as the centre of contraction. We define a null-homotopy $\prod_{(x:\natural\mathbb{S})} x = \underline{¤}^{\natural}$ by performing induction on x as $\underline{¤}^{\natural}$, and then supplying $\mathsf{refl}_{\underline{¤}^{\natural}}$. ☐

This clears up the missing piece in Proposition 1.3.23.

The induction principle for the left unitor says that any term $p : \mathbb{S} \otimes \underline{A}$ may be assumed to be of the form $p \equiv ¤_{i}\ {}_{\mathfrak{y}.\prec\varnothing_{i}.}\!\otimes_{\mathfrak{p}}\ a$, where $a : \underline{A}$. This new $a : \underline{A}$ is a variable of the same colour as p, and so is immediately available to be used. Doing induction of this kind does *not* bind a new unit label i: this unit label only 'exists' on the left side of that $\otimes$.

Definition 1.4.5. For any dull type $\underline{A}$, we can define $\mathsf{unitl}_{\underline{A}} : \mathbb{S} \otimes \underline{A} \to \underline{A}$ and $\mathsf{unitr}_{\underline{A}} : \underline{A} \otimes \mathbb{S} \to \underline{A}$ by:

$$\begin{aligned}\mathsf{unitl}_{\underline{A}}(p) &:\equiv \mathsf{let}\ ¤_{i}\ {}_{\mathfrak{y}.\prec\varnothing_{i}.}\!\otimes_{\mathfrak{p}}\ x = p\ \mathsf{in}\ x\\ \mathsf{unitr}_{\underline{A}}(p) &:\equiv \mathsf{let}\ x\ {}_{\mathfrak{p}}\!\otimes_{\mathfrak{y}.\prec\varnothing_{i}.}\ ¤_{i} = p\ \mathsf{in}\ x\end{aligned}$$

Using an almost identical proof to Proposition 1.3.6, we find

Proposition 1.4.6 (Uniqueness principle for the unitor)**.** *Suppose* $\underline{A}$ *is a dull type and* $C : \mathbb{S} \otimes \underline{A} \to \mathcal{U}$ *is a type family and* $f : \prod_{(p:\mathbb{S}\otimes\underline{A})} C(p)$*. For any* $p : \mathbb{S} \otimes \underline{A}$ *we have*

$$(\mathsf{let}\ ¤_{i}\ {}_{\mathfrak{y}.\prec\varnothing_{i}.}\!\otimes_{\mathfrak{p}}\ x = p\ \mathsf{in}\ f(¤_{i}\ {}_{\mathfrak{y}.\prec\varnothing_{i}.}\!\otimes_{\mathfrak{p}}\ x)) = f(p)$$

Combining this with Proposition 1.3.6, we immediately have

Corollary 1.4.7. *Any use of* $\otimes$*-induction on* $\mathbb{S} \otimes \underline{A}$ *may be replaced with a use of unitor induction:*

$$(\mathsf{let}\ s \otimes y = p\ \mathsf{in}\ f(s \otimes y)) = (\mathsf{let}\ ¤_{i}\ {}_{\mathfrak{y}.\prec\varnothing_{i}.}\!\otimes_{\mathfrak{p}}\ x = p\ \mathsf{in}\ f(¤_{i}\ {}_{\mathfrak{y}.\prec\varnothing_{i}.}\!\otimes_{\mathfrak{p}}\ x))$$

Lemma 1.4.8. $\mathsf{unitl}_{\underline{A}} : \mathbb{S} \otimes \underline{A} \to \underline{A}$ *defined above is an equivalence.*

$$\textsc{$\mathbb{S}$-base-elim}\ \dfrac{\begin{array}{c}\mathfrak{t} \prec \Phi \mid \Gamma, z^{\mathfrak{t}} : \natural\mathbb{S} \vdash C \ \mathsf{type} \\ \mathfrak{t} \prec \Phi \mid \Gamma \vdash c : C[\underline{\currency}^{\natural}/z] \\ \mathfrak{t} \prec \Phi \mid \Gamma \vdash s : \natural\mathbb{S}\end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \mathsf{let}\ \underline{\currency}^{\natural} = s\ \mathsf{in}\ c : C[s/z]}$$

$$\textsc{$\mathbb{S}$-base-beta}\ \dfrac{\begin{array}{c}\mathfrak{t} \prec \Phi \mid \Gamma, y^{\mathfrak{t}} : B \vdash c : C[\currency_i\ {}_{\mathfrak{u}.\prec\varnothing_i}\!\otimes_{\mathfrak{t}} y/z] \\ \mathfrak{t} \prec \Phi \mid \Gamma \vdash b : B\end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash (\mathsf{let}\ \currency_i\ {}_{\mathfrak{u}.\prec\varnothing_i}\!\otimes_{\mathfrak{t}} y = \currency_j\ {}_{\mathfrak{v}.\prec\varnothing_j}\!\otimes_{\mathfrak{t}} b\ \mathsf{in}\ c) \equiv c[b/y] : C[\currency_j\ {}_{\mathfrak{v}.\prec\varnothing_j}\!\otimes_{\mathfrak{t}} b/z]}$$

$$\textsc{$\mathbb{S}$-unitor-left-elim}\ \dfrac{\begin{array}{c}\mathfrak{t} \mid \underline{\Gamma}, \underline{x} : \mathbb{S} \vdash B \ \mathsf{type} \\ \mathfrak{t} \prec \Phi \mid \Gamma, z^{\mathfrak{t}} : \textcircled{$\scriptstyle\sum$}_{(\underline{x}:\mathbb{S})} B \vdash C \ \mathsf{type} \\ \mathfrak{t} \prec \Phi \mid \Gamma, y^{\mathfrak{t}} : B \vdash c : C[\currency_i\ {}_{\mathfrak{u}.\prec\varnothing_i}\!\otimes_{\mathfrak{t}} y/z] \\ \mathfrak{t} \prec \Phi \mid \Gamma \vdash s : \textcircled{$\scriptstyle\sum$}_{(\underline{x}:\mathbb{S})} B\end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \mathsf{let}\ \currency_i\ {}_{\mathfrak{u}.\prec\varnothing_i}\!\otimes_{\mathfrak{t}} y = s\ \mathsf{in}\ c : C[s/z]}$$

$$\textsc{$\mathbb{S}$-unitor-left-beta}\ \dfrac{\begin{array}{c}\mathfrak{t} \mid \underline{\Gamma}, \underline{x} : \mathbb{S} \vdash B \ \mathsf{type} \\ \mathfrak{t} \prec \Phi \mid \Gamma, z^{\mathfrak{t}} : \textcircled{$\scriptstyle\sum$}_{(\underline{x}:\mathbb{S})} B \vdash C \ \mathsf{type} \\ \mathfrak{t} \prec \Phi \mid \Gamma, y^{\mathfrak{t}} : B \vdash c : C[\currency_i\ {}_{\mathfrak{u}.\prec\varnothing_i}\!\otimes_{\mathfrak{t}} y/z] \\ \mathfrak{t} \prec \Phi \mid \Gamma \vdash b : B\end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash (\mathsf{let}\ \currency_i\ {}_{\mathfrak{u}.\prec\varnothing_i}\!\otimes_{\mathfrak{t}} y = \currency_j\ {}_{\mathfrak{v}.\prec\varnothing_j}\!\otimes_{\mathfrak{t}} b\ \mathsf{in}\ c) \equiv c[b/y] : C[\currency_j\ {}_{\mathfrak{v}.\prec\varnothing_j}\!\otimes_{\mathfrak{t}} b/z]}$$

$$\textsc{$\mathbb{S}$-unitor-right-elim}\ \dfrac{\begin{array}{c}\mathfrak{t} \mid \underline{\Gamma} \vdash A \ \mathsf{type} \\ \mathfrak{t} \prec \Phi \mid \Gamma, z^{\mathfrak{t}} : A \otimes \mathbb{S} \vdash C \ \mathsf{type} \\ \mathfrak{t} \prec \Phi \mid \Gamma, x^{\mathfrak{t}} : A \vdash c : C[x\ {}_{\mathfrak{t}}\!\otimes_{\mathfrak{u}.\prec\varnothing_i} \currency_i/z] \\ \mathfrak{t} \prec \Phi \mid \Gamma \vdash s : A \otimes \mathbb{S}\end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \mathsf{let}\ x\ {}_{\mathfrak{t}}\!\otimes_{\mathfrak{u}.\prec\varnothing_i} \currency_i = s\ \mathsf{in}\ c : C[s/z]}$$

$$\textsc{$\mathbb{S}$-unitor-right-beta}\ \dfrac{\begin{array}{c}\mathfrak{t} \mid \underline{\Gamma} \vdash A \ \mathsf{type} \\ \mathfrak{t} \prec \Phi \mid \Gamma, z^{\mathfrak{t}} : A \otimes \mathbb{S} \vdash C \ \mathsf{type} \\ \mathfrak{t} \prec \Phi \mid \Gamma, x^{\mathfrak{t}} : A \vdash c : C[x\ {}_{\mathfrak{t}}\!\otimes_{\mathfrak{u}.\prec\varnothing_i} \currency_i/z] \\ \mathfrak{t} \prec \Phi \mid \Gamma \vdash a : A\end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash (\mathsf{let}\ x\ {}_{\mathfrak{t}}\!\otimes_{\mathfrak{u}.\prec\varnothing_i} \currency_i = a\ {}_{\mathfrak{t}}\!\otimes_{\mathfrak{v}.\prec\varnothing_j} \currency_j\ \mathsf{in}\ c) \equiv c[a/x] : C[a\ {}_{\mathfrak{t}}\!\otimes_{\mathfrak{v}.\prec\varnothing_j} \currency_j/z]}$$

Figure 1.18: Induction Principles for the Monoidal Unit

Proof. The composite $\mathsf{unitl}_{\underline{A}} \circ \mathsf{unitlinv}_{\underline{A}}$ is definitionally the identity by the computation rule for unitor induction.

$$\begin{aligned}\mathsf{unitl}_{\underline{A}}(\mathsf{unitlinv}_{\underline{A}}(a)) &\equiv \mathsf{let}\ \text{⌑}_{i}\ {}_{\varnothing_i}\!\otimes_{\mathfrak{p}} a = \mathsf{unitlinv}_{\underline{A}}(a)\ \mathsf{in}\ a \\ &\equiv \mathsf{let}\ \text{⌑}_{i}\ {}_{\varnothing_i}\!\otimes_{\mathfrak{p}} x = \text{⌑}_{i}\ {}_{\varnothing_i}\!\otimes_{\mathfrak{p}} a\ \mathsf{in}\ x \\ &\equiv x[a/x] \equiv a\end{aligned}$$

For the other direction, we are looking for a term of

$$\prod_{(p:\mathbb{S}\otimes\underline{A})} \mathsf{unitlinv}_{\underline{A}}(\mathsf{unitl}_{\underline{A}}(p)) = p$$

So perform induction on p as $\text{⌑}_{i'}\ {}_{\varnothing_{i'}}\!\otimes_{\mathfrak{p}} x$, again using unitor induction. We can now compute

$$\mathsf{unitlinv}_{\underline{A}}(\mathsf{unitl}_{\underline{A}}(\text{⌑}_{i'}\ {}_{\varnothing_{i'}}\!\otimes_{\mathfrak{p}} x)) \equiv \mathsf{unitrinv}_{\underline{A}}(x) \equiv \text{⌑}_{i}\ {}_{\varnothing_i}\!\otimes_{\mathfrak{p}} x$$

This is judgementally equal (up to α-equivalence of the colour label) with $\text{⌑}_{i'}\ {}_{\varnothing_{i'}}\!\otimes_{\mathfrak{p}} x$, so we can take $\mathsf{refl}_{\text{⌑}_{i'}\ {}_{\varnothing_{i'}}\otimes_{\mathfrak{p}} x}$ as our path. □

1.5 Hom

The ⊗-type has a right adjoint: a restricted, 'linear' version of the ordinary function type. We call these hom types and write the non-dependent instances using the $\multimap$ symbol. Written in our palette notation, we wish for this type former to be part of a bijection of judgement states

$$\frac{\mathfrak{p} \prec \mathfrak{r} \otimes \mathfrak{b} \mid x^{\mathfrak{r}} : A, y^{\mathfrak{b}} : B \vdash (\ldots) : C}{\overline{\mathfrak{r} \mid x^{\mathfrak{r}} : A \vdash (\ldots) : B \multimap C}}$$

Before showing the formal rules, let us demonstrate how a currying map is defined.

Proposition 1.5.1 (Currying for $\multimap$). *There is a map* $((\underline{A} \otimes \underline{B}) \multimap \underline{C}) \to (\underline{A} \multimap (\underline{B} \multimap \underline{C}))$.

Proof. Let us write the top colour as yellow $\mathfrak{y}$. The arrow in the centre of the currying map is an ordinary cartesian arrow, so suppose we have $h^{\mathfrak{y}} : (\underline{A} \otimes \underline{B}) \multimap \underline{C}$.

To form a term of type $\underline{A} \multimap (\underline{B} \multimap \underline{C})$, we use ∂-abstraction. A first use adds an assumption $x^{\mathfrak{r}} : \underline{A}$, where $\mathfrak{r}$ is a fresh colour tensored with $\mathfrak{y}$. It would be reasonable to call this combination 'orange' $\mathfrak{o}$, and the state of the palette is now $\mathfrak{o} \prec \mathfrak{y} \otimes \mathfrak{r}$. A second use of ∂-abstraction gives an assumption $y^{\mathfrak{b}} : \underline{B}$, where $\mathfrak{b}$ is similarly a fresh colour tensored with $\mathfrak{o}$. We choose to call this colour combination $\mathfrak{w}$, so the state of the palette is

$$\mathfrak{w} \prec (\mathfrak{o} \prec \mathfrak{y} \otimes \mathfrak{r}) \otimes \mathfrak{b}$$

We now have to use h, x and y to form a term of $\underline{C}$. To apply a hom to an argument, we must split our resources in the same sense that we do for ⊗-INTRO, using one side to produce the hom and the other side to produce the argument. We choose the split $\mathfrak{y}$ on the left and $\mathfrak{p}. \prec \mathfrak{r} \otimes \mathfrak{b}$ on the right. For the hom, we use the variable h, and for the argument we use $x\ {}_{\mathfrak{r}}\!\otimes_{\mathfrak{b}} y$, forming the 'linear application' $h_{\mathfrak{y}}\langle x\ {}_{\mathfrak{r}}\!\otimes_{\mathfrak{b}} y\rangle_{\mathfrak{p}.\prec\mathfrak{r}\otimes\mathfrak{b}}$.

As a term, we have formed

$$\lambda h.\partial^{\mathfrak{o}} x^{\mathfrak{r}}.\partial^{\mathfrak{w}} y^{\mathfrak{b}}.h_{\mathfrak{y}}\langle x\ {}_{\mathfrak{r}}\!\otimes_{\mathfrak{b}} y\rangle_{\mathfrak{p}.\prec\mathfrak{r}\otimes\mathfrak{b}} : ((\underline{A} \otimes \underline{B}) \multimap \underline{C}) \to (\underline{A} \multimap (\underline{B} \multimap \underline{C}))$$

□

$$\multimap\text{-FORM}\ \frac{\begin{array}{c}\mathfrak{r} \mid \underline{\Gamma} \vdash A \ \mathsf{type} \\ \mathfrak{p} \prec (\mathfrak{t} \prec \Phi) \otimes \mathfrak{r} \mid \Gamma, x^{\mathfrak{r}} : A \vdash B \ \mathsf{type}\end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \textcircled{$\prod$}_{(x^{\mathfrak{r}}:A)}{}^{\mathfrak{p}}B \ \mathsf{type}}$$

$$\multimap\text{-INTRO}\ \frac{\mathfrak{p} \prec (\mathfrak{t} \prec \Phi) \otimes \mathfrak{r} \mid \Gamma, x^{\mathfrak{r}} : A \vdash b : B}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \partial^{\mathfrak{p}} x^{\mathfrak{r}}.b : \textcircled{$\prod$}_{(x^{\mathfrak{r}}:A)}{}^{\mathfrak{p}}B}$$

$$\multimap\text{-ELIM}\ \frac{\begin{array}{c}\mathfrak{t} \prec \Phi \vdash s_L \boxtimes s_R \ \mathsf{split} \qquad \mathfrak{r} :\equiv \ulcorner s_R \urcorner \\ (\mathfrak{t} \prec \Phi)^{s_L} \mid \Gamma^{s_L} \vdash f : \textcircled{$\prod$}_{(x^{\mathfrak{r}}:A)}{}^{\mathfrak{p}}B \qquad (\mathfrak{t} \prec \Phi)^{s_R} \mid \Gamma^{s_R} \vdash a : A\end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash f_{s_L}\langle a \rangle_{s_R} : B[a/x]\llbracket \mathfrak{t} \prec s_L \boxtimes s_R / \mathfrak{p} \rrbracket}$$

$$\multimap\text{-BETA}\ \frac{\begin{array}{c}\mathfrak{t} \prec \Phi \vdash s_L \boxtimes s_R \ \mathsf{split} \qquad \mathfrak{r} :\equiv \ulcorner s_R \urcorner \\ \mathfrak{p} \prec (\Phi^{s_L} \otimes \mathfrak{r}) \mid \Gamma^{s_L}, x^{\mathfrak{r}} : A \vdash b : B \qquad \Phi^{s_R} \mid \Gamma^{s_R} \vdash a : A\end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash (\partial^{\mathfrak{p}} x^{\mathfrak{r}}.b)_{s_L}\langle a \rangle_{s_R} \equiv b[a/x]\llbracket \mathfrak{t} \prec s_L \boxtimes s_R / \mathfrak{p} \rrbracket : B[a/x]\llbracket \mathfrak{t} \prec s_L \boxtimes s_R / \mathfrak{p} \rrbracket}$$

$$\multimap\text{-ETA}\ \frac{\mathfrak{t} \prec \Phi \mid \Gamma \vdash f : \textcircled{$\prod$}_{(x^{\mathfrak{r}}:A)}{}^{\mathfrak{p}}B}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash f \equiv (\partial^{\mathfrak{p}} x^{\mathfrak{r}}.f_{\mathfrak{t}}\langle x \rangle_{\mathfrak{r}}) : \textcircled{$\prod$}_{(x^{\mathfrak{r}}:A)}{}^{\mathfrak{p}}B}$$

$$\text{MERGE}\ \frac{\begin{array}{c}\mathfrak{t} \prec \Phi \vdash s_L \boxtimes s_R \ \mathsf{split} \\ \mathfrak{t}' \prec \Phi^{s_L} \otimes \Phi^{s_R} \mid \Gamma^{\Phi^{s_L} \otimes \Phi^{s_R}} \vdash \mathcal{J}\end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \mathcal{J}\llbracket \mathfrak{t} \prec s_L \boxtimes s_R / \mathfrak{t}' \rrbracket}$$

Figure 1.19: Rules for the Hom Type

1.5.1 Type Former

As with the $\otimes$-type, we will allow a dependent version of $\multimap$ which we call $\textcircled{$\prod$}$. The dependencies we allow in the inputs of the hom type are exactly the dependencies implicit in the premise '$\mathfrak{p} \prec \mathfrak{r} \otimes \mathfrak{b} \mid x^{\mathfrak{r}} : A, y^{\mathfrak{b}} : B \vdash (\ldots) : C$': the domain B is required to be dull (because there are no other $\mathfrak{b}$ variables), but the codomain C can be *any* type in context $\mathfrak{p} \prec \mathfrak{r} \otimes \mathfrak{b} \mid x^{\mathfrak{r}} : A, y^{\mathfrak{b}} : B$. Our actual formation rule relaxes $x^{\mathfrak{r}} : A$ to an arbitrary context.

The rules for the $\multimap$-type are given in Figure 1.19.

- **Formation:** For any dull type A, and any type B using a new assumption $x^{\mathfrak{r}} : A$ as a disjoint resource, there is a type $\textcircled{$\prod$}_{(x^{\mathfrak{r}}:A)}{}^{\mathfrak{p}}B$.

  The linearity constraint means that $\mathfrak{r}$ is a fresh colour, and this colour is *linearly* combined with the top colour $\mathfrak{t}$ of the ambient palette. In B, we need a new label to name this tensor combination of $\mathfrak{t}$ and $\mathfrak{r}$, which is the second colour $\mathfrak{p}$ that is bound in B. In particular, this means that the top colour changes when going under a hom binder, and so we (temporarily) lose access to all the variables labelled the previous top colour. Access can be regained by using one of the splitting rules, either $\otimes$-INTRO or $\multimap$-ELIM.

If B does not depend on A and the labels $\mathfrak{r}$ and $\mathfrak{p}$ do not appear in B, then we write $A \multimap B$.

Chained hom types associate to the right, like ordinary function types: $A \multimap B \multimap C$ is read as $A \multimap (B \multimap C)$. The same is true when mixing Π- and $\multimap$-types: we read $A \to B \multimap C$ as $A \to (B \multimap C)$.

- **Introduction:** If $b : B$ is a term using $x^{\mathfrak{r}} : A$ as a disjoint resource from the ambient context, then we have the ∂-abstraction $\partial^{\mathfrak{p}} x^{\mathfrak{r}}.b : \text{Ⓟ}_{(x^{\mathfrak{r}}:A)}{}^{\mathfrak{p}}B$.

- **Elimination:** If we can split the ambient palette into two slices s_L and s_R, and use s_L to produce a hom $f : \text{Ⓟ}_{(x^{\mathfrak{r}}:A)}{}^{\mathfrak{p}}B$ and s_R to produce $a : A$, then we can apply f to a to form $f_{s_L}\langle a\rangle_{s_R} : B[a/x]\llbracket \mathfrak{t} \prec s_L \boxtimes s_R/\mathfrak{p}\rrbracket$. The split of the palette is specified using the same $\Phi \vdash s_L \boxtimes s_R$ split judgement as used in $\otimes$-INTRO.

  We might hope for the type to be $B[a/x]$ exactly, but unfortunately the palette does not line up: the result of the substitution is in palette $\mathfrak{p} \prec \Phi^{s_L} \otimes \Phi^{s_R}$ rather than Φ, and must be weakened 'by hand' using an admissible $\mathcal{J}\llbracket \mathfrak{t} \prec s_L \boxtimes s_R/\mathfrak{p}\rrbracket$ operation which we discuss below.

  As in $\otimes$-INTRO, it is typically obvious which split of the palettes has been used, just by inspecting how the variables appear on both sides. We will elide the slices, unless there is some ambiguity in which split has been used.

- **Computation:** Whenever a ∂-abstraction is applied directly to an argument, we can compute the result:
$$(\partial^{\mathfrak{p}} x^{\mathfrak{r}}.b)\langle a\rangle \equiv b[a/x]\llbracket \mathfrak{t} \prec s_L \boxtimes s_R/\mathfrak{p}\rrbracket.$$
  As with ordinary functions, this is done by substituting the argument for the variable in the body of the function. Again we apply a 'merge' to the result, the same operation that is performed on the type in the elimination rule.

- **Uniqueness:** Any hom $h : \text{Ⓟ}_{(x^{\mathfrak{r}}:A)}{}^{\mathfrak{p}}B$ is definitionally equal to 'the hom that applies f to its argument':
$$h \equiv \partial^{\mathfrak{p}} x^{\mathfrak{r}}.h\langle x\rangle.$$

- **Merging:** Merging moves a judgement in palette $\Phi^{s_L \otimes s_R}$ into palette Φ, whenever $\Phi \vdash s_L \boxtimes s_R$ split

  Splits correspond roughly to morphisms $\Phi \to \Phi^{s_L} \otimes \Phi^{s_R}$, and merging corresponds to precomposition with this morphism. The original palette Φ may contain more colours than those that appear in s_L and s_R, and so this is another form of weakening.

  On syntax, there is only one modification that needs to be made: the judgement $\mathcal{J}$ may refer to fresh top colours bound by the slices s_L and s_R, and these do not exist in the palette Φ. The fix is to replace the first mention of these colours with the entire corresponding slice.

  Only *first* occurrence of each colour as a slice is replaced, and the operation does not recur deeper into the term. We want any further mentions of that colour to refer to binding site that the substitution has just created, rather a fresh colour with the same name.

  The definition of this merging operation is simplified by instead defining an operation that substitutes an individual slice in this way. The merging operation is then defined by doing two single-slice substitutions followed by a $\mathfrak{t} \leftrightarrow \mathfrak{t}$ recolouring. The result of doing just one slice substitution is *not* well-formed, only the combination of all three operations.

To demonstrate slice substitution, fix a split

$$\mathfrak{t} \prec \mathfrak{a} \otimes \mathfrak{b} \otimes \mathfrak{c} \otimes \mathfrak{d} \vdash (\mathfrak{l}. \prec \mathfrak{a} \otimes \mathfrak{b}) \boxtimes (\mathfrak{r}. \prec \mathfrak{c} \otimes \mathfrak{d}) \ \mathsf{split}$$

Some single slice substitutions on terms are then

$$\begin{aligned}
(m \;{}_{\mathfrak{l}}{\otimes}_{\mathfrak{r}}\; (n \;{}_{\mathfrak{c}}{\otimes}_{\mathfrak{d}}\; o))\llbracket \mathfrak{r}. \prec \mathfrak{c} \otimes \mathfrak{d} / \mathfrak{r} \rrbracket &:\equiv (m \;{}_{\mathfrak{l}}{\otimes}_{\mathfrak{r}.\prec \mathfrak{c}\otimes\mathfrak{d}}\; (n \;{}_{\mathfrak{c}}{\otimes}_{\mathfrak{d}}\; o)) \\
(m \;{}_{\mathfrak{a}}{\otimes}_{\mathfrak{s}.\prec \mathfrak{b}\otimes\mathfrak{r}}\; (n \;{}_{\mathfrak{b}}{\otimes}_{\mathfrak{r}}\; o))\llbracket \mathfrak{r}. \prec \mathfrak{c} \otimes \mathfrak{d} / \mathfrak{r} \rrbracket &:\equiv (m \;{}_{\mathfrak{a}}{\otimes}_{\mathfrak{s}.\prec \mathfrak{b}\otimes\mathfrak{c}\otimes\mathfrak{d}}\; (n \;{}_{\mathfrak{b}}{\otimes}_{\mathfrak{r}.\prec \mathfrak{c}\otimes\mathfrak{d}}\; o)) \\
(m \;{}_{\mathfrak{l}}{\otimes}_{\mathfrak{r}}\; (n \;{}_{\mathfrak{r}}{\otimes}_{\mathfrak{s}.\prec \varnothing_{i.}}\; o))\llbracket \mathfrak{r}. \prec \mathfrak{c} \otimes \mathfrak{d} / \mathfrak{r} \rrbracket &:\equiv (m \;{}_{\mathfrak{l}}{\otimes}_{\mathfrak{r}.\prec \mathfrak{c}\otimes\mathfrak{d}}\; (n \;{}_{\mathfrak{r}}{\otimes}_{\mathfrak{s}.\prec \varnothing_{i.}}\; o))
\end{aligned}$$

In the second example, the slice substitution into the split on the left has not bound the colour $\mathfrak{r}$, and so it bound in the nested split on the right. In the third example, the $\mathfrak{r}$ *does* get bound in the left slice, and so the right slice refers to that $\mathfrak{r}$ rather than binding a new one.

Then, merging on the same terms is given by

$$\begin{aligned}
(m \;{}_{\mathfrak{l}}{\otimes}_{\mathfrak{r}}\; (n \;{}_{\mathfrak{c}}{\otimes}_{\mathfrak{d}}\; o))\llbracket \mathfrak{t} \prec (\mathfrak{l}. \prec \mathfrak{a} \otimes \mathfrak{b}) \boxtimes (\mathfrak{r}. \prec \mathfrak{c} \otimes \mathfrak{d}) / \mathfrak{t}' \rrbracket &:\equiv (m \;{}_{\mathfrak{l}.\prec\mathfrak{a}\otimes\mathfrak{b}}{\otimes}_{\mathfrak{r}.\prec \mathfrak{c}\otimes\mathfrak{d}}\; (n \;{}_{\mathfrak{c}}{\otimes}_{\mathfrak{d}}\; o)) \\
(m \;{}_{\mathfrak{a}}{\otimes}_{\mathfrak{s}.\prec \mathfrak{b}\otimes\mathfrak{r}}\; (n \;{}_{\mathfrak{b}}{\otimes}_{\mathfrak{r}}\; o))\llbracket \mathfrak{t} \prec (\mathfrak{l}. \prec \mathfrak{a} \otimes \mathfrak{b}) \boxtimes (\mathfrak{r}. \prec \mathfrak{c} \otimes \mathfrak{d}) / \mathfrak{t}' \rrbracket &:\equiv (m \;{}_{\mathfrak{a}}{\otimes}_{\mathfrak{s}.\prec \mathfrak{b}\otimes\mathfrak{c}\otimes\mathfrak{d}}\; (n \;{}_{\mathfrak{b}}{\otimes}_{\mathfrak{r}.\prec \mathfrak{c}\otimes\mathfrak{d}}\; o)) \\
(m \;{}_{\mathfrak{l}}{\otimes}_{\mathfrak{r}}\; (n \;{}_{\mathfrak{r}}{\otimes}_{\mathfrak{s}.\prec \varnothing_{i.}}\; o))\llbracket \mathfrak{t} \prec (\mathfrak{l}. \prec \mathfrak{a} \otimes \mathfrak{b}) \boxtimes (\mathfrak{r}. \prec \mathfrak{c} \otimes \mathfrak{d}) / \mathfrak{t}' \rrbracket &:\equiv (m \;{}_{\mathfrak{l}}{\otimes}_{\mathfrak{r}.\prec \mathfrak{c}\otimes\mathfrak{d}}\; (n \;{}_{\mathfrak{r}}{\otimes}_{\mathfrak{s}.\prec \varnothing_{i.}}\; o))
\end{aligned}$$

Only the slice annotations in a term are changed by this operation: the shape of the term remains the same.

Remark 1.5.2. The splits used in the currying example are completely determined by the way the variables are used, and the extra colours bound by the ∂-abstractions are not used at all. Leaving them off yields

$$\lambda h.\partial x.\partial y.h\langle x \otimes y\rangle : ((\underline{A} \otimes \underline{B}) \multimap \underline{C}) \to (\underline{A} \multimap (\underline{B} \multimap \underline{C}))$$

which is significantly more readable. Going forwards we allow ourselves to work more informally, not mentioning splits if they are determined by the terms and not keeping explicit track of the shape of the palette.

Remark 1.5.3. We can again describe the type former as a function into the universe. The domain must be a dull type, stated internally, a term $A : \natural\mathcal{U}$. The codomain can depend on this type in a genuine way without the modality applied to it, but only linearly: this is a non-dull type family $B : \underline{A}_\natural \multimap \mathcal{U}$. Given these inputs we can form the dependent hom, so the type former itself has type

$$\text{“}\textcircled{\textstyle\prod}\text{”} : \prod_{(A:\natural\mathcal{U})}(\underline{A}_\natural \multimap \mathcal{U}) \to \mathcal{U}$$

So interestingly $\textcircled{\textstyle\prod}$ is a *function* of its domain and codomain types, not a hom.

Remark 1.5.4. The rules for $\multimap$-types produce the desired bijection of term judgements

$$\begin{array}{c}
\mathfrak{p} \prec (\mathfrak{r} \prec \Phi) \otimes \mathfrak{b} \mid \Gamma, y^{\mathfrak{b}} : B \vdash c : C \\
\hline\hline
\mathfrak{r} \prec \Phi \mid \Gamma \vdash \partial y.c : B \multimap C
\end{array}$$

via the β- and η- rules.

The entire context Γ is placed onto the left side of the tensor in the premise, and we are forbidden from applying $\multimap$-INTRO when the context is not in this form: semantically we do not expect there to be an operation

$$\frac{\mathfrak{p} \prec (\Phi, \mathfrak{r} \otimes \mathfrak{b}) \mid \Gamma, x^{\mathfrak{r}} : A, y^{\mathfrak{b}} : B \vdash (\dots) : C}{\mathfrak{p} \prec (\Phi, \mathfrak{r}) \mid \Gamma, x^{\mathfrak{r}} : A \vdash (\dots) : ?(B, C)}$$

Remark 1.5.5. Here we reproduce a discussion from the original work on bunched type theories [OHe03]. We might be accustomed to thinking of linear functions as functions that use their argument exactly once. This does not fit with the way variables in our theory behave: instead we should think about *sharing* and *resource access*. The type $A \otimes B$ describes disjoint resources, and $A \multimap B$ describes functions that do not share resources with their arguments.

It is not enough for the body of the hom to use x exactly once, it also has to use the ambient context in a linear way. So we cannot define a closed term $A \otimes B \multimap B \otimes A$, similar to $\mathsf{sym}_{A,B} : A \otimes B \to B \otimes A$ defined using the ordinary function type. Even though the body of the function

$$\mathsf{sym}_{A,B} :\equiv \lambda p.(\mathsf{let}\ x \otimes y = p\ \mathsf{in}\ y \otimes x)$$

'uses p exactly once', it doesn't use *anything* from the ambient context, so fails to satisfy this notion of sharing.

Once we *do* have access to a resource, we can use it as many times as we like: for example, suppose $h : A \multimap B$. We can use this to define a hom

$$\partial x.(h\langle x\rangle, h\langle x\rangle) : A \multimap (B \times B)$$

where the argument to the hom has been used twice. We are also free to not to use the argument at all:

$$\partial x.\star : A \multimap 1$$

Like linear logic, we cannot convert a function $A \to B$ to a hom $A \multimap B$. But a second contrast to linear logic is that we cannot in general convert homs $A \multimap B$ into functions $A \to B$. In intuitionistic linear logic [Gir87], functions $A \to B$ are interpreted as $!A \multimap B$, and we can always go from $A \multimap B$ to $!A \multimap B$ using dereliction. In our theory, however, we are blocked: if $h : A \multimap B$ and $x : A$, we cannot linearly apply h to x because these terms share the 'red' resource of the context.

We can conclude that bunched type systems and intuitionistic linear logic are incomparable extensions of the linear λ-calculus[4] and the ordinary λ-calculus. These differences are why the work on BI uses $*$ and $-\!*$ as the type formers, to remind us that our intuitions from linear logic do not apply. We prefer to stick with $\otimes$ and $\multimap$ because the operations are intended to be modelled by an arbitrary symmetric monoidal closed structure, and $*$ suggests an operation with more structure.

[4] To be specific, the fragment of intuitionistic linear logic with only $\otimes$ and $\multimap$, called Rudimentary Linear Logic by [GSS92] and Tensor-Implication Logic by [HP93].

For several of the proceeds in the following section, we will need to use *hom extensionality*. For $f, g : \bigcirc\!\!\!\!\prod_{(x:\underline{A})} B\langle x\rangle$, there is a canonical function

$$\mathsf{homapp}(f,g) : (f = g) \to \bigcirc\!\!\!\!\prod_{(x:\underline{A})} f\langle x\rangle = g\langle x\rangle$$

given by path induction:

$$\mathsf{homapp}(f,f)(\mathsf{refl}_f) :\equiv \partial x.\mathsf{refl}_{f\langle x\rangle}$$

Axiom Homext. *For any* $f, g : \bigcirc\!\!\!\!\prod_{(x:\underline{A})} B\langle x\rangle$*, the function* $\mathsf{homapp}(f,g)$ *is an equivalence.*

We write $\mathsf{homext}(f,g) : \bigcirc\!\!\!\!\prod_{(x:\underline{A})}(f\langle x\rangle = g\langle x\rangle) \to (f = g)$ for the inverse to $\mathsf{homapp}(f,g)$. From this point on, we will assume hom extensionality holds. Univalence implies hom extensionality, but we defer the proof of this fact to Section 1.5.5 to first build more experience working with homs.

A dependent version of the $\multimap$-type is necessary to state hom extensionality internally, even for non-dependent $\multimap$-types.

1.5.2 Basic Properties of Hom

As expected, $\multimap$ is right adjoint to $\otimes$. There are different ways we might phrase this property depending on what we think of as our 'type of morphisms'. First we give a version that uses another hom as the mediating arrow.

Proposition 1.5.6. *For dull types* $\underline{A}, \underline{B}$ *and* $\underline{C}$*, there is an 'internal' adjunction*

$$((\underline{A} \otimes \underline{B}) \multimap \underline{C}) \simeq (\underline{A} \multimap (\underline{B} \multimap \underline{C}))$$

More dependently, let the top colour be yellow and suppose $\underline{A} : \mathcal{U}$*,* $\underline{B} : \underline{A} \to \mathcal{U}$ *and* $C : (\underline{A} \otimes \underline{B}) \multimap \mathcal{U}$*. Then*

$$\bigcirc\!\!\!\!\prod_{(p:\bigcirc\!\!\!\!\sum_{(\underline{x}:\underline{A})} \underline{B}(\underline{x}))} C\langle p\rangle \simeq \bigcirc\!\!\!\!\prod_{(x:\underline{A})} \bigcirc\!\!\!\!\prod_{(y:\underline{B}(\underline{x}))} C\langle x \otimes y\rangle$$

Proof. The map in the forward direction is defined just as it was in the example above, even in the dependent case: we send h to the hom

$$\lambda h.\partial x.\partial y.h\langle x \otimes y\rangle : \bigcirc\!\!\!\!\prod_{(x:\underline{A})} \bigcirc\!\!\!\!\prod_{(y:\underline{B}(\underline{x}))} C\langle x \otimes y\rangle$$

For the other direction, suppose $f : \bigcirc\!\!\!\!\prod_{(p:\bigcirc\!\!\!\!\sum_{(\underline{x}:\underline{A})} \underline{B}(\underline{x}))} C\langle p\rangle$. We wish to form a term of the hom type $\bigcirc\!\!\!\!\prod_{(x:\underline{A})} \bigcirc\!\!\!\!\prod_{(y:\underline{B}(\underline{x}))} C\langle x \otimes y\rangle$, so we assume we have $x : \underline{A}$ and $y : \underline{B}(\underline{x})$, then we can form $f\langle x \otimes y\rangle : C\langle x \otimes y\rangle$ which has the correct colour. So in all, we have

$$\partial x.\partial y.f\langle x \otimes y\rangle : \bigcirc\!\!\!\!\prod_{(x:\underline{A})} \bigcirc\!\!\!\!\prod_{(y:\underline{B}(\underline{x}))} C\langle x \otimes y\rangle$$

In the other direction, starting with $g : \bigcirc\!\!\!\!\prod_{(x:\underline{A})} \bigcirc\!\!\!\!\prod_{(y:\underline{B}(\underline{x}))} C\langle x \otimes y\rangle$, we define a hom of type $\bigcirc\!\!\!\!\prod_{(p:\bigcirc\!\!\!\!\sum_{(\underline{x}:\underline{A})} \underline{B}(\underline{x}))} C\langle p\rangle$ as follows. Introduce $p : \bigcirc\!\!\!\!\sum_{(x:\underline{A})} \underline{B}(\underline{x})$ by ∂-abstraction. We would like to do induction on p, but it does not have the top colour. Instead, use the stronger eliminator[5] of

[5]Here we are cheating a little: we need to use the pattern-matching eliminator that allows the motive C to not be dull.

Section 1.3.4 on $g \otimes p$ to give $g' \otimes (x \otimes y)$. Then we can apply g' to x and y one at a time giving $g'\langle x\rangle\langle y\rangle : C\langle x \otimes y\rangle$. In all:

$$\partial p.\mathsf{let}\ g' \otimes (x \otimes y) = g \otimes p\ \mathsf{in}\ g'\langle x\rangle\langle y\rangle : \textstyle\prod_{(p:\sum_{(\underline{x}:\underline{A})} \underline{B}(\underline{x}))} C\langle p\rangle$$

Checking the round-trips is straightforward. One round-trip is definitionally the identity, using the η-rule in the last steps.

$$\begin{aligned}
&\partial x.\partial y.\ (\partial p.\mathsf{let}\ g' \otimes (x \otimes y) = g \otimes p\ \mathsf{in}\ g'\langle x\rangle\langle y\rangle)\ \langle x \otimes y\rangle \\
&\equiv \partial x.\partial y.\mathsf{let}\ g' \otimes (x \otimes y) = g \otimes (x \otimes y)\ \mathsf{in}\ g'\langle x\rangle\langle y\rangle \\
&\equiv \partial x.\partial y.g\langle x\rangle\langle y\rangle \\
&\equiv \partial x.g\langle x\rangle \\
&\equiv g
\end{aligned}$$

In the other direction, we will have to use uniqueness for $\otimes$ and hom extensionality. To be completely explicit, let F denote the function defined by

$$F(f \otimes p) :\equiv \mathsf{let}\ g' \otimes (x \otimes y) = (\partial x.\partial y.f\langle x \otimes y\rangle) \otimes p\ \mathsf{in}\ g'\langle x\rangle\langle y\rangle$$

so that uniqueness for the triple eliminator yields a path

$$\left(\mathsf{let}\ f' \otimes (x' \otimes y') = f \otimes p\ \mathsf{in}\ F(f' \otimes (x' \otimes y'))\right) = F(f \otimes p)$$

Now we can calculate

$$\begin{aligned}
F(f' \otimes (x' \otimes y')) &\equiv \mathsf{let}\ g' \otimes (x \otimes y) = (\partial x.\partial y.f'\langle x \otimes y\rangle) \otimes (x' \otimes y')\ \mathsf{in}\ g'\langle x\rangle\langle y\rangle \\
&\equiv (\partial x.\partial y.f'\langle x \otimes y\rangle)\langle x'\rangle\langle y'\rangle \\
&\equiv f'\langle x' \otimes y'\rangle
\end{aligned}$$

so that

$$\begin{aligned}
&\partial p.\mathsf{let}\ g' \otimes (x \otimes y) = (\partial x.\partial y.f\langle x \otimes y\rangle) \otimes p\ \mathsf{in}\ g'\langle x\rangle\langle y\rangle \\
&\equiv \partial p.F(f \otimes p) \\
&= \partial p.\mathsf{let}\ f' \otimes (x' \otimes y') = f \otimes p\ \mathsf{in}\ F(f' \otimes (x' \otimes y')) \\
&\equiv \partial p.\mathsf{let}\ f' \otimes (x' \otimes y') = f \otimes p\ \mathsf{in}\ f'\langle x' \otimes y'\rangle
\end{aligned}$$

Another application of uniqueness in the other direction with

$$G(f \otimes p) :\equiv f\langle p\rangle$$

gives that

$$\begin{aligned}
&\partial p.\mathsf{let}\ f' \otimes (x' \otimes y') = f \otimes p\ \mathsf{in}\ f'\langle x' \otimes y'\rangle \\
&\equiv \partial p.\mathsf{let}\ f' \otimes (x' \otimes y') = f \otimes p\ \mathsf{in}\ G(f' \otimes (x' \otimes y')) \\
&= \partial p.G(f \otimes p) \\
&\equiv \partial p.f\langle p\rangle \\
&\equiv f
\end{aligned}$$

and we are done. □

Remark 1.5.7. The use of the triple eliminator here is a little awkward: again we are running into the issue of only being permitted to apply $\otimes$-induction to terms of the top colour, which the variable bound in a ∂-abstraction is not. In Section 1.6 we will derive rules for hom that allow us to immediately pattern match on the argument. Given $g : \underline{A} \multimap (\underline{B} \multimap \underline{C})$ we will be able to write

$$\lambda(x \otimes y).g\langle x\rangle\langle y\rangle : \underline{A} \otimes \underline{B} \multimap \underline{C},$$

avoiding the need to manually 'copy' g. The round-trip on $\underline{A} \otimes \underline{B} \multimap \underline{C}$ will still be only propositionally equal to the identity, as we will only be able to derive a propositional η-principle for homs with a pattern-matched argument.

Remark 1.5.8. The rules that linearly bind a variable ($\multimap$-FORM and $\multimap$-INTRO) are the first rules we have encountered that can cause the ambient top colour to appear in a term. We are finally forced to explain the slight complication that arises when substituting for marked variables.

Suppose we are substituting for a marked variable usage $\underline{x}[a/x^{\mathfrak{c}}]$. Such a marked variable usage can occur in *any* palette, including when the top colour label is no longer $\mathfrak{c}$, so suppose the terms involved are

$$\begin{aligned} \mathfrak{t} \prec \Psi \mid \Gamma, x^{\mathfrak{c}} : A &\vdash \underline{x} : A \\ \mathfrak{c} \prec \Phi \mid \Gamma &\vdash a : A \end{aligned}$$

where A is a closed type. Typically $\mathfrak{t} \prec \Psi$ is some iterated extension of the palette $\mathfrak{c} \prec \Phi$.

As usual, we mark all the free variables in a giving

$$\mathfrak{c} \mid \underline{\Gamma} \vdash \underline{a} : A,$$

but then rename the top colour to the ambient top colour at the site of $\underline{x}$:

$$\mathfrak{t} \mid \underline{\Gamma} \vdash \underline{x}[a/x^{\mathfrak{c}}] :\equiv \underline{a}^{\mathfrak{t} \leftrightarrow \mathfrak{c}} : A$$

which is then silently weakened to

$$\mathfrak{t} \prec \Psi \mid \Gamma \vdash \underline{a}^{\mathfrak{t} \leftrightarrow \mathfrak{c}} : A$$

To demonstrate, suppose the top colour is $\mathfrak{r}$, and suppose we have a term $x : \underline{A}$. Then we can define a hom

$$\partial y^{\mathfrak{b}}.x \,{}_{\mathfrak{r}}{\otimes}_{\mathfrak{b}}\, y : \underline{B} \multimap (\underline{A} \otimes \underline{B})$$

(This is 'coevaluation' for the tensor-hom adjunction.) If this term is substituted for a marked variable $\underline{z}$ in a context with top colour $\mathfrak{y}$, the free variable x is marked and the top colour is renamed, giving

$$\underline{z}[\partial y^{\mathfrak{b}}.x \,{}_{\mathfrak{r}}{\otimes}_{\mathfrak{b}}\, y/z] :\equiv \partial y^{\mathfrak{b}}.\underline{x} \,{}_{\mathfrak{y}}{\otimes}_{\mathfrak{b}}\, y$$

We will see two examples of this in the following Lemma.

There is a second version of the tensor-hom adjunction, where the 'type of morphisms' is given by the underlying space of the ordinary function type:

Proposition 1.5.9. *There is a dull 'external' adjunction*

$$\natural\,((\underline{A} \otimes \underline{B}) \to \underline{C}) \simeq \natural\,(\underline{A} \to (\underline{B} \multimap \underline{C}))$$

or more dependently, let the top colour be purple and suppose we have dull types $\underline{A}$ and $\underline{B} : \underline{A} \to \mathcal{U}$, and a dull type family $\underline{C} : \textcircled{\scriptsize$\Sigma$}_{(\underline{x}:\underline{A})}\, \underline{B}(\underline{x}) \to \mathcal{U}$.

$$\natural\left(\prod_{(p:\textcircled{\scriptsize$\Sigma$}_{(\underline{x}:\underline{A})}\, \underline{B}(\underline{x}))} \underline{C}(p)\right) \simeq \natural\left(\prod_{(x:\underline{A})} \textcircled{\scriptsize$\Pi$}_{(y:\underline{B}(\underline{x}))} \underline{C}(x \otimes y)\right)$$

Proof. Say we have $\underline{f} : \prod_{(p:\textcircled{\scriptsize$\Sigma$}_{(\underline{x}:\underline{A})}\, \underline{B}(\underline{x}))} \underline{C}(p)$. To build a term of the type on the right, assume we have a term $x : \underline{A}$ and a term $y : \underline{B}(\underline{x})$ of a fresh colour. Then we can form $\underline{f}(x \otimes y) : \underline{C}(x \otimes y)$. As a term, we have defined the function

$$\lambda x.\partial y.\underline{f}(x \otimes y) : \prod_{(x:\underline{A})} \textcircled{\scriptsize$\Pi$}_{(y:\underline{B}(\underline{x}))} \underline{C}(x \otimes y)$$

In the other direction suppose we have $\underline{g} : \prod_{(x:\underline{A})} \textcircled{\scriptsize$\Pi$}_{(y:\underline{B}(\underline{x}))} \underline{C}(x \otimes y)$. To define a function of the type on the left, use $\otimes$-induction to get $x : \underline{A}$ and $y : \underline{B}(\underline{x})$. Then we can form $\underline{g}(x)\langle y \rangle : \underline{C}(x \otimes y)$. As a term:

$$\lambda p.\mathsf{let}\ x \otimes y = p\ \mathsf{in}\ \underline{g}(x)\langle y \rangle : \prod_{(p:\textcircled{\scriptsize$\Sigma$}_{(\underline{x}:\underline{A})}\, \underline{B}(\underline{x}))} \underline{C}(p)$$

Checking the round-trips, first we substitute the first term in for $\underline{g}$ in the latter. Note that the top colour at the usage of $\underline{g}$ is red, so the colour of x is changed:

$$\begin{aligned}
&\lambda p.\mathsf{let}\ x \otimes y = p\ \mathsf{in}\ \left(\lambda x.\partial y.\underline{f}(x \otimes y)\right)(x)\langle y \rangle \\
&\equiv \lambda p.\mathsf{let}\ x \otimes y = p\ \mathsf{in}\ \underline{f}(x \otimes y) \\
&= \lambda p.\underline{f}(p) \\
&\equiv \underline{f}
\end{aligned}$$

Similarly, the top colour at the point $\underline{f}$ is used is the tensor combination of purple and yellow, and so the colour of p is changed:

$$\begin{aligned}
&\lambda x.\partial y.\left(\lambda p.\mathsf{let}\ x \otimes y = p\ \mathsf{in}\ \underline{g}(x)\langle y \rangle\right)(x \otimes y) \\
&\equiv \lambda x.\partial y.\mathsf{let}\ x \otimes y = x \otimes y\ \mathsf{in}\ \underline{g}(x)\langle y \rangle \\
&\equiv \lambda x.\partial y.\underline{g}(x)\langle y \rangle \\
&\equiv \lambda x.\underline{g}(x) \\
&\equiv \underline{g}
\end{aligned}$$

□

This adjunction requires the $\natural$s guarding the two sides, and removing them makes the claim false. The functions $\underline{f}$ and $\underline{g}$ must be marked for the function applications to be well-formed: otherwise they would be purple terms, and the applications $f(x \otimes y)$ and $g(x)$ are not well formed.

Remark 1.5.10. A curious consequence of this equivalence is obtained by combining it with symmetry of $\otimes$:

$$\natural(\underline{A} \to (\underline{B} \multimap \underline{C})) \simeq \natural(\underline{A} \otimes \underline{B} \multimap \underline{C}) \simeq \natural(\underline{B} \otimes \underline{A} \multimap \underline{C}) \simeq \natural(\underline{B} \to (\underline{A} \multimap \underline{C}))$$

This is another reminder that linearity is not attached to a particular type in an expression; the appearance that '$\underline{B}$ is linear' on the left side and '$\underline{A}$ is linear' on the right side is an illusion.

Definition 1.5.11. Given two homs, we can compose them if they are provided to us with 'complementary colours', i.e., as a term of a $\otimes$-type.

$$\begin{aligned}
&\text{homcomp} &&: (\underline{B} \multimap \underline{C}) \otimes (\underline{A} \multimap \underline{B}) \to (\underline{A} \multimap \underline{C}) \\
&\text{homcomp}(g \otimes f) &&:\equiv \partial x.g\langle f\langle x\rangle\rangle
\end{aligned}$$

If we have two homs of the *same* colour $f : \underline{A} \multimap \underline{B}$ and $g : \underline{B} \multimap \underline{C}$, then there is no way to compose them without using at least one of them marked.

Definition 1.5.12. For any type $\underline{A}$ define the identity hom $\mathsf{homid} : \mathbb{S} \to (\underline{A} \multimap \underline{A})$ by

$$\mathsf{homid}(s) :\equiv \partial x.\mathsf{unitl}_{\underline{A}}(s \otimes x) : \underline{A} \multimap \underline{A}$$

Note that the term $\mathsf{unitr}_{\underline{A}}(s \otimes x)$ does not reduce. This function is equal to the transport of $\mathsf{unitl}_{\underline{A}} : \mathbb{S} \otimes \underline{A} \to \underline{A}$ across the adjunction of Proposition 1.5.9, using the fact that $\mathsf{unitl}_{\underline{A}}$ is a dull function.

Remark 1.5.13. We can also define composition for *dependent* homs. The types involved are somewhat restricted by the linearity requirements of the type former. To satisfy any curiosity, here is the best we can do: it is not pretty!

$$\begin{aligned}
&\underline{A} : \mathcal{U} \\
&\underline{B} : \underline{A} \to \mathcal{U} \\
&\underline{C} : \textstyle\prod_{(x:\underline{A})} \underline{B}(\underline{x}) \multimap \mathcal{U}
\end{aligned}$$

Then we can define

$$\begin{aligned}
&\text{homcomp} : \textstyle\prod_{\left(p:\left(\prod_{(x:\underline{A})} \textcircled{$\prod$}_{(y:\underline{B}(\underline{x}))} \underline{C}(\underline{x})\langle y\rangle\right)\otimes\left(\textcircled{$\prod$}_{(x:\underline{A})} \underline{B}(\underline{x})\right)\right)} \left(\mathsf{let}\ g \otimes f = p\ \mathsf{in}\ \textcircled{$\prod$}_{(x:\underline{A})}\underline{C}(\underline{x})\langle f\langle x\rangle\rangle\right) \\
&\text{homcomp}(g \otimes f) :\equiv \partial x.g(\underline{x})\langle f\langle x\rangle\rangle
\end{aligned}$$

The type former $\multimap$ is functorial in both components, in the sense that we can pre- and post-compose with ordinary functions.

Definition 1.5.14. Let the top colour be red. Given $f : \underline{A} \multimap \underline{B}$ and $\underline{g} : \underline{B} \to \underline{C}$, define $\mathsf{postcomp}(\underline{g}, f) : \underline{A} \multimap \underline{C}$ by

$$\mathsf{postcomp}(\underline{g}, f) :\equiv \partial x.\underline{g}(f\langle x\rangle)$$

More dependently, suppose $\underline{A} : \mathcal{U}$, $B : \underline{A} \multimap \mathcal{U}$ and $C : \textcircled{$\prod$}_{(x:\underline{A})} B\langle x\rangle \to \mathcal{U}$. If we have a hom $f : \textcircled{$\prod$}_{(x:\underline{A})} B\langle x\rangle$ and a dependent function $g : \textcircled{$\prod$}_{(x:\underline{A})} \prod_{(y:B\langle x\rangle)} C\langle x\rangle(y)$, define

$$\begin{aligned}
&\mathsf{postcomp}(g, f) : \textcircled{$\prod$}_{(x:\underline{A})} C\langle x\rangle(f\langle x\rangle) \\
&\mathsf{postcomp}(g, f) :\equiv \partial x.g\langle x\rangle(f\langle x\rangle)
\end{aligned}$$

Similarly, we can define precomposition:

Definition 1.5.15. Given $\underline{f} : \underline{A} \to \underline{B}$ and $g : \underline{B} \multimap \underline{C}$, define $\mathsf{precomp}(g, \underline{f}) : \underline{A} \multimap \underline{C}$ by

$$\mathsf{precomp}(g, \underline{f}) :\equiv \partial x.g\langle \underline{f}(x)\rangle$$

Precomposition does not have as pleasant a dependent generalisation: unlike postcomposition, we are forced to use various inputs marked. Let $\underline{A} : \mathcal{U}$, $\underline{B} : \underline{A} \to \mathcal{U}$ and $C : \prod_{(x:\underline{A})} \underline{B}(\underline{x}) \multimap \mathcal{U}$. If we have a function $\underline{f} : \prod_{(x:\underline{A})} \underline{B}(\underline{x})$ and a dependent hom $g : \prod_{(x:\underline{A})} \mathbb{\Pi}_{(y:\underline{B}(\underline{x}))} C(\underline{x})\langle y\rangle$, define

$$\begin{aligned}\mathsf{precomp}(g, \underline{f}) &: \mathbb{\Pi}_{(x:\underline{A})} C(\underline{x})\langle \underline{f}(x)\rangle\\ \mathsf{precomp}(g, \underline{f}) &:\equiv \partial x.g(\underline{x})\langle \underline{f}(x)\rangle\end{aligned}$$

Like Theorem 1.1.15 for ordinary functions, homs into a space hold no more information than functions of the base spaces.

Proposition 1.5.16. *For dull types $\underline{A}$ and $\underline{B}$*

$$(\underline{A} \multimap \natural\underline{B}) \simeq (\underline{A} \to \natural\underline{B})$$

or dependently, if $\underline{A} : \mathcal{U}$ and $\underline{B} : \underline{A} \to \mathcal{U}$:

$$\mathbb{\Pi}_{(x:\underline{A})} \natural\underline{B}(\underline{x}) \simeq \prod_{(x:\underline{A})} \natural\underline{B}(\underline{x})$$

Proof. If $f : \mathbb{\Pi}_{(x:\underline{A})} \natural\underline{B}(\underline{x})$ define $\lambda a.\underline{f}\langle \underline{a}\rangle : \prod_{(x:\underline{A})} \natural\underline{B}(\underline{x})$. For $g : \underline{A} \to \natural\underline{B}$ define $\partial a'.\underline{g}(\underline{a'}) : \underline{A} \multimap \natural\underline{B}$.
Round trips:

$$\lambda a.(\partial a'.\underline{g}(\underline{a'}))\langle \underline{a}\rangle \equiv \lambda a.\underline{g}(\underline{a}) \equiv \lambda a.g(a) \equiv g$$

and

$$\partial a'.(\lambda a.\underline{f}\langle \underline{a}\rangle)(\underline{a'}) \equiv \partial a'.\underline{f}\langle \underline{a'}\rangle \equiv \partial a'.f\langle a'\rangle \equiv f$$

where $\underline{g}(\underline{a'}) \equiv g(a)$ and $\underline{f}\langle \underline{a'}\rangle \equiv f\langle a'\rangle$ because they are both terms of type $\natural\underline{B}$. □

We have a kind of mixed currying for spaces.

Proposition 1.5.17. *If $\underline{A}$, $\underline{B}$ and $\underline{C}$ are dull types then:*

$$\begin{aligned}\underline{A} \multimap (\natural\underline{B} \to \underline{C}) &\simeq (\underline{A} \times \natural\underline{B}) \multimap \underline{C}\\ \natural\underline{A} \to (\underline{B} \multimap \underline{C}) &\simeq (\natural\underline{A} \times \underline{B}) \multimap \underline{C}\end{aligned}$$

Or dependently, if $\underline{A} : \mathcal{U}$, $\underline{B} : \underline{A} \to \mathcal{U}$ and $C : \mathbb{\Pi}_{(a:\underline{A})} (\natural\underline{B}(\underline{a}) \to \mathcal{U})$ then

$$\mathbb{\Pi}_{(a:\underline{A})} \prod_{(b:\natural\underline{B}(\underline{a}))} C\langle a\rangle(\underline{b}) \simeq \mathbb{\Pi}_{((a,b):\sum_{(a:\underline{A})} \natural\underline{B}(\underline{a}))} C\langle a\rangle(\underline{b})$$

and if $\underline{D} : \mathcal{U}$, $\underline{E} : \natural\underline{D} \to \mathcal{U}$ and $F : \prod_{(d:\natural\underline{D})} (\underline{E}(\underline{d}) \multimap \mathcal{U})$ then

$$\prod_{(d:\natural\underline{D})} \mathbb{\Pi}_{(e:\underline{E}(\underline{a}))} F(\underline{d})\langle e\rangle \simeq \mathbb{\Pi}_{((d,e):\sum_{(d:\natural\underline{D})} \underline{E}(\underline{d}))} F(\underline{d})\langle e\rangle$$

Proof. Straightforward, the maps are:

$$\begin{aligned}
f &\mapsto \partial(a,b).f\langle a\rangle(\underline{b}) : (\underline{A} \multimap (\natural\underline{B} \to \underline{C})) \to ((\underline{A} \times \natural\underline{B}) \multimap \underline{C}) \\
g &\mapsto \partial a.\lambda b.g\langle(a,\underline{b})\rangle : ((\underline{A} \times \natural\underline{B}) \multimap \underline{C}) \quad \to (\underline{A} \multimap (\natural\underline{B} \to \underline{C})) \\
h &\mapsto \partial(a,b).h(\underline{a})\langle b\rangle : (\natural\underline{A} \to (\underline{B} \multimap \underline{C})) \to ((\natural\underline{A} \times \underline{B}) \multimap \underline{C}) \\
k &\mapsto \lambda a.\partial b.k\langle(\underline{a},b)\rangle : ((\natural\underline{A} \times \underline{B}) \multimap \underline{C}) \quad \to (\natural\underline{A} \to (\underline{B} \multimap \underline{C}))
\end{aligned}$$

and the round-trips are definitionally the identity, because all types concerned have definitional η-rules. □

Like Lemma 1.3.24, if a hom and its argument are both dull then the split chosen when applying the hom is irrelevant.

Meta-Lemma 1.5.18. *For dull $\underline{h} : \bigoplus_{(x^\flat:\underline{A})} \underline{B}\langle x\rangle$ and $\underline{a} : \underline{B}$, given two splits $s_L \boxtimes s_R$ split and $t_L \boxtimes t_R$ split, there is an equality*

$$\underline{h}_{s_L}\langle\underline{a}\rangle_{s_R} = \underline{h}_{t_L}\langle\underline{a}\rangle_{t_R}$$

Proof. Any hom-application $h_{s_L}\langle a\rangle_{s_R}$ can be rewritten in the following way, by the computation rule for $\otimes$:

$$h_{s_L}\langle a\rangle_{s_R} \equiv \left(\text{let } h' {}_{\mathfrak{l}}\!\otimes_{\mathfrak{r}} a' = h \;{}_{s_L}\!\otimes_{s_R} a \text{ in } h'{}_{\mathfrak{l}}\langle a'\rangle_{\mathfrak{r}}\right)$$

And now we can apply split irrelevance for dull $\otimes$-pairs (Lemma 1.3.24):

$$\begin{aligned}
\underline{h}_{s_L}\langle\underline{a}\rangle_{s_R} &\equiv \left(\text{let } h' {}_{\mathfrak{l}}\!\otimes_{\mathfrak{r}} a' = \underline{h} \;{}_{s_L}\!\otimes_{s_R} \underline{a} \text{ in } h'{}_{\mathfrak{l}}\langle a'\rangle_{\mathfrak{r}}\right) \\
&= \left(\text{let } h' {}_{\mathfrak{l}}\!\otimes_{\mathfrak{r}} a' = \underline{h} \;{}_{t_L}\!\otimes_{t_R} \underline{a} \text{ in } h'{}_{\mathfrak{l}}\langle a'\rangle_{\mathfrak{r}}\right) \\
&\equiv \underline{h}_{t_L}\langle\underline{a}\rangle_{t_R}
\end{aligned}$$

□

1.5.3 Preservation of (Co)Limits

As a right adjoint, we would expect $\bigoplus$ to preserve Σ, and indeed it does:

Proposition 1.5.19 ($\multimap$ preserves Σ)**.** *Suppose $\underline{A}, \underline{B}, \underline{C} : \mathcal{U}$. Then:*

$$\underline{A} \multimap (\underline{B} \times \underline{C}) \simeq (\underline{A} \multimap \underline{B}) \times (\underline{A} \multimap \underline{C})$$

More dependently, suppose $\underline{C} : \underline{B} \to \mathcal{U}$, then:

$$\underline{A} \multimap \Sigma_{(y:\underline{B})}\underline{C}(y) \simeq \Sigma_{(g:\underline{A}\multimap\underline{B})} \bigoplus_{(x:\underline{A})}\underline{C}(\underline{g}\langle\underline{x}\rangle)$$

Most dependently of all, suppose we have type families $B : \underline{A} \multimap \mathcal{U}$ and $C : \bigoplus_{(x:\underline{A})}(B\langle x\rangle \to \mathcal{U})$, then:

$$\bigoplus_{(x:\underline{A})}\Sigma_{(y:B\langle x\rangle)}C\langle x\rangle(y) \simeq \Sigma_{(g:\bigoplus_{(x:\underline{A})} B\langle x\rangle)}\bigoplus_{(x:\underline{A})}C\langle x\rangle(g\langle x\rangle)$$

Proof. Defining round-trips, given a $f : \bigoplus_{(x:\underline{A})} \sum_{(y:B\langle x\rangle)} C\langle x\rangle(y)$ on the left, we can first define a hom of type $\bigoplus_{(x:\underline{A})} B\langle x\rangle$ by $\partial x.\mathsf{pr}_1(f\langle x\rangle)$. In the second component we can use

$$\partial x.\mathsf{pr}_2(f\langle x\rangle) : \bigoplus_{(x:\underline{A})} C\langle x\rangle(\mathsf{pr}_1(f\langle x\rangle))$$

so

$$(\partial x.\mathsf{pr}_1(f\langle x\rangle), \partial x.\mathsf{pr}_2(f\langle x\rangle)) : \sum_{(g:\bigoplus_{(x:\underline{A})} B\langle x\rangle)} \bigoplus_{(x:\underline{A})} C\langle x\rangle(g\langle x\rangle)$$

In the other direction, given

$$(g,h) : \sum_{(g:\bigoplus_{(x:\underline{A})} B\langle x\rangle)} \bigoplus_{(x:\underline{A})} C\langle x\rangle(g\langle x\rangle)$$

we have

$$\partial x.(g\langle x\rangle, h\langle x\rangle) : \bigoplus_{(x:\underline{A})} \sum_{(y:B\langle x\rangle)} C\langle x\rangle(y)$$

The round-trips are both definitionally equal to the identity, thanks to the η-laws for Σ and $\bigoplus$. □

To define the action of homs on paths, we have to be a little careful how the data is supplied:

Definition 1.5.20. For any $h : \underline{A} \multimap \underline{B}$, there is a hom

$$\mathsf{hap} : \bigoplus_{((x,y,p):\sum_{(x:\underline{A})} \sum_{(y:\underline{A})} x=y)} h\langle x\rangle = h\langle y\rangle$$

defined by pattern matching:

$$\mathsf{hap} :\equiv \partial(x, x, \mathsf{refl}_x).\mathsf{refl}_{h\langle x\rangle}$$

Because of the 'external adjunction' of Proposition 1.5.9, we should not be too surprised that hom commutes with colimits in the domain:

Theorem 1.5.21. *Suppose* $\underline{A}, \underline{B}, \underline{S} : \mathcal{U}$ *with* $\underline{f} : \underline{S} \to \underline{A}$ *and* $\underline{g} : \underline{S} \to \underline{B}$. *Then given the data*

$$\begin{aligned} h &: \underline{A} \multimap \underline{C} \\ k &: \underline{B} \multimap \underline{C} \\ H &: \bigoplus_{(s:\underline{S})} \mathsf{precomp}(h, \underline{f})\langle s\rangle = \mathsf{precomp}(k, \underline{g})\langle s\rangle \end{aligned}$$

there is an induced hom

$$\mathsf{lind}_+(h, k, H) : \underline{A} +_{\underline{S}} \underline{B} \multimap \underline{C}$$

such that

$$\begin{aligned} \mathsf{lind}_+(h,k,H)\langle \mathsf{inl}(a)\rangle &\equiv h\langle a\rangle \\ \mathsf{lind}_+(h,k,H)\langle \mathsf{inr}(b)\rangle &\equiv k\langle b\rangle \\ \mathsf{hap}(\mathsf{lind}_+(h,k,H))\langle \mathsf{glue}(s)\rangle &= H\langle s\rangle \end{aligned}$$

There is a nice alignment of dullness conditions in this statement: precomp can only be defined to use the second argument in a dull way.

Proof. Begin by using ∂-introduction to give $t : \underline{A} +_{\underline{S}} \underline{B}$. Then we can use ordinary $+$-induction to form a function

$$\begin{aligned}\underline{I}(t) :\equiv \mathsf{ind}_+(&a.\partial(h', k', H').h'\langle a\rangle,\\ &b.\partial(h', k', H').k'\langle b\rangle,\\ &s.\mathsf{homext}(\partial(h', k', H').H'\langle s\rangle),\\ &t)\\ &: \left(\textstyle\sum_{(h':\underline{A}\multimap\underline{C})}\sum_{(k':\underline{B}\multimap\underline{C})}\bigoplus_{(s:\underline{S})}\mathsf{precomp}(h', \underline{f})\langle s\rangle = \mathsf{precomp}(k', \underline{g})\langle s\rangle\right) \multimap \underline{C}\end{aligned}$$

which can then be applied to the supplied data to give $\underline{I}(t)\langle h, k, H\rangle : \underline{C}$. The computation rules for inl and inr hold by the computation rules for $\multimap$ and $+$: the argument is substituted in for t and then ordinary ind_+ computes. For glue, we need a quick fact about commuting hap with hom-application.

Suppose we have dull types $\underline{A}, \underline{B}, \underline{C}$, a function $\underline{f} : \underline{A} \to \underline{B} \multimap \underline{C}$, a term $b : \underline{B}$ and a path $p : a = a'$. Then there is a path

$$\mathsf{hap}_{\partial x.\underline{f}(x)\langle b\rangle}\langle p\rangle = \mathsf{homapp}(\mathsf{ap}_{\underline{f}}(p))\langle b\rangle$$

given by pattern matching on $b \otimes p$ as $b \otimes \mathsf{refl}_a$, so that both sides reduce to $\mathsf{refl}_{\underline{f}(a)\langle b\rangle}$.

In our case, this gives

$$\begin{aligned}&\mathsf{hap}(\mathsf{lind}_+(h, k, H))\langle \mathsf{glue}(s)\rangle\\ &\equiv \mathsf{hap}_{\partial t.\underline{I}(t)\langle h,k,H\rangle}\langle \mathsf{glue}(s)\rangle\\ &= \mathsf{homapp}(\mathsf{ap}_{\underline{I}}(\mathsf{glue}(s)))\langle h, k, H\rangle\\ &= \mathsf{homapp}(\mathsf{homext}(\partial(h', k', H').H'\langle s\rangle))\langle h, k, H\rangle\\ &= (\partial(h', k', H').H'\langle s\rangle)\langle h, k, H\rangle\\ &\equiv H\langle s\rangle\end{aligned}$$

□

1.5.4 Axiomatic Closedness of the Modality

In our primary models of interest, $\natural$ is a closed monoidal functor: it takes $\bigoplus$-types to Π-types. This does not follow from the rules of the theory, so we add it as an axiom:

Axiom C. *$\natural$ is a closed functor, i.e. for $\underline{A} : \mathcal{U}$ and $\underline{B} : \underline{A} \multimap \mathcal{U}$, the map*

$$\mathsf{dist}_{\underline{A},\underline{B}} : \natural\left(\textstyle\bigoplus_{(a:\underline{A})}\underline{B}\langle a\rangle\right) \to \left(\prod_{(x:\natural\underline{A})}\natural(\underline{B}\langle \underline{x}_\natural\rangle)\right)$$

defined by

$$\mathsf{dist}_{\underline{A},\underline{B}}(\underline{h}^\natural)(\underline{x}^\natural) :\equiv (\underline{h}_1\langle \underline{x}\rangle_1)^\natural$$

is an equivalence.

Non-dependently this is just $\natural(\underline{A} \multimap \underline{B}) \simeq (\natural\underline{A} \to \natural\underline{B})$.

Remark 1.5.22. There is a map the other way

$$\left(\prod_{(x:\natural\underline{A})} \natural(\underline{B}\langle \underline{x}_\natural \rangle)\right) \to \natural\left(\textstyle\bigoplus\!\!\!\!\prod_{(a:\underline{A})} \underline{B}\langle \underline{a} \rangle\right)$$

given by

$$\mathsf{undist}_{\underline{A},\underline{B}}(f) :\equiv (\partial a.\underline{f}(\underline{a}^\natural)_\natural)^\natural$$

which is always a section of $\mathsf{dist}_{\underline{A},\underline{B}}$. Without the axiom, the round-trip on the $\multimap$-type is nearly but not quite the identity:

$$(\partial a.(\lambda \underline{x}.\underline{h}_1\langle \underline{x}_\natural \rangle_1{}^\natural)(\underline{a}^\natural)_\natural)^\natural \equiv (\partial a.\underline{h}_1\langle \underline{a}^\natural{}_\natural \rangle_1{}^\natural{}_\natural)^\natural \equiv (\partial a.\underline{h}_1\langle \underline{a} \rangle_1)^\natural$$

but the η-law for homs *doesn't* apply: the argument $\underline{a}$ is marked. The axiom makes this hom equal to $\underline{h}^\natural$, so a syntactic gloss on the axiom is that 'any dull hom uses its argument dull'.

We can apply this to the hom $\underline{B} : \underline{A} \multimap \mathcal{U}$ used in the dependent type-former itself: the map

$$\natural\left(\textstyle\bigoplus\!\!\!\!\prod_{(a:\underline{A})} \underline{B}\langle a \rangle\right) \to \natural\left(\textstyle\bigoplus\!\!\!\!\prod_{(a:\underline{A})} \underline{B}\langle \underline{a} \rangle\right)$$

is an equivalence, so we could have phrased the axiom using the type on the left without increasing its power.

Axiom C is not provable in the bare type theory. Any proof that it is could be translated to a similar proof for Π simply by replacing each instance of a $\bigoplus\!\!\!\!\prod$-type with a Π-type, and it is certainly not the case that $\natural$ is a closed functor for Π:

Proposition 1.5.23. *If the canonical map*

$$\begin{aligned} &\mathsf{func}_{\underline{A},\underline{B}} : \natural(\underline{A} \to \underline{A}) \to (\natural\underline{A} \to \natural\underline{A}) \\ &\mathsf{func}_{\underline{A},\underline{B}}(\underline{f}^\natural)(\underline{x}^\natural) :\equiv (\underline{f}(\underline{x}))^\natural \end{aligned}$$

is an equivalence, then the $\natural$ modality is equal to the identity modality.

Proof. $\mathsf{func}_{\underline{A},\underline{B}}$ already has a section, given by $f \mapsto (\lambda x.\underline{f}(\underline{x}^\natural)_\natural)^\natural$. If the composite the other way

$$\natural(\underline{A} \to \underline{A}) \to (\natural\underline{A} \to \natural\underline{A}) \to \natural(\underline{A} \to \underline{A})$$

is the identity, then applying this composite to $\mathsf{id}_{\underline{A}}{}^\natural$ yields $(\lambda x.\underline{x})^\natural$, so $\mathsf{id}_{\underline{A}} = (\lambda x.\underline{x})$.

Therefore, for any $a : \underline{A}$ we have $a = \underline{a}$ by applying the two equal functions, and so $\natural\underline{A} \simeq \underline{A}$ by Corollary 1.1.13. If this holds for any type, in particular it holds for the universe, so for any $A : \mathcal{U}$ we have $A = \underline{A}$, and so also $\natural\underline{A} \simeq A$. And so the $\natural$ modality is equal to the identity, because modalities are determined by their modal types [RSS20, Theorem 1.12]. □

The axiom has a number of important consequences. We mark any result that requires Axiom C with $\{C\}$.

Proposition {C} 1.5.24. *Writing the top colour as purple, for any space $\underline{X}$ and dull type $\underline{A}$ the maps*

$$(x \otimes y) \mapsto (\underline{x}, \underline{y}^\natural) : \underline{X} \otimes \underline{A} \to \underline{X} \times \natural\underline{A}$$
$$(x, n) \mapsto (\underline{x}\,{}_{\mathfrak{p}}{\otimes}_1\, \underline{n}_\natural) : \underline{X} \times \natural\underline{A} \to \underline{X} \otimes \underline{A}$$

are inverse equivalences, so in particular $1 \otimes \underline{A} \simeq \natural\underline{A}$.

Proof. There is a typical adjointness argument:

$$\begin{aligned}\natural(\underline{X} \otimes \underline{A} \to \underline{Y}) &\simeq \natural(\underline{X} \to (\underline{A} \multimap \underline{Y}))\\ &\simeq \natural(\natural\underline{X} \to (\underline{A} \multimap \underline{Y}))\\ &\simeq \natural(\underline{X} \to \natural(\underline{A} \multimap \underline{Y}))\\ &\simeq \natural(\underline{X} \to (\natural\underline{A} \to \natural\underline{Y}))\\ &\simeq \natural(\underline{X} \times \natural\underline{A} \to \natural\underline{Y})\\ &\simeq \natural(\underline{X} \times \natural\underline{A} \to \underline{Y})\end{aligned}$$

But there is an arguably more direct method. The round-trip on $\underline{X} \times \natural\underline{A}$ is clearly the identity, as both factors are spaces. For the other composite, we must show that given $x : \underline{X}$ and $y : \underline{A}$, there is a path $(\underline{x}\,{}_{\mathfrak{p}}{\otimes}_1\, \underline{y}) = (x \otimes y)$.

Note that for fixed $\underline{x} : \underline{X}$, the (red) hom

$$\partial a.\underline{x}\,{}_{\mathfrak{r}}{\otimes}_{\mathfrak{y}}\, a : \underline{A} \multimap (\underline{X} \otimes \underline{A})$$

is dull. By the axiom, it is therefore equal to

$$\partial a.\underline{x}\,{}_{\mathfrak{r}}{\otimes}_1\, \underline{a}$$

and we can make the following calculation:

$$x\,{}_{\mathfrak{r}}{\otimes}_{\mathfrak{b}}\, y = \underline{x}\,{}_{\mathfrak{r}}{\otimes}_{\mathfrak{b}}\, y \equiv (\partial a.\underline{x}\,{}_{\mathfrak{r}}{\otimes}_{\mathfrak{y}}\, a)\langle y\rangle = (\partial a.\underline{x}\,{}_{\mathfrak{r}}{\otimes}_1\, \underline{a})\langle y\rangle \equiv \underline{x}\,{}_{\mathfrak{r}}{\otimes}_1\, \underline{y}$$

and finally $\underline{x}\,{}_{\mathfrak{r}}{\otimes}_1\, \underline{y} = \underline{x}\,{}_{\mathfrak{p}}{\otimes}_1\, \underline{y}$ by Lemma 1.3.24. □

Proposition 1.5.16 told us that homs into spaces are given by ordinary functions of the base spaces. It follows from Axiom C that the same is true for homs out of spaces. This is quite different to the ordinary function types, where $\natural\underline{A} \to \underline{B}$ is typically *not* a space.

Proposition {C} 1.5.25. *For $\underline{A}$ a type and $\underline{X}$ a space, the map*

$$h \mapsto (\lambda x.\underline{h}_1\langle\underline{x}\rangle_1)^\natural : (\underline{X} \multimap \underline{A}) \to \natural(\underline{X} \to \underline{A})$$

is an equivalence.

Proof. There is an adjointness argument:

$$\begin{aligned}\natural(\underline{B} \to (\underline{X} \multimap \underline{A})) &\simeq \natural(\underline{B} \otimes \underline{X} \to \underline{A})\\ &\simeq \natural(\natural\underline{B} \times \underline{X} \to \underline{A})\\ &\simeq \natural(\natural\underline{B} \to (\underline{X} \to \underline{A}))\\ &\simeq \natural(\underline{B} \to \natural(\underline{X} \to \underline{A}))\end{aligned}$$

□

Corollary {C} 1.5.26. $1 \multimap \underline{A} \simeq \natural \underline{A}$ □

This Corollary provides an alternative perspective on the self-adjointness of $\natural$: together with Proposition 1.5.9, we see

$$\natural(\natural \underline{A} \to \underline{B}) \simeq \natural(\underline{A} \otimes 1 \to \underline{B}) \simeq \natural(\underline{A} \to (1 \multimap \underline{B})) \simeq \natural(\underline{A} \to \natural \underline{B})$$

Remark 1.5.27. Proposition 1.5.24 is reminiscent of 'dull split irrelevance' (Lemma 1.3.24) which stated that if the two sides of a $\otimes$-INTRO are dull, then the choice of split used to form the pair is irrelevant. This proposition is an upgraded version of that idea: it is enough for one side to be dull, and then the split becomes irrelevant. If we did manage to build this one-sided irrelevance into the type theory definitionally, then we could *define* $\natural \underline{A} :\equiv 1 \multimap \underline{A}$ thanks to the previous Corollary.

1.5.5 Univalence Implies Hom Extensionality

Let us write the top colour as red throughout this section. Recall that for $f, g : \bigoplus_{(x:\underline{A})} B\langle x \rangle$, there is a canonical function

$$\mathsf{homapp}(f,g) : (f = g) \to \bigoplus_{(x:\underline{A})} f\langle x \rangle = g\langle x \rangle$$

given by path induction:

$$\mathsf{homapp}(f,f)(\mathsf{refl}_f) :\equiv \partial x.\mathsf{refl}_{f\langle x \rangle}$$

Axiom Homext. *For any $f, g : \bigoplus_{(x:\underline{A})} B\langle x \rangle$, the function* $\mathsf{homapp}(f,g)$ *is an equivalence.*

Univalence implies hom extensionality. We adapt the proof of ordinary function extensionality given in the HoTT-Agda library [HoTTAgda, Funext], checking carefully that the linearity constraints are satisfied. We do not require Axiom C in this section.

As in the proof of ordinary function extensionality, our argument proceeds in two steps. First we show that univalence implies a weak form of hom extensionality, and then that weak hom extensionality implies hom extensionality.

Lemma {UA} 1.5.28. *Without assuming hom extensionality, any equivalence $e : B \simeq B'$ induces an equivalence $(\underline{A} \multimap \underline{B}) \simeq (\underline{A} \multimap \underline{B'})$ by postcomposition with $\underline{e}$, in the sense of Definition 1.5.14.*

Proof. The equivalence e is the image of some $p : B = B'$ under univalence. By path induction, assume $p \equiv \mathsf{refl}_B$, so $e = \mathsf{id}_B$. Then postcomposition with $\underline{e}$ is an equivalence, by transporting the proof that postcomposition with $\mathsf{id}_{\underline{B}}$ is the identity across the equality $\underline{e} = \mathsf{id}_{\underline{B}}$. □

Proposition {UA} 1.5.29. *'Naive' hom extensionality holds when the codomain is non-dependent and dull: for any $f, g : \underline{A} \multimap \underline{B}$ there is a map $\left(\bigoplus_{(x:\underline{A})} f\langle x \rangle = g\langle x \rangle\right) \to (f = g)$.*

Proof. Given two homs $f, g : \underline{A} \multimap \underline{B}$, and $h : \bigoplus_{(x:\underline{A})} f\langle x \rangle = g\langle x \rangle$, define two homs

$$\begin{aligned} d, e &: \underline{A} \multimap \left(\textstyle\sum_{(y:\underline{B})} \sum_{(y':\underline{B})} y = y'\right) \\ d &:\equiv \partial x.(f\langle x \rangle, f\langle x \rangle, \mathsf{refl}_{f\langle x \rangle}) \\ e &:\equiv \partial x.(f\langle x \rangle, g\langle x \rangle, h\langle x \rangle) \end{aligned}$$

We can now pull the same trick used for ordinary functions. The projection

$$\mathsf{pr}_1 : \textstyle\sum_{(y:\underline{B})}\sum_{(y':\underline{B})}(y = y') \to \underline{B}$$

is an equivalence, and so by the previous lemma there is an equivalence

$$\mathsf{postcomp}(\mathsf{pr}_1, -) : \left[\underline{A} \multimap \left(\textstyle\sum_{(y:\underline{B})}\sum_{(y':\underline{B})} y = y'\right)\right] \to [\underline{A} \multimap \underline{B}]$$

The homs d and e become equal under this postcomposition, because

$$\begin{aligned}\mathsf{postcomp}(\mathsf{pr}_1, d) &\equiv \partial x.\mathsf{pr}_1(d\langle x\rangle) \equiv \partial x.\mathsf{pr}_1(f\langle x\rangle, f\langle x\rangle, \mathsf{refl}_{f\langle x\rangle}) \equiv \partial x.f\langle x\rangle \equiv f\\ \mathsf{postcomp}(\mathsf{pr}_1, e) &\equiv \partial x.\mathsf{pr}_1(e\langle x\rangle) \equiv \partial x.\mathsf{pr}_1(f\langle x\rangle, g\langle x\rangle, h\langle x\rangle) \quad \equiv \partial x.f\langle x\rangle \equiv f\end{aligned}$$

so in fact $d = e$.

Now, ap of $\mathsf{postcomp}(\mathsf{pr}_2, -)$ on this path yield a path $\mathsf{postcomp}(\mathsf{pr}_2, d) = \mathsf{postcomp}(\mathsf{pr}_2, e)$, whose sides similarly compute to f and g. □

We now show that univalence implies weak hom extensionality.

Definition 1.5.30. The *weak hom extensionality principle* asserts that there is a function

$$\textstyle\bigcirc\!\!\!\!\prod_{(x:\underline{A})}\mathsf{isContr}(B\langle x\rangle) \to \mathsf{isContr}\left(\bigcirc\!\!\!\!\prod_{(x:\underline{A})} B\langle x\rangle\right)$$

for any linear family $B : \underline{A} \multimap \mathcal{U}$.

First, an easy Lemma:

Lemma 1.5.31. *For any dull type $\underline{A}$, the hom type $\underline{A} \multimap 1$ is contractible.*

Proof. For any hom $f : \underline{A} \multimap 1$ we have $f \equiv \partial x.f\langle x\rangle \equiv \partial x.\star$ by η-expansion for $\multimap$ and 1. □

Proposition {UA} 1.5.32. *Weak hom extensionality holds.*

Proof. Suppose we have $B : \underline{A} \multimap \mathcal{U}$ and $w : \bigcirc\!\!\!\!\prod_{(x:\underline{A})} \mathsf{isContr}(B\langle x\rangle)$. From w and univalence we can build a term of $\bigcirc\!\!\!\!\prod_{(x:\underline{A})}(B\langle x\rangle = 1)$. Then naive hom extensionality (Proposition 1.5.29) for the type $\underline{A} \multimap \mathcal{U}$ gives a path $p : B = (\partial x.1)$.

Let $F(B') :\equiv \bigcirc\!\!\!\!\prod_{(x:\underline{A})} B'\langle x\rangle$. Then

$$\mathsf{ap}_F(p) : \left(\textstyle\bigcirc\!\!\!\!\prod_{(x:\underline{A})} B\langle x\rangle\right) = (\underline{A} \multimap 1)$$

Transporting the proof of $\mathsf{isContr}(\underline{A} \multimap 1)$ from the previous Lemma along this path gives a proof of $\mathsf{isContr}\left(\bigcirc\!\!\!\!\prod_{(x:\underline{A})} B\langle x\rangle\right)$ as desired. □

This ap makes sense because $\multimap$-FORM is a *function* of its inputs, not a hom, as highlighted in Remark 1.5.3.

Theorem 1.5.33. *Weak hom extensionality implies hom extensionality.*

Proof. We wish to show that for all $f, g : \textcircled{\small$\prod$}_{(x:\underline{A})} B\langle x\rangle$, the map $\mathsf{homapp}(f,g) : (f = g) \to \textcircled{\small$\prod$}_{(x:\underline{A})} f\langle x\rangle = g\langle x\rangle$ is an equivalence. Fixing an f and working fibrewise, it is enough to show that the map

$$\left(\Sigma_{(g:\textcircled{\scriptsize$\prod$}_{(x:\underline{A})} B\langle x\rangle)}(f = g)\right) \to \left(\Sigma_{(g:\textcircled{\scriptsize$\prod$}_{(x:\underline{A})} B\langle x\rangle)} \textcircled{\small$\prod$}_{(x:\underline{A})} f\langle x\rangle = g\langle x\rangle\right)$$

given by $\lambda(g,p).(g, \mathsf{homapp}(f,g)(p))$ is an equivalence. The type on the left of the arrow is contractible by singleton contractibility, so it just remains to show that

$$\Sigma_{(g:\textcircled{\scriptsize$\prod$}_{(x:\underline{A})} B\langle x\rangle)} \textcircled{\small$\prod$}_{(x:\underline{A})} f\langle x\rangle = g\langle x\rangle$$

is also contractible. By Proposition 1.5.19, this type is equivalent to

$$\textcircled{\small$\prod$}_{(x:\underline{A})} \Sigma_{(y:B\langle x\rangle)} f\langle x\rangle = y$$

This is a hom into a family of contractible types, so is contractible by the weak hom extensionality principle. □

Corollary {UA} 1.5.34. *Hom extensionality holds.* □

1.5.6 Linear Equivalence is Ordinary Equivalence

One may ask whether the hom type grants access to any new notion of equivalence between types. One option is to ape the ordinary definition of an equivalence, with homs replacing the functions. Let $\mathfrak{r}$ denote the top colour. To have access to an identity hom (Definition 1.5.12), we must have access to a term of $\mathbb{S}$, and to compose two homs (Definition 1.5.11), we must have them *tensored* together, leading to the following awkward definition.

Definition 1.5.35. For a dull map $\underline{h} : \mathbb{S} \to \underline{A} \multimap \underline{B}$, a *left inverse* of $\underline{h}$ is a dull map $\underline{k} : \mathbb{S} \to \underline{B} \multimap \underline{A}$ such that for all $s : \mathbb{S}$,

$$\mathrm{homcomp}(\underline{h}(s)\ {}_{\mathfrak{r}}\!\otimes_{\varnothing}\ \underline{k}(\mathbin{\Box}))= \mathrm{homid}(s)$$

and similarly for *right inverse*.

Definition 1.5.36. For dull types $\underline{A}$ and $\underline{B}$, a map $\underline{h} : \mathbb{S} \to \underline{A} \multimap \underline{B}$ is a *linear equivalence* if it has a left inverse and a right inverse.

Remark 1.5.37. This precise notion of linear equivalence was studied in [Lun18, §4.3]. In that theory, there is a separation between ordinary and linear types, and a corresponding separation of ordinary and linear and context zones. A linear equivalence (as defined there) involves homs *with an empty linear context*. An empty linear context represents the monoidal unit, and so this matches our definition.

Under the equivalence of Proposition 1.5.9 and using $\mathsf{unitl}_{\underline{A}}$, dull maps $\underline{h} : \mathbb{S} \to \underline{A} \multimap \underline{B}$ correspond to dull maps $\underline{f} : \underline{A} \to \underline{B}$, by

$$\phi(\underline{h})(a) :\equiv \underline{h}(\mathbin{\Box})_{\varnothing}\langle a\rangle_{\mathfrak{r}}$$

Proposition 1.5.38. *A hom $\underline{h} : \mathbb{S} \to \underline{A} \multimap \underline{B}$ is a linear equivalence iff the associated map $\phi(\underline{h}) : \underline{A} \to \underline{B}$ is an ordinary equivalence.*

Proof. Suppose $\underline{k} : \mathbb{S} \to \underline{B} \multimap \underline{A}$ is a left-inverse, so

$$\lambda s.\mathrm{homcomp}(\underline{h}(\text{¤})\ {}_{\varnothing}{\otimes}_{\mathfrak{r}}\ \underline{k}(s)) = \mathrm{homid}_{\underline{A}}$$

Transposing across the adjunction, this is equivalent to an equality

$$\lambda p.\mathsf{let}\ s \otimes a = p\ \mathsf{in}\ \mathrm{homcomp}(\underline{h}(\text{¤})\ {}_{\varnothing}{\otimes}_{\mathfrak{r}}\ \underline{k}(s))\langle a \rangle = \mathsf{unitl}_{\underline{A}}$$

of function $\mathbb{S} \otimes \underline{A} \to \underline{A}$. Working on the left-hand side, inlining the definition of homcomp gives

$$\lambda p.\mathsf{let}\ s \otimes a = p\ \mathsf{in}\ \underline{h}(\text{¤})_{\varnothing}\langle \underline{k}(s)_{\mathfrak{r}}\langle a \rangle_{\mathfrak{y}} \rangle_{\mathfrak{r}\otimes\mathfrak{y}}$$

which by the uniqueness principle for the unitor, is equal to

$$\lambda p.\mathsf{let}\ \text{¤}\ {}_{\varnothing}{\otimes}_{\mathfrak{y}}\ a = p\ \mathsf{in}\ \underline{h}(\text{¤})_{\varnothing}\langle \underline{k}(\text{¤})_{\varnothing}\langle a \rangle_{\mathfrak{y}} \rangle_{\mathfrak{y}}$$

But this is now precisely equal to

$$\phi(\underline{h}) \circ \phi(\underline{k}) \circ \mathsf{unitl}_{\underline{A}}$$

So $\underline{k} : \mathbb{S} \to \underline{B} \multimap \underline{A}$ is a left-inverse to $\underline{h} : \mathbb{S} \to \underline{A} \multimap \underline{B}$ iff $\phi(\underline{h})$ is a left-inverse to $\phi(\underline{k})$ in the ordinary sense.

The same is true for right-inverse and we are done. □

1.6 Pattern Matching

As promised in Section 1.3.4, we can use $\multimap$ to derive some stronger induction principles for $\otimes$, for example allowing us to assume a term $s : (\underline{A} \otimes \underline{B}) \otimes \underline{C}$ is of the form $s \equiv (x \otimes y) \otimes z$. Put simply, the idea is to use $\multimap$ to shuffle pieces of the palette 'out of the way', so that we can apply $\otimes$-induction at the top level. We first show that the induction principles that we used in previous sections (Figure 1.15) are derivable in this way.

This strategy does not work for all the induction principles we hope for, so we then describe an inductive collection of patterns [Coq92]. These will allow us to deconstruct any nested combination of $1/\Sigma/\mathsf{Id}/\natural/\mathbb{S}/\otimes$-types simultaneously.

1.6.1 Derivable Eliminators

As an easy warm-up, let us a show that an induction principle for $(A \otimes B) \otimes C$ is derivable, where there is no internal dependency of A, B and C, and the motive D is a dull type.

Proposition 1.6.1 (Simple Left-Associated Elim). *The rule*

$$\otimes\text{-ELIM-SIMPLE-TRIPLE}\ \frac{\begin{array}{c}\mathfrak{t} \mid \underline{\Gamma} \vdash D\ \mathsf{type} \\ \mathfrak{t} \prec (\mathfrak{l} \prec \mathfrak{l}' \otimes \mathfrak{r}') \otimes \mathfrak{r} \mid \underline{\Gamma}, x^{\mathfrak{l}'} : A, y^{\mathfrak{r}'} : B, z^{\mathfrak{r}} : C \vdash d : D \\ \mathfrak{t} \prec \Phi \mid \Gamma \vdash s : (A \otimes B) \otimes C\end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \mathsf{let}\ (x\ {}_{\mathfrak{l}'}{\otimes}_{\mathfrak{r}'}\ y)\ {}_{\mathfrak{l}}{\otimes}_{\mathfrak{r}}\ z = s\ \mathsf{in}\ d : D}$$

$$\begin{aligned}&(\text{let } (x \,{}_{\mathfrak{l}'}\!\otimes_{\mathfrak{r}'} y) \,{}_{\mathfrak{l}}\!\otimes_{\mathfrak{r}} z = (a \,{}_{s_{LL}}\!\otimes_{s_{LR}} b) \,{}_{s_L}\!\otimes_{s_R} c \text{ in } d)\\ &\quad\equiv d[s_{LL}/\mathfrak{l}' \otimes s_{LR}/\mathfrak{r}' \mid a/x, b/y][c/z]\llbracket \mathfrak{t} \prec s_L \boxtimes s_R/\mathfrak{t}\rrbracket\end{aligned}$$

is derivable.

Proof. First, we can use d to form a dull term

$$\mathfrak{l}' \mid \underline{\Gamma} \vdash \lambda x.\partial^{\mathfrak{l}} y^{\mathfrak{r}'}.\partial^{\mathfrak{t}} z^{\mathfrak{r}}.d : A \to B \multimap C \multimap D$$

To make things line up better below, let us rename $\mathfrak{t}$ to $\mathfrak{t}'$, and $\mathfrak{l}'$ to $\mathfrak{t}$ (by the RECOLOUR rule).

$$\mathfrak{t} \mid \underline{\Gamma} \vdash \lambda x.\partial^{\mathfrak{l}} y^{\mathfrak{r}'}.\partial^{\mathfrak{t}'} z^{\mathfrak{r}}.d^{\mathfrak{t}'\leftrightarrow\mathfrak{t},\mathfrak{t}\leftrightarrow\mathfrak{l}'} : A \to B \multimap C \multimap D$$

It will suffice to produce a function

$$T : (A \to B \multimap C \multimap D) \to ((A \otimes B) \otimes C \to D)$$

because we can then define the triple eliminator as

$$(\text{let } (x \,{}_{\mathfrak{l}'}\!\otimes_{\mathfrak{r}'} y) \,{}_{\mathfrak{l}}\!\otimes_{\mathfrak{r}} z = s \text{ in } d) :\equiv T(\lambda x.\partial^{\mathfrak{l}} y^{\mathfrak{r}'}.\partial^{\mathfrak{t}'} z^{\mathfrak{r}}.d^{\mathfrak{t}'\leftrightarrow\mathfrak{t},\mathfrak{t}\leftrightarrow\mathfrak{l}'})(s)$$

This T can be defined by two uses of Proposition 1.5.9, but because its computational behaviour is important for achieving the correct computation rule, let us define it directly. Suppose we have such an $f : A \to B \multimap C \multimap D$ and a term $s : (A \otimes B) \otimes C$. First apply $\otimes$-induction to s giving $p^{\mathfrak{l}} : A \otimes B$ and $z_0^{\mathfrak{r}_0} : C$. We cannot do $\otimes$-induction on p immediately, but we can use p to build a hom and then apply it to z_0. To build a purple coloured hom, do a $\otimes$-induction on p to get $x^{\mathfrak{l}'}$ and $y^{\mathfrak{r}'}$, then form

$$\partial z.\underline{f}(x)\langle y\rangle\langle z\rangle : C \multimap D$$

Finally, apply this hom to z_0. In all:

$$\text{let } p \,{}_{\mathfrak{l}}\!\otimes_{\mathfrak{r}_0} z_0 = s \text{ in } \Big(\text{let } x \,{}_{\mathfrak{l}'}\!\otimes_{\mathfrak{r}'} y = p \text{ in } (\partial^{\mathfrak{t}'} z^{\mathfrak{r}}.\underline{f}(x)\langle y\rangle\langle z\rangle)\Big)\,{}_{\mathfrak{l}}\langle z_0\rangle_{\mathfrak{r}_0} : D$$

Now, we can calculate

$$\begin{aligned}&T(\lambda x.\partial^{\mathfrak{l}} y^{\mathfrak{r}'}.\partial^{\mathfrak{t}'} z^{\mathfrak{r}}.d^{\mathfrak{t}'\leftrightarrow\mathfrak{t},\mathfrak{t}\leftrightarrow\mathfrak{l}'})((a \,{}_{s_{LL}}\!\otimes_{s_{LR}} b) \,{}_{s_L}\!\otimes_{s_R} c)\\
&\equiv \text{let } p \,{}_{\mathfrak{l}}\!\otimes_{\mathfrak{r}_0} z_0 = (a \,{}_{s_{LL}}\!\otimes_{s_{LR}} b) \,{}_{s_L}\!\otimes_{s_R} c \text{ in } \Big(\text{let } x \,{}_{\mathfrak{l}'}\!\otimes_{\mathfrak{r}'} y = p \text{ in } (\partial^{\mathfrak{t}'} z^{\mathfrak{r}}.\underline{f}(x)\langle y\rangle\langle z\rangle)\Big)\,{}_{\mathfrak{l}}\langle z_0\rangle_{\mathfrak{r}_0}\\
&\equiv \text{let } p \,{}_{\mathfrak{l}}\!\otimes_{\mathfrak{r}_0} z_0 = (a \,{}_{s_{LL}}\!\otimes_{s_{LR}} b) \,{}_{s_L}\!\otimes_{s_R} c \text{ in } \Big(\text{let } x \,{}_{\mathfrak{l}'}\!\otimes_{\mathfrak{r}'} y = p \text{ in } (\partial^{\mathfrak{t}'} z^{\mathfrak{r}}.d^{\mathfrak{t}'\leftrightarrow\mathfrak{t},\mathfrak{t}\leftrightarrow\mathfrak{l}'})\Big)\,{}_{\mathfrak{l}}\langle z_0\rangle_{\mathfrak{r}_0}\\
&\equiv \Big(\text{let } x \,{}_{\mathfrak{l}'}\!\otimes_{\mathfrak{r}'} y = (a \,{}_{s_{LL}}\!\otimes_{s_{LR}} b) \text{ in } (\partial^{\mathfrak{t}'} z^{\mathfrak{r}}.d^{\mathfrak{t}'\leftrightarrow\mathfrak{t},\mathfrak{t}\leftrightarrow\mathfrak{l}'})\Big)\,{}_{s_L}\langle c\rangle_{s_R}\\
&\equiv (\partial^{\mathfrak{t}'} z^{\mathfrak{r}}.d^{\mathfrak{t}'\leftrightarrow\mathfrak{t},\mathfrak{t}\leftrightarrow\mathfrak{l}'}[s_{LL}/\mathfrak{l}' \otimes s_{LR}/\mathfrak{r}' \mid a/x, b/y])_{s_L}\langle c\rangle_{s_R}\\
&\equiv d^{\mathfrak{t}'\leftrightarrow\mathfrak{t},\mathfrak{t}\leftrightarrow\mathfrak{l}'}[s_{LL}/\mathfrak{l}' \otimes s_{LR}/\mathfrak{r}' \mid a/x, b/y][c/z]\llbracket \mathfrak{t} \prec s_L \boxtimes s_R/\mathfrak{t}'\rrbracket\\
&\equiv d[s_{LL}/\mathfrak{l}' \otimes s_{LR}/\mathfrak{r}' \mid a/x, b/y][c/z]\llbracket \mathfrak{t} \prec s_L \boxtimes s_R/\mathfrak{t}\rrbracket\end{aligned}$$

□

The definition of T here uses its argument marked, which is what restricts the branch $d : D$ to only use the ambient context marked.

Next, we allow internal dependency of $(A \otimes B) \otimes C$, and the dependency of D on the tensor.

Proposition 1.6.2 (Left-Associated Triple Elim). *The rule*

$$\otimes\text{-ELIM-TRIPLE-LEFT}\ \dfrac{\begin{array}{c}\mathfrak{t} \mid \underline{\Gamma}, w^{\mathfrak{t}} : \textcircled{\Sigma}_{(\underline{p}:\textcircled{\Sigma}_{(\underline{x}:A)} B)}(\mathsf{let}\ x \otimes y = \underline{p}\ \mathsf{in}\ C) \vdash D\ \mathsf{type} \\ \mathfrak{t} \prec (\mathfrak{l} \prec \mathfrak{l}' \otimes \mathfrak{r}') \otimes \mathfrak{r} \mid \underline{\Gamma}, x^{\mathfrak{l}'} : A, y^{\mathfrak{r}'} : B, z^{\mathfrak{r}} : C \vdash d : D[(x\ {}_{\mathfrak{l}'}\!\otimes_{\mathfrak{r}'} y)\ {}_{\mathfrak{l}}\!\otimes_{\mathfrak{r}} z/w] \\ \mathfrak{t} \prec \Phi \mid \Gamma \vdash s : \textcircled{\Sigma}_{(\underline{p}:\textcircled{\Sigma}_{(\underline{x}:A)} B)}(\mathsf{let}\ x \otimes y = \underline{p}\ \mathsf{in}\ C)\end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \mathsf{let}\ (x\ {}_{\mathfrak{l}'}\!\otimes_{\mathfrak{r}'} y)\ {}_{\mathfrak{l}}\!\otimes_{\mathfrak{r}} z = s\ \mathsf{in}\ d : D[s/w]}$$

$$\begin{aligned}&(\mathsf{let}\ (x\ {}_{\mathfrak{l}'}\!\otimes_{\mathfrak{r}'} y)\ {}_{\mathfrak{l}}\!\otimes_{\mathfrak{r}} z = (a\ {}_{s_{LL}}\!\otimes_{s_{LR}} b)\ {}_{s_L}\!\otimes_{s_R} c\ \mathsf{in}\ d) \\ &\equiv d[s_{LL}/\mathfrak{l}' \otimes s_{LR}/\mathfrak{r}' \mid a/x, b/y][c/z]\llbracket \mathfrak{t} \prec s_L \boxtimes s_R/\mathfrak{t}\rrbracket\end{aligned}$$

is derivable.

Proof. The construction is identical but the types are a little more complicated, so we work through it again.

We are trying to define a function

$$T : \left(\prod_{(x:A)} \textcircled{\Pi}_{(y:B)} \textcircled{\Pi}_{(z:C)} D^{\mathfrak{t}' \leftrightarrow \mathfrak{t}, \mathfrak{t} \leftrightarrow \mathfrak{l}'}[(x \otimes y) \otimes c/w]\right) \to \left(\prod_{(w:\textcircled{\Sigma}_{(\underline{p}:\textcircled{\Sigma}_{(\underline{x}:A)} B)}(\mathsf{let}\ x \otimes y = \underline{p}\ \mathsf{in}\ C))} D\right)$$

Given an $s : \textcircled{\Sigma}_{(\underline{p}:\textcircled{\Sigma}_{(\underline{x}:A)} B)}(\mathsf{let}\ x \otimes y = \underline{p}\ \mathsf{in}\ C)$ we can first apply $\otimes$-induction giving $p^{\mathfrak{l}} : \textcircled{\Sigma}_{(\underline{x}:A)} B$ and $z_0^{\mathfrak{r}_0} : (\mathsf{let}\ x \otimes y = \underline{p}\ \mathsf{in}\ C)$, so the goal is now a term of type $D[p\ {}_{\mathfrak{l}}\!\otimes_{\mathfrak{r}_0} z_0/w]$. We cannot do $\otimes$-induction on p, but we can use p to build a hom h and then apply it to z_0. For the hom, take

$$\begin{aligned}&h : \textcircled{\Pi}_{(z^{\mathfrak{r}}:(\mathsf{let}\ x \otimes y = \underline{p}\ \mathsf{in}\ C))} D^{t' \leftrightarrow t}[p\ {}_{\mathfrak{l}}\!\otimes_{\mathfrak{r}} z/w] \\ &h :\equiv \mathsf{let}\ x\ {}_{\mathfrak{l}'}\!\otimes_{\mathfrak{r}'} y = p\ \mathsf{in}\ (\partial^{t'} z^{\mathfrak{r}}.d^{t' \leftrightarrow t})\end{aligned}$$

Now applying this hom to z_0 gives a term $h_{\mathfrak{l}}\langle z_0 \rangle_{\mathfrak{r}_0}$ of type

$$\begin{aligned}&(D^{t' \leftrightarrow t}[p\ {}_{\mathfrak{l}}\!\otimes_{\mathfrak{r}} z/w])[z_0/z]\llbracket \mathfrak{t} \prec \mathfrak{l} \boxtimes \mathfrak{r}_0/\mathfrak{t}'\rrbracket \\ &\equiv (D^{t \leftrightarrow t'}[p\ {}_{\mathfrak{l}}\!\otimes_{\mathfrak{r}_0} z_0/w])^{t' \leftrightarrow t} \\ &\equiv D[p\ {}_{\mathfrak{l}}\!\otimes_{\mathfrak{r}_0} z_0/w]\end{aligned}$$

which was our goal.

Now if this is applied to $(a\ {}_{s_{LL}}\!\otimes_{s_{LR}} b)\ {}_{s_L}\!\otimes_{s_R} c$ then we can compute

$$\begin{aligned}&\mathsf{let}\ p\ {}_{\mathfrak{l}}\!\otimes_{\mathfrak{r}_0} z_0 = (a\ {}_{s_{LL}}\!\otimes_{s_{LR}} b)\ {}_{s_L}\!\otimes_{s_R} c\ \mathsf{in}\ h_{\mathfrak{l}}\langle z_0 \rangle_{\mathfrak{r}_0} \\ &\equiv h[a\ {}_{s_{LL}}\!\otimes_{s_{LR}} b/p]_{s_L}\langle c \rangle_{s_R} \\ &\equiv \left(\mathsf{let}\ x\ {}_{\mathfrak{l}'}\!\otimes_{\mathfrak{r}'} y = a\ {}_{s_{LL}}\!\otimes_{s_{LR}} b\ \mathsf{in}\ (\partial^{t'} z^{\mathfrak{r}}.d^{t \leftrightarrow t'})\right)_{s_L}\langle c \rangle_{s_R} \\ &\equiv \left((\partial^{t'} z^{\mathfrak{r}}.d^{t \leftrightarrow t'})[s_{LL}/\mathfrak{l}' \otimes s_{LR}/\mathfrak{r}' \mid a/x, b/y]\right)_{s_L}\langle c \rangle_{s_R} \\ &\equiv (\partial^{t'} z^{\mathfrak{r}}.d^{t \leftrightarrow t'}[s_{LL}/\mathfrak{l}' \otimes s_{LR}/\mathfrak{r}' \mid a/x, b/y])_{s_L}\langle c \rangle_{s_R} \\ &\equiv d^{t \leftrightarrow t'}[s_{LL}/\mathfrak{l}' \otimes s_{LR}/\mathfrak{r}' \mid a/x, b/y][c/z]\llbracket \mathfrak{t} \prec s_L \boxtimes s_R/\mathfrak{t}'\rrbracket \\ &\equiv d[s_{LL}/\mathfrak{l}' \otimes s_{LR}/\mathfrak{r}' \mid a/x, b/y][c/z]\llbracket \mathfrak{t} \prec s_L \boxtimes s_R/\mathfrak{t}\rrbracket\end{aligned}$$

$\square$

To derive the eliminator for the right-associated triple tensor, our strategy is to use Proposition 1.3.13 to replace the dependent tensor with a non-dependent one, apply symmetry to produce a left-associated triple tensor, and then use the above left-associated eliminator which we already know is derivable. We inline the equivalence of Proposition 1.3.13 into our argument because, as before, it is important that the construction computes correctly.

Proposition 1.6.3 (Right-Associated Triple Elim)**.** *The rule*

$$\textsc{$\otimes$-elim-triple-right}\ \frac{\begin{array}{c}\mathfrak{t} \mid \underline{\Gamma}, w^{\mathfrak{t}} : \textcircled{Σ}_{(\underline{x}:\underline{A})}\textcircled{Σ}_{(\underline{y}:\underline{B})}\underline{C} \vdash D \ \mathsf{type} \\ \mathfrak{t} \prec \mathfrak{l} \otimes (\mathfrak{r} \prec \mathfrak{l}' \otimes \mathfrak{r}') \mid \underline{\Gamma}, x^{\mathfrak{l}} : A, y^{\mathfrak{l}'} : B, z^{\mathfrak{r}'} : C \vdash d : D[x \ {}_{\mathfrak{l}}{\otimes}_{\mathfrak{r}} \ (y \ {}_{\mathfrak{l}'}{\otimes}_{\mathfrak{r}'} \ z)/w] \\ \mathfrak{t} \prec \Phi \mid \Gamma \vdash s : \textcircled{Σ}_{(\underline{x}:\underline{A})}\textcircled{Σ}_{(\underline{y}:\underline{B})}\underline{C}\end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \mathsf{let}\ x \ {}_{\mathfrak{l}}{\otimes}_{\mathfrak{r}} \ (y \ {}_{\mathfrak{l}'}{\otimes}_{\mathfrak{r}'} \ z) = s \ \mathsf{in}\ d : D[s/w]}$$

$$\begin{aligned}&(\mathsf{let}\ x \ {}_{\mathfrak{l}}{\otimes}_{\mathfrak{r}} \ (y \ {}_{\mathfrak{l}'}{\otimes}_{\mathfrak{r}'} \ z) = a \ {}_{s_L}{\otimes}_{s_R} \ (b \ {}_{s_{RL}}{\otimes}_{s_{RR}} \ c) \ \mathsf{in}\ d) \\ &\equiv d[a/x][s_{RL}/\mathfrak{l}' \otimes s_{RR}/\mathfrak{r}' \mid b/y, c/z]\llbracket \mathfrak{t} \prec s_R \boxtimes s_L/\mathfrak{t} \rrbracket\end{aligned}$$

is derivable.

Proof. Begin by applying $\otimes$-induction to $s : \textcircled{$\Sigma$}_{(\underline{x}:\underline{A})} \textcircled{Σ}_{(\underline{y}:\underline{B})} \underline{C}$ giving variables of type $x_0^{\mathfrak{l}_0} : \underline{A}$ and $q^{\mathfrak{r}} : \textcircled{Σ}_{(\underline{y}:\underline{B}[\underline{x_0}/\underline{x}])} \underline{C}[\underline{x_0}/\underline{x}]$, so the goal is now $D[x_0 \ {}_{\mathfrak{l}_0}{\otimes}_{\mathfrak{r}} \ q/s]$.

We are about to apply the left-associated to $q \otimes x_0$ as in $q \otimes x_0 \equiv (y_3 \ {}_{\mathfrak{l}'}{\otimes}_{\mathfrak{r}'} \ z_3) \ {}_{\mathfrak{r}}{\otimes}_{\mathfrak{l}} \ x_3$. The problem is that the types of y_3 and z_3 will continue to depend on $\underline{x_0} : \underline{A}$, rather than the new variable x_3. Our strategy is to smuggle a path $\underline{e} : \underline{x_0} = \underline{x}$ into the motive in the following way: we perform left-associated induction on $q \otimes x_0$ with motive

$$\prod_{(e:\underline{x_0}=\underline{\mathsf{pr}_2}(v))} \mathsf{let}\ (y_2 \otimes z_2) \otimes x_2 = v \ \mathsf{in}\ D[x_2 \otimes (\underline{e}_*(y_2) \otimes \underline{e}_*(z_2))/w]$$

This type is dull with respect to the ambient context, as v is the only free variable used unmarked, and so this use of the left-associated triple eliminator as given in Proposition 1.6.2 is valid.

And so we may assume $q \otimes x_0 \equiv (y_3 \ {}_{\mathfrak{l}'}{\otimes}_{\mathfrak{r}'} \ z_3) \ {}_{\mathfrak{r}}{\otimes}_{\mathfrak{l}} \ x_3$, and our goal is

$$\begin{aligned}&\Big(\prod_{(e:\underline{x_0}=\underline{\mathsf{pr}_2}(v))} \mathsf{let}\ (y_2 \otimes z_2) \otimes x_2 = v \ \mathsf{in}\ D[x_2 \otimes (\underline{e}_*(y_2) \otimes \underline{e}_*(z_2))/w]\Big) [(y_3 \ {}_{\mathfrak{l}'}{\otimes}_{\mathfrak{r}'} \ z_3) \ {}_{\mathfrak{r}}{\otimes}_{\mathfrak{l}} \ x_3/v] \\ &\equiv \prod_{(e:\underline{x_0}=\underline{\mathsf{pr}_2}((y_3 \otimes z_3) \otimes x_3))} \mathsf{let}\ (y_2 \otimes z_2) \otimes x_2 = (y_3 \ {}_{\mathfrak{l}'}{\otimes}_{\mathfrak{r}'} \ z_3) \ {}_{\mathfrak{r}}{\otimes}_{\mathfrak{l}} \ x_3 \ \mathsf{in}\ D[x_2 \otimes (\underline{e}_*(y_2) \otimes \underline{e}_*(z_2))/w] \\ &\equiv \prod_{(e:\underline{x_0}=\underline{x_3})} D[x_3 \ {}_{\mathfrak{l}}{\otimes}_{\mathfrak{r}} \ (\underline{e}_*(y_3) \ {}_{\mathfrak{l}'}{\otimes}_{\mathfrak{r}'} \ \underline{e}_*(z_3))/w]\end{aligned}$$

Inhabiting this type is simple enough, after weakening d with x_3, y_3 and z_3 we can form

$$\lambda e. d[x_3/x][\mathfrak{l}'/\mathfrak{l}' \otimes \mathfrak{r}'/\mathfrak{r}' \mid \underline{e}_*(y_3)/y, \underline{e}_*(z_3)/z]\llbracket \mathfrak{t} \prec \mathfrak{l} \boxtimes \mathfrak{r}/\mathfrak{t} \rrbracket$$

The result of this left-associated induction has type

$$\prod_{(e:\underline{x_0}=\underline{\mathsf{pr}_2}(q \otimes x_0))} \mathsf{let}\ (y_2 \otimes z_2) \otimes x_2 = q \otimes x_0 \ \mathsf{in}\ D[x_2 \otimes (\underline{e}_*(y_2) \otimes \underline{e}_*(z_2))/w]$$

The endpoint of the path $\underline{\mathsf{pr}_2}(q \otimes x_0)$ computes to $\underline{x_0}$, so we can apply this function to $\mathsf{refl}_{\underline{x_0}}$ yielding

$$\text{let } (y_2 \otimes z_2) \otimes x_2 = q \otimes x_0 \text{ in } D[x_2 \otimes (y_2 \otimes z_2)/w]$$

As in Proposition 1.3.6 for the binary eliminator, the left-associated eliminator satisfies a uniqueness principle, in general giving a path

$$a[t/w] \equiv \text{let } ((x \otimes y) \otimes z) = t \text{ in } a[(x \otimes y) \otimes z/w]$$

In the present case, this gives a family of paths

$$U(t) : (\text{let } (y_2 \otimes z_2) \otimes x_2 = t \text{ in } D[x_2 \otimes (y_2 \otimes z_2)/w]) = D[t/w]$$

Transporting along the path $U(q \otimes x_0)$ gives a term of type $D[q \otimes x_0/w]$, which was our original goal.

All together, we have constructed the term

$$\begin{aligned}&\text{let } x_0 \;{}_{\mathfrak{l}_0}\!\otimes_{\mathfrak{r}} q = s \text{ in } U(q \;{}_{\mathfrak{r}}\!\otimes_{\mathfrak{l}_0} x_0)_*\big(\text{let } (y_3 \otimes z_3) \otimes x_3 = q \;{}_{\mathfrak{r}}\!\otimes_{\mathfrak{l}_0} x_0 \text{ in}\\ &\qquad\qquad\qquad \lambda e.d[x_3/x][\mathfrak{l}'/\mathfrak{l}' \otimes \mathfrak{r}'/\mathfrak{r}' \mid \underline{e}_*(y_3)/y, \underline{e}_*(z_3)/z]\llbracket(\mathfrak{t} \prec \mathfrak{l} \boxtimes \mathfrak{r}/\mathfrak{t})\rrbracket\big)(\mathsf{refl}_{\underline{x_0}})\end{aligned}$$

Finally, when provided an actual triple, everything computes away:

$$\begin{aligned}&\text{let } x_0 \;{}_{\mathfrak{l}_0}\!\otimes_{\mathfrak{r}} q = a \;{}_{s_L}\!\otimes_{s_R} (b \;{}_{s_{RL}}\!\otimes_{s_{RR}} c) \text{ in } U(q \;{}_{\mathfrak{r}}\!\otimes_{\mathfrak{l}_0} x_0)_* \big(\text{let } (y_3 \otimes z_3) \otimes x_3 = q \;{}_{\mathfrak{r}}\!\otimes_{\mathfrak{l}_0} x_0 \text{ in } \lambda e.(\dots)\big)\,(\mathsf{refl}_{\underline{x_0}})\\
&\equiv U((b \;{}_{s_{RL}}\!\otimes_{s_{RR}} c) \;{}_{s_R}\!\otimes_{s_L} a)_* \big(\text{let } (y_3 \otimes z_3) \otimes x_3 = (b \;{}_{s_{RL}}\!\otimes_{s_{RR}} c) \;{}_{s_R}\!\otimes_{s_L} a \text{ in } \lambda e.(\dots)\big)\,(\mathsf{refl}_{\underline{a}})\\
&\equiv \mathsf{refl}_{-*} \big(\text{let } (y_3 \otimes z_3) \otimes x_3 = (b \;{}_{s_{RL}}\!\otimes_{s_{RR}} c) \;{}_{s_R}\!\otimes_{s_L} a \text{ in } \lambda e.(\dots)\big)\,(\mathsf{refl}_{\underline{a}})\\
&\equiv \big(\text{let } (y_3 \otimes z_3) \otimes x_3 = (b \;{}_{s_{RL}}\!\otimes_{s_{RR}} c) \;{}_{s_R}\!\otimes_{s_L} a \text{ in } \lambda e.(\dots)\big)\,(\mathsf{refl}_{\underline{a}})\\
&\equiv \big(\lambda e.d[a/x][s_{RL}/\mathfrak{l}' \otimes s_{RR}/\mathfrak{r}' \mid \underline{e}_*(b)/y, \underline{e}_*(c)/z]\llbracket(\mathfrak{t} \prec s_R \boxtimes s_L/\mathfrak{t})\rrbracket\big)\,(\mathsf{refl}_{\underline{a}})\\
&\equiv d[a/x][s_{RL}/\mathfrak{l}' \otimes s_{RR}/\mathfrak{r}' \mid (\mathsf{refl}_{\underline{a}})_*(b)/y, (\mathsf{refl}_{\underline{a}})_*(c)/z]\llbracket(\mathfrak{t} \prec s_R \boxtimes s_L/\mathfrak{t})\rrbracket\\
&\equiv d[a/x][s_{RL}/\mathfrak{l}' \otimes s_{RR}/\mathfrak{r}' \mid b/y, c/z]\llbracket(\mathfrak{t} \prec s_R \boxtimes s_L/\mathfrak{t})\rrbracket\end{aligned}$$

□

Proposition 1.6.4 (Colourful Path Induction). *The rule*

$$\begin{array}{c}\mathfrak{t} \mid \underline{\Gamma}, w^{\mathfrak{t}} : \left(\sum_{(x:A)}\sum_{(x':A)} x = x'\right) \otimes \left(\sum_{(y:B)}\sum_{(y':B)} y = y'\right) \vdash D \text{ type}\\ \mathfrak{t} \prec \mathfrak{l} \otimes \mathfrak{r} \mid \underline{\Gamma}, x^{\mathfrak{l}} : A, y^{\mathfrak{r}} : B \vdash d : D[(x, x, \mathsf{refl}_x) \;{}_{\mathfrak{l}}\!\otimes_{\mathfrak{r}} (y, y, \mathsf{refl}_y)/w]\\ \mathfrak{t} \prec \Phi \mid \Gamma \vdash s : \left(\sum_{(x:A)}\sum_{(x':A)} x = x'\right) \otimes \left(\sum_{(y:B)}\sum_{(y':B)} y = y'\right)\\ \hdashline \mathfrak{t} \prec \Phi \mid \Gamma \vdash \text{let } (x, x, \mathsf{refl}_x) \;{}_{\mathfrak{l}}\!\otimes_{\mathfrak{r}} (y, y, \mathsf{refl}_y) = s \text{ in } d : D[s/w]\end{array}$$

$$\begin{aligned}&(\text{let } (x, x, \mathsf{refl}_x) \;{}_{\mathfrak{l}}\!\otimes_{\mathfrak{r}} (y, y, \mathsf{refl}_y) = (a, a, \mathsf{refl}_a) \;{}_{s_L}\!\otimes_{s_R} (b, b, \mathsf{refl}_b) \text{ in } d)\\ &\equiv d[s_L/\mathfrak{l} \otimes s_R/\mathfrak{r} \mid a/x, b/y]\end{aligned}$$

is derivable.

Proof. We can use two hom types to expose each path in turn. Suppose (via ordinary $\otimes$-induction) we have $(x, x', p) : \sum_{(x:A)} \sum_{(x':A)} x = x'$ and $(y, y', q) : \sum_{(y:B)} \sum_{(y':B)} y = y'$. We first use (x, x', p) to produce a (red) hom with the following type:

$$\textcircled{\Pi}_{((y_2, y_2', q_2) : \sum_{(y:B)} \sum_{(y':B)} y = y')} D[(x, x', p) \otimes (y_2, y_2', q_2)/w]$$

We can first apply ordinary Id-induction to assume $p \equiv \mathsf{refl}_{x''}$, so that the goal is a hom

$$\textcircled{\Pi}_{((y_2, y_2', q_2) : \sum_{(y:B)} \sum_{(y':B)} y = y')} D[(x'', x'', \mathsf{refl}_{x''}) \otimes (y_2, y_2', q_2)/w]$$

So suppose we now have $(y_2, y_2', q_2) : \sum_{(y:B)} \sum_{(y':B)} y = y'$, and we have to produce a term of type $D[(x'', x'', \mathsf{refl}_{x''}) \otimes (y_2, y_2', q_2)/w]$. We can pull the same trick in the other direction: we use (y_2, y_2', q_2) to produce a hom

$$\textcircled{\Pi}_{(x_2'':A)} D[(x_2'', x_2'', \mathsf{refl}_{x_2''}) \otimes (y_2, y_2', q_2)/w]$$

which we then apply to x''. For this second hom, apply ordinary Id-induction to assume $q_2 \equiv \mathsf{refl}_{y_2''}$ (which is allowed because q_2 is now of the top colour), so the goal becomes a hom of type

$$\textcircled{\Pi}_{(x_2'':A)} D[(x_2'', x_2'', \mathsf{refl}_{x_2''}) \otimes (y_2'', y_2'', \mathsf{refl}_{y_2''})/w]$$

and for this we finally have

$$(\partial x_2''.d[x_2''/x, y_2''/y]) : \textcircled{\Pi}_{(x_2'':A)} D[(x_2'', x_2'', \mathsf{refl}_{x_2''}) \otimes (y_2'', y_2'', \mathsf{refl}_{y_2''})/w]$$

As a single term, we have constructed

$$\begin{aligned}&\mathsf{let}\ (x, x', p) \otimes (y, y', q) = s\ \mathsf{in}\\ &(\mathsf{let}\ \mathsf{refl}_{x''} = p\ \mathsf{in}\ \partial(y_2, y_2', q_2).(\mathsf{let}\ \mathsf{refl}_{y_2''} = q_2\ \mathsf{in}\ \partial x_2''.d[x_2''/x, y_2''/y])\langle x''\rangle)\langle (y, y', q)\rangle\end{aligned}$$

When $s \equiv (a, a, \mathsf{refl}_a) \otimes (b, b, \mathsf{refl}_b)$, the induction on s followed by the induction on p both compute, and we are left with

$$\begin{aligned}&(\partial(y_2, y_2', q_2).(\mathsf{let}\ \mathsf{refl}_{y_2''} = q_2\ \mathsf{in}\ \partial x_2''.d[x_2''/x, y_2''/y])\langle a\rangle)\langle (b, b, \mathsf{refl}_b)\rangle\\ &\equiv (\mathsf{let}\ \mathsf{refl}_{y_2''} = \mathsf{refl}_b\ \mathsf{in}\ \partial x_2''.d[x_2''/x, y_2''/y])\langle a\rangle)\\ &\equiv (\partial x_2''.d[x_2''/x, b/y])\langle a\rangle)\\ &\equiv d[a/x, b/y]\end{aligned}$$

as required. □

Remark 1.6.5. The preceding few propositions are the carcass of an attempted proof that all the induction principles we want are derivable. Unfortunately, the strategy of using $\textcircled{\Pi}$ to 'move things out of the way' eventually fails. Associativity of $\otimes$ means that the eliminators for $\otimes$-types are always derivable, but once $\textcircled{\Sigma}$ and Σ are combined, we run into an infinite regress.

The above arguments only go through when the ambient context is dull in the motive of the eliminators. So consider the type $(A \otimes B) \times (C \otimes (D \otimes E))$. We may use Π to move $C \otimes (D \otimes E)$ into the motive and apply binary $\otimes$-induction on $A \otimes B$. But then the derivable eliminator for $C \otimes (D \otimes E)$ cannot be applied without marking the linear information in A and B.

1.6.2 Telescopes

We now turn to the general pattern-matching rule. The patterns we add here are the 'constructor patterns' of [LZH08] and are reminiscent of a 'positive focusing phase' in which many positive types are decomposed at once, though unlike focused logics we do not require that a type is maximally decomposed by a pattern match.

The common feature of the types that can be matched is that they each have judgemental avatars as context structure, and pattern matching will decompose a composite of these types into their judgemental versions simultaneously. We do not include patterns for $+$-types or the 0-type, or any other (higher) inductive type: this simplifies the rules somewhat as every pattern match will have exactly one branch.

The first new judgement is for a 'telescope', which is the collection of new variables bound by a pattern:

$$\mathfrak{t} \prec \Phi \mid \Gamma \vdash \Psi \mid \Delta \ \mathsf{tele}$$

where $\mathfrak{t} \prec \Phi \mid \Gamma$ ctx and Ψ $\mathsf{palette}$. Extending this context with the telescope yields

$$\mathfrak{t} \prec (\Phi, \Psi) \mid \Gamma, \Delta \ \mathsf{ctx}$$

where the Ψ palette is placed *under* the top colour label, rather than adjacent to it. The rules for forming telescopes, Figure 1.20, are the rules forced by the requirement that this latter context be well-formed.

Remark 1.6.6. In ordinary MLTT, telescopes in a context coincide with all possible extensions of that context that one can encounter in a derivation that ends with that context. This is because the only modifications to the context that appear are extending it with additional variables on the right.

The notion of telescope we use here is more restrictive: we can only extend the palette $\mathfrak{t} \prec \Phi$ with a single additional subpalette Ψ, combined with a comma. This does not include the 'linear' context extension used in $\multimap$-types, nor does it include any filtering of the original palette; both of these 'context extensions' could be encountered in a derivation ending in the palette $\mathfrak{t} \prec \Phi$.

Palette Substitutions. When a pattern is matched, we will need to do a simultaneous substitution for all the variables bound in the pattern. A substitution for a telescope will consist of two pieces. First, we have to decide how the resources of the context are used to produce the resources of the telescope, in the form of a *palette substitution* $\Phi \vdash \kappa : \Psi$. This is an assignment of once slice of the domain for each colour of the codomain, in a manner complying with the linearity restrictions of both palettes.

Palette substitutions are formed by induction on the codomain, the rules are given in Figure 1.21. Palette substitutions are formed using an auxiliary judgement $\Phi^s \vdash \kappa : \Psi$ where $\Phi \vdash s$ slice, the purpose of which is to make the substitution operation easy to compute by hand. Going up a derivation, less of the palette remains to be used, and the slice s keeps track of the part that is accessible.

- There is a unique substitution into 1, as 1 represents the terminal object.

$$\text{TELE-EMPTY}\ \frac{\mathfrak{t} \prec \Phi \mid \Gamma \ \mathsf{ctx} \qquad \Psi \ \mathsf{palette}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \Psi \mid \cdot \ \mathsf{tele}} \qquad \text{TELE-EXT}\ \frac{\begin{array}{c}\mathfrak{t} \prec \Phi \mid \Gamma \vdash \Psi \mid \Omega \ \mathsf{tele} \\ \mathfrak{t} \prec \Psi \vdash \mathfrak{c} \ \mathsf{colour} \\ (\mathfrak{t} \prec \Phi, \Psi)^{\mathfrak{c}} \mid \Gamma^{\mathfrak{c}}, \Omega^{\mathfrak{c}} \vdash A \ \mathsf{type}\end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \Psi \mid \Omega, x^{\mathfrak{c}} : A \ \mathsf{tele}}$$

$$\text{TELE-EXT-MARKED}\ \frac{\begin{array}{c}\mathfrak{t} \prec \Phi \mid \Gamma \vdash \Psi \mid \Omega \ \mathsf{tele} \\ \mathfrak{c} \mid \underline{\Gamma}, \underline{\Omega} \vdash A \ \mathsf{type}\end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \Psi \mid \Omega, \underline{x}^{\mathfrak{c}} : A \ \mathsf{tele}}$$

$$\text{CTX-EXT-TELE}\ \frac{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \Psi \mid \Omega \ \mathsf{tele}}{\mathfrak{t} \prec \Phi, \Psi \mid \Gamma, \Omega \ \mathsf{ctx}}$$

Figure 1.20: Rules for Telescopes

$$\text{PAL-SUB-EMPTY}\ \frac{}{\Phi \vdash \cdot : 1} \qquad \text{PAL-SUB-}\times\ \frac{\Phi \vdash \kappa_1 : \Psi_1 \qquad \Phi \vdash \kappa_2 : \Psi_2}{\Phi \vdash \kappa_1, \kappa_2 : \Psi_1, \Psi_2}$$

$$\text{PAL-SUB-}\otimes\ \frac{\begin{array}{c}\Phi \vdash s_L \boxtimes s_R \ \mathsf{presplit} \\ \Phi^{s_L} \vdash \kappa_L : \Psi_L \qquad \Phi^{s_R} \vdash \kappa_R : \Psi_R\end{array}}{\Phi \vdash \kappa_L \otimes \kappa_R : \Psi_L \otimes \Psi_R} \qquad \text{PAL-SUB-UNIT}\ \frac{\Phi \vdash U \ \mathsf{unit}}{\Phi \vdash (U/j) : \varnothing_j}$$

$$\text{PAL-SUB-NAME}\ \frac{\begin{array}{c}\Phi \vdash s \ \mathsf{slice} \\ \Phi \vdash \mathsf{u}(s) \boxtimes \varnothing \ \mathsf{presplit} \qquad \Phi^{\mathsf{u}(s)} \vdash \kappa : \Psi\end{array}}{\Phi \vdash (s/\mathfrak{c} \prec \kappa) : (\mathfrak{c} \prec \Psi)}$$

Figure 1.21: Rules for Palette Substitutions.

$$\textsc{tele-sub-empty}\ \frac{\mathfrak{t} \prec \Phi \vdash \kappa : \Psi}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash (\kappa \mid \cdot) : \Psi \mid \cdot}$$

$$\textsc{tele-sub-ext}\ \frac{\begin{array}{c}\mathfrak{t} \prec \Phi \mid \Gamma \vdash (\kappa \mid \theta) : \Psi \mid \Omega \\ (\mathfrak{t} \prec \Phi)^{\mathfrak{c}[\kappa]} \mid \Gamma^{\mathfrak{c}[\kappa]} \vdash a : A[\kappa \mid \theta]\end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash (\kappa \mid \theta, a/x^{\mathfrak{c}}) : \Psi \mid \Omega, x^{\mathfrak{c}} : A}$$

$$\textsc{tele-sub-ext-marked}\ \frac{\begin{array}{c}\mathfrak{t} \prec \Phi \mid \Gamma \vdash (\kappa \mid \theta) : \Psi \mid \Omega \\ \mathfrak{c} \mid \underline{\Gamma} \vdash a : A[\kappa \mid \theta]\end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash (\kappa \mid \theta, a/\underline{x}) : \Psi \mid \Omega, \underline{x}^{\mathfrak{c}} : A}$$

$$\textsc{subst}\ \frac{\mathfrak{t} \prec \Phi \mid \Gamma \vdash (\kappa \mid \theta) : \Psi \mid \Omega \qquad \mathfrak{t} \prec \Phi, \Psi \mid \Gamma, \Omega, \Gamma' \vdash \mathcal{J}}{\mathfrak{t} \prec \Phi \mid \Gamma, \Gamma'[\kappa \mid \theta] \vdash \mathcal{J}[\kappa \mid \theta]}$$

Figure 1.22: Rules for Telescope Substitutions

- A substitution into a $\times$-bunch is given by substitutions into the components, keeping the domain palette the same. Allowing the palette Φ unchanged on both sides is what builds in contraction for the palette ,.

- A substitution into a $\otimes$-bunch is given by splitting the domain and providing substitutions from each piece into the two components.

- A substitution into a palette headed by a label is given by an individual slice representing the whole palette, together with a substitution into the palette under the label.

  Here we have used a sneaky trick: the split $\Phi \vdash \mathsf{u}(s) \boxtimes \varnothing$ $\mathsf{presplit}$ ensures that the slice s chosen represents some subpalette that can be cartesian weakened to the original Φ.

Telescope Substitutions. Given a palette substitution $\mathfrak{t} \prec \Phi \vdash \kappa : \Psi$, a telescope substitution $\mathfrak{t} \prec \Phi \mid \Gamma \vdash (\kappa \mid \theta) : \Psi \mid \Delta$ is formed by providing for each variable $x^{\mathfrak{c}} : A$, a term $a : A[\kappa \mid \theta]$ that uses the resources in the preimage of $\mathfrak{c}$, i.e. in the slice $\mathfrak{c}[\kappa]$ slice. The rules for telescope substitutions are given in Figure 1.22.

1.6.3 Patterns

A pattern is represented by a judgement $\mathfrak{t} \prec \Phi \mid \Gamma \vdash \Psi \mid \Delta \vdash p : A$ $\mathsf{pattern}$, which presupposes that

$$\begin{aligned}
&\mathfrak{t} \prec \Phi &&\mid \Gamma &&\vdash \Psi \mid \Delta\ \mathsf{tele} \\
&\mathfrak{t} \prec \Phi &&\mid \Gamma &&\vdash A\ \mathsf{type} \\
&\mathfrak{t} \prec \Phi, \Psi &&\mid \Gamma, \Delta &&\vdash p : A
\end{aligned}$$

We think of the type A as exactly the telescope $\Psi \mid \Delta \ \mathsf{tele}$ packaged into a type, with p the term that constructs a term of the type given the telescope.

There is a second judgement that defines when a term matches a pattern: we write this as

$$\mathfrak{t} \prec \Phi \mid \Gamma \vdash a \bowtie p[\kappa \mid \theta] : A$$

presupposing that

$$\begin{gathered}\mathfrak{t} \prec \Phi \mid \Gamma \vdash a : A \\ \mathfrak{t} \prec \Phi \mid \Gamma \vdash \Psi \mid \Delta \vdash p : A \ \mathsf{pattern} \\ \mathfrak{t} \prec \Phi \mid \Gamma \vdash (\kappa \mid \theta) : \Psi \mid \Delta\end{gathered}$$

It follows quickly by induction that whenever we have $\Phi \mid \Gamma \vdash a \bowtie p[\kappa \mid \theta] : A$, also $a \equiv p[\kappa \mid \theta]$.

The rules for patterns and pattern matching are given in Figures 1.23 and 1.24. Considering the patterns one-at-a-time:

- First, we have the variable pattern, which matches any term.
- A pattern $(p_1, p_2) : \sum_{(x:A)} B$ matches $(a, b) : \sum_{(x:A)} B$ if a matches p_1 and b matches p_2. The telescopes bound by p_1 and p_2 are combined with a comma.
- The pattern $\star : 1$ matches $\star : 1$. (Recall that we include the η-rule for the cartesian unit, so $x \equiv \star$ for any $x : 1$.) Doing so binds no new variables.
- The pattern $(p, p, \mathsf{refl}_p) : \sum_{(x:A)} \sum_{(y:A)} x =_A y$ matches the term (a, a, refl_a) when p matches a. This binds the same telescope that p does.
- The pattern $\underline{p}^{\natural} : \natural A$ matches $a^{\natural}$. By the η-rule for $\natural$, this applies to any term $n : \natural A$, because $n \equiv \underline{n_{\natural}}^{\natural}$. This binds the telescope bound by p with every variable replaced with a zeroed variable.
- The pattern $(p_L \ {}_{\mathfrak{c}_L}\!\otimes_{\mathfrak{c}_R} \ p_R) : \textcircled{Σ}_{(\underline{x}:A)} B$ matches $(a \ {}_{s_L}\!\otimes_{s_R} \ b)$ if a matches p_L and b matches p_R. This binds the telescopes of p_L and p_R with their palettes combined with a $\otimes$, using the two colour names $\mathfrak{c}_L$ and $\mathfrak{c}_R$ the top colours of each subpalette.
- The pattern $\square_i : \mathbb{S}$ matches $\square_j$ for any unit label j. This binds a new judgemental unit $\varnothing_i$.
- The pattern $(p_L \ {}_{\mathfrak{t}}\!\otimes_{\mathfrak{u}.\prec\varnothing_i} \ \square_i) : A \otimes \mathbb{S}$ matches $(a \ {}_{\mathfrak{t}}\!\otimes_{\mathfrak{u}.\prec\varnothing_i} \ \square_i)$ if a matches p_L, where the split used *must* be the unitor split. This binds the same telescope as p_L.

1.6.4 Matching

To use a pattern, we use the let-binding rule as shown in Figure 1.25.

A uniqueness principle holds for any pattern, along the same lines of Proposition 1.3.6 for $\otimes$. For the third and hopefully final time:

Meta-Proposition 1.6.7 (Uniqueness principle for patterns)**.** *Fixing any pattern*

$$\Psi \mid \Delta \vdash p : A \ \mathsf{pattern},$$

suppose $C : A \to \mathcal{U}$ *is a type family and* $f : \prod_{(a:A)} C(a)$. *Then for any* $a : A$ *we have*

$$(\mathsf{let}\ p = a \ \mathsf{in}\ f(p)) = f(a)$$

$$\frac{\mathfrak{t} \prec \Phi \mid \Gamma \vdash A \ \mathsf{type}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash 1 \mid x^{\mathfrak{t}} : A \vdash x : A \ \mathsf{pattern}}$$

$$\frac{\begin{array}{c} \mathfrak{t} \prec \Phi \mid \Gamma \vdash \Psi_1 \mid \Omega_1 \vdash p_1 : A \ \mathsf{pattern} \\ \mathfrak{t} \prec \Phi \mid \Gamma, x^{\mathfrak{t}} : A \vdash B \ \mathsf{type} \\ \mathfrak{t} \prec \Phi, \Psi_1 \mid \Gamma, \Omega_1 \vdash \Psi_2 \mid \Omega_2 \vdash p_2 : B[p_1/x] \ \mathsf{pattern} \end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \Psi_1, \Psi_2 \mid \Omega_1, \Omega_2 \vdash (p_1, p_2) : \sum_{(x:A)} B \ \mathsf{pattern}} \qquad \frac{}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash 1 \mid \cdot \vdash \star : 1 \ \mathsf{pattern}}$$

$$\frac{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \Psi \mid \Omega \vdash p : A \ \mathsf{pattern}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \Psi \mid \Omega \vdash (p, p, \mathsf{refl}_p) : \sum_{(x:A)} \sum_{(y:A)} x =_A y \ \mathsf{pattern}}$$

$$\frac{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \Psi \mid \Omega \vdash p : A \ \mathsf{pattern}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash 1 \mid \underline{\Omega} \vdash \underline{p}^{\natural} : \natural \underline{A} \ \mathsf{pattern}}$$

$$\frac{\begin{array}{c} \mathfrak{c}_L \mid \underline{\Gamma} \vdash \Psi_L \mid \Omega_L \vdash p_L : A^{\mathfrak{c}_L \leftrightarrow \mathfrak{t}} \ \mathsf{pattern} \\ \mathfrak{c}_R \mid \underline{\Gamma, \Omega_L} \vdash \Psi_R \mid \Omega_R \vdash p_R : B[\underline{p_L}/\underline{x}]^{\mathfrak{c}_R \leftrightarrow \mathfrak{t}} \ \mathsf{pattern} \end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash (\mathfrak{c}_L \prec \Psi_L) \otimes (\mathfrak{c}_R \prec \Psi_R) \mid \Omega_L, \Omega_R \vdash (p_L \ {}_{\mathfrak{c}_L}\!\otimes_{\mathfrak{c}_R} p_R) : \mathop{\bigcirc\mkern-15mu\sum}_{(\underline{x}:A)} B \ \mathsf{pattern}}$$

$$\frac{}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \varnothing_i \mid \cdot \vdash \text{⌑}_i : \mathbb{S} \ \mathsf{pattern}}$$

$$\frac{\mathfrak{t} \mid \underline{\Gamma} \vdash \Psi_L \mid \Omega_L \vdash p_L : A \ \mathsf{pattern}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \Psi_L \mid \Omega_L \vdash (p_L \ {}_{\mathfrak{t}}\!\otimes_{\mathfrak{u}.\prec \varnothing_i} \text{⌑}_i) : \mathop{\bigcirc\mkern-15mu\sum}_{(\underline{x}:A)} \mathbb{S} \ \mathsf{pattern}}$$

$$\frac{\begin{array}{c} \mathfrak{t} \mid \underline{\Gamma}, \underline{x} : \mathbb{S} \vdash B \ \mathsf{type} \\ \mathfrak{t} \mid \underline{\Gamma} \vdash \Psi_R \mid \Omega_R \vdash p_R : B[\underline{\text{⌑}}/\underline{x}] \ \mathsf{pattern} \end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \Psi_R \mid \Omega_R \vdash (\text{⌑}_i \ {}_{\mathfrak{u}.\prec \varnothing_i}\!\otimes_{\mathfrak{t}} p_R) : \mathop{\bigcirc\mkern-15mu\sum}_{(\underline{x}:\mathbb{S})} B \ \mathsf{pattern}}$$

Figure 1.23: Rules for Patterns

$$\frac{}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash a \bowtie x[\cdot \mid a/x] : A}$$

$$\frac{\begin{array}{c}\mathfrak{t} \prec \Phi \mid \Gamma \vdash a \bowtie p_1[\kappa_1 \mid \delta_1] : A \\ \mathfrak{t} \prec \Phi \mid \Gamma \vdash b \bowtie p_2[\kappa_2 \mid \delta_2] : B[a/x]\end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash (a, b) \bowtie (p_1, p_2)[\kappa_1, \kappa_2 \mid \delta_1, \delta_2] : \sum_{(x:A)} B} \qquad \frac{}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \star \bowtie \star[\cdot \mid \cdot] : 1}$$

$$\frac{\mathfrak{t} \prec \Phi \mid \Gamma \vdash a \bowtie p[\kappa \mid \delta] : A}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash (a, a, \mathsf{refl}_a) \bowtie (p, p, \mathsf{refl}_p)[\kappa \mid \delta] : A}$$

$$\frac{\mathfrak{t} \mid \underline{\Gamma} \vdash n \bowtie p[\kappa \mid \delta] : A}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash n^{\natural} \bowtie \underline{p}^{\natural}[\kappa \mid \delta] : \natural A \ \mathsf{pattern}}$$

$$\frac{\begin{array}{c}\mathfrak{t} \prec \Phi \vdash s_L \boxtimes s_R \ \mathsf{split} \\ \Phi^{s_L} \mid \Gamma^{s_L} \vdash a \bowtie p_L[\kappa_L \mid \delta_L] : A \\ \Phi^{s_R} \mid \Gamma^{s_R} \vdash b \bowtie p_R[\kappa_R \mid \delta_R] : B[\underline{a}/\underline{x}]\end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash (a \ {}_{s_L}{\otimes}_{s_R} \ b) \bowtie (p_L \ {}_{\mathfrak{c}_L}{\otimes}_{\mathfrak{c}_R} \ p_R)[(s_L/\mathfrak{c}_L \prec \kappa_L) \otimes (s_R/\mathfrak{c}_R \prec \kappa_R) \mid \delta_L, \delta_R] : \textcircled{Σ}_{(\underline{x}:A)} B}$$

$$\frac{}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash ⌑_j \bowtie ⌑_i[j/i \mid \cdot]}$$

$$\frac{\mathfrak{t} \prec \Phi \mid \Gamma \vdash a \bowtie p_L[\kappa_L \mid \delta_L] : A}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash (a \ {}_{\mathfrak{t}}{\otimes}_{\mathfrak{u}.\prec \varnothing_i} \ ⌑_i) \bowtie (p_L \ {}_{\mathfrak{t}}{\otimes}_{\mathfrak{u}.\prec \varnothing_i} \ ⌑_i)[\kappa_L \mid \delta_L] : A \otimes \mathbb{S}}$$

$$\frac{\mathfrak{t} \prec \Phi \mid \Gamma \vdash b \bowtie (p_R[\underline{⌑}/x])[\kappa_R \mid \delta_R] : B[\underline{⌑}/\underline{x}]}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash (⌑_i \ {}_{\mathfrak{u}.\prec \varnothing_i}{\otimes}_{\mathfrak{t}} \ b) \bowtie (⌑_i \ {}_{\mathfrak{u}.\prec \varnothing_i}{\otimes}_{\mathfrak{t}} \ p_R)[\kappa_L \mid \delta_L] : \textcircled{Σ}_{(\underline{x}:\mathbb{S})} B}$$

Figure 1.24: Rules for Pattern Matching

$$\textsc{match}\ \dfrac{\begin{array}{c}\mathfrak{t} \prec \Phi \mid \Gamma \vdash \Psi \mid \Omega \vdash p : A \ \mathsf{pattern} \\ \mathfrak{t} \prec \Phi \mid \Gamma, z^{\mathfrak{t}} : A \vdash C \ \mathsf{type} \\ \mathfrak{t} \prec \Phi, \Psi \mid \Gamma, \Omega \vdash c : C[p/z] \\ \mathfrak{t} \prec \Phi \mid \Gamma \vdash a : A\end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \mathsf{let}\ p = a \ \mathsf{in}\ c : C[a/z]}$$

$$\textsc{match-beta}\ \dfrac{\begin{array}{c}\mathfrak{t} \prec \Phi \mid \Gamma, z^{\mathfrak{t}} : A \vdash C \ \mathsf{type} \\ \mathfrak{t} \prec \Phi, \Psi \mid \Gamma, \Omega \vdash c : C[p/z] \\ \mathfrak{t} \prec \Phi \mid \Gamma \vdash \Psi \mid \Omega \vdash a \bowtie p[\kappa \mid \theta] : A \ \mathsf{pattern}\end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash (\mathsf{let}\ p = a \ \mathsf{in}\ c) \equiv c[\kappa \mid \theta] : C[a/z]}$$

Figure 1.25: Rules for Match

1.6.5 Discussion

Remark 1.6.8. Because 1, Σ-types and $\natural$-types have definitional η-rules, the 1, Σ- and $\natural$-patterns on their own can never be 'stuck': η-expanding the target of the pattern match will expose a term that matches the pattern. (This somewhat justifies the Σ-type pattern-matching syntax used in ordinary HoTT.) Patterns matches containing these types can still be 'stuck' if they are used within a larger pattern that contains $\otimes$ or Id elsewhere, however.

Remark 1.6.9. There is the question of adding patterns for $+$-types, pushouts, and other higher inductive types. Combining pattern-matching for $+$ with pattern-matching for $\otimes$ would force $\otimes$ to preserve coproducts, a property that might not hold in every model (absent the hom right-adjoint).

Remark 1.6.10. Rather than adding pattern matching, one's first instinct might be to let the variable $z^{\mathfrak{t}} : A \otimes B$ used as the target in $\otimes$-ELIM have an arbitrary colour, not necessarily the top colour of the palette. In the $\alpha\lambda$-calculus [OHe03], the $\otimes$-ELIM rule works in this way: the variable eliminated by $\otimes$-ELIM may be placed anywhere in the context. This is reminiscent of 'deep inference' [Gug07], a proof system which allows rewriting of formulas anywhere in a structured context. Horsfall [Hor06] formulates the logic of bunched implication in a system with deep inference.

In our type theory, an eliminator of this kind fails because the resulting rule would not be closed under substitution. If $z^{\mathfrak{l}}$ where $\mathfrak{l}$ is bound in a *different*, outer use of $\otimes$-ELIM, then a substitution for $\mathfrak{l}$ might replace $\mathfrak{l}$ with a general slice of the palette. Our theory (intentionally) has no context extension with shape $z^s : A$ for s a slice.

Instead, we could make $\otimes$-ELIM a 'Frobenius'-eliminator. Rather than extending the context with $z^{\mathfrak{t}} : A \otimes B$, we extend the context with an arbitrary telescope $\Psi \mid \Delta$ that contains $z^{\mathfrak{p}} : A \otimes B$ so that $\mathfrak{p}$ in an arbitrary place in Ψ. The corresponding eliminator would look something like the following:

$$\frac{\begin{array}{c}\mathfrak{t} \prec \Phi \mid \Gamma \vdash \Psi\{\mathfrak{p} \prec \Psi'\} \mid \Delta, z^{\mathfrak{p}} : \textcircled{Σ}_{(\underline{x}:A)}\, B, \Delta' \ \mathsf{tele} \\ \mathfrak{t} \prec \Phi, \Psi \mid \Gamma, \Delta, z^{\mathfrak{p}} : \textcircled{Σ}_{(\underline{x}:A)}\, B, \Delta' \vdash C \ \mathsf{type} \\ \mathfrak{t} \prec \Phi, \Psi\{\mathfrak{p} \prec \Psi', \mathfrak{r} \otimes \mathfrak{b}\} \mid \Gamma, \Delta, x^{\mathfrak{r}} : A, y^{\mathfrak{b}} : B, \Delta'[x\ {}_{\mathfrak{r}}{\otimes}_{\mathfrak{b}}\ y/z] \vdash c : C[x\ {}_{\mathfrak{r}}{\otimes}_{\mathfrak{b}}\ y/z] \\ \mathfrak{t} \prec \Phi \mid \Gamma \vdash (\kappa \mid \theta) : \Psi \mid \Delta, z^{\mathfrak{p}} : \textcircled{Σ}_{(\underline{x}:A)}\, B, \Delta'\end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \mathsf{let}\ \Psi\{\mathfrak{p} \prec \Psi', \mathfrak{r} \otimes \mathfrak{b}\} \mid \Delta, x^{\mathfrak{r}} : A, y^{\mathfrak{b}} : B, \Delta'[x\ {}_{\mathfrak{r}}{\otimes}_{\mathfrak{b}}\ y/z] = (\kappa \mid \theta)\ \mathsf{in}\ c : C[\kappa \mid \theta]}\ \otimes\text{-ELIM-FROB?}$$

$$\begin{aligned}&(\mathsf{let}\ \Psi\{\mathfrak{p} \prec \Psi', \mathfrak{r} \otimes \mathfrak{b}\} \mid \Delta, x^{\mathfrak{r}} : A, y^{\mathfrak{b}} : B, \Delta'[x\ {}_{\mathfrak{r}}{\otimes}_{\mathfrak{b}}\ y/z] = (\kappa \mid \theta, a\ {}_{s_L}{\otimes}_{s_R}\ b/z, \theta')\ \mathsf{in}\ c) \\ &\equiv c[\kappa\{s_L/\mathfrak{r} \otimes s_R/\mathfrak{b}\} \mid \theta, a/x, b/y, \theta'] : C[\kappa \mid \theta, a \otimes b/z, \theta']\end{aligned}$$

An eliminator for $+$ in this style in the context of a bunched type theory is discussed in [Sch06, §4.3]. In ordinary dependent type theory without Π-types, the ordinary two-sided eliminator for Id-types is strictly weaker than a Frobenius eliminator for Id-types [GG08, Remark 3], and the one-sided eliminator is sufficient to derive a Frobenius Id-eliminator [Lum18; BK19]. In MTT [GKNB20], a Frobenius eliminator for Id-types locked behind a modality is not derivable and must be added as a base rule [Gra21].

The practical issue with the Frobenius rule for $\otimes$-types given above is the proliferation of new judgements and operations needed to make sense of it. These rules involve telescopes-of-telescopes (Δ'), a special substitution into these telescopes-of-telescopes ($\Delta'[x \otimes y/z]$ and $C[x \otimes y/z]$), identifying the middle of a telescope substitution ($\kappa \mid \theta, a\ {}_{s_L}{\otimes}_{s_R}\ b/z, \theta'$) and rewriting the middle of a telescope substitution to divide the $\otimes$-INTRO into its components: ($\kappa\{s_L/\mathfrak{r} \otimes s_R/\mathfrak{b}\} \mid \theta, a/x, b/y, \theta'$).

Pattern matching is much simpler, and is equivalent in power to the above Frobenius rule. It is also more economical: if we wish to perform induction on a deeply nested $\otimes$-type, the above eliminator forces us to eliminate each instance of $\otimes$ individually, possibly duplicating large sections of the context as part of the telescope Δ each time. With pattern matching we eliminate all of the $\otimes$-types at once, only extending the context once.

1.7 Related Work

We survey other work that tackles similar issues to the present type theory.

1.7.1 Modal Type Theories

The natural modality follows the work on constructive modal logics such constructive S4, intuitionistic linear logic, and adjoint logic — especially presentations with two contexts or different judgements for modal assumptions — and their generalisations with dependent types [Bar96; BW96; PD01; AMPR01; NPP08; Ree09; PR16]. The most directly related calculi are spatial type theory [Shu18] and the calculi for right adjoint functors/comonads [Clo18; BCMEPS20; GSB19]. While these previous works inform our design in Section 1.1, none consider a bireflective modality as we do here, and there were still some interesting design questions specific to our setting. For example, because $\natural$ is both a monad and a comonad, the previous work suggests two possible designs: [PD01; Shu18] have special context structure corresponding to a comonad, and the counit is a "silent" operation (not marked in the proof term), while [GSB19] has special context structure for a monad, and the unit is a silent operation. In our setting, we have both a monad and a comonad, but the "roundtrip" of the unit followed by the counit is *not* the identity, so we cannot make both the unit and counit silent. After some experimentation, we chose to make the unit silent and the counit explicit via "marked" variables.

The presence of both the unit and counit mean that there is no separation of the context into two zones: the MARKWK rule can weaken any term to one where marked and unmarked assumptions are interleaved. This is not so important for the $\natural$-fragment of the theory, because the rules for $\natural$-types mark the entire context. It becomes crucial once $\otimes$-types are included, because filtering the context to an arbitrary slice could mark variables anywhere in the context.

Some additional related work develops frameworks for modal type theories in general [LSR17; LRS22; GKNB20], but our setting is not quite an instance of these frameworks. The first [LSR17] lacks dependent types, but can describe the simply-typed fragment of our type theory. The in-progress extension of this work to dependent types [LRS22] should be able to capture the same semantic situation as that captured by our theory. The mode theories of the second [GKNB20], do not allow making the left adjoint types ("locks") into CwF morphisms, which corresponds in our setting to defining the natural operation on contexts as a context of individually marked variables rather than an operation that applies at once to the entire context.

Additionally, our contribution in Section 1.1 is an "optimised" syntax where structural rules are combined with other rules, and as much is admissible as possible, and the step from these frameworks to an optimised syntax is currently one that must be undertaken for each type theory separately in any case.

1.7.2 Bunched Type Theories

The existing work most comparable to our $\textcircled{\Sigma}$ and $\textcircled{\Pi}$ types is Schöpp and Stark's Bunched Dependent Type Theory [Sch06; SS04]. Their work is motivated by similar semantic considerations to ours: their goal is to find a type theory that has a natural interpretation in the 'Schanuel topos' [GP01], which has a useful monoidal structure. Beginning with ordinary dependent type theory, they add a context formation rule

$$\frac{\Gamma\ \mathsf{ctx} \qquad \Delta\ \mathsf{ctx}}{\Gamma \otimes \Delta\ \mathsf{ctx}}$$

i.e., there is no dependency across bunches. The corresponding monoidal pair type therefore requires the input types to be closed. The other critical difference to our theory is that the monoidal product is affine, meaning that the monoidal unit is the terminal object. This is certainly not the case in our models of interest. Affineness manifests syntactically as a permissive variable rule: every variable in scope is usable, regardless of where in the context it lies.

Cheney's $\lambda^{\Pi\text{И}}$ [Che09; Che12] is a subsystem of the previous type theory, where rather than adding binary bunches of contexts, there is a special context extension that creates a linear bunch:

$$\frac{\Gamma \ \mathsf{ctx} \qquad \alpha \ \mathsf{name}}{\Gamma \# a{:}\alpha \ \mathsf{ctx}}$$

which should be thought of as the bunch $\Gamma \otimes \alpha$. Here name is a special syntactic class, we are not permitted to form a bunch with an arbitrary type. Like Schöpp and Stark, the theory is inherently affine: the variable rule permits the use of any variable from the context, regardless of whether that variable is to the left of one of these special context extensions. Working with a simpler system allows Cheney to show decidability of typechecking and strong normalisation. Pitts, Matthiesen and Derikx's FreshMLTT [PMD15] similarly uses a special context extension for names that is treated affinely.

In his book on bunched implication, Pym [Pym02, §15.15] briefly speculates on what a dependent bunched theory might look like. The following formation rule for bunches is suggested:

$$\frac{\Gamma \vdash \Delta \ \mathsf{bunch}}{\vdash \Gamma \otimes \Delta \ \mathsf{bunch}}$$

which appears to build in a kind of affineness: variables on the left are permitted to be used on the right. A curious idea is considered: to allow variables of the same name and type to appear multiple times in the context[6]. For example, if we have $h : \textcircled{\scriptsize$\prod$}_{(x:A)} B(x) \multimap C(x) \multimap D$, the theory allows a term

$$(x : A) \otimes (y : B(x)) \otimes (x : A) \otimes (z : C(x)) \vdash h\langle x\rangle\langle y\rangle\langle z\rangle : D$$

It is not entirely clear how this might be interpreted categorically. In our system we sidestep this issue by requiring the argument type of a hom to be dull.

There have been some extensions of the simple $\alpha\lambda$-calculus. Collinson, Pym and Robinson [CPR08] add polymorphism, where both the context of type variables and context of ordinary variables are both bunched. Atkey [Atk04; Atk06] allows a more refined notion of separation between variables: rather than only having a binary tensor that separates two variables, an arbitrary symmetric relation describing 'separatedness' is allowed. An unpublished note by Zeilberger [Zei05] uses a second context zone to extend propositional BI with a modal operator corresponding to the $\Box$ modality of S4. Such a modality is a comonad presented in a positive style, making it more like the $\flat$ modality than $\natural$. These systems do not discuss dependent types.

A blog post by Krishnaswami [Kri11] tackles the issue of weakening and contraction not being admissible in the (simple) $\alpha\lambda$-calculus. Each shift between linear and non-linear bunches in the context is annotated with a label, and there is special term syntax that adds and removes these labels from the context. This is similar to our colour labels, but $\otimes$ bunches are also given a label, not only $\times$ bunches. This makes some forms of weakening admissible: any $\times$ bunch can silently have additional sub-bunches added.

But this does not capture all weakenings that are possible in the original $\alpha\lambda$-calculus, specifically those that create new $\times$ bunches. Recall that in the $\alpha\lambda$-calculus, $\times$ of contexts is represented by

[6]Pym even states that "the two occurrences of x may be seen as different 'colourings' of x"!

a semicolon and $\otimes$ by a comma. The weakening rule allows us to go from any judgement in context $\Delta_1, \Delta_2, \Delta_3$ to one in context $((\Delta_1, \Delta_2); \Gamma), \Delta_3$, corresponding to substitution by the projection $((\Delta_1 \otimes \Delta_2) \times \Gamma) \otimes \Delta_3 \to \Delta_1 \otimes \Delta_2 \otimes \Delta_3$.

Now consider weakening a hom application in the following way:

$$\textsc{wk}\ \dfrac{\multimap\text{-}\textsc{elim}\ \dfrac{\Delta_1 \vdash f : A \multimap B \qquad \Delta_2, \Delta_3 \vdash a : A}{\Delta_1, \Delta_2, \Delta_3 \vdash fa : B}}{((\Delta_1, \Delta_2); \Gamma), \Delta_3 \vdash fa : B}$$

In the rules of the blog post, the final context is denoted $r[s[\Delta_1, \Delta_2]; \Gamma], \Delta_3$ with the labels marking the shift between the two kinds of bunches (the names r and s chosen arbitrarily). There is no way to push this weakening into the premises of the $\multimap$-ELIM because of the way Γ binds Δ_1 and Δ_2 together, and the $\multimap$-ELIM rule does not build any weakening into its conclusion. This is not easily fixed: $\multimap$-ELIM would need to build in a way to reassociate the context, bind new labels, and weaken, which is exactly what the palette notation in our theory allows us to do.

1.7.3 Indexed Type Theories

Schreiber has hypothesised [Sch14, §3], [Sch17, §5.5], [nLabb] a kind of linear dependent type theory suitable for doing synthetic stable homotopy theory. The natural semantics for such a type theory would be in indexed monoidal (∞-)categories, so a category $\mathcal{C}$ with appropriate structure (typically a topos), and a functor $\mathcal{L} : \mathcal{C}^{\mathsf{op}} \to \mathrm{SymMonCat}$ equipped with base-change operations. We will see that this kind of structure can be recovered internally to our type theory in Section 2.4.

A formal syntax for working in such an indexed setting directly is Vákár's Intuitionistic Linear Dependent Type Theory (ILDTT) [Vák14; Vák15]. The judgements for types and terms have the shape

$$\Gamma \vdash A \ \mathsf{type} \qquad\qquad \Gamma; \Delta \vdash a : A$$

Contexts have an 'intuitionistic zone' Γ and a 'linear zone' Δ, both ordinary lists of variables with types. The Γ zone corresponds to an object of $\mathcal{C}$, and the Δ zone to an object of the fibre $\mathcal{L}(\Gamma)$. Assumptions in Γ may depend on previous assumptions in Γ, but Δ has no internal dependency. Types are formed only with reference to the intuitionistic zone, terms may reference the linear zone: a term $\Gamma; \Delta \vdash a : A$ corresponds to a map $a : \Delta \to A$ in the category $\mathcal{L}(\Gamma)$.

The non-dependent $\otimes$- and $\multimap$-types, together with other linear logic type formers such as &-types, are then defined with a fixed intuitionistic zone. These all correspond to the fibrewise structure available in each symmetric monoidal category $\mathcal{L}(\Gamma)$.

All types in this system are 'linear', and extending the Γ context with another assumption $x : A$ means using a comprehension operation ([Vák14, Definition 1]) to construct an object $\Gamma.A \in \mathcal{C}/\Gamma$ from $A \in \mathcal{L}(\Gamma)$. In a hypothetical model of ILDTT in parameterised spectra, this operation takes a parameterised spectrum to the space with Ω^∞ applied fibrewise [7]. This means that the only 'spaces' that can be manipulated in ILDTT are those of the form $\Omega^\infty A$ for $A : \mathrm{Spec}$, which is an extremely special class of spaces. ILDTT has a couple of other disadvantages: the dependent type-formers $\Sigma_!$

[7]We can define this operation internally to our type theory; see Definition 2.2.30

and $\Pi_!$ only allow their domain types to be used 'nonlinearly', i.e., via a trip through Ω^∞, and it seems unlikely that there is a notion of universe for this system.

Recent work of Isaev on Indexed Type Theory [Isa21] also divides the context into two pieces, but importantly divides types into two different sorts: ordinary types $\Gamma \vdash A$ type and indexed types $\Gamma \mid \Delta \vdash B$ ixtype. The fragment of the theory that does not concern the indexed context Δ is ordinary dependent type theory, avoiding the Ω^∞ issue of the previous theory. He suggests in a recent talk [Isa20] that one could add stability axioms and possibly type formers $\otimes$ and $\multimap$ to the indexed portion of the type theory to allow working with spectra over the base space described by Γ.

The downside of this approach is that, because base types and indexed types live in different sorts, one needs to replicate a lot of constructions in both the base theory and the indexed theory and assert axioms relating the two. For example, one needs a specialised 'indexed' version of the suspension and loop-space operations, whereas in our theory the ordinary operations apply even to types with linear information. Our theory also allows a 'type of all spectra': spectra are simply the $\natural$-null types, and so we can quantify over spectra using ordinary Σ and Π-types.

A third line of work with this sort of judgemental structure is the LNL_D theory of Krishnaswami, Pradic and Benton [KPB15]. The nonlinear and linear types again live in different sorts, with adjoint operators that transport between the two. Linear types can only depend on nonlinear types, so there is no 'genuine linear Σ-type', only a kind of pair type where the domain is a nonlinear type. As in ILDTT and ITT above, the linear operations all happen 'in a fibre', keeping the nonlinear context fixed.

LNL_D includes a number of equations, such as equality reflection, that make typechecking undecidable and a homotopical interpretation unavailable. These points are not the focus of their work however, and their dependent type theory is used to give a proof-theoretic analysis of an imperative programming language; such applications are far outside the scope of our work.

1.7.4 Quantitative Type Theories

Quantitative Type Theory, introduced by McBride [McB16] and extended by Atkey [Atk18], is closer to our theory: there is only one notion of type, with the linearity restrictions tracked by annotations on each variable in the context. These annotations are drawn from a fixed semiring, and control when a variable may be used. There are many non-dependent type theories where variables are annotated similarly [GSS92; POM14; BGMZ14; Abe15; AB20; AW18; WA20; OLE19], with the annotations controlled by some kind of ring-like algebraic structure.

The semiring used most often with QTT is the 'none-one-tons' semiring $\{0, 1, \omega\}$. A 0-use variable may not be used in a term, but can still be used to form types. These roughly correspond to our marked variables: those variables whose 'linear' content has been expended, but which is still available to be used in a 'nonlinear' way. In QTT this is often thought of as meaning the variable is not used 'at run-time', or is used 'non-computationally'. A context consisting all of 1-use variables would correspond in our theory to a context where each variable was in its own $\otimes$-bunch, and a context consisting of all ω-use variables would be one where each variable is labelled with the top colour.

A key difference to our theory is that *all* dependency in QTT occurs through 0-use variables: every type formation rule requires an entirely zeroed context. In our theory, we can certainly use non-marked variables to form types.

The bunched context structure that we use lets us describe variable use in a more refined way than QTT. For example, our context

$$\mathfrak{r} \otimes \mathfrak{b} \mid x^{\mathfrak{r}} : A, y^{\mathfrak{r}} : B, z^{\mathfrak{b}} : C$$

specifies that the red and blue pieces must be used linearly, but once we have access to the red part of the context, say, each of x and y can be used arbitrarily many times. This has no analogue in QTT: x and y would either be 1-use or ω-use, and neither option captures the correct relationship with z.

An advantage of QTT is that it unifies Σ-types with the $\otimes$-types and Π-types with the hom-types, recording the difference between each pair with different usage annotations in the type formers. QTT is also parameterised over the choice of usage semiring, with each choice of semiring giving a different notion of substructurality.

There are many other linear dependent type theories that share the property with QTT that variables are marked as either linear or nonlinear, and types may only depend on nonlinear variables. Among them, [CP96; WCPW03; IP98; IP99], and [Pym02, §15.3].

The Graded Modal Dependent Type Theory (GRTT) of Moon, Eades and Orchard [MEO21] is a significant generalisation of QTT which allows types to depend on variables in the same way that terms do. Each assumption in the context has an associated vector of usage values describing how it uses all the prior variables, and the variable rule ensures that these usages are coherent over the context. This added flexibility does not seem to capture bunched structures.

Proto-Quipper-M of Fu, Kishida and Selinger [FKS20; FKRS20] extends QTT in a different direction, instead carving out a subset of terms and types that behave nonlinearly: their 'parameter' terms and types. There is a 'shape' transformation on syntax that takes any type A to a parameter type $\mathsf{Sh}(A)$. This Sh operation is like our $\natural$ modality, but as a transformation on syntax it builds in the various equivalences that commute $\natural$ with the other type formers (Proposition 1.1.18, Proposition 1.1.11, Axiom C). Linear types may depend non-trivially on the shape of other linear types, and this makes the dependency structure of their $\otimes$ like ours: in a $\otimes$ type, the right type may depend on the nonlinear resources of the left.

Chapter 2

Mathematics

Many of the results of Section 2.1, Section 2.2 and Section 2.3 concerning the $\natural$ modality previously appeared in joint work with Dan Licata and Eric Finster [RFL21]. Axiom N arose out of conversations with Mike Shulman.

Our type theory is put to work: we prove some basic results in stable homotopy theory synthetically.

- In Section 2.1 we study the 'spectra'; the types such that $\natural \underline{A} \simeq 1$. We will see that they are pointed and that ordinary functions between them are automatically also pointed functions, avoiding the need to carry around proofs of pointedness.
- In Section 2.2 we describe a stability axiom that forces the internal category of synthetic spectra to be *stable*, in particular making $\Sigma \dashv \Omega$ an adjoint equivalence for these synthetic spectra.

  The stability axiom also has effects on the other type formers. Like any type in the theory, the universe $\mathcal{U}$ corresponds to a space-valued family of spectra. In Section 2.2.1 we use the stability axiom to determine what this family of spectra actually is.
- In Section 2.3 we suggest a second axiom, which connects synthetic spectra to 'analytic' spectra, defined concretely as sequences of pointed types with connecting maps. This uses some recent work on sequential colimits [SDR20] in type theory.
- In Section 2.4 we show that for any space $\underline{X}$ there is an internal category of parameterised spectra over $\underline{X}$, and for any internal map of spaces $\underline{X} \to \underline{Y}$, there is an induced "six functor formalism" between these internal categories.

2.1 Spectra

Informally, the $\natural$-modal types $A \simeq \natural \underline{A}$ are those with *no* synthetic spectral information—we think of $\natural \underline{A}$ as forgetting the spectra and replacing them with the trivial one, so if A is equivalent to $\natural \underline{A}$, then A had no spectral information to begin with. Dually, we can consider types with *only* synthetic spectral information, which can be defined by demanding that its underlying space is contractible.

In our intended model, such a type corresponds to an individual spectrum indexed by the point — finding the spectra among the parameterised families of spectra as those families where the index space is trivial.

Definition 2.1.1. A type E is a *spectrum* if $\natural\underline{E}$ is contractible. We have

$$\mathsf{Spec} :\equiv \textstyle\sum_{(E:\mathcal{U})} \mathsf{isContr}(\natural\underline{E})$$

for the type of spectra.

This is an instance of a general definition: for any monadic modality $\bigcirc$, a $\bigcirc$-*connected* type A is one such that $\bigcirc A$ is contractible [RSS20], so a spectrum is a $\natural$-connected type.

Remark 2.1.2. The type theory thus far admits more models than the intended model in parameterised spectra, so we should more properly refer to $\natural$-connected types by some other name, for example "reduced" types, by analogy with the reduced excisive functors of Goodwille calculus.

Remark 2.1.3. With spectra in mind, we can clarify why the syntactic property of a type $\underline{E}$ being dull is not the same as it being modal. Thinking of our intended model, the only spectrum that is also a space is the point. However, we can have a non-trivial spectrum that is dull, which describes the relationship of $\underline{E}$ to the *context*—a dull spectrum is one that only varies over the underlying space of the context.

These $\natural$-connected types are not difficult to come by:

Definition 2.1.4. If A is any type and $\underline{x} : \natural\underline{A}$ is a dull point of its base, the *spectrum over* $\underline{x}$ is the type

$$A_{\underline{x}} :\equiv \textstyle\sum_{(y:A)} (\underline{x} = \underline{y}^{\natural})$$

When $\underline{A}$ is dull, there is a canonical point of $\underline{A}_{\underline{x}}$ given by $(\underline{x}_{\natural}, \mathsf{refl}_{\underline{x}}) : \sum_{(y:\underline{A})} (\underline{x} = \underline{y}^{\natural})$.

Proposition 2.1.5. *$A_{\underline{x}}$ is a spectrum.*

Proof. We calculate:

$$\begin{aligned}
\natural\left(\textstyle\sum_{(y:A)} (\underline{x} = \underline{y}^{\natural})\right) &\simeq \textstyle\sum_{(u:\natural\underline{A})} \natural(\underline{x} = \underline{u_{\natural}}^{\natural}) && \text{(Proposition 1.1.18)}\\
&\simeq \textstyle\sum_{(u:\natural\underline{A})} (\underline{x} = \underline{u_{\natural}}^{\natural}) && (=_{\natural\underline{A}} \text{ is modal by Proposition 1.1.17})\\
&\equiv \textstyle\sum_{(u:\natural\underline{A})} (\underline{x} = u)
\end{aligned}$$

which is contractible. □

This lets us internalise the idea that every type is a 'space-valued family of spectra':

Corollary 2.1.6. *For any type A,*

$$A \simeq \textstyle\sum_{(x:\natural\underline{A})} A_{\underline{x}}$$

Proof. Expanding $A_{\underline{x}}$ on the right, and using the fact that $x = \underline{x}$:

$$\textstyle\sum_{(x:\natural\underline{A})}\sum_{(y:A)} (\underline{x} = \underline{y}^{\natural}) \simeq \sum_{(x:\natural\underline{A})}\sum_{(y:A)} (x = \underline{y}^{\natural})$$

Interchanging Σ-types we obtain a contractible pair. □

Recall the following standard definitions:

Definition 2.1.7. A *pointed type* is a pair (A, a) of a type A and a term $a : A$. A *pointed map* from (A, a) to (B, b) is a function $f : A \to B$ and a path $p : f(a) = b$. Write $\mathcal{U}_\star$ for the type of pointed types and

$$A \to_\star B :\equiv \textstyle\sum_{(f:A\to B)} f(a) = b$$

for the type of pointed maps.

Note that a type or map being pointed is structure, not a property. However, it is common to abuse notation and write $A \to_\star B$ rather than $(A, a) \to_\star (B, b)$ when the points of A and B can be inferred from context.

Definition 2.1.8. Any dull spectrum $\underline{E}$ has a canonical point $\star_{\underline{E}} : \underline{E}$ given by the composite $1 \to \natural\underline{E} \to \underline{E}$, where the first map is part of the data of $\natural\underline{E}$ being contractible, and the second is $\varepsilon_{\underline{E}}$.

Proposition 2.1.9. *Any dull map $\underline{f} : \underline{E} \to \underline{F}$ between dull spectra is a pointed map in a canonical way.*

Proof. In the diagram

$$\begin{tikzcd}
 & 1 \arrow[dl] \arrow[dr] & \\
\natural\underline{E} \arrow[rr, "\natural\underline{f}"] \arrow[d] & & \natural\underline{F} \arrow[d] \\
\underline{E} \arrow[rr, "\underline{f}"'] & & \underline{F}
\end{tikzcd}$$

the top triangle commutes by contractibility of $\natural\underline{F}$, and the bottom square commutes by naturality of the counit, so we have a path equipping $\underline{f}$ with the structure of a pointed map. ☐

We can't show that this pointing of $\underline{f} : \underline{E} \to \underline{F}$ is unique, but we can show that the underlying space of '$\underline{f}$ is pointed' is contractible, i.e. '$\underline{f}$ is pointed' is itself a spectrum. Hence:

Proposition 2.1.10. *If $\underline{E}$ and $\underline{F}$ are dull spectra then $\natural(\underline{E} \to \underline{F}) \simeq \natural(\underline{E} \to_\star \underline{F})$.*

Proof. We verify

$$\begin{aligned}
\natural(\underline{E} \to_\star \underline{F}) &\equiv \natural\left(\textstyle\sum_{(f:\underline{E}\to\underline{F})} f(\star_{\underline{E}}) = \star_{\underline{F}}\right) & \\
&\simeq \textstyle\sum_{(f:\natural(\underline{E}\to\underline{F}))} \natural(\underline{f}_{\natural}(\star_{\underline{E}}) = \star_{\underline{F}}) & \text{(Proposition 1.1.18)} \\
&\simeq \textstyle\sum_{(f:\natural(\underline{E}\to\underline{F}))} \underline{f}_{\natural}(\star_{\underline{E}})^{\natural} =_{\natural\underline{F}} \star_{\underline{F}}^{\natural} & \text{(Proposition 1.1.19)} \\
&\simeq \natural(\underline{E} \to \underline{F}) & (\natural\underline{F} \text{ is contractible})
\end{aligned}$$

☐

Remark 2.1.11. In the pointed spaces model of Section 3.3, these internal spectra interpreted as "synthetic pointed types". In the notation of that section, a $\natural$-connected type A has $\mathsf{B}A$ contractible, so $\mathsf{E}A$ is just a single type, and its section $\mathsf{p}A$ is just an element of $\mathsf{E}A$. Moreover, any function $f : A \to B$ between such types is a "synthetic pointed map"—inside the type theory, we do not need to carry around the data saying that f preserves the point, but in the pointed spaces model it will be interpreted as a function that preserves the sections of A and B, i.e. the points.

Spectra are closed under many operations:

Proposition 2.1.12. *Spectra are closed under Σ-types, $\otimes$-types, identity types, pullbacks, pushouts, suspensions and loop spaces.*

Proof. Closure under Σ, identity types and pullbacks holds for any lex modality [RSS20, Theorem 3.1]. Closure under $\otimes$ follows from Proposition 1.3.21. Closure under pushouts follows from Proposition 1.1.29. Suspensions and loop spaces (defined via identity types) are special cases of pullbacks and pushouts. $\square$

Proposition {C} 2.1.13. *Spectra are closed under $\multimap$-types.*

Proof. This is immediate from Axiom C: we have $\natural(\underline{E} \multimap \underline{F}) \simeq (\natural\underline{E} \to \natural\underline{F}) \simeq (1 \to 1) \simeq 1$. $\square$

For each of the type formers of our theory, we can investigate what the spectrum over each point in the base is.

Proposition 2.1.14. *For any types A and B and any fixed $\underline{x} : \natural\underline{A}$ and $\underline{y} : \natural\underline{B}$,*

$$(A \times B)_{(\underline{x}_\natural, \underline{y}_\natural)^\natural} \simeq A_{\underline{x}} \otimes B_{\underline{y}}.$$

More dependently, given $A : \mathcal{U}$ and $B : \underline{A} \to \mathcal{U}$, for fixed $\underline{x} : \natural\underline{A}$ and $\underline{y} : \natural\underline{B}(\underline{x})$ we have

$$\left(\Sigma_{(x:A)} B(x)\right)_{(\underline{x}_\natural, \underline{y}_\natural)^\natural} \simeq A_{\underline{x}} \otimes B(x)_{\underline{y}}.$$

In the interest of reducing clutter, we will simply write $(A \times B)_{(\underline{x},\underline{y})}$ etc., implicitly using the equivalence $\natural(\underline{A} \times \underline{B}) \simeq (\natural\underline{A} \times \natural\underline{B})$ of Lemma 1.1.17.

Proof. We write each type as the 'sum of its fibres' as in Corollary 2.1.6:

$$\begin{aligned} A \times B &\simeq \left(\Sigma_{(x:\natural\underline{A})} A_{\underline{x}}\right) \times \left(\Sigma_{(y:\natural\underline{B})} B_{\underline{y}}\right) \\ &\simeq \Sigma_{(x:\natural\underline{A})}\Sigma_{(y:\natural\underline{B})} \left(A_{\underline{x}} \times B_{\underline{y}}\right) \end{aligned}$$

It is clear that this map composed with the projection onto the first two factors is equal to the canonical map $A \times B \to \natural\underline{A} \times \natural\underline{B}$, and so the fibre of this map over $(\underline{x}, \underline{y})$ is observed to be $A_{\underline{x}} \times B_{\underline{y}}$. $\square$

We expect our $\otimes$-type to behave like an external tensor product and to be calculated 'pointwise', and indeed it is:

Proposition 2.1.15. *For any dull types $\underline{A}$ and $\underline{B}$ and any fixed $\underline{x} : \natural\underline{A}$ and $\underline{y} : \natural\underline{B}$,*

$$(\underline{A} \otimes \underline{B})_{(\underline{x},\underline{y})} \simeq \underline{A}_{\underline{x}} \otimes \underline{B}_{\underline{y}},$$

and so

$$\underline{A} \otimes \underline{B} \simeq \textstyle\sum_{((x,y):\natural\underline{A}\times\natural\underline{B})} \underline{A}_{\underline{x}} \otimes \underline{B}_{\underline{y}}.$$

More dependently, given $\underline{A} : \mathcal{U}$ and $\underline{B} : \underline{A} \to \mathcal{U}$, for fixed $\underline{x} : \natural\underline{A}$ and $\underline{y} : \natural\underline{B}(\underline{x})$ we have

$$\left(\textstyle\bigcirc\!\!\!\!\!\!\sum_{(\underline{x}:\underline{A})} \underline{B}(\underline{x})\right)_{(\underline{x},\underline{y})} \simeq \underline{A}_{\underline{x}} \otimes \underline{B}(\underline{x})_{\underline{y}}$$

Proof. This is very similar, but we use Proposition 1.3.12 to pull the spaces out of the $\otimes$-type:

$$\begin{aligned}\underline{A} \otimes \underline{B} &\simeq \left(\textstyle\sum_{(x:\natural\underline{A})} \underline{A}_{\underline{x}}\right) \otimes \left(\textstyle\sum_{(y:\natural\underline{B})} \underline{B}_{\underline{y}}\right) \\ &\simeq \textstyle\sum_{(x:\natural\underline{A})} \sum_{(y:\natural\underline{B})} \left(\underline{A}_{\underline{x}} \otimes \underline{B}_{\underline{y}}\right)\end{aligned}$$

Again, the projection onto the first two factors is equal to the canonical map $\underline{A} \otimes \underline{B} \to \natural\underline{A} \times \natural\underline{B}$, and so the fibre is $\underline{A}_{\underline{x}} \otimes \underline{B}_{\underline{y}}$ as we hoped. □

The situation for function types is not as simple; we do not know of a general characterisation of the spectra over some point $\underline{f} : \natural(\underline{A} \to \underline{B})$. We can however say the following:

Proposition 2.1.16. *For any $A, B : \mathcal{U}$, there is an equivalence*

$$A \to B \simeq \textstyle\sum_{(f:\natural\underline{A}\to\natural\underline{B})} \prod_{(x:\natural\underline{A})} A_{\underline{x}} \to B_{\underline{f}(\underline{x})}$$

Proof. It is enough to calculate the fibre of the map $(A \to B) \to (\natural\underline{A} \to \natural\underline{B})$ over some function $\underline{f} : \natural\underline{A} \to \natural\underline{B}$.

$$\begin{aligned}&\textstyle\sum_{(g:A\to B)} \underline{f} = (\lambda n.(\underline{g}(\underline{n}_\natural))^\natural) \\ &\simeq \textstyle\sum_{(g:A\to B)} \prod_{(n:\natural\underline{A})} \underline{f}(n) = (\underline{g}\langle\underline{n}_\natural\rangle)^\natural && \text{(Function Extensionality)} \\ &\simeq \textstyle\sum_{(g:A\to B)} \prod_{(a:\underline{A})} \underline{f}(\underline{a}^\natural) = (\underline{g}(\underline{a}))^\natural && \text{(Theorem 1.1.15)} \\ &\equiv \textstyle\prod_{(a:A)} \sum_{(b:B)} \underline{f}(\underline{a}^\natural) = \underline{b}^\natural && \text{(Univ. Property of } \Sigma) \\ &\simeq \textstyle\prod_{(a:A)} B_{\underline{f}(\underline{a}^\natural)} && \text{(Definition 2.1.4)} \\ &\simeq \textstyle\prod_{((x,a):\sum_{(x:\natural\underline{A})} A_{\underline{x}})} B_{\underline{f}(\underline{x})} && \text{(Corollary 2.1.6)} \\ &\simeq \textstyle\prod_{(x:\natural\underline{A})} A_{\underline{x}} \to B_{\underline{f}(\underline{x})} && \text{(Currying)}\end{aligned}$$

□

This is not enough to calculate the spectrum over a function $\underline{f} : \natural(\underline{A} \to \underline{B})$, because in general $A_{\underline{x}} \to B_{\underline{f}(\underline{x})}$ is not a spectrum, even though the domain and codomain are themselves spectra.
We can give a similar description of the hom type:

Proposition 2.1.17. *For any $\underline{A}, \underline{B} : \mathcal{U}$, there is an equivalence*

$$\underline{A} \multimap \underline{B} \simeq \textstyle\sum_{(f:\natural\underline{A}\to\natural\underline{B})}\prod_{(x:\natural\underline{A})}\underline{A}_x \multimap \underline{B}_{\underline{f}(\underline{x})}$$

Proof. It is enough to calculate the fibre of the map $(\underline{A} \multimap \underline{B}) \to (\natural\underline{A} \to \natural\underline{B})$ over some function $\underline{f} : \natural\underline{A} \to \natural\underline{B}$.

$$\begin{aligned}
&\textstyle\sum_{(g:\underline{A}\multimap\underline{B})}\underline{f} = (\lambda n.(\underline{g}\langle \underline{n}_\natural\rangle)^\natural) && \\
&\simeq \textstyle\sum_{(g:\underline{A}\multimap\underline{B})}\prod_{(n:\natural\underline{A})}\underline{f}(n) = (\underline{g}\langle \underline{n}_\natural\rangle)^\natural && \text{(Function Extensionality)}\\
&\simeq \textstyle\sum_{(g:\underline{A}\multimap\underline{B})}\prod_{(a:\underline{A})}\underline{f}(\underline{a}^\natural) = (\underline{g}\langle \underline{a}\rangle)^\natural && \text{(Theorem 1.1.15)}\\
&\simeq \textstyle\sum_{(g:\underline{A}\multimap\underline{B})}\textcircled{$\prod$}_{(a:\underline{A})}\underline{f}(\underline{a}^\natural) = (\underline{g}\langle \underline{a}\rangle)^\natural && \text{(Proposition 1.5.16)}\\
&\equiv \textstyle\textcircled{$\prod$}_{(a:\underline{A})}\sum_{(b:\underline{B})}\underline{f}(\underline{a}^\natural) = \underline{b}^\natural && \text{(Proposition 1.5.19)}\\
&\simeq \textstyle\textcircled{$\prod$}_{(a:\underline{A})}\underline{B}_{\underline{f}(\underline{a}^\natural)} && \text{(Definition 2.1.4)}\\
&\simeq \textstyle\textcircled{$\prod$}_{((x,a):\sum_{(x:\natural\underline{A})}\underline{A}_x)}\underline{B}_{\underline{f}(\underline{x})} && \text{(Corollary 2.1.6)}\\
&\simeq \textstyle\prod_{(x:\natural\underline{A})}\underline{A}_x \multimap \underline{B}_{\underline{f}(\underline{x})} && \text{(Proposition 1.5.17)}
\end{aligned}$$

□

Again, in general this does not let us calculate the spectrum $(\underline{A} \multimap \underline{B})_{\underline{h}}$ because $\underline{A}_x \multimap \underline{B}_{\underline{f}(\underline{x})}$ is not necessarily itself a spectrum. But Axiom C forces this to be so:

Proposition {C} 2.1.18. *For any $\underline{A}, \underline{B} : \mathcal{U}$ and $\underline{h} : \natural(\underline{A} \multimap \underline{B})$, there is an equivalence*

$$(\underline{A} \multimap \underline{B})_{\underline{h}} \simeq \textstyle\prod_{(x:\natural\underline{A})}\underline{A}_x \multimap \underline{B}_{\underline{h}_\natural\langle\underline{x}\rangle}$$

Proof. Follows by composing the above equivalence with Axiom C, and observing that the resulting map $(\underline{A} \multimap \underline{B}) \to \natural(\underline{A} \multimap \underline{B})$ is the canonical one. □

2.1.1 The Modality and Pointed Types

We need to record some interactions of $\natural$ with pointed types and pointed functions.

Lemma 2.1.19. *If B is a space then then $A \to_\star B$ is a space.*

Proof. By Lemma 1.1.17, spaces are closed under Σ, $=$ and Π where the codomain is a space. □

Proposition 2.1.20. *For any pointed type (A, a), there are canonical points $\underline{a} : \underline{A}$ and $\underline{a}^\natural : \natural\underline{A}$ for which the unit $\eta_A : A \to \natural\underline{A}$ and counit $\varepsilon_{\underline{A}} : \natural\underline{A} \to \underline{A}$ are pointed maps.*

Proof. Both are immediate from the definitions:

$$\begin{aligned}
(\lambda x.\underline{x}^\natural)(a) &\equiv \underline{a}^\natural\\
(\lambda n.n_\natural)(\underline{a}^\natural) &\equiv \underline{a}^\natural{}_\natural \equiv \underline{a}
\end{aligned}$$

□

The following is a standard characterisation of equivalence between Σ-types:

Lemma 2.1.21. *Suppose we have type families $P : A \to \mathcal{U}$ and $Q : B \to \mathcal{U}$. If we have an equivalence $f : A \simeq B$ and a family of equivalences $g_x : P(x) \simeq Q(f(x))$ then the map*

$$(\lambda(x,p).(f(x), g_x(p))) : \textstyle\sum_{(x:A)} P(x) \to \sum_{(y:B)} Q(y)$$

is an equivalence.

Proof. We have $\Sigma : (\Sigma A : \mathcal{U}.A \to \mathcal{U}) \to \mathcal{U}$, and the given data is equivalent to $(A, P) =_{(\Sigma A:\mathcal{U}.A\to\mathcal{U})} (B, Q)$ by univalence and the definitions of paths in Σ and Π types. □

Proposition 2.1.22. *Let (A, a) and (B, b) be pointed types with B a space. Precomposition with $A \to_\star \natural\underline{A}$ induces an equivalence*

$$(\natural\underline{A} \to_\star B) \simeq (A \to_\star B)$$

Proof. We have seen (Theorem 1.1.15) that there is an equivalence

$$w : (\natural\underline{A} \to B) \simeq (A \to B).$$

So we need to show that for every $f : \natural\underline{A} \to B$, there is an equivalence

$$(f(\underline{a}^\natural) = b) \simeq (w(f)(a) = b)$$

but w is precomposition with the unit, so the type on the right is also $(f(\underline{a}^\natural) = b)$. □

Proposition 2.1.23. *For dull pointed types $(\underline{A}, \underline{a})$ and $(\underline{B}, \underline{b})$, post-composition with $(-)_\natural : \natural\underline{B} \to \underline{B}$ induces an equivalence*

$$\natural(\natural\underline{A} \to_\star \natural\underline{B}) \simeq \natural(\natural\underline{A} \to_\star \underline{B})$$

Proof. Because ♮ commutes with Σ by Proposition 1.1.18, we can again apply Lemma 2.1.21 so it is enough to show that the map

$$\textstyle\sum_{(f:\natural(\natural\underline{A}\to\natural\underline{B}))} \natural(\underline{f}_\natural(\underline{a}^\natural) = \underline{b}^\natural) \simeq \sum_{(g:\natural(\natural\underline{A}\to\underline{B}))} \natural(\underline{g}_\natural(\underline{a}^\natural) = \underline{b})$$

induced by post-composition is an equivalence.

First, we know by Theorem 1.1.24 that

$$w : \natural(\natural\underline{A} \to \natural\underline{B}) \simeq \natural(\natural\underline{A} \to \underline{B})$$

Explicitly, this is

$$w(f) \equiv (\lambda x.\underline{f}_\natural(x)_\natural)^\natural$$

So we need to produce for every $f : \natural(\natural\underline{A} \to \natural\underline{B})$ an equivalence

$$\natural(\underline{f}_\natural(\underline{a}^\natural) = \underline{b}^\natural) \simeq \natural((\underline{w(f)})_\natural(\underline{a}^\natural) = \underline{b})$$

On the right we calculate

$$(\underline{w}(\underline{f}))_\natural(\underline{a}^\natural) \equiv [(\lambda x.\underline{f}_\natural(x)_\natural)^\natural]_\natural(\underline{a}^\natural) \equiv (\lambda x.\underline{f}_\natural(x)_\natural)(\underline{a}^\natural) \equiv \underline{f}_\natural(\underline{a}^\natural)_\natural$$

And then we have

$$\begin{aligned} \natural(\underline{f}_\natural(\underline{a}^\natural) = \underline{b}^\natural) &\equiv \natural((\underline{f}_\natural(\underline{a}^\natural))_\natural{}^\natural = \underline{b}^\natural) && \text{(by the } \eta\text{-rule)} \\ &\equiv \natural((\underline{w}(\underline{f}))_\natural(\underline{a}^\natural)^\natural = \underline{b}^\natural) && \text{(by the above calculation)} \\ &\simeq \natural((\underline{w}(\underline{f}))_\natural(\underline{a}^\natural) = \underline{b}) && \text{(by left exactness and idempotence)} \end{aligned}$$

One then has to check that this equivalence is the one that is induced by composition with the counit, but this follows from the definition of the left-exactness equivalence. □

Proposition 2.1.24 (Dull Self-adjointness for Pointed Types). *For any dull pointed types A and B, there is an equivalence*

$$\natural(\underline{A} \to_\star \natural\underline{B}) \simeq \natural(\natural\underline{A} \to_\star \underline{B})$$

Proof. By Proposition 2.1.22 (where $\natural\underline{B}$ is a space by Proposition 1.1.11), $(\underline{A} \to_\star \natural\underline{B}) \simeq (\natural\underline{A} \to_\star \natural\underline{B})$. Using univalence, $(\lambda X : \mathcal{U}.\natural\underline{X})$ preserves equivalences, so we have

$$\natural(\underline{A} \to_\star \natural\underline{B}) \simeq \natural(\natural\underline{A} \to_\star \natural\underline{B})$$

Then by Proposition 2.1.23 we have $\natural(\natural\underline{A} \to_\star \natural\underline{B}) \simeq \natural(\natural\underline{A} \to_\star \underline{B})$. □

2.1.2 Synthetic Stabilisation

One important feature of spectra that we would like to capture synthetically is an adjunction relating spaces and spectra:

$$\begin{tikzcd} \mathrm{Spec} \arrow[d, bend left, "\Omega^\infty"] \arrow[d, phantom, "\dashv"] \\ \mathrm{Space}_\star \arrow[u, bend left, "\Sigma^\infty"] \end{tikzcd}$$

The Σ^∞ operation 'freely stabilises' a pointed space. Once we have imposed some axioms, we will find that the homotopy groups of the spectrum $\Sigma^\infty X$ correspond to the stable homotopy groups of the space X.

Recall from the introduction that the ∞-category of spectra may be defined as the limit

$$\mathrm{Spec} = \varprojlim \left(\cdots \xrightarrow{\Omega} \mathcal{S}_* \xrightarrow{\Omega} \mathcal{S}_* \xrightarrow{\Omega} \mathcal{S}_* \right)$$

and that spectra can presented concretely as a sequence of pointed spaces $X_0, X_1, \ldots$ with $X_0 \simeq \Omega X_1$, $X_1 \simeq \Omega X_2, \ldots$. Then Ω^∞ sends a spectrum to the space X_0. It is named Ω^∞ as we have 'applied

Ω infinitely many times' to reach the end of the limit. The left adjoint Σ^∞ sends a space X to the spectrification of its suspension pre-spectrum (see Section 2.3 below), i.e. we can make a sequence of pointed spaces $X, \Sigma X, \Sigma\Sigma X, \ldots$ with maps $X \to_\star \Omega\Sigma X$, $\Sigma X \to_\star \Omega\Sigma\Sigma X \ldots$ given by the unit of the $\Sigma \dashv \Omega$ adjunction. These maps are not equivalences, which is corrected by replacing each space by a certain colimit, a process called spectrification.

Here, we instead define an abstract/synthetic variant of this adjunction, which can be interpreted in this way in PSpec. In this section, we will use the notation $\mathbb{S}$ and $\Sigma^\infty/\Omega^\infty$ because of the intended interpretation in PSpec, but these will be abstract operations that exist in any model.

Recall from Proposition 2.1.12 that our synthetic spectra are closed under Σ and Ω. These are the *ordinary* suspension and loop-space constructions for types, which do turn out to correspond semantically to the correct operations of suspension and loop space on actual spectra. This is because Spec is a full subcategory of PSpec, and the limits and colimits defining suspension and loop space already land in Spec, so they coincide with the limits and colimits calculated in the subcategory.

Proposition 2.1.25. *For dull spectra $\underline{E}$ and $\underline{F}$, suspension and loop space are dull adjoints, i.e.*

$$\natural(\Sigma\underline{E} \to \underline{F}) \simeq \natural(\underline{E} \to \Omega\underline{F})$$

In this statement we are implicitly using the canonical point $\star_{\underline{F}} : \underline{F}$ to form the loop-space $\Omega\underline{F}$.

Proof. By Proposition 2.1.10, it is equivalent to show

$$\natural(\Sigma\underline{E} \to_\star \underline{F}) \simeq \natural(\underline{E} \to_\star \Omega\underline{F})$$

so this follows by functoriality of $\natural$ from the fact that suspension and loop space are adjoint for (general) types and pointed maps [HoTTBook, Lemma 6.5.4]. □

Using $\mathbb{S}$, we define $\Omega^\infty : \natural\mathrm{Spec} \to \mathrm{Space}_\star$ as follows:

Definition 2.1.26. For $\underline{E}$: Spec, define the space

$$\Omega^\infty\underline{E} :\equiv \natural(\mathbb{S} \to \underline{E})$$

which is pointed by the constant zero map $(\lambda\_.\star_{\underline{E}})^\natural$.

Remark 2.1.27. To gain intuition for why this is the right definition, we can consider the pointed spaces model of Section 3.3. The analogue of $\mathbb{S}$ there is $\mathbb{B}$, the two-point space S^0 living over the point. The base of the function type $\mathbb{B} \to \underline{E}$ is equivalent to the type of all basepoint-preserving maps from the upstairs of $\mathbb{B}$ to the upstairs of $\underline{E}$. The basepoint is fixed, and there is one free point that can be mapped to any point of the upstairs $\underline{E}$. So the base space of $\mathbb{B} \to \underline{E}$ indeed corresponds to the upstairs of $\underline{E}$.

For spectra, similar reasoning applies, using the $\Sigma^\infty \dashv \Omega^\infty$ adjunction more explicitly, and that the sphere spectrum is the stabilisation of the two-point space:

$$\mathrm{Map}_{\mathrm{Spec}}(\mathbb{S}, E) \simeq \mathrm{Map}_{\mathrm{Spec}}(\Sigma^\infty S^0, E) \simeq \mathrm{Map}_{\mathcal{S}_\star}(S^0, \Omega^\infty E) \simeq \mathrm{Map}_{\mathcal{S}}(1, \Omega^\infty E) \simeq \Omega^\infty E$$

So we take '$\mathrm{Map}_{\mathrm{Spec}}(\mathbb{S}, E)$' as our definition of $\Omega^\infty E$.

One application of Ω^∞ is defining the homotopy groups of a spectrum, in the sense that is used in stable homotopy theory:

Definition 2.1.28. The $\mathbb{S}$-*shifted homotopy groups* of a spectrum $\underline{E}$ are defined by

$$\pi_n^s \underline{E} :\equiv \pi_n(\Omega^\infty \underline{E})$$

Simply calculating $\pi_n(\underline{E})$ will not do the correct thing: we will see later that, after imposing a stability axiom (Section 2.2), spectra are ∞-connected 2.2.12, so $\pi_n(\underline{E}) \simeq 1$ always.

The left adjoint $\Sigma^\infty : \mathrm{Space}_\star \to \mathrm{Spec}$ then has a surprisingly simple formula:

Definition 2.1.29. For X a pointed space, define $\Sigma^\infty X :\equiv X \wedge \mathbb{S}$.

This is left adjoint roughly because maps of spectra are pointed maps, and $- \wedge A \dashv A \to_\star -$ for any A.

First, we check that Σ^∞ lands in spectra:

Proposition 2.1.30. *For any space X, $\Sigma^\infty X$ is a spectrum.*

Proof. The pushout diagrams defining $X \vee \mathbb{S}$ and $X \wedge \mathbb{S}$ are

$$\begin{tikzcd} 1 \arrow[r] \arrow[d] & X \arrow[d] \\ \mathbb{S} \arrow[r] & X \vee \mathbb{S} \arrow[ul, phantom, "\ulcorner"] \end{tikzcd} \qquad \begin{tikzcd} X \vee \mathbb{S} \arrow[r] \arrow[d] & X \times \mathbb{S} \arrow[d] \\ 1 \arrow[r] & \Sigma^\infty X \arrow[ul, phantom, "\ulcorner"] \end{tikzcd}$$

Because $\natural$ preserves pushouts by Proposition 1.1.29 and products by Proposition 1.1.18, and $\mathbb{S}$ is a spectrum, so $\natural\mathbb{S} \simeq 1$, we can calculate

$$\begin{tikzcd} 1 \arrow[r] \arrow[d] & \natural\underline{X} \arrow[d] \\ 1 \arrow[r] & \natural(\underline{X} \vee \mathbb{S}) \simeq \natural\underline{X} \arrow[ul, phantom, "\ulcorner"] \end{tikzcd} \qquad \begin{tikzcd} \natural\underline{X} \arrow[r] \arrow[d] & \natural\underline{X} \arrow[d] \\ 1 \arrow[r] & \natural(\Sigma^\infty \underline{X}) \simeq 1 \arrow[ul, phantom, "\ulcorner"] \end{tikzcd}$$

because the top of the second diagram is the identity, and the pushout of a map along the identity is the same map.

□

Proposition 2.1.31. *Σ^∞ and Ω^∞ are (dull) adjoints: there is an equivalence*

$$\natural(\Sigma^\infty \underline{X} \to \underline{E}) \simeq (\underline{X} \to_\star \Omega^\infty \underline{E})$$

where $\underline{X}$ is a pointed space and $\underline{E}$ is a spectrum.

Proof.

$$\begin{aligned}
\underline{X} \to_\star \Omega^\infty \underline{E} &\equiv (\underline{X} \to_\star \natural(\mathbb{S} \to_\star \underline{E})) \\
&\simeq \natural(\underline{X} \to_\star \natural(\mathbb{S} \to_\star \underline{E})) && (- \to_\star \natural - \text{ is a space }) \\
&\equiv \natural(\natural\underline{X} \to_\star (\mathbb{S} \to_\star \underline{E})) && (\text{Proposition 2.1.24}) \\
&\simeq \natural(\underline{X} \to_\star (\mathbb{S} \to_\star \underline{E})) && (\underline{X} \text{ is a space}) \\
&\simeq \natural(\underline{X} \wedge \mathbb{S} \to_\star \underline{E}) \\
&\equiv \natural(\Sigma^\infty \underline{X} \to_\star \underline{E}) \\
&\simeq \natural(\Sigma^\infty \underline{X} \to \underline{E}) && (\text{Proposition 2.1.10, Proposition 2.1.30})
\end{aligned}$$

Here we use currying for pointed maps $A \to_\star (B \to_\star C) \simeq (A \wedge B) \to_\star C$, which has been proved in type theory [van18, Theorem 4.3.28]. □

Remark 2.1.32. In the pointed spaces model, for a pointed space X, $X \wedge \mathbb{B}$ works out to be $\mathsf{B}\underline{X} \wedge 1$ in the base (which is indeed contractible). Over this point, we are calculating the cofibre of the map

$$X + 1 \simeq X \vee S^0 \to X \times S^0 \simeq X + X$$

which is the identity on the first component X and the basepoint inclusion on the second. So the first copy of X is crushed to a point, and identified with the basepoint of the second copy of X.

In all, the operation takes a pointed 'space' X to a 'spectrum' $\Sigma^\infty X$, moving X from the base to the unique fibre, internalising the analytic pointing as synthetic pointing.

In PSpec we have to work a little harder to justify this definition. Σ^∞ is a functor from pointed spaces to spectra, but we can precompose with the functor $(-)_+ : \mathcal{S} \to \mathcal{S}_*$ to get a functor from unpointed spaces. This is typically written $\Sigma^\infty_+ : \mathcal{S} \to \mathrm{Spec}$, and is left adjoint to the functor given by computing the pointed space Ω^∞ and forgetting the basepoint.

As a left adjoint, Σ^∞_+ preserves colimits, and because every space is the colimit of its points, we calculate

$$\Sigma^\infty_+(X) \simeq \Sigma^\infty_+(\mathrm{colim}_X\, 1) \simeq \mathrm{colim}_X\, \Sigma^\infty_+(1) \simeq \mathrm{colim}_X\, \Sigma^\infty(S^0) \simeq \mathrm{colim}_X\, \mathbb{S}$$

I.e., the colimit of the constant diagram on X at $\mathbb{S}$. For us, such a constant diagram is given by the parameterised spectrum $X \times \mathbb{S}$, placing a copy of $\mathbb{S}$ over every point in X. The desired colimit can be computed as the cofibre of $X \to X \times \mathbb{S}$ which is the basepoint in the second component, crushing the base X to a point. We can take this as our type-theoretic definition of Σ^∞_+:

$$\Sigma^\infty_+(X) :\equiv \mathrm{cofib}(X \to X \times \mathbb{S})$$

To get back to $\Sigma^\infty(X)$ for X a pointed type, we have to crush the 'extra copy of $\mathbb{S}$' that was added over the new basepoint: this is now taking the cofibre of the composite $\mathbb{S} \to X \times \mathbb{S} \to \Sigma^\infty_+ X$ which is the basepoint in the *first* component. Combining the two cofibre diagrams, in all we have calculated the smash product $X \wedge \mathbb{S}$.

2.1.3 Commutativity of the Adjunctions

So far, we have seen that we have the three adjunctions in the following diagram (the two vertical sides are the same):

$$\begin{tikzcd}
\mathsf{Space}_\star \arrow[r, bend left, "\Sigma"] \arrow[r, phantom, "\perp"] \arrow[d, bend left, "\Sigma^\infty"] \arrow[d, phantom, "\vdash"] & \mathsf{Space}_\star \arrow[l, bend left, "\Omega"] \arrow[d, bend left, "\Sigma^\infty"] \arrow[d, phantom, "\vdash"] \\
\mathsf{Spec} \arrow[r, bend left, "\Sigma"] \arrow[r, phantom, "\perp"] \arrow[u, bend left, "\Omega^\infty"] & \mathsf{Spec} \arrow[l, bend left, "\Omega"] \arrow[u, bend left, "\Omega^\infty"]
\end{tikzcd}$$

We can also show that the diagram commutes.

Remark 2.1.33. In pointed spaces it is clear this should be true, as the $\Sigma^\infty \dashv \Omega^\infty$ adjunction simply moves pointed spaces into the fibre over a point and back, and the suspension/loop space of a spectrum is calculating the suspension/loop space of the unique fibre.

For actual spectra, consider a spectrum again presented as a sequence of pointed spaces $(E_0, E_1, \dots)$. The loop space of such an Ω-spectrum can be calculated by shifting the spaces over by one, giving Ω-spectrum $(\Omega E_0, E_0, E_1, \dots)$, so extracting the 0th space commutes with calculating the loop space.

The suspension of an Ω-spectrum can be calculated by shifting the spaces the other way, giving $(E_1, E_2, \dots)$. Leaving aside spectrification briefly, $\Sigma^\infty X$ is given by the prespectrum $(X, \Sigma X, \Sigma^2 X, \dots)$, so $\Sigma^\infty \Sigma X \equiv (\Sigma X, \Sigma^2 X, \dots)$ is exactly $\Sigma^\infty X$ shifted by one.

Proposition 2.1.34. *Ω^∞ commutes with the ordinary loop space operation Ω:*

$$\Omega^\infty \Omega \underline{E} \simeq_\star \Omega \Omega^\infty \underline{E}$$

naturally in $\underline{E}$.

Note that the left-hand side is the loop space on spectra (and therefore pointed) types, while the right-hand side is the loop space on pointed spaces, but both are implemented by the usual loop space on types.

This is easy, after the following basic fact about loop spaces.

Lemma 2.1.35. *For any pointed types A and B,*

$$(A \to_\star \Omega B) \simeq_\star \Omega(A \to_\star B)$$

naturally in B.

Proof.

$$\begin{aligned}
(A \to_\star \Omega B) &\simeq_\star (A \to_\star (S^1 \to_\star B)) \\
&\simeq_\star (S^1 \to_\star (A \to_\star B)) \\
&\simeq_\star \Omega(A \to_\star B)
\end{aligned}$$

The first equivalence is essentially the universal property of the higher inductive circle S^1, while the second follows from exchange for function types. □

Proof of Proposition.

$$\Omega^\infty \Omega \underline{E} \equiv \natural(\mathbb{S} \to_\star \Omega \underline{E}) \simeq_\star \natural(\Omega(\mathbb{S} \to_\star \underline{E})) \simeq_\star \Omega\natural(\mathbb{S} \to_\star \underline{E}) \equiv \Omega\Omega^\infty \underline{E}$$

For any pointed type, $\natural\Omega(\underline{A}, \underline{a}) \simeq_\star \Omega(\natural \underline{A}, \underline{a}^\natural)$ follows from Proposition 1.1.19. □

In the other direction, we have:

Proposition 2.1.36. Σ^∞ *commutes with* Σ*:*

$$\Sigma\Sigma^\infty X \simeq \Sigma^\infty \Sigma X$$

naturally in X*.*

Proof. First, some properties of the suspension and smash higher inductive types in ordinary homotopy type theory are that $\Sigma X \simeq S^1 \wedge X$ [Bru16, Proposition 4.2.1] and smash is associative [van18, Definition 4.3.33]. Thus, we can calculate

$$\Sigma\Sigma^\infty X \equiv \Sigma(X \wedge \mathbb{S}) \simeq S^1 \wedge (X \wedge \mathbb{S}) \simeq (S^1 \wedge X) \wedge \mathbb{S} \simeq \Sigma^\infty \Sigma X$$

□

2.2 Stability

Classically, the category of spectra has a number of special properties, including a zero object, biproducts (products and coproducts that are isomorphic), any pushout square is a pullback square and vice versa, and suspension and loop space are inverse (not only adjoint). Thus far, our definition of synthetic spectra has a zero object: 1 is initial as well as terminal because $\natural(1 \to \underline{E}) \simeq \natural\underline{E} \simeq 1$. To establish the other properties, it turns out to suffice to add an apparently weaker axiom asserting that products and coproducts in Spec coincide—we will show that this implies stability in the sense of pullback and pushout squares coinciding, which in turn implies that suspension and loop space are an equivalence.

Externalised, the argument in this section amounts to a proof of the following:

Theorem 2.2.1. *If an* ∞*-locus is semiadditive then it is stable.*

Spectra are pointed, so the coproduct in the category of spectra is the wedge $\vee$, in the following sense:

Proposition 2.2.2. *For any dull spectra* $\underline{E}$*,* $\underline{F}$ *and* $\underline{G}$*, there is an equivalence*

$$\natural(\underline{E} \vee \underline{F} \to \underline{G}) \simeq \natural(\underline{E} \to \underline{G}) \times \natural(\underline{F} \to \underline{G})$$

induced by the inclusions of $\underline{E}$ *and* $\underline{F}$ *into* $\underline{E} \vee \underline{F}$*.*

Proof. We have

$$\begin{aligned}
\natural(\underline{E} \vee \underline{F} \to \underline{G}) &\simeq \natural(\underline{E} \vee \underline{F} \to_\star \underline{G}) \\
&\simeq \natural((\underline{E} \to_\star \underline{G}) \times (\underline{F} \to_\star \underline{G})) \\
&\simeq \natural(\underline{E} \to_\star \underline{G}) \times \natural(\underline{F} \to_\star \underline{G}) \\
&\simeq \natural(\underline{E} \to \underline{G}) \times \natural(\underline{F} \to \underline{G})
\end{aligned}$$

by using Proposition 2.1.10 to add and remove the pointing of the maps. □

For any pointed types A and B there is a canonical wedge inclusion $\iota_{A,B} : A \vee B \to A \times B$.

Axiom S. *For any dull spectra $\underline{E}$ and $\underline{F}$, the wedge inclusion $\iota_{\underline{E},\underline{F}} : \underline{E} \vee \underline{F} \to \underline{E} \times \underline{F}$ is an equivalence.*

This is asserting the existence of a term

$$\mathsf{ax}_S : \prod_{(E:\natural\mathrm{Spec})} \prod_{(F:\natural\mathrm{Spec})} \mathsf{isEquiv}(\iota_{E_\natural, F_\natural})$$

Remark 2.2.3. Some care must be taken when devising an internal version of the external fact that the subcategory Spec $\hookrightarrow$ PSpec is stable. The obvious thing to try is an axiom that applies to any E : Spec. The issue is that when asserting the existence of a closed term, like the axiom above, the term can be weakened to any ambient context.

Semantically, the question is whether the axiom holds in all slice categories $P\mathrm{Spec}/\Gamma$, and it does *not*. A 'spectrum' (in the sense of a $\natural$-connected type) in $P\mathrm{Spec}/\Gamma$ consists of a family of spectra E over the base space of Γ, together with spectrum maps between the fibres of E and the corresponding fibres of Γ. This category is not stable in general, as it lacks a zero object (among other things). If the context Γ itself consists of a single spectrum F, then the existence of a zero object would imply that any map $E \to F$ splits.

The axiom *does* hold in $P\mathrm{Spec}/X$ when X is a space, as $\natural$-connected types in $P\mathrm{Spec}/X$ are exactly the families of spectra over X, comprising a stable category. Syntactically this requirement corresponds to having a dull context, hence the restriction to types in $\natural$Spec in Axiom S.

We can weaken our ax_S to an arbitrary context Γ and still be safe, as the spectra $E_\natural$ and $F_\natural$ will continue to only depend on the underlying space of Γ.

Remark 2.2.4. This axiom rules out the pointed spaces model, as the wedge and product of 'spectra' in that model correspond to the ordinary wedge and product of the pointed types in the unique fibre. But it does *not* rule out the trivial model where $\natural$ is the identity functor. There, the only 'spectrum' is the point, and the wedge inclusion $1 \vee 1 \to 1 \times 1$ is certainly an equivalence.

Proposition {S} 2.2.5. *For two dull spectra $\underline{E}$ and $\underline{F}$, the smash product $\underline{E} \wedge \underline{F}$ and the join $\underline{E} * \underline{F}$ are both contractible.*

Proof. Recall that the smash product $\underline{E} \wedge \underline{F}$ is the cofibre of the wedge inclusion, i.e. the pushout

$$\begin{tikzcd}
\underline{E} \vee \underline{F} \arrow[r] \arrow[d] & \underline{E} \times \underline{F} \arrow[d] \\
1 \arrow[r] \arrow[ur, phantom, "\ulcorner"] & \underline{E} \wedge \underline{F}
\end{tikzcd}$$

Axiom S asserts that the top map is an equivalence, so the bottom-right corner is equivalent to the bottom-left. For the join, we have $\underline{E} * \underline{F} \simeq \Sigma(\underline{E} \wedge \underline{F})$ by [Cav15, Theorem 4.19], and the suspension of a contractible type is contractible. □

Here, $\underline{E} \wedge \underline{F}$ is the smash product *of types*, and is not related to the $\otimes$ type.

Next, we show that Axiom S indeed makes pullbacks and pushouts in Spec coincide, mainly as a consequence of the Little Blakers-Massey Theorem [ABFJ20]. As a first step:

Lemma 2.2.6. *For any pointed type A, there is a pullback square*

$$\begin{tikzcd}
\Sigma\Omega A \arrow[r] \arrow[d] & A \vee A \arrow[d, "\iota"] \\
A \arrow[r, "\Delta"'] & A \times A
\end{tikzcd}$$

where the map on the left is the counit of the $\Sigma \dashv \Omega$ adjunction.

This will follow quickly from the following consequence of descent for pushouts:

Theorem 2.2.7 ([Rij18, Theorem 2.2.12])**.** *Consider a commuting cube of types*

$$\begin{tikzcd}
& S' \arrow[dl] \arrow[d] \arrow[dr] & \\
A' \arrow[d] \arrow[dr] & S \arrow[dl] \arrow[dr] & B' \arrow[dl] \arrow[d] \\
A \arrow[dr] & X' \arrow[d] & B \arrow[dl] \\
& X &
\end{tikzcd}$$

and suppose the vertical squares are pullback squares. Then the commuting square

$$\begin{tikzcd}
A' \sqcup_{S'} B' \arrow[r] \arrow[d] & X' \arrow[d] \\
A \sqcup_{S'} B \arrow[r] & X
\end{tikzcd}$$

is a pullback square.

Proof of Lemma 2.2.6. Consider the commutative cube

$$\begin{tikzcd}
& \Omega A \arrow[dl] \arrow[d] \arrow[dr] & \\
1 \arrow[d] \arrow[dr] & 1 \arrow[dl] \arrow[dr] & 1 \arrow[dl] \arrow[d] \\
A \arrow[dr] & A \arrow[d] & A \arrow[dl] \\
& A \times A, &
\end{tikzcd}$$

where the vertical map $A \to A \times A$ is the diagonal, and the maps $A \to A \times A$ on the left and right are the identity on one component and constant at the point on the other. All of the vertical squares are pullbacks.

Now note that the pushout of the top span is $\Sigma\Omega A$, and the pushout of the bottom span is $A \vee A$, so by the above theorem we have the desired pullback square (reflected diagonally). □

Corollary {S} 2.2.8. *For any dull spectrum, the canonical map $\Sigma\Omega\underline{E} \to \underline{E}$ is an equivalence.*

Proof. By Lemma 2.2.6, this map is the pullback of the wedge inclusion ι along Δ. By Axiom S, ι is an equivalence, and the pullback of an equivalence along any map is an equivalence. □

Definition 2.2.9. Recall [HoTTBook, Definition 7.5.1] that a type A is n-connected if its n-truncation is contractible, and a map is n-connected if its fibre is an n-connected type for all base points. A type or function is ∞-connected if it is n-connected for every n.

We will use the following Lemmas about n-connected types:

Lemma 2.2.10. *Suppose that a type A is 0-connected and $P : A \to \mathsf{Prop}$ is a family of propositions (-1-types). Then if $P(a)$ holds for some $a : A$, then $P(a')$ holds for all a'.*

Proof. Assume an a such that $P(a)$ and another point a'. Since A is 0-connected, its 0-truncation is contractible, and therefore its 0-truncation is a proposition [HoTTBook, Theorem 7.1.10], so we get a path $|a'| =_{\|A\|_0} |a|$. Commuting the truncation with the loop space gives $\|a' = a\|_{-1}$ [HoTTBook, Theorem 7.3.12]. That is, a' is merely equal to a. But $P(a')$ is a proposition by assumption, so to prove it, we can assume $a' = a$, and then transport the assumed proof of $P(a)$. □

Lemma 2.2.11.
- *If A and B are 0-connected then so is $A \times B$*
- *If A, B are 0-connected and C is 1-connected, then for any maps $f : A \to C$ and $g : B \to C$, the pullback $A \times_C B$ is 0-connected.*

Proof. For the first part, truncation preserves products [HoTTBook, Theorem 7.3.8], so to show $\|A \times B\|_0$ is contractible, we can equivalently show that $\|A\|_0 \times \|B\|_0$ is contractible. But $\|A\|_0$ and $\|B\|_0$ are contractible by assumption, and $1 \times 1 \simeq 1$.

For the second, the pullback is given by the type $\sum_{((x,y):A\times B)} f(x) =_C g(y)$. By [HoTTBook, Theorem 7.3.9, Theorem 7.3.12], we have

$$\begin{aligned}
&\left\|\textstyle\sum_{((x,y):A\times B)} f(x) =_C g(y)\right\|_0 \\
&\simeq \left\|\textstyle\sum_{((x,y):A\times B)} \|f(x) =_C g(y)\|_0\right\|_0 \\
&\simeq \left\|\textstyle\sum_{((x,y):A\times B)} |f(x)| =_{\|C\|_1} |g(y)|\right\|_0 \\
&\simeq \|A \times B\|_0 \\
&\simeq 1
\end{aligned}$$

The second-to-last step is because $\|C\|_1$ is contractible by assumption, so any identity type in it is as well, and the last step is by the previous part, since A and B are 0-connected. □

Corollary {S} 2.2.12. *Dull spectra and dull maps between them are ∞-connected.*

Proof. For types, we prove by induction on n that every dull spectrum $\underline{E}$ is n-connected. Every type is (-2)-connected, since the (-2)-truncation is contractible by definition. For the inductive step, suppose $\underline{E}$ is a spectrum, and we want to show that it is $(n+1)$-connected. Then $\Omega\underline{E}$ is also a spectrum by Proposition 2.1.12 and dull, so by the inductive hypothesis (which applies to all dull spectra, so in particular $\Omega\underline{E}$) it is n-connected. Suspension increases connectivity by 1 [HoTTBook, Theorem 8.2.1], so $\Sigma\Omega\underline{E}$ is $(n+1)$-connected. But by Corollary 2.2.8, $\underline{E} \simeq \Sigma\Omega\underline{E}$, so $\underline{E}$ is $(n+1)$-connected as well.

Now for maps, fix an n. The fibre of a dull map $\underline{f} : \underline{E} \to \underline{F}$ over the basepoint is the type $\mathsf{fib}_{\underline{f}}(\star_{\underline{F}}) :\equiv \sum_{(x:\underline{E})} \underline{f}(x) = \star_{\underline{F}}$, which is a dull spectrum and thus is an ∞-connected type by above, and therefore n-connected. We now show that this implies that all fibres are n-connected using Lemma 2.2.10. First, $\underline{F}$ is a dull spectrum, and thus by the previous part it is in particular 0-connected. For any type A, the type "A is n-connected" is a proposition, because it unfolds to "the n-truncation of A is contractible", and being contractible (like all h-levels) is a proposition [HoTTBook, Theorem 7.1.7]. Thus, $\mathsf{fib}_{\underline{f}}(\star_{\underline{F}})$ being n-connected implies the same for $\mathsf{fib}_{\underline{f}}(x)$ for any $x : \underline{F}$, so $\underline{f}$ is an n-connected map. □

We will now make use of the Little Blakers-Massey Theorem and its dual, which is the specialisation of the Generalised Blakers-Massey Theorem [ABFJ20] to the identity modality. The Generalised Theorem has been formalised in The HoTT Library [Shu19b]. First, some notation:

Definition 2.2.13. For $f : A \to B$, let Δf denote the canonical map $A \to A \times_B A$.

Definition 2.2.14. For $f : A \to B$ and $g : C \to D$, the pushout product $f \mathbin{\square} g$ is defined to be the canonical gap map

$$\begin{tikzcd}
A \times C \arrow[r] \arrow[d] & A \times D \arrow[d] \arrow[ddr, bend left] & \\
B \times C \arrow[r] \arrow[drr, bend right] & P \arrow[dr, dotted, "f \square g"] \arrow[ul, phantom, "\ulcorner", very near start] & \\
 & & B \times D
\end{tikzcd}$$

where $P :\equiv (A \times D) \sqcup_{A\times C} (B \times C)$.

Lemma 2.2.15. *The fibres of Δf are given by*

$$\mathsf{fib}_{\Delta f}(a, a', p) \simeq ((a, p) =_{\mathsf{fib}_f(f(a'))} (a', \mathsf{refl}_{f(a)})),$$

and in particular,

$$\mathsf{fib}_{\Delta f}(a, a, \mathsf{refl}_a) \simeq \Omega_{(a,\mathsf{refl}_a)}\mathsf{fib}_f(f(a)).$$

Proof. Direct calculation:

$$\begin{aligned}
\mathsf{fib}_{\Delta f}(a, a', p) &:\equiv \textstyle\sum_{(x:A)}(x, x, \mathsf{refl}_{f(x)}) = (a, a', p) \\
&\simeq \textstyle\sum_{(x:A)}\sum_{(l:x=a)}\sum_{(r:x=a')} !\mathsf{ap}_f(l) \cdot \mathsf{ap}_f(r) = p \\
&\simeq \textstyle\sum_{(r:a=a')} \mathsf{ap}_f(r) = p \\
&\simeq \textstyle\sum_{(r:a=a')} !\mathsf{ap}_f(r) \cdot p = \mathsf{refl}_{f(a')} \\
&\simeq \textstyle\sum_{(r:a=a')} r_*(p) = \mathsf{refl}_{f(a')} \\
&\simeq (a, p) = (a', \mathsf{refl}_{f(a')})
\end{aligned}$$

$\square$

Proposition 2.2.16. *The pushout product is the 'external fibrewise join', in the sense that for $b : B$ and $d : D$,*

$$\mathsf{fib}_{f \square g}(b, d) \simeq \mathsf{fib}_f(b) * \mathsf{fib}_g(d)$$

Proof. This is another application of descent. Consider the cube

$$\begin{tikzcd}
& \mathsf{fib}_f(b) \times \mathsf{fib}_g(d) \arrow[dl] \arrow[d] \arrow[dr] & \\
\mathsf{fib}_f(b) \arrow[d] \arrow[dr] & A \times C \arrow[dl] \arrow[dr] & \mathsf{fib}_g(d) \arrow[dl] \arrow[d] \\
A \times D \arrow[dr] & 1 \arrow[d] & B \times C \arrow[dl] \\
& B \times D &
\end{tikzcd}$$

The back vertical map extracts the A and C from the fibres, the side vertical maps extract the A or C and pair with b or d in the other component, and the front vertical map is (b, d). By singleton contractibility and paths in products being component-wise, all the vertical sides are pullback squares. The corresponding square

$$\begin{tikzcd}
\mathsf{fib}_f(b) * \mathsf{fib}_g(d) \arrow[r] \arrow[d] & 1 \arrow[d] \\
P \arrow[r, "f \square g"'] & B \times D
\end{tikzcd}$$

is a pullback by Theorem 2.2.7, so $\mathsf{fib}_f(b) * \mathsf{fib}_g(d)$ is equivalent to the fibre of $f \square g$ at (b, d). $\square$

We now quote the Theorem that actually does the work:

Theorem 2.2.17 (Little Blakers-Massey Theorem, [ABFJ20, Corollary 4.1.4, Theorem 3.5.1])**.** *Consider the following square.*

$$\begin{tikzcd}
Z \arrow[r, "g"] \arrow[d, "f"'] & Y \arrow[d, "h"] \\
X \arrow[r, "k"'] & W
\end{tikzcd}$$

- *If the square is a pushout and $\Delta f \mathbin{\square} \Delta g$ is an equivalence, then the square is also a pullback.*
- *If the square is a pullback and $h \mathbin{\square} k$ is an equivalence, then the square is also a pushout.*

Theorem {S} 2.2.18. *A dull commutative square in spectra is a pullback square iff it is a pushout square.*

Proof. Suppose a dull commutative square in spectra

$$\begin{tikzcd} \underline{E} \arrow[r, "\underline{g}"] \arrow[d, "\underline{f}"'] & \underline{G} \arrow[d, "\underline{h}"] \\ \underline{F} \arrow[r, "\underline{k}"'] & \underline{H} \end{tikzcd}$$

By Definition 2.1.8 and Proposition 2.1.9, we have base points $\star_{\underline{E}}, \star_{\underline{F}}, \star_{\underline{G}}, \star_{\underline{H}}$, and the maps are all pointed. Thus, the fibres all have dull points— e.g. $\mathsf{fib}_{\underline{f}}(\star_{\underline{F}}) :\equiv \sum_{(z:\underline{E})} \underline{f}(z) = \star_{\underline{F}}$ has a point given by $\star_{\underline{E}}$ and the path showing $\underline{f}$ is pointed.

To use Theorem 2.2.17, we just need to show that $\Delta\underline{f} \mathbin{\square} \Delta\underline{g}$ and $\underline{h} \mathbin{\square} \underline{k}$ are equivalences, which we will do using the "contractible fibres" definition of equivalence.

The fibre of $\Delta\underline{f} \mathbin{\square} \Delta\underline{g}$ over the basepoint $p_0 :\equiv ((\star_{\underline{E}}, \star_{\underline{E}}, \mathsf{refl}_{\underline{f}(\star_{\underline{E}})}), (\star_{\underline{E}}, \star_{\underline{E}}, \mathsf{refl}_{\underline{g}(\star_{\underline{E}})}))$ is $\Omega(\mathsf{fib}_{\underline{f}}(\star_{\underline{F}})) * \Omega(\mathsf{fib}_{\underline{g}}(\star_{\underline{G}}))$ by Lemma 2.2.15 (where $f(\star_{\underline{E}}) = \star_{\underline{F}}$ and $g(\star_{\underline{E}}) = \star_{\underline{G}}$ because the maps are pointed) and Proposition 2.2.16. Again by Proposition 2.2.16, the fibre of $\underline{h} \mathbin{\square} \underline{k}$ over $\underline{q_0} :\equiv (\star_{\underline{H}}, \star_{\underline{H}})$ is $\mathsf{fib}_{\underline{h}}(\star_{\underline{H}}) * \mathsf{fib}_{\underline{k}}(\star_{\underline{H}})$. To show that these are both contractible, by Proposition 2.2.5, it suffices to show that the pieces of the join are dull spectra. By Proposition 2.1.12, spectra are closed under fibres and loop spaces, and for the basepoints these types are dull because the basepoints are. This shows that $\mathsf{fib}_{\Delta\underline{f}\square\Delta\underline{g}}(\underline{p_0})$ and $\mathsf{fib}_{\underline{h}\square\underline{k}}(\underline{q_0})$ are contractible.

Since the fibres over $\underline{p_0}$ and $\underline{q_0}$ are contractible, to show that general fibres $\mathsf{fib}_{\Delta\underline{f}\square\Delta\underline{g}}(p)$ and $\mathsf{fib}_{\underline{h}\square\underline{k}}(q)$ over any p and q are contractible, by Lemma 2.2.10 it suffices to show that $p : (\underline{E} \times_{\underline{F}} \underline{E}) \times (\underline{E} \times_{\underline{G}} \underline{E})$ and $q : \underline{H} \times \underline{H}$ are elements of 0-connected types, since being contractible is a proposition. For q, by Corollary 2.2.12, $\underline{H}$ is ∞-connected and in particular 0-connected, so $\underline{H} \times \underline{H}$ is 0-connected by Lemma 2.2.11. For p, again using closure under products, we need to show that $(\underline{E} \times_{\underline{F}} \underline{E})$ and $(\underline{E} \times_{\underline{G}} \underline{E})$ are 0-connected. By Corollary 2.2.12, $\underline{E}$ is 0-connected and $\underline{F}, \underline{G}$ are 1-connected, so the pullbacks are 0-connected as well by Lemma 2.2.11. This use of 0-connectedness is necessary because the fibre of a map between dull spectra is only dull when the point over which we are taking the fibre is, so the argument in the previous paragraph can be applied directly to p and q. $\square$

We showed that the counit $\Sigma\Omega\underline{E} \to \underline{E}$ is an equivalence in Corollary 2.2.8, and we can now show that the unit is as well:

Corollary {S} 2.2.19. *For any $\underline{E}$: Spec, the unit map $\underline{E} \to \Omega\Sigma\underline{E}$ is an equivalence.*

Proof. The pushout square defining the suspension is

$$\begin{tikzcd} \underline{E} \arrow[r] \arrow[d] \arrow[dr, phantom, "\lrcorner" very near start, "\ulcorner" very near end] & 1 \arrow[d, "\mathsf{s}"] \\ 1 \arrow[r, "\mathsf{n}"'] & \Sigma\underline{E} \end{tikzcd}$$

By Theorem 2.2.18, this is also a pullback square. However, we also have a pullback square

$$\begin{tikzcd}
\mathsf{n} =_{\Sigma \underline{E}} \mathsf{s} \arrow[r] \arrow[d] \arrow[dr, phantom, "\lrcorner", very near start] & 1 \arrow[d, "\mathsf{s}"] \\
1 \arrow[r, "\mathsf{n}"'] \arrow[ur, phantom, "\ulcorner", very near end] & \Sigma \underline{E}
\end{tikzcd}$$

so uniqueness of pullbacks gives an equivalence $\underline{E} \simeq (\mathsf{n} = \mathsf{s})$. If we consider the suspension to be pointed by n, then $\Omega(\Sigma \underline{E})$ is the type $\mathsf{n} = \mathsf{n}$. But since $\underline{E}$ is pointed, we have a path $\mathsf{mer}(\star_{\underline{E}}) : \mathsf{s} = \mathsf{n}$, and composition with this path gives an equivalence $(\mathsf{n} = \mathsf{s}) \simeq (\mathsf{n} = \mathsf{n})$, so $\underline{E} \cong \Omega(\Sigma \underline{E})$.

The unit of the adjunction $\Sigma \vdash \Omega$ sends $e : \underline{E}$ to the path $\mathsf{mer}(e) \cdot ! \mathsf{mer}(\star_{\underline{E}})$, so the composite equivalence is indeed the unit. □

2.2.1 The Universe and Function Types as Families

Like any type, the universe can be decomposed as a family of spectra over a space via Corollary 2.1.6. The stability axiom lets us describe this decomposition surprisingly explicitly: in this section we will show that

$$\mathcal{U} \simeq \sum_{(X:\mathrm{Space})} \sum_{(E:X \to \natural\mathrm{Spec})} \prod_{(x:X)} \Sigma(\underline{E}(\underline{x}))_{\natural}$$

So, a term of the universe consists of a parameterised spectrum specified by a space X : Space and a family of *dull* spectra $E : X \to \natural$Spec, together with an element of the spectrum $\prod_{(x:X)} \Sigma(\underline{E}(\underline{x}))_{\natural}$. This latter spectrum can be thought of as the 'twisted cohomology' of the parameterised spectrum (X, E), fibrewise shifted one dimension.

There is a description of function types as a family that looks perhaps even more peculiar. If $\underline{E}$ and $\underline{F}$ are dull spectra, then

$$(\underline{E} \to \underline{F}) \simeq \natural(\underline{E} \to \underline{F}) \times \underline{F}$$

and so the spectral information of $\underline{E} \to \underline{F}$ is completely determined by the spectral information of the codomain.

We begin with a fact that holds for any lex modality:

Proposition 2.2.20.

$$\mathcal{U} \simeq \sum_{(X:\mathrm{Space})} X \to \mathrm{Spec}$$

Proof. Consider the unit map $\natural(-) : \mathcal{U} \to$ Space. We can write $\mathcal{U}$ as the sum of fibres of this map, by [HoTTBook, Lemma 4.8.2].

$$\mathcal{U} \simeq \sum_{(X:\mathrm{Space})} \sum_{(B:\mathcal{U})} (\natural \underline{B} = X)$$

Using univalence, this is

$$\mathcal{U} \simeq \sum_{(X:\mathrm{Space})} \sum_{(B:\mathcal{U})} (\natural \underline{B} \simeq X)$$

So it remains to show that $(X \to \mathrm{Spec}) \simeq \sum_{(B:\mathcal{U})} (\natural \underline{B} \simeq X)$. For this, first use the fact that Spec classifies $\natural$-connected maps, by [RSS20, Corollary 1.42]:

$$(X \to \mathrm{Spec}) \simeq \sum_{(B:\mathcal{U})} \sum_{(f:B \to X)} \mathsf{is\text{-}\natural\text{-}connected}(f)$$

Now we need that $\sum_{(f:B\to X)} \mathsf{is}\text{-}\natural\text{-}\mathsf{connected}(f)$ is equivalent to $(\natural\underline{B} \simeq X)$. We are in the special case that X is modal, so by the monadic universal property of $\natural$,

$$\sum_{(f:B\to X)} \mathsf{is}\text{-}\natural\text{-}\mathsf{connected}(f) \simeq \sum_{(g:\natural\underline{B}\to X)} \mathsf{is}\text{-}\natural\text{-}\mathsf{connected}(g\circ\eta_B)$$

Because $\natural$ is lex, the 2-out-of-3 property holds for $\natural$-connected functions [RSS20, Theorem 3.1.xi]. The unit η_B is always $\natural$-connected [RSS20, Theorem 1.32], so $g\circ\eta_B$ is $\natural$-connected iff g is. Finally, any function between modal types is modal, and any function that is both modal and connected is an equivalence, so

$$\mathsf{is}\text{-}\natural\text{-}\mathsf{connected}(g\circ\eta_B) \simeq \mathsf{is}\text{-}\natural\text{-}\mathsf{connected}(g) \simeq \mathsf{isEquiv}(g)$$

and we are done. □

Proposition 2.2.21. *The above equivalence is implemented by* $A \mapsto (\natural\underline{A}, \lambda x.A_{\underline{x}})$ *in the forward direction and* $(X, E) \mapsto \sum_{(x:X)} E(x)$ *in the backwards direction.*

Maybe confusingly, the equivalence of Proposition 2.2.20 does not exhibit $\mathcal{U}$ *itself* as a space (X : Space) equipped with a family of spectra over each point ($X \to$ Spec). The problem is that Spec is not itself a spectrum, and so neither is $X \to$ Spec.

The type $\natural\mathcal{U}$ also contains data from the $X \to$ Spec component:

Proposition 2.2.22. *There is an equivalence*

$$\natural\mathcal{U} \simeq \textstyle\sum_{(X:\mathrm{Space})} X \to \natural\mathrm{Spec}$$

Proof. From the equivalence earlier we know

$$\begin{aligned} \natural\mathcal{U} &\simeq \natural\left(\textstyle\sum_{(X:\mathrm{Space})} X \to \mathrm{Spec}\right) \\ &\simeq \textstyle\sum_{(X:\natural\mathrm{Space})} \natural\left(\underline{X}_\natural \to \mathrm{Spec}\right) && (\natural \text{ preserves } \Sigma) \\ &\simeq \textstyle\sum_{(X:\mathrm{Space})} \natural\left(X \to \mathrm{Spec}\right) && (\mathrm{Space\ is\ modal}) \\ &\simeq \textstyle\sum_{(X:\mathrm{Space})} X \to \natural\mathrm{Spec} && (X \text{ is modal}) \end{aligned}$$

□

We can use this to decompose the universe more explicitly into a space with a family of spectra over it.

Proposition 2.2.23.

$$\mathcal{U} \simeq \textstyle\sum_{(X:\mathrm{Space})}\sum_{(E:X\to\natural\mathrm{Spec})}\prod_{(x:X)}\mathrm{Spec}_{\underline{E}(\underline{x})}$$

where $\mathrm{Spec}_{\underline{E}(\underline{x})}$ *is the spectrum over the point* $\underline{E}(\underline{x}) : \natural\mathrm{Spec}$.

Proof. Fix a pair $(\underline{X}, \underline{E}) : \sum_{(X:\mathrm{Space})} X \to \natural\mathrm{Spec}$, thinking of the pair as an element of $\natural\mathcal{U}$. We have a 'unit' map

$$\left(\textstyle\sum_{(X:\mathrm{Space})} X \to \mathrm{Spec}\right) \to \left(\textstyle\sum_{(X:\mathrm{Space})} X \to \natural\mathrm{Spec}\right)$$

given by transporting the unit $\eta_{\mathcal{U}} : \mathcal{U} \to \natural\mathcal{U}$ along the equivalences of Proposition 2.2.20 and Proposition 2.2.22, and we want to calculate the fibre of this map over $(\underline{X}, \underline{E})$. This map is just postcomposition with the unit $\eta_{\mathsf{Spec}} : \mathsf{Spec} \to \natural\mathsf{Spec}$, and so this fibre is given by

$$\mathcal{U}_{(\underline{X},\underline{E})} \simeq \mathsf{fib}_{\eta\circ -}(\underline{E}) \simeq \prod_{(x:\underline{X})} \mathsf{fib}_{\eta_{\mathsf{Spec}}}(\underline{E}(\underline{x})) \simeq \prod_{(x:\underline{X})} \mathsf{Spec}_{\underline{E}(\underline{x})}$$

using an easy Lemma on the fibre of the postcomposition map. □

Lemma 2.2.24. *Let $A, B, C : \mathcal{U}$, and $g : B \to C$. Then for $h : A \to C$,*

$$\mathsf{fib}_{g\circ -}(h) \simeq \prod_{(a:A)} \mathsf{fib}_g(h(a))$$

Proof.

$$\begin{aligned}\mathsf{fib}_{g\circ -}(h) &\simeq \sum_{(f:A\to B)} g \circ f = h \\ &\simeq \sum_{(f:A\to B)} \prod_{(a:A)} g(f(a)) = h(a) \\ &\simeq \prod_{(a:A)} \sum_{(b:B)} g(b) = h(a) \\ &\simeq \prod_{(a:A)} \mathsf{fib}_g(h(a))\end{aligned}$$

□

The above equivalence is about as far as we can go without assuming any axioms.

Our goal now is to show that the stability axiom implies that $\mathsf{Spec}_{\underline{E}} \simeq \Sigma\underline{E}_\natural$ for any dull spectrum $\underline{E}$. The stability axiom implies that Σ is an equivalence, so it will suffice to show that $\underline{E}_\natural \simeq \mathsf{Spec}_{\Omega\underline{E}}$ for any $\underline{E}$: our original claim will follow when this is instantiated at $\Sigma\underline{E}$, transporting $\mathsf{Spec}_{\Omega\Sigma\underline{E}}$ along the $\Omega\Sigma\underline{E} \to \underline{E}$ equivalence.

Lemma 2.2.25. *There is a pullback square*

$$\begin{tikzcd} 1 \arrow[r] \arrow[d, "\star_{\underline{E}}"'] & \sum_{(F:\mathsf{Spec}_{\Omega\underline{E}})} F \arrow[d, "\mathsf{pr}_1"] \\ \underline{E} \arrow[r] & \mathsf{Spec}_{\Omega\underline{E}} \end{tikzcd}$$

Proof. Define the map $\underline{E} \to \mathsf{Spec}_{\Omega\underline{E}}$ by sending $e : \underline{E}$ to $(e = \star_{\underline{E}}) : \mathsf{Spec}$. This type does lie over $\Omega\underline{E}$ via the equivalence $(\underline{e} = \star_{\underline{E}})^\natural \simeq (\star_{\underline{E}} = \star_{\underline{E}})^\natural$ given by composing with the canonical map $\star_{\underline{E}} = \underline{e}$.

Now the pullback can be calculated as

$$\begin{aligned}&\sum_{(e:\underline{E})} \sum_{(F:\mathsf{Spec}_{\Omega\underline{E}})} F \times (F = (e = \star_{\underline{E}})) \\ &\simeq \sum_{(e:\underline{E})} (e = \star_{\underline{E}}) \\ &\simeq 1\end{aligned}$$

using singleton elimination twice. □

To define the map the other way, we need to use stability.

Lemma {S} 2.2.26. *There is a pullback square*

$$\begin{tikzcd}
\sum_{(F:\mathsf{Spec}_{\Omega\underline{E}})} F \arrow[r] \arrow[d, "\mathsf{pr}_1"'] & 1 \arrow[d, "\star_{\underline{E}}"] \\
\mathsf{Spec}_{\Omega\underline{E}} \arrow[r] & \underline{E}
\end{tikzcd}$$

Proof. Note that both $\mathsf{Spec}_{\Omega\underline{E}}$ and $\sum_{(F:\mathsf{Spec}_{\Omega\underline{E}})} F$ are spectra. The first because it 'the spectrum over a point in a type' and these are always spectra by Proposition 2.1.5, and the second because $\natural$ preserves Σ-types and both components are spectra.

The basepoint point of $\mathsf{Spec}_{\Omega\underline{E}}$ is given by $\Omega\underline{E}$, so the fibre of the left map is the type $\Omega\underline{E}$ and by stability we have a fibre sequence

$$\Omega\underline{E} \to \left(\sum_{(F:\mathsf{Spec}_{\Omega\underline{E}})} F\right) \to \mathsf{Spec}_{\Omega\underline{E}} \to \underline{E}$$

Looking at the right side of that sequence gives the pullback square we wanted. $\square$

Lemma {S} 2.2.27. *The maps $\underline{E} \to \mathsf{Spec}_{\Omega\underline{E}}$ and $\mathsf{Spec}_{\Omega\underline{E}} \to \underline{E}$ in the above pullback squares are inverse.*

Proof. Pasting the pullback squares together one way, we have that

$$\begin{tikzcd}
1 \arrow[r] \arrow[d] & 1 \arrow[d] \\
\underline{E} \arrow[r] & \underline{E}
\end{tikzcd}$$

is a pullback. By stability it is also a pushout, and because the top and left maps are necessarily the canonical inclusion of the basepoint, the map along the bottom must be equal to the identity.

Pasting the other way around, we have a pullback square

$$\begin{tikzcd}
\sum_{(F:\mathsf{Spec}_{\Omega\underline{E}})} F \arrow[r] \arrow[d, "\mathsf{pr}_1"'] & \sum_{(F:\mathsf{Spec}_{\Omega\underline{E}})} F \arrow[d, "\mathsf{pr}_1"] \\
\mathsf{Spec}_{\Omega\underline{E}} \arrow[r, "g"'] & \mathsf{Spec}_{\Omega\underline{E}}
\end{tikzcd}$$

But we can also calculate the pullback of the same two maps in the 'ordinary' way:

$$\begin{tikzcd}
\sum_{(F:\mathsf{Spec}_{\Omega\underline{E}})} \underline{g}(F) \arrow[r, "{(F,f)\mapsto(\underline{g}(F),f)}"] \arrow[d, "\mathsf{pr}_1"'] & \sum_{(F:\mathsf{Spec}_{\Omega\underline{E}})} F \arrow[d, "\mathsf{pr}_1"] \\
\mathsf{Spec}_{\Omega\underline{E}} \arrow[r, "\underline{g}"'] & \mathsf{Spec}_{\Omega\underline{E}}
\end{tikzcd}$$

By the uniqueness of pullbacks, there is an equivalence

$$\begin{tikzcd}
\sum_{(F:\mathsf{Spec}_{\Omega\underline{E}})} \underline{g}(F) \arrow[rr, "\sim"] \arrow[dr, "\mathsf{pr}_1"'] & & \sum_{(F:\mathsf{Spec}_{\Omega\underline{E}})} F \arrow[dl, "\mathsf{pr}_1"] \\
& \mathsf{Spec}_{\Omega\underline{E}} &
\end{tikzcd}$$

Therefore there is a fibrewise equivalence $\prod_{(F:\mathrm{Spec}_{\Omega \underline{E}})} \underline{g}(F) \simeq F$, which by univalence gives a homotopy $\prod_{(F:\mathrm{Spec}_{\Omega \underline{E}})} \underline{g}(F) = F$, so $\underline{g}$ is equal to the identity. □

Combining the above equivalence with Proposition 2.2.23, we have our result:

Theorem {S} 2.2.28.

$$\mathcal{U} \simeq \sum_{(X:\mathrm{Space})} \sum_{(E:X \to \natural\mathrm{Spec})} \prod_{(x:X)} \Sigma(\underline{E}(\underline{x}))_\natural$$

□

The above has a couple of curious consequences. We know from the above that $\sum_{(F:\mathrm{Spec}_{\underline{E}})} F$ is contractible for any spectrum $\underline{E}$, so:

Proposition {S} 2.2.29. *The map $\sum_{(E:\mathrm{Spec})} E \to \mathrm{Spec}$ classifying $\natural$-connected maps is equal to the counit $\natural\mathrm{Spec} \to \mathrm{Spec}$.*

Proof. First, as with any type, $\mathrm{Spec} \simeq \sum_{(N:\natural\mathrm{Spec})} \mathrm{Spec}_{\underline{N}}$. Then

$$\begin{aligned} \sum_{(E:\mathrm{Spec})} E &\simeq \sum_{((N,F):\sum_{(N:\natural\mathrm{Spec})} \mathrm{Spec}_{\underline{N}})} F \\ &\simeq \sum_{(N:\natural\mathrm{Spec})} \sum_{(F:\mathrm{Spec}_{\underline{N}})} F \\ &\simeq \natural\mathrm{Spec} \end{aligned}$$

and we can check that the map $\natural\mathrm{Spec} \to \sum_{(E:\mathrm{Spec})} E$ is given by $N \mapsto (\underline{N}_\natural, \star_{\underline{N}})$. □

The stability axiom implies that suspension and loop-space are automorphisms $\Sigma, \Omega : \natural\mathrm{Spec} \to \natural\mathrm{Spec}$. We can extend these to automorphisms of $\natural\mathcal{U}$ via Proposition 2.2.20, computing Σ and Ω fibrewise.

Definition 2.2.30. Define

$$\begin{aligned} \tilde{\Sigma}\underline{A} &:\equiv \sum_{(x:\natural\underline{A})} \Sigma \underline{A}_{\underline{x}} \\ \tilde{\Omega}\underline{A} &:\equiv \sum_{(x:\natural\underline{A})} \Omega \underline{A}_{\underline{x}} \end{aligned}$$

Our decomposition of the universe can be equivalently described using this $\tilde{\Sigma}$ operation:

Lemma {S} 2.2.31.

$$\mathcal{U} \simeq \sum_{(A:\natural\mathcal{U})} \prod_{(x:\natural\underline{A})} (\tilde{\Sigma}\underline{A})_{\underline{x}}$$

These fibrewise operations do *not* extend to automorphisms of $\mathcal{U}$, but we do have the following:

Proposition {S} 2.2.32. *The fibrewise shifts extend to equivalences $\tilde{\Sigma} : \mathcal{U} \simeq \tilde{\Omega}\mathcal{U}$ and $\tilde{\Omega} : \mathcal{U} \simeq \tilde{\Sigma}\mathcal{U}$*

Proof. Using Theorem 2.2.28, we can calculate $\tilde{\Omega}\mathcal{U}$ to be

$$\begin{aligned} \tilde{\Omega}\mathcal{U} &\equiv \sum_{(A:\natural\mathcal{U})} \Omega \mathcal{U}_{\underline{A}} \\ &\simeq \sum_{(A:\natural\mathcal{U})} \Omega \prod_{(x:\natural\underline{A})} \Sigma \underline{A}_{\underline{x}} \\ &\simeq \sum_{(A:\natural\mathcal{U})} \prod_{(x:\natural\underline{A})} \Omega\Sigma \underline{A}_{\underline{x}} \\ &\simeq \sum_{(A:\natural\mathcal{U})} \prod_{(x:\natural\underline{A})} \underline{A}_{\underline{x}} \end{aligned}$$

And now we can define an equivalence

$$\left(\Sigma_{(A:\natural\mathcal{U})}\Pi_{(x:\natural\underline{A})}(\tilde{\Sigma}\underline{A})_{\underline{x}}\right) \to \left(\Sigma_{(A:\natural\mathcal{U})}\Pi_{(x:\natural\underline{A})}\underline{A}_{\underline{x}}\right)$$

by

$$(A, H) \mapsto (\tilde{\Sigma}\underline{A}, H)$$

Pre- and post-composing the above equivalence with the equivalences to $\mathcal{U}$ and $\tilde{\Omega}\mathcal{U}$ gives the result. □

So $\mathcal{U}$ is also equivalent to

$$\Sigma_{(A:\natural\mathcal{U})}\Pi_{(x:\natural\underline{A})}(\tilde{\Sigma}^n\underline{A})_{\underline{x}}$$

for any $n \in \mathbb{Z}$, by repeatedly applying the above equivalence or its inverse. The choice $n = 1$ used in Theorem 2.2.28 is distinguished from the others however: only when $n = 1$ is the map from $\mathcal{U}$ given by the 'obvious thing' $A \mapsto \underline{A}^\natural$ in the first component.

Turning to function types, we can show that functions between dull types contain less information than one might expect.

Proposition {S} 2.2.33. *For dull spectra $\underline{E}$ and $\underline{F}$,*

$$(\underline{E} \to \underline{F}) \simeq \underline{F} \times \natural(\underline{E} \to \underline{F})$$

Proof. We have the following Yoneda-style argument. For any dull type $\underline{X}$,

$$\begin{aligned}
&\natural(\underline{X} \to (\underline{E} \to \underline{F})) \\
&\simeq \natural\left(\left(\Sigma_{(x:\natural\underline{X})}\underline{X}_{\underline{x}}\right) \to (\underline{E} \to \underline{F})\right) && \text{(Corollary 2.1.6)} \\
&\simeq \natural\Pi_{(x:\natural\underline{X})}\underline{X}_{\underline{x}} \to (\underline{E} \to \underline{F}) && \text{(Currying)} \\
&\simeq \natural\Pi_{(x:\natural\underline{X})}\underline{X}_{\underline{x}} \times \underline{E} \to \underline{F} && \text{(Uncurrying)} \\
&\simeq \natural\Pi_{(x:\natural\underline{X})}\underline{X}_{\underline{x}} \vee \underline{E} \to \underline{F} && \text{(Axiom S)} \\
&\simeq \natural\Pi_{(x:\natural\underline{X})}(\underline{X}_{\underline{x}} \to \underline{F}) \times (\underline{E} \to \underline{F}) && \text{(Proposition 2.2.2)} \\
&\simeq \natural\left(\Pi_{(x:\natural\underline{X})}\underline{X}_{\underline{x}} \to \underline{F}\right) \times \natural\left(\Pi_{(x:\natural\underline{X})}(\underline{E} \to \underline{F})\right) \\
&\simeq \natural(\underline{X} \to \underline{F}) \times \natural(\natural\underline{X} \to (\underline{E} \to \underline{F})) \\
&\simeq \natural(\underline{X} \to \underline{F}) \times \natural(\underline{X} \to \natural(\underline{E} \to \underline{F})) && \text{(Corollary 1.1.26)} \\
&\simeq \natural(\underline{X} \to (\underline{F} \times \natural(\underline{E} \to \underline{F})))
\end{aligned}$$

Now applying this equivalence to the dull map $\mathsf{id}_{(\underline{E} \to \underline{F})}$ yields the equivalence we were looking for. □

This can be extended to functions between any dull types without too much trouble, by decomposing the function type until the previous fact can be applied.

Theorem {S} 2.2.34. *For any dull types $\underline{A}$ and $\underline{B}$*

$$(\underline{A} \to \underline{B}) \simeq \left(\sum_{(f:\natural(\underline{A}\to\underline{B}))}\prod_{(x:\natural\underline{A})}\underline{B}_{\underline{f}_\natural(\underline{x}_\natural)}\right)$$

Proof. Decomposing both $\underline{A}$ and $\underline{B}$ using Corollary 2.1.6 and rearranging:

$$\begin{aligned}
\underline{A} \to \underline{B} &\simeq \left(\sum_{(x:\natural\underline{A})}\underline{A}_{\underline{x}}\right) \to \left(\sum_{(y:\natural\underline{B})}\underline{B}_{\underline{y}}\right) \\
&\simeq \sum_{(h:\natural\underline{A}\to\natural\underline{B})}\prod_{(x:\natural\underline{A})}\underline{A}_{\underline{x}} \to \underline{B}_{\underline{h}(\underline{x})} \\
&\simeq \sum_{(h:\natural\underline{A}\to\natural\underline{B})}\prod_{(x:\natural\underline{A})}\natural(\underline{A}_{\underline{x}} \to \underline{B}_{\underline{h}(\underline{x})}) \times \underline{B}_{\underline{h}(\underline{x})} \\
&\simeq \sum_{(h:\natural\underline{A}\to\natural\underline{B})}\left(\prod_{(x:\natural\underline{A})}\natural(\underline{A}_{\underline{x}} \to \underline{B}_{\underline{h}(\underline{x})})\right) \times \left(\prod_{(x:\natural\underline{A})}\underline{B}_{\underline{h}(\underline{x})}\right)
\end{aligned}$$

And then

$$\sum_{(h:\natural\underline{A}\to\natural\underline{B})}\prod_{(x:\natural\underline{A})}\natural(\underline{A}_{\underline{x}} \to \underline{B}_{\underline{h}(\underline{x})}) \simeq \natural(\underline{A} \to \underline{B})$$

by Proposition 2.1.16. □

2.3 Relating Synthetic and Analytic Spectra

In the previous section, we showed that Axiom S gives synthetic spectra many of the properties that we expect spectra to have. In this section, we investigate an additional axiom, which relates the synthetic spectra to the concrete/analytic spectra that can be defined in pure homotopy type theory. This allows results proved using the synthetic spectra to be transferred to analytic spectra, and vice versa.

One can define ("Ω-")spectra internally in type theory [van18; Cav15] as sequences of types and connecting maps.

Definition 2.3.1. A *sequential prespectrum* $\underline{J}$ is a sequence of pointed modal types $\underline{J} : \mathbb{N} \to \mathrm{Space}_\star$ together with pointed maps $\underline{\alpha}_n : \underline{J}_n \to_\star \Omega\underline{J}_{n+1}$. A *sequential spectrum* is a prespectrum such that the $\underline{\alpha}_n$ are pointed equivalences. The types of such objects are denoted SeqPreSpec and SeqSpec respectively.

Definition 2.3.2. A morphism of sequential (pre)spectra $\underline{f} : \mathrm{Mor}(\underline{I}, \underline{J})$ is a sequence of pointed maps $\underline{f}_n : \underline{I}_n \to_\star \underline{J}_n$ that commute with the $\underline{\alpha}_n$.

Remark 2.3.3. We need to restrict the types in the sequence to be modal so that semantically they correspond to sequences of spaces. Otherwise we would be describing a spectrum object in $P\mathrm{Spec}_\star$, an object more complicated than an ordinary spectrum.

Our goal is to relate our spectra with these sequential spectra. We do this by describing a series of (dull) adjoints:

$$\begin{tikzcd}
\mathrm{Space}_\star \arrow[r, bend left, "\mathrm{susp}"] \arrow[r, phantom, "\perp"] & \mathrm{SeqPreSpec} \arrow[l, bend left, "0th"] \arrow[r, bend left, "\mathrm{spec}"] \arrow[r, phantom, "\perp"] & \mathrm{SeqSpec} \arrow[l, bend left, "\iota"] \arrow[r, bend left, "L"] \arrow[r, phantom, "\perp"] & \mathrm{Spec} \arrow[l, bend left, "R"]
\end{tikzcd}$$

The left two adjunctions take place almost entirely in pure homotopy type theory: susp takes a space $\underline{X}$ to the suspension prespectrum $(\underline{X}, \Sigma\underline{X}, \Sigma\Sigma\underline{X}, \dots)$, and spec is spectrification, inverting the connecting maps to equivalences. Their composite spec $\circ$ susp is thus an analytic analogue of the Σ^∞ stabilisation functor we defined in Section 2.1, taking a modal type to its suspension analytic spectrum. The right adjoint ι is forgetful/an inclusion, while $0th$ selects the $0th$ term of a pre-spectrum, and the composite $0th \circ \iota$ is an analytic analogue of Ω^∞. All that takes us out of ordinary HoTT is the requirement that the types involved are modal types. The new construction in this section is the rightmost adjunction relating analytic and synthetic spectra, and an axiom stating that it is an adjoint equivalence, making the two notions of spectra coincide.

Definition 2.3.4. For any pointed modal type $\underline{X}$, we have the *suspension sequential prespectrum* susp$X$ where $(\text{susp}\underline{X})_n :\equiv \Sigma^n\underline{X}$, and the structure maps are the unit maps $\alpha_n : \Sigma^n\underline{X} \to \Omega\Sigma^{n+1}\underline{X}$.

Note that all the $\Sigma^n\underline{X}$ are modal, by Proposition 1.1.29.

Proposition 2.3.5. susp *is left adjoint to taking the zeroth type of a sequential prespectrum.*

Proof. Suppose we have a map $\underline{X} \to_\star \underline{J}_0$. We need a map $\Sigma\underline{X} \to_\star \underline{J}_1$ such that

$$\begin{tikzcd}
\underline{X} \arrow[r] \arrow[d] & \Omega\Sigma\underline{X} \arrow[d] \\
\underline{J}_0 \arrow[r] & \Omega\underline{J}_1
\end{tikzcd}$$

commutes, equivalently, one such that

$$\begin{tikzcd}
\Sigma\underline{X} \arrow[r, equal] \arrow[d] & \Sigma\underline{X} \arrow[d] \\
\Sigma\underline{J}_0 \arrow[r] & \underline{J}_1
\end{tikzcd}$$

commutes. Then $\Sigma\underline{X} \to_\star \underline{J}_1$ is forced to be the composite of $\Sigma\underline{X} \to \Sigma\underline{J}_0 \to \underline{J}_1$. This argument iterates to produce a map Mor(susp$\underline{X}$, $\underline{J}$), showing the data of such a morphism is equivalent to that of a map $\underline{X} \to_\star \underline{J}_0$. □

Definition 2.3.6. For a sequential prespectrum $\underline{J}$, the *spectrification* of $\underline{J}$ is given by

$$(\text{spec}\underline{J})_n :\equiv \text{colim}_k\, \Omega^k \underline{J}_{n+k}$$

Each $(\text{spec}\underline{J})_n$ is modal by Proposition 1.1.30 and Lemma 1.1.17. That these types actually assemble into a sequential spectrum has not yet been proven in type theory, so we leave it as an unjustified assertion — note that this assertion is a statement in pure homotopy type theory, and is not dependent on the modal extension we use here, a proof of the assertion would certainly apply when the types concerned happen to be modal.

Assertion 1. *This formula defines a sequential spectrum, and the operation is left adjoint to the inclusion of sequential spectra into sequential prespectra.*

Now we turn to the $L \dashv R$ adjunction relating these sequential spectra to our synthetic spectra. First, we can extract a sequential prespectrum from any spectrum $\underline{E}$.

Definition 2.3.7. For $\underline{E}$: Spec, define $R\underline{E}$: SeqPreSpec by

$$(R\underline{E})_n :\equiv \Omega^\infty \Sigma^n \underline{E}$$

with connecting maps

$$(R\underline{E})_n \equiv \Omega^\infty \Sigma^n \underline{E} \to_\star \Omega^\infty \Omega\Sigma\Sigma^n \underline{E} \simeq_\star \Omega\Omega^\infty \Sigma\Sigma^n \underline{E} \simeq_\star \Omega\Omega^\infty \Sigma^{n+1} \underline{E} \equiv \Omega(R\underline{E})_{n+1}$$

The first map is given by functoriality of Ω^∞ on the unit $X \to_\star \Omega\Sigma X$ for $X = \Sigma^n \underline{E}$. The second is derived from Proposition 2.1.34 and the third is essentially by definition, depending on how iterated suspension is defined.

Proposition {S} 2.3.8. *$R\underline{E}$ is a sequential spectrum.*

Proof. By Proposition 2.1.12, spectra are closed under suspensions, so an induction shows that $\Sigma^n \underline{E}$ is a spectrum. Therefore the unit map $\Sigma^n \underline{E} \to_\star \Omega\Sigma\Sigma^n \underline{E}$ is an equivalence by Corollary 2.2.19, so the connecting map defined above is a composite of equivalences. □

Conversely, suppose we have a sequential prespectrum J : SeqPreSpec. We can produce a spectrum LJ : Spec.

Definition 2.3.9. For $\underline{J}$: SeqPreSpec let $L\underline{J}$: Spec be

$$L\underline{J} :\equiv \operatorname{colim}(\Sigma^\infty \underline{J}_0 \to \Omega\Sigma^\infty \underline{J}_1 \to \Omega^2 \Sigma^\infty \underline{J}_2 \to \dots)$$

where the maps $\Sigma^\infty \underline{J}_n \to \Omega\Sigma^\infty \underline{J}_{n+1}$ are given by

$$\Sigma^\infty \underline{J}_n \to \Omega\Sigma\Sigma^\infty \underline{J}_n \simeq \Omega\Sigma^\infty \Sigma \underline{J}_n \to \Omega\Sigma^\infty \Sigma\Omega \underline{J}_{n+1} \to \Omega\Sigma^\infty \underline{J}_{n+1}$$

where the maps are the unit of $\Sigma \vdash \Omega$, Proposition 2.1.36, functoriality on the connecting map $\underline{J}_n \to \Omega \underline{J}_{n+1}$ of the prespectrum, and then the counit of $\Sigma \vdash \Omega$.

To see that this type is a spectrum, by Proposition 1.1.30, $\natural L\underline{J}$ is equivalent to the colimit of the $\natural\Omega^n \Sigma^\infty \underline{J}_i$, and spectra are closed under loop spaces by Proposition 2.1.12, and Σ^∞ of any type is a spectrum by Proposition 2.1.30, so each of the terms of that colimit is contractible, so the colimit is as well.

Remark 2.3.10. Let S : SeqPreSpec denote the sphere (analytic) prespectrum, i.e. the suspension prespectrum of S^0. Then $LS \simeq \mathbb{S}$, because at each level of the colimit we have

$$\Omega^n \Sigma^\infty S^n \equiv \Omega^n (S^n \wedge \mathbb{S}) \simeq \Omega^n (\Sigma^n \mathbb{S}) \simeq \mathbb{S}$$

The first equivalence is $\Sigma^n A \simeq S^1 \wedge (S^1 \wedge \dots (S^1 \wedge A)) \simeq (S^1 \wedge S^1 \dots \wedge S^1) \wedge A \simeq S^n \wedge A$, using associativity of smash and $\Sigma A \simeq S^1 \wedge A$ [Bru16, Proposition 4.2.1], [van18, Definition 4.3.33]. The second is given by iterating $\Omega\Sigma\underline{E} \simeq \underline{E}$ for a dull spectrum $\underline{E}$ (Corollary 2.2.19), noting that the suspension of a dull spectrum is a spectrum by Proposition 2.1.12, and $\mathbb{S}$ is itself a spectrum.

Proposition {S} 2.3.11. *The operations L and R are dull adjoints:*

$$\natural(L\underline{J} \to_\star \underline{E}) \simeq \mathrm{Mor}(\underline{J}, R\underline{E})$$

Proof. Suppose we have a dull map $\underline{k} : L\underline{J} \to_\star \underline{E}$. We can forget the fact that $\underline{k}$ is pointed using Proposition 2.1.10. The data of $\underline{k}$ (using the universal property for maps out of a colimit) is equivalent to a sequence of dull maps $\underline{k}_n : \Omega^n\Sigma^\infty\underline{J}_n \to \underline{E}$ so that the squares

$$\begin{tikzcd}
\Omega^n\Sigma^\infty\underline{J}_n \arrow[r] \arrow[d, "\underline{k}_n"'] & \Omega^{n+1}\Sigma^\infty\underline{J}_{n+1} \arrow[d, "\underline{k}_{n+1}"] \\
\underline{E} \arrow[r, equal] & \underline{E}
\end{tikzcd}$$

commute.

We can transpose the $\underline{k}$ across the adjunctions to get maps $\hat{k}_n : \underline{J}_n \to_\star \Omega^\infty\Sigma^n\underline{E}$. This type on the right is exactly $(R\underline{E})_n$, so we just have to show that this collection of maps forms a morphism of sequential prespectra. This mostly involves unwinding the definition of the map $\Omega^n\Sigma^\infty\underline{J}_n \to \Omega^{n+1}\Sigma^\infty\underline{J}_{n+1}$

Precompose the upper left corner with the equivalence

$$\Omega^{n+1}\Sigma^\infty\Sigma\underline{J}_n \simeq \Omega^n\Omega\Sigma^\infty\Sigma\underline{J}_n \simeq \Omega^n\Omega\Sigma\Sigma^\infty\underline{J}_n \simeq \Omega^n\Sigma^\infty\underline{J}_n$$

to yield

$$\begin{tikzcd}
\Omega^{n+1}\Sigma^\infty\Sigma\underline{J}_n \arrow[r, "\Omega^{n+1}\Sigma^\infty\hat{\alpha}_n"] \arrow[d, "\cdots"'] & \Omega^{n+1}\Sigma^\infty\underline{J}_{n+1} \arrow[d, "\underline{k}_{n+1}"] \\
\underline{E} \arrow[r, equal] & \underline{E}
\end{tikzcd}$$

Since $\Sigma \dashv \Omega$, and for dull spectra Σ and Ω are inverses, we also have an adjunction $\Omega \dashv \Sigma$ on dull spectra. Transposing across this and $\Sigma^\infty \dashv \Omega^\infty$ vertically, such squares are equivalent to squares

$$\begin{tikzcd}
\Sigma\underline{J}_n \arrow[r, "\hat{\alpha}_n"] \arrow[d, "\cdots"'] & \underline{J}_{n+1} \arrow[d, "\hat{k}_{n+1}"] \\
(R\underline{E})_{n+1} \arrow[r, equal] & (R\underline{E})_{n+1}
\end{tikzcd}$$

Now transposing 'diagonally' along $\Sigma \dashv \Omega$, we get

$$\begin{tikzcd}
\underline{J}_n \arrow[r, "\alpha_n"] \arrow[d, "\cdots"'] & \Omega\underline{J}_{n+1} \arrow[d, "\Omega\hat{\underline{k}}_{n+1}"] \\
\Omega(R\underline{E})_{n+1} \arrow[r, equal] & \Omega(R\underline{E})_{n+1}
\end{tikzcd}$$

Finally, precompose the lower left corner with the equivalence $\beta_n : (R\underline{E})_n \simeq \Omega(R\underline{E})_{n+1}$ to get

$$\begin{tikzcd}
\underline{J}_n \arrow[r, "\alpha_n"] \arrow[d, "\cdots"'] & \Omega\underline{J}_{n+1} \arrow[d, "\Omega\hat{\underline{k}}_{n+1}"] \\
(R\underline{E})_n \arrow[r, "\beta_n"'] & \Omega(R\underline{E})_{n+1}
\end{tikzcd}$$

All we have left is to check is that the vertical map on the left is equal to $\hat{k}_n$. Tracing through what we have done, it is equal to

$$\begin{aligned}
\underline{J}_n &\to \Omega\Sigma\underline{J}_n \to \Omega\Omega^\infty\Sigma^{n+1}\Omega^{n+1}\Sigma^\infty\Sigma\underline{J}_n \to \Omega\Omega^\infty\Sigma^{n+1}\Omega^{n+1}\Sigma\Sigma^\infty\underline{J}_n \\
&\to \Omega\Omega^\infty\Sigma^{n+1}\Omega^n\Sigma^\infty\underline{J}_n \to \Omega\Omega^\infty\Sigma^{n+1}\underline{E} \to \Omega^\infty\Omega\Sigma^{n+1}\underline{E} \to \Omega^\infty\Sigma^n\underline{E}
\end{aligned}$$

In the above sequence, the map $\Omega\Omega^\infty\Sigma^{n+1}\Omega^n\Sigma^\infty\underline{J}_n \to \Omega\Omega^\infty\Sigma^{n+1}\underline{E}$ is given by $\Omega\Omega^\infty\Sigma^{n+1}\underline{k}_n$, so by naturality, we can move this use of $\underline{k}_n$ to the end. The above chain is then equal to the composite

$$\begin{aligned}
\underline{J}_n &\to \Omega\Sigma\underline{J}_n \to \Omega\Omega^\infty\Sigma^{n+1}\Omega^{n+1}\Sigma^\infty\Sigma\underline{J}_n \to \Omega\Omega^\infty\Sigma^{n+1}\Omega^{n+1}\Sigma\Sigma^\infty\underline{J}_n \\
&\to \Omega\Omega^\infty\Sigma^{n+1}\Omega^n\Sigma^\infty\underline{J}_n \to \Omega^\infty\Omega\Sigma^{n+1}\Omega^n\Sigma^\infty\underline{J}_n \to \Omega^\infty\Sigma^n\Omega^n\Sigma^\infty\underline{J}_n \to \Omega^\infty\Sigma^n\underline{E}
\end{aligned}$$

For this to be the transpose of $\underline{k}_n$, what we need is for the composite $\underline{J}_n \to \Omega^\infty\Sigma^n\Omega^n\Sigma^\infty\underline{J}_n$, leaving off the last map, is equal to the unit. And it is, by liberal use of the triangle inequalities, as every map in the string is either unit/counit or an equality of composites in a system of commuting adjunctions.

□

Having established this adjunction, we introduce the following axiom, which identifies the synthetic and analytic spectra:

Axiom N. *The adjunction between* Spec *and* SeqSpec *is a dull adjoint equivalence, i.e. the map* $LR\underline{E} \to_\star \underline{E}$ *is an equivalence and* $\mathrm{Mor}(\underline{J}, RL\underline{J})$ *is a level-wise equivalence.*

As an application, we show that this axiom fixes the stable homotopy groups of $\mathbb{S}$, in the sense of Definition 2.1.28, to be the actual stable homotopy groups of the ordinary spheres.

Remark 2.3.12. The composite right adjoint Spec $\to$ SeqSpec $\to$ SeqPreSpec $\to$ $\mathrm{Space}_\star$ in the diagram at the beginning of this section is $(RE)_0 \equiv \Omega^\infty E$. So the composite adjunction must be equal to the adjunction

$$\begin{tikzcd}
\mathrm{Space}_\star \arrow[r, bend left, "\Sigma^\infty"] & \mathrm{Spec} \arrow[l, bend left, "\Omega^\infty"]
\end{tikzcd}$$

that we already have from Proposition 2.1.31.

Definition 2.3.13. Let $Q\underline{X} :\equiv \Omega^\infty\Sigma^\infty\underline{X}$.

Lemma {S, N} 2.3.14.

$$Q\underline{X} \simeq \operatorname{colim}_k \Omega^k \Sigma^k \underline{X}$$

Proof. By Remark 2.3.12, we have $Q\underline{X} \equiv \Omega^\infty \Sigma^\infty \underline{X} \simeq 0\text{th}(\iota(R(L(\text{spec}(\text{susp}(\underline{X}))))))$. Axiom N allows us to chop the $L \dashv R$ adjunction off this roundtrip on $\text{Space}_\star$, and ι is just forgetful, so we get $0\text{th}(\text{spec}(\text{susp}(\underline{X})))$. Then $\operatorname{colim}_k \Omega^k \Sigma^k \underline{X}$ is exactly the 0th type of the spectrification of the suspension prespectrum of $\underline{X}$. □

Proposition {S, N} 2.3.15.

$$\pi_n^s(\mathbb{S}) \simeq \operatorname{colim}_k \pi_{n+k}(S^k)$$

Proof. We will make use of some properties of sequential colimits proven in [SDR20]. Specifically, sequential colimits commute with taking loop spaces [SDR20, Corollary 7.4] and truncations [SDR20, Corollary 7.6], and thus calculating homotopy groups.

$$\begin{aligned}
\pi_n^s(\mathbb{S}) &\equiv \pi_n(\Omega^\infty \mathbb{S}) && \text{(by definition)}\\
&\simeq \pi_n(\Omega^\infty \Sigma^\infty S^0) && \text{(by Remark 2.3.10)}\\
&\simeq \pi_n(\operatorname{colim}_k \Omega^k \Sigma^k S^0) && \text{(by the Lemma 2.3.14)}\\
&\simeq \operatorname{colim}_k \pi_n(\Omega^k \Sigma^k S^0) && \text{(sequential colimit commutes with } \pi_n)\\
&\simeq \operatorname{colim}_k \pi_{n+k}(\Sigma^k S^0) && \text{(definition of } \pi)\\
&\simeq \operatorname{colim}_k \pi_{n+k}(S^k) && \text{(definition of } S^n)
\end{aligned}$$

□

2.4 An Internal Six Functor Formalism

We show that any function $\underline{f} : \underline{X} \to \underline{Y}$ between spaces induces a 'Wirthmüller context' between the types over each base. This is a special case of Grothendieck's six functor formalism, surveyed in [FHM03]. We will see that for any space $\underline{X}$ there is a closed monoidal category $\text{Fam}(\underline{X})$ of types over that base, and that any function $\underline{f} : \underline{X} \to \underline{Y}$ induces an adjoint triple

$$\begin{tikzcd}
\text{Fam}(\underline{X}) \arrow[r, bend left=40, "f_!"] \arrow[r, bend right=40, "f_*"'] & \text{Fam}(\underline{Y}) \arrow[l, "f^*" description]
\end{tikzcd}$$

such that f^* is a monoidal closed functor.

Because we have such a context for *every* function of spaces, we expect the map $\natural(-) : \mathcal{U} \to \text{Space}$ to define an internal, weak, 'closed monoidal $*$-bifibration', in the sense of [Shu08, §3, Definitions 12.1 and 13.3]. In [Sch14, §3], there is speculation on a notion of 'linear homotopy-type theory' which would have semantics in such bifibrations, with the adjoint operations given by specialised type constructors. Our theory is more general: we can construct these adjoints internally using the type formers so far described.

There are two equivalent ways to define the categories involved. For a space $\underline{X}$, our preferred way is to use the category with type of objects given by $\mathrm{Fam}(\underline{X}) :\equiv \underline{X} \to \mathrm{Spec}$. This behaves quite nicely type-theoretically, in particular, the 'pullback' functors will compose strictly: $g^* \circ f^* \equiv (g \circ f)^*$.

The other way to define the spectra over $\underline{X}$ is to think of every type A as living intrinsically over the base $\natural\underline{A}$, and defining $\mathrm{Fib}_\natural(\underline{X})$ to be all types equipped with an equivalence between their intrinsic base and $\underline{X}$. Some operations will be more natural to construct using one description than the other. We will define both versions and show that they are equivalent.

We write the top colour as purple throughout this section.

2.4.1 Families of Spectra

Definition 2.4.1. For $\underline{X}$ a space, define $\mathrm{Fam}(\underline{X}) :\equiv \underline{X} \to \mathrm{Spec}$.

Definition 2.4.2. For $A : \mathrm{Fam}(\underline{X})$ and $B : \mathrm{Fam}(\underline{Y})$, a *map over* $\underline{f} : \underline{X} \to \underline{Y}$ is a map $h(\underline{x}) : A(\underline{x}) \to B(\underline{f}(\underline{x}))$ for every $\underline{x} : \underline{X}$. Let $\mathrm{Map}_{\underline{f}}(A, B)$ denote the type of such maps:

$$\mathrm{Map}_{\underline{f}}(A, B) :\equiv \prod_{(\underline{x}:\underline{X})} A(\underline{x}) \to B(\underline{f}(\underline{x}))$$

and write $\mathrm{Map}_{\underline{X}}(A, B)$ as a shorthand for $\mathrm{Map}_{\mathrm{id}_{\underline{X}}}(A, B)$.

Definition 2.4.3. For $m : \mathrm{Map}_{\underline{f}}(A, B)$ and $n : \mathrm{Map}_{\underline{g}}(B, C)$, there is $n \circ m : \mathrm{Map}_{\underline{g}\circ\underline{f}}(A, C)$ defined by

$$(n \circ m)(\underline{x}) :\equiv n(\underline{f}(\underline{x})) \circ m(\underline{x})$$

2.4.2 Pullback and Pushforwards

Definition 2.4.4. For a family $B : \mathrm{Fam}(\underline{Y})$, the family $\underline{f}^*B : \mathrm{Fam}(\underline{X})$ is given by composition:

$$\underline{f}^*B :\equiv B \circ \underline{f}$$

Proposition 2.4.5.

$$\mathrm{Map}_{\underline{f}}(A, B) \equiv \mathrm{Map}_{\underline{X}}(A, \underline{f}^*B)$$

Proof. Immediate from the definitions:

$$\prod_{(\underline{x}:\underline{X})} A(\underline{x}) \to B(\underline{f}(\underline{x})) \equiv \prod_{(\underline{x}:\underline{X})} A(\underline{x}) \to (\underline{f}^*B)(\underline{x})$$

□

Proposition 2.4.6. *Pullback is functorial definitionally, in that $f^*g^*B \equiv (g \circ f)^*B$*

Proof. Immediate by associativity of function composition. □

The functor $\underline{f}^*$ has both left and right adjoint. First the right adjoint:

Definition 2.4.7. For $A : \mathrm{Fam}(\underline{X})$, define $\underline{f}_* A : \mathrm{Fam}(\underline{Y})$ by

$$\underline{f}_* A(\underline{y}) :\equiv \prod_{((x,p):\mathsf{fib}_{\underline{f}}(\underline{y}))} A(\underline{x})$$

Lemma 2.4.8. *This defines a family of spectra on $\underline{Y}$.*

Proof. $\mathsf{fib}_{\underline{f}}(\underline{y})$ is a space, so by Theorem 1.1.24,

$$\begin{aligned}\natural\left(\prod_{((x,p):\mathsf{fib}_{\underline{f}}(\underline{y}))} \underline{A}(\underline{x})\right) &\simeq \prod_{((x,p):\mathsf{fib}_{\underline{f}}(\underline{y}))} \natural \underline{A}(\underline{x}) \\ &\simeq \prod_{((x,p):\mathsf{fib}_{\underline{f}}(\underline{y}))} 1 \\ &\simeq 1\end{aligned}$$

□

Remark 2.4.9. If $\pi : \underline{X} \to 1$ denotes the unique map, then for any $A : \mathrm{Fib}_\natural(\underline{X})$, we can compute

$$(\pi_* A)(\star) \equiv \prod_{((x,p):\mathsf{fib}_\pi(\star))} A(\underline{x}) \simeq \prod_{(x:\underline{X})} A(\underline{x})$$

This is the 'spectrum of sections' of A, the twisted cohomology of the space $\underline{X}$ with coefficients in A.

Theorem 2.4.10. *There is an adjunction*

$$\mathrm{Map}_{\underline{X}}(\underline{f}^* B, A) \simeq \mathrm{Map}_{\underline{Y}}(B, \underline{f}_* A)$$

Proof.

$$\begin{aligned}\mathrm{Map}_{\underline{X}}(\underline{f}^* B, A) &\equiv \prod_{(x:\underline{X})} B(\underline{f}(\underline{x})) \to A(\underline{x}) \\ &\simeq \prod_{((y,(x,p)):\sum_{(y:\underline{Y})} \mathsf{fib}_{\underline{f}}(\underline{y}))} B(\underline{y}) \to A(\underline{x}) && \text{([HoTTBook, Lemma 4.8.2])} \\ &\simeq \prod_{(y:\underline{Y})} \prod_{((x,p):\mathsf{fib}_{\underline{f}}(\underline{y}))} B(\underline{y}) \to A(\underline{x}) && \text{(Currying)} \\ &\simeq \prod_{(y:\underline{Y})} \left(B(\underline{y}) \to \prod_{((x,p):\mathsf{fib}_{\underline{f}}(\underline{y}))} A(\underline{x})\right) && \text{(Symmetry)} \\ &\equiv \mathrm{Map}_{\underline{Y}}(B, \underline{f}_* A)\end{aligned}$$

□

Now the left-adjoint to $\underline{f}^*$. The reader might expect a definition like

$$\underline{f}_! A(\underline{y}) :\overset{?}{\equiv} \sum_{((x,p):\mathsf{fib}_{\underline{f}}(\underline{y}))} A(\underline{x})$$

but the right side is not a spectrum unless $\mathsf{fib}_{\underline{f}}(\underline{y})$ happens to be contractible. We have to make a correction:

Definition 2.4.11. For $\underline{A} : \mathrm{Fam}(\underline{X})$, define

$$\begin{aligned} \mathsf{zeros}_{\underline{A}}(\underline{y}) &: \mathsf{fib}_{\underline{f}}(\underline{y}) \to \textstyle\sum_{((x,p):\mathsf{fib}_{\underline{f}}(\underline{y}))} \underline{A}(\underline{x}) \\ \mathsf{zeros}_{\underline{A}}(\underline{y})(x,p) &:\equiv (x, p, \star_{\underline{A}(\underline{x})}) \end{aligned}$$

And then for each $\underline{y} : \underline{Y}$, define $\underline{f}_!\underline{A}(\underline{y})$ as the cofibre

$$\begin{tikzcd} \mathsf{fib}_{\underline{f}}(\underline{y}) \arrow[r, "\mathsf{zeros}_{\underline{A}}(\underline{y})"] \arrow[d] & \sum_{((x,p):\mathsf{fib}_{\underline{f}}(\underline{y}))} \underline{A}(\underline{x}) \arrow[d] \\ 1 \arrow[r] \arrow[ur, phantom, "\ulcorner", very near end] & \underline{f}_!\underline{A}(\underline{y}) \end{tikzcd}$$

This definition can only be made for dull $\underline{A}$, so that we have access to a canonical point $\star_{\underline{A}(\underline{x})} : \underline{A}(\underline{x})$ for each $\underline{x}$.

Lemma 2.4.12. *This pushout defines a spectrum for each $\underline{y}$.*

Proof. $\natural$ commutes with pushouts (Proposition 1.1.29) and Σ-types (Proposition 1.1.18). The map $\mathsf{zeros}_{\underline{A}}(\underline{y})$ becomes an equivalence under the action of $\natural$ because $\underline{A}(\underline{x})$ is a spectrum, and then the pushout of an equivalence is an equivalence giving $1 \simeq \natural \underline{f}_!\underline{A}(\underline{y})$. □

Remark 2.4.13. Note that if $\underline{X}$ is a space and $\pi : \underline{X} \to 1$ is the unique map, then $\pi_!\pi^\star(\mathbb{S}) \simeq \Sigma^\infty_+\underline{X}$.

Proposition 2.4.14. $\natural\mathrm{Map}_{\underline{Y}}(\underline{f}_!\underline{A}, \underline{B}) \simeq \natural\mathrm{Map}_{\underline{f}}(\underline{A}, \underline{B})$

Proof. Fix $\underline{y} : \underline{Y}$. Then by the universal property of pushouts, we have that functions $\underline{f}_!\underline{A}(\underline{y}) \to \underline{B}(\underline{y})$ are equivalent to triples

$$\begin{aligned} i &: \left(\textstyle\sum_{((x,p):\mathsf{fib}_{\underline{f}}(\underline{y}))} \underline{A}(\underline{x})\right) \to \underline{B}(\underline{y}) \\ j &: \underline{B}(\underline{y}) \\ H &: \textstyle\prod_{((x,p):\mathsf{fib}_{\underline{f}}(\underline{y}))} i((x,p), \star_{\underline{A}(\underline{x})}) = j \end{aligned}$$

and so dull functions $\natural(\underline{f}_!\underline{A}(\underline{y}) \to \underline{B}(\underline{y}))$ correspond to *dull* triples

$$\begin{aligned} \underline{i} &: \left(\textstyle\sum_{((x,p):\mathsf{fib}_{\underline{f}}(\underline{y}))} \underline{A}(\underline{x})\right) \to \underline{B}(\underline{y}) \\ \underline{j} &: \underline{B}(\underline{y}) \\ \underline{H} &: \textstyle\prod_{((x,p):\mathsf{fib}_{\underline{f}}(\underline{y}))} \underline{i}((\underline{x}, \underline{p}), \star_{\underline{A}(\underline{x})}) = \underline{j} \end{aligned}$$

Now $\underline{j}$ is a dull point of a spectrum, and $\underline{H}$ is a dull family of paths in a spectrum, so in fact $\underline{j}$ and $\underline{H}$ contain no data at all, and

$$\natural(\underline{f}_!\underline{A}(\underline{y}) \to \underline{B}(\underline{y})) \simeq \natural\left(\left(\textstyle\sum_{((x,p):\mathsf{fib}_{\underline{f}}(\underline{y}))} \underline{A}(\underline{x})\right) \to \underline{B}(\underline{y})\right)$$

Then

$$\begin{aligned}
\natural\mathrm{Map}_{\underline{Y}}(\underline{f}_!\underline{A},\underline{B}) &\equiv \natural\textstyle\prod_{(y:\underline{Y})}\left(\underline{f}_!\underline{A}(\underline{y}) \to \underline{B}(\underline{y})\right)\\
&\simeq \textstyle\prod_{(y:\underline{Y})}\natural\left(\underline{f}_!\underline{A}(\underline{y}) \to \underline{B}(\underline{y})\right)\\
&\simeq \textstyle\prod_{(y:\underline{Y})}\natural\left(\left(\sum_{((x,p):\mathsf{fib}_{\underline{f}}(\underline{y}))}\underline{A}(\underline{x})\right) \to \underline{B}(\underline{y})\right)\\
&\simeq \natural\textstyle\prod_{(y:\underline{Y})}\left(\sum_{((x,p):\mathsf{fib}_{\underline{f}}(\underline{y}))}\underline{A}(\underline{x})\right) \to \underline{B}(\underline{y})\\
&\simeq \natural\left(\textstyle\prod_{((y,(x,p)):\sum_{(y:\underline{Y})}\mathsf{fib}_{\underline{f}}(\underline{y}))}\underline{A}(\underline{x}) \to \underline{B}(\underline{y})\right)\\
&\simeq \natural\left(\textstyle\prod_{(x:\underline{X})}\underline{A}(\underline{x}) \to \underline{B}(\underline{f}(\underline{x}))\right)\\
&\equiv \natural\mathrm{Map}_{\underline{f}}(\underline{A},\underline{B})
\end{aligned}$$

□

Corollary 2.4.15. $\natural\mathrm{Map}_{\underline{Y}}(\underline{f}_!\underline{A},\underline{B}) \simeq \natural\mathrm{Map}_{\underline{X}}(\underline{A},\underline{f}^*\underline{B})$

Corollary 2.4.16. Spec *is 'externally' a reflective subcategory of* $\mathcal{U}$*, in that, for any* $\underline{A} : \mathcal{U}$ *and* $\underline{E}$: Spec*,*

$$\natural(\pi_!\underline{A} \to \underline{E}) \simeq \natural(\underline{A} \to \underline{E})$$

where $\pi : \natural\underline{A} \to 1$ *denotes the unique map.*

Proof. $\natural(\pi_!\underline{A} \to \underline{E}) \simeq \natural\mathrm{Map}_1(\pi_!\underline{A},\underline{E}) \simeq \natural\mathrm{Map}_\pi(\underline{A},\underline{E}) \simeq \natural(\underline{A} \to \underline{E})$ □

2.4.3 Tensor and Hom

First, there is an 'internal' tensor product and hom on each Fam($\underline{X}$).

Definition 2.4.17. For $\underline{A}$: Fam($\underline{X}$) and $\underline{A}'$: Fam($\underline{X}$), define $\underline{A} \otimes_{\underline{X}} \underline{A}'$: Fam($\underline{X}$) by applying the actual $\otimes$-type former pointwise:

$$(\underline{A} \otimes_{\underline{X}} \underline{A}')(\underline{x}) :\equiv \underline{A}(\underline{x}) \otimes \underline{A}'(\underline{x})$$

which defines a family of spectra by Proposition 1.3.21. The unit of this operation $\mathbb{S}_X$ is given by $\mathbb{S}_X(x) :\equiv \mathbb{S}$.

The internal hom on Fam($\underline{X}$) is similarly calculated pointwise:

Definition {C} 2.4.18. For $\underline{A}$: Fam($\underline{X}$) and $\underline{A}'$: Fam($\underline{X}$), define

$$(\underline{A} \multimap_{\underline{X}} \underline{A}')(\underline{x}) :\equiv \underline{A}(\underline{x}) \multimap \underline{A}'(\underline{x})$$

which is a family of spectra by Axiom C.

Proposition {C} 2.4.19. *There is an adjunction*

$$\natural\mathrm{Map}_{\underline{X}}(\underline{A} \otimes_{\underline{X}} \underline{A}', \underline{A}'') \simeq \natural\mathrm{Map}_{\underline{X}}(\underline{A}, \underline{A}' \multimap_{\underline{X}} \underline{A}'')$$

Proof. Follows quickly from Proposition 1.5.9. $\square$

Proposition 2.4.20. *The f^* operation is strong closed monoidal with respect to the internal tensor and hom.*

Proof. Because the constructions are done pointwise, they commute definitionally with f^*. $\square$

There is also an 'external' tensor, which is more like how our $\otimes$-type former behaves intrinsically:

Definition 2.4.21. For $\underline{A} : \mathrm{Fam}(\underline{X})$ and $\underline{B} : \mathrm{Fam}(\underline{Y})$, define $\underline{A} \underline{\otimes} \underline{B} : \mathrm{Fam}(\underline{X} \times \underline{Y})$ by

$$(\underline{A} \underline{\otimes} \underline{B})(\underline{x}, \underline{y}) :\equiv \underline{A}(\underline{x}) \otimes \underline{B}(\underline{y})$$

Proposition 2.4.22. *Given maps $\underline{f} : \underline{X} \to \underline{Y}$ and $\underline{g} : \underline{X}' \to \underline{Y}'$ between spaces, and types $\underline{B} : \mathrm{Fam}(\underline{Y})$ and $\underline{B}' : \mathrm{Fam}(\underline{Y}')$ respectively,*

$$(\underline{f} \times \underline{g})^*(\underline{B} \underline{\otimes} \underline{B}') = \underline{f}^*\underline{B} \otimes \underline{g}^*\underline{B}'$$

Proof.

$$\begin{aligned}
(\underline{f} \times \underline{g})^*(\underline{B} \underline{\otimes} \underline{B}')(\underline{x}, \underline{x}') &\equiv (\underline{B} \underline{\otimes} \underline{B}')(\underline{f}(\underline{x}), \underline{g}(\underline{x}')) \\
&\equiv \underline{B}(\underline{f}(\underline{x})) \otimes \underline{B}'(\underline{g}(\underline{x}')) \\
&\equiv \underline{f}^*\underline{B}(\underline{x}) \otimes \underline{g}^*\underline{B}'(\underline{x}') \\
&\equiv (\underline{f}^*\underline{B} \underline{\otimes} \underline{g}^*\underline{B}')(\underline{x}, \underline{x}')
\end{aligned}$$

$\square$

We can also wonder whether this external tensor product has a right adjoint, an operation with the following shape: for $\underline{B} : \mathrm{Fam}(\underline{Y})$ and $\underline{C} : \mathrm{Fam}(\underline{X} \times \underline{Y})$ there should be $\underline{B} \rhd \underline{C} : \mathrm{Fam}(\underline{X})$. This is also easy to construct:

Definition {C} 2.4.23. For dull $\underline{B} : \mathrm{Fam}(\underline{Y})$ and $\underline{C} : \mathrm{Fam}(\underline{X} \times \underline{Y})$, define $\underline{B} \rhd \underline{C} : \mathrm{Fam}(\underline{X})$ by

$$(\underline{B} \rhd \underline{C})(\underline{x}) :\equiv \textstyle\prod_{(y:\underline{Y})} \underline{B}(\underline{y}) \multimap \underline{C}(\underline{x}, \underline{y})$$

Again, we need to assume Axiom C for each fibre to be a spectrum.

Proposition {C} 2.4.24. *There is a dull adjunction*

$$\natural\mathrm{Map}_{\underline{X} \times \underline{Y}}(\underline{A} \underline{\otimes} \underline{B}, \underline{C}) \simeq \natural\mathrm{Map}_{\underline{X}}(\underline{A}, \underline{B} \rhd \underline{C})$$

Proof.

$$\begin{aligned}
\natural\mathrm{Map}_{\underline{X}\times\underline{Y}}(\underline{A}\otimes\underline{B},\underline{C}) &\equiv \natural\left(\textstyle\prod_{((x,y):\underline{X}\times\underline{Y})}(\underline{A}\otimes\underline{B})(\underline{x},\underline{y})\to\underline{C}(\underline{x},\underline{y})\right)\\
&\simeq \natural\left(\textstyle\prod_{((x,y):\underline{X}\times\underline{Y})}\underline{A}(\underline{x})\otimes\underline{B}(\underline{y})\to\underline{C}(\underline{x},\underline{y})\right)\\
&\simeq \textstyle\prod_{((x,y):\underline{X}\times\underline{Y})}\natural\left(\underline{A}(\underline{x})\otimes\underline{B}(\underline{y})\to\underline{C}(\underline{x},\underline{y})\right) && \text{(Theorem 1.1.24)}\\
&\simeq \textstyle\prod_{((x,y):\underline{X}\times\underline{Y})}\natural\left(\underline{A}(\underline{x})\to(\underline{B}(\underline{y})\multimap\underline{C}(\underline{x},\underline{y}))\right) && \text{(Proposition 1.5.9)}\\
&\simeq \natural\left(\textstyle\prod_{((x,y):\underline{X}\times\underline{Y})}\underline{A}(\underline{x})\to(\underline{B}(\underline{y})\multimap\underline{C}(\underline{x},\underline{y}))\right) && \text{(Theorem 1.1.24 again)}\\
&\simeq \natural\left(\textstyle\prod_{(x:\underline{X})}\underline{A}(\underline{x})\to(\prod_{(y:\underline{Y})}\underline{B}(\underline{y})\multimap\underline{C}(\underline{x},\underline{y}))\right) && \text{(Currying)}\\
&\equiv \natural\left(\textstyle\prod_{(x:\underline{X})}\underline{A}(\underline{x})\to(\underline{B}\rhd\underline{C})(\underline{x})\right)\\
&\simeq \natural\mathrm{Map}_{\underline{X}}(\underline{A},\underline{B}\rhd\underline{C})
\end{aligned}$$

□

Proposition {C} 2.4.25. *The internal and external operations are related by*

$$\begin{gathered}
\underline{A}\otimes_{\underline{X}}\underline{A}' \equiv \Delta_{\underline{X}}{}^*(\underline{A}\otimes\underline{A}')\\
\mathbb{S}_{\underline{X}} \equiv \pi_{\underline{X}}{}^*(\mathbb{S})\\
\underline{A}\multimap_{\underline{X}}\underline{A}' \simeq \underline{A}\rhd(\Delta_{\underline{X}})_*\underline{A}'
\end{gathered}$$

where $\pi_{\underline{X}} : \underline{X}\to 1$ *and* $\Delta_{\underline{X}} : \underline{X}\to\underline{X}\times\underline{X}$.

Proof. The first two follow immediately from expanding definitions, and the third by uniqueness of adjoints, seeing that

$$\begin{aligned}
\natural\mathrm{Map}_{\underline{X}}(\underline{A}\otimes_{\underline{X}}\underline{A}',\underline{A}'') &\equiv \natural\mathrm{Map}_{\underline{X}}(\Delta_{\underline{X}}{}^*(\underline{A}\otimes\underline{A}'),\underline{A}'')\\
&\simeq \natural\mathrm{Map}_{\underline{X}\times\underline{X}}(\underline{A}\otimes\underline{A}',(\Delta_{\underline{X}})_*\underline{A}'')\\
&\simeq \natural\mathrm{Map}_{\underline{X}}(\underline{A},\underline{A}\rhd(\Delta_{\underline{X}})_*\underline{A}'')
\end{aligned}$$

□

2.4.4 Types with Fixed Base

Rather than considering functions $\underline{X}\to\mathrm{Spec}$, instead we can use the fact that every type A is intrinsically a family of spectra over the space $\natural\underline{A}$ via Corollary 2.1.6.

Definition 2.4.26. Let $\underline{X}$ be a dull space. A *type with base* $\underline{X}$ is a type A with an equivalence $v : \natural\underline{A}\simeq\underline{X}$. The type of types with base $\underline{X}$ is written $\mathrm{Fib}_\natural(\underline{X})$.

The type $(\natural\underline{A}\simeq\underline{X})$ is a space because it is the type of equivalence between two spaces, and so we may assume the provided equivalence v is dull.

Any type A can be paired with the identity equivalence $\natural\underline{A}\simeq\natural\underline{A}$ to give $(A,\mathrm{id}_{\natural\underline{A}}) : \mathrm{Fib}_\natural(\natural\underline{A})$.

Definition 2.4.27. For any $(A, \underline{w}) : \mathrm{Fib}_\natural(\underline{X})$ there are functions $\mathsf{base}_A : A \to \underline{X}$ and $\mathsf{zero}_{\underline{A}} : \underline{X} \to \underline{A}$ given by

$$\begin{aligned}\mathsf{base}_A(a) &:\equiv \underline{w}(\underline{a}^\natural)\\ \mathsf{zero}_{\underline{A}}(x) &:\equiv (\underline{w}^{-1}(x))_\natural\end{aligned}$$

Note that the codomain of zero is only $\underline{A}$, not the original A.

Proposition 2.4.28. *There is an equivalence* $\phi : \mathrm{Fam}(\underline{X}) \to \mathrm{Fib}_\natural(\underline{X})$ *via the forward and reverse assignments*

$$\begin{aligned}\phi(E) &:\equiv \left(\textstyle\sum_{(x:\underline{X})} E(x), \lambda((\underline{x}, \underline{e})^\natural).\underline{x}\right)\\ \phi^{-1}(A, \underline{v}) &:\equiv \lambda x.A_{\underline{v}^{-1}(\underline{x})}\end{aligned}$$

Proof. This equivalence was constructed in the process of proving Proposition 2.2.20. □

Definition 2.4.29. Given $(A, \underline{v}) : \mathrm{Fib}_\natural(\underline{X})$ and $(B, \underline{w}) : \mathrm{Fib}_\natural(\underline{Y})$, a *map over* $\underline{f} : \underline{X} \to \underline{Y}$ is a map $h : A \to B$ together with a witness $\underline{H}$ that

$$\begin{tikzcd}\natural\underline{A} \arrow[r, "\natural\underline{h}"] \arrow[d, "\underline{v}"'] & \natural\underline{B} \arrow[d, "\underline{w}"] \\ \underline{X} \arrow[r, "\underline{f}"'] & \underline{Y}\end{tikzcd}$$

commutes, where $\natural\underline{h}$ is defined by Definition 1.1.5. The type of such pairs $(h, \underline{H})$ is denoted $\mathrm{Map}_{\underline{f}}(A, B)$, so

$$\mathrm{Map}_{\underline{f}}(A, B) :\equiv \textstyle\sum_{(h:A\to B)} \underline{w} \circ \natural\underline{h} \sim \underline{f} \circ \underline{v}$$

The case of $\underline{f} = \mathsf{id}_{\underline{X}}$ is written $\mathrm{Map}_{\underline{X}}(A, B)$.

There is now a dictionary translating between the operations on $\mathrm{Fam}(\underline{X})$ and the same operations in terms of $\mathrm{Fib}_\natural(\underline{X})$.

Definition 2.4.30. For $B : \mathrm{Fib}_\natural(\underline{Y})$, the type $\underline{f}^*B : \mathrm{Fib}_\natural(\underline{X})$ is defined to be the pullback

$$\begin{tikzcd}\underline{f}^*B \arrow[r] \arrow[d] \arrow[dr, phantom, "\lrcorner", very near start] & B \arrow[d, "\mathsf{base}_B"] \\ \underline{X} \arrow[r, "\underline{f}"'] & \underline{Y}\end{tikzcd}$$

To see that this is a type over $\underline{X}$, note that the map on the right becomes an equivalence under the functorial action of $\natural$. Because $\natural$ preserves pullbacks, the induced map $\natural(\underline{f}^*\underline{B}) \to \natural\underline{X}$ is an equivalence.

Definition 2.4.31. For a type $A : \mathrm{Fib}_\natural(\underline{X})$, the type $\underline{f}_* A : \mathrm{Fib}_\natural(\underline{Y})$ is defined by

$$\underline{f}_* A :\equiv \textstyle\sum_{(y:\underline{Y})} \prod_{((x,p):\mathsf{fib}_{\underline{f}}(y))} A_{\underline{x}}$$

Definition 2.4.32. For a dull type $\underline{A} : \mathrm{Fib}_\natural(\underline{X})$, the type $\underline{f}_! \underline{A}$ is the pushout

$$\begin{tikzcd}
\underline{X} \arrow[r, "\underline{f}"] \arrow[d, "\mathsf{zero}_{\underline{A}}"'] \arrow[dr, phantom, "\ulcorner", very near end] & \underline{Y} \arrow[d] \\
\underline{A} \arrow[r] & \underline{f}_! \underline{A}
\end{tikzcd}$$

Because $\natural$ preserves pushouts and the map on the left becomes an equivalence under the action of $\natural$, applying $\natural$ to the map $\underline{Y} \to \underline{f}_! \underline{A}$ also yields an equivalence. We use this map as our witness that $\underline{f}_! \underline{A}$ is a type over $\underline{Y}$.

Definition 2.4.33. Given $\underline{A} : \mathrm{Fib}_\natural(\underline{X})$ and $\underline{B} : \mathrm{Fib}_\natural(\underline{Y})$ define $\underline{A} \otimes \underline{B} : \mathrm{Fib}_\natural(\underline{X} \times \underline{Y})$ via the equivalence $\natural(\underline{A} \otimes \underline{B}) \simeq \natural\underline{A} \times \natural\underline{B} \simeq \underline{X} \times \underline{Y}$ of Proposition 1.3.21.

This time, the type $\underline{A} \otimes \underline{B}$ really is just the $\otimes$-type former applied directly to the inputs.

Definition {C} 2.4.34. For dull $\underline{B} : \mathrm{Fib}_\natural(\underline{Y})$ and $\underline{C} : \mathrm{Fib}_\natural(\underline{X} \times \underline{Y})$, define $\underline{B} \rhd \underline{C} : \mathrm{Fib}_\natural(\underline{X})$ by

$$\underline{B} \rhd \underline{C} :\equiv \textstyle\sum_{(x:\underline{X})} (\underline{B} \multimap \underline{C})_{(\lambda y.(\underline{x},\underline{y}))}$$

Definition {C} 2.4.35. For dull $\underline{A} : \mathrm{Fib}_\natural(\underline{X})$ and $\underline{A}' : \mathrm{Fib}_\natural(\underline{X}')$, define $\underline{A} \multimap_{\underline{X}} \underline{A}' : \mathrm{Fib}_\natural(\underline{X})$ by

$$\underline{A} \multimap_{\underline{X}} \underline{A}' :\equiv \textstyle\sum_{(x:\underline{X})} \underline{A}_{\underline{x}} \multimap \underline{A}'_{\underline{x}}$$

Theorem {C} 2.4.36. *Each of $f^*, f_*, f_!, \otimes_{(-)}, \multimap_{(-)}$ on $\mathrm{Fam}((-))$ and $\mathrm{Fib}_\natural((-))$ commutes with the equivalences $\phi_{(-)} : \mathrm{Fam}(-) \to \mathrm{Fib}_\natural(-)$.*

For most of the operations this is immediate. The most interesting is $f_!$:

Proposition 2.4.37.

$$\phi_{\underline{Y}}(\underline{f}_! \underline{A}) \simeq \underline{f}_!(\phi_{\underline{X}}(\underline{A}))$$

Proof. Starting with a type $\underline{A}$ over $\underline{X}$, we have to calculate the fibre of $\underline{f}_! \underline{A} \to \underline{Y}$ over a point $\underline{y} : \underline{Y}$. Consider the diagram

$$\begin{tikzcd}
 & \mathsf{fib}_{\underline{f}}(\underline{y}) \arrow[dl] \arrow[d] \arrow[dr] & \\
\mathsf{fib}_{\underline{f} \circ \mathsf{base}}(\underline{y}) \arrow[d] \arrow[dr] & \underline{X} \arrow[dl] \arrow[dr] & 1 \arrow[dl] \arrow[d] \\
\underline{A} \arrow[dr] & 1 \arrow[d] & \underline{Y} \arrow[dl] \\
 & \underline{Y} &
\end{tikzcd}$$

where all the vertical squares are pullbacks: the front-left and back-right by the definition of fib, and the back-left by the pasting Lemma applied to

$$\begin{tikzcd}
\mathsf{fib}_{\underline{f}\circ\mathsf{base}\circ\mathsf{zero}}(y) \arrow[r] \arrow[d] & \mathsf{fib}_{\underline{f}\circ\mathsf{base}}(y) \arrow[r] \arrow[d] & 1 \arrow[d] \\
\underline{X} \arrow[r, "\mathsf{zero}"'] & \underline{A} \arrow[r, "\underline{f}\circ\mathsf{base}"'] & \underline{Y}
\end{tikzcd}$$

noting that $\underline{f} \circ \mathsf{base} \circ \mathsf{zero} \equiv \underline{f}$. Now by descent for pushouts (Theorem 2.2.7) applied to the cube, the diagram

$$\begin{tikzcd}
\mathsf{cofib}(\mathsf{fib}_{\underline{f}}(\underline{y}) \to \mathsf{fib}_{\underline{f}\circ\mathsf{base}}(\underline{y})) \arrow[r] \arrow[d] & 1 \arrow[d] \\
\underline{f}_!\underline{A} \arrow[r] & \underline{Y}
\end{tikzcd}$$

is a pullback. And

$$\mathsf{fib}_{\underline{f}\circ\mathsf{base}}(\underline{y}) \simeq \textstyle\sum_{((x,p):\mathsf{fib}_{\underline{f}}(\underline{y}))} \underline{A}_{\underline{x}}$$

so the fibre over $\underline{y}$ may be calculated as the cofibre of the map

$$\mathsf{cofib}\left(\mathsf{fib}_{\underline{f}}(\underline{y}) \to \textstyle\sum_{((x,p):\mathsf{fib}_{\underline{f}}(\underline{y}))} \underline{A}_{\underline{x}}\right)$$

as required. □

2.4.5 Working Fibrationally

We can rephrase some of the above definitions to be more yet more type-theoretically pleasant. A family of spectra can be described 'fibrationally', interpreting a term $\underline{A} : \natural\mathsf{Spec}$ in context Γ as a parameterised spectrum over '$\natural\Gamma$'. From this perspective, a map of spaces $X \to$ '$\natural\Gamma$' is specified by a term $\underline{X}$: Space in context Γ.

The downside of this approach is that we can no longer get a handle on the base space: it is always the base space of the ambient context. Internally, $\underline{A} : \natural\mathsf{Spec}$ looks like a single (dull) spectrum, and it is only after interpretation into the semantics that its parameterisation over the context becomes visible. In particular, it is impossible to express the external operations $\underline{\otimes}$ and $\rhd$ in this style (Definitions 2.4.21 and 2.4.23), because that would require as inputs types $\underline{A} : \natural\mathsf{Spec}$ and $\underline{B} : \natural\mathsf{Spec}$ lying in 'disjoint contexts', something we have no ability to specify internally.

Weakening the ambient context will change base space without any visible difference in the term. Such a weakening with $\underline{X}$: Space is exactly the operation that corresponds to $\underline{X}^*$, where $\underline{X}$ denotes the projection $(\natural\Gamma, \underline{X}) \to \natural\Gamma$. In the case that $\underline{X}$ is a closed type, this is pullback along the projection $\underline{X} \times \natural\Gamma \to \natural\Gamma$, which aligns with the usual trick of conflating an object X with the map $X \to 1$.

The upside of this approach is that the adjoints to this weakening are now exceptionally simple to describe. Let us write $\underline{X}_!$ and $\underline{X}_*$ for the adjoints to weakening with $\underline{X}$.

$$\begin{aligned} \underline{X}_! &: (\underline{X} \to \natural\mathrm{Spec}) \to \natural\mathrm{Spec} \\ \underline{X}_!(\underline{A}) &:\equiv \mathrm{cofib}\left(\underline{X} \to \textstyle\sum_{(\underline{x}:\underline{X})} \underline{A}(\underline{x})_\natural\right)^\natural \\ \underline{X}_* &: (\underline{X} \to \natural\mathrm{Spec}) \to \natural\mathrm{Spec} \\ \underline{X}_*(\underline{A}) &:\equiv \left(\textstyle\prod_{(\underline{x}:\underline{X})} \underline{A}(\underline{x})_\natural\right)^\natural \end{aligned}$$

and the linear operation use the type formers directly: if $\underline{A}, \underline{B} : \natural\mathrm{Spec}$, then

$$\begin{aligned} \tilde{\otimes} &: \natural\mathrm{Spec} \times \natural\mathrm{Spec} \to \natural\mathrm{Spec} \\ \underline{A} \mathbin{\tilde{\otimes}} \underline{B} &:\equiv \left(\underline{A}_\natural \otimes \underline{B}_\natural\right)^\natural \\ \tilde{\multimap} &: \natural\mathrm{Spec} \times \natural\mathrm{Spec} \to \natural\mathrm{Spec} \\ \underline{A} \mathbin{\tilde{\multimap}} \underline{B} &:\equiv \left(\underline{A}_\natural \multimap \underline{B}_\natural\right)^\natural \end{aligned}$$

Chapter 3

Metatheory

We now turn to metatheoretic aspects of our type theory.

- Section 3.1 justifies some of the choices we have made in how the type theory is designed, and discusses some possible improvements.
- Section 3.2 describes the proof that the various operations we have used in the theory are admissible. The overall structure of the proof is typical — a giant mutual induction on derivations — but some of the tricks used to work with the palettes changing through a derivation are novel.

  Further details are given in Appendix B
- In Section 3.3 we describe a toy model of our type theory which is simple enough to be described internally: type-indexed families of pointed types.

3.1 On the Design of the Type Theory

The author readily admits that the rules for palettes and splits are a little odd. In this section I will try to justify some of the design choices, and explain why some 'obvious' simplifications fail.

3.1.1 Palettes and Slices.

Comma, Tensor, and Colour Labels. In the palettes that appear in the derivations of terms, there is an asymmetry between the comma palette constructor and the $\otimes$ palette constructor: the children of a (possibly iterated) $\otimes$ palette all begin with a colour label, but the children of comma palette never do. For example, we never encounter a palette with the shape $(\mathfrak{l} \prec \Phi_L), (\mathfrak{r} \prec \Phi_R)$. We can get away with this because both weakening and contraction for comma mean we never have to refer to the two sides separately. Thinking semantically, any construction that only uses one side of the comma can be weakened to the palette containing both by precomposing with the projection, and any construction that uses 'functoriality' of comma does not need to divide the palette between the two sides: one can instead apply functoriality and precompose with the diagonal map. This is no different to how contexts and variables work in ordinary (cartesian) type theory: the context never shrinks as you move up a derivation, and at the leaves you may project a variable from anywhere in the context.

For the same reason, we do not have a rule that constructs a slice $\Phi, \Phi' \vdash s, s'$ slice where $\Phi \vdash s$ slice and $\Phi' \vdash s'$ slice. Without loss of generality one can instead use the colour label immediately above the Φ, Φ'.

On the other hand, we should not arrange the rules for palettes such so that the two sides of a $\otimes$ *always* have a label immediately as the first child on either side, because associativity of $\otimes$ would not preserve this property. For example, reassociating

$$(\mathfrak{p} \prec \mathfrak{r} \otimes \mathfrak{b}) \otimes \mathfrak{y}$$

to

$$\mathfrak{r} \otimes (\mathfrak{b} \otimes \mathfrak{y})$$

leaves the right side of the outermost $\otimes$ without a label.

This would affect the syntax for slices. Anywhere that a slice implies such a reassociation of the palette, that slice would need to bind a new colour label. This would complicate the syntax for slices considerably. Additionally, whenever one replaces a spot in a palette with some other subpalette, one would need to make sure that this does not break the invariant: whether a replacement is valid will depend on the surrounding context of the spot in an irritating way.

Silent Weakening. For this type theory to be usable informally, it is crucial that cartesian weakening of the palette is invisible on the raw syntax of terms. This kind of weakening occurs whenever we use pattern matching: in a term $\mathsf{let}\ p = s \mathsf{\ in\ } c : C[s/z]$, the type of the term $c : C[p/z]$ has been weakened to include the new palette bound by the pattern p. It would be extremely confusing if this weakening were explicit throughout the type C, even places that have no relation to the variable z.

Non-Democratic Contexts. Our rules for palettes allow us to form palettes that can't possibly appear in the derivation of a closed term, for example, $\mathfrak{r} \otimes (\mathfrak{b}, \mathfrak{b}')$. Similarly, because of our slick context extension rule it is possible to form contexts that do not occur in the derivation of a closed term, even when the palette on its own is an 'ordinary' one. For example, nothing prevents us from sandwiching colours in the following sense:

$$\mathfrak{p} \prec \mathfrak{r} \otimes \mathfrak{b} \mid x^{\mathfrak{r}} : A, y^{\mathfrak{b}} : B, z^{\mathfrak{r}} : C \ \mathsf{ctx}$$

There is no binder that can produce that final context extension with z, nor can it be arrived at by filtering to a slice: the context before the filtering would also have to be sandwiched in this way.

This fact causes another departure from what we are used to from MLTT: in ordinary dependent type theory, any context or context extension can be packed into an iterated Σ-type. The same is not true here: the above sandwiched colours prevent any sensible packing of y and z into a type together.

We note that the entire context can be described (up to equivalence) as a type. The main obstacle is the possible dependence of C on $\underline{y}$, which prevents us from using symmetry to resolve the sandwiched colours. We can use Proposition 1.3.13 to break this dependency and then apply symmetry.

Contexts that may be packed into a type are sometimes called democratic [CD14], so to summarise, not all contexts are democratic in our theory, but all contexts appearing in the derivation of a closed term are democratic. Any telescope formed by using a pattern match is also democratic over the context.

3.1.2 The Top Colour

We always need some method of referring to the entire ambient context, so that variables bound by the ordinary type formers have a colour to be labelled by. In our theory we maintain the invariant that the palette of any term has a label at the top. The downside of this approach is that we need to know the name of the top colour to determine whether even a closed term is well-typed; this top colour name is not stored in the syntax of a term.

There is another option: we could use a distinguished symbol to refer to the 'top colour', say, $\mathfrak{T}$. This would solve the above issue, and have some other pleasant effects: we no longer need the 'recolouring' operation in $\otimes$-INTRO and $\multimap$-ELIM, for example. But there are some unexpected knock-on effects which made us not pursue this.

- When going under a hom-binder, the previously-unnamed top colour $\mathfrak{T}$ is assigned a name $\mathfrak{c}$. This does not only affect the type of the consequent, but also all types in the context where $\mathfrak{T}$ was previously mentioned. Worse, when later restricting to the slice $\mathfrak{c}$ (say in a use of $\otimes$-INTRO), every such occurrence of $\mathfrak{c}$ has to be restored to an occurrence of $\mathfrak{T}$.

- There is no reason that this new name has to be consistent between different hom-binders, leading to an unpleasant situation where variables that have access to the same colours are labelled with different colours themselves.

- The combination of the unitors and the 1 slice can cause complications in niche situations. For example, we are able to form the term $a \;{}_{\mathfrak{T}}\!\otimes_{\varnothing_i} (b \;{}_1\!\otimes_1 c)$. If later (due to pattern matching on this term) we have to perform the substitution

$$(x \;{}_{\mathfrak{l}\otimes\mathfrak{m}}\!\otimes_{\mathfrak{r}}\, y)[\mathfrak{T}/\mathfrak{l}, 1/\mathfrak{m}, 1/\mathfrak{r} \mid a/x, b/y]$$

  then we are in trouble: the slice on the left should be $\mathfrak{T} \otimes 1$, and now we have to determine the meaning of restricting to this slice. We cannot continue to use $\mathfrak{T}$ as the label, because that location in the palette is no longer at the top. And so any unitor slice would have to bind a new colour label to use in place of $\mathfrak{T}$ in just this case, and restricting a context to a slice would also need to handle this case specially. The reader can imagine that there is a resulting explosion of special cases to consider in all the definitions and proofs of the admissible rules.

3.1.3 The Unit

Our rules for splits allow the following split to be formed:

$$\mathfrak{p} \prec \mathfrak{r} \otimes \mathfrak{b} \vdash (\mathfrak{r} \otimes \mathfrak{b}) \boxtimes \varnothing \;\; \mathsf{presplit}$$

This might seem redundant: why not force a unitor split to use the top colour label, as in $\mathfrak{p} \boxtimes \varnothing$? The problem arises when defining substitution on raw syntax. If a substitution is built using the unitor split, then applying that substitution to a slice may yield a slice of the form $\mathfrak{r} \otimes \mathfrak{b}$. On raw syntax, there is no way to know that this slice should be replaced with $\mathfrak{p}$ on its own.

The induction principle for $\mathbb{S}$ on its own is not so useful. The only way to use it is in $\mathbb{S}$-INTRO, in which case one may as well η-contract and use the term of $\mathbb{S}$ that you started with. We include this induction principle so as to not break the analogy with $\otimes$-types.

The judgemental unit itself is necessary even if the theory did not have this induction principle. Without the judgemental unit, the unitor splits would have to bind an ordinary variable. This mixes the syntax of palettes and contexts in an uncomfortable way and would complicate the metatheory.

It may be possible to extend the type theory to make judgemental units more useful. For example, it would be convenient when working in a palette such as

$$(\mathfrak{r} \prec \varnothing_i) \otimes (\mathfrak{b} \prec \varnothing_j) \ \mathsf{palette},$$

to be able to write $⌑_{ij} : \mathbb{S}$. As it stands one has to asymmetrically use the left or right unitor map (defined via pattern match).

A further extension would be a new term former $\mathcal{J}^U$ whenever there is a split $s_L \boxtimes s_R$ split such that s_L proves $\mathcal{J}$ and s_R contains a unit. The hope would be that this term former could be made to commute with commute judgementally with the other term formers, which would make arguments that manipulate $\mathbb{S}$ more convenient. This term former would be stuck on variables, which is unproblematic, but we run into an issue when trying to commute it with $\otimes$-INTRO: there is no way to know which side U should be pushed into, and it cannot be pushed into both. The split judgement would need the ability to 'consume' units of this form. This may be possible but again leads to an explosion of special cases.

3.2 Admissible Rules

The rules for the type theory given in this thesis can be applied to various precise formulations of type theory. For example, if the starting Martin-Löf type theory is thought of algebraically as an essentially algebraic theory or quotient inductive-inductive type [ACDKN18], then we could make $\natural\Gamma$ a new context former, $\underline{a}$ a new term former, the unit a new explicit substitution, and the rules defining these operations new judgemental equalities. The bunched structure is more interesting, but could be presented using a telescope judgement as in the following:

$$\frac{}{\cdot\ \mathsf{ctx}} \qquad \frac{\Gamma \vdash A\ \mathsf{type}}{\Gamma \vdash A\ \mathsf{tele}}$$

$$\frac{\Gamma\ \mathsf{ctx}}{\natural\Gamma\ \mathsf{ctx}} \qquad \frac{\Gamma \vdash \Omega\ \mathsf{tele}}{\natural\Gamma \vdash \natural\Omega\ \mathsf{tele}}$$

$$\frac{\Gamma \vdash \Omega\ \mathsf{tele}}{\Gamma, \Omega\ \mathsf{ctx}} \qquad \frac{\Gamma \vdash \Omega\ \mathsf{tele} \qquad \Gamma, \Omega \vdash \Omega'\ \mathsf{tele}}{\Gamma \vdash \Omega, \Omega'\ \mathsf{tele}}$$

$$\frac{\natural\Gamma \vdash \Omega\ \mathsf{tele}}{\Gamma \otimes \Omega\ \mathsf{ctx}} \qquad \frac{\natural\Gamma \vdash \Omega\ \mathsf{tele} \qquad \natural(\natural\Gamma, \Omega) \vdash \Omega'\ \mathsf{tele}}{\natural\Gamma \vdash \Omega \otimes \Omega'\ \mathsf{tele}}$$

One would then add a plethora of equations asserting the correct relationships between these context constructors.

However, this kind of algebraic formulation does not immediately capture some important aspects of our syntax. The first is that the $\natural\Gamma$ and $\underline{a}$ are definable in terms of marked context extension and marked variables — this would need to be recovered as part of a canonicity proof. The second is that the unit is "silent," i.e. it does not change the raw proof term — in the algebraic style, it would be made explicit analogously to weakening. But perhaps the most inconvenient aspect is that the palette and slice syntax is lost entirely. When applying $\otimes$-INTRO or $\multimap$-ELIM, one would need to apply a possibly complicated stuck substitution to the context to massage it into the desired split $\Gamma \otimes \Omega$ ctx.

The simplest way to make these observations formal is to adopt a more traditional syntax, where the subjects of a judgement $\Phi \mid \Gamma \vdash a : A$ are a raw syntax context $\Phi \mid \Gamma$, a raw syntax term a, and a raw syntax type A. These weak invariants "break the loop" so that the typing rules can be defined prior to the admissible rules — otherwise, one requires the admissible rules to know that the presuppositions of the judgements are satisfied. For example, a premise of $\mathfrak{t} \prec \Phi, \Gamma, \underline{x} : A$ ctx is $\mathfrak{t} \mid \underline{\Gamma} \vdash A$ type, but $\underline{\Gamma}$ ctx only follows from an admissible rule.

3.2.1 Official Rules

When making rules in this style precise, there are some somewhat arbitrary choices about the presuppositions of a judgement. For example, a derivation of $\mathfrak{t} \prec \Phi \mid \Gamma \vdash a : A$ might

1. presuppose that $\mathfrak{t} \prec \Phi \mid \Gamma$ is well-formed, i.e. the subject is really a raw context $\mathfrak{t} \prec \Phi \mid \Gamma$ rawctx such that there is a derivation $\mathfrak{t} \prec \Phi \mid \Gamma$ ctx.

2. require $\mathfrak{t} \prec \Phi \mid \Gamma$ ctx at each step of the derivation of $a : A$.

3. neither of the above, i.e. formally one can make derivations of $\mathfrak{t} \prec \Phi \mid \Gamma \vdash a : A$ for an ill-formed context Γ, but we generally will only be interested in derivations when $\mathfrak{t} \prec \Phi \mid \Gamma$ ctx.

The first has the same problem as the algebraic syntax — it requires the admissible rules to be mutual with the basic ones — and the second is a bit far from an implementation, which inductively maintains the invariant that the context is well-formed without repeatedly re-checking it, so we follow the third option for contexts in all judgements. However, for terms, the rules will ensure that the type is well-formed, and for equality rules, the rules will ensure that the terms are well-typed: we can prove Lemmas of the form

- If Γ ctx and $\Gamma \vdash a : A$ then $\Gamma \vdash A$ type.
- If Γ ctx and $\Gamma \vdash a \equiv a' : A$ then $\Gamma \vdash a : A$ and $\Gamma \vdash a' : A$.

This is because, following [Str91; Hof97]'s approach to categorical semantics and initiality, we officially adopt a fully annotated term syntax, where every inference rule has a direct typing premise for each term/type metavariable appearing in the rule. (Of course, this is also a bit far from an implementation.) Together with the admissible rules, these premises will be enough to ensure that the types in a typing judgement and terms in an equality judgement are well-formed. De Boer, Brunerie, Lumsdaine and Mörtberg (see [Boe20]) have given a fully mechanised initiality proof for roughly this style of presentation, though we treat variable binding informally rather than using de Bruijn indices.

For the judgements for well-formed contexts, terms and types, we do presuppose that the palette is well formed: the operations on palettes, slices and splits may be defined prior to any contact with contexts, terms and types. The structure of palettes is sufficiently simple that there is no need to "break the loop" here; and palettes do not appear in terms or types, only the raw syntax of slices.

3.2.2 Operations on Raw Syntax

The admissible operations on contexts and terms are defined on raw syntax (prior to typing). This accords with the way we use these operations when working informally: we do not want to have to think about the exact structure of a derivation as we apply these operations to terms.

We use the following judgements for raw terms:

- ϕ denotes a palette scope, which is an unstructured list of colour names and unit names.
- $\phi \vdash s$ rawpreslice denotes a raw preslice in scope ϕ, so list with entries either a colour from the scope ϕ or the symbol 1.
- $\phi \vdash s$ rawslice denotes a raw slice in the scope ϕ, which is a raw preslice that may bind a top colour.
- $\phi \mid \gamma \vdash a$ rawterm denotes a raw term in the palette scope ϕ and scope γ. We think of raw syntax as being intrinsically scoped, so a raw term a is judged relative to a scope γ consisting of variable names only, with no associated types or marks.

  We do not distinguish between ordinary and marked variables in scopes γ, nor do we record a colour label for each variable.
- $\phi \mid \Gamma$ rawctx denotes a raw context consisting of a list of variables with a raw term as a 'type' of each. Each variable is unmarked or marked. Unmarked variables are annotated with a colour name from the scope ϕ. Marked variables bind a fresh colour name that they use as their 'top colour'.

- $\phi \mid \gamma \vdash \psi \mid \Delta$ rawext similarly denotes a raw context extension in the scope $\phi \mid \gamma$ with marks and colours on the variables. This extension Δ is also permitted to use the new colours in the scope ψ. This judgement serves double duty: both well-typed context extensions and context telescopes have underlying raw extensions of this type. (We touched on the difference between these notions in Remark 1.6.6.)

- $\phi \vdash \kappa : \psi$ denotes a palette substitution, as raw syntax this is a derivation of a palette substitution where the slices that appear do not need to be well-formed.

- $\phi \mid \gamma \vdash (\kappa \mid \theta) : \psi \mid \omega$ denotes a telescope substitution, so θ is a list of raw terms in scope $\phi \mid \gamma$.

The complete list of operations on raw syntax that we need are as follows:

- Extracting the top colour from a slice $\ulcorner s \urcorner$.

- Extracting the underlying preslice $\mathsf{u}(s)$.

- Recolouring $\mathcal{J}^{\mathfrak{c} \leftrightarrow \mathfrak{d}}$ of slices, contexts, telescopes, terms and types.

- Marking $\mathcal{J}^{\mathsf{m}\Phi}$ of slices.

- Marking $\mathcal{J}^{\mathsf{m}\Phi|\Gamma}$ of contexts, telescopes, terms and types.

- Marking $\underline{\Gamma}$ of contexts.

- Filtering Γ^{s} of contexts and telescopes.

- Substitution $\mathcal{J}[\kappa]$ of slices.

- Substitution $\mathcal{J}[\kappa \mid \theta]$ of telescopes, terms and types.

- Tensor merging $\mathcal{J}[\![(\mathfrak{t} \prec s_L \boxtimes s_R / \mathfrak{t}')]\!]$ of slices, telescopes, terms and types.

The shapes of these operations on raw syntax are given in Figure 3.1. We use $\mathcal{j}$ to stand in for one of a raw telescope or a raw term.

When we apply these operations to well-typed telescopes and terms, we will just write, for example, $\Delta^{\mathsf{m}\Gamma}$ and $a^{\mathsf{m}\Gamma}$, letting Γ represent its underlying list of variables.

The actual definitions are given in Section A.4; the only definition that is not entirely obvious is slice substitution, which stops when we reach the first occurrence of the colour as its own slice.

Many of the equations relating these operations now hold on the level of raw syntax. For example,

Lemma 3.2.1. *Marking is idempotent on the level of raw syntax.*

- *If* $\mathfrak{t}, \phi \mid \Gamma$ rawctx *then* $\underline{\Gamma} \equiv_\alpha \underline{\underline{\Gamma}}$

- *If* $\mathfrak{t}, \phi \mid \Gamma$ rawctx *and* $\mathfrak{t}, \phi \mid \gamma \vdash \psi \mid \Delta$ rawext *then* $\underline{\Gamma, \Delta} \equiv_\alpha \underline{\underline{\Gamma}, \Delta^{\mathsf{m}\phi|\gamma}}$

- *If* $\phi, \phi', \phi'' \mid \gamma, \gamma' \vdash \psi \mid \Delta$ rawext *then* $(\Delta^{\mathsf{m}\phi|\gamma})^{\mathsf{m}(\phi,\phi'|\gamma,\gamma')} \equiv_\alpha \Delta^{\mathsf{m}(\phi,\phi'|\gamma,\gamma')} \equiv_\alpha (\Delta^{\mathsf{m}(\phi,\phi'|\gamma,\gamma')})^{\mathsf{m}\phi'|\gamma'}$

Slice operations:

$$\frac{\Phi \ \mathsf{palette} \qquad \phi \vdash s \ \mathsf{rawslice}}{\Phi^{s} \vdash \ulcorner s \urcorner \ \mathsf{colour}} \qquad \frac{\phi \vdash s \ \mathsf{rawslice}}{\phi \vdash \mathsf{u}(s) \ \mathsf{rawpreslice}}$$

Marking:

$$\frac{\mathfrak{t}, \phi \mid \Gamma \ \mathsf{rawctx}}{\mathfrak{t} \mid \underline{\Gamma} \ \mathsf{rawctx}} \qquad \frac{\mathfrak{t}, \phi \mid \gamma \vdash \psi \mid \Delta \ \mathsf{rawext}}{\mathfrak{t} \mid \gamma \vdash \cdot \mid \underline{\Delta} \ \mathsf{rawext}}$$

$$\frac{\phi, \phi' \vdash s \ \mathsf{rawslice}}{\phi' \vdash s^{\mathsf{m}\phi} \ \mathsf{rawslice}} \qquad \frac{\psi, \phi, \psi' \mid \gamma, \gamma' \vdash \Delta \ \mathsf{rawext}}{\psi, \psi' \mid \gamma, \gamma' \vdash \Delta^{\mathsf{m}\phi|\gamma} \ \mathsf{rawext}} \qquad \frac{\psi, \phi, \psi' \mid \gamma, \gamma' \vdash a \ \mathsf{rawterm}}{\psi, \psi' \mid \gamma, \gamma' \vdash a^{\mathsf{m}\phi|\gamma} \ \mathsf{rawterm}}$$

Tinting:

$$\frac{\Phi \ \mathsf{palette} \qquad \phi \vdash s \ \mathsf{rawslice} \qquad \phi \mid \Gamma \ \mathsf{rawctx}}{\phi \mid \Gamma^{s} \ \mathsf{rawctx}}$$

Recolouring:

$$\frac{\phi, \mathfrak{c}, \phi' \mid \gamma \vdash \mathcal{J}}{\phi, \mathfrak{d}, \phi' \mid \gamma \vdash \mathcal{J}^{\mathfrak{d} \leftrightarrow \mathfrak{c}}}$$

Telescope substitution:

$$\frac{\phi \vdash \kappa : \psi \qquad \phi, \psi, \phi' \vdash s \ \mathsf{rawpreslice}}{\phi, \phi' \vdash s[\kappa] \ \mathsf{rawpreslice}} \qquad \frac{\phi \vdash \kappa : \psi \qquad \phi, \psi, \phi' \vdash s \ \mathsf{rawslice}}{\phi, \phi' \vdash s[\kappa] \ \mathsf{rawslice}}$$

$$\frac{\phi \mid \gamma \vdash (\kappa \mid \theta) : \psi \mid \omega \qquad \mathfrak{t} \in \phi, \psi, \phi' \qquad \phi, \psi, \phi' \mid \gamma, \omega, \gamma' \vdash \mathcal{J}}{\phi, \phi' \mid \gamma, \gamma' \vdash \mathcal{J}[\kappa \mid \theta]^{\mathsf{att}}}$$

Slice substitution:

$$\frac{\phi \vdash t \ \mathsf{rawslice} \qquad \phi, \mathfrak{c}, \phi' \mid \gamma \vdash \mathcal{J}}{\phi, \phi' \mid \gamma \vdash \mathcal{J}[\![t/\mathfrak{c}]\!] \ \mathsf{rawslice}}$$

Figure 3.1: Operations on Raw Syntax

- *If* $\phi, \psi, \psi' \vdash s$ $\mathsf{rawpreslice}$ *then* $(s^{\mathsf{m}\phi})^{\mathsf{m}(\phi,\psi)} \equiv_\alpha a^{\mathsf{m}(\phi,\psi)} \equiv_\alpha (a^{\mathsf{m}(\phi,\psi)})^{\mathsf{m}\phi}$
- *If* $\phi, \psi, \psi' \vdash s$ $\mathsf{rawpreslice}$ *or* $\phi, \psi, \psi' \vdash s$ $\mathsf{rawslice}$ *then* $(s^{\mathsf{m}\phi})^{\mathsf{m}(\phi,\psi)} \equiv_\alpha s^{\mathsf{m}(\phi,\psi)} \equiv_\alpha (s^{\mathsf{m}(\phi,\psi)})^{\mathsf{m}\phi}$
- *If* $\phi, \psi, \psi' \mid \gamma, \delta \vdash a$ $\mathsf{rawterm}$ *then* $(a^{\mathsf{m}\phi|\gamma})^{\mathsf{m}(\phi,\psi|\gamma,\delta)} \equiv_\alpha a^{\mathsf{m}(\phi,\psi|\gamma,\delta)} \equiv_\alpha (a^{\mathsf{m}(\phi,\psi|\gamma,\delta)})^{\mathsf{m}\phi|\gamma}$

Proof. Induction on the structure of the judgement.

For preslices, $s^{\mathsf{m}\phi}$ is calculated by replacing any colour in ϕ with 1, doing so twice has no additional effect. For slices, the bound colour is not affected by the marking operation and so equality follows by the equation for preslices.

For terms, variable cases are clear, as marked variables always remain marked, and unmarked variable become marked iff they are in γ, γ'. The remaining cases are immediate by induction. □

3.2.3 Proof Ideas

We give a general overview of the strategy used to prove the operations well-typed, with the complete details in Section B.

Palette Spots. When moving through a derivation, the palette changes in various ways. To have sufficiently general inductive hypotheses in the proofs, we need the rules to be general enough to apply under any change that happens to the palette when moving from the conclusion to a premise. For us, there are five possible changes:

- An ordinary binding of a single variable of the top colour (Σ-FORM, Π-FORM, Π-INTRO),
- A binding of a telescope (MATCH),
- A linear binding of a single variable (⊸-FORM, ⊸-INTRO),
- Marking of the entire context (♮-FORM, ♮-INTRO, ⊗-FORM, ⊸-FORM),
- Restriction of the context to a slice (⊗-INTRO, ⊸-ELIM).

To describe rules that are general enough, we introduce some additional judgements that represent a particular 'spot' in the palette. These are similar to the 'context holes' one sees in other work on bunched implication; locations in a context that can be filled with another context.

$$\frac{}{\Xi\{\Xi\}\ \mathsf{spot}} \qquad \frac{\Xi\{\Phi\}\ \mathsf{spot}}{(\mathfrak{c} \prec \Xi)\{\Phi\}\ \mathsf{spot}}$$

$$\frac{\Xi\{\Phi\}\ \mathsf{spot}}{(\Xi, \Xi')\{\Phi\}\ \mathsf{spot}} \qquad \frac{\Xi\{\Phi\}\ \mathsf{spot}}{(\Xi', \Xi)\{\Phi\}\ \mathsf{spot}} \qquad \frac{\Xi\{\Phi\}\ \mathsf{spot}}{(\Xi \otimes \Xi')\{\Phi\}\ \mathsf{spot}} \qquad \frac{\Xi\{\Phi\}\ \mathsf{spot}}{(\Xi' \otimes \Xi)\{\Phi\}\ \mathsf{spot}}$$

We also introduce a related second (and third) kind of spot $\Xi\{\Phi, \dots\}$ and $\Xi\{\mathfrak{t} \prec (\Phi, \dots)\}$, that keeps track of the enclosing label of a comma bunch: We will need this because the pattern matching rule places a new comma bunch below the top colour: we move from $\mathfrak{t} \prec \Phi$ to $\mathfrak{t} \prec (\Phi, \Phi')$.

$$\frac{}{\Xi\{\Xi, \ldots\} \ \mathsf{spot}} \qquad \frac{\Xi\{\Phi, \ldots\} \ \mathsf{spot}}{(\Xi, \Xi')\{\Phi, \ldots\} \ \mathsf{spot}} \qquad \frac{\Xi\{\Phi, \ldots\} \ \mathsf{spot}}{(\mathfrak{t} \prec \Xi)\{\mathfrak{t} \prec \Phi, \ldots\} \ \mathsf{spot}}$$

$$\frac{\Xi\{\mathfrak{t} \prec \Phi, \ldots\} \ \mathsf{spot}}{(\mathfrak{c} \prec \Xi)\{\mathfrak{t} \prec \Phi, \ldots\} \ \mathsf{spot}}$$

$$\frac{\Xi\{\mathfrak{t} \prec \Phi, \ldots\} \ \mathsf{spot}}{(\Xi', \Xi)\{\mathfrak{t} \prec \Phi, \ldots\} \ \mathsf{spot}} \qquad \frac{\Xi\{\mathfrak{t} \prec \Phi, \ldots\} \ \mathsf{spot}}{(\Xi, \Xi')\{\mathfrak{t} \prec \Phi, \ldots\} \ \mathsf{spot}}$$

$$\frac{\Xi\{\mathfrak{t} \prec \Phi, \ldots\} \ \mathsf{spot}}{(\Xi \otimes \Xi')\{\mathfrak{t} \prec \Phi, \ldots\} \ \mathsf{spot}} \qquad \frac{\Xi\{\mathfrak{t} \prec \Phi, \ldots\} \ \mathsf{spot}}{(\Xi' \otimes \Xi)\{\mathfrak{t} \prec \Phi, \ldots\} \ \mathsf{spot}}$$

This new palette is associated 'the wrong way': in a palette spot $\Xi\{\mathfrak{t} \prec \Phi, \ldots\}$ spot, the label $\mathfrak{t}$ becomes distant to Φ in the underlying palette (but still ultimately with only comma bunches in between.)

The most important operation on spots is replacing the contents of a spot with some other palette:

$$\frac{\Xi\{\Phi\} \ \mathsf{spot} \qquad \Psi \ \mathsf{palette}}{\Xi\{\downarrow\Psi\} \ \mathsf{palette}} \qquad \frac{\Xi\{\Phi\} \ \mathsf{spot} \qquad \Psi \ \mathsf{palette}}{(\Xi\{\downarrow\Psi\})\{\Psi\} \ \mathsf{spot}}$$

The second rule somewhat awkwardly states that, after replacing a spot in a palette, the replacement palette is again a spot in the result.

Throughout the proofs we will use a more convenient notation for spots, writing the spot inline at the location in the palette it appears. So, if $(\Xi, \Xi')\{\Phi\}$ spot is derived from $\Xi\{\Phi\}$ spot, we will simply write $\Xi\{\Phi\}, \Xi'$ spot, and so on.

Useful Invariants. The rules of the theory are also carefully arranged so that all typing rules monotonically add marks as one moves from the conclusion to the premises. Additionally, the colour label on any variable in the context is never changed by any of the rules when moving up a derivation.

The marking operation on contexts applies the marking operation to the types, but that is the only change to types in the context that occurs. This marking operation is identical on raw syntax regardless of what the current top colour of the palette is. There is therefore no repeated changing of the raw syntax of types in the context as you move up a derivation; the type is marked at most one time.

The 'Dispatching' Rules. The operations on raw syntax are defined such that the data used in the operation is left as undisturbed as possible as we traverse the term. This means that the context in which the operation is typed can gradually drift away from the context in which the target is typed.

For example, when applying a substitution $\mathfrak{t} \prec \Phi \mid \Gamma \vdash (\kappa \mid \theta) : \Psi \mid \Omega$ to a term $\mathfrak{t} \prec \Phi, \Psi \mid \Gamma, \Omega \vdash a : A$, induction into the premises of $\otimes$-INTRO causes the palette and context in the term to be restricted to some slice $\mathfrak{t} \prec \Phi, \Psi \vdash s \ \mathsf{slice}$, and this no longer aligns with the context of $(\kappa \mid \theta)$.

This issue is resolved by having *multiple* typing rules for the *same* operation on raw syntax, for the various situations that can occur. In proof that each typing rule is admissible, the inductive cases may refer to some other typing rule for the same operation, as the situation requires.

For example, the 'headline' substitution result is the following.

$$\textsc{subst}\ \frac{\mathfrak{t} \prec \Phi \mid \Gamma \vdash (\kappa \mid \theta) : \Psi \mid \Omega \qquad \mathfrak{t} \prec \Phi, \Psi \mid \Gamma, \Omega \vdash \mathcal{J}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \mathcal{J}[\kappa \mid \theta]}$$

This is the rule as it is actually needed in the pattern-matching rules, as well as the ordinary single-variable substitutions present in the rules for Σ and Π. The first step is, as usual, to allow the judgement being substituted into to use an extension Γ' of the context with additional variables. For this to make sense we need the colour labels in Γ' to not be from Ψ, because κ substitutes these colours for an arbitrary slice of Φ, and our theory has no notion of 'variables labelled with a slice'. Cartesian extensions are fine as they bind variables of colour $\mathfrak{t}$, which is not in Ψ.

$$\textsc{subst}\ \frac{\mathfrak{t} \prec \Phi \mid \Gamma \vdash (\kappa \mid \theta) : \Psi \mid \Omega \qquad \mathfrak{t} \prec \Phi, \Psi \mid \Gamma, \Omega, \Gamma' \vdash \mathcal{J}}{\mathfrak{t} \prec \Phi \mid \Gamma, \Gamma'[\kappa \mid \theta] \vdash \mathcal{J}[\kappa \mid \theta]}$$

The next step is to allow substitution into a judgement where the palette has also been extended. For this we use a palette spot, like so:

$$\textsc{subst}\ \frac{\begin{array}{c}\mathfrak{s} \prec \Xi\{\mathfrak{t} \prec \Phi, \Psi, \ldots\}\ \mathsf{spot} \\ \mathfrak{t} \prec \Phi \mid \Gamma \vdash (\kappa \mid \theta) : \Psi \mid \Omega \qquad \mathfrak{s} \prec \Xi\{\mathfrak{t} \prec \Phi, \Psi, \ldots\} \mid \Gamma, \Omega, \Gamma' \vdash \mathcal{J}\end{array}}{\mathfrak{s} \prec \Xi\{\downarrow\mathfrak{t} \prec \Phi, \ldots\} \mid \Gamma, \Gamma'[\kappa \mid \theta] \vdash \mathcal{J}[\kappa \mid \theta]^{\mathsf{at}\mathfrak{s}}}$$

This covers all the possible changes to the palette except one: the restriction to a slice that occurs in $\otimes$-INTRO and $\multimap$-ELIM. Here the substitution rule branches into four alternatives. The different versions correspond to the mutually exclusive ways an arbitrary slice $\mathfrak{s} \prec \Xi \vdash s \ \mathsf{slice}$ might relate to the spot $\Xi\{\mathfrak{t} \prec \Phi, \Psi, \ldots\}\ \mathsf{spot}$:

- s contains $\mathfrak{t}$, and so all of the spot. Then there is a spot $\Xi^s\{\mathfrak{t} \prec \Phi, \Psi, \ldots\}\ \mathsf{spot}$, and we can apply the substitution rule given above.
- s has a nontrivial intersection v with Ψ. Then the result of the substitution in the conclusion should use the colours in $s[\kappa]$, which has a nontrivial intersection with Φ.
- s has a nontrivial intersection w with Φ. Then the rules for slices ensure that s uses no colours from Ψ, and so $s[\kappa] \equiv s$. This continues to have a non-trivial intersection with Φ in the conclusion.
- s does not intersect the spot whatsoever. Then the judgement under the slice must already be dull with respect to Γ

The rules corresponding to each of these possibilities are as follows.

$$\textsc{subst/cod}\;\frac{\Psi \vdash v\ \mathsf{preslice}_\epsilon \qquad \mathfrak{s} \prec \Xi\{\Psi^v\}\ \mathsf{spot} \qquad \mathfrak{t} \prec \Phi \mid \Gamma \vdash (\kappa \mid \theta) : \Psi \mid \Omega \qquad \mathfrak{s} \prec \Xi\{\Psi^v\} \mid \underline{\Gamma}, \Omega^v, \Gamma' \vdash \mathcal{J}}{\mathfrak{s} \prec \Xi\{{\downarrow}(\mathfrak{t} \prec \Phi)^{v[\kappa]}\} \mid \Gamma^{v[\kappa]}, \Gamma'[\kappa \mid \theta] \vdash \mathcal{J}[\kappa \mid \theta]^{\mathsf{at}\mathfrak{s}}}$$

$$\textsc{subst/dom}\;\frac{\Phi \vdash w\ \mathsf{preslice}_\epsilon \qquad \mathfrak{s} \prec \Xi\{\Phi^w\}\ \mathsf{spot} \qquad \mathfrak{t} \prec \Phi \mid \Gamma \vdash (\kappa \mid \theta) : \Psi \mid \Omega \qquad \mathfrak{s} \prec \Xi\{\Phi^w\} \mid \Gamma^w, \underline{\Omega}, \Gamma' \vdash \mathcal{J}}{\mathfrak{s} \prec \Xi\{\Phi^w\} \mid \Gamma^w, \Gamma'[\kappa \mid \theta] \vdash \mathcal{J}[\kappa \mid \theta]^{\mathsf{at}\mathfrak{s}}}$$

$$\textsc{subst/marked}\;\frac{\mathfrak{t} \prec \Phi \mid \Gamma \vdash (\kappa \mid \theta) : \Psi \mid \Omega \qquad \mathfrak{s} \prec \Xi \mid \underline{\Gamma}, \underline{\Omega}, \Gamma' \vdash \mathcal{J}}{\mathfrak{s} \prec \Xi \mid \underline{\Gamma}, \Gamma'[\kappa \mid \theta] \vdash \mathcal{J}[\kappa \mid \theta]^{\mathsf{at}\mathfrak{s}}}$$

The final ingredient is a 'dispatch' rule, that given a slice $\mathfrak{s} \prec \Xi \vdash s$ slice in *any* of the above situations, chooses the appropriate rule to use to perform the substitution under that slice. Specifically, we prove the following.

Lemma 3.2.2. *Let* $\mathfrak{s} \prec \Xi\{(\mathfrak{t} \prec \Phi, \Psi)^?\} \mid \Gamma, \Omega, \Gamma'$ ctx *denote one of the above four situations, so one of*

$$\begin{array}{lll}
\mathfrak{s} \prec \Xi\{\mathfrak{t} \prec \Phi, \Psi, \dots\} & \mid \Gamma, \Omega, \Gamma' & \mathsf{ctx} \\
\mathfrak{s} \prec \Xi\{\Psi^v\} & \mid \underline{\Gamma}, \Omega^v, \Gamma' & \mathsf{ctx}\ \textit{where}\ \Psi \vdash v\ \mathsf{preslice} \\
\mathfrak{s} \prec \Xi\{\Phi^w\} & \mid \Gamma^w, \underline{\Omega}, \Gamma' & \mathsf{ctx}\ \textit{where}\ \Phi \vdash w\ \mathsf{preslice} \\
\mathfrak{s} \prec \Xi & \mid \underline{\Gamma}, \underline{\Omega}, \Gamma' & \mathsf{ctx}
\end{array}$$

and $\mathfrak{s} \prec \Xi\{{\downarrow}(\mathfrak{t} \prec \Phi)^{?[\kappa]}\} \mid \Gamma, \Gamma'[\kappa \mid \theta]$ ctx *the corresponding conclusion contexts*

$$\begin{array}{lll}
\mathfrak{s} \prec \Xi\{{\downarrow}\mathfrak{t} \prec \Phi, \dots\} & \mid \Gamma, \Gamma'[\kappa \mid \theta] & \mathsf{ctx} \\
\mathfrak{s} \prec \Xi\{{\downarrow}(\mathfrak{t} \prec \Phi)^{v[\kappa]}\} & \mid \Gamma^{v[\kappa]}, \Gamma'[\kappa \mid \theta] & \mathsf{ctx}\ \textit{where}\ \Psi \vdash v\ \mathsf{preslice} \\
\mathfrak{s} \prec \Xi\{\Phi^w\} & \mid \Gamma^w, \Gamma'[\kappa \mid \theta] & \mathsf{ctx}\ \textit{where}\ \Phi \vdash w\ \mathsf{preslice} \\
\mathfrak{s} \prec \Xi & \mid \underline{\Gamma}, \Gamma'[\kappa \mid \theta] & \mathsf{ctx}
\end{array}$$

respectively. In each case we can substitute 'under a slice':

$$\textsc{subst/dispatch}\;\frac{\mathfrak{s} \prec \Xi\{(\mathfrak{t} \prec \Phi, \Psi)^?\} \vdash s\ \mathsf{slice} \qquad \mathfrak{t} \prec \Phi \mid \Gamma \vdash (\kappa \mid \theta) : \Psi \mid \Omega \qquad (\mathfrak{s} \prec \Xi\{{\downarrow}(\mathfrak{t} \prec \Phi, \Psi)^?\})^s \mid (\Gamma, \Omega, \Gamma')^s \vdash \mathcal{J}}{(\mathfrak{s} \prec \Xi\{{\downarrow}(\mathfrak{t} \prec \Phi)^{?[\kappa]}\})^{s[\kappa]} \mid (\Gamma, \Gamma'[\kappa \mid \theta])^{s[\kappa]} \vdash \mathcal{J}[\kappa \mid \theta]^{\mathsf{at}\ulcorner s \urcorner}}$$

□

Dispatching rules of a similar form are also used for the other operations, and this significantly streamlines the cases for ⊗-INTRO and ⊸-ELIM.

The central results we have proved are the following.

Theorem 3.2.3. *Substitution is admissible.*

$$\textsc{subst}\ \frac{\mathsf{t} \prec \Phi \mid \Gamma \vdash (\kappa \mid \theta) : \Psi \mid \Omega \qquad \mathsf{t} \prec \Phi, \Psi \mid \Gamma, \Omega, \Gamma' \vdash \mathcal{J}}{\mathsf{t} \prec \Phi \mid \Gamma, \Gamma'[\kappa \mid \theta] \vdash \mathcal{J}[\kappa \mid \theta]}$$

□

Theorem 3.2.4. *Marking of contexts is admissible.*

$$\textsc{mark-ctx}\ \frac{\mathsf{t} \prec \Phi \mid \Gamma\ \mathsf{ctx}}{\mathsf{t} \mid \underline{\Gamma}\ \mathsf{ctx}}$$

□

Theorem 3.2.5. *Filtering of contexts is admissible.*

$$\textsc{filter}\ \frac{\Phi \vdash s\ \mathsf{slice} \qquad \Phi \mid \Gamma\ \mathsf{ctx}}{\Phi^{s} \mid \Gamma^{s}\ \mathsf{ctx}}$$

□

Theorem 3.2.6. *Marking is admissible.*

$$\textsc{mark}\ \frac{\mathsf{t} \prec \Phi \mid \Gamma \vdash \mathcal{J}}{\mathsf{t} \mid \underline{\Gamma} \vdash \underline{\mathcal{J}}}$$

□

Theorem 3.2.7. *Recolouring is admissible.*

$$\textsc{recolour}\ \frac{\mathsf{t}' \prec \Phi \mid \Gamma \vdash \mathcal{J}}{\mathsf{t} \prec \Phi \mid \Gamma^{\mathsf{t} \leftrightarrow \mathsf{t}'} \vdash \mathcal{J}^{\mathsf{t} \leftrightarrow \mathsf{t}'}}$$

□

Theorem 3.2.8. *Tensor merging is admissible.*

$$\textsc{merge}\ \frac{\begin{array}{c}\mathsf{t} \prec \Phi \vdash t_L \boxtimes t_R\ \mathsf{split} \\ \mathsf{t}' \prec \Phi^{t_L} \otimes \Phi^{t_R} \mid \Gamma^{s_L \otimes s_R} \vdash \mathcal{J}\end{array}}{\mathsf{t} \prec \Phi \mid \Gamma \vdash \mathcal{J}[\![(\mathsf{t} \prec t_L \boxtimes t_R / \mathsf{t}')]\!]}$$

□

3.3 A Toy Model in Parameterised Pointed Types

We present a concrete interpretation of the type formers of our theory in parameterised pointed types, which has a similar structure to parameterised spectra but is much simpler to describe. The ∞-topos of *parameterised pointed spaces* is defined similarly to PSpec, but with the ∞-category of pointed spaces $\mathcal{S}_\star$ taking the place of Spec, so an object consists of a space B and a family $E : \mathrm{Fun}(B, \mathcal{S}_\star)$ of pointed spaces. In intensional type theory, we can define a weak version of this model, where many equations that should be strict equalities (such as the β- and η-rules for the type formers) hold only up to paths. Indeed, in this internal presentation, function types in the theory are interpreted as functions that preserve the point only up to a path whereas we conjecture that there is an external model in parameterised pointed spaces where functions are pointed up to equality.

This model has been considered independently in some related work. Some aspects, including the interpretation of the universe, were described in a talk by Buchholtz [Buc19], which discussed simple, internally definable models of cohesion. Parameterised pointed spaces can be seen as the category of homotopy coherent diagrams over the walking section/retraction. An ∞-presheaf Γ on $\{s : 1 \to 0, r : 0 \to 1 \mid r \circ s = \mathsf{id}_0\}$ consists of a space $\Gamma(0)$ and a space $\Gamma(1)$ with a projection $\Gamma(1) \to \Gamma(0)$ and a section $\Gamma(0) \to \Gamma(1)$. Taking the fibre of the projection, we can think of $\Gamma(1)$ as a dependent type on $\Gamma(0)$, with each type in the fibre equipped with a distinguished point determined by the section s.

A paper of Kraus and Sattler [KS17] discusses how, for certain "Reedy" diagrams, the data of a coherent diagram can be given a finite internal description, without needing to specify infinitely many coherences. Kraus and Sattler's construction applies to this index category, and their construction yields a description equivalent to the one presented here. Coquand, Ruch, and Sattler [CRS20] also consider constructive sheaf models of univalent type theory on the walking section-retraction.

3.3.1 Contexts and Substitutions

In this (weak) model, each context 'Γ ctx' is interpreted as an element of a record type

Definition 3.3.1 (Contexts in Pointed Spaces)**.**

$$\begin{aligned} \mathsf{Ctx} :\equiv \{&\mathsf{B} : \mathcal{U}, \\ &\mathsf{E} : \mathsf{B} \to \mathcal{U}, \\ &\mathsf{p} : \textstyle\prod_{(g:\mathsf{B})} \mathsf{E}(g)\} \end{aligned}$$

consisting of a type B, a family of types over it E, and a section p. Equivalently, we could package E and p together into a family of pointed types. This corresponds to the section/retraction pair

$$\begin{tikzcd} \sum_{(g:\mathsf{B}\Gamma)} \mathsf{E}\Gamma(g) \arrow[d, bend left, "\mathsf{pr}_1"] \\ \mathsf{B}\Gamma \arrow[u, "{(g,\mathsf{p}g)}"] \end{tikzcd}$$

With this in mind, we refer to $\mathsf{B}\Gamma$ as the *base* or *index space*, each $\mathsf{E}\Gamma(-)$ as a *fibre*, and p as the *point* (though this is a little imprecise, since really it is a family of points, one for each fibre).

A substitution '$\Gamma \vdash \theta : \Delta$' between $\Gamma, \Delta : \mathsf{Ctx}$ is then a map of the base and a map of the fibres that commute with the section, represented by the record type

Definition 3.3.2 (Substitutions in Pointed Spaces)**.**

$$\begin{aligned}
\{&\mathsf{B} : \mathsf{B}\Gamma \to \mathsf{B}\Delta, \\
&\mathsf{E} : \textstyle\prod_{(g:\mathsf{B}\Gamma)} \mathsf{E}\Gamma(g) \to \mathsf{E}\Delta(\mathsf{B}(g)), \\
&\mathsf{p} : \textstyle\prod_{(g:\mathsf{B}\Gamma)} \mathsf{E}(g, \mathsf{p}\Gamma(g)) =_{\mathsf{E}\Delta(\mathsf{B}(g))} \mathsf{p}\Delta(\mathsf{B}(g))\}
\end{aligned}$$

This amounts to a commutative diagram

$$\begin{tikzcd}
\sum_{(g:\mathsf{B}\Gamma)} \mathsf{E}\Gamma(g) \arrow[r] \arrow[d] & \sum_{(d:\mathsf{B}\Delta)} \mathsf{E}\Delta(d) \arrow[d] \\
\mathsf{B}\Gamma \arrow[r] \arrow[u, bend left] & \mathsf{B}\Delta \arrow[u, bend left]
\end{tikzcd}$$

Definition 3.3.3 (Empty Context in Pointed Spaces)**.** The empty context '$\varnothing$ ctx' is given by the unit type over the unit type.

$$\begin{aligned}
\mathsf{B}(\varnothing) &:\equiv 1 \\
\mathsf{E}(\varnothing)(\star) &:\equiv 1 \\
\mathsf{p}(\varnothing)(\star) &:\equiv \star
\end{aligned}$$

Note that we use "copattern" notation [APTS13] to describe elements of records/functions, i.e. this is shorthand for the record $\varnothing :\equiv \{\mathsf{B} :\equiv 1, \mathsf{E} :\equiv \lambda g.1, \mathsf{p} :\equiv \lambda g.\star\}$.

For a context $\Gamma : \mathsf{Ctx}$, the context $\natural\Gamma : \mathsf{Ctx}$ is defined by keeping the base the same but replacing the fibre with the point:

Definition 3.3.4 (Natural Context in Pointed Spaces)**.**

$$\begin{aligned}
\mathsf{B}(\natural\Gamma) &:\equiv \mathsf{B}\Gamma \\
\mathsf{E}(\natural\Gamma)(g) &:\equiv 1 \\
\mathsf{p}(\natural\Gamma)(g) &:\equiv \star
\end{aligned}$$

The unit substitution $\Gamma \vdash \eta : \natural\Gamma$ is given by

$$\begin{aligned}
\mathsf{B}\eta(g) &:\equiv g \\
\mathsf{E}\eta(g, g') &:\equiv \star \\
\mathsf{p}\eta(g) &:\equiv \mathsf{refl}_\star
\end{aligned}$$

i.e. it is the identity in the base, and sends everything in $\mathsf{E}\Gamma$ to the point, which in particular sends the sections of Γ to the point. Because the fibres are pointed, there is also a counit $\natural\Gamma \vdash \varepsilon : \Gamma$

$$\begin{aligned}
\mathsf{B}\varepsilon(g) &:\equiv g \\
\mathsf{E}\varepsilon(g, \star) &:\equiv \mathsf{p}\Gamma(g) \\
\mathsf{p}\varepsilon(g, \star) &:\equiv \mathsf{refl}_{\mathsf{p}\Gamma(g)}
\end{aligned}$$

i.e. it is again the identity on the base, and sends the point in the fibre to the chosen point in each fibre.

For a context Γ (thought of as a closed type), the context $\natural\Gamma$ is contractible in this model when $\mathsf{B}\Gamma$ is a contractible type. A $\natural$-connected context is up to equivalence $\{\mathsf{B}\Gamma :\equiv 1, \mathsf{E}\Gamma : 1 \to \mathcal{U}, \mathsf{p}\Gamma : \mathsf{E}\Gamma(\star)\}$ and thus is exactly a pointed type. Moreover, the substitutions between them are exactly pointed maps, with preservation of the point given by the third component.

Supposing that Γ as a context contains a single type A, we are used to thinking of substitutions $\varnothing \vdash \theta : A$ as corresponding to 'elements of A'. In our setting, however, if A is a $\natural$-connected type then the type of such substitutions is contractible — there is a unique pointed map from 1 to A that sends the point in the fibre to the point of A. Thus, substitutions from $\varnothing$ do not capture the elements of the *fibre* of A. To access the elements of the fibre, we can instead consider substitutions from $\mathbb{B} \vdash \theta : A$, where $\mathbb{B}$ is the booleans over the unit type:

Definition 3.3.5.

$$\begin{aligned} \mathsf{B}\mathbb{B} &:\equiv 1 \\ \mathsf{E}\mathbb{B}(\star) &:\equiv 2 \\ \mathsf{p}\mathbb{B}(\star) &:\equiv \mathsf{true} \end{aligned}$$

Substitutions $\mathbb{B} \vdash \theta : A$ correspond to pointed maps from the booleans into $\mathsf{E}A$. Such a pointed map sends true to the point, but has one remaining degree of freedom — where it sends false. Thus, the substitutions from $\mathbb{B}$ into a $\natural$-connected type A are equivalent to $\mathsf{E}A(\star)$. We use an analogue of $\mathbb{B}$ in Section 2.1.

The fact that substitutions $\varnothing \vdash \theta : A$ 'miss' data from A does not contradict the equivalence of types $A \simeq (1 \to A)$: we will see that the interpretation of function types does capture the fibre of A in the fibre of the function type.

3.3.2 Types and Terms

Once we have the definition of a substitution, one can determine what the data of a 'type in context' should be: the data of an arbitrary substitution into that context $A \vdash \mathsf{pr} : \Gamma$. In the present case a type '$\Gamma \vdash A$ type' unwinds an element of the record type

Definition 3.3.6 (Dependent Type in the Pointed Spaces Model)**.**

$$\begin{aligned} \{&\mathsf{B} : \mathsf{B}\Gamma \to \mathcal{U}, \\ &\mathsf{E} : \textstyle\prod_{(g:\mathsf{B}\Gamma)} \mathsf{B}(g) \to \mathsf{E}\Gamma(g) \to \mathcal{U}, \\ &\mathsf{p} : \textstyle\prod_{(g:\mathsf{B}\Gamma)} \prod_{(a:\mathsf{B}(g))} \mathsf{E}(g, a, \mathsf{p}\Gamma(g))\} \end{aligned}$$

The base of a type A depends only on the base of the context, while the fibre depends on both the base and the fibre of the context, and the base of the type. Interestingly, not every fibre of the $\mathsf{E}A$ family has a specified point, only the fibres that additionally lie over the basepoint of $\mathsf{E}\Gamma$.

The data of a type in context can arranged into the following diagram.

$$\begin{tikzcd}
\sum_{(g:\mathsf{B}\Gamma)} \sum_{(a:\mathsf{B}A(g))} \sum_{(e:\mathsf{E}\Gamma(g))} \mathsf{E}A(g,a,e) \arrow[r] \arrow[d, shift left] & \sum_{(g:\mathsf{B}\Gamma)} \mathsf{E}\Gamma(g) \arrow[d, shift left] \\
\sum_{(g:\mathsf{B}\Gamma)} \mathsf{B}A(g) \arrow[r] \arrow[u, bend left] & \mathsf{B}\Gamma \arrow[u, bend left]
\end{tikzcd}$$

where the upwards maps are sections definable from $\mathsf{p}\Gamma$ and $\mathsf{p}A$.

This section/retraction on the left of the diagram is exactly the interpretation of context extension '$\Gamma.A$ ctx', and the horizontal maps describe the projection substitution $\Gamma.A \vdash \mathsf{pr}_A : \Gamma$

Definition 3.3.7 (Context Extension in Pointed Spaces)**.** The context extension '$\Gamma.A$ ctx' is defined by

$$\begin{aligned}
\mathsf{B}(\Gamma.A) &:\equiv \textstyle\sum_{(g:\mathsf{B}\Gamma)} \mathsf{B}A(g) \\
\mathsf{E}(\Gamma.A)(g,a) &:\equiv \textstyle\sum_{(e:\mathsf{E}\Gamma(g))} \mathsf{E}A(g,a,e) \\
\mathsf{p}(\Gamma.A)(g,a) &:\equiv (\mathsf{p}\Gamma(g), \mathsf{p}A(g,a))
\end{aligned}$$

A 'term in context' must be the data of a section of the projection substitution '$\Gamma.A \to \Gamma$' shown in the diagram above — this is now a section in the *horizontal* direction. Building the section coherence in via dependency gives

Definition 3.3.8 (Terms in Pointed Spaces)**.** A term '$\Gamma \vdash t : A$' is an element of the record type

$$\begin{aligned}
\{&\mathsf{b} : \textstyle\prod_{(g:\mathsf{B}\Gamma)} \mathsf{B}A(g) \\
&\mathsf{e} : \textstyle\prod_{(g:\mathsf{B}\Gamma)} \prod_{(e:\mathsf{E}\Gamma(g))} \mathsf{E}A(g, \mathsf{b}(g), e) \\
&\mathsf{p} : \textstyle\prod_{(g:\mathsf{B}\Gamma)} \mathsf{e}(g, \mathsf{p}\Gamma(g)) =_{\mathsf{E}A(g,\mathsf{b}(g),\mathsf{p}\Gamma(g))} \mathsf{p}A(g, \mathsf{b}(g))\}
\end{aligned}$$

For $\natural\Gamma \vdash A$ type, we can define the natural type $\Gamma \vdash \natural A$ type by replacing the fibres with the unit type:

Definition 3.3.9 (Natural Type in Pointed Spaces)**.**

$$\begin{aligned}
\mathsf{B}(\natural A)(g) &:\equiv \mathsf{B}A(g) \\
\mathsf{E}(\natural A)(g,a,e) &:\equiv 1 \\
\mathsf{E}(\natural A)(g,a) &:\equiv \star
\end{aligned}$$

Note that this definition only uses $g : \mathsf{B}\Gamma$ and not the fibre or the section of Γ, so the definition is independent of whether we ask for an A that depends on Γ or on $\natural\Gamma$.

3.3.3 Ordinary Type Constructors

This internal model supports both dependent sums and dependent products satisfying the expected equations propositionally. Our goal here is to demonstrate how these type formers vary over the data of the context, so we instead describe non-dependent $\times$-types and $\to$-types, but the analogous definitions replacing $\times$ with Σ and $\to$ with Π give the dependent versions.

Product types are easy: we form the product both downstairs and upstairs. The pairing and projection terms are defined using the pairing and projection of the underlying $\times$-types.

Definition 3.3.10 (Product Types in Pointed Spaces)**.** For two types '$\Gamma \vdash A$ type' and '$\Gamma \vdash B$ type' define '$\Gamma \vdash A \times B$ type' by

$$\begin{aligned}
\mathsf{B}(A \times B)(g) &:\equiv \mathsf{B}A(g) \times \mathsf{B}B(g) \\
\mathsf{E}(A \times B)(g,(a,b),e) &:\equiv \mathsf{E}A(g,a,e) \times \mathsf{E}B(g,b,e) \\
\mathsf{p}(A \times B)(g,(a,b)) &:\equiv (\mathsf{p}A(g,a), \mathsf{p}B(g,b))
\end{aligned}$$

Function types are more interesting. The base of $A \to B$ is not the type of functions $\mathsf{B}A \to \mathsf{B}B$, but rather the entire type of substitutions $\Gamma.A$ to $\Gamma.B$ over Γ. For a fixed $g : \mathsf{B}\Gamma$, the data of such a substitution unwinds to triples

$$\begin{aligned}
\mathsf{Arr}(A,B)(g : \mathsf{B}\Gamma) :\equiv \{&\mathsf{b} : \mathsf{B}A(g) \to \mathsf{B}B(g) \\
&\mathsf{e} : \textstyle\prod_{(a:\mathsf{B}A(g))} \mathsf{E}A(g,a,\mathsf{p}\Gamma(g)) \to \mathsf{E}B(g,\mathsf{b}(a),\mathsf{p}\Gamma(g)) \\
&\mathsf{p} : \textstyle\prod_{(a:\mathsf{B}A(g))} \mathsf{e}(a,\mathsf{p}A(g,a)) = \mathsf{p}B(g,\mathsf{b}(a))\}
\end{aligned}$$

Definition 3.3.11 (Function Types in Pointed Spaces)**.** For two types '$\Gamma \vdash A$ type' and '$\Gamma \vdash B$ type' define '$\Gamma \vdash A \to B$ type' by

$$\begin{aligned}
\mathsf{B}(A \to B)(g) &:\equiv \mathsf{Arr}(A,B)(g) \\
\mathsf{E}(A \to B)(g,(\mathsf{b}f,\mathsf{e}f,\mathsf{p}f),e) &:\equiv \textstyle\prod_{(a:\mathsf{B}A(g))} \mathsf{E}A(g,a,e) \to \mathsf{E}B(g,\mathsf{b}f(a),e) \\
\mathsf{p}(A \to B)(g,(\mathsf{b}f,\mathsf{e}f,\mathsf{p}f)) &:\equiv \mathsf{e}f
\end{aligned}$$

This definition makes it clear that the $\natural$ modality does *not* preserve Π-types: the base of $\natural(A \to B)$ is $\mathsf{Arr}(A,B)$, while the base of $(\natural A \to \natural B)$ is essentially $\mathsf{B}A \to \mathsf{B}B$ (because the fibres of $\natural B$ are 1, the rest of the data of $\mathsf{Arr}(\natural A, \natural B)$ is determined).

We can also define identity types and the universe. Similar to Σ-types, Id-types are given component-wise. The type '$a = a'$' has as its base paths in the base of 'A', with the family over a path $p : \mathsf{B}(a)(g) = \mathsf{B}(a')(g)$ given by dependent paths in $\mathsf{E}A(g,-,e)$ that lie over it.

Definition 3.3.12 (Identity Types in Pointed Spaces)**.** For '$\Gamma \vdash a : A$' and '$\Gamma \vdash a' : A$', define '$\Gamma \vdash (a = a')$ type' by

$$\begin{aligned}
\mathsf{B}(a = a')(g) &:\equiv (\mathsf{B}(a)(g) = \mathsf{B}(a')(g)) \\
\mathsf{E}(a = a')(g,p,e) &:\equiv (\mathsf{E}(a)(g,e) =^{\mathsf{ap}_{\mathsf{E}A(g,-,e)}(p)} \mathsf{E}(a')(g,e)) \\
\mathsf{p}(a = a')(g,p) &:\equiv \mathsf{p}(a)(g) \cdot \mathsf{apd}_{\mathsf{p}A(g,\mathsf{B}B(-)(g))}(p) \cdot \mathsf{p}(a')(g)^{-1}
\end{aligned}$$

The types of the paths used to define the basepoint of each family above are

$$\begin{aligned}
\mathsf{p}(a)(g) &: \mathsf{E}(a)(g,\mathsf{p}\Gamma(g)) = \mathsf{p}A(g,\mathsf{B}(a)(g)) \\
\mathsf{apd}_{\mathsf{p}A(g,\mathsf{B}B(-)(g))}(p) &: \mathsf{p}A(g,\mathsf{B}(a)(g)) =^{\mathsf{ap}_{\mathsf{E}A(g,-,e)}(p)} \mathsf{p}A(g,\mathsf{B}(a')(g)) \\
\mathsf{p}(a')(g)^{-1} &: \mathsf{p}A(g,\mathsf{B}(a')(g)) = \mathsf{E}(a')(g,\mathsf{p}\Gamma(g))
\end{aligned}$$

Finally, the universe has as its base the entire type of pointed families, and as its fibres, the type of unpointed families over the same base.

Definition 3.3.13 (The Universe in Pointed Spaces)**.** The universe '$\varnothing \vdash \mathcal{U}$ type' is given by

$$\begin{aligned}
\mathsf{B}(\mathcal{U})(\star) &:\equiv \textstyle\sum_{(B:\mathcal{U})}\sum_{(E:B\to\mathcal{U})}\prod_{(b:B)} E(b) \\
\mathsf{E}(\mathcal{U})(\star,(B,E,p),\star) &:\equiv B \to \mathcal{U} \\
\mathsf{p}(\mathcal{U})(\star,(B,E,p)) &:\equiv E
\end{aligned}$$

3.3.4 Linear Type Formers

The linear type formers finally use the monoidal structure of pointed types. Recall that pointed types are monoidal for the *smash* product: the cofibre of the wedge inclusion $(A, a_0) \wedge (B, b_0) :\equiv (A, a_0) \vee (B, b_0) \to A \times B$. We write $(a, b) : A \wedge B$ implicitly including the pair into the cofibre. The smash product $A \wedge B$ is pointed by (a_0, b_0).

Definition 3.3.14 (Tensor Types in Pointed Spaces)**.** For two types '$\natural\Gamma \vdash A$ type' and '$\natural\Gamma \vdash B$ type' define '$\natural\Gamma \vdash A \otimes B$ type' by

$$\begin{aligned}
\mathsf{B}(A \otimes B)(g) &:\equiv \mathsf{B}A(g) \times \mathsf{B}B(g) \\
\mathsf{E}(A \otimes B)(g,(a,b),\star) &:\equiv (\mathsf{E}A(g,a,\star), \mathsf{p}A(g,a)) \wedge (\mathsf{E}B(g,b,\star), \mathsf{p}B(g,b)) \\
\mathsf{p}(A \otimes B)(g,(a,b)) &:\equiv (\mathsf{p}A(g,a), \mathsf{p}B(g,b))
\end{aligned}$$

This definition can only be made because the fibres of $\natural\Gamma$ are trivial. If instead we have arbitrary $e : \mathsf{E}\Gamma$, then we are stuck: in general we do not have points $? : \mathsf{E}A(g,a,e)$ and $? : \mathsf{E}B(g,b,e)$ to use to form the smash product.

It is not difficult to see how the dependent tensor (with dependency still mediated by $\natural$) may be implemented. The formulation is slightly awkward, because of the need to also apply $\natural$ to the type A in the context.

Definition 3.3.15 (Dependent Tensor Types in Pointed Spaces)**.** For two types '$\natural\Gamma \vdash A$ type' and '$\natural(\natural\Gamma).A \vdash B$ type' define '$\natural\Gamma \vdash A \otimes B$ type' by

$$\begin{aligned}
\mathsf{B}(A \otimes B)(g) &:\equiv \textstyle\sum_{(a:\mathsf{B}A(g))} \mathsf{B}B(g,a) \\
\mathsf{E}(A \otimes B)(g,(a,b),\star) &:\equiv \mathsf{E}A(g,a,\star) \wedge \mathsf{E}B((g,a),b,\star) \\
\mathsf{p}(A \otimes B)(g,(a,b)) &:\equiv ((\mathsf{p}A)(g,a), (\mathsf{p}B)((g,a),b))
\end{aligned}$$

The (non-dependent) $\multimap$-type then simply uses the type of pointed functions for each fibre, following the format of Proposition 2.1.17:

Definition 3.3.16 (Hom Types in Pointed Spaces)**.** For two types '$\natural\Gamma \vdash A$ type' and '$\natural\Gamma \vdash B$ type', define '$\Gamma \vdash A \multimap B$ type' by

$$\begin{aligned}
\mathsf{B}(A \multimap B)(g) &:\equiv \textstyle\prod_{(a:\mathsf{B}A(g))} \mathsf{B}B(g) \\
\mathsf{E}(A \multimap B)(g,f,e) &:\equiv \textstyle\prod_{(a:\mathsf{B}A(g))} \mathsf{E}A(g,a,\star) \to_\star \mathsf{E}B(g,f(a),\star) \\
\mathsf{p}(A \multimap B)(g,f) &:\equiv \lambda a.\mathsf{const}_{\star_{\mathsf{E}B(g,f(a),\star)}}
\end{aligned}$$

so that the specified point is the hom that is constantly the base point of each $\mathsf{E}B$.

Chapter 4

Outlook

4.1 Type Theory

The type theory presented in this thesis is the first example of a type theory that successfully combines four incompatible seeming elements: dependency, linearity, informal internal reasoning, and a homotopical interpretation. We hope it will not be the last!

We anticipate that the fibrational framework [LSR17; LRS22] (once appropriately extended to dependent types), will be able to capture the rules of our theory, if not useful strict equalities on raw syntax. Our $\otimes$-type can be seen as an example of a 'binary' modality that reifies the structure of contexts; exactly the kind of situation that the fibrational framework is designed to capture.

This particular syntactic presentation, however, feels to the author like somewhat of a dead-end from an extensibility standpoint. The syntax, all though it feels very nice to use informally, is quite brittle: As we try to make clear in Section 3.1, there are quite a lot of innocent changes or simplifications to the theory that have wide-ranging consequences. The only 'major' extensions to this type theory that the author is comfortable to claim are possible, are additional monoidal products (with no structural rules, like $\otimes$). This is not likely to be very useful!

Unpointed Linear Types. Our syntax builds the fact that the linear types have a zero object into its foundations, in particular, this is the reason that marked variable usage is available. This matches nicely with our intended models and makes the rules for modality $\natural$ very pleasant. Knowing that there is a zero object allows us to formulate $\natural$ as a negative type former, adjoint to itself, and furnishes us with the map $\natural A \to A$ that would not otherwise exist.

However, this assumption does restrict the potential models of the theory. One can form the category of parameterised objects for any category $\mathcal{C}$, and in favourable cases the result is an ∞-topos. In general, the self-adjointness coincidence of $\natural$ disappears and the category $P\mathcal{C}$ admits an adjoint quadruple

$$\begin{tikzcd}
P\mathcal{C} \arrow[d, shift right=1, "U"'] \arrow[d, shift left=3, "\Gamma"] \\
\mathsf{Set} \arrow[u, shift left=3, "0"] \arrow[u, shift right=1, "1"']
\end{tikzcd}$$

where Γ denotes the global sections functor, U denotes the underlying set functor, and 0 and 1 represent the constant family functors on the initial and terminal objects of $\mathcal{C}$ respectively. Unfortunately this does not line up with the adjoint quadruple that defines cohesive toposes setup [Law07; Shu18]: the global sections functor here is the *rightmost* adjoint. A hypothetical type theory that captures this situation therefore have $\int :\equiv 1 \circ U$ and $\flat :\equiv 1 \circ \Gamma$, but no modality corresponding to $\sharp$. The $\int$ modality would mediate dependency, like the $\natural$ mediates dependency in the present theory.

Such a type theory would have a wealth of models by taking $\mathcal{C}$ to be $\mathrm{Psh}(\mathcal{M})$ for any symmetric monoidal category $\mathcal{M}$. The presheaf topos $\mathrm{Psh}(\mathcal{M})$ is always a symmetric monoidal closed category, with Day convolution as the monoidal product [Day70]. The model is then in set-indexed families of these presheaves $P(\mathrm{Psh}(\mathcal{M}))$, which admits a fully faithful monoidal inclusion $\mathcal{M} \hookrightarrow P(\mathrm{Psh}(\mathcal{M}))$. This two-step construction would admit a more expressive type theory than working in directly in the monoidal category $\mathrm{Psh}(\mathcal{M})$, as we explain in Remark 1.3.15. A hope is that we could use the type theory to reason about arbitrary symmetric monoidal categories, even if the original $\mathcal{M}$ has no other structure, perhaps as an alternative to [Shu21]. This family of models is the similar to the one targeted by the Proto-Quipper-M [FKS20] variant of Quantitative Type Theory.

In an unpointed version, some aspects of the type theory simplify: for example, there is no need for marked variable uses or the related admissible operations on terms. But some aspects become more complicated: we can no longer assume that $1 \otimes 1 \simeq 1$, and so the syntax of palettes and slices will at the very least need to count how many instances of 1 have accumulated!

The rules for $\int$, as a left-adjoint monadic modality, would involve a kind of 'locking' of variables in the context, in the style of [Clo18, §5]. It remains to be seen how this would interact with the colour labels:.

4.2 Synthetic Homotopy Theory

There is far more synthetic stable homotopy theory that may be internalisable in this type theory. The most obvious next step is to investigate synthetic homology and cohomology:

Definition 4.2.1. For a dull pointed space $\underline{X}$ and reduced type $\underline{E}$, the *nth homology and cohomology of $\underline{X}$ with coefficients in $\underline{E}$* are defined by

$$\underline{E}_n(\underline{X}) :\equiv \pi^s_n(\Sigma^\infty(\underline{X}) \otimes \underline{E})$$
$$\underline{E}^n(\underline{X}) :\equiv \pi^s_n(\Sigma^\infty(\underline{X}) \multimap \underline{E})$$

Using Axiom S, we expect these groups to form a '(co)homology theory', in the sense of the Eilenberg-Steenrod axioms (see [Cav15, §3] for a type-theoretic description) for any reduced type $\underline{E}$. To recover ordinary cohomology, we need to define a reduced type '$H\mathbb{Z}$' that satisfies $\pi^s_0(H\mathbb{Z}) = \mathbb{Z}$ and $\pi^s_n(H\mathbb{Z}) = 0$ for $n \neq 0$. Axiom N implies that $\pi^s_0\mathbb{S} = \mathbb{Z}$, so we expect to form such an $H\mathbb{Z}$ by *stably truncating* $\mathbb{S}$ to remove the stable homotopy groups above dimension 0. We distinguish this from ordinary type-theoretic truncation, which removes the *ordinary* homotopy groups of a type. This stable truncation operation would be specified by a higher inductive type: nullification at $\Sigma\mathbb{S}$. It may also be possible to define $H\mathbb{Z}/2$ internally, and work with the Steenrod operations as maps $H\mathbb{Z}/2 \to H\mathbb{Z}/2$. We leave this promising line for future work.

Bibliography

[AB20] Andreas Abel and Jean-Philippe Bernardy. "A Unified View of Modalities in Type Systems". In: *Proceedings of the 25th ACM SIGPLAN International Conference on Functional Programming*. ICFP '20. Jersey City, USA: Association for Computing Machinery, 2020. DOI: 10.1145/3408972.

[Abe15] Andreas Abel. "The Next 700 Modal Type Assignment Systems". In: *21st International Conference on Types for Proofs and Programs*. Ed. by Tarmo Uustalu. Vol. 69. TYPES '15. Tallinn, Estonia: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2015. ISBN: 978-3-95977-030-9.

[ABFJ20] Mathieu Anel, Georg Biedermann, Eric Finster, and André Joyal. "A generalized Blakers-Massey theorem". In: *Journal of Topology* 13.4 (2020), pp. 1521–1553. ISSN: 1753-8416. DOI: 10.1112/topo.12163.

[ABG18] Matthew Ando, Andrew Blumberg, and David Gepner. "Parametrized Spectra, Multiplicative Thom Spectra and the Twisted Umkehr Map". In: *Geometry & Topology* 22.7 (Dec. 2018), pp. 3761–3825. ISSN: 1465-3060. DOI: 10.2140/gt.2018.22.3761.

[ACDKN18] Thorsten Altenkirch, Paolo Capriotti, Gabe Dijkstra, Nicolai Kraus, and Fredrik Nordvall Forsberg. "Quotient Inductive-Inductive Types". In: *21st International Conference on Foundations of Software Science and Computation Structures*. Vol. 10803. FOSSACS '18. Springer International Publishing, 2018, pp. 293–310. ISBN: 978-3-319-89366-2. DOI: 10.1007/978-3-319-89366-2_16.

[AK11] Peter Arndt and Krzysztof Kapulkin. "Homotopy-Theoretic Models of Type Theory". In: *Typed lambda calculi and applications*. Vol. 6690. Lecture Notes in Computer Science. Springer, Heidelberg, 2011, pp. 45–60. DOI: 10.1007/978-3-642-21691-6_7.

[AMPR01] Natasha Alechina, Michael Mendler, Valeria de Paiva, and Eike Ritter. "Categorical and Kripke semantics for constructive S4 modal logic". In: *Computer Science Logic*. Vol. 2142. Lecture Notes in Computer Science. Springer, Berlin, 2001, pp. 292–307. DOI: 10.1007/3-540-44802-0_21.

[APTS13] Andreas Abel, Brigitte Pientka, David Thibodeau, and Anton Setzer. "Copatterns: Programming Infinite Structures by Observations". In: *Proceedings of the 40th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*. Ed. by Roberto Giacobazzi and Radhia Cousot. POPL '13. Rome, Italy: Association for

Computing Machinery, 2013, pp. 27–38. ISBN: 978-1-4503-1832-7. DOI: 10.1145/2429069.2429075.

[Atk04] Robert Atkey. "A λ-Calculus for Resource Separation". In: *Automata, languages and programming*. Vol. 3142. Lecture Notes in Computer Science. Springer, Berlin, 2004, pp. 158–170. DOI: 10.1007/978-3-540-27836-8_16.

[Atk06] Robert Atkey. "Substructural Simple Type Theories for Separation and In-place Update". PhD thesis. University of Edinburgh, 2006. URL: http://hdl.handle.net/1842/892.

[Atk18] Robert Atkey. "Syntax and Semantics of Quantitative Type Theory". In: *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science*. LICS '18. Oxford, United Kingdom: Association for Computing Machinery, 2018, pp. 56–65. ISBN: 978-1-4503-5583-4. DOI: 10.1145/3209108.3209189.

[AW09] Steve Awodey and Michael A. Warren. "Homotopy theoretic models of identity types". In: *Mathematical Proceedings of the Cambridge Philosophical Society* 146.1 (2009), pp. 45–55. ISSN: 0305-0041. DOI: 10.1017/S0305004108001783.

[AW18] Robert Atkey and James Wood. "Context Constrained Computation (Extended Abstract)". In: *3rd Workshop on Type-Driven Development*. TyDe '18. 2018.

[Bar96] Andrew Barber. *Dual Intuitionistic Linear Logic*. Tech. rep. ECS-LFCS-96-347. University of Edinburgh, 1996. URL: https://www.lfcs.inf.ed.ac.uk/reports/96/ECS-LFCS-96-347/.

[BCMEPS20] Lars Birkedal, Ranald Clouston, Bassel Mannaa, Rasmus Ejlers Møgelberg, Andrew M. Pitts, and Bas Spitters. "Modal dependent type theory and dependent right adjoints". In: *Mathematical Structures in Computer Science* 30.2 (2020), pp. 118–138. DOI: 10.1017/S0960129519000197.

[BG13] Jean-Philippe Bernardy and Moulin Guilhem. "Type-Theory in Color". In: *Proceedings of the 18th ACM SIGPLAN International Conference on Functional Programming*. ICFP '13. Boston, Massachusetts, USA: Association for Computing Machinery, 2013, pp. 61–72. ISBN: 978-1-4503-2326-0. DOI: 10.1145/2500365.2500577.

[BGMZ14] Aloïs Brunel, Marco Gaboardi, Damiano Mazza, and Steve Zdancewic. "A Core Quantitative Coeffect Calculus". In: *Programming Languages and Systems*. Ed. by Zhong Shao. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, pp. 351–370. ISBN: 978-3-642-54833-8. DOI: 10.1007/978-3-642-54833-8_19.

[BK19] Rafael Bocquet and András Kovács. *Deriving a Frobenius variant of J from J*. 2019. URL: https://github.com/akaposi/hiit-signatures/blob/master/formalization/FrobeniusJDeriv.agda.

[BO06] Josh Berdine and Peter W. O'Hearn. "Strong Update, Disposal, and Encapsulation in Bunched Typing". In: *Proceedings of the 22nd Annual Conference on Mathematical Foundations of Programming Semantics*. Vol. 158. MFPS XXII. Jan. 2006, pp. 81–98. DOI: 10.1016/j.entcs.2006.04.006.

[Boe20] Menno de Boer. "A Proof and Formalization of the Initiality Conjecture of Dependent Type Theory". Licentiate Thesis. Stockholm University, 2020. URL: http://su.diva-portal.org/smash/record.jsf?pid=diva2%5C%3A1431287&dswid=-3494.

[Bru16] Guillaume Brunerie. "On the homotopy groups of spheres in homotopy type theory". PhD thesis. Université de Nice, 2016. arXiv: 1606.05916 [math.AT].

[Buc19] Ulrik Buchholtz. "Universes in toy models of spatial type theory". In: *Workshop on Geometry in Modal Homotopy Type Theory*. Carnegie Mellon University, 2019. URL: https://www.youtube.com/watch?v=KpwFDVMxnsM.

[BW96] Nick Benton and Philip Wadler. "Linear logic, monads and the lambda calculus". In: *Proceedings of the 11th Annual IEEE Symposium on Logic in Computer Science*. LICS '96. IEEE Computer Society, 1996, pp. 420–431. ISBN: 978-0-8186-7463-1. DOI: 10.1109/LICS.1996.561458.

[Cav15] Evan Cavallo. "Synthetic Cohomology in Homotopy Type Theory". MA thesis. Carnegie Mellon University, Dec. 2015. URL: https://staff.math.su.se/evan.cavallo/works/thesis15.pdf.

[CD14] Pierre Clairambault and Peter Dybjer. "The biequivalence of locally cartesian closed categories and Martin-Löf type theories". In: *Mathematical Structures in Computer Science* 24.6 (2014), e240606. ISSN: 0960-1295. DOI: 10.1017/S0960129513000881.

[Che09] James Cheney. "A Simple Nominal Type Theory". In: *Proceedings of the International Workshop on Logical Frameworks and Metalanguages: Theory and Practice (LFMTP 2008)*. Vol. 228. 2009, pp. 37–52. DOI: 10.1016/j.entcs.2008.12.115.

[Che12] James Cheney. "A dependent nominal type theory". In: *Logical Methods in Computer Science* 8.1 (2012), 1:08, 29. DOI: 10.2168/LMCS-8(1:8)2012.

[Clo18] Ranald Clouston. "Fitch-Style Modal Lambda Calculi". In: *21st International Conference on Foundations of Software Science and Computation Structures*. Vol. 10803. FOSSACS '18. Springer International Publishing, 2018, pp. 258–275. ISBN: 978-3-319-89366-2. DOI: 10.1007/978-3-319-89366-2_14.

[Coq92] Thierry Coquand. "Pattern Matching with Dependent Types". In: *Types for Proofs and Programs*. 1992. URL: https://wonks.github.io/type-theory-reading-group/papers/proc92-coquand.pdf.

[CP96] Iliano Cervesato and Frank Pfenning. "A Linear Logical Framework". In: *Proceedings of the 11th Annual IEEE Symposium on Logic in Computer Science*. LICS '96. IEEE Computer Society, 1996, pp. 19–75. ISBN: 978-0-8186-7463-1. DOI: 10.1006/inco.2001.2951.

[CPR08] Matthew Collinson, David Pym, and Edmund Robinson. "Bunched polymorphism". In: *Mathematical Structures in Computer Science* 18.6 (2008), pp. 1091–1132. ISSN: 0960-1295. DOI: 10.1017/S0960129508007159.

[CRS20] Thierry Coquand, Fabian Ruch, and Christian Sattler. *Constructive sheaf models of type theory*. 2020. arXiv: 1912.10407 [math.LO].

[Day70] Brian Day. "On closed categories of functors". In: *Reports of the Midwest Category Seminar, IV*. Lecture Notes in Mathematics, Vol. 137. Springer, Berlin, 1970, pp. 1–38. DOI: 10.1007/BFb0060438.

[Dyb96] Peter Dybjer. "Internal type theory". In: *Types for Proofs and Programs*. Vol. 1158. Lecture Notes in Computer Science. Torino: Springer, Berlin, 1996, pp. 120–134. ISBN: 978-3-540-70722-6. DOI: 10.1007/3-540-61780-9_66.

[FHM03] H. Fausk, P. Hu, and J. P. May. "Isomorphisms between left and right adjoints". In: *Theory and Applications of Categories* 11 (2003), No. 4, 107–131 (electronic). ISSN: 1201-561X. URL: http://www.math.uchicago.edu/~may/PAPERS/FormalFinalMarch.pdf.

[FKRS20] Peng Fu, Kohei Kishida, Neil J. Ross, and Peter Selinger. "A Tutorial Introduction to Quantum Circuit Programming in Dependently Typed Proto-Quipper". In: *Reversible Computation*. Cham: Springer International Publishing, 2020, pp. 153–168. ISBN: 978-3-030-52482-1. DOI: 10.1007/978-3-030-52482-1_9.

[FKS20] Peng Fu, Kohei Kishida, and Peter Selinger. "Linear Dependent Type Theory for Quantum Programming Languages: Extended Abstract". In: *Proceedings of the 35th Annual ACM/IEEE Symposium on Logic in Computer Science*. LICS '20. Saarbrücken, Germany: Association for Computing Machinery, 2020, pp. 440–453. ISBN: 978-1-4503-7104-9. DOI: 10.1145/3373718.3394765.

[FOPTST99] P. J. Freyd, P. W. O'Hearn, A. J. Power, M. Takeyama, R. Street, and R. D. Tennent. "Bireflectivity". In: *Theoretical Computer Science* 228.1-2 (1999), pp. 49–76. ISSN: 0304-3975. DOI: 10.1016/S0304-3975(98)00354-5.

[GG08] Nicola Gambino and Richard Garner. "The identity type weak factorisation system". In: *Theoretical Computer Science* 409.1 (2008), pp. 94–109. ISSN: 0304-3975. DOI: 10.1016/j.tcs.2008.08.030.

[Gir87] Jean-Yves Girard. "Linear logic". In: *Theoretical Computer Science* 50.1 (1987), p. 101. ISSN: 0304-3975. DOI: 10.1016/0304-3975(87)90045-4.

[GK17] David Gepner and Joachim Kock. "Univalence in locally cartesian closed ∞-categories". In: *Forum Mathematicum* 29.3 (2017), pp. 617–652. ISSN: 0933-7741. DOI: 10.1515/forum-2015-0228.

[GKNB20] Daniel Gratzer, G. A. Kavvos, Andreas Nuyts, and Lars Birkedal. "Multimodal Dependent Type Theory". In: *Proceedings of the 35th Annual ACM/IEEE Symposium on Logic in Computer Science*. LICS '20. Saarbrücken, Germany: Association for Computing Machinery, 2020, pp. 492–506. ISBN: 978-1-4503-7104-9. DOI: 10.1145/3373718.3394736.

[GP01] Murdoch J. Gabbay and Andrew M. Pitts. "A New Approach to Abstract Syntax with Variable Binding". In: *Formal Aspects of Computing* 13.3–5 (July 2001), pp. 341–363. DOI: 10.1007/s001650200016.

[Gra18] Robert Graham. *Synthetic Homology in Homotopy Type Theory*. 2018. arXiv: 1706.01540 [math.LO].

[Gra21] Daniel Gratzer. *Crisp induction for intensional identity types*. 2021. URL: https://jozefg.github.io/papers/crisp-induction-for-intensional-identity-types.pdf.

[GSB19] Daniel Gratzer, Jonathan Sterling, and Lars Birkedal. "Implementing a modal dependent type theory". In: *Proceedings of the 24th ACM SIGPLAN International Conference on Functional Programming*. ICFP '19. Boston, Massachusetts, USA: Association for Computing Machinery, 2019, 107:1–107:29. DOI: 10.1145/3341711.

[GSS92] Jean-Yves Girard, Andre Scedrov, and Philip J. Scott. "Bounded linear logic: a modular approach to polynomial-time computability". In: *Theoretical Computer Science* 97.1 (1992), pp. 1–66. ISSN: 0304-3975. DOI: 10.1016/0304-3975(92)90386-T.

[Gug07] Alessio Guglielmi. "A system of interaction and structure". In: *ACM Transactions on Computational Logic* 8.1 (2007), Art. 1, 64. ISSN: 1529-3785. DOI: 10.1145/1182613.1182614.

[Hof97] Martin Hofmann. "Syntax and Semantics of Dependent Types". In: *Semantics and Logics of Computation*. Vol. 14. Cambridge University Press, 1997, pp. 79–130. DOI: 10.1017/CBO9780511526619.004.

[Hor06] Benjamin Robert Horsfall. "The Logic of Bunched Implications: A Memoir". MA thesis. The University of Melbourne, 2006. URL: http://hdl.handle.net/11343/39480.

[HoTTAgda] Guillaume Brunerie, Kuen-Bang Hou (Favonia), Evan Cavallo, Tim Baumann, Eric Finster, Jesper Cockx, Christian Sattler, Chris Jeris, Michael Shulman, et al. *Homotopy Type Theory in Agda*. URL: https://github.com/HoTT/HoTT-Agda.

[HoTTBook] The Univalent Foundations Program. *Homotopy Type Theory: Univalent Foundations of Mathematics*. Institute for Advanced Study: https://homotopytypetheory.org/book, 2013.

[HP93] Martin Hyland and Valeria de Paiva. "Full intuitionistic linear logic (extended abstract)". In: *Ann. Pure Appl. Logic* 64.3 (1993), pp. 273–291. ISSN: 0168-0072. DOI: 10.1016/0168-0072(93)90146-5.

[IP98] S. S. Ishtiaq and D. J. Pym. "A Relevant Analysis of Natural Deduction". In: *Journal of Logic and Computation* 8.6 (1998), pp. 809–838. ISSN: 0955-792X. DOI: 10.1093/logcom/8.6.809.

[IP99] Samin Ishtiaq and David J. Pym. "Kripke Resource Models of a Dependently-Typed Bunched λ-Calculus (Extended Abstract)". In: *Computer Science Logic*. Vol. 1683. Lecture Notes in Computer Science. Springer, Berlin, 1999, pp. 235–249. DOI: 10.1007/3-540-48168-0_17.

[Isa20] Valery Isaev. "Indexed type theories". In: *Homotopy Type Theory Electronic Seminar Talks*. 2020. URL: https://www.youtube.com/watch?v=PzjmjIlc-kE.

[Isa21] Valery Isaev. "Indexed type theories". In: *Mathematical Structures in Computer Science* 31.1 (2021), pp. 3–63. DOI: 10.1017/S0960129520000092.

[Joy08] André Joyal. *Notes on Logoi*. 2008. URL: http://www.math.uchicago.edu/~may/IMA/JOYAL/Joyal.pdf.

[KL21] Krzysztof Kapulkin and Peter LeFanu Lumsdaine. "The simplicial model of Univalent Foundations (after Voevodsky)". In: *Journal of the European Mathematical Society* 23.6 (2021), pp. 2071–2126. ISSN: 1435-9855. DOI: 10.4171/JEMS/1050.

[KPB15] Neelakantan R. Krishnaswami, Pierre Pradic, and Nick Benton. "Integrating Linear and Dependent Types". In: *Proceedings of the 42nd Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*. POPL '15. Mumbai, India: Association for Computing Machinery, 2015, pp. 17–30. ISBN: 978-1-4503-3300-9. DOI: 10.1145/2676726.2676969.

[Kri11] Neel Krishnaswami. *A new lambda calculus for bunched implications*. July 27, 2011. URL: https://semantic-domain.blogspot.com/2011/07/new-lambda-calculus-for-bunched.html.

[KS17] Nicolai Kraus and Christian Sattler. *Space-Valued Diagrams, Type-Theoretically (Extended Abstract)*. 2017. arXiv: 1704.04543 [math.LO].

[Law07] F. William Lawvere. "Axiomatic Cohesion". In: *Theory and Applications of Categories* 19 (2007), No. 3, 41–49. URL: http://www.tac.mta.ca/tac/volumes/19/3/19-03abs.html.

[LRS22] Daniel R. Licata, Mitchell Riley, and Michael Shulman. "A Fibrational Framework for Substructural and Modal Dependent Type Theories". 2022. In preparation.

[LS20] Peter LeFanu Lumsdaine and Michael Shulman. "Semantics of higher inductive types". In: *Mathematical Proceedings of the Cambridge Philosophical Society* 169.1 (2020), pp. 159–208. ISSN: 0305-0041. DOI: 10.1017/s030500411900015x.

[LSR17] Daniel R. Licata, Michael Shulman, and Mitchell Riley. "A Fibrational Framework for Substructural and Modal Logics". In: *2nd International Conference on Formal Structures for Computation and Deduction*. Ed. by Dale Miller. Vol. 84. FSCD '17. Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2017. ISBN: 978-3-95977-047-7. URL: http://dlicata.web.wesleyan.edu/pubs/lsr17multi/lsr17multi-ex.pdf.

[Lum18] Peter LeFanu Lumsdaine. *What is known and/or written about "Frobenius eliminators"?* July 2018. URL: https://groups.google.com/g/homotopytypetheory/c/b96TBt0Bn7Y/m/IiAcnUL2AQAJ.

[Lun18] Martin Lundfall. *A diagram model of linear dependent type theory*. 2018. arXiv: 1806.09593 [math.LO].

[LW15] Peter LeFanu Lumsdaine and Michael A. Warren. "The Local Universes Model: An Overlooked Coherence Construction for Dependent Type Theories". In: *ACM Transactions on Computational Logic* 16.3 (2015), Art. 23, 31. ISSN: 1529-3785. DOI: 10.1145/2754931.

[LZH08] Daniel R. Licata, Noam Zeilberger, and Robert Harper. “Focusing on Binding and Computation”. In: *Proceedings of the Twenty-Third Annual IEEE Symposium on Logic in Computer Science*. LICS ’08. Pittsburgh, PA, USA: IEEE Computer Society, 2008, pp. 241–252. DOI: 10.1109/LICS.2008.48.

[McB16] Conor McBride. “I Got Plenty o’ Nuttin’”. In: *A list of successes that can change the world*. Vol. 9600. Lecture Notes in Computer Science. Springer International Publishing, 2016, pp. 207–233. DOI: 10.1007/978-3-319-30936-1_12.

[MEO21] Benjamin Moon, Harley Eades III, and Dominic Orchard. “Graded Modal Dependent Type Theory”. In: *Programming Languages and Systems*. Ed. by Nobuko Yoshida. Cham: Springer International Publishing, 2021, pp. 462–490. DOI: 10.1007/978-3-030-72019-3_17.

[nLaba] nLab authors. *Mathematics Presented in Homotopy Type Theory*. 2019. URL: https://ncatlab.org/nlab/show/mathematics+presented+in+homotopy+type+theory.

[nLabb] nLab authors. *Dependent linear type theory*. 2020. URL: https://ncatlab.org/nlab/show/dependent+linear+type+theory.

[NPP08] Aleksandar Nanevski, Frank Pfenning, and Brigitte Pientka. “Contextual modal type theory”. In: *ACM Transactions on Computational Logic* 9.3 (2008), Art. 23, 49. ISSN: 1529-3785. DOI: 10.1145/1352582.1352591.

[OHe03] Peter O’Hearn. “On bunched typing”. In: *Journal of Functional Programming* 13.4 (2003), pp. 747–796. ISSN: 0956-7968. DOI: 10.1017/S0956796802004495.

[OLE19] Dominic Orchard, Vilem-Benjamin Liepelt, and Harley Eades III. “Quantitative Program Reasoning with Graded Modal Types”. In: ICFP ’19 (2019). ISSN: 2475-1421. DOI: 10.1145/3341714.

[OP99] Peter W. O’Hearn and David J. Pym. “The Logic of Bunched Implications”. In: *Bulletin of Symbolic Logic* 5.2 (1999), pp. 215–244. ISSN: 1079-8986. DOI: 10.2307/421090.

[PD01] Frank Pfenning and Rowan Davies. “A judgmental reconstruction of modal logic”. In: *Mathematical Structures in Computer Science* 11 (2001), pp. 511–540. ISSN: 0960-1295. DOI: 10.1017/S0960129501003322.

[PMD15] Andrew M. Pitts, Justus Matthiesen, and Jasper Derikx. “A Dependent Type Theory with Abstractable Names”. In: *Ninth Workshop on Logical and Semantic Frameworks, with Applications (LSFA 2014)*. Vol. 312. Electronic Notes in Theoretical Computer Science. Elsevier Sci. B. V., Amsterdam, 2015, pp. 19–50. DOI: 10.1016/j.entcs.2015.04.003.

[POM14] Tomas Petricek, Dominic Orchard, and Alan Mycroft. “Coeffects: A Calculus of Context-Dependent Computation”. In: *Proceedings of the 19th ACM SIGPLAN International Conference on Functional Programming*. ICFP ’14. Gothenburg, Sweden: Association for Computing Machinery, 2014. ISBN: 978-1-4503-2873-9. DOI: 10.1145/2628136.2628160.

[PR16] Valeria de Paiva and Eike Ritter. “Fibrational Modal Type Theory”. In: *Proceedings of the Tenth Workshop on Logical and Semantic Frameworks, with Applications (LSFA 2015)*. Vol. 323. Electronic Notes in Theoretical Computer Science. 2016, pp. 143–161. DOI: 10.1016/j.entcs.2016.06.010.

[Pym02] D. J. Pym. *The Semantics and Proof Theory of the Logic of Bunched Implications*. Vol. 26. Applied Logic Series. Kluwer Academic Publishers, 2002. DOI: 10.1007/978-94-017-0091-7. Errata in *Errata and Remarks for The Semantics and Proof Theory of the Logic of Bunched Implications*. 2008. URL: http://www0.cs.ucl.ac.uk/staff/D.Pym/BI-monograph-errata.pdf.

[Ree09] Jason Reed. “A Judgmental Deconstruction of Modal Logic”. 2009. URL: http://www.cs.cmu.edu/~jcreed/papers/jdml.pdf.

[RFL21] Mitchell Riley, Eric Finster, and Daniel R. Licata. *Synthetic Spectra via a Monadic and Comonadic Modality*. 2021. arXiv: 2102.04099 [math.CT].

[Rij18] Egbert Rijke. “Classifying Types”. PhD thesis. Carnegie Mellon University, July 2018. arXiv: 1906.09435 [math.LO].

[RSS20] Egbert Rijke, Michael Shulman, and Bas Spitters. “Modalities in Homotopy Type Theory”. In: *Logical Methods in Computer Science* 16.1 (2020), Paper No. 2, 79.

[Sch06] Ulrich Schöpp. “Names and Binding in Type Theory”. PhD thesis. University of Edinburgh, 2006. URL: https://ethos.bl.uk/OrderDetails.do?uin=uk.bl.ethos.561934.

[Sch14] Urs Schreiber. *Quantization via Linear homotopy types*. 2014. arXiv: 1402.7041 [math-ph].

[Sch17] Urs Schreiber. *Differential cohomology in a cohesive infinity-topos*. 2017. URL: https://ncatlab.org/schreiber/files/dcct170811.pdf.

[SDR20] Kristina Sojakova, Floris van Doorn, and Egbert Rijke. “Sequential Colimits in Homotopy Type Theory”. In: *Proceedings of the 35th Annual ACM/IEEE Symposium on Logic in Computer Science*. LICS ’20. Saarbrücken, Germany: Association for Computing Machinery, 2020. ISBN: 978-1-4503-7104-9. DOI: 10.1145/3373718.3394801.

[SDRPS19] Michael Shulman, Floris van Doorn, Egbert Rijke, Stefano Piceghello, and Yuri Sulyma. *Spectral Sequences in Homotopy Type Theory*. 2019. URL: https://github.com/cmu-phil/Spectral/tree/master/spectrum.

[Shu08] Michael Shulman. “Framed bicategories and monoidal fibrations”. In: *Theory and Applications of Categories* 20 (2008), No. 18, 650–738. URL: http://www.tac.mta.ca/tac/volumes/20/18/20-18abs.html.

[Shu15] Michael Shulman. “The univalence axiom for elegant Reedy presheaves”. In: *Homology, Homotopy and Applications* 17.2 (2015), pp. 81–106. ISSN: 1532-0073. DOI: 10.4310/HHA.2015.v17.n2.a6.

[Shu18] Michael Shulman. “Brouwer’s fixed-point theorem in real-cohesive homotopy type theory”. In: *Mathematical Structures in Computer Science* 28.6 (2018), pp. 856–941. ISSN: 0960-1295. DOI: 10.1017/S0960129517000147.

[Shu19a] Michael Shulman. *All* $(\infty, 1)$*-toposes have strict univalent universes*. 2019. arXiv: 1904.07004 [math.AT].

[Shu19b] Michael Shulman. *The Generalized Blakers-Massey Theorem*. The HoTT Library. 2019. URL: https://github.com/HoTT/HoTT/blob/master/theories/Homotopy/BlakersMassey.v.

[Shu21] Michael Shulman. “A practical type theory for symmetric monoidal categories”. In: *Theory and Applications of Categories* 37 (2021), pp. 863–907. URL: http://www.tac.mta.ca/tac/volumes/37/25/37-25abs.html.

[SS04] Ulrich Schöpp and Ian Stark. “A Dependent Type Theory with Names and Binding”. In: *Computer Science Logic*. Vol. 3210. Lecture Notes in Computer Science. Springer, Berlin, 2004, pp. 235–249. DOI: 10.1007/978-3-540-30124-0_20.

[Str91] Thomas Streicher. *Semantics of Type Theory*. Progress in Theoretical Computer Science. Birkhäuser Boston, Inc., Boston, MA, 1991, pp. xii+298. ISBN: 0-8176-3594-7. DOI: 10.1007/978-1-4612-0433-6.

[Vák14] Matthjis Vákár. *Syntax and Semantics of Linear Dependent Types*. 2014. arXiv: 1405.0033 [cs.AT].

[Vák15] Matthijs Vákár. “A Categorical Semantics for Linear Logical Frameworks”. In: *18th International Conference on Foundations of Software Science and Computation Structures*. Ed. by Andrew Pitts. FOSSACS ’15. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, pp. 102–116. ISBN: 978-3-662-46678-0. DOI: 10.1007/978-3-662-46678-0_7.

[van18] Floris van Doorn. “On the Formalization of Higher Inductive Types and Synthetic Homotopy Theory”. PhD thesis. 2018. arXiv: 1808.10690 [math.AT].

[WA20] James Wood and Robert Atkey. “A Linear Algebra Approach to Linear Metatheory”. In: *Linearity/TLLA ’20*. 2020. arXiv: 2005.02247 [cs.PL].

[WCPW03] Kevin Watkins, Iliano Cervesato, Frank Pfenning, and David Walker. *A concurrent logical framework I: Judgments and properties*. Tech. rep. CMU-CS-02-101. Carnegie Mellon University, 2003. URL: https://www.cs.cmu.edu/~fp/papers/CMU-CS-02-101.pdf.

[Zei05] Noam Zeilberger. “Modal BI and Separation Logic”. 2005. URL: http://www.cs.cmu.edu/~noam/papers/s4bi.pdf.

Appendix A

Complete Rules of the Theory

Judgements for palettes:

- Φ palette
- $\Phi \vdash \mathfrak{c}$ colour
- $\Phi \vdash s$ preslice
- $\Phi \vdash s$ preslice$_\epsilon$
- $\Phi \vdash s$ slice
- $\Phi \vdash s_L \boxtimes s_R$ presplit
- $\Phi \vdash s_L \boxtimes s_R$ split
- $\Phi \vdash i$ unit
- $\Phi \vdash \kappa : \Psi$

Judgements for contexts, types and terms:

- $\Phi \mid \Gamma$ ctx
- $\Phi \mid \Gamma \vdash A$ type
- $\Phi \mid \Gamma \vdash a : A$
- $\Phi \mid \Gamma \vdash A \equiv B$ type
- $\Phi \mid \Gamma \vdash a \equiv a' : A$

Judgements related to pattern matching:

- $\Phi \mid \Gamma \vdash \Psi \mid \Omega$ tele
- $\Phi \mid \Gamma \vdash (\kappa \mid \theta) : \Psi \mid \Omega$
- $\Phi \mid \Gamma \vdash \Psi \mid \Delta \vdash p : A$ pattern
- $\Phi \mid \Gamma \vdash a \bowtie p[\kappa \mid \theta] : A$ pattern

A.1 Palettes

Palettes:

$$\textsc{pal-empty}\ \frac{}{1\ \mathsf{palette}} \qquad \textsc{pal-}\times\ \frac{\Phi_1\ \mathsf{palette} \qquad \Phi_2\ \mathsf{palette}}{\Phi_1, \Phi_2\ \mathsf{palette}}$$

$$\textsc{pal-sphere}\ \frac{}{\varnothing\ \mathsf{palette}} \qquad \textsc{pal-sphere-named}\ \frac{}{\varnothing_i\ \mathsf{palette}} \qquad \textsc{pal-}\otimes\ \frac{\Phi_1\ \mathsf{palette} \qquad \Phi_2\ \mathsf{palette}}{\Phi_1 \otimes \Phi_2\ \mathsf{palette}}$$

$$\textsc{pal-col}\ \frac{\Phi\ \mathsf{palette}}{\mathfrak{c} \prec \Phi\ \mathsf{palette}}$$

Colours:

$$\textsc{col-here}\ \frac{}{\mathfrak{c} \prec \Phi \vdash \mathfrak{c}\ \mathsf{colour}} \qquad \textsc{col-sub}\ \frac{\Phi \vdash \mathfrak{c}\ \mathsf{colour}}{\mathfrak{t} \prec \Phi \vdash \mathfrak{c}\ \mathsf{colour}}$$

$$\textsc{col-}\times\textsc{-left}\ \frac{\Phi_1 \vdash \mathfrak{c}\ \mathsf{colour}}{\Phi_1, \Phi_2 \vdash \mathfrak{c}\ \mathsf{colour}} \qquad \textsc{col-}\times\textsc{-right}\ \frac{\Phi_2 \vdash \mathfrak{c}\ \mathsf{colour}}{\Phi_1, \Phi_2 \vdash \mathfrak{c}\ \mathsf{colour}}$$

$$\textsc{col-}\otimes\textsc{-left}\ \frac{\Phi_1 \vdash \mathfrak{c}\ \mathsf{colour}}{\Phi_1 \otimes \Phi_2 \vdash \mathfrak{c}\ \mathsf{colour}} \qquad \textsc{col-}\otimes\textsc{-right}\ \frac{\Phi_2 \vdash \mathfrak{c}\ \mathsf{colour}}{\Phi_1 \otimes \Phi_2 \vdash \mathfrak{c}\ \mathsf{colour}}$$

Slices:

$$\frac{}{\Phi \vdash \varnothing\ \mathsf{preslice}} \qquad \frac{}{\mathfrak{c} \prec \Phi \vdash \mathfrak{c}\ \mathsf{preslice}} \qquad \frac{\Phi \vdash s\ \mathsf{preslice}}{\mathfrak{c} \prec \Phi \vdash s\ \mathsf{preslice}}$$

$$\frac{\Phi_1 \vdash s\ \mathsf{preslice}}{\Phi_1, \Phi_2 \vdash s\ \mathsf{preslice}} \qquad \frac{\Phi_2 \vdash s\ \mathsf{preslice}}{\Phi_1, \Phi_2 \vdash s\ \mathsf{preslice}}$$

$$\frac{\Phi_L \vdash s_L\ \mathsf{preslice} \qquad \Phi_R \vdash s_R\ \mathsf{preslice}}{\Phi_L \otimes \Phi_R \vdash s_L \otimes s_R\ \mathsf{preslice}}$$

A preslice is then a pair of a pure preslice $\Phi \vdash s\ \mathsf{preslice}$ and a flag ϵ representing the presence or absense of 1. We write preslices as $s \otimes \epsilon$, and use the shorthand s when $\epsilon \equiv \bot$ and $s \otimes 1$ when $s \equiv \top$.

A slice is then formed by

$$\frac{\Phi \vdash \mathfrak{c}\ \mathsf{colour}}{\Phi \vdash \mathfrak{c}\ \mathsf{slice}} \qquad \frac{\Phi \vdash s\ \mathsf{preslice}_\epsilon}{\Phi \vdash (\mathfrak{t}. \prec s)\ \mathsf{slice}} \qquad \frac{}{\Phi \vdash (\mathfrak{t}. \prec \varnothing_{i.})\ \mathsf{slice}}$$

For any slice s, let $\mathsf{u}(s)$ be the underlying preslice, defined by

$$\begin{aligned}\mathsf{u}(\mathfrak{c}) &:\equiv \mathfrak{c}\\ \mathsf{u}(\mathfrak{t}. \prec s) &:\equiv s\\ \mathsf{u}(\mathfrak{t}. \prec \varnothing_i) &:\equiv \varnothing\end{aligned}$$

Palette Splits: A presplit $\Phi \vdash s_L \boxtimes s_R\ \mathsf{presplit}$ presupposes that Φ is a palette s_L and s_R are each a preslice.

$$\frac{}{\varnothing \vdash (\varnothing \otimes \epsilon_L) \boxtimes (\varnothing \otimes \epsilon_R)\ \mathsf{presplit}} \qquad \frac{\epsilon_L \equiv \top \text{ or } \epsilon_R \equiv \top}{\Phi \vdash (\varnothing \otimes \epsilon_L) \boxtimes (\varnothing \otimes \epsilon_R)\ \mathsf{presplit}}$$

$$\frac{}{\mathfrak{c} \prec \Phi \vdash (\mathfrak{c} \otimes \epsilon_L) \boxtimes (\varnothing \otimes \epsilon_R)\ \mathsf{presplit}} \qquad \frac{}{\mathfrak{c} \prec \Phi \vdash (\varnothing \otimes \epsilon_L) \boxtimes (\mathfrak{c} \otimes \epsilon_R)\ \mathsf{presplit}}$$

$$\frac{\Phi \vdash s_L \boxtimes s_R\ \mathsf{presplit}}{\mathfrak{c} \prec \Phi \vdash s_L \boxtimes s_R\ \mathsf{presplit}}$$

$$\frac{\Phi_1 \vdash s_{1L} \boxtimes s_{1R}\ \mathsf{presplit}}{\Phi_1, \Phi_2 \vdash s_{1L} \boxtimes s_{1R}\ \mathsf{presplit}} \qquad \frac{\Phi_2 \vdash s_{2L} \boxtimes s_{2R}\ \mathsf{presplit}}{\Phi_1, \Phi_2 \vdash s_{2L} \boxtimes s_{2R}\ \mathsf{presplit}}$$

$$\frac{\begin{array}{c}\Phi_1 \vdash s_{1L} \boxtimes s_{1R}\ \mathsf{presplit}\\ \Phi_2 \vdash s_{2L} \boxtimes s_{2R}\ \mathsf{presplit}\end{array}}{\Phi_1 \otimes \Phi_2 \vdash (s_{1L} \otimes s_{2L}) \boxtimes (s_{1R} \otimes s_{2R})\ \mathsf{presplit}}$$

A split $\Phi \vdash s_L \boxtimes s_R\ \mathsf{split}$ is then

$$\frac{\Phi \vdash \mathsf{u}(s_L) \boxtimes \mathsf{u}(s_R)\ \mathsf{presplit}}{\Phi \vdash s_L \boxtimes s_R\ \mathsf{split}}$$

Palette Units

$$\textsc{unit-zero}\ \frac{}{\Phi \vdash 0\ \mathsf{unit}} \qquad \textsc{unit-here}\ \frac{}{\varnothing_i \vdash i\ \mathsf{unit}} \qquad \textsc{unit-sub}\ \frac{\Phi \vdash i\ \mathsf{unit}}{(\mathfrak{c} \prec \Phi) \vdash i\ \mathsf{unit}}$$

$$\textsc{unit-left}\ \frac{\Phi_1 \vdash i\ \mathsf{unit}}{\Phi_1, \Phi_2 \vdash i\ \mathsf{unit}} \qquad \textsc{unit-right}\ \frac{\Phi_2 \vdash i\ \mathsf{unit}}{\Phi_1, \Phi_2 \vdash i\ \mathsf{unit}}$$

Palette Substitutions

$$\textsc{pal-sub-empty}\ \frac{}{\Phi \vdash \cdot : 1} \qquad \textsc{pal-sub-}\times\ \frac{\Phi \vdash \kappa_1 : \Psi_1 \qquad \Phi \vdash \kappa_2 : \Psi_2}{\Phi \vdash \kappa_1, \kappa_2 : \Psi_1, \Psi_2}$$

$$\textsc{pal-sub-}\otimes\ \frac{\begin{array}{c}\Phi \vdash s_L \boxtimes s_R\ \mathsf{presplit}\\ \Phi^{s_L} \vdash \kappa_L : \Psi_L \qquad \Phi^{s_R} \vdash \kappa_R : \Psi_R\end{array}}{\Phi \vdash \kappa_L \otimes \kappa_R : \Psi_L \otimes \Psi_R} \qquad \textsc{pal-sub-unit}\ \frac{\Phi \vdash U\ \mathsf{unit}}{\Phi \vdash (U/j) : \varnothing_j}$$

$$\textsc{pal-sub-name}\ \frac{\begin{array}{c}\Phi \vdash s\ \mathsf{slice}\\ \Phi \vdash \mathsf{u}(s) \boxtimes \varnothing\ \mathsf{presplit} \qquad \Phi^{\mathsf{u}(s)} \vdash \kappa : \Psi\end{array}}{\Phi \vdash (s/\mathfrak{c} \prec \kappa) : (\mathfrak{c} \prec \Psi)}$$

A.1.1 Admissible Operations

Filtering

$$\textsc{pal-filter}\ \frac{\Phi \vdash s\ \mathsf{slice}}{\Phi^s\ \mathsf{palette}}$$

Composing Slices

$$\frac{\Phi \vdash s\ \mathsf{slice} \qquad \Phi^s \vdash t\ \mathsf{slice}}{\Phi \vdash t\ \mathsf{slice}}$$

Recolouring

$$\textsc{recolour-slice}\ \frac{\mathfrak{r} \prec \Phi \vdash s\ \mathsf{slice}}{\mathfrak{t} \prec \Phi \vdash s^{\mathfrak{t}\leftrightarrow\mathfrak{r}}\ \mathsf{slice}} \qquad \textsc{recolour-split}\ \frac{\mathfrak{r} \prec \Phi \vdash s_L \boxtimes s_R\ \mathsf{split}}{\mathfrak{t} \prec \Phi \vdash {s_L}^{\mathfrak{t}\leftrightarrow\mathfrak{r}} \boxtimes {s_R}^{\mathfrak{t}\leftrightarrow\mathfrak{r}}\ \mathsf{split}}$$

Weakening

$$\textsc{wk-slice}\ \frac{\Phi \vdash s\ \mathsf{slice}}{\Phi, \Psi \vdash s\ \mathsf{slice}} \qquad \textsc{wk-slice-one}\ \frac{1 \vdash s\ \mathsf{slice}}{\Phi \vdash s\ \mathsf{slice}}$$

Marking

$$\textsc{mark-slice}\ \frac{\mathfrak{t} \prec \Phi \vdash s\ \mathsf{slice}}{\mathfrak{t} \vdash s^{\mathsf{m}\Phi}\ \mathsf{slice}} \qquad \textsc{mark-split}\ \frac{\mathfrak{t} \prec \Phi \vdash s_L \boxtimes s_R\ \mathsf{split}}{\mathfrak{t} \vdash {s_L}^{\mathsf{m}\Phi} \boxtimes {s_R}^{\mathsf{m}\Phi}\ \mathsf{split}}$$

Substitution

$$\textsc{subst}\ \frac{\mathfrak{t} \prec \Phi \vdash \kappa : \Psi \qquad \mathfrak{t} \prec \Phi, \Psi \vdash \mathfrak{c}\ \mathsf{colour}}{\mathfrak{t} \prec \Phi \vdash \mathfrak{c}[\kappa]\ \mathsf{slice}}$$

$$\textsc{subst}\ \frac{\mathfrak{t} \prec \Phi \vdash \kappa : \Psi \qquad \mathfrak{t} \prec \Phi, \Psi \vdash s\ \mathsf{slice}}{\mathfrak{t} \prec \Phi \vdash s[\kappa]\ \mathsf{slice}} \qquad \textsc{subst}\ \frac{\mathfrak{t} \prec \Phi \vdash \kappa : \Psi \qquad \mathfrak{t} \prec \Phi, \Psi \vdash s_L \boxtimes s_R\ \mathsf{split}}{\mathfrak{t} \prec \Phi \vdash s_L[\kappa] \boxtimes s_R[\kappa]\ \mathsf{split}}$$

Tensor Merge

$$\textsc{merge}\ \frac{\begin{array}{c}\mathfrak{t} \prec \Phi \vdash s_L \boxtimes s_R\ \mathsf{split} \\ \mathfrak{t}' \prec \Phi^{s_L} \otimes \Phi^{s_R} \vdash t\ \mathsf{slice}\end{array}}{\mathfrak{t} \prec \Phi \vdash t\llbracket \mathfrak{t} \prec s_L \boxtimes s_R / \mathfrak{t}' \rrbracket}$$

A.2 Contexts

Contexts:

$$\textsc{ctx-empty}\ \frac{\Phi\ \mathsf{palette}}{\mathfrak{t} \prec \Phi \mid \cdot\ \mathsf{ctx}} \qquad \textsc{ctx-ext}\ \frac{\begin{array}{c}\Phi \vdash \mathfrak{c}\ \mathsf{colour} \\ \Phi^{\mathfrak{c}} \mid \Gamma^{\mathfrak{c}} \vdash A\ \mathsf{type}\end{array}}{\Phi \mid \Gamma, x^{\mathfrak{c}} : A\ \mathsf{ctx}} \qquad \textsc{ctx-ext-marked}\ \frac{\mathfrak{r} \mid \underline{\Gamma} \vdash A\ \mathsf{type}}{\mathfrak{t} \prec \Phi \mid \Gamma, \underline{x}^{\mathfrak{r}} : A\ \mathsf{ctx}}$$

$$\textsc{var}\ \frac{}{\mathfrak{t} \prec \Phi \mid \Gamma, x^{\mathfrak{t}} : A, \Gamma' \vdash x : A} \qquad \textsc{var-roundtrip}\ \frac{}{\mathfrak{t} \prec \Phi \mid \Gamma, x^{\mathfrak{c}} : A, \Gamma' \vdash \underline{x} : \underline{A}^{\mathfrak{c} \leftrightarrow \mathfrak{t}}}$$

$$\textsc{var-marked}\ \frac{}{\mathfrak{t} \prec \Phi \mid \Gamma, \underline{x}^{\mathfrak{c}} : A, \Gamma' \vdash \underline{x} : A^{\mathfrak{c} \leftrightarrow \mathfrak{t}}}$$

A.2.1 Telescopes and Substitutions

Telescopes

$$\textsc{tele-empty}\ \frac{\mathfrak{t} \prec \Phi \mid \Gamma\ \mathsf{ctx} \qquad \Psi\ \mathsf{palette}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \Psi \mid \cdot\ \mathsf{tele}} \qquad \textsc{tele-ext}\ \frac{\begin{array}{c}\mathfrak{t} \prec \Phi \mid \Gamma \vdash \Psi \mid \Omega\ \mathsf{tele} \\ \mathfrak{t} \prec \Psi \vdash \mathfrak{c}\ \mathsf{colour} \\ (\mathfrak{t} \prec \Phi, \Psi)^{\mathfrak{c}} \mid \Gamma^{\mathfrak{c}}, \Omega^{\mathfrak{c}} \vdash A\ \mathsf{type}\end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \Psi \mid \Omega, x^{\mathfrak{c}} : A\ \mathsf{tele}}$$

$$\textsc{tele-ext-marked}\ \frac{\begin{array}{c}\mathfrak{t} \prec \Phi \mid \Gamma \vdash \Psi \mid \Omega\ \mathsf{tele} \\ \mathfrak{c} \mid \underline{\Gamma, \Omega} \vdash A\ \mathsf{type}\end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \Psi \mid \Omega, \underline{x}^{\mathfrak{c}} : A\ \mathsf{tele}}$$

We can extend a context with a telescope:

$$\textsc{ctx-ext-tele}\ \frac{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \Psi \mid \Omega \ \mathsf{tele}}{\mathfrak{t} \prec \Phi, \Psi \mid \Gamma, \Omega \ \mathsf{ctx}}$$

Substitutions

$$\textsc{tele-sub-empty}\ \frac{\mathfrak{t} \prec \Phi \vdash \kappa : \Psi}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash (\kappa \mid \cdot) : \Psi \mid \cdot}$$

$$\textsc{tele-sub-ext}\ \frac{\begin{array}{c}\mathfrak{t} \prec \Phi \mid \Gamma \vdash (\kappa \mid \theta) : \Psi \mid \Omega \\ (\mathfrak{t} \prec \Phi)^{\mathfrak{c}[\kappa]} \mid \Gamma^{\mathfrak{c}[\kappa]} \vdash a : A[\kappa \mid \theta]\end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash (\kappa \mid \theta, a/x^{\mathfrak{c}}) : \Psi \mid \Omega, x^{\mathfrak{c}} : A}$$

$$\textsc{tele-sub-ext-marked}\ \frac{\begin{array}{c}\mathfrak{t} \prec \Phi \mid \Gamma \vdash (\kappa \mid \theta) : \Psi \mid \Omega \\ \mathfrak{c} \mid \underline{\Gamma} \vdash a : A[\kappa \mid \theta]\end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash (\kappa \mid \theta, a/\underline{x}) : \Psi \mid \Omega, \underline{x}^{\mathfrak{c}} : A}$$

A.2.2 Admissible Operations

In all of the following admissible rules, $\mathcal{J}$ stands in for the context, a context extension, type or term.

Filtering

$$\textsc{ctx-filter}\ \frac{\Phi \vdash s \ \mathsf{slice} \qquad \Phi \mid \Gamma \ \mathsf{ctx}}{\Phi^{s} \mid \Gamma^{s} \ \mathsf{ctx}} \qquad \textsc{ctx-mark}\ \frac{\mathfrak{t} \prec \Phi \mid \Gamma \ \mathsf{ctx}}{\mathfrak{t} \mid \underline{\Gamma} \ \mathsf{ctx}}$$

The latter is defined from the former by filtering at the slice $\mathfrak{t}. \prec 1$.

Recolouring We sometimes need to change the name of the top colour in a term:

$$\textsc{recolour}\ \frac{\mathfrak{r} \prec \Phi \mid \Gamma \vdash \mathcal{J}}{\mathfrak{t} \prec \Phi \mid \Gamma^{\mathfrak{t} \leftrightarrow \mathfrak{r}} \vdash \mathcal{J}^{\mathfrak{t} \leftrightarrow \mathfrak{r}}}$$

Weakening Variable weakening:

$$\textsc{wk}\ \frac{\Phi \mid \Gamma, \Gamma' \vdash \mathcal{J}}{\Phi \mid \Gamma, x^{\mathfrak{c}} : A, \Gamma' \vdash \mathcal{J}} \qquad \textsc{wk-marked}\ \frac{\Phi \mid \Gamma, \Gamma' \vdash \mathcal{J}}{\Phi \mid \Gamma, \underline{x}^{\mathfrak{c}} : A, \Gamma' \vdash \mathcal{J}}$$

Marking

$$\textsc{mark}\ \begin{array}{c} \mathsf{t} \prec \Phi \mid \Gamma \vdash \mathcal{J} \\ \hdashline \mathsf{t} \mid \underline{\Gamma} \vdash \underline{\mathcal{J}} \end{array}$$

Mark-weakening

$$\textsc{mark-wk}\ \begin{array}{c} \mathsf{t} \prec \Phi \mid \Gamma\ \mathsf{ctx} \\ \mathsf{t} \mid \underline{\Gamma} \vdash \mathcal{J} \\ \hdashline \mathsf{t} \prec \Phi \mid \Gamma \vdash \mathcal{J} \end{array}$$

Substitution

$$\textsc{subst}\ \begin{array}{c} \mathsf{t} \prec \Phi \mid \Gamma \vdash (\kappa \mid \theta) : \Psi \mid \Omega \qquad \mathsf{t} \prec \Phi, \Psi \mid \Gamma, \Omega, \Gamma' \vdash \mathcal{J} \\ \hdashline \mathsf{t} \prec \Phi \mid \Gamma, \Gamma'[\kappa \mid \theta] \vdash \mathcal{J}[\kappa \mid \theta] \end{array}$$

Tensor Merging

$$\textsc{merge}\ \begin{array}{c} \mathsf{t} \prec \Phi \vdash s_L \boxtimes s_R\ \mathsf{split} \\ \mathsf{t}' \prec \Phi^{s_L} \otimes \Phi^{s_R} \mid \Gamma^{\Phi^{s_L} \otimes \Phi^{s_R}} \vdash \mathcal{J} \\ \hdashline \mathsf{t} \prec \Phi \mid \Gamma \vdash \mathcal{J}[\![\mathsf{t} \prec s_L \boxtimes s_R / \mathsf{t}']\!] \end{array}$$

A.3 Types

Natural Modality

$$\natural\textsc{-form}\ \frac{\mathsf{t} \mid \underline{\Gamma} \vdash A\ \mathsf{type}}{\mathsf{t} \prec \Phi \mid \Gamma \vdash \natural A\ \mathsf{type}}$$

$$\natural\textsc{-intro}\ \frac{\mathsf{t} \mid \underline{\Gamma} \vdash a : A}{\mathsf{t} \prec \Phi \mid \Gamma \vdash a^{\natural} : \natural A} \qquad \natural\textsc{-elim}\ \frac{\mathsf{t} \prec \Phi \mid \Gamma \vdash a : \natural A}{\mathsf{t} \prec \Phi \mid \Gamma \vdash a_{\natural} : A}$$

$$\natural\textsc{-beta}\ \frac{\mathsf{t} \mid \underline{\Gamma} \vdash a : A}{\mathsf{t} \prec \Phi \mid \Gamma \vdash a^{\natural}{}_{\natural} \equiv a : A} \qquad \natural\textsc{-eta}\ \frac{\mathsf{t} \prec \Phi \mid \Gamma \vdash v : \natural A}{\mathsf{t} \prec \Phi \mid \Gamma \vdash v \equiv \underline{v_{\natural}}^{\natural} : \natural A}$$

Tensor

$$\otimes\text{-FORM}\ \frac{\mathfrak{t} \mid \underline{\Gamma} \vdash A \ \mathsf{type} \qquad \mathfrak{t} \mid \underline{\Gamma}, \underline{x}^{\mathfrak{t}} : \underline{A} \vdash B \ \mathsf{type}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \textcircled{Σ}_{(\underline{x}:A)}\, B \ \mathsf{type}}$$

$$\otimes\text{-INTRO}\ \frac{\begin{array}{c}\mathfrak{t} \prec \Phi \vdash s_L \boxtimes s_R \ \mathsf{split} \\ (\mathfrak{t} \prec \Phi)^{s_L} \mid \Gamma^{s_L} \vdash a : A^{\ulcorner s_L \urcorner \leftrightarrow \mathfrak{t}} \qquad (\mathfrak{t} \prec \Phi)^{s_R} \mid \Gamma^{s_R} \vdash b : B[\underline{a}^{\mathfrak{t} \leftrightarrow \ulcorner s_L \urcorner} / \underline{x}^{\mathfrak{t}}]^{\ulcorner s_R \urcorner \leftrightarrow \mathfrak{t}}\end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash a \ {}_{s_L}\!\otimes_{s_R} b : \textcircled{Σ}_{(\underline{x}:A)}\, B}$$

Monoidal Unit

$$\mathbb{S}\text{-FORM}\ \frac{}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \mathbb{S} \ \mathsf{type}}$$

$$\mathbb{S}\text{-INTRO}\ \frac{\mathfrak{t} \prec \Phi \vdash i \ \mathsf{unit}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \text{¤}_i : \mathbb{S}}$$

A.3.1 Pattern Matching

Match

$$\text{MATCH}\ \frac{\begin{array}{c}\mathfrak{t} \prec \Phi \mid \Gamma \vdash \Psi \mid \Omega \vdash p : A \ \mathsf{pattern} \\ \mathfrak{t} \prec \Phi \mid \Gamma, z^{\mathfrak{t}} : A \vdash C \ \mathsf{type} \\ \mathfrak{t} \prec \Phi, \Psi \mid \Gamma, \Omega \vdash c : C[p/z] \\ \mathfrak{t} \prec \Phi \mid \Gamma \vdash a : A\end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \mathsf{let}\ p = a \ \mathsf{in}\ c : C[a/z]}$$

$$\text{MATCH-BETA}\ \frac{\begin{array}{c}\mathfrak{t} \prec \Phi \mid \Gamma \vdash \Psi \mid \Omega \vdash p : A \ \mathsf{pattern} \\ \mathfrak{t} \prec \Phi \mid \Gamma, z^{\mathfrak{t}} : A \vdash C \ \mathsf{type} \\ \mathfrak{t} \prec \Phi, \Psi \mid \Gamma, \Omega \vdash c : C[p/z] \\ \mathfrak{t} \prec \Phi \mid \Gamma \vdash \Psi \mid \Omega \vdash s \bowtie p[\kappa \mid \theta] : A \ \mathsf{pattern}\end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash (\mathsf{let}\ p = s \ \mathsf{in}\ c) \equiv c[\kappa \mid \theta] : C[\kappa \mid \theta]}$$

Patterns

$$\frac{\mathfrak{t} \prec \Phi \mid \Gamma \vdash A \ \mathsf{type}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash 1 \mid x^{\mathfrak{t}} : A \vdash x : A \ \mathsf{pattern}}$$

$$\frac{\begin{array}{c}\mathfrak{t} \prec \Phi \mid \Gamma \vdash \Psi_1 \mid \Omega_1 \vdash p_1 : A \ \mathsf{pattern} \\ \mathfrak{t} \prec \Phi \mid \Gamma, x^{\mathfrak{t}} : A \vdash B \ \mathsf{type} \\ \mathfrak{t} \prec \Phi, \Psi_1 \mid \Gamma, \Omega_1 \vdash \Psi_2 \mid \Omega_2 \vdash p_2 : B[p_1/x] \ \mathsf{pattern}\end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \Psi_1, \Psi_2 \mid \Omega_1, \Omega_2 \vdash (p_1, p_2) : \sum_{(x:A)} B \ \mathsf{pattern}} \qquad \frac{}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash 1 \mid \cdot \vdash \star : 1 \ \mathsf{pattern}}$$

$$\frac{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \Psi \mid \Omega \vdash p : A \ \mathsf{pattern}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \Psi \mid \Omega \vdash (p, p, \mathsf{refl}_p) : \sum_{(x:A)} \sum_{(y:A)} x =_A y \ \mathsf{pattern}}$$

$$\frac{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \Psi \mid \Omega \vdash p : A \ \mathsf{pattern}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash 1 \mid \underline{\Omega} \vdash \underline{p}^{\natural} : \natural \underline{A} \ \mathsf{pattern}}$$

$$\frac{\begin{array}{c}\mathfrak{c}_L \mid \underline{\Gamma} \vdash \Psi_L \mid \Omega_L \vdash p_L : A^{\mathfrak{c}_L \leftrightarrow \mathfrak{t}} \ \mathsf{pattern} \\ \mathfrak{c}_R \mid \underline{\Gamma}, \underline{\Omega_L} \vdash \Psi_R \mid \Omega_R \vdash p_R : B[\underline{p_L}^{\mathfrak{t} \leftrightarrow \mathfrak{c}_L} / \underline{x}]^{\mathfrak{c}_R \leftrightarrow \mathfrak{t}} \ \mathsf{pattern}\end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash (\mathfrak{c}_L \prec \Psi_L) \otimes (\mathfrak{c}_R \prec \Psi_R) \mid \Omega_L, \Omega_R \vdash (p_L \ {}_{\mathfrak{c}_L}\!\otimes_{\mathfrak{c}_R} \ p_R) : \mathop{\textcircled{Σ}}_{(\underline{x}:A)} B \ \mathsf{pattern}}$$

$$\frac{}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \varnothing_i \mid \cdot \vdash ⌑_i : \mathbb{S} \ \mathsf{pattern}}$$

$$\frac{\mathfrak{t} \mid \underline{\Gamma} \vdash \Psi_L \mid \Omega_L \vdash p_L : A \ \mathsf{pattern}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \Psi_L \mid \Omega_L \vdash (p_L \ {}_{\mathfrak{t}}\!\otimes_{\mathfrak{u}.\prec \varnothing_i} \ ⌑_i) : \mathop{\textcircled{Σ}}_{(\underline{x}:A)} \mathbb{S} \ \mathsf{pattern}}$$

$$\frac{\begin{array}{c}\mathfrak{t} \mid \underline{\Gamma}, \underline{x} : \mathbb{S} \vdash B \ \mathsf{type} \\ \mathfrak{t} \mid \underline{\Gamma} \vdash \Psi_R \mid \Omega_R \vdash p_R : B[\underline{⌑}/\underline{x}] \ \mathsf{pattern}\end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \Psi_R \mid \Omega_R \vdash (⌑_i \ {}_{\mathfrak{u}.\prec \varnothing_i}\!\otimes_{\mathfrak{t}} \ p_R) : \mathop{\textcircled{Σ}}_{(\underline{x}:\mathbb{S})} B \ \mathsf{pattern}}$$

$$\frac{}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash a \bowtie x[\cdot \mid a/x] : A}$$

$$\frac{\begin{array}{c}\mathfrak{t} \prec \Phi \mid \Gamma \vdash a \bowtie p_1[\kappa_1 \mid \delta_1] : A \\ \mathfrak{t} \prec \Phi \mid \Gamma \vdash b \bowtie p_2[\kappa_2 \mid \delta_2] : B[a/x]\end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash (a, b) \bowtie (p_1, p_2)[\kappa_1, \kappa_2 \mid \delta_1, \delta_2] : \sum_{(x:A)} B} \qquad \frac{}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \star \bowtie \star[\cdot \mid \cdot] : 1}$$

$$\frac{\mathfrak{t} \prec \Phi \mid \Gamma \vdash a \bowtie p[\kappa \mid \delta] : A}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash (a, a, \mathsf{refl}_a) \bowtie (p, p, \mathsf{refl}_p)[\kappa \mid \delta] : A}$$

$$\frac{\mathfrak{t} \mid \underline{\Gamma} \vdash n \bowtie p[\kappa \mid \delta] : A}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash n^{\natural} \bowtie \underline{p}^{\natural}[\kappa \mid \delta] : \natural A \ \ \mathsf{pattern}}$$

$$\frac{\begin{array}{c}\mathfrak{t} \prec \Phi \vdash s_L \boxtimes s_R \ \mathsf{split} \\ \Phi^{s_L} \mid \Gamma^{s_L} \vdash a \bowtie p_L[\kappa_L \mid \delta_L] : A \\ \Phi^{s_R} \mid \Gamma^{s_R} \vdash b \bowtie p_R[\kappa_R \mid \delta_R] : B[\underline{a}/\underline{x}]\end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash (a \ {}_{s_L}\!\otimes_{s_R} b) \bowtie (p_L \ {}_{\mathfrak{c}_L}\!\otimes_{\mathfrak{c}_R} p_R)[(s_L/\mathfrak{c}_L \prec \kappa_L) \otimes (s_R/\mathfrak{c}_R \prec \kappa_R) \mid \delta_L, \delta_R] : \bigcirc\hspace{-1em}\sum_{(\underline{x}:A)} B}$$

$$\frac{}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash ⌑_j \bowtie ⌑_i[j/i \mid \cdot]}$$

$$\frac{\mathfrak{t} \prec \Phi \mid \Gamma \vdash a \bowtie p_L[\kappa_L \mid \delta_L] : A}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash (a \ {}_{\mathfrak{t}}\!\otimes_{\mathfrak{u}.\prec \varnothing_i} ⌑_i) \bowtie (p_L \ {}_{\mathfrak{t}}\!\otimes_{\mathfrak{u}.\prec \varnothing_i} ⌑_i)[\kappa_L \mid \delta_L] : A \otimes \mathbb{S}}$$

$$\frac{\mathfrak{t} \prec \Phi \mid \Gamma \vdash b \bowtie (p_R[\underline{⌑}/x])[\kappa_R \mid \delta_R] : B[\underline{⌑}/\underline{x}]}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash (⌑_i \ {}_{\mathfrak{u}.\prec \varnothing_i}\!\otimes_{\mathfrak{t}} b) \bowtie (⌑_i \ {}_{\mathfrak{u}.\prec \varnothing_i}\!\otimes_{\mathfrak{t}} p_R)[\kappa_L \mid \delta_L] : \bigcirc\hspace{-1em}\sum_{(\underline{x}:\mathbb{S})} B}$$

A.3.2 Hom

$$\multimap\text{-FORM}\ \frac{\begin{array}{c}\mathfrak{r} \mid \underline{\Gamma} \vdash A\ \mathsf{type} \\ \mathfrak{p} \prec (\mathfrak{t} \prec \Phi) \otimes \mathfrak{r} \mid \Gamma, x^{\mathfrak{r}} : A \vdash B\ \mathsf{type}\end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \mathop{\textcircled{$\prod$}}_{(x^{\mathfrak{r}}:A)}{}^{\mathfrak{p}}B\ \mathsf{type}}$$

$$\multimap\text{-INTRO}\ \frac{\mathfrak{p} \prec (\mathfrak{t} \prec \Phi) \otimes \mathfrak{r} \mid \Gamma, x^{\mathfrak{r}} : A \vdash b : B}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \partial^{\mathfrak{p}} x^{\mathfrak{r}}.b : \mathop{\textcircled{$\prod$}}_{(x^{\mathfrak{r}}:A)}{}^{\mathfrak{p}}B}$$

$$\multimap\text{-ELIM}\ \frac{\begin{array}{c}\mathfrak{t} \prec \Phi \vdash s_L \boxtimes s_R\ \mathsf{split} \qquad \mathfrak{r} :\equiv \ulcorner s_R \urcorner \\ (\mathfrak{t} \prec \Phi)^{s_L} \mid \Gamma^{s_L} \vdash f : \mathop{\textcircled{$\prod$}}_{(x^{\mathfrak{r}}:A)}{}^{\mathfrak{p}}B \qquad (\mathfrak{t} \prec \Phi)^{s_R} \mid \Gamma^{s_R} \vdash a : A\end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash f_{s_L}\langle a \rangle_{s_R} : B[a/x]\llbracket \mathfrak{t} \prec s_L \boxtimes s_R / \mathfrak{p} \rrbracket}$$

$$\multimap\text{-BETA}\ \frac{\begin{array}{c}\mathfrak{t} \prec \Phi \vdash s_L \boxtimes s_R\ \mathsf{split} \qquad \mathfrak{r} :\equiv \ulcorner s_R \urcorner \\ \mathfrak{p} \prec (\Phi^{s_L} \otimes \mathfrak{r}) \mid \Gamma^{s_L}, x^{\mathfrak{r}} : A \vdash b : B \qquad \Phi^{s_R} \mid \Gamma^{s_R} \vdash a : A\end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash (\partial^{\mathfrak{p}} x^{\mathfrak{r}}.b)_{s_L}\langle a \rangle_{s_R} \equiv b[a/x]\llbracket \mathfrak{t} \prec s_L \boxtimes s_R / \mathfrak{p} \rrbracket : B[a/x]\llbracket \mathfrak{t} \prec s_L \boxtimes s_R / \mathfrak{p} \rrbracket}$$

$$\multimap\text{-ETA}\ \frac{\mathfrak{t} \prec \Phi \mid \Gamma \vdash f : \mathop{\textcircled{$\prod$}}_{(x^{\mathfrak{r}}:A)}{}^{\mathfrak{p}}B}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash f \equiv (\partial^{\mathfrak{p}} x^{\mathfrak{r}}.f_{\mathfrak{t}}\langle x \rangle_{\mathfrak{r}}) : \mathop{\textcircled{$\prod$}}_{(x^{\mathfrak{r}}:A)}{}^{\mathfrak{p}}B}$$

A.4 Definitions of Admissible Rules

A.4.1 Recolouring

On contexts, we recolour each type and change the label on the variable if necessary:

$$\begin{aligned}
(\cdot)^{\mathfrak{d}\leftrightarrow\mathfrak{c}} &:\equiv \cdot \\
(\Gamma, x^{\mathfrak{c}} : A)^{\mathfrak{d}\leftrightarrow\mathfrak{c}} &:\equiv \Gamma^{\mathfrak{d}\leftrightarrow\mathfrak{c}}, x^{\mathfrak{d}} : A^{\mathfrak{d}\leftrightarrow\mathfrak{c}} \\
(\Gamma, x^{\mathfrak{c}'} : A)^{\mathfrak{d}\leftrightarrow\mathfrak{c}} &:\equiv \Gamma^{\mathfrak{d}\leftrightarrow\mathfrak{c}}, x^{\mathfrak{c}'} : A^{\mathfrak{d}\leftrightarrow\mathfrak{c}} \\
(\Gamma, \underline{x}^{\mathfrak{c}'} : A)^{\mathfrak{d}\leftrightarrow\mathfrak{c}} &:\equiv \Gamma^{\mathfrak{d}\leftrightarrow\mathfrak{c}}, \underline{x}^{\mathfrak{c}'} : A
\end{aligned}$$

On slices, replace $\mathfrak{c}$ with $\mathfrak{d}$ whenever it appears..

On types and terms, induct on the syntax to the leaves, applying the rule to any slices encountered.

A.4.2 Marking

On preslices:

$$\begin{aligned}
1^{\mathsf{m}\Phi} &:\equiv 1 \\
\varnothing^{\mathsf{m}\Phi} &:\equiv \varnothing \\
(s \otimes \mathfrak{c})^{\mathsf{m}\Phi} &:\equiv s^{\mathsf{m}\Phi} \otimes 1 && \text{if } \mathfrak{c} \in \Phi \\
(s \otimes \mathfrak{c})^{\mathsf{m}\Phi} &:\equiv s^{\mathsf{m}\Phi} \otimes \mathfrak{c} && \text{otherwise} \\
(s \otimes 1)^{\mathsf{m}\Phi} &:\equiv s^{\mathsf{m}\Phi} \otimes 1
\end{aligned}$$

On slices:

$$\begin{aligned}
\mathfrak{r}^{\mathsf{m}\Phi} &:\equiv \mathfrak{r}. \prec 1 && \text{if } \mathfrak{r} \in \Phi \\
\mathfrak{r}^{\mathsf{m}\Phi} &:\equiv \mathfrak{r} && \text{otherwise} \\
(\mathfrak{r}. \prec s)^{\mathsf{m}\Phi} &:\equiv \mathfrak{r}. \prec s^{\mathsf{m}\Phi} \\
(\mathfrak{r}. \prec \varnothing_i)^{\mathsf{m}\Phi} &:\equiv \mathfrak{r}. \prec \varnothing_i
\end{aligned}$$

On telescopes:

$$\begin{aligned}
(\cdot)^{\mathsf{m}\Phi|\Gamma} &:\equiv \cdot \\
(\Omega, x^{\mathfrak{r}} : A)^{\mathsf{m}\Phi|\Gamma} &:\equiv \Omega^{\mathsf{m}\Phi|\Gamma}, x^{\mathfrak{r}} : A^{\mathsf{m}\Phi|\Gamma} \\
(\Omega, \underline{x}^{\mathfrak{r}} : A)^{\mathsf{m}\Phi|\Gamma} &:\equiv \Omega^{\mathsf{m}\Phi|\Gamma}, \underline{x}^{\mathfrak{r}} : A^{\mathsf{m}\Phi|\Gamma}
\end{aligned}$$

On types and terms:

$$\begin{aligned}
x^{\mathsf{m}\Phi|\Gamma} &:\equiv \underline{x} && \text{if } x \in \Gamma \\
x^{\mathsf{m}\Phi|\Gamma} &:\equiv x && \text{otherwise} \\
\underline{x}^{\mathsf{m}\Phi|\Gamma} &:\equiv \underline{x} \\
\Big(\textstyle\sum_{(x:A)} B\Big)^{\mathsf{m}\Phi|\Gamma} &:\equiv \textstyle\sum_{(x:A^{\mathsf{m}\Phi|\Gamma})} B^{\mathsf{m}\Phi|\Gamma} \\
(a, b)^{\mathsf{m}\Phi|\Gamma} &:\equiv (a^{\mathsf{m}\Phi|\Gamma}, b^{\mathsf{m}\Phi|\Gamma}) \\
\mathsf{pr}_1(p)^{\mathsf{m}\Phi|\Gamma} &:\equiv \mathsf{pr}_1(p^{\mathsf{m}\Phi|\Gamma}) \\
\mathsf{pr}_2(p)^{\mathsf{m}\Phi|\Gamma} &:\equiv \mathsf{pr}_2(p^{\mathsf{m}\Phi|\Gamma})
\end{aligned}$$

$$\begin{aligned}
\Big(\textstyle\prod_{(x:A)} B\Big)^{\mathsf{m}\Phi|\Gamma} &:\equiv \textstyle\prod_{(x:A^{\mathsf{m}\Phi|\Gamma})} B^{\mathsf{m}\Phi|\Gamma} \\
(\lambda p.b)^{\mathsf{m}\Phi|\Gamma} &:\equiv (\lambda p.b^{\mathsf{m}\Phi|\Gamma}) \\
f(a)^{\mathsf{m}\Phi|\Gamma} &:\equiv f^{\mathsf{m}\Phi|\Gamma}(a^{\mathsf{m}\Phi|\Gamma})
\end{aligned}$$

$$\begin{aligned}
(\natural A)^{\mathsf{m}\Phi|\Gamma} &:\equiv \natural A^{\mathsf{m}\Phi|\Gamma} \\
(a^{\natural})^{\mathsf{m}\Phi|\Gamma} &:\equiv (a^{\mathsf{m}\Phi|\Gamma})^{\natural} \\
(a_{\natural})^{\mathsf{m}\Phi|\Gamma} &:\equiv (a^{\mathsf{m}\Phi|\Gamma})_{\natural}
\end{aligned}$$

$$\begin{aligned}
\Big(\textcircled{$\textstyle\sum$}_{(\underline{x}:A)} B\Big)^{\mathsf{m}\Phi|\Gamma} &:\equiv \textcircled{$\textstyle\sum$}_{(\underline{x}:A^{\mathsf{m}\Phi|\Gamma})} B^{\mathsf{m}\Phi|\Gamma} \\
(a \; {}_{s_L}\!\otimes_{s_R} \; b)^{\mathsf{m}\Phi|\Gamma} &:\equiv a^{\mathsf{m}\Phi|\Gamma} \; {}_{s_L{}^{\mathsf{m}\Phi}}\!\otimes_{s_R{}^{\mathsf{m}\Phi}} \; b^{\mathsf{m}\Phi|\Gamma} \\
(\mathsf{let}\ x \; {}_{\mathfrak{l}}\!\otimes_{\mathfrak{r}} \; y = s \ \mathsf{in}\ c)^{\mathsf{m}\Phi|\Gamma} &:\equiv \mathsf{let}\ x \; {}_{\mathfrak{l}}\!\otimes_{\mathfrak{r}} \; y = s^{\mathsf{m}\Phi|\Gamma} \ \mathsf{in}\ c^{\mathsf{m}\Phi|\Gamma}
\end{aligned}$$

$$\begin{aligned}
\Big(\textcircled{$\textstyle\prod$}_{(x^{\mathfrak{r}}:A)} {}^{\mathfrak{p}}B\Big)^{\mathsf{m}\Phi|\Gamma} &:\equiv \textcircled{$\textstyle\prod$}_{(x^{\mathfrak{r}}:A^{\mathsf{m}\Phi|\Gamma})} {}^{\mathfrak{p}}B^{\mathsf{m}\Phi|\Gamma} \\
(\partial^{\mathfrak{p}} x^{\mathfrak{r}}.b)^{\mathsf{m}\Phi|\Gamma} &:\equiv \partial^{\mathfrak{p}} x^{\mathfrak{r}}.(b^{\mathsf{m}\Phi|\Gamma}) \\
(f_{s_L} \langle a \rangle_{s_R})^{\mathsf{m}\Phi|\Gamma} &:\equiv f^{\mathsf{m}\Phi|\Gamma}{}_{s_L{}^{\mathsf{m}\Phi}} \langle a^{\mathsf{m}\Phi|\Gamma} \rangle_{s_R{}^{\mathsf{m}\Phi}}
\end{aligned}$$

$$\begin{aligned}
\mathbb{S}^{\mathsf{m}\Phi|\Gamma} &:\equiv \mathbb{S} \\
\text{⌑}_i{}^{\mathsf{m}\Phi|\Gamma} &:\equiv \underline{\text{⌑}} && \text{if } \varnothing_i \in \Phi \\
\text{⌑}_i{}^{\mathsf{m}\Phi|\Gamma} &:\equiv \text{⌑}_i && \text{otherwise} \\
(\mathsf{let}\ \varnothing_i = s \ \mathsf{in}\ c)^{\mathsf{m}\Phi|\Gamma} &:\equiv \mathsf{let}\ \varnothing_i = s^{\mathsf{m}\Phi|\Gamma} \ \mathsf{in}\ c^{\mathsf{m}\Phi|\Gamma}
\end{aligned}$$

A.4.3 Filtering

Filtering a palette, defined on a well-formed slice.

$$\frac{\Phi \vdash s\ \mathsf{preslice}}{\Phi^s\ \mathsf{palette}} \qquad \frac{\Phi \vdash s\ \mathsf{preslice}_\epsilon}{\Phi^s\ \mathsf{palette}} \qquad \frac{\Phi \vdash s\ \mathsf{slice}}{\Phi^s\ \mathsf{palette}}$$

defined by

$$\begin{aligned}
(\mathfrak{c} \prec \Phi)^s &:\equiv \Phi^s && \text{if } \Phi \vdash s\ \mathsf{preslice}_\epsilon \\
(\mathfrak{c} \prec \Phi)^{\mathfrak{c}} &:\equiv \mathfrak{c} \prec \Phi && \\
(\Phi, \Phi')^s &:\equiv \Phi^s && \text{if } \Phi \vdash s\ \mathsf{preslice}_\epsilon \\
(\Phi, \Phi')^s &:\equiv \Phi'^s && \text{if } \Phi' \vdash s\ \mathsf{preslice}_\epsilon \\
(\Phi \otimes \Phi')^{s \otimes \varnothing} &:\equiv \Phi^s && \\
(\Phi \otimes \Phi')^{\varnothing \otimes s'} &:\equiv \Phi^{s'} && \\
(\Phi \otimes \Phi')^{s \otimes s'} &:\equiv \Phi^s \otimes \Phi'^{s'} &&
\end{aligned}$$

$$\Phi^{s \otimes 1} :\equiv \Phi^s \otimes 1$$

$$\begin{aligned}
\Phi^{\mathfrak{t}. \prec s} &:\equiv \mathfrak{t} \prec \Phi^s \\
\Phi^{\mathfrak{t}. \prec \varnothing_{i.}} &:\equiv \mathfrak{t} \prec \varnothing_i
\end{aligned}$$

Filtering a context:

$$\begin{aligned}
(\cdot)^s &:\equiv \cdot && \\
(\Gamma, x^{\mathfrak{c}} : A)^s &:\equiv \Gamma^s, x^{\mathfrak{c}} : A && \text{if } \mathfrak{c} \in \Phi^s \\
(\Gamma, x^{\mathfrak{c}} : A)^s &:\equiv \Gamma^s, \underline{x}^{\mathfrak{c}} : \underline{A} && \text{otherwise} \\
(\Gamma, \underline{x}^{\mathfrak{c}} : A)^s &:\equiv \Gamma^s, \underline{x}^{\mathfrak{c}} : A &&
\end{aligned}$$

Define marking by filtering at the slice $\mathfrak{t}. \prec 1$.

A.4.4 Telescope Substitution

On colours:

$$\mathfrak{c}[\ldots, s/\mathfrak{c}, \ldots] :\equiv s$$

On pure preslices, preslices and slices:

$$(\mathfrak{c}_1 \otimes \cdots \otimes \mathfrak{c}_n)[\kappa] :\equiv \mathsf{u}(\mathfrak{c}_1[\kappa]) \otimes \cdots \otimes \mathsf{u}(\mathfrak{c}_n[\kappa])$$

$$\begin{aligned}
(s \otimes 1)[\kappa] &:\equiv s[\kappa] \otimes 1 \\
(\mathfrak{s}. \prec s)[\kappa] &:\equiv \mathfrak{t}. \prec s[\kappa] \\
(\mathfrak{s}. \prec \varnothing_i)[\kappa] &:\equiv \mathfrak{t}. \prec \varnothing_i
\end{aligned}$$

On context extensions:

$$\begin{aligned}
(\cdot)[\kappa \mid \theta] &:\equiv \cdot \\
(\Gamma', x^{\mathfrak{r}} : A)[\kappa \mid \theta] &:\equiv \Gamma'[\kappa \mid \theta], x^{\mathfrak{r}} : A[\kappa \mid \theta]^{\mathsf{at}\mathfrak{r}} \\
(\Gamma', \underline{x}^{\mathfrak{r}} : A)[\kappa \mid \theta] &:\equiv \Gamma'[\kappa \mid \theta], \underline{x}^{\mathfrak{r}} : A[\kappa \mid \theta]^{\mathsf{at}\mathfrak{c}}
\end{aligned}$$

On types and terms:

$$\begin{aligned}
x[\kappa \mid \theta, a/x^{\mathfrak{t}}, \theta']^{\mathsf{at}\mathfrak{t}} &:\equiv a \\
\underline{x}[\kappa \mid \theta, a/x^{\mathfrak{c}}, \theta']^{\mathsf{at}\mathfrak{t}} &:\equiv \underline{a}^{\mathfrak{t}\leftrightarrow\mathfrak{c}[\kappa]} \\
\underline{x}[\kappa \mid \theta, a/\underline{x}^{\mathfrak{c}}, \theta']^{\mathsf{at}\mathfrak{t}} &:\equiv a^{\mathfrak{t}\leftrightarrow\mathfrak{c}} \\
y[\kappa \mid \theta]^{\mathsf{at}\mathfrak{t}} &:\equiv y \\
\underline{y}[\kappa \mid \theta]^{\mathsf{at}\mathfrak{t}} &:\equiv \underline{y}
\end{aligned}$$

$$\begin{aligned}
\left(\sum_{(x:A)} B\right)[\kappa \mid \theta]^{\mathsf{at}\mathfrak{t}} &:\equiv \sum_{(x:A[\kappa\mid\theta]^{\mathsf{at}\mathfrak{t}})} B[\kappa \mid \theta]^{\mathsf{at}\mathfrak{t}} \\
(a, b)[\kappa \mid \theta]^{\mathsf{at}\mathfrak{t}} &:\equiv (a[\kappa \mid \theta]^{\mathsf{at}\mathfrak{t}}, b[\kappa \mid \theta]^{\mathsf{at}\mathfrak{t}}) \\
\mathsf{pr}_1(p)[\kappa \mid \theta]^{\mathsf{at}\mathfrak{t}} &:\equiv \mathsf{pr}_1(p[\kappa \mid \theta]^{\mathsf{at}\mathfrak{t}}) \\
\mathsf{pr}_2(p)[\kappa \mid \theta]^{\mathsf{at}\mathfrak{t}} &:\equiv \mathsf{pr}_2(p[\kappa \mid \theta]^{\mathsf{at}\mathfrak{t}})
\end{aligned}$$

$$\begin{aligned}
\left(\prod_{(x:A)} B\right)[\kappa \mid \theta]^{\mathsf{at}\mathfrak{t}} &:\equiv \prod_{(x:A[\kappa\mid\theta]^{\mathsf{at}\mathfrak{t}})} B[\kappa \mid \theta]^{\mathsf{at}\mathfrak{t}} \\
(\lambda p.b)[\kappa \mid \theta]^{\mathsf{at}\mathfrak{t}} &:\equiv (\lambda p.b[\kappa \mid \theta]^{\mathsf{at}\mathfrak{t}}) \\
f(a)[\kappa \mid \theta]^{\mathsf{at}\mathfrak{t}} &:\equiv f[\kappa \mid \theta]^{\mathsf{at}\mathfrak{t}}(a[\kappa \mid \theta]^{\mathsf{at}\mathfrak{t}})
\end{aligned}$$

$$\begin{aligned}
(\natural A)[\kappa \mid \theta]^{\mathsf{at}\mathfrak{t}} &:\equiv \natural A[\kappa \mid \theta]^{\mathsf{at}\mathfrak{t}} \\
(a^{\natural})[\kappa \mid \theta]^{\mathsf{at}\mathfrak{t}} &:\equiv (a[\kappa \mid \theta]^{\mathsf{at}\mathfrak{t}})^{\natural} \\
(a_{\natural})[\kappa \mid \theta]^{\mathsf{at}\mathfrak{t}} &:\equiv (a[\kappa \mid \theta]^{\mathsf{at}\mathfrak{t}})_{\natural}
\end{aligned}$$

$$\begin{aligned}
\left(\textcircled{Σ}_{(\underline{x}:A)} B\right)[\kappa \mid \theta]^{\mathsf{at}\mathfrak{t}} &:\equiv \textcircled{Σ}_{(\underline{x}:A[\kappa\mid\theta]^{\mathsf{at}\mathfrak{t}})} B[\kappa \mid \theta]^{\mathsf{at}\mathfrak{t}} \\
(a \ {}_{s_L}\!\otimes_{s_R} b)[\kappa \mid \theta]^{\mathsf{at}\mathfrak{t}} &:\equiv a[\kappa \mid \theta]^{\mathsf{at}\ulcorner s_L \urcorner} \ {}_{s_L}\!\otimes_{s_R} b[\kappa \mid \theta]^{\mathsf{at}\ulcorner s_R \urcorner} \\
(\mathsf{let}\ x \ {}_{\mathfrak{l}}\!\otimes_{\mathfrak{r}} y = s \ \mathsf{in}\ c)[\kappa \mid \theta]^{\mathsf{at}\mathfrak{t}} &:\equiv \mathsf{let}\ x \ {}_{\mathfrak{l}}\!\otimes_{\mathfrak{r}} y = s[\kappa \mid \theta]^{\mathsf{at}\mathfrak{t}} \ \mathsf{in}\ c[\kappa \mid \theta]^{\mathsf{at}\mathfrak{t}}
\end{aligned}$$

$$\begin{aligned}
\left(\textcircled{Π}_{(x^{\mathfrak{r}}:A)} {}^{\mathfrak{p}}B\right)[\kappa \mid \theta]^{\mathsf{at}\mathfrak{t}} &:\equiv \textcircled{Π}_{(x^{\mathfrak{r}}:A[\kappa\mid\theta]^{\mathsf{at}\mathfrak{r}})} {}^{\mathfrak{p}}B[\kappa \mid \theta]^{\mathsf{at}\mathfrak{p}} \\
(\partial^{\mathfrak{p}} x^{\mathfrak{r}}.b)[\kappa \mid \theta]^{\mathsf{at}\mathfrak{t}} &:\equiv \partial^{\mathfrak{p}} x^{\mathfrak{r}}.(b[\kappa \mid \theta]^{\mathsf{at}\mathfrak{p}}) \\
(f_{s_L}\langle a\rangle_{s_R})[\kappa \mid \theta]^{\mathsf{at}\mathfrak{t}} &:\equiv f[\kappa \mid \theta]^{\mathsf{at}\ulcorner s_L \urcorner}{}_{s_L}\langle a[\kappa \mid \theta]^{\mathsf{at}\ulcorner s_R \urcorner}\rangle_{s_R}
\end{aligned}$$

$$\begin{aligned}
\mathbb{S}[\kappa \mid \theta]^{\mathsf{at}\mathfrak{t}} &:\equiv \mathbb{S} \\
\text{⌑}_i[\kappa \mid \theta]^{\mathsf{at}\mathfrak{t}} &:\equiv \text{⌑}_i \\
(\mathsf{let}\ \text{⌑}_i = s \ \mathsf{in}\ c)[\kappa \mid \theta]^{\mathsf{at}\mathfrak{t}} &:\equiv \mathsf{let}\ \text{⌑}_i = s[\kappa \mid \theta]^{\mathsf{at}\mathfrak{t}} \ \mathsf{in}\ c[\kappa \mid \theta]^{\mathsf{at}\mathfrak{t}}
\end{aligned}$$

A.4.5 Single Slice Substitution

On preslices:

$$\begin{aligned}
(s \otimes \mathfrak{c} \otimes s')[\mathfrak{r}/\mathfrak{c}] &:\equiv s \otimes \mathfrak{r} \otimes s' \\
(s \otimes \mathfrak{c} \otimes s')[\mathfrak{t}'. \prec t/\mathfrak{c}] &:\equiv s \otimes t \otimes s' \\
(s \otimes \mathfrak{c} \otimes s')[\mathfrak{t}'. \prec \varnothing_i/\mathfrak{c}] &:\equiv s \otimes s' \\
s[t/\mathfrak{c}] &:\equiv s & \text{otherwise}
\end{aligned}$$

On actual slices, where we may be binding a new top colour, this bound top colour is maintained:

$$\begin{aligned}
\mathfrak{c}[t/\mathfrak{c}] &:\equiv t \\
(\mathfrak{s}. \prec s)[t/\mathfrak{c}] &:\equiv \mathfrak{t}. \prec s[t/\mathfrak{c}] \\
(\mathfrak{s}. \prec \varnothing_i)[t/\mathfrak{c}] &:\equiv \mathfrak{t}. \prec \varnothing_i
\end{aligned}$$

On terms, we need to be more intelligent: after we have substituted t for $\mathfrak{c}$ once, we do not want use t in the subterm, rather, just the top colour of t.

$$
\begin{aligned}
x\llbracket t/\mathfrak{c}\rrbracket &:\equiv x \\
\\
\left(\textstyle\sum_{(x:A)} B\right)\llbracket t/\mathfrak{c}\rrbracket &:\equiv \textstyle\sum_{(x:A\llbracket t/\mathfrak{c}\rrbracket)} B\llbracket t/\mathfrak{c}\rrbracket \\
(a,b)\llbracket t/\mathfrak{c}\rrbracket &:\equiv (a\llbracket t/\mathfrak{c}\rrbracket, b\llbracket t/\mathfrak{c}\rrbracket) \\
\mathsf{pr}_1(p)\llbracket t/\mathfrak{c}\rrbracket &:\equiv \mathsf{pr}_1(p\llbracket t/\mathfrak{c}\rrbracket) \\
\mathsf{pr}_2(p)\llbracket t/\mathfrak{c}\rrbracket &:\equiv \mathsf{pr}_2(p\llbracket t/\mathfrak{c}\rrbracket) \\
\\
\left(\textstyle\prod_{(x:A)} B\right)\llbracket t/\mathfrak{c}\rrbracket &:\equiv \textstyle\prod_{(x:A\llbracket t/\mathfrak{c}\rrbracket)} B\llbracket t/\mathfrak{c}\rrbracket \\
(\lambda p.b)\llbracket t/\mathfrak{c}\rrbracket &:\equiv (\lambda p.b\llbracket t/\mathfrak{c}\rrbracket) \\
f(a)\llbracket t/\mathfrak{c}\rrbracket &:\equiv f\llbracket t/\mathfrak{c}\rrbracket(a\llbracket t/\mathfrak{c}\rrbracket) \\
\\
(\natural A)\llbracket t/\mathfrak{c}\rrbracket &:\equiv \natural A\llbracket t/\mathfrak{c}\rrbracket \\
(a^{\natural})\llbracket t/\mathfrak{c}\rrbracket &:\equiv (a\llbracket t/\mathfrak{c}\rrbracket)^{\natural} \\
(a_{\natural})\llbracket t/\mathfrak{c}\rrbracket &:\equiv (a\llbracket t/\mathfrak{c}\rrbracket)_{\natural} \\
\\
\left(\textcircled{\scriptsize$\Sigma$}_{(\underline{x}:A)} B\right)\llbracket t/\mathfrak{c}\rrbracket &:\equiv \textcircled{\scriptsize$\Sigma$}_{(\underline{x}:A\llbracket t/\mathfrak{c}\rrbracket)} B\llbracket t/\mathfrak{c}\rrbracket \\
(a\ {}_{\mathfrak{c}}{\otimes}_{s_R}\ b)\llbracket t/\mathfrak{c}\rrbracket &:\equiv a\llbracket \ulcorner t\urcorner/\mathfrak{c}\rrbracket\ {}_{t}{\otimes}_{s_R}\ b \\
(a\ {}_{s_L}{\otimes}_{\mathfrak{c}}\ b)\llbracket t/\mathfrak{c}\rrbracket &:\equiv a\ {}_{s_L}{\otimes}_{t}\ b\llbracket \ulcorner t\urcorner/\mathfrak{c}\rrbracket \\
(a\ {}_{s_L}{\otimes}_{s_R}\ b)\llbracket t/\mathfrak{c}\rrbracket &:\equiv a\llbracket t/\mathfrak{c}\rrbracket\ {}_{s_L}{\otimes}_{s_R}\ b\llbracket t/\mathfrak{c}\rrbracket \\
(\mathsf{let}\ x\ {}_{\mathfrak{l}}{\otimes}_{\mathfrak{r}}\ y = s\ \mathsf{in}\ c)\llbracket t/\mathfrak{c}\rrbracket &:\equiv \mathsf{let}\ x\ {}_{\mathfrak{l}}{\otimes}_{\mathfrak{r}}\ y = s\llbracket t/\mathfrak{c}\rrbracket\ \mathsf{in}\ c\llbracket t/\mathfrak{c}\rrbracket \\
\\
\left(\textcircled{\scriptsize$\prod$}_{(x^{\mathfrak{r}}:A)}{}^{\mathfrak{p}}B\right)\llbracket t/\mathfrak{c}\rrbracket &:\equiv \textcircled{\scriptsize$\prod$}_{(x^{\mathfrak{r}}:A\llbracket t/\mathfrak{c}\rrbracket)}{}^{\mathfrak{p}}B\llbracket t/\mathfrak{c}\rrbracket \\
(\partial^{\mathfrak{p}}x^{\mathfrak{r}}.b)\llbracket t/\mathfrak{c}\rrbracket &:\equiv \partial^{\mathfrak{p}}x^{\mathfrak{r}}.(b\llbracket t/\mathfrak{c}\rrbracket) \\
(f_{\mathfrak{c}}\langle a\rangle_{s_R})\llbracket t/\mathfrak{c}\rrbracket &:\equiv f\llbracket \ulcorner t\urcorner/\mathfrak{c}\rrbracket_{t}\langle a\rangle_{s_R} \\
(f_{s_L}\langle a\rangle_{\mathfrak{c}})\llbracket t/\mathfrak{c}\rrbracket &:\equiv f_{s_L}\langle a\llbracket \ulcorner t\urcorner/\mathfrak{c}\rrbracket\rangle_{t} \\
(f_{s_L}\langle a\rangle_{s_R})\llbracket t/\mathfrak{c}\rrbracket &:\equiv f\llbracket t/\mathfrak{c}\rrbracket_{s_L}\langle a\llbracket t/\mathfrak{c}\rrbracket\rangle_{s_R} \\
\\
\mathbb{S}\llbracket t/\mathfrak{c}\rrbracket &:\equiv \mathbb{S} \\
\text{⌑}_i\llbracket t/\mathfrak{c}\rrbracket &:\equiv \text{⌑}_i \\
(\mathsf{let}\ \text{⌑}_i = s\ \mathsf{in}\ c)\llbracket t/\mathfrak{c}\rrbracket &:\equiv \mathsf{let}\ \text{⌑}_i = s\llbracket t/\mathfrak{c}\rrbracket\ \mathsf{in}\ c\llbracket t/\mathfrak{c}\rrbracket
\end{aligned}
$$

A.4.6 Tensor Merging

$$\mathcal{J}\llbracket \mathfrak{t} \prec s_L \boxtimes s_R / \mathfrak{t}' \rrbracket :\equiv \mathcal{J}\llbracket \mathfrak{t} / \mathfrak{t}' \rrbracket \llbracket s_L / \ulcorner s_L \urcorner \rrbracket \llbracket s_R / \ulcorner s_R \urcorner \rrbracket$$

Appendix B

Proofs of Admissibilty

B.1 Equations on Raw Syntax

The only cases which are less than entirely obvious are equations involving slices, so we check them explicitly.

B.1.1 Marking

Lemma B.1.1. *Marking is idempotent on the level of raw syntax.*

- *If* $\mathsf{t}, \phi \mid \Gamma$ rawctx *then* $\underline{\Gamma} \equiv \underline{\underline{\Gamma}}$
- *If* $\mathsf{t}, \phi \mid \Gamma$ rawctx *and* $\mathsf{t}, \phi \mid \gamma \vdash \psi \mid \Delta$ rawext *then* $\underline{\Gamma, \Delta} \equiv \underline{\underline{\Gamma}, \Delta^{\mathsf{m}\phi|\gamma}}$
- *If* $\phi, \phi', \phi'' \mid \gamma, \gamma' \vdash \psi \mid \Delta$ rawext *then* $(\Delta^{\mathsf{m}\phi|\gamma})^{\mathsf{m}(\phi,\phi'|\gamma,\gamma')} \equiv \Delta^{\mathsf{m}(\phi,\phi'|\gamma,\gamma')} \equiv (\Delta^{\mathsf{m}(\phi,\phi'|\gamma,\gamma')})^{\mathsf{m}\phi'|\gamma'}$
- *If* $\phi, \psi, \psi' \vdash s$ $\mathsf{rawpreslice}$ *or* $\phi, \psi, \psi' \vdash s$ $\mathsf{rawslice}$ *then* $(s^{\mathsf{m}\phi})^{\mathsf{m}(\phi,\psi)} \equiv s^{\mathsf{m}(\phi,\psi)} \equiv (s^{\mathsf{m}(\phi,\psi)})^{\mathsf{m}\phi}$
- *If* $\phi, \psi, \psi' \mid \gamma, \delta \vdash a$ $\mathsf{rawterm}$ *then* $(a^{\mathsf{m}\phi|\gamma})^{\mathsf{m}(\phi,\psi|\gamma,\delta)} \equiv a^{\mathsf{m}(\phi,\psi|\gamma,\delta)} \equiv (a^{\mathsf{m}(\phi,\psi|\gamma,\delta)})^{\mathsf{m}\phi|\gamma}$

In words, zeroing a larger piece of the context subsumes zeroing a smaller piece.

Proof. Induction on the structure of the judgement.

For preslices, $s^{\mathsf{m}\phi}$ is calculated by replacing any colour in ϕ with 1, doing so twice has no additional effect. For slices, the bound colour is not affected by the marking operation and so equality follows by the equation for preslices.

For terms, variable cases are clear, as marked variables always remain marked, and unmarked variable become marked iff they are are in γ, γ'. The remaining cases are immediate by induction. □

B.1.2 Cartesian Substitution

Lemma B.1.2. *Telescope substitution is associative:*

$$\mathcal{J}[\kappa_1 \mid \theta_1]^{\mathsf{att}}[\kappa_2 \mid \theta_2]^{\mathsf{att}} \equiv \mathcal{J}[\kappa_2 \mid \theta_2]^{\mathsf{att}}[\kappa_1[\kappa_2] \mid \theta_1[\kappa_2 \mid \theta_2]]^{\mathsf{att}}$$

Proof. By induction on $\mathcal{J}$, as usual.

For preslices, this follows by considering the case of a single colour. If $\mathfrak{c}$ occurs in the first substitution $[\kappa_1, s/\mathfrak{c}, \kappa_1']$, say, then

$$\mathfrak{c}[\kappa_1, s/\mathfrak{c}, \kappa_1'][\kappa_2] \equiv s[\kappa_2] \equiv \mathfrak{c}[\kappa_1[\kappa_2], s[\kappa_2]/\mathfrak{c}, \kappa_1'[\kappa_2]] \equiv \mathfrak{c}[\kappa_2][\kappa_1[\kappa_2], s[\kappa_2]/\mathfrak{c}, \kappa_1'[\kappa_2]]$$

If it occurs in the second, then

$$\mathfrak{c}[\kappa_1][\kappa_2, s/\mathfrak{c}, \kappa_2'] \equiv \mathfrak{c}[\kappa_2, s/\mathfrak{c}, \kappa_2'] \equiv \mathfrak{c}[\kappa_2, s/\mathfrak{c}, \kappa_2'][\kappa_1]$$

because the colours occurring in s cannot be substituted for by κ_1.

For slices, the bound top colour is unaffected by substitutions. $\square$

Lemma B.1.3. *Substitution and weakening commute:* $\mathcal{J}[\kappa \mid \theta]^{\mathsf{att}} \equiv \mathcal{J}$ *if* κ *and* θ *are fresh for* $\mathcal{J}$.

Proof. By induction. No colours or variables substituted for by κ or θ occur in $\mathcal{J}$, and so at the leaves the substitution has no effect. $\square$

Lemma B.1.4. *Substitution and marking commute:*

- *If* $\phi, \phi' \mid \gamma, \gamma' \vdash (\kappa \mid \theta) : \psi \mid \omega$ *and* $\phi, \phi', \psi, \phi'' \mid \gamma, \gamma', \omega, \gamma'' \vdash \mathcal{J}$ *then*

$$(\mathcal{J}[\kappa \mid \theta]^{\mathsf{att}})^{\mathsf{m}\phi|\gamma} \equiv \mathcal{J}^{\cdot\mathsf{m}\phi|\gamma}[\kappa^{\mathsf{m}\phi} \mid \theta^{\mathsf{m}\phi|\gamma}]^{\mathsf{att}}$$

- *If* $\phi \mid \gamma \vdash (\kappa \mid \theta) : \psi \mid \omega$ *and* $\phi, \psi, \phi' \mid \gamma, \omega, \gamma' \vdash \mathcal{J}$ *then*

$$(\mathcal{J}[\kappa \mid \theta]^{\mathsf{att}})^{\mathsf{m}\phi,\phi'|\gamma,\gamma'} \equiv \mathcal{J}^{\cdot\mathsf{m}\phi,\psi,\phi'|\gamma,\omega,\gamma'}[\kappa \mid \theta]^{\mathsf{att}}$$

Note that in the second, the substitution does not need to be marked on the right-hand side.

Proof. By induction. For the second, any variable in ω is marked in $\mathcal{J}^{\cdot\mathsf{m}\phi,\psi,\phi'|\gamma,\omega,\gamma'}$, and so all terms in θ have the marking operation applied to them at the leaves of $\mathcal{J}$, by the definition of substitution for marked variables. $\square$

Lemma B.1.5. *Substitution and recolouring commute: if* $\mathfrak{t}, \phi \mid \gamma \vdash (\kappa \mid \theta) : \psi \mid \omega$ *and* $\mathfrak{t}, \phi, \psi, \phi' \mid \gamma, \omega, \gamma' \vdash \mathcal{J}$, *then*

$$\mathcal{J}^{\cdot\mathfrak{c}\leftrightarrow\mathfrak{t}}[\kappa \mid \theta]^{\mathsf{atc}} \equiv (\mathcal{J}[\kappa \mid \theta]^{\mathsf{att}})^{\mathfrak{c}\leftrightarrow\mathfrak{t}}$$

Proof. On slices, this follows for individual colours by

$$\mathfrak{t}^{\mathfrak{c}\leftrightarrow\mathfrak{t}}[\kappa] \equiv \mathfrak{c}[\kappa] \equiv \mathfrak{c} \equiv \mathfrak{t}^{\mathfrak{c}\leftrightarrow\mathfrak{t}} \equiv \mathfrak{t}[\kappa]^{\mathfrak{c}\leftrightarrow\mathfrak{t}}$$

and this is easily extended to the other judgements. $\square$

Lemma B.1.6. *Telescope and slice substitution commute: if* $\mathfrak{t}, \phi \mid \gamma \vdash (\kappa \mid \theta) : \psi \mid \omega$ *and* $\phi, \psi, \phi' \vdash \mathcal{J}$ *and* $\phi, \psi, \phi' \vdash s$ slice *with* $\mathfrak{c} \notin \psi$, *then*

$$\mathcal{J}[\![t/\mathfrak{c}]\!][\kappa \mid \theta]^{\mathsf{att}} \equiv \mathcal{J}[\kappa \mid \theta]^{\mathsf{att}}[\![t[\kappa]/\mathfrak{c}]\!]$$

Proof. Similar to Lemma B.1.2. On the colour $\mathfrak{c}$ (in a preslice) we see

$$\mathfrak{c}⟦t/\mathfrak{c}⟧[\kappa] \equiv t[\kappa] \equiv \mathfrak{c}⟦t[\kappa]/\mathfrak{c}⟧ \equiv \mathfrak{c}[\kappa]⟦t[\kappa]/\mathfrak{c}⟧$$

where $\mathfrak{c} \equiv \mathfrak{c}[\kappa]$ because $\mathfrak{c}$ is not in ψ.

On terms, the interesting cases are the splitting terms where this slice substitution is handed specially. For these:

$$\begin{aligned}(a\ {}_{\mathfrak{c}}{\otimes}_{s_R}\ b)⟦t/\mathfrak{c}⟧[\kappa \mid \theta]^{\mathsf{att}} &\equiv (a\ {}_{t}{\otimes}_{s_R}\ b)[\kappa \mid \theta]^{\mathsf{att}} \\ &\equiv a[\kappa \mid \theta]^{\mathsf{att}}\ {}_{t[\kappa]}{\otimes}_{s_R[\kappa]}\ b[\kappa \mid \theta]^{\mathsf{att}} \\ &\equiv (a[\kappa \mid \theta]^{\mathsf{att}}\ {}_{\mathfrak{c}}{\otimes}_{s_R[\kappa]}\ b[\kappa \mid \theta]^{\mathsf{att}})⟦t[\kappa]/\mathfrak{c}⟧\end{aligned}$$

using the assumption that $\mathfrak{c}$ is not in ψ. The case of $a\ {}_{s_L}{\otimes}_{\mathfrak{c}}\ b$ is symmetric, and the other cases are standard. □

Lemma B.1.7. *Substitution and merging commute:* $\mathcal{J}⟦\mathsf{t} \prec s_L \boxtimes s_R/\mathsf{t}'⟧[\kappa \mid \theta]^{\mathsf{att}} \equiv \mathcal{J}[\kappa \mid \theta]^{\mathsf{att}'}⟦\mathsf{t} \prec s_L[\kappa] \boxtimes s_R[\kappa]/\mathsf{t}'⟧$

Proof. By expanding the definition of $⟦\mathsf{t} \prec s_L \boxtimes s_R/\mathsf{t}'⟧$ and applying Lemma B.1.5 and Lemma B.1.6. □

B.1.3 Recolouring

Lemma B.1.8. $(\mathcal{J}^{\mathfrak{d}\leftrightarrow\mathfrak{c}})^{\mathfrak{e}\leftrightarrow\mathfrak{d}} \equiv \mathcal{J}^{\mathfrak{e}\leftrightarrow\mathfrak{c}}$

Proof. On slices this is clear because $(\mathfrak{c}^{\mathfrak{d}\leftrightarrow\mathfrak{c}})^{\mathfrak{e}\leftrightarrow\mathfrak{d}} \equiv \mathfrak{d}^{\mathfrak{e}\leftrightarrow\mathfrak{d}} \equiv \mathfrak{e} \equiv \mathfrak{c}^{\mathfrak{e}\leftrightarrow\mathfrak{c}}$, then on all other judgements by induction. □

Lemma B.1.9. $(\mathcal{J}^{\mathfrak{d}\leftrightarrow\mathfrak{c}})^{\mathsf{m}\phi|\gamma} \equiv (\mathcal{J}^{\mathsf{m}\phi|\gamma})^{\mathfrak{d}\leftrightarrow\mathfrak{c}}$ *if* $\mathfrak{c} \notin \phi$.

Proof. By induction. On the colour

$$(\mathfrak{c}^{\mathfrak{d}\leftrightarrow\mathfrak{c}})^{\mathsf{m}\phi|\gamma} \equiv \mathfrak{d}^{\mathsf{m}\phi|\gamma} \equiv \mathfrak{d} \equiv \mathfrak{c}^{\mathfrak{d}\leftrightarrow\mathfrak{c}} \equiv (\mathfrak{c}^{\mathsf{m}\phi|\gamma})^{\mathfrak{d}\leftrightarrow\mathfrak{c}}$$

because neither $\mathfrak{c}$ or $\mathfrak{d}$ are in ϕ. □

Lemma B.1.10. *Weakening and recolouring commute:* $\mathcal{J}^{\mathfrak{d}\leftrightarrow\mathfrak{c}} \equiv \mathcal{J}$ *if* $\mathfrak{c}$ *is fresh for* $\mathcal{J}$.

Proof. By induction. □

B.1.4 Slice Substitution and Merging

Lemma B.1.11. $\mathcal{J}⟦s/\mathfrak{s}⟧⟦t/\mathfrak{t}⟧ \equiv \mathcal{J}⟦t/\mathfrak{t}⟧⟦s⟦t/\mathfrak{t}⟧/\mathfrak{s}⟧$

Proof. By induction. □

Lemma B.1.12. $\mathcal{J}⟦s/\mathfrak{c}⟧ \equiv \mathcal{J}$ *if* $\mathfrak{c}$ *is fresh for* $\mathcal{J}$

Proof. By induction. □

Lemma B.1.13. $(\mathcal{I}^{\mathsf{m}\Phi|\Gamma})\llbracket s/\ulcorner s\urcorner\rrbracket \equiv \mathcal{I}^{\mathsf{m}\Phi|\Gamma}$ *if* $\mathfrak{t} \prec \Phi \vdash s$ slice.

Proof. If $\ulcorner s\urcorner \in \Phi$ then this is Lemma B.1.12, because $\ulcorner s\urcorner$ does not occur in $\mathcal{I}^{\mathsf{m}\Phi|\Gamma}$. The remaining case is $s \equiv \mathfrak{t}$, in which case we are calculating $(\mathcal{I}^{\mathsf{m}\Phi|\Gamma})\llbracket \mathfrak{t}/\mathfrak{t}\rrbracket \equiv \mathcal{I}^{\mathsf{m}\Phi|\Gamma}$. □

Lemma B.1.14. $(\mathcal{I}\llbracket s/\mathfrak{c}\rrbracket)^{\mathfrak{d}\leftrightarrow\mathfrak{e}} \equiv (\mathcal{I}^{\mathfrak{d}\leftrightarrow\mathfrak{e}})\llbracket s^{\mathfrak{d}\leftrightarrow\mathfrak{e}}/\mathfrak{c}\rrbracket$

Proof. By induction. □

Lemma B.1.15. $(\mathcal{I}^{\mathfrak{d}\leftrightarrow\mathfrak{e}})\llbracket s/\mathfrak{c}\rrbracket \equiv (\mathcal{I}\llbracket s/\mathfrak{c}\rrbracket)^{\mathfrak{d}\leftrightarrow\mathfrak{e}}$

Proof. By induction. □

Lemma B.1.16.

$$\begin{aligned}&\mathcal{I}\llbracket \mathfrak{s} \prec s_L \boxtimes s_R/\mathfrak{s}'\rrbracket\llbracket \mathfrak{t} \prec t_L \boxtimes t_R/\mathfrak{t}'\rrbracket\\ &\equiv \mathcal{I}\llbracket \mathfrak{t} \prec t_L \boxtimes t_R/\mathfrak{t}'\rrbracket\llbracket \mathfrak{s} \prec s_L\llbracket \mathfrak{t} \prec t_L \boxtimes t_R/\mathfrak{t}'\rrbracket \boxtimes s_R\llbracket \mathfrak{t} \prec t_L \boxtimes t_R/\mathfrak{t}'\rrbracket/\mathfrak{s}'^{\mathfrak{t}\leftrightarrow\mathfrak{t}'}\rrbracket\end{aligned}$$

Proof. By Lemma B.1.11, Lemma B.1.15 and Lemma B.1.8. □

Lemma B.1.17. $(\mathcal{I}^{\mathsf{m}\Phi|\Gamma})\llbracket \mathfrak{t} \prec s_L \boxtimes s_R/\mathfrak{t}'\rrbracket \equiv (\mathcal{I}^{\mathsf{m}\Phi|\Gamma})^{\mathfrak{t}\leftrightarrow\mathfrak{t}'}$ *if* $\mathfrak{t}' \prec \Phi \vdash s_L$ slice *and* $\mathfrak{t}' \prec \Phi \vdash s_R$ slice.

Proof. Apply Lemma B.1.13 twice. □

Lemma B.1.18. $(\mathcal{I}\llbracket \mathfrak{t} \prec s_L \boxtimes s_R/\mathfrak{t}'\rrbracket)^{\mathfrak{d}\leftrightarrow\mathfrak{c}} \equiv (\mathcal{I}\llbracket \mathfrak{t}^{\mathfrak{d}\leftrightarrow\mathfrak{c}} \prec s_L{}^{\mathfrak{d}\leftrightarrow\mathfrak{c}} \boxtimes s_R{}^{\mathfrak{d}\leftrightarrow\mathfrak{c}}/\mathfrak{t}'\rrbracket)$

Proof. Apply Lemma B.1.14 twice. □

Lemma B.1.19. $(\mathcal{I}^{\mathfrak{d}\leftrightarrow\mathfrak{c}})\llbracket \mathfrak{t} \prec s_L \boxtimes s_R/\mathfrak{t}'\rrbracket \equiv (\mathcal{I}\llbracket \mathfrak{t} \prec s_L \boxtimes s_R/\mathfrak{t}'\rrbracket)^{\mathfrak{d}^{\mathfrak{t}\leftrightarrow\mathfrak{t}'}\leftrightarrow\mathfrak{c}^{\mathfrak{t}\leftrightarrow\mathfrak{t}'}})$ *if* $\mathfrak{t}' \prec \Phi \vdash s_L$ slice *and* $\mathfrak{t}' \prec \Phi \vdash s_R$ slice.

Proof. By Lemma B.1.15 and Lemma B.1.8. □

Lemma B.1.20. $\underline{\mathcal{I}}\llbracket \mathfrak{t} \prec s_L \boxtimes s_R/\mathfrak{t}'\rrbracket \equiv \underline{\mathcal{I}\llbracket \mathfrak{t} \prec s_L \boxtimes s_R/\mathfrak{t}'\rrbracket}$

Proof. By Lemma B.1.13 and Lemma B.1.9. □

B.1.5 Context Filtering

Lemma B.1.21. *If* $\phi \mid \gamma \vdash \psi \mid \Gamma'$ rawext *and* $\Xi\{\Phi\} \vdash s$ slice *then* $(\Gamma'^{s})^{\mathsf{m}\phi|\gamma} \equiv (\Gamma'^{\mathsf{m}\phi|\gamma})^{s^{\mathsf{m}\phi}}$

Proof. This can be checked one variable at a time. Marked variables are left the same by both operations. For an unmarked variable $x^{\mathfrak{c}} : A$,

- If $\mathfrak{c} \notin \Phi$ and $\mathfrak{c} \notin s$ then x is not marked by any operations on either side.
- If $\mathfrak{c} \notin \Phi$ and $\mathfrak{c} \in s$ then also $\mathfrak{c} \in s^{\mathsf{m}\phi}$, and so x is marked by $-^{s}$ on the left and marked by $-^{s^{\mathsf{m}\phi}}$ on the right
- If $\mathfrak{c} \in \Phi$ then x is marked by $-^{\mathsf{m}\phi|\gamma}$ on both sides.

□

Lemma B.1.22. *If* $\mathfrak{t}, \phi \mid \Gamma$ rawctx *and* $\mathfrak{t} \prec \Phi \vdash s$ slice *then* $\underline{\Gamma}^s \equiv \underline{\Gamma}$ *and* $\underline{\Gamma^s} \equiv \underline{\Gamma}$.

Proof. This can be checked one variable at a time. In all cases, all variables in Γ are marked on both sides. □

Lemma B.1.23. *Substitution and context filtering commute on context extensions: for* $\mathfrak{t}, \phi \mid \gamma \vdash (\kappa \mid \theta) : \psi \mid \omega$ *and* $\phi, \psi \mid \gamma, \omega \vdash \xi \mid \Gamma'$ rawext *and* $\Xi\{\Phi, \Psi\} \vdash s$ slice *such that* Γ' *does not use colours from* ϕ *or* ψ*,*

$$\Gamma'^s[\kappa \mid \theta] \equiv (\Gamma'[\kappa \mid \theta])^{s[\kappa]}$$

Proof. Can be checked one variable at at time. Marked variables in Γ' are always marked on both sides, with the types unchanged. For an unmarked variable $x^{\mathfrak{c}} : A$ with $\mathfrak{c} \in s$, also $\mathfrak{c} \in s[\kappa]$ by the assumption that $\mathfrak{c} \notin \psi$, and so x becomes marked on both sides. If $\mathfrak{c} \notin s$, then $\mathfrak{c} \notin s[\kappa]$ for the same reason, and then x remains unmarked on both sides. □

Lemma B.1.24. *Merging and context filtering commute on context extensions:*

$$\Gamma'^t[\![\mathfrak{t} \prec s_L \boxtimes s_R / \mathfrak{t}')]\!] \equiv (\Gamma'[\![\mathfrak{t} \prec s_L \boxtimes s_R / \mathfrak{t}')]\!])^{t[\![\mathfrak{t} \prec s_L \boxtimes s_R / \mathfrak{t}')]\!]}$$

Proof. Can similarly be checked one variable at at time. □

B.2 Palettes

Lemma B.2.1. *If* $\Xi\{\Phi\}$ spot *and* $\Xi \vdash s$ slice*, then exactly one of the following holds:*

- *Case 1: s contains all of Φ, so* $\Xi^s\{\Phi\}$ spot
- *Case 2: s intersects Φ, and there is* $\Phi \vdash s \cap \Phi$ $\mathsf{preslice}$ *so that* $\Xi^s\{\Phi^{s \cap \Phi}\}$ spot.
- *Case 3: s and Φ are disjoint, and* $\Xi\{\downarrow\Psi\} \vdash s$ slice *for any palette Ψ.*

Proof. Induction on the spot. □

Lemma B.2.2. *If* $\Xi\{\Phi\}$ spot *and* $\Xi \vdash s$ slice *contains Φ. Then* $(\Xi\{\downarrow\Psi\})^s \equiv \Xi^s\{\downarrow\Psi\}$

Proof. By induction on the spot, and the definition of the spot $\Xi^s\{\Phi\}$ spot given in Lemma B.2.1. □

Lemma B.2.3. *If* $\Xi\{\Phi, \Psi\}$ spot *and* $\Xi \vdash s$ slice *intersects Φ. Then* $(\Xi\{\Phi, \Psi\})^s \equiv (\Xi\{\downarrow\Phi\})^s$

Proof. By induction on the spot. Once we reach $\Phi, \Psi \vdash s$ slice we must have $\Phi \vdash s$ $\mathsf{preslice}_\epsilon$, and then $(\Phi, \Psi)^s \equiv \Phi^s$. □

Lemma B.2.4. *Symmetry and associativity of the palette are admissible.*

$$\frac{\Xi\{\Phi_1, \Phi_2\}\ \mathsf{spot} \quad \Xi\{\Phi_1, \Phi_2\} \vdash \mathcal{J}}{\Xi\{\downarrow\Phi_2, \Phi_1\} \vdash \mathcal{J}} \qquad \frac{\Xi\{\Phi_L \otimes \Phi_R\}\ \mathsf{spot} \quad \Xi\{\Phi_L \otimes \Phi_R\} \vdash \mathcal{J}}{\Xi\{\downarrow\Phi_R \otimes \Phi_L\} \vdash \mathcal{J}}$$

$$\frac{\Xi\{(\Phi_1, \Phi_2), \Phi_3\}\ \mathsf{spot} \quad \Xi\{(\Phi_1, \Phi_2), \Phi_3\} \vdash \mathcal{J}}{\Xi\{\downarrow\Phi_1, (\Phi_2, \Phi_3)\} \vdash \mathcal{J}} \qquad \frac{\Xi\{\Phi_1, (\Phi_2, \Phi_3)\}\ \mathsf{spot} \quad \Xi\{\Phi_1, (\Phi_2, \Phi_3)\} \vdash \mathcal{J}}{\Xi\{\downarrow(\Phi_1, \Phi_2), \Phi_3\} \vdash \mathcal{J}}$$

$$\frac{\Xi\{(\Phi_{LL} \otimes \Phi_{LR}) \otimes \Phi_R\}\ \mathsf{spot} \quad \Xi\{(\Phi_{LL} \otimes \Phi_{LR}) \otimes \Phi_R\} \vdash \mathcal{J}}{\Xi\{\downarrow\Phi_{LL} \otimes (\Phi_{LR} \otimes \Phi_R)\} \vdash \mathcal{J}} \qquad \frac{\Xi\{\Phi_{LL} \otimes (\Phi_{LR} \otimes \Phi_R)\}\ \mathsf{spot} \quad \Xi\{\Phi_{LL} \otimes (\Phi_{LR} \otimes \Phi_R)\} \vdash \mathcal{J}}{\Xi\{\downarrow(\Phi_{LL} \otimes \Phi_{LR}) \otimes \Phi_R\} \vdash \mathcal{J}}$$

Proof. For each, induction on the spot followed by induction on $\mathcal{J}$ when the spot is reached. □

Lemma B.2.5. *Slices can be palette-weakened.*

$$\frac{\Phi \vdash s\ \mathsf{preslice} \quad \Xi\{\Phi\}\ \mathsf{spot}}{\Xi\{\Phi\} \vdash s\ \mathsf{preslice}} \qquad \frac{\Phi \vdash s\ \mathsf{preslice}_\epsilon \quad \Xi\{\Phi\}\ \mathsf{spot}}{\Xi\{\Phi\} \vdash s\ \mathsf{preslice}_\epsilon} \qquad \frac{\Phi \vdash s\ \mathsf{slice} \quad \Xi\{\Phi\}\ \mathsf{spot}}{\Xi\{\Phi\} \vdash s\ \mathsf{slice}}$$

Proof. Induction on the spot. □

Lemma B.2.6. *Preslices in a tensor palette can be decomposed.*

$$\frac{\Phi_L \vdash s_L\ \mathsf{slice} \quad \Phi_R \vdash s_R\ \mathsf{slice} \quad (\Phi_L \otimes \Phi_R)^{s_L \otimes s_R} \vdash t\ \mathsf{preslice}_\epsilon}{{\Phi_L}^{s_L} \vdash t|_{s_L}\ \mathsf{preslice}_\epsilon \quad {\Phi_R}^{s_R} \vdash t|_{s_R}\ \mathsf{preslice}_\epsilon \quad t \equiv t|_{s_L} \otimes t|_{s_R}}$$

Proof. We do this in a few stages, first proving the simpler rules

$$\frac{\Phi_L \vdash s_L\ \mathsf{preslice} \quad \Phi_R \vdash s_R\ \mathsf{preslice} \quad (\Phi_L \otimes \Phi_R)^{s_L \otimes s_R} \vdash t\ \mathsf{preslice}}{{\Phi_L}^{s_L} \vdash t|_{s_L}\ \mathsf{preslice} \quad {\Phi_R}^{s_R} \vdash t|_{s_R}\ \mathsf{preslice} \quad t \equiv t|_{s_L} \otimes t|_{s_R}}$$

$$\frac{\Phi_L \vdash s_L\ \mathsf{preslice}_\epsilon \quad \Phi_R \vdash s_R\ \mathsf{preslice}_\epsilon \quad (\Phi_L \otimes \Phi_R)^{s_L \otimes s_R} \vdash t\ \mathsf{preslice}_\epsilon}{{\Phi_L}^{s_L} \vdash t|_{s_L}\ \mathsf{preslice}_\epsilon \quad {\Phi_R}^{s_R} \vdash t|_{s_R}\ \mathsf{preslice} \quad t \equiv t|_{s_L} \otimes t|_{s_R}}$$

before the one given in the statement.

- **Pure preslices:** If neither of s_L and s_R are equal to $\varnothing$, then $(\Phi_L \otimes \Phi_R)^{s_L \otimes s_R} \equiv {\Phi_L}^{s_L} \otimes {\Phi_R}^{s_R}$. And so t itself must be of the form $t_L \otimes t_R$, so we define $t|_{s_L} :\equiv t_L$ and $t|_{s_R} :\equiv t_R$.

  If $s_R \equiv \varnothing$, then $(\Phi_L \otimes \Phi_R)^{s_L \otimes s_R} \equiv {\Phi_L}^{s_L}$, in which case we can take $t|_{s_L} :\equiv t$ and $t|_{s_R} :\equiv \varnothing$. The other case follows by symmetry.

- **Preslices:** If s_L is of the form $s'_L \otimes 1$ and s_R does not contain 1, then by definition $(\Phi_L \otimes \Phi_R)^{s_L \otimes s_R} \equiv (\Phi_L \otimes \Phi_R)^{s'_L \otimes s_R} \otimes 1$. Because the 1 subpalette contains no colour names, we must have $(\Phi_L \otimes \Phi_R)^{s'_L \otimes s_R} \vdash t \ \mathsf{preslice}_\epsilon$, and we can apply the previous version of the rule for pure preslices. The remaining cases where 1 occurs in s_R, or in both s_L and s_R, are similar.

- **Slices:** The only difference is that s_L and s_R may bind new top colours and t may mention these top colours.

  If it does not mention either then we have $(\Phi_L \otimes \Phi_R)^{s_L \otimes s_R} \vdash t \ \mathsf{preslice}_\epsilon$ and we can apply the previous version.

  If t is of the form $\ulcorner s_L \urcorner \otimes t_R$ for ${\Phi_R}^{s_R} \vdash t_R \ \mathsf{preslice}_\epsilon$, then certainly also $\Phi_L^{s_L} \vdash \ulcorner s_L \urcorner \ \mathsf{preslice}_\epsilon$. The other cases are similar.

□

Lemma B.2.7. *Presplits in a tensor palette can be decomposed.*

$$\frac{\Phi_L \vdash s_L \ \mathsf{preslice} \qquad \Phi_R \vdash s_R \ \mathsf{preslice} \qquad (\Phi_L \otimes \Phi_R)^{s_L \otimes s_R} \vdash t_L \boxtimes t_R \ \mathsf{presplit}}{{\Phi_L}^{s_L} \vdash t_L|_{s_L} \boxtimes t_R|_{s_L} \ \mathsf{presplit} \qquad {\Phi_R}^{s_R} \vdash t_L|_{s_R} \boxtimes t_R|_{s_R} \ \mathsf{presplit}}$$

$$\frac{\Phi_L \vdash s_L \ \mathsf{preslice}_\epsilon \qquad \Phi_R \vdash s_R \ \mathsf{preslice}_\epsilon \qquad (\Phi_L \otimes \Phi_R)^{s_L \otimes s_R} \vdash t_L \boxtimes t_R \ \mathsf{presplit}}{{\Phi_L}^{s_L} \vdash t_L|_{s_L} \boxtimes t_R|_{s_L} \ \mathsf{presplit} \qquad {\Phi_R}^{s_R} \vdash t_L|_{s_R} \boxtimes t_R|_{s_R} \ \mathsf{presplit}}$$

Proof. First, for s_L and s_R pure preplits, we check the same cases as the previous lemma. If neither of s_L and s_R are $\varnothing$, then $(\Phi_L \otimes \Phi_R)^{s_L \otimes s_R} \equiv {\Phi_L}^{s_L} \otimes {\Phi_R}^{s_R}$, and we must have $t_L \equiv t_{LL} \otimes t_{RL}$ and $t_R \equiv t_{LR} \otimes t_{RR}$ with

$$\begin{gathered} {\Phi_L}^{s_L} \vdash t_{LL} \boxtimes t_{LR} \ \mathsf{presplit} \\ {\Phi_R}^{s_R} \vdash t_{RL} \boxtimes t_{RR} \ \mathsf{presplit} \end{gathered}$$

But by definition of the decomposition in this case, $t_L|_{S_L} \equiv t_{LL}$, and similarly for the others, so we have the required presplit.

If $s_R \equiv \varnothing$, then $(\Phi_L \otimes \Phi_R)^{s_L \otimes s_R} \equiv {\Phi_L}^{s_L}$ and by definition $t_L|_{s_R} \equiv t_R|_{s_R} \equiv \varnothing$ and $t_L|_{s_L} \equiv t_L$ and $t_R|_{s_L} \equiv t_R$. By assumption ${\Phi_L}^{s_L} \vdash t_L \boxtimes t_R \ \mathsf{presplit}$, which is the required split on the left. For the split on the right, each of $t_L|_{s_R}$ and $t_R|_{s_R}$ are either $\varnothing$ or 1, and any combination of such is a valid presplit in palette $\varnothing \ \mathsf{palette}$.

If s_L or s_R contain 1, let s'_L and s'_R denote the underlying pure presplits. We can apply Lemma B.2.12 to see $(\Phi_L \otimes \Phi_R)^{s'_L \otimes s'_R} \vdash t_L \boxtimes t_R \ \mathsf{presplit}$, and then apply the version for pure presplits, weakening with 1 afterwards if necessary. □

Lemma B.2.8. *Slices compose:*

$$\frac{\Phi \vdash s \ \textsf{preslice} \qquad \Phi^s \vdash t \ \textsf{preslice}}{\Phi \vdash t \ \textsf{preslice}} \qquad\qquad \frac{\Phi \vdash s \ \textsf{preslice}_\epsilon \qquad \Phi^s \vdash t \ \textsf{slice}}{\Phi \vdash t \ \textsf{slice}}$$

Proof. For the rule on the left, induction on s:

- $\varnothing$ $\textsf{preslice}$: The only pure preslice is $t \equiv \varnothing$, which is well-formed in the conclusion.
- $\mathfrak{c} \prec \Phi \vdash \mathfrak{c}$ $\textsf{preslice}$: Because $(\mathfrak{c} \prec \Phi)^{\mathfrak{c}} \equiv (\mathfrak{c} \prec \Phi)$, the conclusion is well-formed.
- $\mathfrak{c} \prec \Phi \vdash s$ $\textsf{preslice}$ where $\Phi \vdash s$ $\textsf{preslice}$: We have $(\mathfrak{c} \prec \Phi)^s \equiv \Phi^s$, and so $\Phi^s \vdash t$ $\textsf{preslice}$. By induction then, $\Phi \vdash t$ $\textsf{preslice}$ and so also $\mathfrak{c} \prec \Phi \vdash t$ $\textsf{preslice}$.
- $\Phi_1, \Phi_2 \vdash s$ $\textsf{preslice}$ where $\Phi_1 \vdash s$ $\textsf{preslice}$: Then $(\Phi_1, \Phi_2)^s \equiv {\Phi_1}^s$, and so ${\Phi_1}^s \vdash t$ $\textsf{preslice}$. By induction $\Phi_1 \vdash t$ $\textsf{preslice}$, and so $\Phi_1, \Phi_2 \vdash t$ $\textsf{preslice}$
- $\Phi_1, \Phi_2 \vdash s$ $\textsf{preslice}$ where $\Phi_2 \vdash s$ $\textsf{preslice}$: Follows by symmetry.
- $\Phi_L \otimes \Phi_R \vdash s_L \otimes s_R$ $\textsf{preslice}$ where $\Phi_L \vdash s_L$ $\textsf{preslice}$ and $\Phi_R \vdash s_R$ $\textsf{preslice}$: By Lemma B.2.6 there are

  $$\begin{gathered} {\Phi_L}^{s_L} \vdash t|_{s_L} \ \textsf{preslice} \\ {\Phi_R}^{s_R} \vdash t|_{s_R} \ \textsf{preslice} \end{gathered}$$

  and so inductively

  $$\begin{gathered} \Phi_L \vdash t|_{s_L} \ \textsf{preslice} \\ \Phi_R \vdash t|_{s_R} \ \textsf{preslice} \end{gathered}$$

  from which we can form $\Phi_L \otimes \Phi_R \vdash t|_{s_L} \otimes t|_{s_R}$ $\textsf{preslice}$, and finally $t|_{s_L} \otimes t|_{s_R} \equiv t$ so we are done.

For the rule on the right, the presence of 1 in s has no effect on the possibilities for t, so we just have to check the new possibilities for t:

- $t \equiv \mathfrak{c}$: Then the colour $\mathfrak{c}$ must occur in Φ, so also $\Phi \vdash \mathfrak{c}$ $\textsf{slice}$.
- $t \equiv \mathfrak{t}. \prec t' \otimes \epsilon$ for $\Phi^s \vdash t' \otimes \epsilon$ $\textsf{preslice}_\epsilon$: Then we can apply the rule for pure preslices to t' giving $\Phi \vdash t'$ $\textsf{preslice}$, and then re-form $\Phi \vdash \mathfrak{t}. \prec t' \otimes \epsilon$ $\textsf{slice}$.
- $t \equiv \mathfrak{t}. \prec \varnothing_{i.}$: Then also $\Phi \vdash \mathfrak{t}. \prec \varnothing_{i.}$ because this slice can be formed in any palette.

□

Lemma B.2.9. *Slices 'compose': $(\Phi^s)^t \equiv \Phi^t$ for $\Phi \vdash s$ $\textsf{preslice}_\epsilon$ and $\Phi^s \vdash t$ $\textsf{slice}$.*

Proof. On pure preslices, this follows by inspecting each case of Lemma B.2.8. The only interesting case is $\Phi_L \otimes \Phi_R \vdash s_L \otimes s_R \ \mathsf{preslice}$. Suppose neither of s_1 and s_2 are $\varnothing$. Inductively,

$$\begin{aligned}({\Phi_L}^{s_L})^{t|_{s_L}} &\equiv {\Phi_L}^{t|_{s_L}} \\ ({\Phi_R}^{s_R})^{t|_{s_R}} &\equiv {\Phi_R}^{t|_{s_R}}\end{aligned}$$

and so

$$\begin{aligned}((\Phi_L \otimes \Phi_R)^{s_L \otimes s_R})^t &\equiv ((\Phi_L \otimes \Phi_R)^{s_L \otimes s_R})^{t|_{s_L} \otimes t|_{s_R}} \\ &\equiv ({\Phi_L}^{s_L} \otimes {\Phi_R}^{s_R})^{t|_{s_L} \otimes t|_{s_R}} \\ &\equiv ({\Phi_L}^{s_L})^{t|_{s_L}} \otimes ({\Phi_R}^{s_R})^{t|_{s_R}} \\ &\equiv {\Phi_L}^{t|_{s_L}} \otimes {\Phi_R}^{t|_{s_R}} \\ &\equiv (\Phi_L \otimes \Phi_R)^{t|_{s_L} \otimes t|_{s_R}} \\ &\equiv (\Phi_L \otimes \Phi_R)^t\end{aligned}$$

For slices, again we check the possibilities for t:

- $t \equiv \mathfrak{c}$: We have $(\Phi^s)^{\mathfrak{c}} \equiv \Phi^{\mathfrak{c}}$, because restricting to the preslice s does not change the subpalette with top colour $\mathfrak{c}$.
- $t \equiv \mathfrak{t}. \prec t' \otimes \epsilon$ for $\Phi^s \vdash t' \otimes \epsilon \ \mathsf{preslice}_\epsilon$: Follows from the version for pure preslices.
- $t \equiv \mathfrak{t}. \prec \varnothing_{i.}$: Follows because restricting to the slice $\mathfrak{t}. \prec \varnothing_{i.}$ discards the palette entirely.

□

Lemma B.2.10. *Splits are symmetric:*

$$\frac{\Phi \vdash s_L \boxtimes s_R \ \mathsf{presplit}}{\Phi \vdash s_R \boxtimes s_L \ \mathsf{presplit}} \qquad \frac{\Phi \vdash s_L \boxtimes s_R \ \mathsf{split}}{\Phi \vdash s_R \boxtimes s_L \ \mathsf{split}}$$

Proof. By induction. □

Lemma B.2.11. *Splits may be weakened with* 1*:*

$$\frac{\Phi \vdash s_L \boxtimes s_R \ \mathsf{presplit}}{\Phi \vdash (s_L \otimes 1) \boxtimes s_R \ \mathsf{presplit}}$$

Proof. Easy induction. The ϵ flag only has an effect in the $\Phi \vdash (\varnothing \otimes \epsilon_L) \boxtimes (\varnothing \otimes \epsilon_R) \ \mathsf{presplit}$ case, and the rule can be reapplied regardless of what ϵ_L was in the premise. □

Lemma B.2.12. *A slice or split in a palette containing* 1 *can be strengthened:*

$$\frac{\Phi \otimes 1 \vdash s \ \mathsf{preslice}_\epsilon}{\Phi \vdash s \ \mathsf{preslice}_\epsilon} \qquad \frac{\Phi \otimes 1 \vdash s_L \boxtimes s_R \ \mathsf{presplit}}{\Phi \vdash s_L \boxtimes s_R \ \mathsf{presplit}}$$

Proof. The preslice is formed as $s \equiv s_L \otimes s_R$ with two preslices

$$\Phi \vdash s_L \ \mathsf{preslice}_\epsilon$$
$$1 \vdash s_R \ \mathsf{preslice}_\epsilon$$

The only preslices in palette 1 are $\varnothing$ and 1, and so either $s \equiv s_L$ or $s \equiv s_L \otimes 1$. In either case we have $\Phi \vdash s_L \ \mathsf{preslice}_\epsilon$ or $\Phi \vdash s_L \ \mathsf{preslice}_\epsilon$.

The presplit is formed from the two presplits

$$\Phi \vdash s_{LL} \boxtimes s_{LR} \ \mathsf{presplit}$$
$$1 \vdash s_{RL} \boxtimes s_{RR} \ \mathsf{presplit}$$

Again each of s_{RL} and s_{RR} are either $\varnothing$ or 1. In any case, we also have $\Phi \vdash (s_{LL} \otimes s_{RL}) \boxtimes (s_{LR} \otimes s_{RR}) \ \mathsf{presplit}$ by applying Lemma B.2.11 if necessary. □

Lemma B.2.13. *Splits can be 'interchanged':*

$$\frac{\Phi \vdash s_L \boxtimes s_R \ \mathsf{presplit} \qquad \Phi^{s_L} \vdash t_{LL} \boxtimes t_{LR} \ \mathsf{presplit} \qquad \Phi^{s_R} \vdash t_{RL} \boxtimes t_{RR} \ \mathsf{presplit}}{\Phi \vdash (t_{LL} \otimes t_{RL}) \boxtimes (t_{LR} \otimes t_{RR}) \ \mathsf{presplit}}$$

Proof. Induction on the outer presplit.

- $\varnothing \vdash (\varnothing \otimes \epsilon_L) \boxtimes (\varnothing \otimes \epsilon_R)$: Each $t_?$ must have $\varnothing$ as the underlying pure presplit because $\Phi^{\varnothing \otimes \epsilon_L}$ is equal to $\varnothing$ or 1, and so we have

  $$\varnothing \vdash (\varnothing \otimes \epsilon_{LL} \otimes \epsilon_{RL}) \boxtimes (\varnothing \otimes \epsilon_{LR} \otimes \epsilon_{RR}) \ \mathsf{presplit}$$

  as required.

- $\Phi \vdash (\varnothing \otimes \epsilon_L) \boxtimes (\varnothing \otimes \epsilon_R)$ with $\epsilon_L \equiv \top$ or $\epsilon_R \equiv \top$: Again each $t_?$ must have $\varnothing$ as the underlying pure presplit, so the the same rule applies.

- $\mathfrak{c} \prec \Phi \vdash (\mathfrak{c} \otimes \epsilon_L) \boxtimes (\varnothing \otimes \epsilon_R) \ \mathsf{presplit}$: On the left side we have

  $$(\mathfrak{c} \prec \Phi)^{\mathfrak{c} \otimes \epsilon_L} \vdash t_{LL} \boxtimes t_{LR} \ \mathsf{presplit}$$

  and so using Lemma B.2.12 if necessary, also

  $$\mathfrak{c} \prec \Phi \vdash t_{LL} \boxtimes t_{LR} \ \mathsf{presplit}$$

  The preslices t_{RL} and t_{RR} must have underlying presplit $\varnothing$, and so we use Lemma B.2.11 to see $(\mathfrak{c} \prec \Phi)^{\mathfrak{c} \otimes \epsilon_L} \vdash (t_{LL} \otimes t_{RL}) \boxtimes (t_{LR} \otimes t_{RR}) \ \mathsf{presplit}$ as required.

- $\mathfrak{c} \prec \Phi \vdash s_L \boxtimes s_R \ \mathsf{presplit}$ where $\Phi \vdash s_L \boxtimes s_R \ \mathsf{presplit}$: Because $(\mathfrak{c} \prec \Phi)^{s_L} \equiv \Phi^{s_L}$ and $(\mathfrak{c} \prec \Phi)^{s_R} \equiv \Phi^{s_R}$ in this case, induction gives $\Phi \vdash (t_{LL} \otimes t_{RL}) \boxtimes (t_{LR} \otimes t_{RR}) \ \mathsf{presplit}$ which can be weakened to $\mathfrak{c} \prec \Phi$.

- $\Phi_1, \Phi_2 \vdash s_{1L} \boxtimes s_{1R}$ presplit where $\Phi_1 \vdash s_{1L} \boxtimes s_{1R}$ presplit: Similar to the previous, $(\Phi_1, \Phi_2)^{s_{1L}} \equiv {\Phi_1}^{s_{1L}}$ and $(\Phi_1, \Phi_2)^{s_{1R}} \equiv {\Phi_1}^{s_{1R}}$, so induction gives $\Phi_1 \vdash (t_{LL} \otimes t_{RL}) \boxtimes (t_{LR} \otimes t_{RR})$ presplit which can be weakened to Φ_1, Φ_2.

- $\Phi_1 \otimes \Phi_2 \vdash (s_{1L} \otimes s_{2L}) \boxtimes (s_{1R} \otimes s_{2R})$ presplit where $\Phi_1 \vdash s_{1L} \boxtimes s_{1R}$ presplit and $\Phi_2 \vdash s_{2L} \boxtimes s_{2R}$ presplit: Then the other inputs are

$$(\Phi_1 \otimes \Phi_2)^{s_{1L} \otimes s_{2L}} \vdash t_{LL} \boxtimes t_{LR} \text{ presplit}$$
$$(\Phi_1 \otimes \Phi_2)^{s_{1R} \otimes s_{2R}} \vdash t_{RL} \boxtimes t_{RR} \text{ presplit}$$

  which by Lemma B.2.7 decompose into

$${\Phi_1}^{s_{1L}} \vdash t_{LL}|_{s_{1L}} \boxtimes t_{LR}|_{s_{1L}} \text{ presplit}$$
$${\Phi_2}^{s_{2L}} \vdash t_{LL}|_{s_{2L}} \boxtimes t_{LR}|_{s_{2L}} \text{ presplit}$$
$${\Phi_1}^{s_{1R}} \vdash t_{RL}|_{s_{1R}} \boxtimes t_{RR}|_{s_{1R}} \text{ presplit}$$
$${\Phi_2}^{s_{2R}} \vdash t_{RL}|_{s_{2R}} \boxtimes t_{RR}|_{s_{2R}} \text{ presplit}$$

  Now by induction

$$\Phi_1 \vdash (t_{LL}|_{s_{1L}} \otimes t_{RL}|_{s_{1R}}) \boxtimes (t_{LR}|_{s_{1L}} \otimes t_{RR}|_{s_{1R}}) \text{ presplit}$$
$$\Phi_2 \vdash (t_{LL}|_{s_{2L}} \otimes t_{RL}|_{s_{2R}}) \boxtimes (t_{LR}|_{s_{2L}} \otimes t_{RR}|_{s_{2R}}) \text{ presplit}$$

  And so reapplying the tensor constructor for splits:

$$\begin{aligned}\Phi_1 \otimes \Phi_2 \vdash &((t_{LL}|_{s_{1L}} \otimes t_{RL}|_{s_{1R}}) \otimes (t_{LL}|_{s_{2L}} \otimes t_{RL}|_{s_{2R}})) \\ &\boxtimes ((t_{LR}|_{s_{2L}} \otimes t_{RR}|_{s_{2R}}) \otimes (t_{LR}|_{s_{1L}} \otimes t_{RR}|_{s_{1R}})) \text{ presplit}\end{aligned}$$

  These are the correct slices, by rearranging the tensors and reconstructing the input preslices.

□

Lemma B.2.14. *Slices can be conatenated:*

$$\frac{\Phi \vdash s_L \boxtimes s_R \text{ presplit} \qquad \Phi^{s_L} \vdash t_L \text{ preslice} \qquad \Phi^{s_R} \vdash t_R \text{ preslice}}{\Phi \vdash t_L \otimes t_R \text{ preslice}}$$

Proof. By induction on the presplit. The ϵ_L and ϵ_R flags have no effect on the available pure preslices, so are omitted when describing the cases below.

- $\mathfrak{c} \prec \Phi \vdash \varnothing \boxtimes \varnothing$ presplit: Then the only possibility is $t_L \equiv t_R \equiv \varnothing$ and $\varnothing$ is also preslice in $\mathfrak{c} \prec \Phi$.

- $\mathfrak{c} \prec \Phi \vdash \mathfrak{c} \boxtimes \varnothing$ presplit: Only two possibilities. If $t_L \equiv t_R \equiv \varnothing$ then $\mathfrak{c} \prec \Phi \vdash \varnothing$ preslice in the conclusion. And if $t_L \equiv \mathfrak{c}$ and $t_R \equiv \varnothing$, then $\mathfrak{c} \prec \Phi \vdash \mathfrak{c}$ preslice in the conclusion.

- $\mathfrak{c} \prec \Phi \vdash s_L \boxtimes s_R$ presplit where $\Phi \vdash s_L \boxtimes s_R$ presplit: Then by induction $\Phi \vdash t_L \otimes t_R$ preslice, and so also $\mathfrak{c} \prec \Phi \vdash t_L \otimes t_R$ preslice.

- $\Phi_1, \Phi_2 \vdash s_{1L} \boxtimes s_{1R}$ presplit where $\Phi_1 \vdash s_{1L} \boxtimes s_{1R}$ presplit: Then inductively $\Phi_1 \vdash t_L \otimes t_R$ preslice, and so also $\Phi_1, \Phi_2 \vdash t_L \otimes t_R$ preslice.
- $\Phi_1 \otimes \Phi_2 \vdash (s_{1L} \otimes s_{2L}) \boxtimes (s_{1R} \otimes s_{2R})$ presplit where $\Phi_1 \vdash s_{1L} \boxtimes s_{1R}$ presplit and $\Phi_2 \vdash s_{2L} \boxtimes s_{2R}$ presplit: Then the input slices are

  $$(\Phi_1 \otimes \Phi_2)^{s_{1L} \otimes s_{2L}} \vdash t_L \text{ preslice}$$
  $$(\Phi_1 \otimes \Phi_2)^{s_{1R} \otimes s_{2R}} \vdash t_R \text{ preslice}$$

  whose palettes are equal to ${\Phi_1}^{s_{1L}} \otimes {\Phi_2}^{s_{2L}}$ and ${\Phi_1}^{s_{1R}} \otimes {\Phi_2}^{s_{2R}}$ by definition. Now use Lemma B.2.6 to find preslices

  $$\begin{gathered} {\Phi_1}^{s_{1L}} \vdash t_L|_{{\Phi_1}^{s_{1L}}} \text{ preslice} \\ {\Phi_2}^{s_{2L}} \vdash t_L|_{{\Phi_2}^{s_{2L}}} \text{ preslice} \\ {\Phi_1}^{s_{1R}} \vdash t_R|_{{\Phi_1}^{s_{1R}}} \text{ preslice} \\ {\Phi_2}^{s_{2R}} \vdash t_R|_{{\Phi_2}^{s_{2R}}} \text{ preslice} \end{gathered}$$

  Inductively

  $$\begin{gathered} \Phi_1 \vdash t_L|_{{\Phi_1}^{s_{1L}}} \otimes t_R|_{{\Phi_1}^{s_{1R}}} \text{ preslice} \\ \Phi_2 \vdash t_L|_{{\Phi_2}^{s_{2L}}} \otimes t_R|_{{\Phi_2}^{s_{2R}}} \text{ preslice} \end{gathered}$$

  and so

  $$\Phi_1 \otimes \Phi_2 \vdash (t_L|_{s_{1L}} \otimes t_R|_{s_{1R}}) \otimes (t_L|_{s_{2L}} \otimes t_R|_{s_{2R}}) \text{ preslice}$$

  Finally, by symmetry and the equation of Lemma B.2.6, this preslice is equal to $t_L \otimes t_R$.

The remaining rules follow by symmetry from one of the above. □

Lemma B.2.15. *Slices under another slice can be concatenated:*

$$\frac{\begin{array}{c} \Phi \vdash s_L \boxtimes s_R \text{ presplit} \\ \Phi^{s_L} \vdash v_L \text{ preslice}_\epsilon \qquad \Phi^{s_R} \vdash v_R \text{ preslice}_\epsilon \\ (\Phi^{s_L})^{v_L} \vdash t_L \text{ preslice} \qquad (\Phi^{s_R})^{v_R} \vdash t_R \text{ preslice} \end{array}}{\Phi^{v_L \otimes v_R} \vdash t_L \otimes t_R \text{ preslice}}$$

This is the 'under a slice' version of the ordinary tensor constructor for slices.

Proof. By induction on the split and s_L and s_R, using Lemma B.2.6 on v_L and v_R in the case that Φ is a tensor palette. □

Lemma B.2.16.

$$\frac{\Phi \vdash s \text{ preslice}}{\Phi^s \vdash s \text{ preslice}} \qquad\qquad \frac{\Phi \vdash s \text{ preslice}_\epsilon}{\Phi^{s \otimes \epsilon} \vdash s \text{ preslice}_\epsilon}$$

Proof. For pure preslices, induction on s:

- $\varnothing$ preslice: The preslice $\varnothing$ is well-formed in any context.
- $\mathfrak{c}$ preslice: Then $\Phi^{\mathfrak{c}}$ has the form $\mathfrak{c} \prec \Psi$ for some Ψ, and $\mathfrak{c} \prec \Psi \vdash \mathfrak{c}$ preslice.
- $\mathfrak{c} \prec \Phi \vdash s$ preslice where $\Phi \vdash s$ preslice: By induction, because in this case $(\mathfrak{c} \prec \Phi)^s \equiv \Phi^s$.
- $\Phi_1, \Phi_2 \vdash s$ preslice where $\Phi_1 \vdash s$ preslice: By induction, because in this case $(\Phi_1, \Phi_2)^s \equiv {\Phi_1}^s$.
- $\Phi_L \otimes \Phi_R \vdash s_L \otimes s_R$ preslice where $\Phi_L \vdash s_L$ preslice and $\Phi_R \vdash s_R$ preslice: By induction

$$\begin{gathered} {\Phi_L}^{s_L} \vdash s_L \text{ preslice} \\ {\Phi_R}^{s_R} \vdash s_R \text{ preslice} \end{gathered}$$

  There are cases depending on whether s_L and s_R are equal to $\varnothing$.

  – If neither are, then

$$ {\Phi_L}^{s_L} \otimes {\Phi_R}^{s_R} \vdash s_L \otimes s_R \text{ preslice}$$

  and $(\Phi_L \otimes \Phi_R)^{s_L \otimes s_R} \equiv {\Phi_L}^{s_L} \otimes {\Phi_R}^{s_R}$ by definition.

  – If $s_L \equiv \varnothing$ then $(\Phi_L \otimes \Phi_R)^{s_L \otimes s_R} \equiv {\Phi_R}^{s_R}$ and and $s_L \otimes s_R \equiv s_R$, so ${\Phi_R}^{s_R} \vdash s_R$ preslice is the required preslice.

  – If $s_L \equiv s_R \equiv \varnothing$ then the result is $\varnothing$ which is well-formed in any context.

The remaining cases follow by symmetry.

For preslices, by definition $\Phi^{s \otimes 1} \equiv (\Phi^s \otimes 1)$. The previous shows that $\Phi^s \vdash s$ preslice, and so also $\Phi^s \otimes 1 \vdash s$ preslice, and then $\Phi^s \otimes 1 \vdash s \otimes 1$ preslice. □

Lemma B.2.17.

$$\frac{\Phi \vdash s \text{ preslice}}{\Phi^s \vdash s \boxtimes \varnothing \text{ presplit} \qquad \Phi^s \vdash \varnothing \boxtimes s \text{ presplit}} \qquad \frac{\Phi \vdash s \text{ preslice}_\epsilon}{\Phi^s \vdash s \boxtimes \varnothing \text{ presplit} \qquad \Phi^s \vdash \varnothing \boxtimes s \text{ presplit}}$$

Proof. Induction on s, as in Lemma B.2.16:

- $\varnothing$ preslice: Then $\Phi^{\varnothing} \equiv \varnothing$, and $\varnothing \vdash \varnothing \boxtimes \varnothing$ presplit.
- $\mathfrak{c}$ preslice: Then $\Phi^{\mathfrak{c}}$ has the form $\mathfrak{c} \prec \Psi$ for some Ψ, and so $\mathfrak{c} \boxtimes \varnothing$ presplit.
- $\mathfrak{c} \prec \Phi \vdash s$ preslice where $\Phi \vdash s$ preslice: By induction, because in this case $(\mathfrak{c} \prec \Phi)^s \equiv \Phi^s$.
- $\Phi_1, \Phi_2 \vdash s$ preslice where $\Phi_1 \vdash s$ preslice: By induction, because in this case $(\Phi_1, \Phi_2)^s \equiv {\Phi_1}^s$.
- $\Phi_L \otimes \Phi_R \vdash s_L \otimes s_R$ preslice where $\Phi_L \vdash s_L$ preslice and $\Phi_R \vdash s_R$ preslice: By induction

$$\begin{gathered} {\Phi_L}^{s_L} \vdash s_L \boxtimes \varnothing \text{ presplit} \\ {\Phi_R}^{s_R} \vdash s_R \boxtimes \varnothing \text{ presplit} \end{gathered}$$

  There are cases depending on whether s_L and s_R are equal to $\varnothing$.

– If neither are, then

$$\Phi_L{}^{s_L} \otimes \Phi_R{}^{s_R} \vdash (s_L \otimes s_R) \boxtimes (\varnothing \otimes \varnothing)\ \mathsf{presplit}$$

and $(\Phi_L \otimes \Phi_R)^{s_L \otimes s_R} \equiv \Phi_L{}^{s_L} \otimes \Phi_R{}^{s_R}$ and $\varnothing \otimes \varnothing \equiv \varnothing$ by definition.

– If $s_L \equiv \varnothing$ then $(\Phi_L \otimes \Phi_R)^{s_L \otimes s_R} \equiv \Phi_R{}^{s_R}$ and and $s_L \otimes s_R \equiv s_R$, so $\Phi_R{}^{s_R} \vdash s_R \boxtimes \varnothing\ \mathsf{presplit}$ is the required presplit.

– If $s_L \equiv s_R \equiv \varnothing$ then we have $\varnothing \vdash \varnothing \boxtimes \varnothing\ \mathsf{presplit}$.

The remaining cases follow by symmetry.

For s a preslice, by definition $\Phi^{s \otimes 1} \equiv (\Phi^s \otimes 1)$. The previous shows that $\Phi^s \vdash s \boxtimes \varnothing\ \mathsf{presplit}$, and combining this with the split $1 \vdash 1 \boxtimes \varnothing\ \mathsf{presplit}$ gives

$$\Phi^s \otimes 1 \vdash (s \otimes 1) \boxtimes \varnothing\ \mathsf{presplit}$$

as required. □

B.2.1 Marking

Theorem B.2.18. *Marking on slices is admissible.*

$$\textsc{mark}\ \frac{(\mathfrak{s} \prec \Xi)\{\Phi\} \vdash s\ \mathsf{preslice}}{(\mathfrak{s} \prec \Xi)\{\downarrow 1\} \vdash s^{\mathsf{m}\Phi}\ \mathsf{preslice}_\epsilon} \qquad \textsc{mark}\ \frac{(\mathfrak{s} \prec \Xi)\{\Phi\} \vdash s\ \mathsf{preslice}_\epsilon}{(\mathfrak{s} \prec \Xi)\{\downarrow 1\} \vdash s^{\mathsf{m}\Phi}\ \mathsf{preslice}_\epsilon}$$

$$\textsc{mark}\ \frac{(\mathfrak{s} \prec \Xi)\{\Phi\} \vdash s\ \mathsf{slice}}{(\mathfrak{s} \prec \Xi)\{\downarrow 1\} \vdash s^{\mathsf{m}\Phi}\ \mathsf{slice}}$$

Proof. Induction on the spot.

- $\Phi\{\Phi\}$ spot: Then $s^{\mathsf{m}\Phi} \equiv 1$ because all colours in s are marked, and we have $1 \vdash 1\ \mathsf{preslice}_\epsilon$.

- $\mathfrak{c} \prec \Xi\{\Phi\}$ spot: Two subcases:

  – $\Xi\{\Phi\} \vdash s\ \mathsf{preslice}$: Then inductively $\Xi\{\downarrow 1\} \vdash s^{\mathsf{m}\Phi}\ \mathsf{preslice}_\epsilon$ which we can weaken with $\mathfrak{c}$.

  – $s \equiv \mathfrak{c}$: Then $s^{\mathsf{m}\Phi} \equiv \mathfrak{c}$ and indeed $\mathfrak{c} \prec \Xi\{\downarrow 1\} \vdash \mathfrak{c}\ \mathsf{preslice}_\epsilon$.

- $\Xi\{\Phi\}, \Xi'$ spot: Two subcases:

  – $\Xi\{\Phi\} \vdash s\ \mathsf{preslice}$: Then inductively $\Xi\{\downarrow 1\} \vdash s^{\mathsf{m}\Phi}\ \mathsf{preslice}_\epsilon$ which we can weaken with Ξ'.

  – $\Xi' \vdash s\ \mathsf{preslice}$: Then s does not contain any colours from Φ and so $s \equiv s^{\mathsf{m}\Phi}$, and we can weaken with $\Xi\{\downarrow 1\}$.

- $\Xi\{\Phi\} \otimes \Xi'$: Then preslice is of the form $s \otimes s'$. Inductively, $\Xi\{\downarrow 1\} \vdash s^{\mathsf{m}\Phi}\ \mathsf{preslice}_\epsilon$, and so $\Xi\{\downarrow 1\} \otimes \Xi' \vdash s^{\mathsf{m}\Phi} \otimes s'\ \mathsf{preslice}_\epsilon$, and $s^{\mathsf{m}\Phi} \otimes s' \equiv (s \otimes s')^{\mathsf{m}\Phi}$ because colours in Φ do not occur in s'.

□

Lemma B.2.19. *If* $\Xi\{\Phi\}$ spot *and* $\Xi \vdash s$ slice *with* Φ *contained in* s*, then*

$$(\mathfrak{s} \prec \Xi)^{s}\{\downarrow 1\} \equiv (\mathfrak{s} \prec \Xi)\{\downarrow 1\}^{s^{\mathsf{m}\Phi}}$$

Proof. Note that we have implicitly used Lemma B.2.1 to know that $(\mathfrak{s} \prec \Xi)^{s}\{\Phi\}$ spot.

Induction on the spot and the slice. The crucial case is when we reach the slice $\mathfrak{c}$ so that $\Xi^{\mathfrak{c}}$ contains Φ. Then $\mathfrak{c}^{\mathsf{m}\Phi} \equiv \mathfrak{c}$, and indeed

$$(\mathfrak{s} \prec \Xi)^{\mathfrak{c}}\{\downarrow 1\} \equiv (\mathfrak{s} \prec \Xi)\{\downarrow 1\}^{\mathfrak{c}}$$

□

B.2.2 Mark-Weakening

Theorem B.2.20. *Mark-weakening is admissible on slices and splits.*

$$\textsc{MarkWk}\ \frac{\begin{array}{c}(\mathfrak{s} \prec \Xi)\{1\}\ \mathsf{spot} \\ (\mathfrak{s} \prec \Xi)\{1\} \vdash s\ \mathsf{slice}\end{array}}{(\mathfrak{s} \prec \Xi)\{\downarrow\Phi\} \vdash s\ \mathsf{slice}} \qquad \textsc{MarkWk}\ \frac{\begin{array}{c}(\mathfrak{s} \prec \Xi)\{1\}\ \mathsf{spot} \\ (\mathfrak{s} \prec \Xi)\{1\} \vdash s_L \boxtimes s_R\ \mathsf{presplit}\end{array}}{(\mathfrak{s} \prec \Xi)\{\downarrow\Phi\} \vdash s_L \boxtimes s_R\ \mathsf{presplit}}$$

B.2.3 Cartesian Substitution

Theorem B.2.21. *Substitution on slices is admissible.*

$$\textsc{Subst}\ \frac{\Phi \vdash \kappa : \Psi \qquad \Psi \vdash t\ \mathsf{preslice}}{\Phi \vdash t[\kappa]\ \mathsf{preslice}_{\epsilon}} \qquad \textsc{Subst}\ \frac{\Phi \vdash \kappa : \Psi \qquad \Psi \vdash \mathfrak{c}\ \mathsf{colour}}{\Phi \vdash \mathfrak{c}[\kappa]\ \mathsf{slice}}$$

$$\textsc{Subst}\ \frac{\Phi \vdash \kappa : \Psi \qquad \Psi \vdash t\ \mathsf{slice}}{\Phi \vdash t[\kappa]\ \mathsf{slice}}$$

Note that substitution into a pure preslice can result in a preslice, because each individual colour may be replaced by a preslice containing 1.

Proof. When t is a pure preslice we proceed by induction on the structure of Ψ. It is enough to show that the underlying pure preslice of $t[\kappa]$ is well-formed, because the 1 bit has no effect on typing.

- $\varnothing$ and $\varnothing_i$: The only pure preslices of these palettes are $\varnothing$, and $\varnothing[\kappa] \equiv \varnothing$ is well-formed in any context.
- Ψ_1, Ψ_2: Then the substitution is of the form κ_1, κ_2. Without loss of generality, $\Psi_1 \vdash t\ \mathsf{preslice}_{\epsilon}$ and by induction $\Phi \vdash t[\kappa_1]\ \mathsf{preslice}_{\epsilon}$. This is equal to $t[\kappa_1, \kappa_2]$ because variables in κ_2 do not occur in t.

- $\Psi_L \otimes \Psi_R$: Then the substitution is of the form $\kappa_L \otimes \kappa_R$ for

$$\begin{aligned}\Phi &\vdash s_L \boxtimes s_R \ \mathsf{presplit}\\ \Phi^{s_L} &\vdash \kappa_L : \Psi_L\\ \Phi^{s_R} &\vdash \kappa_R : \Psi_R\end{aligned}$$

  and the slice is of the form $t_L \otimes t_R$ for

$$\begin{aligned}\Psi_L &\vdash t_L \ \mathsf{preslice}_\epsilon\\ \Psi_R &\vdash t_R \ \mathsf{preslice}_\epsilon\end{aligned}$$

  Inductively,

$$\begin{aligned}\Phi^{s_L} &\vdash t_L[\kappa_L] \ \mathsf{preslice}_\epsilon\\ \Phi^{s_R} &\vdash t_R[\kappa_R] \ \mathsf{preslice}_\epsilon\end{aligned}$$

  and by Lemma B.2.14, $\Phi \vdash t_L[\kappa_L] \otimes t_R[\kappa_R]$ $\mathsf{preslice}_\epsilon$. This is equal to $(t_L \otimes t_R)[\kappa_L \otimes \kappa_R]$ because variables in κ_R do not occur in t_L and vice versa.

- $\mathfrak{c} \prec \Psi$: Then the substitution is of the form $(s/\mathfrak{c} \prec \kappa)$ for $\Phi \vdash s$ slice. There are two subcases:
  - $t \equiv \mathfrak{c}$: Then $\mathfrak{c}[s/\mathfrak{c} \prec \kappa] \equiv \mathsf{u}(s)$, which we know is well-formed.
  - $\Psi \vdash t \ \mathsf{preslice}_\epsilon$: Then by induction, $\Phi^{\mathsf{u}(s)} \vdash t[\kappa] \ \mathsf{preslice}_\epsilon$. Applying Lemma B.2.14 together with $\varnothing \vdash \varnothing$ preslice gives $\Phi \vdash t[\kappa] \ \mathsf{preslice}_\epsilon$, and $t[\kappa] \equiv t[s/\mathfrak{c} \prec \kappa]$ because $t \not\equiv c$.

□

Theorem B.2.22. *Substitution on slices under another slice is admissible.*

$$\textsc{subst/cod}\ \frac{\Psi \vdash v \ \mathsf{preslice} \quad \Phi \vdash \kappa : \Psi \quad \Psi^v \vdash t \ \mathsf{preslice}}{\Phi^{v[\kappa]} \vdash t[\kappa] \ \mathsf{preslice}_\epsilon} \qquad \textsc{subst/cod}\ \frac{\Psi \vdash v \ \mathsf{preslice} \quad \Phi \vdash \kappa : \Psi \quad \Psi^v \vdash t \ \mathsf{slice}}{\Phi^{v[\kappa]} \vdash t[\kappa] \ \mathsf{slice}}$$

Proof. Similar to the previous, induction on Ψ and v.

- $\varnothing$ and $\varnothing_i$: The only possible v is $v \equiv \varnothing$, and then $\Psi^\varnothing \equiv \varnothing$ and again the only possible t is $\varnothing$.
- Ψ_1, Ψ_2: Then the substitution is of the form κ_1, κ_2. Without loss of generality, $\Psi_1 \vdash v$ preslice, so ${\Psi_1}^v \vdash t$ preslice. Inductively $\Phi^{v[\kappa_1]} \vdash t[\kappa_1] \ \mathsf{preslice}_\epsilon$, and $t[\kappa_1] \equiv t[\kappa_1, \kappa_2]$.
- $\Psi_L \otimes \Psi_R$: Then the substitution is of the form $\kappa_L \otimes \kappa_R$ for

$$\begin{aligned}\Phi &\vdash s_L \boxtimes s_R \ \mathsf{presplit}\\ \Phi^{s_L} &\vdash \kappa_L : \Psi_L\\ \Phi^{s_R} &\vdash \kappa_R : \Psi_R\end{aligned}$$

and v is of the form $v_L \otimes v_R$ with

$$\Psi_L \vdash v_L \ \mathsf{preslice}$$
$$\Psi_R \vdash v_R \ \mathsf{preslice}$$

There are subcases, depending on whether v_L and v_R are $\varnothing$.

- With v_L and v_R both non-trivial, then t is of the form $t_L \otimes t_R$ with

  $$\Psi_L{}^{v_L} \vdash t_L \ \mathsf{preslice}$$
  $$\Psi_R{}^{v_R} \vdash t_R \ \mathsf{preslice}$$

  Inductively

  $$(\Phi^{s_L})^{v_L[\kappa_L]} \vdash t_L[\kappa_L] \ \mathsf{preslice}_\epsilon$$
  $$(\Phi^{s_R})^{v_R[\kappa_R]} \vdash t_R[\kappa_R] \ \mathsf{preslice}_\epsilon$$

  and by Lemma B.2.15,

  $$\Phi^{v_L[\kappa_L] \otimes v_R[\kappa_R]} \vdash t_L[\kappa_L] \otimes t_R[\kappa_R] \ \mathsf{preslice}_\epsilon$$

  Finally, $v_L[\kappa_L] \otimes v_R[\kappa_R] \equiv (v_L \otimes v_R)[\kappa] \equiv v[\kappa]$ and $t_L[\kappa_L] \otimes t_R[\kappa_R] \equiv (t_L \otimes t_R)[\kappa] \equiv t[\kappa]$.
- If only v_L is non-trivial, then

  $$\Psi_L{}^{v_L} \vdash t \ \mathsf{preslice}$$

  and inductively

  $$(\Phi^{s_L})^{v_L[\kappa_L]} \vdash t[\kappa_L] \ \mathsf{preslice}_\epsilon$$

  For the palette, $(\Phi^{s_L})^{v_L[\kappa_L]} \equiv \Phi^{v_L[\kappa_L]}$ by Lemma B.2.9. And $v_L[\kappa_L] \equiv v_L[\kappa_L] \otimes \varnothing \equiv v_L[\kappa_L] \otimes v_R[\kappa_R] \equiv v[\kappa]$, and $t[\kappa_L] \equiv t[\kappa]$ because t contains no colours in s_R.
- If only v_R is non-trivial, apply the symmetric argument.
- If both $v_L \equiv v_R \equiv \varnothing$ then also $t \equiv \varnothing$.

- $\mathfrak{c} \prec \Psi$: Then the substitution is of the form $(s/\mathfrak{c} \prec \kappa)$ for $\Phi \vdash s \ \mathsf{slice}$ and $\Phi^{\mathsf{u}(s)} \vdash \kappa : \Psi$. There are two subcases:

  - $v \equiv \mathfrak{c}$: Then $\mathfrak{c} \prec \Psi \vdash t \ \mathsf{preslice}$, and so we can apply ordinary substitution to get $\Phi^{\mathsf{u}(s)} \vdash t[\kappa] \ \mathsf{preslice}_\epsilon$, and $\mathfrak{c}[s/\mathfrak{c} \prec \kappa] \equiv \mathsf{u}(s)$ by definition.
  - $\Psi \vdash v \ \mathsf{preslice}$: Then $\Psi^v \vdash t \ \mathsf{preslice}$, and by induction, $(\Phi^{\mathsf{u}(s)})^{v[\kappa]} \vdash t[\kappa] \ \mathsf{preslice}_\epsilon$. Applying Lemma B.2.15 together with $\varnothing \vdash \varnothing \ \mathsf{preslice}$ gives $\Phi^{v[\kappa]} \vdash t[\kappa] \ \mathsf{preslice}_\epsilon$. Finally $v[\kappa] \equiv v[s/\mathfrak{c} \prec \kappa]$ and $t[\kappa] \equiv t[s/\mathfrak{c} \prec \kappa]$ because $v \not\equiv \mathfrak{c}$ and $t \not\equiv c$.

□

Lemma B.2.23. *Substitution is admissible on unit labels.*

$$\frac{\Phi \vdash \kappa : \Psi \qquad \Psi \vdash i \ \mathsf{unit}}{\Psi \vdash i[\kappa] \ \mathsf{unit}}$$

Proof. By induction on Ψ. □

Theorem B.2.24. *Substitution on splits is admissible.*

$$\textsc{subst}\ \frac{\Phi \vdash \kappa : \Psi \qquad \Psi \vdash s_L \boxtimes s_R \ \mathsf{presplit}}{\Phi \vdash s_L[\kappa] \boxtimes s_R[\kappa] \ \mathsf{presplit}} \qquad \textsc{subst}\ \frac{\Phi \vdash \kappa : \Psi \qquad \Psi \vdash s_L \boxtimes s_R \ \mathsf{split}}{\Phi \vdash s_L[\kappa] \boxtimes s_R[\kappa] \ \mathsf{split}}$$

Proof. For presplits, induction on the derivation of the presplit:

- $\varnothing \vdash (\varnothing \otimes \epsilon_L) \boxtimes (\varnothing \otimes \epsilon_R)$ presplit: There are no substitutions into $\varnothing$.
- $\Psi \vdash (\varnothing \otimes \epsilon_L) \boxtimes (\varnothing \otimes \epsilon_R)$ presplit with $\epsilon_L \equiv \top$ or $\epsilon_R \equiv \top$: Then also $\Phi \vdash (\varnothing \otimes \epsilon_L) \boxtimes (\varnothing \otimes \epsilon_R)$ presplit.
- $\mathfrak{c} \prec \Psi \vdash (\mathfrak{c} \otimes \epsilon_L) \boxtimes (\varnothing \otimes \epsilon_R)$ presplit: Then the substitution has the form $(s/\mathfrak{c} \prec \kappa)$ for $\Phi \vdash s$ slice, and such that $\Phi \vdash \mathsf{u}(s) \boxtimes \varnothing$ presplit.

  By definition, $(\mathfrak{c} \otimes \epsilon_L)[s/\mathfrak{c} \prec \kappa] \equiv (\mathsf{u}(s) \otimes \epsilon_L)$. Lemma B.2.11 twice then gives $\Phi \vdash (\mathsf{u}(s) \otimes \epsilon_L) \boxtimes (\varnothing \otimes \epsilon_R)$ presplit as required.
- $\mathfrak{c} \prec \Psi \vdash s_L \boxtimes s_R$ presplit where $\Psi \vdash s_L \boxtimes s_R$ presplit: Then the substitution has the form $(s/\mathfrak{c} \prec \kappa)$ for $\Phi^{\mathsf{u}(s)} \vdash \kappa : \Psi$, and inductively $\Phi^{\mathsf{u}(s)} \vdash s_L[\kappa] \boxtimes s_R[\kappa]$ presplit. Finally $\Phi \vdash s_L[\kappa] \boxtimes s_R[\kappa]$ presplit by applying interchange (Lemma B.2.13) on this split and $\varnothing \vdash \varnothing \boxtimes \varnothing$ presplit.
- $\Psi_1, \Psi_2 \vdash s_{1L} \boxtimes s_{1R}$ presplit where $\Psi_1 \vdash s_{1L} \boxtimes s_{1R}$ presplit: The substitution is of the form κ_1, κ_2 with $\Phi \vdash \kappa_1 : \Psi_1$. Inductively $\Phi \vdash s_{1L}[\kappa_1] \boxtimes s_{1R}[\kappa_1]$ presplit, and $s_{1L}[\kappa_1] \equiv s_{1L}[\kappa_1, \kappa_2]$ and $s_{1R}[\kappa_1] \equiv s_{1R}[\kappa_1, \kappa_2]$ because Ψ_1 does not occur in either of the slices.
- $\Psi_1 \otimes \Psi_2 \vdash (s_{1L} \otimes s_{2L}) \boxtimes (s_{1R} \otimes s_{2R})$ presplit where $\Psi_1 \vdash s_{1L} \boxtimes s_{1R}$ presplit and $\Psi_2 \vdash s_{2L} \boxtimes s_{2R}$ presplit: Then the substitution is of the form $\kappa_L \otimes \kappa_R$ for

  $$\Phi \vdash s_L \boxtimes s_R \ \mathsf{presplit}$$
  $$\Phi^{s_L} \vdash \kappa_L : \Psi_L$$
  $$\Phi^{s_R} \vdash \kappa_R : \Psi_R$$

  Inductively

  $$\Phi^{s_L} \vdash s_{1L}[\kappa_L] \boxtimes s_{1R}[\kappa_L] \ \mathsf{presplit}$$
  $$\Phi^{s_R} \vdash s_{2L}[\kappa_R] \boxtimes s_{2R}[\kappa_R] \ \mathsf{presplit}$$

  Applying Lemma B.2.13 gives

  $$\Phi \vdash (s_{1L}[\kappa_L] \otimes s_{2L}[\kappa_R]) \boxtimes (s_{1R}[\kappa_L] \otimes s_{2R}[\kappa_R]) \ \mathsf{presplit}$$

Now s_{1L} does not contain variables from κ_R, and similarly for the others, so these slices are equal to

$$\Phi \vdash (s_{1L} \otimes s_{2L})[\kappa_L \otimes \kappa_R] \boxtimes (s_{1R} \otimes s_{2R})[\kappa_L \otimes \kappa_R] \text{ presplit}$$

as required.

$\square$

Theorem B.2.25. *Substitution on slices at a spot is admissible.*

$$\textsc{subst/spot}\ \frac{\begin{array}{c}\Xi\{\mathfrak{t} \prec \Phi, \Psi, \dots\} \text{ spot} \\ \mathfrak{t} \prec \Phi \vdash \kappa : \Psi \qquad \Xi\{\mathfrak{t} \prec \Phi, \Psi, \dots\} \vdash s \text{ slice}\end{array}}{\Xi\{\downarrow \mathfrak{t} \prec \Phi, \dots\} \vdash s[\kappa] \text{ slice}}$$

$$\textsc{subst/cod}\ \frac{\begin{array}{cc}\Psi \vdash v \text{ preslice} & \Xi\{\Psi^v\} \text{ spot} \\ \mathfrak{t} \prec \Phi \vdash \kappa : \Psi & \Xi\{\Psi^v\} \vdash s \text{ slice}\end{array}}{\Xi\{\downarrow(\mathfrak{t} \prec \Phi)^{v[\kappa]}\} \vdash s[\kappa] \text{ slice}} \qquad \textsc{subst/dom}\ \frac{\begin{array}{cc}\Phi \vdash w \text{ preslice} & \Xi\{\Phi^w\} \text{ spot} \\ \mathfrak{t} \prec \Phi \vdash \kappa : \Psi & \Xi\{\Phi^w\} \vdash s \text{ slice}\end{array}}{\Xi\{\Phi^w\} \vdash s[\kappa] \text{ slice}}$$

$$\textsc{subst/marked}\ \frac{\mathfrak{t} \prec \Phi \vdash \kappa : \Psi \qquad \Xi \vdash s \text{ slice}}{\Xi \vdash s[\kappa] \text{ slice}}$$

Proof. The idea is: we do induction until we reach the spot and then apply the previous substitution rule. First SUBST/SPOT, induction on the spot:

- $(\mathfrak{t} \prec \Phi, \Psi)\{\mathfrak{t} \prec \Phi, \Psi\}$ spot: Two subcases:
  - $\Psi \vdash s$ slice: Then the actual substitution rule (Theorem B.2.21) gives $\mathfrak{t} \prec \Phi \vdash s[\kappa]$ slice.
  - $\mathfrak{t} \prec \Phi \vdash s$ slice: Then $s \equiv s[\kappa]$ is already the desired slice.
- $(\mathfrak{t} \prec \Xi, \Xi')\{\mathfrak{t} \prec \Phi, \dots\}$ spot where $\Xi\{\Phi, \dots\}, \Xi'$ spot: Two subcases:
  - $\mathfrak{t} \prec \Xi \vdash s$ slice: Then inductively $(\mathfrak{t} \prec \Xi)\{\downarrow \mathfrak{t} \prec \Phi\} \vdash s[\kappa]$ slice, which can be weakened to $(\mathfrak{t} \prec \Xi)\{\downarrow \mathfrak{t} \prec \Phi\}, \Xi' \vdash s[\kappa]$ slice
  - $\Xi' \vdash s$ slice: Then $s \equiv s[\kappa]$ is already the desired slice.
- $\mathfrak{c} \prec \Xi\{\mathfrak{t} \prec \Phi, \dots\}$ spot: Then inductively $\Xi\{\downarrow \mathfrak{t} \prec \Phi, \dots\} \vdash s[\kappa]$ slice, which can be weakened to $\mathfrak{c} \prec \Xi\{\downarrow \mathfrak{t} \prec \Phi, \dots\} \vdash s[\kappa]$ slice.
- $\Xi\{\mathfrak{t} \prec \Phi, \dots\}, \Xi'$ spot: Then inductively $\Xi\{\downarrow \mathfrak{t} \prec \Phi, \dots\} \vdash s[\kappa]$ slice, which can be weakened to $\Xi\{\downarrow \mathfrak{t} \prec \Phi, \dots\}, \Xi \vdash s[\kappa]$ slice.
- $\Xi\{\mathfrak{t} \prec \Phi, \dots\} \otimes \Xi'$ spot: Then the slice has an underlying preslice of the form $s_L \otimes s_R$ for

$$\Xi\{\mathfrak{t} \prec \Phi, \Psi, \dots\} \vdash s_L \text{ presplit}$$
$$\Xi' \vdash s_R \text{ presplit}$$

  Inductively $\Xi\{\downarrow \mathfrak{t} \prec \Phi, \dots\} \vdash s_L[\kappa]$ presplit, and $(s_L \otimes s_R)[\kappa] \equiv s_L[\kappa] \otimes s_R$ because s_R does not contain colours in Ψ.

The rule SUBST/COD is similar, in the base case we have $\Psi^v \vdash s$ slice and so apply Theorem B.2.22 instead of Theorem B.2.21.

The final two rules hold because in both cases s contains no colours in Ψ. ☐

Theorem B.2.26. *Substitution on splits is admissible, with rules analogous to Theorem B.2.25.*

Proof. By induction with the same structure as Theorem B.2.25. ☐

Lemma B.2.27. *Suppose $\Phi \vdash \kappa : \Psi$ and $\Xi\{\Psi\}$ spot. If $\Xi \vdash s$ slice intersects Ψ, then $(s \cap \Psi)[\kappa] \equiv (s[\kappa] \cap \Phi)$.*

Proof. Induction on the spot and the slice s. ☐

B.2.4 Slice Intersection

Definition B.2.28. The intersection of two pure preslices

$$\frac{\Phi \vdash s \ \mathsf{preslice} \qquad \Phi \vdash t \ \mathsf{preslice}}{\Phi \vdash s \cap t \ \mathsf{preslice}}$$

defined by induction on Φ:

- 1 palette, $\varnothing$ palette and $\varnothing_i$ palette: The only pure preslice is $\varnothing$ preslice and define $\varnothing \cap \varnothing :\equiv \varnothing$.
- Φ_1, Φ_2 palette: If $\Phi_1 \vdash s$ preslice and $\Phi_1 \vdash t$ preslice, then induct and weaken. If $\Phi_1 \vdash s$ preslice and $\Phi_2 \vdash t$ preslice, then define $s \cap t :\equiv \varnothing$. The other cases are similar by symmetry.
- $\Phi_1 \otimes \Phi_2$ palette: Define $(s_L \otimes s_R) \cap (t_L \otimes t_R) :\equiv (s_L \cap t_L) \otimes (s_R \cap t_R)$.
- $\mathfrak{c} \prec \Phi$ palette: Define

$$\begin{aligned} s \cap \varnothing &:\equiv \varnothing \\ \varnothing \cap s &:\equiv \varnothing \\ s \cap \mathfrak{c} &:\equiv s \\ \mathfrak{c} \cap s &:\equiv s \\ \mathfrak{c} \cap \mathfrak{c} &:\equiv \mathfrak{c} \end{aligned}$$

This definition requires the preslices to be well-formed, as the raw syntax of preslices does not give enough information to know what is in the palette under a particular colour label. For example, in palette $\mathfrak{p} \prec \mathfrak{r} \otimes \mathfrak{b}$ palette, the raw syntax does not tell us that $\mathfrak{p} \cap \mathfrak{r} \equiv \mathfrak{r}$ rather than $\mathfrak{p}$.

Lemma B.2.29. *If $\Phi \vdash s$ preslice and $\Phi^s \vdash t$ preslice then $s \cap t \equiv t$, where t has been weakened to Φ using Lemma B.2.8.*

Proof. By induction on s and the definition of slice intersection. ☐

Lemma B.2.30. *For* $\Phi \vdash s$ preslice *and* $\Phi \vdash t$ preslice,

$$\mathfrak{c} \in s \cap t \textit{ iff } \mathfrak{c} \in s \textit{ and } \mathfrak{c} \in t$$

for any $\Phi \vdash \mathfrak{c}$ colour.

Proof. Induction on the palette Φ. ☐

Lemma B.2.31. *For*

$$\begin{aligned} \mathfrak{t} \prec \Phi &\vdash \kappa : \Psi \\ \Psi &\vdash s \text{ preslice} \\ \Psi &\vdash t \text{ preslice} \end{aligned}$$

we have

$$\mathfrak{c} \in (s \cap t)[\kappa] \textit{ iff } \mathfrak{c} \in s[\kappa] \textit{ and } \mathfrak{c} \in t[\kappa]$$

for any $\mathfrak{t} \prec \Phi \vdash \mathfrak{c}$ colour.

Proof. By induction on Ψ. ☐

It is not true that $(s \cap t)[\kappa] \equiv (s[\kappa] \cap t[\kappa])$ because it is possible for $s \cap t$ to lie in different cartesian bunches that are contracted together by the substitution.

Lemma B.2.32. *If* $\Xi\{\Phi^v\}$ spot *and s intersects* Φ^v*, then* $\Xi^s\{\Phi^{v \cap s}\}$ spot.

Proof. By induction on the spot and the pure preslice s. ☐

Lemma B.2.33. *If* $\Phi \vdash \kappa : \Psi$ *and* $\Xi \vdash s$ slice *and* $\Xi\{\Psi^v\}$ spot*, then* $\Xi^s\{\downarrow\Phi^{(v\cap s)[\kappa]}\} \equiv (\Xi\{\downarrow\Phi^{v[\kappa]}\})^{s[\kappa]}$

Proof. Induction on the spot and the slice s. Once we reach $\Psi^v \vdash s$ slice, we know $\Phi^{v[\kappa]} \vdash s[\kappa]$ preslice by SUBST/COD, and then $(\Phi^{v[\kappa]})^{s[\kappa]} \equiv \Phi^{s[\kappa]} \equiv \Phi^{(v\cap s)[\kappa]}$ because $v \cap s \equiv s$. ☐

Lemma B.2.34. *If* $\Phi^{s_L} \otimes \Phi^{s_R} \vdash v$ preslice *and* $\Phi^{s_L} \otimes \Phi^{s_R} \vdash w$ preslice*, then*

$$\mathfrak{c} \in (v \cap w)[\![\mathfrak{t} \prec s_L \boxtimes s_R / \mathfrak{t}']\!] \textit{ iff } \mathfrak{c} \in v[\![\mathfrak{t} \prec s_L \boxtimes s_R / \mathfrak{t}']\!] \textit{ and } \mathfrak{c} \in w[\![\mathfrak{t} \prec s_L \boxtimes s_R / \mathfrak{t}']\!]$$

for any $\mathfrak{t} \prec \Phi \vdash \mathfrak{c}$ colour.

Proof. If $\mathfrak{c}$ is the top colour of s_L then v and w must be of the form $\mathfrak{c} \otimes v'$ and $\mathfrak{c} \otimes w'$, and then certainly $\mathfrak{c} \in v \cap w$. If $\mathfrak{c}$ is contained in Φ^{s_L} then we use Lemma B.2.30. The other cases are similar. ☐

Lemma B.2.35. *If* $\Phi \vdash s_L \boxtimes s_R$ split *and* $\Xi \vdash t$ slice *and* $\Xi\{(\Phi^{s_L} \otimes \Phi^{s_R})^v\}$ spot*, then*

$$\Xi^t\{\downarrow\Phi^{(v\cap t)[\![\mathfrak{t}\prec s_L \boxtimes s_R/\mathfrak{t}']\!]}\} \equiv (\Xi\{\downarrow\Phi^{v[\![\mathfrak{t}\prec s_L \boxtimes s_R/\mathfrak{t}']\!]}\})^{t[\![\mathfrak{t}\prec s_L \boxtimes s_R/\mathfrak{t}']\!]}$$

Proof. Induction on the spot and the slice t. Once we reach $(\Phi^{s_L} \otimes \Phi^{s_R})^v \vdash t$ slice, we consider cases on the pure preslice v. For example, if $v \equiv \ulcorner s_L \urcorner$, then we have two subcases. When $t \equiv \ulcorner s_L \urcorner$, we have $v \cap t \equiv \ulcorner s_L \urcorner$ also, and the result follows by Lemma B.2.16. Otherwise, we have $v \cap t \equiv t$ and the result follows by Lemma B.2.8. The other cases are similar. ☐

B.2.5 Other

Lemma B.2.36. *If* $\Xi\{\mathfrak{t} \prec \Phi, \dots\}$ spot *and* $\mathfrak{t} \prec \Phi \vdash \mathfrak{c}$ colour, *then* $\Xi^{\mathfrak{c}}$ *is a cartesian weakening of* $\Phi^{\mathfrak{c}}$.

Proof. By induction on the spot, until we reach the cartesian bunch containing $\mathfrak{t} \prec \Phi, \dots$. Then either $\mathfrak{c}$ is equal to $\mathfrak{t}$, in which case $(\mathfrak{t} \prec \Phi, \dots)^{\mathfrak{c}} \equiv \mathfrak{t} \prec \Phi, \dots$ is certainly a cartesian weakening of $\mathfrak{t} \prec \Phi$, or $\mathfrak{c} \in \Phi$, in which case $(\mathfrak{t} \prec \Phi, \dots)^{\mathfrak{c}} \equiv \Phi^{\mathfrak{c}}$ exactly. □

Lemma B.2.37.

$$\Phi^{s_L} \otimes \Phi^{s_R} \vdash w \ \mathsf{preslice}_\epsilon$$
$$\mathfrak{t} \prec \Phi \vdash s_L \boxtimes s_R \ \mathsf{presplit}$$

and $\mathfrak{c} \in w$ *then* $\mathfrak{c} \in w[\![(\mathfrak{t} \prec s_L \boxtimes s_R / \mathfrak{t}')]\!]$.

Proof. By induction on the presplit until we reach a colour label containing the colour $\mathfrak{c}$. □

Lemma B.2.38. *If* $\mathfrak{s} \prec \Xi\{\mathfrak{t}' \prec \Psi\}$ spot *then the top colour of* $\mathfrak{s} \prec \Xi\{\downarrow\mathfrak{t} \prec \Phi\}$ spot *is* $\mathfrak{s}^{\mathfrak{t} \leftrightarrow \mathfrak{t}'}$.

Proof. If the spot is immediately at the top of the palette, then $\mathfrak{s} \equiv \mathfrak{t}'$, and then $\mathfrak{s}^{\mathfrak{t} \leftrightarrow \mathfrak{s}} \equiv \mathfrak{t}$, which is the top colour of $\mathfrak{t} \prec \Phi$. Otherwise, the top colour of the resulting palette is unchanged, and indeed $\mathfrak{s}^{\mathfrak{t} \leftrightarrow \mathfrak{t}'} \equiv \mathfrak{s}$ □

B.3 Contexts

B.3.1 Palette Weakening

Lemma B.3.1. *Palette weakening of contexts is admissible.*

$$\textsc{wk-pal-ctx}\ \frac{\Phi \mid \Gamma \ \mathsf{ctx}}{\Xi\{\Phi\} \mid \Gamma \ \mathsf{ctx}}$$

Proof. Considering one variable at a time, for ordinary labelled variables $x^{\mathfrak{c}} : A$ we have $(\Xi\{\Phi\})^{\mathfrak{c}} \equiv \Phi^{\mathfrak{c}}$, and so the type A remains well-formed. For marked variables $\underline{x}^{\mathfrak{c}} : A$, the type $\mathfrak{c} \mid \underline{\Gamma} \vdash A$ type used in the premise is exactly what is required to extend the context with $\underline{x}^{\mathfrak{c}} : A$ in the conclusion. □

Lemma B.3.2. *Cartesian palette weakening for all judgements is admissible.*

$$\textsc{wk-pal-cart}\ \frac{\Xi\{\Phi\} \mid \Gamma \vdash \mathcal{J}}{\Xi\{\Phi, \Psi\} \mid \Gamma \vdash \mathcal{J}}$$

Proof. For slices, splits and terms, induction on the derivation. For contexts and context extensions, we may consider one variable at a time, applying the rule to the type. □

B.3.2 Context Weakening

Lemma B.3.3. *Weakening a term by a single variable is admissible.*

$$\textsc{wk}\;\frac{\Phi \mid \Gamma, \Gamma' \vdash \mathcal{J}}{\Phi \mid \Gamma, x^{\mathfrak{c}} : A, \Gamma' \vdash \mathcal{J}} \qquad \textsc{wk-marked}\;\frac{\Phi \mid \Gamma, \Gamma' \vdash \mathcal{J}}{\Phi \mid \Gamma, \underline{x}^{\mathfrak{c}} : A, \Gamma' \vdash \mathcal{J}}$$

Proof. Induction on derivations. For WK, encountering a splitting rule may require a use of WK-MARKED, if the variable to be added is not contained in the slice used in the premise of the splitting rule. □

Lemma B.3.4. *Weakening a term by a context extension is admissible.*

$$\textsc{wk-ext}\;\frac{\Phi \mid \Gamma, \Gamma'' \vdash \mathcal{J}}{\Phi \mid \Gamma, \Gamma', \Gamma'' \vdash \mathcal{J}}$$

Proof. Repeated application of WK and WK-MARKED. □

B.3.3 Recolouring

$$\textsc{recolour}\;\frac{\mathfrak{t}' \prec \Phi \mid \Gamma \vdash \mathcal{J}}{\mathfrak{t} \prec \Phi \mid \Gamma^{\mathfrak{t} \leftrightarrow \mathfrak{t}'} \vdash \mathcal{J}^{\mathfrak{t} \leftrightarrow \mathfrak{t}'}}$$

Or more generally

$$\textsc{recolour}\;\frac{\Xi\{\mathfrak{t}' \prec \Phi\} \mid \Gamma \vdash \mathcal{J}}{\Xi\{\mathfrak{t} \prec \Phi\} \mid \Gamma^{\mathfrak{t} \leftrightarrow \mathfrak{t}'} \vdash \mathcal{J}^{\mathfrak{t} \leftrightarrow \mathfrak{t}'}}$$

Theorem B.3.5. *Recolouring is admissible on terms.*

Proof. Straightforward induction on derivations. In the cases for VAR-ROUNDTRIP and VAR-MARKED, we use Lemma B.1.8 to show that the result has the correct type, and in the splitting rules we us Lemma B.1.10 when the chosen slice does not contain the colour being replaced. □

Lemma B.3.6. *The colour label on a marked variable can be changed, and this is silent in the judgement.*

$$\frac{\Phi \mid \Gamma, \underline{x}^{\mathfrak{c}'} : A, \Gamma' \vdash \mathcal{J}}{\Phi \mid \Gamma, \underline{x}^{\mathfrak{c}} : A^{\mathfrak{c} \leftrightarrow \mathfrak{c}'}, \Gamma' \vdash \mathcal{J}}$$

Proof. By induction on $\mathcal{J}$. The important case is VAR-ZERO. If

$$\frac{\mathfrak{c}' \mid \underline{\Gamma} \vdash A \ \mathsf{type}}{\mathfrak{t} \prec \Phi \mid \Gamma, \underline{x}^{\mathfrak{c}'} : A, \Gamma' \vdash \underline{x} : A^{\mathfrak{t} \leftrightarrow \mathfrak{c}'}}$$

then also

$$\frac{\mathfrak{c} \mid \underline{\Gamma} \vdash A^{\mathfrak{c} \leftrightarrow \mathfrak{c}'} \ \mathsf{type}}{\mathfrak{t} \prec \Phi \mid \Gamma, \underline{x}^{\mathfrak{c}} : A^{\mathfrak{c} \leftrightarrow \mathfrak{c}'}, \Gamma' \vdash \underline{x} : (A^{\mathfrak{c} \leftrightarrow \mathfrak{c}'})^{\mathfrak{t} \leftrightarrow \mathfrak{c}}}$$

and $(A^{\mathfrak{c} \leftrightarrow \mathfrak{c}'})^{\mathfrak{t} \leftrightarrow \mathfrak{c}} \equiv A^{\mathfrak{t} \leftrightarrow \mathfrak{c}'}$ by Lemma B.1.8. □

B.3.4 Filtering

Theorem B.3.7. *Filtering of contexts, telescopes and context extensions is admissible.*

$$\textsc{Filter}\ \frac{\Phi \vdash s \ \mathsf{slice} \qquad \Phi \mid \Gamma \ \mathsf{ctx}}{\Phi^s \mid \Gamma^s \ \mathsf{ctx}} \qquad \textsc{Filter}\ \frac{\Psi \vdash s \ \mathsf{preslice} \qquad \mathfrak{t} \prec \Phi \mid \Gamma \vdash \Psi \mid \Omega \ \mathsf{tele}}{\mathfrak{t} \mid \underline{\Gamma} \vdash \Psi^s \mid \Omega^s \ \mathsf{tele}}$$

$$\textsc{Filter}\ \frac{\Xi\{\mathfrak{t} \prec \Phi, \dots\} \vdash s \ \mathsf{slice} \qquad (\mathfrak{s} \prec \Xi)\{\Phi\} \mid \Gamma; \Gamma' \ \mathsf{ext}}{((\mathfrak{s} \prec \Xi)\{\Phi\})^s \mid \Gamma^s; \Gamma'^s \ \mathsf{ext}}$$

Proof. Each of these may be proved one variable at a time. For contexts,

- $x^\mathfrak{c} : A$ is unmarked: Then the context was formed by

  $$\frac{\Phi^\mathfrak{c} \mid \Gamma^\mathfrak{c} \vdash A \ \mathsf{type}}{\Phi \mid \Gamma, x^\mathfrak{c} : A \ \mathsf{ctx}}$$

  There are two subcases:

  - $\mathfrak{c} \in s$: Then $\Phi^\mathfrak{c} \equiv (\Phi^s)^\mathfrak{c}$ by Lemma B.2.8 and $\Gamma^\mathfrak{c} \equiv (\Gamma^s)^\mathfrak{c}$ by Lemma B.3.9, so we have $(\Phi^s)^\mathfrak{c} \mid (\Gamma^s)^\mathfrak{c} \vdash A$ type and can re-form the context extension.
  - $\mathfrak{c} \notin s$: Then marking A gives $\mathfrak{c} \mid \underline{\Gamma} \vdash \underline{A}$ type, and $\underline{\Gamma} \equiv \underline{\Gamma^s}$ by idempotence, so we can re-form the context extension $\Phi^s \mid \Gamma^s, \underline{x}^\mathfrak{c} : \underline{A}$ ctx.

- $\underline{x}^\mathfrak{c} : A$ is marked: Then the context was formed by

  $$\frac{\mathfrak{c} \mid \underline{\Gamma} \vdash A \ \mathsf{type}}{\Phi \mid \Gamma, \underline{x}^\mathfrak{c} : A \ \mathsf{ctx}}$$

  but $\underline{\Gamma} \equiv \underline{\Gamma^s}$ by idempotence, so $\mathfrak{c} \mid \underline{\Gamma^s} \vdash A$ type and we can re-form the context $\Phi^s \mid \Gamma^s, \underline{x}^\mathfrak{c} : A$ ctx.

Telescopes and context extensions are similar. ☐

Lemma B.3.8. $(\Gamma^s)^t \equiv \Gamma^{(s \cap t)}$

Proof. By Lemma B.2.30 operating one variable at a time. ☐

Corollary B.3.9. $(\Gamma^s)^t \equiv \Gamma^t$ *when* $\Phi^s \vdash t$ preslice

Proof. Combining Lemma B.3.8 with Lemma B.2.29. ☐

Lemma B.3.10. $(\Gamma^{s[\kappa]})^{t[\kappa]} \equiv \Gamma^{(s \cap t)[\kappa]}$

Proof. By Lemma B.2.31 operating one variable at a time. ☐

Lemma B.3.11. $(\Gamma^{u\llbracket \mathfrak{t} \prec s_L \boxtimes s_R / t' \rrbracket})^{v\llbracket \mathfrak{t} \prec s_L \boxtimes s_R / t' \rrbracket} \equiv \Gamma^{(u \cap v)\llbracket \mathfrak{t} \prec s_L \boxtimes s_R / t' \rrbracket}$

Proof. By Lemma B.2.34 operating one variable at a time. □

Lemma B.3.12. *If* $\Phi \vdash \Gamma$ ctx *is weakened to* $\Xi\{\Phi\} \vdash \Gamma$ ctx *and* $\Xi \vdash s$ slice, *then* $\Gamma^s \equiv \Gamma^{s\cap\Phi}$.

Proof. All colour labels in Γ are in Φ, so for all such labels $\mathfrak{c} \in s$ iff $\mathfrak{c} \in s \cap \Phi$. □

Lemma B.3.13. $\Gamma^s \equiv \underline{\Gamma}$ *if* s *is fresh for* Γ.

Proof. Checking one variable at a time, every unmarked $x^{\mathfrak{c}} : A$ becomes marked by Γ^s, because $\mathfrak{c} \notin s$. □

Lemma B.3.14. *If* $(\mathfrak{s} \prec \Xi)\{\Phi\} \mid \Gamma; \Gamma'$ ext *and* $\mathfrak{c}$ *does not contain* Φ *then* $\Gamma'^{\mathfrak{c}} \equiv (\Gamma'^{\mathsf{m}\Phi|\Gamma})^{\mathfrak{c}}$.

Proof. Marked variables in Γ' remain marked on both sides. For each unmarked variable $x^{\mathfrak{d}} : A$ in Γ', either

- $\mathfrak{d} \in \mathfrak{c}$, in which case x remains unmarked on both sides. We know $(\mathfrak{s} \prec \Xi)^{\mathfrak{d}}$ also does not contain Φ and so $A \equiv A^{\mathsf{m}\Phi|\Gamma}$ by Lemma B.3.17.
- $\mathfrak{d} \notin \mathfrak{c}$, in which case x is marked on both sides.

□

B.3.5 Marking

$$\textsc{mark}\ \frac{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \mathcal{J}}{\mathfrak{t} \mid \underline{\Gamma} \vdash \underline{\mathcal{J}}}$$

The generalised form:

$$\textsc{mark}\ \frac{\mathfrak{t} \prec \Phi \mid \Gamma \text{ ctx} \qquad (\mathfrak{s} \prec \Xi)\{\Phi\} \mid \Gamma; \Gamma' \text{ ext} \qquad (\mathfrak{s} \prec \Xi)\{\Phi\} \mid \Gamma, \Gamma' \vdash \mathcal{J}}{(\mathfrak{s} \prec \Xi)\{\downarrow 1\} \mid \underline{\Gamma}, \Gamma'^{\mathsf{m}\Phi|\Gamma} \vdash \mathcal{J}^{\mathsf{m}\Phi|\Gamma}}$$

Lemma B.3.15. *We can mark 'under a slice':*

$$\textsc{mark/dispatch}\ \frac{(\mathfrak{s} \prec \Xi)\{\Phi\} \vdash s \text{ slice} \qquad (\mathfrak{s} \prec \Xi)\{\Phi\}^s \mid (\Gamma, \Gamma')^s \vdash \mathcal{J}}{(\mathfrak{s} \prec \Xi)\{\downarrow 1\}^{s\mathsf{m}\Phi} \mid (\underline{\Gamma}, \Gamma'^{\mathsf{m}\Phi|\Gamma})^{s\mathsf{m}\Phi} \vdash \mathcal{J}^{\mathsf{m}\Phi|\Gamma}}$$

Proof. We distinguish two cases:

- s intersects Φ: Then $(\mathfrak{s} \prec \Xi)^s\{\Phi^{s\cap\Phi}\}$ spot and $(\Gamma, \Gamma')^s \equiv \Gamma^{s\cap\Phi}, \Gamma'^s$, and applying MARK gives

  $$(\mathfrak{s} \prec \Xi)^s\{\downarrow 1\} \mid \underline{\Gamma^{s\cap\Phi}}, (\Gamma'^s)^{\mathsf{m}\Phi|\Gamma} \vdash \mathcal{J}^{\mathsf{m}\Phi|\Gamma}$$

  For the palette, we know $(\mathfrak{s} \prec \Xi)^s\{\downarrow 1\} \equiv (\mathfrak{s} \prec \Xi)\{\downarrow 1\}^{s\mathsf{m}\Phi}$ by Lemma B.2.19, and for the context $\underline{\Gamma^{s\cap\Phi}}, (\Gamma'^s)^{\mathsf{m}\Phi|\Gamma} \equiv (\underline{\Gamma}, \Gamma'^{\mathsf{m}\Phi|\Gamma})^{s\mathsf{m}\Phi}$ by idempotence on Γ and Lemma B.1.21 on Γ'. So

  $$(\mathfrak{s} \prec \Xi)\{\downarrow 1\}^{s\mathsf{m}\Phi} \mid (\underline{\Gamma}, \Gamma'^{\mathsf{m}\Phi|\Gamma})^{s\mathsf{m}\Phi} \vdash \mathcal{J}^{\mathsf{m}\Phi|\Gamma}$$

  as required.

- s does not intersect Φ: Then $s \equiv s^{\mathsf{m}\Phi}$ so $(\mathfrak{s} \prec \Xi)\{\Phi\}^{s} \equiv (\mathfrak{s} \prec \Xi)\{\downarrow 1\}^{s^{\mathsf{m}\Phi}}$. The input $\mathcal{J}$ is already marked with respect to Φ, so $\mathcal{J} \equiv \mathcal{J}^{\mathsf{m}\Phi|\Gamma}$ by Lemma B.3.17. Similarly, $\Gamma' \equiv \Gamma'^{\mathsf{m}\Phi|\Gamma}$ and so already

$$(\mathfrak{s} \prec \Xi)\{\downarrow 1\}^{s^{\mathsf{m}\Phi}} \mid (\underline{\Gamma}, \Gamma'^{\mathsf{m}\Phi|\Gamma})^{s^{\mathsf{m}\Phi}} \vdash \mathcal{J}^{\mathsf{m}\Phi|\Gamma}$$

$\square$

Theorem B.3.16. *Marking is admissible on terms.*

$$\textsc{mark}\ \frac{\mathfrak{t} \prec \Phi \mid \Gamma \vdash a : A}{\mathfrak{t} \mid \underline{\Gamma} \vdash \underline{a} : \underline{A}}$$

Proof.

- VAR: Two cases:
  - $x^{\mathfrak{s}} : A \in \Gamma$: Because $x^{\mathfrak{s}} \in \Gamma$, it must be the case that $\mathfrak{s} \in (\mathfrak{t} \prec \Phi)$. The derivation is then

$$\frac{(\mathfrak{s} \prec \Xi)\{\Phi\} \mid \Gamma_1 \vdash A \ \mathsf{type}}{(\mathfrak{s} \prec \Xi)\{\Phi\} \mid \Gamma_1, x^{\mathfrak{s}} : A, \Gamma_2, \Gamma' \vdash x : A}$$

    By induction,

$$(\mathfrak{s} \prec \Xi)\{\downarrow 1\} \mid \underline{\Gamma_1} \vdash A^{\mathsf{m}\Phi|\Gamma_1} \ \mathsf{type}$$

    and so also

$$(\mathfrak{s} \prec \Xi)\{\downarrow 1\} \mid \underline{\underline{\Gamma_1}} \vdash A^{\mathsf{m}\Phi|\Gamma_1} \ \mathsf{type}$$

    by Lemma B.1.1.
    By definition, $\underline{\Gamma_1, x^{\mathfrak{s}} : A, \Gamma_2} \equiv \underline{\Gamma_1}, \underline{x}^{\mathfrak{s}} : A^{\mathsf{m}\Phi|\Gamma_1}, \underline{\Gamma_2}$. Applying VAR-MARKED gives

$$(\mathfrak{s} \prec \Xi)\{\downarrow 1\} \mid \underline{\Gamma_1, x^{\mathfrak{s}} : A, \Gamma_2}, \Gamma'^{\mathsf{m}\Phi|\Gamma_1, x^{\mathfrak{s}}:A, \Gamma_2} \vdash \underline{x} : A^{\mathsf{m}\Phi|\Gamma_1}$$

    Finally, $A^{\mathsf{m}\Phi|\Gamma_1} \equiv A^{\mathsf{m}\Phi|\Gamma_1, x^{\mathfrak{s}}:A, \Gamma_2}$ because x and Γ_2 do not occur in A.
  - $x^{\mathfrak{s}} : A \in \Gamma'$: Then the derivation is

$$\frac{(\mathfrak{s} \prec \Xi)\{\Phi\} \mid \Gamma, \Gamma'_1 \vdash A \ \mathsf{type}}{(\mathfrak{s} \prec \Xi)\{\Phi\} \mid \Gamma, \Gamma'_1, x^{\mathfrak{s}} : A, \Gamma'_2 \vdash x : A}$$

    By induction also

$$(\mathfrak{s} \prec \Xi)\{\downarrow 1\} \mid \underline{\Gamma}, {\Gamma'_1}^{\mathsf{m}\Phi|\Gamma} \vdash A^{\mathsf{m}\Phi|\Gamma} \ \mathsf{type}$$

    and because $(\Gamma'_1, x^{\mathfrak{s}} : A, \Gamma'_2)^{\mathsf{m}\Phi|\Gamma} \equiv {\Gamma'_1}^{\mathsf{m}\Phi|\Gamma}, x^{\mathfrak{s}} : A^{\mathsf{m}\Phi|\Gamma}, {\Gamma'_2}^{\mathsf{m}\Phi|\Gamma}$, we can reapply VAR.

- VAR-ROUNDTRIP: Again, two cases:
  - $x^{\mathfrak{c}} : A \in \Gamma$: Then the derivation is

$$\frac{((\mathfrak{s} \prec \Xi)\{\Phi\})^{\mathfrak{c}} \mid {\Gamma_1}^{\mathfrak{c}} \vdash A \ \mathsf{type}}{(\mathfrak{s} \prec \Xi)\{\Phi\} \mid \Gamma_1, x^{\mathfrak{c}} : A, \Gamma_2, \Gamma' \vdash \underline{x} : (A^{\mathsf{m}\Phi|\Gamma_1})^{\mathfrak{s}\leftrightarrow\mathfrak{c}}}$$

    Then in the conclusion, $\underline{x}^{\mathfrak{c}} : A^{\mathsf{m}\Phi|\Gamma_1} \in \underline{\Gamma_1, x^{\mathfrak{c}} : A, \Gamma_2}$ by the definition of marking on contexts. Applying VAR-MARKED gives

$$(\mathfrak{s} \prec \Xi)\{\downarrow 1\} \mid \underline{\Gamma_1, x^{\mathfrak{c}} : A, \Gamma_2}, \Gamma'^{\mathsf{m}\Phi|\Gamma_1, x^{\mathfrak{c}}:A, \Gamma_2} \vdash \underline{x} : (A^{\mathsf{m}\Phi|\Gamma_1})^{\mathfrak{s}\leftrightarrow\mathfrak{c}}$$

    because $\mathfrak{c} \mid \underline{\underline{\Gamma_1}} \vdash A^{\mathsf{m}\Phi|\Gamma_1}$ type by Lemma B.1.1. This is of the required type, because

$$(A^{\mathsf{m}\Phi|\Gamma_1})^{\mathfrak{s}\leftrightarrow\mathfrak{c}} \equiv ((A^{\mathsf{m}\Phi|\Gamma_1})^{\mathfrak{s}\leftrightarrow\mathfrak{c}})^{\mathsf{m}\Phi|\Gamma_1, x, \Gamma_2}$$

    by Lemma B.1.1 and Lemma B.1.9.
  - $x^{\mathfrak{c}} : A \in \Gamma'$: By assumption, $\mathfrak{c} \notin \Phi$, so $x^{\mathfrak{c}} : A^{\mathsf{m}\Phi|\Gamma}$ in the context $\underline{\Gamma}, (\Gamma'_1, x^{\mathfrak{c}} : A, \Gamma'_2)^{\mathsf{m}\Phi|\Gamma}$ is unmarked in the conclusion. There are now two subcases:
    * $\Phi \subseteq \mathfrak{c}$: Then $(\mathfrak{s} \prec \Xi)^{\mathfrak{c}}\{\Phi\}$ spot and the derivation is

$$\frac{(\mathfrak{s} \prec \Xi)^{\mathfrak{c}}\{\Phi\} \mid \Gamma, \Gamma'^{\mathfrak{c}} \vdash A \ \mathsf{type}}{(\mathfrak{s} \prec \Xi)\{\Phi\} \mid \Gamma, \Gamma'_1, x^{\mathfrak{c}} : A, \Gamma'_2 \vdash \underline{x} : A^{\mathfrak{s}\leftrightarrow\mathfrak{c}}}$$

      Inductively,

$$(\mathfrak{s} \prec \Xi)^{\mathfrak{c}}\{\downarrow 1\} \mid \underline{\Gamma}, (\Gamma'^{\mathfrak{c}}_1)^{\mathsf{m}\Phi|\Gamma} \vdash A^{\mathsf{m}\Phi|\Gamma} \ \mathsf{type}$$

      And by Lemma B.1.23, this context extension is equal to $(\Gamma'^{\mathsf{m}\Phi|\Gamma}_1)^{\mathfrak{c}}$ and by Lemma B.2.2, the palette is $((\mathfrak{s} \prec \Xi)\{\downarrow 1\})^{\mathfrak{c}}$, so

$$((\mathfrak{s} \prec \Xi)\{\downarrow 1\})^{\mathfrak{c}} \mid (\underline{\Gamma}, \Gamma'^{\mathsf{m}\Phi|\Gamma}_1)^{\mathfrak{c}} \vdash A^{\mathsf{m}\Phi|\Gamma} \ \mathsf{type}$$

      and we can reapply VAR-ROUNDTRIP to conclude

$$(\mathfrak{s} \prec \Xi)\{\downarrow 1\} \mid \underline{\Gamma}, (\Gamma'_1, x^{\mathfrak{c}} : A, \Gamma'_2)^{\mathsf{m}\Phi|\Gamma} \vdash \underline{x} : (A^{\mathsf{m}\Phi|\Gamma})^{\mathfrak{s}\leftrightarrow\mathfrak{c}}$$

      Finally, $(A^{\mathsf{m}\Phi|\Gamma})^{\mathfrak{s}\leftrightarrow\mathfrak{c}} \equiv (A^{\mathfrak{s}\leftrightarrow\mathfrak{c}})^{\mathsf{m}\Phi|\Gamma}$ because $\mathfrak{c}$ is not in Φ.
    * $\Phi \nsubseteq \mathfrak{c}$: Then $(\mathfrak{s} \prec \Xi)^{\mathfrak{c}}$ does not contain Φ at all, and $\Gamma^{\mathfrak{c}} \equiv \underline{\Gamma}$ by Lemma B.3.13, so the derivation is

$$\frac{(\mathfrak{s} \prec \Xi)^{\mathfrak{c}} \mid \underline{\Gamma}, \Gamma'^{\mathfrak{c}}_1 \vdash A \ \mathsf{type}}{(\mathfrak{s} \prec \Xi)\{\Phi\} \mid \Gamma, \Gamma'_1, x^{\mathfrak{c}} : A, \Gamma'_2 \vdash \underline{x} : A^{\mathfrak{s}\leftrightarrow\mathfrak{c}}}$$

      And so also

$$(\mathfrak{s} \prec \Xi\{\downarrow 1\})^{\mathfrak{c}} \mid \underline{\Gamma}, (\Gamma'^{\mathsf{m}\Phi|\Gamma}_1)^{\mathfrak{c}} \vdash A^{\mathsf{m}\Phi|\Gamma} \ \mathsf{type}$$

because $\Gamma_1'^{\mathfrak{c}} \equiv (\Gamma_1'^{\mathsf{m}\Phi|\Gamma})^{\mathfrak{c}}$ by Lemma B.3.14 and $A \equiv A^{\mathsf{m}\Phi|\Gamma}$ by Lemma B.3.17. Reapplying VAR-ROUNDTRIP gives

$$\mathfrak{s} \prec \Xi\{\downarrow 1\} \mid \underline{\Gamma}, (\Gamma_1', x^{\mathfrak{c}} : A, \Gamma_2')^{\mathsf{m}\Phi|\Gamma} \vdash \underline{x} : (A^{\mathsf{m}\Phi|\Gamma})^{\mathfrak{s}\leftrightarrow\mathfrak{c}}$$

Finally, again $(A^{\mathsf{m}\Phi|\Gamma})^{\mathfrak{s}\leftrightarrow\mathfrak{c}} \equiv (A^{\mathfrak{s}\leftrightarrow\mathfrak{c}})^{\mathsf{m}\Phi|\Gamma}$ by Lemma B.1.9 because $\mathfrak{c}$ is not in Φ.

- VAR-MARKED: We have $\underline{x}^{\mathfrak{c}} : A \in \Gamma, \Gamma'$ and so also $\underline{x}^{\mathfrak{c}} : A \in \underline{\Gamma}, \Gamma'^{\mathsf{m}\Phi|\Gamma}$ in the conclusion. Two cases:
  - $\underline{x}^{\mathfrak{c}} : A \in \Gamma$: Then the derivation is

$$\frac{\mathfrak{c} \mid \underline{\Gamma_1} \vdash A \ \mathsf{type}}{(\mathfrak{s} \prec \Xi)\{\Phi\} \mid \Gamma_1, \underline{x}^{\mathfrak{c}} : A, \Gamma_2, \Gamma' \vdash \underline{x} : A^{\mathfrak{s}\leftrightarrow\mathfrak{c}}}$$

    Also $\mathfrak{c} \mid \underline{\underline{\Gamma_1}} \vdash A$ type by Lemma B.1.1 so we can reapply the rule, giving

$$(\mathfrak{s} \prec \Xi)\{\downarrow 1\} \mid \underline{\Gamma_1}, \underline{x}^{\mathfrak{c}} : A, \underline{\Gamma_2}, \Gamma'^{\mathsf{m}\Phi|\Gamma_1, x^{\mathfrak{c}}:A, \Gamma_2} \vdash \underline{x} : A^{\mathfrak{s}\leftrightarrow\mathfrak{c}}$$

    Finally $A^{\mathfrak{s}\leftrightarrow\mathfrak{c}} \equiv (A^{\mathfrak{s}\leftrightarrow\mathfrak{c}})^{\mathsf{m}\Phi|\Gamma_1, x, \Gamma_2}$ by Lemma B.3.17 because $A^{\mathfrak{s}\leftrightarrow\mathfrak{c}}$ is well-typed in a completely marked context.
  - $\underline{x}^{\mathfrak{c}} : A \in \Gamma'$: Suppose $\Gamma' \equiv \Gamma_1', \underline{x}^{\mathfrak{c}} : A, \Gamma_2'$. Then the derivation is

$$\frac{\mathfrak{c} \mid \underline{\Gamma, \Gamma_1'} \vdash A \ \mathsf{type}}{(\mathfrak{s} \prec \Xi)\{\Phi\} \mid \Gamma, \Gamma_1', \underline{x}^{\mathfrak{c}} : A, \Gamma_2' \vdash \underline{x} : A^{\mathfrak{s}\leftrightarrow\mathfrak{c}}}$$

    and again $\underline{\Gamma, \Gamma_1'} \equiv \underline{\underline{\Gamma}, \Gamma_1'^{\mathsf{m}\Phi|\Gamma}}$ by Lemma B.1.1 so we can reapply the rule, giving

$$(\mathfrak{s} \prec \Xi)\{\downarrow 1\} \mid \underline{\Gamma}, (\Gamma_1', \underline{x}^{\mathfrak{c}} : A, \Gamma_2')^{\mathsf{m}\Phi|\Gamma} \vdash \underline{x} : A^{\mathfrak{s}\leftrightarrow\mathfrak{c}}$$

    Finally $A^{\mathfrak{s}\leftrightarrow\mathfrak{c}} \equiv (A^{\mathfrak{s}\leftrightarrow\mathfrak{c}})^{\mathsf{m}\Phi|\Gamma}$ by Lemma B.3.17 because $A^{\mathfrak{s}\leftrightarrow\mathfrak{c}}$ is well-typed in a completely marked context.

- Π-FORM: Given inputs

$$(\mathfrak{s} \prec \Xi)\{\Phi\} \mid \Gamma, \Gamma' \vdash A \ \mathsf{type}$$
$$(\mathfrak{s} \prec \Xi)\{\Phi\} \mid \Gamma, \Gamma', x^{\mathfrak{s}} : A \vdash B \ \mathsf{type}$$

  inductively

$$(\mathfrak{s} \prec \Xi)\{\downarrow 1\} \mid \underline{\Gamma}, \Gamma'^{\mathsf{m}\Phi|\Gamma} \vdash A^{\mathsf{m}\Phi|\Gamma} \ \mathsf{type}$$
$$(\mathfrak{s} \prec \Xi)\{\downarrow 1\} \mid \underline{\Gamma}, \Gamma'^{\mathsf{m}\Phi|\Gamma}, x^{\mathfrak{s}} : A^{\mathsf{m}\Phi|\Gamma} \vdash B^{\mathsf{m}\Phi|\Gamma} \ \mathsf{type}$$

  where for B we have added $x^{\mathfrak{s}} : A$ to the extension Γ' used in the inductive case. Reapplying the rule then gives

$$(\mathfrak{s} \prec \Xi)\{\downarrow 1\} \mid \underline{\Gamma}, \Gamma'^{\mathsf{m}\Phi|\Gamma} \vdash \prod_{(x:A^{\mathsf{m}\Phi|\Gamma})} B^{\mathsf{m}\Phi|\Gamma} \ \mathsf{type}$$

- $\natural$-FORM: The instance of the rule is

$$\natural\text{-FORM}\ \frac{\mathfrak{s} \mid \underline{\Gamma, \Gamma'} \vdash A \ \mathsf{type}}{(\mathfrak{s} \prec \Xi)\{\Phi\} \mid \Gamma, \Gamma' \vdash \natural A \ \mathsf{type}}$$

By idempotence, also

$$\mathfrak{s} \mid \underline{\Gamma, \Gamma'^{\mathsf{m}\Phi|\Gamma}} \vdash A \ \mathsf{type}$$

and $A \equiv A^{\mathsf{m}\Phi|\Gamma}$ because A is well-typed in a marked context. Reapplying the rule gives

$$(\mathfrak{s} \prec \Xi)\{\downarrow 1\} \mid \underline{\Gamma}, \Gamma'^{\mathsf{m}\Phi|\Gamma} \vdash \natural A^{\mathsf{m}\Phi|\Gamma}$$

as required.

- $\otimes$-INTRO: The inputs are

$$(\mathfrak{s} \prec \Xi)\{\Phi\} \vdash s_L \boxtimes s_R \ \mathsf{split}$$
$$(\mathfrak{s} \prec \Xi)\{\Phi\}^{s_L} \mid (\Gamma, \Gamma')^{s_L} \vdash a : A^{\ulcorner s_L \urcorner \leftrightarrow \mathsf{t}}$$
$$(\mathfrak{s} \prec \Xi)\{\Phi\}^{s_R} \mid (\Gamma, \Gamma')^{s_R} \vdash b : B[\underline{a}^{\mathsf{t} \leftrightarrow \ulcorner s_L \urcorner} / \underline{x}^{\mathsf{t}}]^{\ulcorner s_R \urcorner \leftrightarrow \mathsf{t}}$$

Marking the split and applying MARK/DISPATCH to the terms, gives

$$(\mathfrak{s} \prec \Xi)\{\downarrow 1\} \vdash {s_L}^{\mathsf{m}\Phi} \boxtimes {s_R}^{\mathsf{m}\Phi} \ \mathsf{split}$$
$$(\mathfrak{s} \prec \Xi)\{\downarrow 1\}^{{s_L}^{\mathsf{m}\Phi}} \mid (\underline{\Gamma}, \Gamma'^{\mathsf{m}\Phi|\Gamma})^{{s_L}^{\mathsf{m}\Phi}} \vdash a^{\mathsf{m}\Phi|\Gamma} : (A^{\ulcorner s_L \urcorner \leftrightarrow \mathsf{t}})^{\mathsf{m}\Phi|\Gamma}$$
$$(\mathfrak{s} \prec \Xi)\{\downarrow 1\}^{{s_R}^{\mathsf{m}\Phi}} \mid (\underline{\Gamma}, \Gamma'^{\mathsf{m}\Phi|\Gamma})^{{s_R}^{\mathsf{m}\Phi}} \vdash b^{\mathsf{m}\Phi|\Gamma} : (B[\underline{a}^{\mathsf{t} \leftrightarrow \ulcorner s_L \urcorner} / \underline{x}^{\mathsf{t}}]^{\ulcorner s_R \urcorner \leftrightarrow \mathsf{t}})^{\mathsf{m}\Phi|\Gamma}$$

To reapply the rule, we need that the type of $b^{\mathsf{m}\Phi|\Gamma}$ is is the right form to be paired with $a^{\mathsf{m}\Phi|\Gamma}$. Lemma B.1.9 gives

$$(A^{\ulcorner s_L \urcorner \leftrightarrow \mathsf{t}})^{\mathsf{m}\Phi|\Gamma} \equiv (A^{\mathsf{m}\Phi|\Gamma})^{\ulcorner s_L \urcorner \leftrightarrow \mathsf{t}}$$
$$(B[\underline{a}^{\mathsf{t} \leftrightarrow \ulcorner s_L \urcorner} / \underline{x}^{\mathsf{t}}]^{\ulcorner s_R \urcorner \leftrightarrow \mathsf{t}})^{\mathsf{m}\Phi|\Gamma} \equiv (B[\underline{a}^{\mathsf{t} \leftrightarrow \ulcorner s_L \urcorner} / \underline{x}^{\mathsf{t}}]^{\mathsf{m}\Phi|\Gamma})^{\ulcorner s_R \urcorner \leftrightarrow \mathsf{t}}$$

and then

$$(B[\underline{a}^{\mathsf{t} \leftrightarrow \ulcorner s_L \urcorner} / \underline{x}^{\mathsf{t}}]^{\mathsf{m}\Phi|\Gamma})^{\ulcorner s_R \urcorner \leftrightarrow \mathsf{t}} \equiv (B^{\mathsf{m}\Phi|\Gamma}[\underline{a^{\mathsf{m}\Phi|\Gamma}}^{\mathsf{t} \leftrightarrow \ulcorner s_L \urcorner} / \underline{x}^{\mathsf{t}}])^{\ulcorner s_R \urcorner \leftrightarrow \mathsf{t}}$$

by Lemma B.1.4 and Lemma B.1.1. And these types are finally of the form to reapply the rule:

$$(\mathfrak{s} \prec \Xi)\{\downarrow 1\} \mid \underline{\Gamma}, \Gamma'^{\mathsf{m}\Phi|\Gamma} \vdash a^{\mathsf{m}\Phi|\Gamma} \ {}_{{s_L}^{\mathsf{m}\Phi}}\!\otimes_{{s_R}^{\mathsf{m}\Phi}} b^{\mathsf{m}\Phi|\Gamma} : \textcircled{Σ}_{(\underline{x} : A^{\mathsf{m}\Phi|\Gamma})} B^{\mathsf{m}\Phi|\Gamma}$$

with $\textcircled{$\Sigma$}_{(\underline{x} : A^{\mathsf{m}\Phi|\Gamma})} B^{\mathsf{m}\Phi|\Gamma} \equiv \left(\textcircled{Σ}_{(\underline{x} : A)} B\right)^{\mathsf{m}\Phi|\Gamma}$ by definition.

- $\multimap$-FORM: The input types are

$$\mathfrak{r} \mid \underline{\Gamma, \Gamma'} \vdash A \ \mathsf{type}$$
$$\mathfrak{p} \prec (\mathfrak{s} \prec \Xi)\{\Phi\} \otimes \mathfrak{r} \mid \Gamma, \Gamma', x^{\mathfrak{r}} : A \vdash B \ \mathsf{type}$$

Now A is already marked with respect to Φ and Γ, so $A \equiv A^{\mathsf{m}\Phi|\Gamma}$. And by idempotence,

$$\underline{\underline{\Gamma}, \Gamma'} \equiv \underline{\underline{\Gamma}, \Gamma'^{\mathsf{m}\Phi|\Gamma}}$$

so together,

$$\mathfrak{r} \mid \underline{\underline{\Gamma}, \Gamma'^{\mathsf{m}\Phi|\Gamma}} \vdash A^{\mathsf{m}\Phi|\Gamma} \ \mathsf{type}$$

Inductively

$$\mathfrak{p} \prec (\mathfrak{s} \prec \Xi)\{\downarrow 1\} \otimes \mathfrak{r} \mid \underline{\Gamma}, \Gamma'^{\mathsf{m}\Phi|\Gamma}, x^{\mathfrak{r}} : A^{\mathsf{m}\Phi|\Gamma} \vdash B^{\mathsf{m}\Phi|\Gamma} \ \mathsf{type}$$

where we have extended $(\mathfrak{s} \prec \Xi)\{\Phi\}$ spot to $\mathfrak{p} \prec (\mathfrak{s} \prec \Xi)\{\Phi\} \otimes \mathfrak{r}$ spot and the context extension Γ' to $\Gamma', x^{\mathfrak{r}} : A$. Reapplying the rule gives

$$(\mathfrak{s} \prec \Xi)\{\downarrow 1\} \mid \underline{\Gamma}, \Gamma'^{\mathsf{m}\Phi|\Gamma} \vdash \textstyle\coprod_{(x^{\mathfrak{r}} : A^{\mathsf{m}\Phi|\Gamma})}{}^{\mathfrak{p}} B^{\mathsf{m}\Phi|\Gamma}$$

as required.

□

Lemma B.3.17. *Terms in marked context are already dull.*

$$\frac{\mathfrak{t} \prec \Phi \mid \Gamma \ \mathsf{ctx} \qquad \mathfrak{s} \prec \Xi \mid \underline{\Gamma}, \Gamma' \vdash \mathcal{J}}{\mathcal{J} \equiv \mathcal{J}^{\mathsf{m}\Phi|\Gamma}}$$

For this, we do need that $\mathcal{J}$ is well-typed so that variables from Γ are used in the 'correct' way: a well-typed $\mathcal{J}$ can only use variables from Γ via VAR-ZERO, but raw terms might incorrectly use them unmarked.

Proof. On slices, this immediate because none of the colour in Φ occur in s.

On terms, all typing rules monotonically add marks as one moves from the conclusion to the premises, and so all variable uses happen in a context beginning with $\underline{\Gamma}$. The only well-typed such variable uses are already marked, and $-^{\mathsf{m}\Phi|\Gamma}$ has no effect on already marked variables uses. □

B.3.6 Mark-Weakening

$$\text{MARKWK}\ \frac{\mathfrak{t} \mid \underline{\Gamma} \vdash \mathcal{J}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \mathcal{J}}$$

Lemma B.3.18. *We can mark-weaken 'under a slice':*

$$\text{MARKWK/DISPATCH}\ \frac{\begin{array}{c}(\mathfrak{s} \prec \Xi)\{\mathfrak{t} \prec 1, \dots\} \vdash s \ \mathsf{slice} \\ \mathfrak{t} \prec \Phi \mid \Gamma \ \mathsf{ctx} \qquad \mathfrak{s} \prec \Xi\{\mathfrak{t} \prec 1, \dots\} \mid \underline{\Gamma}; \Gamma' \ \mathsf{ext} \\ (\mathfrak{s} \prec \Xi)\{\mathfrak{t} \prec 1, \dots\}^{s} \mid (\underline{\Gamma}, \Gamma')^{s} \vdash \mathcal{J}\end{array}}{(\mathfrak{s} \prec \Xi)\{\downarrow \mathfrak{t} \prec \Phi, \dots\}^{s} \mid (\Gamma, \Gamma')^{s} \vdash \mathcal{J}}$$

where s has itself been mark-weakened in the conclusion by Theorem B.2.20.

Proof. We distinguish two cases:

- s contains the spot $\{\mathsf{t} \prec 1, \dots\}$: Then $(\mathfrak{s} \prec \Xi)^s\{\mathsf{t} \prec 1, \dots\}$ spot and $(\underline{\Gamma}, \Gamma')^s \equiv \underline{\Gamma}, \Gamma'^s$, and applying MARKWK gives

  $$(\mathfrak{s} \prec \Xi)^s\{\downarrow\!\mathsf{t} \prec \Phi, \dots\} \mid \Gamma, \Gamma'^s \vdash \mathcal{J}$$

  And $(\mathfrak{s} \prec \Xi)^s\{\downarrow\!\mathsf{t} \prec \Phi, \dots\} \equiv (\mathfrak{s} \prec \Xi)\{\downarrow\!\mathsf{t} \prec \Phi, \dots\}^s$ by Lemma B.2.2.

- s does not contain $\{\mathsf{t} \prec 1, \dots\}$: Then already $(\mathfrak{s} \prec \Xi)\{\mathsf{t} \prec 1, \dots\}^s \equiv (\mathfrak{s} \prec \Xi)\{\downarrow\!\mathsf{t} \prec \Phi, \dots\}$ and $(\Gamma, \Gamma')^s \equiv \underline{\Gamma}, \Gamma'^s$, so

  $$(\mathfrak{s} \prec \Xi)\{\downarrow\!\mathsf{t} \prec \Phi, \dots\}^s \mid \Gamma, \Gamma'^s \vdash \mathcal{J}$$

$\square$

Theorem B.3.19. *Mark-weakening is admissible on terms.*

$$\textsc{markwk}\ \frac{\begin{array}{c}\mathsf{t} \prec \Phi \mid \Gamma\ \mathsf{ctx} \qquad \mathfrak{s} \prec \Xi\{\mathsf{t} \prec 1, \dots\} \mid \underline{\Gamma}; \Gamma'\ \mathsf{ext} \\ \mathfrak{s} \prec \Xi \mid \underline{\Gamma}, \Gamma' \vdash \mathcal{J}\end{array}}{\mathfrak{s} \prec \Xi\{\downarrow\!\mathsf{t} \prec \Phi, \dots\} \mid \Gamma, \Gamma' \vdash \mathcal{J}}$$

Proof. Induction on derivations. The raw term is unchanged. The only interesting cases are the variable rules, where an instance of VAR-MARKED may become an instance of VAR-ROUNDTRIP if an variable is marked in the premise but not the conclusion.

- VAR: The only case to consider is $x^{\mathfrak{s}} : A \in \Gamma'$; we cannot have $x^{\mathfrak{s}} : A \in \underline{\Gamma}$ because all variables in $\underline{\Gamma}$ are marked by definition. Because we have used the ordinary VAR rule, the colour of x must be the top colour $\mathfrak{s}$. So the derivation to consider is

  $$\frac{\mathfrak{s} \prec \Xi\{\mathsf{t} \prec 1, \dots\} \mid \underline{\Gamma}, \Gamma'_1 \vdash A\ \mathsf{type}}{\mathfrak{s} \prec \Xi\{\mathsf{t} \prec 1, \dots\} \mid \underline{\Gamma}, \Gamma'_1, x^{\mathfrak{s}} : A, \Gamma'_2 \vdash x : A}$$

  By induction also

  $$\mathfrak{s} \prec \Xi\{\downarrow\!\mathsf{t} \prec \Phi, \dots\} \mid \Gamma, \Gamma'_1 \vdash A\ \mathsf{type}$$

  and so we can reapply VAR, giving

  $$\mathfrak{s} \prec \Xi\{\downarrow\!\mathsf{t} \prec \Phi, \dots\} \mid \Gamma, \Gamma'_1, x^{\mathfrak{s}} : A, \Gamma'_2 \vdash x : A$$

- VAR-ROUNDTRIP: Again, x cannot be in Γ, so the derivation to consider is

  $$\frac{(\mathfrak{s} \prec \Xi\{\mathsf{t} \prec 1, \dots\})^{\mathfrak{c}} \mid (\underline{\Gamma}, \Gamma'_1)^{\mathfrak{c}} \vdash A\ \mathsf{type}}{\mathfrak{s} \prec \Xi\{\mathsf{t} \prec 1, \dots\} \mid \underline{\Gamma}, \Gamma'_1, x^{\mathfrak{c}} : A, \Gamma'_2 \vdash \underline{x} : A^{\mathfrak{s} \leftrightarrow \mathfrak{c}}}$$

  This time the colour of x may not be the top colour. First, $\underline{\Gamma}^{\mathfrak{c}} \equiv \underline{\Gamma}$ by Lemma B.1.22. We now distinguish two cases.

– If the spot is contained in the slice $\mathfrak{c}$ then

$$(\mathfrak{s} \prec \Xi)^{\mathfrak{c}}\{\mathfrak{t} \prec 1, \dots\} \mid \underline{\Gamma}, \Gamma_1'^{\mathfrak{c}} \vdash A \text{ type}$$

and inductively

$$(\mathfrak{s} \prec \Xi)^{\mathfrak{c}}\{\downarrow\mathfrak{t} \prec \Phi, \dots\} \mid \Gamma^{\mathfrak{c}}, \Gamma_1'^{\mathfrak{c}} \vdash A \text{ type}$$

allowing us to re-apply the VAR-ROUNDTRIP rule giving $\mathfrak{s} \prec \Xi\{\downarrow\mathfrak{t} \prec \Phi, \dots\} \mid \Gamma, \Gamma_1', x^{\mathfrak{c}} : A, \Gamma_2' \vdash \underline{x} : \underline{A}^{\mathfrak{s}\leftrightarrow\mathfrak{c}}$

– If the spot is not contained in the slice $\mathfrak{c}$ then already $(\mathfrak{s} \prec \Xi\{\mathfrak{t} \prec 1, \dots\})^{\mathfrak{c}} \equiv (\mathfrak{s} \prec \Xi\{\downarrow\mathfrak{t} \prec \Phi, \dots\})^{\mathfrak{c}}$ and $\underline{\Gamma} \equiv \Gamma^{\mathfrak{c}}$ by Lemma B.3.13. And so already

$$(\mathfrak{s} \prec \Xi\{\downarrow\mathfrak{t} \prec \Phi, \dots\})^{\mathfrak{c}} \mid \Gamma^{\mathfrak{c}}, \Gamma_1'^{\mathfrak{c}} \vdash A \text{ type}$$

and we can reapply VAR-ROUNDTRIP.

- VAR-MARKED: Now it is possible that $\underline{x}^{\mathfrak{c}}$ is either in $\underline{\Gamma}$ or Γ'. In the former case we must distinguish two further possibilities; whether x is already marked in the context Γ.

  – $\underline{x}^{\mathfrak{c}} \in \underline{\Gamma}$ and $x^{\mathfrak{c}} \in \Gamma$: Here we will keep more careful track of which variables are being marked in the type. The derivation to consider is

$$\frac{\mathfrak{c} \mid \underline{\underline{\Gamma_1}} \vdash A^{\mathsf{m}\Phi|\Gamma_1} \text{ type}}{\mathfrak{s} \prec \Xi\{\mathfrak{t} \prec 1, \dots\} \mid \underline{\Gamma_1, x^{\mathfrak{c}} : A, \Gamma_2}, \Gamma' \vdash \underline{x} : (A^{\mathsf{m}\Phi|\Gamma_1})^{\mathfrak{s}\leftrightarrow\mathfrak{c}}}$$

  Because the context $\mathfrak{t} \prec \Phi \mid \Gamma_1, x^{\mathfrak{c}} : A, \Gamma_2$ ctx is assumed to be well-formed, we know

$$(\mathfrak{t} \prec \Phi)^{\mathfrak{c}} \mid {\Gamma_1}^{\mathfrak{c}} \vdash A \text{ type}$$

  Because $\mathfrak{c} \in (\mathfrak{t} \prec \Phi)$, we also know $(\mathfrak{s} \prec \Xi\{\downarrow\mathfrak{t} \prec \Phi, \dots\})^{\mathfrak{c}}$ is a cartesian weakening of $(\mathfrak{t} \prec \Phi)^{\mathfrak{c}}$ by Lemma B.2.36, and so we can weaken A to

$$(\mathfrak{s} \prec \Xi\{\downarrow\mathfrak{t} \prec \Phi, \dots\})^{\mathfrak{c}} \mid {\Gamma_1}^{\mathfrak{c}} \vdash A \text{ type}$$

  and apply VAR-ROUNDTRIP giving

$$\mathfrak{s} \prec \Xi\{\downarrow\mathfrak{t} \prec \Phi, \dots\} \mid \Gamma_1, x^{\mathfrak{c}} : A, \Gamma_2, \Gamma' \vdash \underline{x} : (A^{\mathsf{m}\Xi|\Gamma_1})^{\mathfrak{s}\leftrightarrow\mathfrak{c}}$$

  And $(A^{\mathsf{m}\Xi|\Gamma_1})^{\mathfrak{s}\leftrightarrow\mathfrak{c}} \equiv (A^{\mathsf{m}\Phi|\Gamma_1})^{\mathfrak{s}\leftrightarrow\mathfrak{c}}$ because A only contains colours from $(\mathfrak{t} \prec \Phi)$.

  – $\underline{x}^{\mathfrak{c}} \in \underline{\Gamma}$ and $\underline{x}^{\mathfrak{c}} \in \Gamma$: Here the derivation to consider is

$$\frac{\mathfrak{c} \mid \underline{\underline{\Gamma_1}} \vdash A \text{ type}}{\mathfrak{s} \prec \Xi\{\mathfrak{t} \prec 1, \dots\} \mid \underline{\Gamma_1, \underline{x}^{\mathfrak{c}} : A, \Gamma_2}, \Gamma' \vdash \underline{x} : A^{\mathfrak{s}\leftrightarrow\mathfrak{c}}}$$

  By idempotence of marking (Lemma B.1.1), also $\mathfrak{c} \mid \underline{\Gamma_1} \vdash A$ type, and so

$$\mathfrak{s} \prec \Xi\{\downarrow\mathfrak{t} \prec \Phi, \dots\} \mid \Gamma_1, \underline{x}^{\mathfrak{c}} : A, \Gamma_2, \Gamma' \vdash \underline{x} : A^{\mathfrak{s}\leftrightarrow\mathfrak{c}}$$

  is well formed.

– $\underline{x}^{\mathfrak{c}} \in \Gamma'$: The derivation is

$$\frac{\mathfrak{c} \mid \underline{\underline{\Gamma}, \Gamma'_1} \vdash A \ \mathsf{type}}{\mathfrak{s} \prec \Xi\{\mathfrak{t} \prec 1, \dots\} \mid \underline{\Gamma}, \Gamma'_1, \underline{x}^{\mathfrak{c}} : A, \Gamma'_2 \vdash \underline{x} : A^{\mathfrak{s} \leftrightarrow \mathfrak{c}}}$$

and by idempotence, $\underline{\underline{\Gamma}, \Gamma'_1} \equiv \underline{\Gamma, \Gamma'_1}$ and so $\mathfrak{c} \mid \underline{\Gamma, \Gamma'_1} \vdash A$ type. Reapplying VAR-MARKED gives

$$\mathfrak{s} \prec \Xi\{\downarrow\mathfrak{t} \prec \Phi, \dots\} \mid \Gamma, \Gamma'_1, \underline{x}^{\mathfrak{c}} : A, \Gamma'_2 \vdash \underline{x} : A^{\mathfrak{s} \leftrightarrow \mathfrak{c}}$$

as required.

- Π-FORM: The inputs are

$$(\mathfrak{s} \prec \Xi)\{\mathfrak{t} \prec 1, \dots\} \mid \underline{\Gamma}, \Gamma' \vdash A \ \mathsf{type}$$
$$(\mathfrak{s} \prec \Xi)\{\mathfrak{t} \prec 1, \dots\} \mid \underline{\Gamma}, \Gamma', x^{\mathfrak{s}} : A \vdash B \ \mathsf{type}$$

and inductively

$$(\mathfrak{s} \prec \Xi)\{\downarrow\mathfrak{t} \prec \Phi, \dots\} \mid \Gamma, \Gamma' \vdash A \ \mathsf{type}$$
$$(\mathfrak{s} \prec \Xi)\{\downarrow\mathfrak{t} \prec \Phi, \dots\} \mid \Gamma, \Gamma', x^{\mathfrak{s}} : A \vdash B \ \mathsf{type}$$

and we can re-apply Π-FORM.

- ♮-FORM: The input type is

$$\mathfrak{s} \mid \underline{\underline{\Gamma}, \Gamma'} \vdash A \ \mathsf{type}$$

and by idempotence, $\underline{\underline{\Gamma}, \Gamma'} \equiv \underline{\Gamma, \Gamma'}$, so we can reapply ♮-FORM.

- ⊗-INTRO: The inputs are

$$(\mathfrak{s} \prec \Xi)\{\mathfrak{t} \prec 1, \dots\} \vdash s_L \boxtimes s_R \ \mathsf{split}$$
$$(\mathfrak{s} \prec \Xi)\{\mathfrak{t} \prec 1, \dots\}^{s_L} \mid (\underline{\Gamma}, \Gamma')^{s_L} \vdash a : A^{\ulcorner s_L \urcorner \leftrightarrow \mathfrak{t}}$$
$$(\mathfrak{s} \prec \Xi)\{\mathfrak{t} \prec 1, \dots\}^{s_R} \mid (\underline{\Gamma}, \Gamma')^{s_R} \vdash b : B[\underline{a}^{\mathfrak{t} \leftrightarrow \ulcorner s_L \urcorner} / \underline{x}^{\mathfrak{t}}]^{\ulcorner s_R \urcorner \leftrightarrow \mathfrak{t}}$$

Applying Theorem B.2.20 to the split gives $(\mathfrak{s} \prec \Xi)\{\downarrow\mathfrak{t} \prec \Phi, \dots\} \vdash s_L \boxtimes s_R$ split, and applying MARKWK/DISPATCH to the terms gives

$$(\mathfrak{s} \prec \Xi)\{\downarrow\mathfrak{t} \prec \Phi, \dots\}^{s_L} \mid (\Gamma, \Gamma')^{s_L} \vdash a : A^{\ulcorner s_L \urcorner \leftrightarrow \mathfrak{t}}$$
$$(\mathfrak{s} \prec \Xi)\{\downarrow\mathfrak{t} \prec \Phi, \dots\}^{s_R} \mid (\Gamma, \Gamma')^{s_R} \vdash b : B[\underline{a}^{\mathfrak{t} \leftrightarrow \ulcorner s_L \urcorner} / \underline{x}^{\mathfrak{t}}]^{\ulcorner s_R \urcorner \leftrightarrow \mathfrak{t}}$$

and so we can reapply the rule.

- ⊸-FORM: The input types are

$$\mathfrak{r} \mid \underline{\underline{\Gamma}, \Gamma'} \vdash A \ \mathsf{type}$$
$$\mathfrak{p} \prec (\mathfrak{s} \prec \Xi)\{\mathfrak{t} \prec 1, \dots\} \otimes \mathfrak{r} \mid \underline{\Gamma}, \Gamma', x^{\mathfrak{r}} : A \vdash B \ \mathsf{type}$$

As in the case for ♮-FORM, idempotence gives

$$\underline{\underline{\Gamma}, \Gamma'} \equiv \underline{\Gamma, \Gamma'}$$

and so $\mathfrak{r} \mid \underline{\Gamma, \Gamma'} \vdash A$ type. Induction on B gives

$$\mathfrak{p} \prec (\mathfrak{s} \prec \Xi)\{\downarrow\mathfrak{t} \prec \Phi, \dots\} \otimes \mathfrak{r} \mid \Gamma, \Gamma', x^{\mathfrak{r}} : A \vdash B \ \mathsf{type}$$

and so we can reapply the rule.

□

B.3.7 Cartesian Substitution

$$\textsc{subst}\ \dfrac{\mathfrak{t} \prec \Phi \mid \Gamma \vdash (\kappa \mid \theta) : \Psi \mid \Omega \qquad \mathfrak{t} \prec \Phi, \Psi \mid \Gamma, \Omega, \Gamma' \vdash \mathcal{J}}{\mathfrak{t} \prec \Phi \mid \Gamma, \Gamma'[\kappa \mid \theta] \vdash \mathcal{J}[\kappa \mid \theta]}$$

There are several generalised versions, all of which operate the same way on raw syntax:

$$\textsc{subst}\ \dfrac{\begin{array}{c}\mathfrak{s} \prec \Xi\{\mathfrak{t} \prec \Phi, \Psi, \dots\}\ \mathsf{spot} \\ \mathfrak{t} \prec \Phi \mid \Gamma \vdash (\kappa \mid \theta) : \Psi \mid \Omega \qquad \mathfrak{s} \prec \Xi\{\mathfrak{t} \prec \Phi, \Psi, \dots\} \mid \Gamma, \Omega, \Gamma' \vdash \mathcal{J}\end{array}}{\mathfrak{s} \prec \Xi\{\downarrow\mathfrak{t} \prec \Phi, \dots\} \mid \Gamma, \Gamma'[\kappa \mid \theta] \vdash \mathcal{J}[\kappa \mid \theta]^{\mathsf{at}\mathfrak{s}}}$$

$$\textsc{subst/cod}\ \dfrac{\begin{array}{c}\Psi \vdash v\ \mathsf{preslice} \qquad \mathfrak{s} \prec \Xi\{\Psi^{v}\}\ \mathsf{spot} \\ \mathfrak{t} \prec \Phi \mid \Gamma \vdash (\kappa \mid \theta) : \Psi \mid \Omega \qquad \mathfrak{s} \prec \Xi\{\Psi^{v}\} \mid \underline{\Gamma}, \Omega^{v}, \Gamma' \vdash \mathcal{J}\end{array}}{\mathfrak{s} \prec \Xi\{\downarrow(\mathfrak{t} \prec \Phi)^{v[\kappa]}\} \mid \Gamma^{v[\kappa]}, \Gamma'[\kappa \mid \theta] \vdash \mathcal{J}[\kappa \mid \theta]^{\mathsf{at}\mathfrak{s}}}$$

$$\textsc{subst/dom}\ \dfrac{\begin{array}{c}\Phi \vdash w\ \mathsf{preslice} \qquad \mathfrak{s} \prec \Xi\{\Phi^{w}\}\ \mathsf{spot} \\ \mathfrak{t} \prec \Phi \mid \Gamma \vdash (\kappa \mid \theta) : \Psi \mid \Omega \qquad \mathfrak{s} \prec \Xi\{\Phi^{w}\} \mid \Gamma^{w}, \underline{\Omega}, \Gamma' \vdash \mathcal{J}\end{array}}{\mathfrak{s} \prec \Xi\{\Phi^{w}\} \mid \Gamma^{w}, \Gamma'[\kappa \mid \theta] \vdash \mathcal{J}[\kappa \mid \theta]^{\mathsf{at}\mathfrak{s}}}$$

$$\textsc{subst/marked}\ \dfrac{\mathfrak{t} \prec \Phi \mid \Gamma \vdash (\kappa \mid \theta) : \Psi \mid \Omega \qquad \mathfrak{s} \prec \Xi \mid \underline{\Gamma}, \underline{\Omega}, \Gamma' \vdash \mathcal{J}}{\mathfrak{s} \prec \Xi \mid \underline{\Gamma}, \Gamma'[\kappa \mid \theta] \vdash \mathcal{J}[\kappa \mid \theta]^{\mathsf{at}\mathfrak{s}}}$$

In the SUBST/COD case, the top colour of the conclusion might not be the same as the top colour of $\mathcal{J}$, in particular, $\mathcal{J}[\kappa \mid \theta]^{\mathsf{at}\mathfrak{s}}$ does not necessarily have top colour $\mathfrak{s}$.

To fractionally save on space, we will write $\mathcal{J}[\kappa \mid \theta]^{\mathsf{at}\mathfrak{s}}$ for what should really be $\mathcal{J}[\kappa \mid \theta]^{\mathsf{at}\ulcorner s \urcorner}$, letting the slice stand in for its top colour.

Lemma B.3.20. *Let* $\mathfrak{s} \prec \Xi\{(\mathfrak{t} \prec \Phi, \Psi)^{?}\} \mid \Gamma, \Omega, \Gamma'$ ctx *denote one of the above four situations, so one of*

$$\begin{array}{lll}
\mathfrak{s} \prec \Xi\{\mathfrak{t} \prec \Phi, \Psi, \dots\} & \mid \Gamma, \Omega, \Gamma' & \mathsf{ctx} \\
\mathfrak{s} \prec \Xi\{\Psi^{v}\} & \mid \underline{\Gamma}, \Omega^{v}, \Gamma' & \mathsf{ctx}\ \textit{where}\ \Psi \vdash v\ \mathsf{preslice} \\
\mathfrak{s} \prec \Xi\{\Phi^{w}\} & \mid \Gamma^{w}, \underline{\Omega}, \Gamma' & \mathsf{ctx}\ \textit{where}\ \Phi \vdash w\ \mathsf{preslice} \\
\mathfrak{s} \prec \Xi & \mid \underline{\Gamma}, \underline{\Omega}, \Gamma' & \mathsf{ctx}
\end{array}$$

and $\mathfrak{s} \prec \Xi\{\downarrow(\mathfrak{t} \prec \Phi)^{?[\kappa]}\} \mid \Gamma, \Gamma'[\kappa \mid \theta]$ ctx *the corresponding conclusion contexts*

$$\begin{array}{lll}
\mathfrak{s} \prec \Xi\{\downarrow\mathfrak{t} \prec \Phi, \dots\} & \mid \Gamma, \Gamma'[\kappa \mid \theta] & \mathsf{ctx} \\
\mathfrak{s} \prec \Xi\{\downarrow(\mathfrak{t} \prec \Phi)^{v[\kappa]}\} & \mid \Gamma^{v[\kappa]}, \Gamma'[\kappa \mid \theta] & \mathsf{ctx}\ \textit{where}\ \Psi \vdash v\ \mathsf{preslice} \\
\mathfrak{s} \prec \Xi\{\Phi^{w}\} & \mid \Gamma^{w}, \Gamma'[\kappa \mid \theta] & \mathsf{ctx}\ \textit{where}\ \Phi \vdash w\ \mathsf{preslice} \\
\mathfrak{s} \prec \Xi & \mid \underline{\Gamma}, \Gamma'[\kappa \mid \theta] & \mathsf{ctx}
\end{array}$$

respectively. In each case we can substitute 'under a slice':

$$\textsc{subst/dispatch}\ \dfrac{\begin{array}{c}\mathfrak{s} \prec \Xi\{(\mathfrak{t} \prec \Phi, \Psi)^{?}\} \vdash s\ \mathsf{slice} \\ \mathfrak{t} \prec \Phi \mid \Gamma \vdash (\kappa \mid \theta) : \Psi \mid \Omega \qquad (\mathfrak{s} \prec \Xi\{\downarrow(\mathfrak{t} \prec \Phi, \Psi)^{?}\})^{s} \mid (\Gamma, \Omega, \Gamma')^{s} \vdash \mathcal{J}\end{array}}{(\mathfrak{s} \prec \Xi\{\downarrow(\mathfrak{t} \prec \Phi)^{?[\kappa]}\})^{s[\kappa]} \mid (\Gamma, \Gamma'[\kappa \mid \theta])^{s[\kappa]} \vdash \mathcal{J}[\kappa \mid \theta]^{\mathsf{ats}}}$$

Proof.

- SUBST: There are four cases:
  - s contains all of $\mathfrak{t} \prec \Phi, \Psi, \dots$: Then

    $$(\mathfrak{s} \prec \Xi)^{s}\{\mathfrak{t} \prec \Phi, \Psi, \dots\} \mid \Gamma, \Omega, \Gamma'^{s} \vdash \mathcal{J}$$

    Then applying SUBST gives

    $$(\mathfrak{s} \prec \Xi)^{s}\{\downarrow\mathfrak{t} \prec \Phi, \dots\} \mid \Gamma, \Gamma'^{s}[\kappa \mid \theta] \vdash \mathcal{J}[\kappa \mid \theta]^{\mathsf{ats}}$$

    By Lemma B.2.2,

    $$(\mathfrak{s} \prec \Xi)^{s}\{\downarrow\mathfrak{t} \prec \Phi, \dots\} \equiv (\mathfrak{s} \prec \Xi\{\downarrow\mathfrak{t} \prec \Phi, \dots\})^{s}$$

    In this case $s \equiv s[\kappa]$, and by Lemma B.1.23,

    $$\Gamma'[\kappa \mid \theta]^{s[\kappa]} \equiv \Gamma'^{s}[\kappa \mid \theta]$$

    so

    $$(\mathfrak{s} \prec \Xi\{\downarrow\mathfrak{t} \prec \Phi, \dots\})^{s} \mid (\Gamma, \Gamma'[\kappa \mid \theta])^{s[\kappa]} \vdash \mathcal{J}[\kappa \mid \theta]^{\mathsf{ats}}$$

    as required.
  - s intersects Ψ: Then $\Psi \vdash s \cap \Psi$ slice and $\Xi^{s}\{\Psi^{s\cap\Psi}\}$ spot. The input judgement is

    $$(\mathfrak{s} \prec \Xi)^{s}\{\Psi^{s\cap\Psi}\} \mid \underline{\Gamma}, \Omega^{s\cap\Psi}, \Gamma'^{s} \vdash \mathcal{J}$$

    and applying SUBST/COD,

    $$(\mathfrak{s} \prec \Xi)^{s}\{\downarrow(\mathfrak{t} \prec \Phi)^{(s\cap\Psi)[\kappa]}\} \mid \Gamma^{(s\cap\Psi)[\kappa]}, \Gamma'^{s}[\kappa \mid \theta] \vdash \mathcal{J}[\kappa \mid \theta]^{\mathsf{ats}}$$

This is the correct palette by Lemma B.2.27. Also

$$\Gamma^{(s\cap\Psi)[\kappa]} \equiv \Gamma^{s[\kappa]}$$

by Lemma B.3.12 and

$$\Gamma'^{s}[\kappa \mid \theta] \equiv \Gamma'[\kappa \mid \theta]^{s[\kappa]}$$

by Lemma B.1.23. And so we have

$$(\mathfrak{s} \prec \Xi\{\downarrow(\mathfrak{t} \prec \Phi)\})^{s[\kappa]} \mid (\Gamma, \Gamma'[\kappa \mid \theta])^{s[\kappa]} \vdash \mathcal{J}[\kappa \mid \theta]^{\mathrm{ats}}$$

as required.

– s intersects Φ: Then $\Phi \vdash s \cap \Phi$ slice and $\Xi^{s}\{\Phi^{s\cap\Phi}\}$ spot. The input judgement is

$$(\mathfrak{s} \prec \Xi)^{s}\{\Phi^{s\cap\Phi}\} \mid \Gamma^{s\cap\Phi}, \underline{\Omega}, \Gamma'^{s} \vdash \mathcal{J}$$

and applying SUBST/DOM,

$$(\mathfrak{s} \prec \Xi)^{s}\{\Phi^{s\cap\Phi}\} \mid \Gamma^{s\cap\Phi}, \Gamma'^{s}[\kappa \mid \theta] \vdash \mathcal{J}[\kappa \mid \theta]^{\mathrm{ats}}$$

This palette is equal to $(\mathfrak{s} \prec \Xi\{\downarrow\mathfrak{t} \prec \Phi, \dots\})^{s}$ by Lemma B.2.1. Because s does not intersect Ψ, $s[\kappa] \equiv s$. Then Lemma B.3.12 gives

$$\Gamma^{s\cap\Phi} \equiv \Gamma^{s} \equiv \Gamma^{s[\kappa]}$$

because Ψ does not occur in Γ, and

$$\Gamma'^{s}[\kappa \mid \theta] \equiv \Gamma'[\kappa \mid \theta]^{s[\kappa]}$$

by Lemma B.1.23. So

$$(\mathfrak{s} \prec \Xi\{\mathfrak{t} \prec \Phi, \dots\})^{s[\kappa]} \mid (\Gamma, \Gamma'[\kappa \mid \theta])^{s[\kappa]} \vdash \mathcal{J}[\kappa \mid \theta]^{\mathrm{ats}}$$

as required.

– s does not intersect $\mathfrak{t} \prec \Phi, \Psi, \dots$: Then the input is

$$(\mathfrak{s} \prec \Xi)^{s} \mid \underline{\Gamma}, \underline{\Omega}, \Gamma'^{s} \vdash \mathcal{J}$$

and applying SUBST/MARKED gives

$$(\mathfrak{s} \prec \Xi)^{s} \mid \underline{\Gamma}, \Gamma'^{s}[\kappa \mid \theta] \vdash \mathcal{J}[\kappa \mid \theta]^{\mathrm{ats}}$$

and Lemma B.1.23 gives that $\underline{\Gamma}, \Gamma'^{s}[\kappa \mid \theta] \equiv (\underline{\Gamma}, \Gamma'[\kappa \mid \theta])^{s[\kappa]}$ as required.

- SUBST/COD: There are two cases:

– s intersects Ψ^v: Then $\Psi \vdash s \cap v$ preslice, and the input is

$$(\mathfrak{s} \prec \Xi)^s\{\Psi^{s\cap v}\} \mid \underline{\Gamma}, \Omega^{s\cap v}, \Gamma'^s \vdash \mathcal{J}$$

where $(\mathfrak{s} \prec \Xi)^s\{\Psi^{s\cap v}\}$ spot by Lemma B.2.32 and $(\underline{\Gamma})^s \equiv \underline{\Gamma}$ by Lemma B.1.22. Then we can reapply SUBST/COD, giving

$$(\mathfrak{s} \prec \Xi)^s\{\downarrow(\mathfrak{t} \prec \Phi)^{(s\cap v)[\kappa]}\} \mid \Gamma^{(s\cap v)[\kappa]}, \Gamma'^s[\kappa \mid \theta] \vdash \mathcal{J}[\kappa \mid \theta]^{\mathsf{ats}}$$

This is the correct palette by Lemma B.2.33. Also

$$\Gamma^{(s\cap v)[\kappa]} \equiv (\Gamma^{v[\kappa]})^{s[\kappa]}$$

by Lemma B.3.10 and

$$\Gamma'^s[\kappa \mid \theta] \equiv \Gamma'[\kappa \mid \theta]^{s[\kappa]}$$

by Lemma B.1.23. And so we have

$$(\mathfrak{s} \prec \Xi\{\downarrow(\mathfrak{t} \prec \Phi)^{v[\kappa]}\})^{s[\kappa]} \mid (\Gamma^{v[\kappa]}, \Gamma'[\kappa \mid \theta])^{s[\kappa]} \vdash \mathcal{J}[\kappa \mid \theta]^{\mathsf{ats}}$$

as required.

– s is disjoint from Ψ^v: Then the input is

$$(\mathfrak{s} \prec \Xi)^s \mid \underline{\Gamma}, \underline{\Omega}, \Gamma'^s \vdash \mathcal{J}$$

to which we can apply SUBST/MARKED, giving

$$(\mathfrak{s} \prec \Xi)^s \mid \underline{\Gamma}, \Gamma'^s[\kappa \mid \theta] \vdash \mathcal{J}[\kappa \mid \theta]^{\mathsf{ats}}$$

Because s does not intersect the spot, we know $(\mathfrak{s} \prec \Xi)^s \equiv (\mathfrak{s} \prec \Xi\{\downarrow(\mathfrak{t} \prec \Phi)^{v[\kappa]}\})^s$, so this is already the correct palette. Again, $\Gamma'[\kappa \mid \theta]^{s[\kappa]} \equiv \Gamma'^s[\kappa \mid \theta]$ by Lemma B.1.23.

• SUBST/DOM: There are two cases:

– s intersects Φ^w: Then $\Phi \vdash s \cap w$ preslice$_\epsilon$, and the input is

$$(\mathfrak{s} \prec \Xi)^s\{\Phi^{s\cap v}\} \mid \Gamma^{s\cap w}, \underline{\Omega}, \Gamma'^s \vdash \mathcal{J}$$

Then applying SUBST/DOM gives

$$(\mathfrak{s} \prec \Xi)^s\{\Psi^{s\cap v}\} \mid \Gamma^{s\cap w}, \Gamma'^s[\kappa \mid \theta] \vdash \mathcal{J}[\kappa \mid \theta]^{\mathsf{ats}}$$

And $\Gamma^{s\cap w}, \Gamma'^s[\kappa \mid \theta] \equiv (\Gamma^w, \Gamma'[\kappa \mid \theta])^s$ by Lemma B.3.8 and Lemma B.1.23.

– s is disjoint from Φ^w: Then the input is

$$(\mathfrak{s} \prec \Xi)^s \mid \underline{\Gamma}, \underline{\Omega}, \Gamma'^s \vdash \mathcal{J}$$

to which we can apply SUBST/MARKED, giving

$$(\mathfrak{s} \prec \Xi)^s \mid \underline{\Gamma}, \Gamma'^s[\kappa \mid \theta] \vdash \mathcal{J}[\kappa \mid \theta]^{\mathsf{ats}}$$

This is already the correct palette, and $\Gamma'[\kappa \mid \theta]^{s[\kappa]} \equiv \Gamma'^s[\kappa \mid \theta]$ by Lemma B.1.23.

- SUBST/MARKED: The input is equal to

$$(\mathfrak{s} \prec \Xi)^s \mid \underline{\Gamma}, \underline{\Omega}, \Gamma'^s \vdash \mathcal{J}$$

and so we can apply SUBST/MARKED giving

$$(\mathfrak{s} \prec \Xi)^s \mid \underline{\Gamma}, \Gamma'^s[\kappa \mid \theta] \vdash \mathcal{J}[\kappa \mid \theta]^{\mathrm{ats}}$$

By Lemma B.1.23, this is the correct context, and $s[\kappa] \equiv s$ because s is fresh for κ.

□

Theorem B.3.21. *Substitution on terms is admissible.*

$$\text{SUBST}\ \frac{\mathfrak{t} \prec \Phi \mid \Gamma \vdash (\kappa \mid \theta) : \Psi \mid \Omega \qquad \mathfrak{t} \prec \Phi, \Psi \mid \Gamma, \Omega, \Gamma' \vdash a : A}{\mathfrak{t} \prec \Phi \mid \Gamma, \Gamma'[\kappa \mid \theta] \vdash a[\kappa \mid \theta] : A[\kappa \mid \theta]}$$

Proof. As usual, each variable rule has cases for the position of the variable in the context.

Variables:

- VAR:
  - $x^{\mathfrak{s}} : A \in \Gamma$: Then also $x^{\mathfrak{s}} : A \in \Gamma$ in the conclusion, and so we can reapply VAR.
  - $x^{\mathfrak{s}} : A \in \Omega$: Then we must have $\mathfrak{s} \equiv \mathfrak{t}$, and the palette is a cartesian extension $\mathfrak{t} \prec \Phi, \Psi, \Phi'$ of $\mathfrak{t} \prec \Phi$. The substitution has the form

    $$\mathfrak{t} \prec \Phi \mid \Gamma \vdash (\kappa \mid \theta, a/x^{\mathfrak{s}}, \theta') : \Psi \mid \Omega, x^{\mathfrak{s}} : A, \Omega'$$

    where

    $$\mathfrak{t} \prec \Phi \mid \Gamma \vdash a : A[\kappa \mid \theta]^{\mathrm{att}}$$

    and so by weakening, also

    $$\mathfrak{t} \prec \Phi, \Phi' \mid \Gamma, \Gamma'[\kappa \mid \theta, a/x^{\mathfrak{s}}, \theta']^{\mathrm{att}} \vdash a : A[\kappa \mid \theta]^{\mathrm{att}}$$

    and finally, because x and Ω' do not occur in A,

    $$\mathfrak{t} \prec \Phi, \Phi' \mid \Gamma, \Gamma'[\kappa \mid \theta, a/x^{\mathfrak{s}}, \theta']^{\mathrm{att}} \vdash a : A[\kappa \mid \theta, a/x^{\mathfrak{s}}, \theta']^{\mathrm{att}}$$

  - $x^{\mathfrak{s}} : A \in \Gamma'$: Then $x^{\mathfrak{s}} : A[\kappa \mid \theta]^{\mathrm{att}} \in \Gamma'[\kappa \mid \theta]$, and so we can reapply VAR.

- VAR/COD:
  - $x^{\mathfrak{s}} : A \in \underline{\Gamma}$: Cannot occur, as all variables in $\underline{\Gamma}$ are marked.

– $x^{\mathfrak{s}} : A \in \Omega^{v}$: We must have $v \equiv \mathfrak{s}$. The substitution has the form

$$\mathfrak{t} \prec \Phi \mid \Gamma \vdash (\kappa \mid \theta, a/x^{\mathfrak{s}}, \theta') : \Psi \mid \Omega, x^{\mathfrak{s}} : A, \Omega'$$

where

$$(\mathfrak{t} \prec \Phi)^{\mathfrak{s}[\kappa]} \mid \Gamma^{\mathfrak{s}[\kappa]} \vdash a : A[\kappa \mid \theta]^{\mathfrak{ats}[\kappa]}$$

Again, by weakening and that x and the variables in Ω' do not occur in A, also

$$(\mathfrak{t} \prec \Phi)^{\mathfrak{s}[\kappa]} \mid \Gamma^{\mathfrak{s}[\kappa]}, \Gamma'[\kappa \mid \theta, a/x^{\mathfrak{s}}, \theta']^{\mathfrak{ats}[\kappa]} \vdash a : A[\kappa \mid \theta, a/x^{\mathfrak{s}}, \theta']^{\mathfrak{ats}[\kappa]}$$

– $x^{\mathfrak{s}} : A \in \Gamma'$: Then $x^{\mathfrak{s}} : A[\kappa \mid \theta]^{\mathfrak{att}} \in \Gamma'[\kappa \mid \theta]$, and so we can reapply VAR.

- VAR/DOM:
  - $x^{\mathfrak{s}} : A \in \Gamma^{w}$: We must have $w \equiv \mathfrak{s}$. Then also $x^{\mathfrak{s}} : A \in \Gamma^{w}$ in the conclusion, and we can reapply VAR.
  - $x^{\mathfrak{s}} : A \in \underline{\Omega}$: Cannot occur, as all variables in $\underline{\Omega}$ are marked.
  - $x^{\mathfrak{s}} : A \in \Gamma'$: Then $x^{\mathfrak{s}} : A[\kappa \mid \theta]^{\mathfrak{att}} \in \Gamma'[\kappa \mid \theta]$, and so we can reapply VAR.
- VAR/MARKED:
  - $x^{\mathfrak{s}} : A \in \underline{\Gamma}$: Cannot occur, as all variables in $\underline{\Gamma}$ are marked.
  - $x^{\mathfrak{s}} : A \in \underline{\Omega}$: Cannot occur, as all variables in $\underline{\Omega}$ are marked.
  - $x^{\mathfrak{s}} : A \in \Gamma'$: Then $x^{\mathfrak{s}} : A[\kappa \mid \theta]^{\mathfrak{att}} \in \Gamma'[\kappa \mid \theta]$, and so we can reapply VAR.
- VAR-ROUNDTRIP:
  - $x^{\mathfrak{c}} : A \in \Gamma$: Then also $x^{\mathfrak{c}} : A \in \Gamma$ in the conclusion, and so we can reapply VAR-ROUNDTRIP.
  - $x^{\mathfrak{c}} : A \in \Omega$: The substitution has the form

$$\mathfrak{t} \prec \Phi \mid \Gamma \vdash (\kappa \mid \theta, a/x^{\mathfrak{c}}, \theta') : \Psi \mid \Omega, x^{\mathfrak{c}} : A, \Omega'$$

where

$$(\mathfrak{t} \prec \Phi)^{\mathfrak{c}[\kappa]} \mid \Gamma^{\mathfrak{c}[\kappa]} \vdash a : A[\kappa \mid \theta]^{\mathfrak{atc}[\kappa]}$$

Marking followed by relabelling gives

$$\mathfrak{s} \mid \underline{\Gamma} \vdash \underline{a}^{\mathfrak{s}\leftrightarrow\mathfrak{c}[\kappa]} : \underline{A[\kappa \mid \theta]^{\mathfrak{atc}[\kappa]}}^{\mathfrak{s}\leftrightarrow\mathfrak{c}[\kappa]}$$

This can be mark-weakened to

$$\mathfrak{s} \prec \Xi\{\mathfrak{t} \prec \Phi, \dots\} \mid \Gamma \vdash \underline{a}^{\mathfrak{s}\leftrightarrow\mathfrak{c}[\kappa]} : \underline{A[\kappa \mid \theta]^{\mathfrak{atc}[\kappa]}}^{\mathfrak{s}\leftrightarrow\mathfrak{c}[\kappa]}$$

and then weakened to

$$\mathfrak{s} \prec \Xi\{\mathfrak{t} \prec \Phi, \dots\} \mid \Gamma, \Gamma'[\kappa \mid \theta]^{\mathrm{at}\mathfrak{s}} \vdash \underline{a}^{\mathfrak{s}\leftrightarrow\mathfrak{c}[\kappa]} : \underline{A[\kappa \mid \theta]^{\mathrm{at}\mathfrak{c}[\kappa]}}^{\mathfrak{s}\leftrightarrow\mathfrak{c}[\kappa]}$$

Finally, the type is

$$\underline{A[\kappa \mid \theta]^{\mathrm{at}\mathfrak{c}[\kappa]}}^{\mathfrak{s}\leftrightarrow\mathfrak{c}[\kappa]} \equiv (\underline{A}[\kappa \mid \theta]^{\mathrm{at}\mathfrak{c}[\kappa]})^{\mathfrak{s}\leftrightarrow\mathfrak{c}[\kappa]} \equiv \underline{A}^{\mathfrak{s}\leftrightarrow\mathfrak{c}}[\kappa \mid \theta]^{\mathrm{at}\mathfrak{s}}$$

by Lemma B.1.4 followed by Lemma B.1.5.

– $x^{\mathfrak{c}} : A \in \Gamma'$: Then $x^{\mathfrak{c}} : A[\kappa \mid \theta]^{\mathrm{at}\mathfrak{c}} \in \Gamma'[\kappa \mid \theta]$ and so

$$\mathfrak{s} \prec \Xi\{\mathfrak{t} \prec \Phi, \dots\} \mid \Gamma, \Gamma'[\kappa \mid \theta]^{\mathrm{at}\mathfrak{s}} \vdash \underline{x} : (\underline{A[\kappa \mid \theta]^{\mathrm{at}\mathfrak{c}}})^{\mathfrak{s}\leftrightarrow\mathfrak{c}}$$

Again,

$$(\underline{A[\kappa \mid \theta]^{\mathrm{at}\mathfrak{c}}})^{\mathfrak{s}\leftrightarrow\mathfrak{c}} \equiv (\underline{A}[\kappa \mid \theta]^{\mathrm{at}\mathfrak{c}})^{\mathfrak{s}\leftrightarrow\mathfrak{c}} \equiv \underline{A}^{\mathfrak{s}\leftrightarrow\mathfrak{c}}[\kappa \mid \theta]^{\mathrm{at}\mathfrak{s}}$$

by Lemma B.1.4 followed by Lemma B.1.5, because $\mathfrak{c}$ is not substituted for by κ.

- VAR-ROUNDTRIP/COD:
  - $x^{\mathfrak{c}} : A \in \underline{\Gamma}$: Cannot occur, as all variables in $\underline{\Gamma}$ are marked.
  - $x^{\mathfrak{c}} : A \in \Omega^{v}$: Similar to the case for VAR-ROUNDTRIP, instead mark-weakening $\underline{\Gamma}$ to $\Gamma^{v[\kappa]}$
  - $x^{\mathfrak{c}} : A \in \Gamma'$: Then $x^{\mathfrak{c}} : A[\kappa \mid \theta]^{\mathrm{at}\mathfrak{t}} \in \Gamma'[\kappa \mid \theta]$, and so we can reapply VAR-ROUNDTRIP.
- VAR-ROUNDTRIP/DOM:
  - $x^{\mathfrak{c}} : A \in \Gamma^{w}$: Then $x^{\mathfrak{c}} : A \in \Gamma^{w}$, in the conclusion and so we can reapply VAR-ROUNDTRIP.
  - $x^{\mathfrak{c}} : A \in \underline{\Omega}$: Cannot occur, as all variables in $\underline{\Omega}$ are marked.
  - $x^{\mathfrak{c}} : A \in \Gamma'$: Then $x^{\mathfrak{c}} : A[\kappa \mid \theta]^{\mathrm{at}\mathfrak{t}} \in \Gamma'[\kappa \mid \theta]$, and so we can reapply VAR-ROUNDTRIP.
- VAR-ROUNDTRIP/MARKED:
  - $x^{\mathfrak{c}} : A \in \underline{\Gamma}$: Cannot occur, as all variables in $\underline{\Gamma}$ are marked.
  - $x^{\mathfrak{c}} : A \in \underline{\Omega}$: Cannot occur, as all variables in $\underline{\Omega}$ are marked.
  - $x^{\mathfrak{c}} : A \in \Gamma'$: Then $x^{\mathfrak{c}} : A[\kappa \mid \theta]^{\mathrm{at}\mathfrak{t}} \in \Gamma'[\kappa \mid \theta]$, and so we can reapply VAR-ROUNDTRIP.
- VAR-MARKED:
  - $\underline{x}^{\mathfrak{c}} : A \in \Gamma$: Then also $\underline{x}^{\mathfrak{c}} : A \in \Gamma$ in the conclusion, so we can reapply VAR-MARKED.
  - $\underline{x}^{\mathfrak{c}} : A \in \Omega$: Then the substitution contains

$$\mathfrak{c} \mid \underline{\Gamma} \vdash a : A[\kappa \mid \theta]^{\mathrm{at}\mathfrak{c}}$$

and this can be re-labelled and mark-weakened to

$$\mathfrak{s} \prec \Xi\{\downarrow\mathfrak{t} \prec \Phi, \dots\} \mid \Gamma \vdash a^{\mathfrak{s}\leftrightarrow\mathfrak{c}} : (A[\kappa \mid \theta]^{\mathrm{at}\mathfrak{c}})^{\mathfrak{s}\leftrightarrow\mathfrak{c}}$$

Finally, $(A[\kappa \mid \theta]^{\mathrm{at}\mathfrak{c}})^{\mathfrak{s}\leftrightarrow\mathfrak{c}} \equiv (A[\kappa \mid \theta]^{\mathfrak{s}\leftrightarrow\mathfrak{c}})^{\mathrm{at}\mathfrak{s}}$ by Lemma B.1.5, using that $\mathfrak{s}[\kappa] \equiv \mathfrak{s}$ and $\mathfrak{c}[\kappa] \equiv \mathfrak{c}$ in this case.

  - $\underline{x}^{\mathfrak{c}} : A \in \Gamma'$: Then $\underline{x}^{\mathfrak{c}} : A[\kappa \mid \theta]^{\mathsf{atc}} \in \Gamma'[\kappa \mid \theta]$, so we can reapply the rule.

- VAR-MARKED/COD:
  - $\underline{x}^{\mathfrak{c}} : A \in \underline{\Gamma}$: There are two further cases, depending on whether $\underline{x}$ is marked in the conclusion. This is determined by whether $\mathfrak{c}$ is contained in the slice $v[\kappa]$.
    - $\mathfrak{c} \in v[\kappa]$: Then there is $x^{\mathfrak{c}} : A' \in \Gamma^{v[\kappa]}$, where A' is a type such that $\underline{A'} \equiv A$. And so we may use VAR-ROUNDTRIP to form

$$\mathfrak{s} \prec \Xi\{\downarrow(\mathfrak{t} \prec \Phi)^{v[\kappa]}\} \mid \Gamma^{v[\kappa]}, \Gamma'[\kappa \mid \theta] \vdash \underline{x}^{\mathfrak{s}\leftrightarrow\mathfrak{c}} : \underline{A'}$$

      whose type is $\underline{A'} \equiv A \equiv A[\kappa \mid \theta]^{\mathsf{ats}}$, because A is fresh for θ.
    - $\mathfrak{c} \notin v[\kappa]$: Then there is $\underline{x}^{\mathfrak{c}} : A \in \Gamma^{v[\kappa]}$, and we can reapply VAR-MARKED.
  - $\underline{x}^{\mathfrak{c}} : A \in \Omega^{v}$: There are two further cases, depending on whether $\underline{x}$ is marked in the original telescope Ω.
    - $x^{\mathfrak{c}} : A \in \Omega$: The substitution has the form

$$\mathfrak{t} \prec \Phi \mid \Gamma \vdash (\kappa \mid \theta, a/x^{\mathfrak{c}}, \theta') : \Psi \mid \Omega, x^{\mathfrak{c}} : A, \Omega'$$

      where

$$(\mathfrak{t} \prec \Phi)^{\mathfrak{c}[\kappa]} \mid \Gamma^{\mathfrak{c}[\kappa]} \vdash a : A[\kappa \mid \theta]^{\mathsf{atc}[\kappa]}$$

      Marking this term gives

$$\ulcorner \mathfrak{c}[\kappa] \urcorner \mid \underline{\Gamma^{\mathfrak{c}[\kappa]}} \vdash \underline{a} : \underline{A[\kappa \mid \theta]^{\mathsf{atc}[\kappa]}}$$

      By Lemma B.1.22 and Lemma B.1.4 on the context and type respectively, this is

$$\ulcorner \mathfrak{c}[\kappa] \urcorner \mid \underline{\Gamma} \vdash \underline{a} : \underline{A}[\kappa \mid \theta]^{\mathsf{atc}[\kappa]}$$

      which can be recoloured and then silently mark-weakened and weakened to

$$\mathfrak{s} \prec \Xi\{\downarrow(\mathfrak{t} \prec \Phi)^{v[\kappa]}\} \mid \Gamma^{v[\kappa]}, \Gamma'[\kappa \mid \theta] \vdash \underline{a}^{\mathfrak{s}\leftrightarrow\ulcorner \mathfrak{c}[\kappa] \urcorner} : (\underline{A}[\kappa \mid \theta]^{\mathsf{atc}[\kappa]})^{\mathfrak{s}\leftrightarrow\ulcorner \mathfrak{c}[\kappa] \urcorner}$$

      Finally $(\underline{A}[\kappa \mid \theta]^{\mathsf{atc}[\kappa]})^{\mathfrak{s}\leftrightarrow\ulcorner \mathfrak{c}[\kappa] \urcorner} \equiv (\underline{A}^{\mathfrak{s}\leftrightarrow\ulcorner \mathfrak{c}[\kappa] \urcorner})[\kappa \mid \theta]^{\mathsf{ats}}$ as required.
    - $\underline{x}^{\mathfrak{c}} : A \in \Omega$: The substitution has the form

$$\mathfrak{t} \prec \Phi \mid \Gamma \vdash (\kappa \mid \theta, a/\underline{x}^{\mathfrak{c}}, \theta') : \Psi \mid \Omega, \underline{x}^{\mathfrak{c}} : A, \Omega'$$

      where

$$\mathfrak{c} \mid \underline{\Gamma} \vdash a : A[\kappa \mid \theta]^{\mathsf{atc}}$$

      As in the last case, this term can be recoloured and then silently mark-weakened and weakened to

$$\mathfrak{s} \prec \Xi\{\downarrow(\mathfrak{t} \prec \Phi)^{v[\kappa]}\} \mid \Gamma^{v[\kappa]}, \Gamma'[\kappa \mid \theta] \vdash a^{\mathfrak{s}\leftrightarrow\mathfrak{c}} : (A[\kappa \mid \theta]^{\mathsf{atc}})^{\mathfrak{s}\leftrightarrow\mathfrak{c}}$$

      And $(A[\kappa \mid \theta]^{\mathsf{atc}})^{\mathfrak{s}\leftrightarrow\mathfrak{c}} \equiv A^{\mathfrak{s}\leftrightarrow\mathfrak{c}}[\kappa \mid \theta]^{\mathsf{ats}}$ as required.

- $\underline{x}^{\mathsf{c}} : A \in \Gamma'$: Then $\underline{x}^{\mathsf{c}} : A[\kappa \mid \theta]^{\mathsf{atc}} \in \Gamma'[\kappa \mid \theta]$, so we can reapply the rule.

- VAR-MARKED/DOM:
  - $\underline{x}^{\mathsf{c}} : A \in \Gamma^{w}$: Then also $\underline{x}^{\mathsf{c}} : A \in \Gamma^{w}$ in the conclusion, and so we can reapply VAR-MARKED.
  - $\underline{x}^{\mathsf{c}} : A \in \underline{\Omega}$: Essentially the same as in VAR-MARKED/COD, with cases depending on whether x is marked in Ω.
  - $\underline{x}^{\mathsf{c}} : A \in \Gamma'$: Then $\underline{x}^{\mathsf{c}} : A[\kappa \mid \theta]^{\mathsf{atc}} \in \Gamma'[\kappa \mid \theta]$, so we can reapply the rule.

- VAR-MARKED/MARKED: Same as previous, but with no mark-weakening of $\underline{\Gamma}$ needed.

Ordinary type formers: Immmediate by induction, in every case.
$\natural$-type:

- $\natural$-FORM: Immediate by SUBST/MARKED and reapplying the rule.
- $\natural$-INTRO: Immediate by SUBST/MARKED and reapplying the rule.
- $\natural$-ELIM: Immediate by induction (with the same instance of SUBST/?) and reapplying the rule.

$\otimes$-type:

- $\otimes$-FORM: The premises of the rule are in completely marked contexts, so in all cases apply SUBST/MARKED to give

$$\mathfrak{s} \mid \underline{\Gamma}, \underline{\Gamma'}[\kappa \mid \theta] \vdash A[\kappa \mid \theta]^{\mathsf{at}\mathfrak{s}} \ \mathsf{type}$$
$$\mathfrak{s} \mid \underline{\Gamma}, \underline{\Gamma'}[\kappa \mid \theta], \underline{x}^{\mathsf{t}} : A[\kappa \mid \theta]^{\mathsf{att}} \vdash B[\kappa \mid \theta]^{\mathsf{at}\mathfrak{s}} \ \mathsf{type}$$

  By Lemma B.1.4, these contexts are

$$\mathfrak{s} \mid \underline{\Gamma, \Gamma'[\kappa \mid \theta]} \vdash A[\kappa \mid \theta]^{\mathsf{at}\mathfrak{s}} \ \mathsf{type}$$
$$\mathfrak{s} \mid \underline{\Gamma, \Gamma'[\kappa \mid \theta]}, \underline{x}^{\mathfrak{s}} : A[\kappa \mid \theta]^{\mathsf{at}\mathfrak{s}} \vdash B[\kappa \mid \theta]^{\mathsf{at}\mathfrak{s}} \ \mathsf{type}$$

  and so we may reapply the rule, giving

$$\mathfrak{s} \prec \Xi \mid \Gamma, \Gamma'[\kappa \mid \theta] \vdash \textstyle\bigcirc\!\!\!\!\!\sum_{(\underline{x}:A[\kappa\mid\theta]^{\mathsf{at}\mathfrak{s}})}[\kappa \mid \theta]^{\mathsf{at}\mathfrak{s}} \ \mathsf{type}$$

- $\otimes$-INTRO: Suppose we have inputs

$$\mathfrak{s} \prec \Xi\{(\mathfrak{t} \prec \Phi, \Psi)^{?}\} \vdash s_L \boxtimes s_R \ \mathsf{split}$$
$$(\mathfrak{s} \prec \Xi\{(\mathfrak{t} \prec \Phi, \Psi)^{?}\})^{s_L} \mid (\Gamma, \Omega, \Gamma')^{s_L} \vdash a : A^{\ulcorner s_L \urcorner \leftrightarrow \mathfrak{t}}$$
$$(\mathfrak{s} \prec \Xi\{(\mathfrak{t} \prec \Phi, \Psi)^{?}\})^{s_R} \mid (\Gamma, \Omega, \Gamma')^{s_R} \vdash b : B[\underline{a}^{\mathfrak{t} \leftrightarrow \ulcorner s_L \urcorner} / \underline{x}^{\mathfrak{t}}]^{\ulcorner s_R \urcorner \leftrightarrow \mathfrak{t}}$$

Applying SUBST/DISPATCH to each,

$$\mathfrak{s} \prec \Xi\{\downarrow(\mathfrak{t} \prec \Phi)^{?[\kappa]}\} \vdash s_L[\kappa] \boxtimes s_R[\kappa] \ \mathsf{split}$$
$$(\mathfrak{s} \prec \Xi\{\downarrow(\mathfrak{t} \prec \Phi)^{?[\kappa]}\})^{s_L[\kappa]} \mid (\Gamma, \Gamma'[\kappa \mid \theta])^{s_L[\kappa]} \vdash a[\kappa \mid \theta]^{\mathsf{at}s_L} : A^{\ulcorner s_L \urcorner \leftrightarrow \mathfrak{s}}[\kappa \mid \theta]^{\mathsf{at}s_L}$$
$$(\mathfrak{s} \prec \Xi\{\downarrow(\mathfrak{t} \prec \Phi)^{?[\kappa]}\})^{s_R[\kappa]} \mid (\Gamma, \Gamma'[\kappa \mid \theta])^{s_R[\kappa]} \vdash b[\kappa \mid \theta]^{\mathsf{at}s_R} : B[\underline{a}^{\mathfrak{s} \leftrightarrow \ulcorner s_L \urcorner} / \underline{x}^{\mathfrak{s}}]^{\ulcorner s_R \urcorner \leftrightarrow \mathfrak{s}}[\kappa \mid \theta]^{\mathsf{at}s_R}$$

and so we just have to check that those types are correct. For A,

$$A^{\ulcorner s_L \urcorner \leftrightarrow \mathfrak{s}}[\kappa \mid \theta]^{\mathsf{at}s_L} \equiv (A[\kappa \mid \theta]^{\mathsf{at}\mathfrak{s}})^{\ulcorner s_L[\kappa] \urcorner \leftrightarrow \ulcorner \mathfrak{s}[\kappa] \urcorner}$$

by Lemma B.1.5. For B,

$$\begin{aligned}
&B[\underline{a}^{\mathfrak{s} \leftrightarrow \ulcorner s_L \urcorner} / \underline{x}^{\mathfrak{s}}]^{\ulcorner s_R \urcorner \leftrightarrow \mathfrak{s}}[\kappa \mid \theta]^{\mathsf{at}s_R} \\
&\equiv (B[\underline{a}^{\mathfrak{s} \leftrightarrow \ulcorner s_L \urcorner} / \underline{x}^{\mathfrak{s}}][\kappa \mid \theta]^{\mathsf{at}\mathfrak{s}})^{\ulcorner s_R[\kappa] \urcorner \leftrightarrow \ulcorner \mathfrak{s}[\kappa] \urcorner} && \text{(Lemma B.1.5)} \\
&\equiv (B[\kappa \mid \theta]^{\mathsf{at}\mathfrak{s}}[\underline{a}^{\mathfrak{s} \leftrightarrow \ulcorner s_L \urcorner}[\kappa \mid \theta]^{\mathsf{at}\mathfrak{s}} / \underline{x}^{\mathfrak{s}}])^{\ulcorner s_R[\kappa] \urcorner \leftrightarrow \ulcorner \mathfrak{s}[\kappa] \urcorner} && \text{(Lemma B.1.2)} \\
&\equiv (B[\kappa \mid \theta]^{\mathsf{at}\mathfrak{s}}[(\underline{a}[\kappa \mid \theta]^{\mathsf{at}s_L})^{\ulcorner \mathfrak{s}[\kappa] \urcorner \leftrightarrow \ulcorner s_L[\kappa] \urcorner} / \underline{x}^{\mathfrak{s}}])^{\ulcorner s_R[\kappa] \urcorner \leftrightarrow \ulcorner \mathfrak{s}[\kappa] \urcorner} && \text{(Lemma B.1.5)}
\end{aligned}$$

And so $a[\kappa \mid \theta]^{\mathsf{at}s_L}$ and $b[\kappa \mid \theta]^{\mathsf{at}s_R}$ have the correct types to re-apply $\otimes$-INTRO:

$$\mathfrak{s} \prec \Xi\{\downarrow(\mathfrak{t} \prec \Phi)^{?[\kappa]}\} \mid \Gamma, \Gamma'[\kappa \mid \theta] \vdash a[\kappa \mid \theta]^{\mathsf{at}s_L} \ {}_{s_L[\kappa]}\!\otimes_{s_R[\kappa]} b[\kappa \mid \theta]^{\mathsf{at}s_R} : \textcircled{Σ}_{(\underline{x}:A[\kappa|\theta]^{\mathsf{at}\mathfrak{s}})} B[\kappa \mid \theta]^{\mathsf{at}\mathfrak{s}}$$

Pattern matching: The desired rule is:

$$\text{SUBST}\ \frac{\mathfrak{t} \prec \Phi \mid \Gamma \vdash (\kappa \mid \theta) : \Psi \mid \Omega \qquad \begin{array}{l}\mathfrak{s} \prec \Xi\{\mathfrak{t} \prec \Phi, \Psi, \dots\} \ \mathsf{spot} \\ \mathfrak{s} \prec \Xi\{\mathfrak{t} \prec \Phi, \Psi, \dots\} \mid \Gamma, \Omega, \Gamma' \vdash \Psi_p \mid \Omega_p \vdash p : A \ \mathsf{pattern}\end{array}}{\mathfrak{s} \prec \Xi\{\downarrow\mathfrak{t} \prec \Phi, \dots\} \mid \Gamma, \Gamma'[\kappa \mid \theta] \vdash \Psi_p \mid \Omega_p[\kappa \mid \theta] \vdash p : A[\kappa \mid \theta]^{\mathsf{at}\mathfrak{s}} \ \mathsf{pattern}}$$

so that, in particular, the raw syntax of the pattern p is not changed.

On patterns, almost every case is immediate by induction. The interesting cases:

- $\natural$-PAT: We start with

$$\mathfrak{s} \prec \Xi\{(\mathfrak{t} \prec \Phi, \Psi)^?\} \mid \Gamma, \Omega, \Gamma' \vdash \Psi \mid \Omega_p \vdash p : A \ \mathsf{pattern}$$

and inductively

$$\mathfrak{s} \prec \Xi\{\downarrow(\mathfrak{t} \prec \Phi)^{?[\kappa]}\} \mid \Gamma, \Gamma'[\kappa \mid \theta] \vdash \Psi_p \mid \Omega_p[\kappa \mid \theta] \vdash p : A[\kappa \mid \theta]^{\mathsf{at}\mathfrak{s}} \ \mathsf{pattern}$$

Reapplying the rule gives

$$\mathfrak{s} \prec \Xi\{\downarrow(\mathfrak{t} \prec \Phi)^{?[\kappa]}\} \mid \Gamma, \Gamma'[\kappa \mid \theta] \vdash 1 \mid \underline{\Omega_p[\kappa \mid \theta]} \vdash \underline{p}^{\natural} : \natural\underline{A[\kappa \mid \theta]}^{\mathsf{at}\mathfrak{s}} \ \mathsf{pattern}$$

By Lemma B.1.4, the telescope and type here are equal to $\underline{\Omega_p}[\kappa \mid \theta]$ and $\underline{A}[\kappa \mid \theta]^{\mathsf{at}\mathfrak{s}}$ respectively, as was required.

- $\otimes$-PAT: The inputs are

$$\mathfrak{c}_L \mid \underline{\Gamma}, \underline{\Omega}, \underline{\Gamma'} \vdash \Psi_L \mid \Omega_L \vdash p_L : A^{\mathfrak{c}_L \leftrightarrow \mathfrak{s}} \ \textsf{pattern}$$
$$\mathfrak{c}_R \mid \underline{\Gamma}, \underline{\Omega}, \underline{\Gamma'}, \underline{\Omega_L} \vdash \Psi_R \mid \Omega_R \vdash p_R : B[\underline{p_L}^{\mathfrak{s} \leftrightarrow \mathfrak{c}_L} / \underline{x}]^{\mathfrak{c}_R \leftrightarrow \mathfrak{s}} \ \textsf{pattern}$$

  Applying SUBST/MARKED to each gives

$$\mathfrak{c}_L \mid \underline{\Gamma}, \underline{\Gamma'}[\kappa \mid \theta] \vdash \Psi_L \mid \Omega_L[\kappa \mid \theta] \vdash p_L : A^{\mathfrak{c}_L \leftrightarrow \mathfrak{s}}[\kappa \mid \theta] \ \textsf{pattern}$$
$$\mathfrak{c}_R \mid \underline{\Gamma}, \underline{\Gamma'}[\kappa \mid \theta], \underline{\Omega_L}[\kappa \mid \theta] \vdash \Psi_R \mid \Omega_R[\kappa \mid \theta] \vdash p_R : B[\underline{p_L}^{\mathfrak{s} \leftrightarrow \mathfrak{c}_L} / \underline{x}]^{\mathfrak{c}_R \leftrightarrow \mathfrak{s}}[\kappa \mid \theta] \ \textsf{pattern}$$

  The same calculation as in $\otimes$-INTRO shows that

$$A^{\mathfrak{c}_L \leftrightarrow \mathfrak{s}}[\kappa \mid \theta]^{\mathsf{at}\mathfrak{c}_L} \equiv (A[\kappa \mid \theta]^{\mathsf{at}\mathfrak{s}})^{\ulcorner \mathfrak{c}_L[\kappa] \urcorner \leftrightarrow \ulcorner \mathfrak{s}[\kappa] \urcorner}$$
$$B[\underline{a}^{\mathfrak{s} \leftrightarrow \mathfrak{c}_L} / \underline{x}^{\mathfrak{s}}]^{\mathfrak{c}_R \leftrightarrow \mathfrak{s}}[\kappa \mid \theta]^{\mathsf{at}\mathfrak{c}_R} \equiv (B[\kappa \mid \theta]^{\mathsf{at}\mathfrak{s}}[(\underline{p_L}[\kappa \mid \theta]^{\mathsf{at}\mathfrak{c}_L})^{\ulcorner \mathfrak{s}[\kappa] \urcorner \leftrightarrow \ulcorner \mathfrak{c}_L[\kappa] \urcorner} / \underline{x}^{\mathfrak{s}}])^{\ulcorner \mathfrak{c}_R[\kappa] \urcorner \leftrightarrow \ulcorner \mathfrak{s}[\kappa] \urcorner}$$

  The terms p_L, $\mathfrak{c}_L$ and $\mathfrak{c}_R$ are fresh for κ and θ, so these types are in turn equal to

$$A^{\mathfrak{c}_L \leftrightarrow \mathfrak{s}}[\kappa \mid \theta]^{\mathsf{at}\mathfrak{c}_L} \equiv (A[\kappa \mid \theta]^{\mathsf{at}\mathfrak{s}})^{\mathfrak{c}_L \leftrightarrow \ulcorner \mathfrak{s}[\kappa] \urcorner}$$
$$B[\underline{a}^{\mathfrak{s} \leftrightarrow \mathfrak{c}_L} / \underline{x}^{\mathfrak{s}}]^{\mathfrak{c}_R \leftrightarrow \mathfrak{s}}[\kappa \mid \theta]^{\mathsf{at}\mathfrak{c}_R} \equiv (B[\kappa \mid \theta]^{\mathsf{at}\mathfrak{s}}[\underline{p_L}^{\ulcorner \mathfrak{s}[\kappa] \urcorner \leftrightarrow \mathfrak{c}_L} / \underline{x}^{\mathfrak{s}}])^{\mathfrak{c}_R \leftrightarrow \ulcorner \mathfrak{s}[\kappa] \urcorner}$$

  which are of the correct form to reapply $\otimes$-PAT.

The match rule itself follows by induction, and that substitutions commute (Lemma B.1.2).
$\multimap$-types:

- $\multimap$-FORM: The inputs are

$$\mathfrak{r} \mid \underline{\Gamma}, \underline{\Omega}, \underline{\Gamma'} \vdash A \ \textsf{type}$$
$$\mathfrak{p} \prec (\mathfrak{s} \prec \Xi\{(\mathfrak{t} \prec \Phi, \Psi)^{?}\}) \otimes \mathfrak{r} \mid \Gamma, \Omega, \Gamma', x^{\mathfrak{r}} : A \vdash B \ \textsf{type}$$

  then by SUBST/MARKED on A and the same case on B,

$$\mathfrak{r} \mid \underline{\Gamma}, \underline{\Gamma'}[\kappa \mid \theta] \vdash A[\kappa \mid \theta]^{\mathsf{at}\mathfrak{r}} \ \textsf{type}$$
$$\mathfrak{p} \prec (\mathfrak{s} \prec \Xi\{\downarrow(\mathfrak{t} \prec \Phi)^{?[\kappa]}\}) \otimes \mathfrak{r} \mid \Gamma, \Gamma'[\kappa \mid \theta], x^{\mathfrak{r}} : A[\kappa \mid \theta]^{\mathsf{at}\mathfrak{r}} \vdash B[\kappa \mid \theta]^{\mathsf{at}\mathfrak{p}} \ \textsf{type}$$

  After rewriting $\underline{\Gamma'}[\kappa \mid \theta] \equiv \underline{\Gamma'[\kappa \mid \theta]}$ using Lemma B.1.4, these are of the right shape to reapply $\multimap$-FORM:

$$\mathfrak{s} \prec \Xi\{\downarrow(\mathfrak{t} \prec \Phi)^{?[\kappa]}\} \mid \Gamma, \Gamma'[\kappa \mid \theta] \vdash \textcircled{\Pi}_{(x^{\mathfrak{r}} : A[\kappa \mid \theta]^{\mathsf{at}\mathfrak{r}})}{}^{\mathfrak{p}} B[\kappa \mid \theta]^{\mathsf{at}\mathfrak{p}} \ \textsf{type}$$

- $\multimap$-INTRO: Essentially identical to the previous.

- $\multimap$-ELIM: The inputs are

$$\mathfrak{s} \prec \Xi\{(\mathfrak{t} \prec \Phi, \Psi)^{?}\} \vdash s_L \boxtimes s_R \ \textsf{split}$$
$$(\mathfrak{s} \prec \Xi\{(\mathfrak{t} \prec \Phi, \Psi)^{?}\})^{s_L} \mid (\Gamma, \Omega, \Gamma')^{s_L} \vdash f : \textcircled{\Pi}_{(x^{\mathfrak{r}} : A)}{}^{\mathfrak{p}} B$$
$$(\mathfrak{s} \prec \Xi\{(\mathfrak{t} \prec \Phi, \Psi)^{?}\})^{s_R} \mid (\Gamma, \Omega, \Gamma')^{s_R} \vdash a : A$$

then applying SUBST/DISPATCH,

$$\begin{array}{l}
\mathfrak{s} \prec \Xi\{\downarrow(\mathfrak{t} \prec \Phi)^{?[\kappa]}\} \vdash s_L[\kappa] \boxtimes s_R[\kappa] \ \mathsf{split} \\
(\mathfrak{s} \prec \Xi\{\downarrow(\mathfrak{t} \prec \Phi)^{?[\kappa]}\})^{s_L[\kappa]} \mid (\Gamma, \Gamma'[\kappa \mid \theta])^{s_L[\kappa]} \vdash f[\kappa \mid \theta]^{\mathsf{at}s_L} : \left(\mathop{\bigcirc\mkern-15mu\prod}_{(x^{\mathfrak{r}}:A)}{}^{\mathfrak{p}} B\right)[\kappa \mid \theta]^{\mathsf{at}s_L} \\
(\mathfrak{s} \prec \Xi\{\downarrow(\mathfrak{t} \prec \Phi)^{?[\kappa]}\})^{s_R[\kappa]} \mid (\Gamma, \Gamma'[\kappa \mid \theta])^{s_R[\kappa]} \vdash a[\kappa \mid \theta]^{\mathsf{at}s_R} : A[\kappa \mid \theta]^{\mathsf{at}s_R}
\end{array}$$

(recall that $\mathfrak{r} :\equiv \ulcorner s_R \urcorner$). By definition,

$$\left(\mathop{\bigcirc\mkern-15mu\prod}_{(x^{\mathfrak{r}}:A)}{}^{\mathfrak{p}} B\right)[\kappa \mid \theta]^{\mathsf{at}s_L} \equiv \mathop{\bigcirc\mkern-15mu\prod}_{(x^{\mathfrak{r}}:A[\kappa \mid \theta]^{\mathsf{at}s_R})}{}^{\mathfrak{p}} B[\kappa \mid \theta]^{\mathsf{at}\mathfrak{p}}$$

so reapplying the rule gives

$$\begin{array}{r}
\mathfrak{s} \prec \Xi\{\downarrow(\mathfrak{t} \prec \Phi)^{?[\kappa]}\} \mid \Gamma, \Gamma'[\kappa \mid \theta] \vdash f[\kappa \mid \theta]^{\mathsf{at}s_L}{}_{s_L[\kappa]}\langle a[\kappa \mid \theta]^{\mathsf{at}s_R}\rangle_{s_R[\kappa]} \\
: B[\kappa \mid \theta]^{\mathsf{at}\mathfrak{p}}[a[\kappa \mid \theta]^{\mathsf{at}s_R}/x^{\mathfrak{r}}]⟦\mathfrak{s} \prec s_L[\kappa] \boxtimes s_R[\kappa]/\mathfrak{p}⟧
\end{array}$$

This type is equal to

$$\begin{array}{ll}
B[\kappa \mid \theta]^{\mathsf{at}\mathfrak{p}}[a[\kappa \mid \theta]^{\mathsf{at}\mathfrak{r}}/x^{\mathfrak{r}}]⟦\mathfrak{s} \prec s_L[\kappa] \boxtimes s_R[\kappa]/\mathfrak{p}⟧ & \\
\equiv B[a/x^{\mathfrak{r}}][\kappa \mid \theta]^{\mathsf{at}\mathfrak{p}}⟦\mathfrak{s} \prec s_L[\kappa] \boxtimes s_R[\kappa]/\mathfrak{p}⟧ & \text{(Lemma B.1.2)} \\
\equiv B[a/x^{\mathfrak{r}}]⟦\mathfrak{s} \prec s_L \boxtimes s_R/\mathfrak{p}⟧[\kappa \mid \theta]^{\mathsf{at}\mathfrak{s}} & \text{(Lemma B.1.7)}
\end{array}$$

as required.

□

B.3.8 Merging

$$\textsc{merge}\ \dfrac{\begin{array}{c}\mathfrak{t} \prec \Phi \vdash t_L \boxtimes t_R \ \mathsf{split} \\ \mathfrak{t}' \prec \Phi^{t_L} \otimes \Phi^{t_R} \mid \Gamma^{s_L \otimes s_R} \vdash \mathcal{J}\end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash \mathcal{J}⟦\mathfrak{t} \prec t_L \boxtimes t_R/\mathfrak{t}'⟧}$$

This is very similar to mark-weakening: marked variables in the premise become unmarked in the conclusion.

And the generalised rules:

$$\textsc{merge}\ \dfrac{\begin{array}{c}\mathfrak{t} \prec \Phi \mid \Gamma \ \mathsf{ctx} \qquad (\mathfrak{s} \prec \Xi)\{\mathfrak{t}' \prec \Phi^{t_L} \otimes \Phi^{t_R}, \dots\} \mid \Gamma^{s_L \otimes s_R}; \Gamma' \ \mathsf{ext} \\ (\mathfrak{s} \prec \Xi)\{\mathfrak{t}' \prec \Phi^{t_L} \otimes \Phi^{t_R}, \dots\} \mid \Gamma^{s_L \otimes s_R}, \Gamma' \vdash \mathcal{J}\end{array}}{(\mathfrak{s} \prec \Xi)\{\downarrow\mathfrak{t} \prec \Phi, \dots\} \mid \Gamma, \Gamma'⟦\mathfrak{t} \prec t_L \boxtimes t_R/\mathfrak{t}'⟧ \vdash \mathcal{J}⟦\mathfrak{t} \prec t_L \boxtimes t_R/\mathfrak{t}'⟧}$$

$$\textsc{merge/partial}\ \dfrac{\begin{array}{c}\Phi^{t_L} \otimes \Phi^{t_R} \vdash w \ \mathsf{preslice} \\ \mathfrak{t} \prec \Phi \mid \Gamma \ \mathsf{ctx} \qquad (\mathfrak{s} \prec \Xi)\{(\Phi^{t_L} \otimes \Phi^{t_R})^{w}\} \mid (\Gamma^{s_L \otimes s_R})^{w}; \Gamma' \ \mathsf{ext} \\ (\mathfrak{s} \prec \Xi)\{(\Phi^{t_L} \otimes \Phi^{t_R})^{w}\} \mid (\Gamma^{s_L \otimes s_R})^{w}, \Gamma' \vdash \mathcal{J}\end{array}}{(\mathfrak{s} \prec \Xi)\{\downarrow\Phi^{w⟦\mathfrak{t} \prec t_L \boxtimes t_R/\mathfrak{t}'⟧}\} \mid \Gamma^{w⟦\mathfrak{t} \prec t_L \boxtimes t_R/\mathfrak{t}'⟧}, \Gamma'⟦\mathfrak{t} \prec t_L \boxtimes t_R/\mathfrak{t}'⟧ \vdash \mathcal{J}⟦\mathfrak{t} \prec t_L \boxtimes t_R/\mathfrak{t}'⟧}$$

The following equation will come up a few times:

Lemma B.3.22. *If A is well-typed in a marked context, then* $A^{\mathfrak{s} \leftrightarrow \mathfrak{c}}\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket \equiv A^{\mathfrak{s}^{\mathfrak{t} \leftrightarrow \mathfrak{t}'} \leftrightarrow \mathfrak{c}^{\mathfrak{t} \leftrightarrow \mathfrak{t}'}}$

Proof.

$$\begin{array}{ll}
A^{\mathfrak{s} \leftrightarrow \mathfrak{c}}\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket & \\
\equiv (A\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket)^{\mathfrak{s}^{\mathfrak{t} \leftrightarrow \mathfrak{t}'} \leftrightarrow \mathfrak{c}^{\mathfrak{t} \leftrightarrow \mathfrak{t}'}} & \text{(Lemma B.1.19)} \\
\equiv (\underline{A}\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket)^{\mathfrak{s}^{\mathfrak{t} \leftrightarrow \mathfrak{t}'} \leftrightarrow \mathfrak{c}^{\mathfrak{t} \leftrightarrow \mathfrak{t}'}} & \text{(Lemma B.3.17)} \\
\equiv (\underline{A}^{\mathfrak{t} \leftrightarrow \mathfrak{t}'})^{\mathfrak{s}^{\mathfrak{t} \leftrightarrow \mathfrak{t}'} \leftrightarrow \mathfrak{c}^{\mathfrak{t} \leftrightarrow \mathfrak{t}'}} & \text{(Lemma B.1.17)} \\
\equiv (A^{\mathfrak{t} \leftrightarrow \mathfrak{t}'})^{\mathfrak{s}^{\mathfrak{t} \leftrightarrow \mathfrak{t}'} \leftrightarrow \mathfrak{c}^{\mathfrak{t} \leftrightarrow \mathfrak{t}'}} & \text{(Lemma B.3.17)} \\
\equiv A^{\mathfrak{s}^{\mathfrak{t} \leftrightarrow \mathfrak{t}'} \leftrightarrow \mathfrak{c}^{\mathfrak{t} \leftrightarrow \mathfrak{t}'}} & (\mathfrak{t}' \text{ fresh for } A)
\end{array}$$

□

Recall that on the raw syntax of a term, this merging operation only affects the slice annotations on the each instance of the splitting rules; the underlying term is unchanged. This includes the marks on variable uses, which are also unchanged.

Lemma B.3.23. *Let* $(\mathfrak{s} \prec \Xi)\{(\mathfrak{t}' \prec \Phi^{t_L} \otimes \Phi^{t_R})^?\} \mid \Gamma^{s_L \otimes s_R}, \Gamma'\ \mathsf{ctx}$ *denote one of the above two situations, so one of*

$$\begin{array}{ll}
(\mathfrak{s} \prec \Xi)\{\mathfrak{t}' \prec \Phi^{t_L} \otimes \Phi^{t_R}, \dots\} \mid \Gamma^{s_L \otimes s_R}, \Gamma' & \mathsf{ctx} \\
(\mathfrak{s} \prec \Xi)\{(\Phi^{t_L} \otimes \Phi^{t_R})^w\} \qquad \mid (\Gamma^{s_L \otimes s_R})^w, \Gamma' & \mathsf{ctx}
\end{array}$$

and $\mathfrak{s} \prec \Xi\{\downarrow(\mathfrak{t} \prec \Phi)^{?\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket}\} \mid \Gamma, \Gamma'\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket\ \mathsf{ctx}$ *the respective conclusion contexts*

$$\begin{array}{lll}
(\mathfrak{s} \prec \Xi)\{\downarrow \mathfrak{t} \prec \Phi, \dots\} & \mid \Gamma, \Gamma'\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket & \mathsf{ctx} \\
(\mathfrak{s} \prec \Xi)\{\downarrow \Phi^{w\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket}\} & \mid \Gamma^{w\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket}, \Gamma'\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket & \mathsf{ctx}
\end{array}$$

In both cases we can merge 'under a slice':

MERGE/DISPATCH

$$\dfrac{\begin{array}{c}
(\mathfrak{s} \prec \Xi)\{(\mathfrak{t}' \prec \Phi^{t_L} \otimes \Phi^{t_R}, \dots)^?\} \vdash s\ \mathsf{slice} \\
\mathfrak{t} \prec \Phi \mid \Gamma\ \mathsf{ctx} \qquad (\mathfrak{s} \prec \Xi)\{(\mathfrak{t}' \prec \Phi^{t_L} \otimes \Phi^{t_R}, \dots)^?\} \mid \Gamma^{s_L \otimes s_R}; \Gamma'\ \mathsf{ext} \\
((\mathfrak{s} \prec \Xi)\{(\mathfrak{t}' \prec \Phi^{t_L} \otimes \Phi^{t_R}, \dots)^?\})^s \mid (\Gamma^{s_L \otimes s_R}, \Gamma')^s \vdash \mathcal{J}
\end{array}}{(\mathfrak{s} \prec \Xi\{\downarrow(\mathfrak{t} \prec \Phi)^{?\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket}\})^{s\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket} \mid (\Gamma, \Gamma'\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket)^{s\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket} \vdash \mathcal{J}\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket}$$

Proof.

- MERGE: There are a few cases:
  - The slice s contains the entire spot: Then

    $$(\mathfrak{s} \prec \Xi)^s\{\mathfrak{t}' \prec \Phi^{t_L} \otimes \Phi^{t_R}, \dots\} \mid \Gamma^{s_L \otimes s_R}, \Gamma'^t \vdash \mathcal{J}$$

    and we can apply SUBST to get

    $$(\mathfrak{s} \prec \Xi)^s\{\downarrow \mathfrak{t} \prec \Phi, \dots\} \mid \Gamma, \Gamma'^s\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket \vdash \mathcal{J}\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket$$

By Lemma B.2.2 we know

$$(\mathfrak{s} \prec \Xi)^{t}\{\downarrow\mathfrak{t} \prec \Phi, \dots\} \equiv ((\mathfrak{s} \prec \Xi)\{\downarrow\mathfrak{t} \prec \Phi, \dots\})^{s}$$

In this case $s \equiv s\llbracket\mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}'\rrbracket$, and Lemma B.1.24 gives

$$\Gamma'^{s}\llbracket\mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}'\rrbracket \equiv \Gamma'\llbracket\mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}'\rrbracket^{s}$$

so

$$((\mathfrak{s} \prec \Xi)\{\downarrow\mathfrak{t} \prec \Phi, \dots\})^{s} \mid (\Gamma, \Gamma'\llbracket\mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}'\rrbracket)^{s} \vdash \mathcal{J}\llbracket\mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}'\rrbracket$$

as required.

– If the slice s intersects $\Phi^{t_L} \otimes \Phi^{t_R}$: Then $\Phi^{t_L} \otimes \Phi^{t_R} \vdash s \cap (\Phi^{t_L} \otimes \Phi^{t_R})$ preslice and

$$(\mathfrak{s} \prec \Xi)^{s}\{(\Phi^{t_L} \otimes \Phi^{t_R})^{s \cap (\Phi^{t_L} \otimes \Phi^{t_R})}\} \text{ spot}$$

We can apply MERGE/PARTIAL to get

$$(\mathfrak{s} \prec \Xi)^{s}\{\downarrow\Phi^{(s\cap(\Phi^{t_L}\otimes\Phi^{t_R}))\llbracket\mathfrak{t}\prec t_L\boxtimes t_R/\mathfrak{t}'\rrbracket}\} \mid \Gamma^{(s\cap(\Phi^{t_L}\otimes\Phi^{t_R}))\llbracket\mathfrak{t}\prec t_L\boxtimes t_R/\mathfrak{t}'\rrbracket}, \Gamma'\llbracket\mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}'\rrbracket$$
$$\vdash \mathcal{J}\llbracket\mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}'\rrbracket$$

Now we just need that

$$(\mathfrak{s} \prec \Xi)^{s}\{\downarrow\Phi^{(s\cap(\Phi^{t_L}\otimes\Phi^{t_R}))\llbracket\mathfrak{t}\prec t_L\boxtimes t_R/\mathfrak{t}'\rrbracket}\} \equiv (\mathfrak{s} \prec \Xi)\{\downarrow(\mathfrak{t} \prec \Phi, \dots)\}^{s\llbracket\mathfrak{t}\prec t_L\boxtimes t_R/\mathfrak{t}'\rrbracket}$$

but this is Lemma B.2.35, and

$$\Gamma^{(s\cap(\Phi^{t_L}\otimes\Phi^{t_R}))\llbracket\mathfrak{t}\prec t_L\boxtimes t_R/\mathfrak{t}'\rrbracket} \equiv \Gamma^{s\llbracket\mathfrak{t}\prec t_L\boxtimes t_R/\mathfrak{t}'\rrbracket}$$

but this is Lemma B.3.12.

– The slice s does not intersect the spot: Then

$$(\mathfrak{s} \prec \Xi)^{s} \mid \underline{\Gamma}, \Gamma'^{s} \vdash \mathcal{J}$$

and already

$$\mathcal{J} \equiv \mathcal{J}\llbracket\mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}'\rrbracket$$

by Lemma B.1.12, and

$$(\mathfrak{s} \prec \Xi)^{s} \equiv (\mathfrak{s} \prec \Xi\{\downarrow\mathfrak{t} \prec \Phi, \dots\})^{s}$$

by Lemma B.2.1, so

$$((\mathfrak{s} \prec \Xi)\{\downarrow\mathfrak{t} \prec \Phi, \dots\})^{s} \mid (\Gamma, \Gamma'\llbracket(\mathfrak{t} \prec s_L \boxtimes s_R / \mathfrak{t}')\rrbracket)^{s} \vdash \mathcal{J}\llbracket\mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}'\rrbracket$$

as required.

• MERGE/PARTIAL: There are two cases:

– s intersects $(\Phi^{t_L} \otimes \Phi^{t_R})^w$: then $\Phi^{t_L} \otimes \Phi^{t_R} \vdash w \cap s$ preslice, and the input is

$$(\mathfrak{s} \prec \Xi)^s\{(\Phi^{t_L} \otimes \Phi^{t_R})^{w \cap s}\} \mid (\Gamma^{s_L \otimes s_R})^{w \cap s}, \Gamma'^s \vdash \mathcal{J}$$

Then we can reapply MERGE/PARTIAL, giving

$$(\mathfrak{s} \prec \Xi)^s\{\downarrow\Phi^{(w \cap s)[\![\mathfrak{t} \prec t_L \boxtimes t_R/\mathfrak{t}']\!]}\} \mid \Gamma^{(w \cap s)[\![\mathfrak{t} \prec t_L \boxtimes t_R/\mathfrak{t}']\!]}, \Gamma'^s[\![\mathfrak{t} \prec t_L \boxtimes t_R/\mathfrak{t}']\!] \vdash \mathcal{J}[\![\mathfrak{t} \prec t_L \boxtimes t_R/\mathfrak{t}']\!]$$

Again this is the correct palette by Lemma B.2.35, and the context is equal to

$$(\Gamma^{w[\![\mathfrak{t} \prec t_L \boxtimes t_R/\mathfrak{t}']\!]}, \Gamma'[\![\mathfrak{t} \prec t_L \boxtimes t_R/\mathfrak{t}']\!])^{s[\![\mathfrak{t} \prec t_L \boxtimes t_R/\mathfrak{t}']\!]}$$

by Lemma B.3.11 for Γ and Lemma B.1.24 for Γ'.

– s does not intersect the spot: Similar to the case for MERGE, because also

$$(\Gamma^{w[\![\mathfrak{t} \prec t_L \boxtimes t_R/\mathfrak{t}']\!]})^{s[\![\mathfrak{t} \prec t_L \boxtimes t_R/\mathfrak{t}']\!]} \equiv \underline{\Gamma}$$

in the conclusion.

□

Theorem B.3.24. *Merging on terms is admissible.*

$$\text{MERGE}\ \frac{\begin{array}{c}\mathfrak{t} \prec \Phi \vdash t_L \boxtimes t_R \text{ split} \\ \mathfrak{t}' \prec \Phi^{t_L} \otimes \Phi^{t_R} \mid \Gamma^{s_L \otimes s_R} \vdash a : A\end{array}}{\mathfrak{t} \prec \Phi \mid \Gamma \vdash a[\![\mathfrak{t} \prec t_L \boxtimes t_R/\mathfrak{t}']\!] : A[\![\mathfrak{t} \prec t_L \boxtimes t_R/\mathfrak{t}']\!]}$$

Proof. Many of the cases rely on Lemma B.2.38 to know that the top colour of the conclusion is $\mathfrak{s}^{\mathfrak{t} \leftrightarrow \mathfrak{t}'}$.

As usual, each variable rule has cases for the position of the variable in the context. Variables:

- VAR:
  – $x^{\mathfrak{s}} : A \in \Gamma^{t_L \otimes t_R}$: Cannot occur, because $\mathfrak{s}$ does not occur in $t_L \otimes t_R$ (even in the case that $\mathfrak{s} \equiv \mathfrak{t}'$, i.e., the spot used is the trivial spot).
  – $x^{\mathfrak{s}} : A \in \Gamma'$: Then the derivation is

$$\frac{\mathfrak{s} \prec \Xi\{\mathfrak{t}' \prec \Phi^{t_L} \otimes \Phi^{t_R}, \dots\} \mid \Gamma^{t_L \otimes t_R}, \Gamma'_1 \vdash A \text{ type}}{\mathfrak{s} \prec \Xi\{\mathfrak{t}' \prec \Phi^{t_L} \otimes \Phi^{t_R}, \dots\} \mid \Gamma^{t_L \otimes t_R}, \Gamma'_1, x^{\mathfrak{s}} : A, \Gamma'_2 \vdash x : A}$$

  In the conclusion we have $x^{\mathfrak{s}^{\mathfrak{t} \leftrightarrow \mathfrak{t}'}} : A[\![\mathfrak{t} \prec t_L \boxtimes t_R/\mathfrak{t}']\!] \in \Gamma'[\![\mathfrak{t} \prec t_L \boxtimes t_R/\mathfrak{t}']\!]$ and $\mathfrak{s}[\![\mathfrak{t} \prec t_L \boxtimes t_R/\mathfrak{t}']\!]$ is the top colour, so we can reapply VAR, giving

$$\mathfrak{s} \prec \Xi\{\downarrow\mathfrak{t} \prec \Phi, \dots\} \mid \Gamma, (\Gamma'_1, x^{\mathfrak{s}} : A, \Gamma'_2)[\![\mathfrak{t} \prec t_L \boxtimes t_R/\mathfrak{t}']\!] \vdash x : A[\![\mathfrak{t} \prec t_L \boxtimes t_R/\mathfrak{t}']\!]$$

- VAR/PARTIAL:
  – $x^{\mathfrak{s}} : A \in (\Gamma^{t_L \otimes t_R})^w$: Then $\mathfrak{s} \in w$, and so also $\mathfrak{s} \in w[\![\mathfrak{t} \prec t_L \boxtimes t_R/\mathfrak{t}']\!]$ by Lemma B.2.37. So $x^{\mathfrak{s}} : A \in \Gamma^{w[\![\mathfrak{t} \prec t_L \boxtimes t_R/\mathfrak{t}']\!]}$ in the conclusion, and we can reapply VAR.

  - $x^{\mathfrak{s}} : A \in \Gamma'$: Then $x^{\mathfrak{s}} : A\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket \in \Gamma'\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket$ and so we can reapply VAR.

- VAR-ROUNDTRIP:
  - $x^{\mathfrak{c}} : A \in \Gamma^{t_L \otimes t_R}$: The derivation is

    Also $x^{\mathfrak{c}} : A \in \Gamma$ in the conclusion, and so we can reuse VAR-ROUNDTRIP to form

$$\mathfrak{s} \prec \Xi\{\downarrow\mathfrak{t} \prec \Phi, \dots\} \mid \Gamma_1, x^{\mathfrak{c}} : A, \Gamma_2, \Gamma'\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket \vdash \underline{x} : \underline{A}^{\mathfrak{s}^{\mathfrak{t} \leftrightarrow \mathfrak{t}'} \leftrightarrow \mathfrak{c}}$$

    and this type is $\underline{A}^{\mathfrak{s}^{\mathfrak{t} \leftrightarrow \mathfrak{t}'} \leftrightarrow \mathfrak{c}}$ by Lemma B.3.22.
  - $x^{\mathfrak{c}} : A \in \Gamma'$: The derivation is

$$\frac{(\mathfrak{s} \prec \Xi\{\mathfrak{t}' \prec \Phi^{t_L} \otimes \Phi^{t_R}, \dots\})^{\mathfrak{c}} \mid (\Gamma^{t_L \otimes t_R}, \Gamma_1)^{\mathfrak{c}} \vdash A \ \mathsf{type}}{\mathfrak{s} \prec \Xi\{\mathfrak{t}' \prec \Phi^{t_L} \otimes \Phi^{t_R}, \dots\} \mid \Gamma^{t_L \otimes t_R}, \Gamma'_1, x^{\mathfrak{c}} : A, \Gamma'_2 \vdash \underline{x} : \underline{A}^{\mathfrak{s} \leftrightarrow \mathfrak{c}}}$$

    By the definition of merging on contexts, $x^{\mathfrak{c}^{\mathfrak{t} \leftrightarrow \mathfrak{t}'}} : A\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket \in \Gamma'\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket$ in the conclusion, and so we can use VAR-ROUNDTRIP to form

$$\begin{gathered}\mathfrak{s} \prec \Xi\{\downarrow\mathfrak{t} \prec \Phi, \dots\} \mid \Gamma, \Gamma'_1\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket, x^{\mathfrak{c}^{\mathfrak{t} \leftrightarrow \mathfrak{t}'}} : A\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket, \Gamma'_2\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket \\ \vdash \underline{x} : \underline{A\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket}^{\mathfrak{s}^{\mathfrak{t} \leftrightarrow \mathfrak{t}'} \leftrightarrow \mathfrak{c}^{\mathfrak{t} \leftrightarrow \mathfrak{t}'}}\end{gathered}$$

    and this type is

$$\underline{A\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket}^{\mathfrak{s}^{\mathfrak{t} \leftrightarrow \mathfrak{t}'} \leftrightarrow \mathfrak{c}^{\mathfrak{t} \leftrightarrow \mathfrak{t}'}} \equiv (\underline{A}\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket)^{\mathfrak{s}^{\mathfrak{t} \leftrightarrow \mathfrak{t}'} \leftrightarrow \mathfrak{c}^{\mathfrak{t} \leftrightarrow \mathfrak{t}'}} \equiv \underline{A}^{\mathfrak{s} \leftrightarrow \mathfrak{c}}\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket$$

    by Lemma B.1.20 followed by Lemma B.1.19.

- VAR-ROUNDTRIP/PARTIAL:
  - $x^{\mathfrak{c}} : A \in (\Gamma^{t_L \otimes t_R})^{w}$: The derivation is

$$\frac{(\mathfrak{s} \prec \Xi\{(\Phi^{t_L} \otimes \Phi^{t_R})^{w}\})^{\mathfrak{c}} \mid ((\Gamma_1{}^{t_L \otimes t_R})^{w})^{\mathfrak{c}} \vdash A \ \mathsf{type}}{\mathfrak{s} \prec \Xi\{(\Phi^{t_L} \otimes \Phi^{t_R})^{w}\} \mid (\Gamma_1{}^{t_L \otimes t_R})^{w}, x^{\mathfrak{c}} : A, (\Gamma_2{}^{t_L \otimes t_R})^{w}, \Gamma' \vdash x : A}$$

    Then $\mathfrak{c} \in w$, and so also $\mathfrak{c} \in w\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket$ by Lemma B.2.37. Therefore the conclusion context $\Gamma^{w\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket}$ also contains $x^{\mathfrak{c}} : A$. And so reapplying VAR-ROUNDTRIP gives

$$\begin{gathered}\mathfrak{s} \prec \Xi\{\downarrow\Phi^{w\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket}\} \mid \Gamma_1{}^{w\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket}, x^{\mathfrak{c}} : A, \Gamma_2{}^{w\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket}, \Gamma'\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket \\ \vdash \underline{x} : \underline{A}^{\mathfrak{s}^{\mathfrak{t} \leftrightarrow \mathfrak{t}'} \leftrightarrow \mathfrak{c}}\end{gathered}$$

    This type is equal to $\underline{A}^{\mathfrak{s} \leftrightarrow \mathfrak{c}}\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket$ by Lemma B.3.22.
  - $x^{\mathfrak{c}} : A \in \Gamma'$: The same reasoning as in the ordinary VAR-ROUNDTRIP case applies, because the action of this rule on Γ' is the same.

- VAR-MARKED: As in mark-weakening, we must distinguish additional cases, for whether $x^{\mathfrak{c}}$ is marked in the conclusion.

  - $\underline{x}^{\mathfrak{c}} \in \Gamma^{t_L \otimes t_R}$ and x unmarked in Γ: The derivation to consider is

$$\frac{\mathfrak{c} \mid \underline{{\Gamma_1}^{t_L \otimes t_R}} \vdash \underline{A} \text{ type}}{\mathfrak{s} \prec \Xi\{\mathfrak{t}' \prec \Phi^{t_L} \otimes \Phi^{t_R}, \dots\} \mid {\Gamma_1}^{t_L \otimes t_R}, \underline{x}^{\mathfrak{c}} : \underline{A}, {\Gamma_2}^{t_L \otimes t_R}, \Gamma' \vdash \underline{x} : \underline{A}^{\mathfrak{s} \leftrightarrow \mathfrak{c}}}$$

    Because the context $\mathfrak{t} \prec \Phi \mid \Gamma_1, x^{\mathfrak{c}} : A, \Gamma_2$ ctx is assumed to be well-formed, we know

$$\Phi^{\mathfrak{c}} \mid {\Gamma_1}^{\mathfrak{c}} \vdash A \text{ type}$$

    Because $\mathfrak{c} \in (\mathfrak{t} \prec \Phi)$, we also know $(\mathfrak{s} \prec \Xi\{\downarrow\mathfrak{t} \prec \Phi, \dots\})^{\mathfrak{c}}$ is a cartesian weakening of $\Phi^{\mathfrak{c}}$ by Lemma B.2.36, and so we can weaken A to

$$(\mathfrak{s} \prec \Xi\{\downarrow\mathfrak{t} \prec \Phi, \dots\})^{\mathfrak{c}} \mid {\Gamma_1}^{\mathfrak{c}} \vdash A \text{ type}$$

    and apply VAR-ROUNDTRIP giving

$$\mathfrak{s} \prec \Xi\{\downarrow\mathfrak{t} \prec \Phi, \dots\} \mid \Gamma_1, x^{\mathfrak{c}} : A, \Gamma_2, \Gamma'[\![\mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}']\!] \vdash \underline{x} : (A^{\mathsf{m}\Xi|\Gamma_1})^{\mathfrak{s} \leftrightarrow \mathfrak{c}}$$

    And $(A^{\mathsf{m}\Xi|\Gamma_1})^{\mathfrak{s} \leftrightarrow \mathfrak{c}} \equiv (A^{\mathsf{m}\Phi|\Gamma_1})^{\mathfrak{s} \leftrightarrow \mathfrak{c}}$ because A only contains colours from Φ.

  - $\underline{x}^{\mathfrak{c}} \in \Gamma^{t_L \otimes t_R}$ and x marked in Γ: Here the derivation to consider is

$$\frac{\mathfrak{c} \mid \underline{{\Gamma_1}^{t_L \otimes t_R}} \vdash A \text{ type}}{\mathfrak{s} \prec \Xi\{\mathfrak{t}' \prec \Phi^{t_L} \otimes \Phi^{t_R}, \dots\} \mid (\Gamma_1, \underline{x}^{\mathfrak{c}} : A, \Gamma_2)^{t_L \otimes t_R}, \Gamma' \vdash \underline{x} : A^{\mathfrak{s} \leftrightarrow \mathfrak{c}}}$$

    By Lemma B.1.22, also $\mathfrak{c} \mid \underline{\Gamma_1} \vdash A$ type, and so

$$\mathfrak{s} \prec \Xi\{\downarrow\mathfrak{t} \prec \Phi, \dots\} \mid \Gamma_1, \underline{x}^{\mathfrak{c}} : A, \Gamma_2, \Gamma'[\![\mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}']\!] \vdash \underline{x} : A^{\mathfrak{s}^{\mathfrak{t} \leftrightarrow \mathfrak{t}'} \leftrightarrow \mathfrak{c}}$$

    is well formed. This type is equal to $A^{\mathfrak{s} \leftrightarrow \mathfrak{c}}[\![\mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}']\!]$ by Lemma B.3.22, and that $\mathfrak{c} \not\equiv \mathfrak{t}'$.

  - $\underline{x}^{\mathfrak{c}} \in \Gamma'$: The derivation is

$$\frac{\mathfrak{c} \mid \underline{\Gamma^{t_L \otimes t_R}, \Gamma_1'} \vdash A \text{ type}}{\mathfrak{s} \prec \Xi\{\mathfrak{t}' \prec \Phi^{t_L} \otimes \Phi^{t_R}, \dots\} \mid \Gamma^{t_L \otimes t_R}, \Gamma_1', \underline{x}^{\mathfrak{c}} : A, \Gamma_2' \vdash \underline{x} : A^{\mathfrak{s} \leftrightarrow \mathfrak{c}}}$$

    and by Lemma B.1.22, $\underline{\Gamma^{t_L \otimes t_R}} \equiv \underline{\Gamma}$ and so

$$\mathfrak{c} \mid \underline{\Gamma, \Gamma_1'} \vdash A \text{ type}$$

    Applying Lemma B.3.25,

$$\mathfrak{c} \mid \underline{\Gamma, \Gamma_1'[\![\mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}']\!]} \vdash A \text{ type}$$

    So VAR-MARKED gives

$$\mathfrak{s} \prec \Xi \mid \Gamma, \Gamma_1'[\![\mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}']\!], \underline{x}^{\mathfrak{c}} : A, \Gamma_2'[\![\mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}']\!] \vdash \underline{x} : A^{\mathfrak{s}^{\mathfrak{t} \leftrightarrow \mathfrak{t}'} \leftrightarrow \mathfrak{c}}$$

    This type is equal to $A^{\mathfrak{s} \leftrightarrow \mathfrak{c}}[\![\mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}']\!]$ again by Lemma B.3.22.

- VAR-MARKED/PARTIAL:
  - $\underline{x}^{\mathfrak{c}} : A \in (\Gamma^{t_L \otimes t_R})^w$ and x unmarked in $\Gamma^{w[\![\mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}']\!]}$: The derivation to consider is

$$\frac{\mathfrak{c} \mid \underline{({\Gamma_1}^{t_L \otimes t_R})^w} \vdash \underline{A} \text{ type}}{\mathfrak{s} \prec \Xi\{(\Phi^{t_L} \otimes \Phi^{t_R})^w\} \mid ({\Gamma_1}^{t_L \otimes t_R})^w, \underline{x}^{\mathfrak{c}} : \underline{A}, ({\Gamma_2}^{t_L \otimes t_R})^w, \Gamma' \vdash \underline{x} : \underline{A}^{\mathfrak{s} \leftrightarrow \mathfrak{c}}}$$

    By the definition of context filtering, x must also be unmarked in Γ. And so in as in the VAR-MARKED case, the context $\mathfrak{t} \prec \Phi \mid \Gamma_1, x^{\mathfrak{c}} : A, \Gamma_2$ ctx is assumed to be well-formed, so

$$\Phi^{\mathfrak{c}} \mid {\Gamma_1}^{\mathfrak{c}} \vdash A \text{ type}$$

    For x to be unmarked in $\Gamma^{w[\![\mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}']\!]}$, it must be the case that $\mathfrak{c} \in w[\![\mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}']\!]$. By Lemma B.2.36 we know $(\mathfrak{s} \prec \Xi\{\downarrow\Phi^{w[\![\mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}']\!]}\})^{\mathfrak{c}}$ is a cartesian weakening of $(\Phi^{w[\![\mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}']\!]})^{\mathfrak{c}}$ which is itself equal to $\Phi^{\mathfrak{c}}$ by Lemma B.2.9. Also

$${\Gamma_1}^{\mathfrak{c}} \equiv ({\Gamma_1}^{w[\![\mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}']\!]})^{\mathfrak{c}}$$

    because $\mathfrak{c}$ is in $w[\![\mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}']\!]$, so we have

$$(\mathfrak{s} \prec \Xi\{\downarrow\Phi^{w[\![\mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}']\!]}\})^{\mathfrak{c}} \mid ({\Gamma_1}^{w[\![\mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}']\!]})^{\mathfrak{c}} \vdash A \text{ type}$$

    Applying VAR-ROUNDTRIP gives

$$\mathfrak{s} \prec \Xi\{\downarrow\Phi^{w[\![\mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}']\!]}\} \mid {\Gamma_1}^{w[\![\mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}']\!]}, x^{\mathfrak{c}} : A, {\Gamma_2}^{w[\![\mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}']\!]}, \Gamma'[\![\mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}']\!] \vdash \underline{x} : \underline{A}^{\mathfrak{s} \leftrightarrow \mathfrak{c}}$$

    the type of which is equal to $\underline{A}^{\mathfrak{s} \leftrightarrow \mathfrak{c}}[\![\mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}']\!]$ by Lemma B.3.22, and that $\mathfrak{s} \not\equiv \mathfrak{t}'$.
  - $\underline{x}^{\mathfrak{c}} : A \in (\Gamma^{t_L \otimes t_R})^w$ and $\underline{x}$ marked in $\Gamma^{w[\![\mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}']\!]}$: The derivation to consider is

$$\frac{\mathfrak{c} \mid \underline{({\Gamma_1}^{t_L \otimes t_R})^w} \vdash A \text{ type}}{\mathfrak{s} \prec \Xi\{(\Phi^{t_L} \otimes \Phi^{t_R})^w\} \mid ({\Gamma_1}^{t_L \otimes t_R})^w, \underline{x}^{\mathfrak{c}} : A, ({\Gamma_2}^{t_L \otimes t_R})^w, \Gamma' \vdash \underline{x} : A^{\mathfrak{s} \leftrightarrow \mathfrak{c}}}$$

    We know

$$\underline{({\Gamma_1}^{t_L \otimes t_R})^w} \equiv \underline{{\Gamma_1}^{t_L \otimes t_R}} \equiv \underline{\Gamma_1} \equiv \underline{{\Gamma_1}^{w[\![\mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}']\!]}}$$

    by applying Lemma B.1.22 thrice, so

$$\mathfrak{c} \mid \underline{{\Gamma_1}^{w[\![\mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}']\!]}} \vdash A \text{ type}$$

    and so applying VAR-MARKED,

$$\begin{gathered}\mathfrak{s} \prec \Xi\{\downarrow\Phi^{w[\![\mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}']\!]}\} \mid {\Gamma_1}^{w[\![\mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}']\!]}, \underline{x}^{\mathfrak{c}} : A, {\Gamma_2}^{w[\![\mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}']\!]}, \Gamma'[\![\mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}']\!] \\ \vdash \underline{x} : A^{\mathfrak{s} \leftrightarrow \mathfrak{c}}\end{gathered}$$

– $\underline{x}^{\mathfrak{c}} : A \in \Gamma'$: The derivation is

$$\frac{\mathfrak{c} \mid \underline{(\Gamma^{t_L \otimes t_R})^{w}, \Gamma'_1} \vdash A \ \mathsf{type}}{\mathfrak{s} \prec \Xi\{\mathfrak{t}' \prec \Phi^{t_L} \otimes \Phi^{t_R}, \dots\} \mid \Gamma^{t_L \otimes t_R}, \Gamma'_1, \underline{x}^{\mathfrak{c}} : A, \Gamma'_2 \vdash \underline{x} : A^{\mathfrak{s} \leftrightarrow \mathfrak{c}}}$$

As in the previous case,

$$\underline{(\Gamma^{t_L \otimes t_R})^{w}} \equiv \underline{\Gamma^{w\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket}}$$

via Lemma B.1.22, so

$$\mathfrak{c} \mid \underline{\Gamma^{w\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket}, \Gamma'_1} \vdash A \ \mathsf{type}$$

Applying Lemma B.3.25, also

$$\mathfrak{c} \mid \underline{\Gamma^{w\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket}, \Gamma'_1\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket} \vdash A \ \mathsf{type}$$

So VAR-MARKED gives

$$\begin{aligned}\mathfrak{s} \prec \Xi\{\downarrow\Phi^{w\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket}\} \mid \Gamma^{w\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket}, \Gamma'_1\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket, \underline{x}^{\mathfrak{c}} : A, \Gamma'_2\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket \\ \vdash \underline{x} : A^{\mathfrak{s} \leftrightarrow \mathfrak{c}}\end{aligned}$$

- Π-FORM: The inputs are

$$\begin{aligned}(\mathfrak{s} \prec \Xi)\{\mathfrak{t}' \prec \Phi^{t_L} \otimes \Phi^{t_R}, \dots\} \mid \Gamma^{t_L \otimes t_R}, \Gamma' \vdash A \ \mathsf{type} \\ (\mathfrak{s} \prec \Xi)\{\mathfrak{t}' \prec \Phi^{t_L} \otimes \Phi^{t_R}, \dots\} \mid \underline{\Gamma}, \Gamma', x^{\mathfrak{s}} : A \vdash B \ \mathsf{type}\end{aligned}$$

and inductively

$$\begin{aligned}(\mathfrak{s} \prec \Xi\{\downarrow\mathfrak{t} \prec \Phi, \dots\}) \mid \Gamma, \Gamma'\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket \\ \vdash A\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket \ \mathsf{type} \\ (\mathfrak{s} \prec \Xi\{\downarrow\mathfrak{t} \prec \Phi, \dots\}) \mid \Gamma, \Gamma'\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket, x^{\mathfrak{s}^{\mathfrak{t} \leftrightarrow \mathfrak{t}'}} : A\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket \\ \vdash B\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket \ \mathsf{type}\end{aligned}$$

and we can re-apply Π-FORM.

- ♮-FORM: The input type is

$$\mathfrak{s} \mid \underline{\Gamma^{t_L \otimes t_R}, \Gamma'} \vdash A \ \mathsf{type}$$

and $A\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket \equiv A^{\mathfrak{t} \leftrightarrow \mathfrak{t}'}$ by Lemma B.1.17.

Idempotence gives that $\underline{\Gamma} \equiv \underline{\Gamma^{t_L \otimes t_R}}$ and by Lemma B.1.20,

$$\underline{\Gamma'\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket} \equiv \underline{\Gamma'}\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket$$

Therefore

$$\mathfrak{s}^{\mathfrak{t} \leftrightarrow \mathfrak{t}'} \mid \underline{\Gamma, \Gamma'\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket} \vdash A\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket \ \mathsf{type}$$

and we can reapply the rule.

- $\otimes$-INTRO: First suppose we are not in the special case where one of the slices s_L or s_R used in the term is exactly the top colour of t_L or t_R. Then we have inputs

$$\mathfrak{s} \prec \Xi\{(\mathfrak{t}' \prec \Phi^{t_L} \otimes \Phi^{t_R}, \dots)^?\} \vdash s_L \boxtimes s_R \text{ split}$$
$$(\mathfrak{s} \prec \Xi\{(\mathfrak{t}' \prec \Phi^{t_L} \otimes \Phi^{t_R}, \dots)^?\})^{s_L} \mid (\Gamma^{t_L \otimes t_R}, \Gamma')^{s_L} \vdash a : A^{\ulcorner s_L \urcorner \leftrightarrow \mathfrak{s}}$$
$$(\mathfrak{s} \prec \Xi\{(\mathfrak{t}' \prec \Phi^{t_L} \otimes \Phi^{t_R}, \dots)^?\})^{s_R} \mid (\Gamma^{t_L \otimes t_R}, \Gamma')^{s_R} \vdash b : B[\underline{a}^{\mathfrak{s} \leftrightarrow \ulcorner s_L \urcorner} / \underline{x}^{\mathfrak{s}}]^{\ulcorner s_R \urcorner \leftrightarrow \mathfrak{s}}$$

Applying MERGE/DISPATCH to each,

$$\mathfrak{s} \prec \Xi\{\downarrow(\mathfrak{t} \prec \Phi)^{?\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket}\} \vdash s_L\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket \boxtimes s_R\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket \text{ split}$$
$$\begin{aligned}
&(\mathfrak{s} \prec \Xi\{\downarrow(\mathfrak{t} \prec \Phi)^{?\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket}\})^{s_L\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket} \\
&\mid (\Gamma, \Gamma'\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket)^{s_L\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket} \\
&\vdash a\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket : A^{\ulcorner s_L \urcorner \leftrightarrow \mathfrak{s}}\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket \\
&(\mathfrak{s} \prec \Xi\{\downarrow(\mathfrak{t} \prec \Phi)^{?\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket}\})^{s_R\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket} \\
&\mid (\Gamma, \Gamma'\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket)^{s_R\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket} \\
&\vdash b\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket : B[\underline{a}^{\mathfrak{s} \leftrightarrow \ulcorner s_L \urcorner} / \underline{x}^{\mathfrak{s}}]^{\ulcorner s_R \urcorner \leftrightarrow \mathfrak{s}}\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket
\end{aligned}$$

and so we just have to check that those types are correct. For A,

$$\begin{aligned}
&A^{\ulcorner s_L \urcorner \leftrightarrow \mathfrak{s}}\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket \\
&\equiv (A\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket)^{\ulcorner s_L \urcorner^{\mathfrak{t} \leftrightarrow \mathfrak{t}'} \leftrightarrow \mathfrak{s}^{\mathfrak{t} \leftrightarrow \mathfrak{t}'}} \\
&\equiv (A\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket)^{\ulcorner s_L\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket \urcorner \leftrightarrow \ulcorner \mathfrak{s}\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket \urcorner}
\end{aligned}$$

by Lemma B.1.19. For B, and letting $\llbracket \dots \rrbracket$ stand in for $\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket$,

$$\begin{aligned}
&B[\underline{a}^{\mathfrak{s} \leftrightarrow \ulcorner s_L \urcorner} / \underline{x}^{\mathfrak{s}}]^{\ulcorner s_R \urcorner \leftrightarrow \mathfrak{s}}\llbracket \dots \rrbracket \\
&\equiv (B[\underline{a}^{\mathfrak{s} \leftrightarrow \ulcorner s_L \urcorner} / \underline{x}^{\mathfrak{s}}]\llbracket \dots \rrbracket)^{\ulcorner s_R\llbracket \dots \rrbracket \urcorner \leftrightarrow \ulcorner \mathfrak{s}\llbracket \dots \rrbracket \urcorner} \\
&\quad \text{(Lemma B.1.5)} \\
&\equiv (B\llbracket \dots \rrbracket[\underline{a}^{\mathfrak{s} \leftrightarrow \ulcorner s_L \urcorner}\llbracket \dots \rrbracket / \underline{x}^{\mathfrak{s}}])^{\ulcorner s_R\llbracket \dots \rrbracket \urcorner \leftrightarrow \ulcorner \mathfrak{s}\llbracket \dots \rrbracket \urcorner} \\
&\quad \text{(Lemma B.1.2)} \\
&\equiv (B\llbracket \dots \rrbracket[(\underline{a}\llbracket \dots \rrbracket)^{\ulcorner \mathfrak{s}\llbracket \dots \rrbracket \urcorner \leftrightarrow \ulcorner s_L\llbracket \dots \rrbracket \urcorner} / \underline{x}^{\mathfrak{s}}])^{\ulcorner s_R\llbracket \dots \rrbracket \urcorner \leftrightarrow \ulcorner \mathfrak{s}\llbracket \dots \rrbracket \urcorner} \\
&\quad \text{(Lemma B.1.5)}
\end{aligned}$$

And so $a\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket$ and $b\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket$ have the correct types to re-apply $\otimes$-INTRO:

$$\begin{aligned}
&\mathfrak{s} \prec \Xi\{\downarrow(\mathfrak{t} \prec \Phi)^{?\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket}\} \\
&\mid \Gamma, \Gamma'\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket \\
&\vdash a\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket \;{}_{s_L\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket}\!\otimes_{s_R\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket}\; b\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket \\
&: \textcircled{\Sigma}_{(\underline{x} : A\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket)} B\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket
\end{aligned}$$

Now consider the special case that $s_L \equiv \ulcorner t_L \urcorner$.

$$\mathfrak{s} \prec \Xi\{(\mathfrak{t}' \prec \Phi^{t_L} \otimes \Phi^{t_R}, \dots)^?\} \vdash \ulcorner t_L \urcorner \boxtimes s_R \text{ split}$$
$$(\mathfrak{s} \prec \Xi\{(\mathfrak{t}' \prec \Phi^{t_L} \otimes \Phi^{t_R}, \dots)^?\})^{\ulcorner t_L \urcorner} \mid (\Gamma^{t_L \otimes t_R}, \Gamma')^{\ulcorner t_L \urcorner} \vdash a : A^{\ulcorner t_L \urcorner \leftrightarrow \mathfrak{s}}$$
$$(\mathfrak{s} \prec \Xi\{(\mathfrak{t}' \prec \Phi^{t_L} \otimes \Phi^{t_R}, \dots)^?\})^{s_R} \mid (\Gamma^{t_L \otimes t_R}, \Gamma')^{s_R} \vdash b : B[\underline{a}^{\mathfrak{s} \leftrightarrow \ulcorner t_L \urcorner} / \underline{x}^{\mathfrak{s}}]^{\ulcorner s_R \urcorner \leftrightarrow \mathfrak{s}}$$

But now

$$\Phi^{t_L} \mid \Gamma^{t_L}, \underline{\Gamma'} \vdash a : A^{\ulcorner t_L \urcorner \leftrightarrow \mathfrak{s}}$$

because $(\mathfrak{s} \prec \Xi\{(\mathfrak{t}' \prec \Phi^{t_L} \otimes \Phi^{t_R}, \dots)^?\})^{\ulcorner t_L \urcorner} \equiv \Phi^{t_L}$, and Γ' does not use colours from Φ. This term is already of the required form to re-apply the $\otimes$ rule together with $b\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket$, because

$$\Phi^{t_L} \equiv (\mathfrak{s} \prec \Xi\{\downarrow(\mathfrak{t} \prec \Phi)^{?\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket}\})^{\ulcorner t_L \urcorner \llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket}$$

and

$$\underline{\Gamma'} \equiv \Gamma' \llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket)^{s_L \llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket}$$

There are three other cases: $t_L \equiv \ulcorner s_R \urcorner$, $t_R \equiv \ulcorner s_L \urcorner$ and $t_R \equiv \ulcorner s_R \urcorner$, but they are all similar.

- $\multimap$-FORM: The inputs are

$$\mathfrak{r} \mid \underline{\Gamma}, \underline{\Gamma'} \vdash A \text{ type}$$
$$\mathfrak{p} \prec (\mathfrak{s} \prec \Xi)\{(\mathfrak{t}' \prec \Phi^{t_L} \otimes \Phi^{t_R})^?\} \otimes \mathfrak{r} \mid \Gamma^{s_L \otimes s_R}, \Gamma', x^{\mathfrak{r}} : A \vdash B \text{ type}$$

By Lemma B.3.22 on A and induction on B,

$$\mathfrak{r} \mid \underline{\Gamma, \Gamma' \llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket} \vdash A \llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket \text{ type}$$
$$\mathfrak{p} \prec (\mathfrak{s} \prec \Xi\{\downarrow(\mathfrak{t} \prec \Phi)^{?\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket}\}) \otimes \mathfrak{r} \mid \Gamma, \Gamma' \llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket, x^{\mathfrak{r}} : A \llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket$$
$$\vdash B \llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket \text{ type}$$

and these are of the right shape to reapply the rule.

- $\multimap$-ELIM: As in the case for $\otimes$-INTRO, first suppose we are not in the special case where one of the slices s_L or s_R used in the term is exactly the top colour of t_L or t_R. Then we have inputs

$$\mathfrak{s} \prec \Xi\{(\mathfrak{t}' \prec \Phi^{t_L} \otimes \Phi^{t_R}, \dots)^?\} \vdash s_L \boxtimes s_R \text{ split}$$
$$(\mathfrak{s} \prec \Xi\{(\mathfrak{t}' \prec \Phi^{t_L} \otimes \Phi^{t_R}, \dots)^?\})^{s_L} \mid (\Gamma^{t_L \otimes t_R}, \Gamma')^{s_L} \vdash f : \textcircled{\prod}_{(x^{\mathfrak{r}} : A)}{}^{\mathfrak{p}} B$$
$$(\mathfrak{s} \prec \Xi\{(\mathfrak{t}' \prec \Phi^{t_L} \otimes \Phi^{t_R}, \dots)^?\})^{s_R} \mid (\Gamma^{t_L \otimes t_R}, \Gamma')^{s_R} \vdash a : A$$

Then applying MERGE/DISPATCH,

$$\mathfrak{s} \prec \Xi\{\downarrow(\mathfrak{t} \prec \Phi)^{?\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket}\} \vdash s_L \llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket \boxtimes s_R \llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket \text{ split}$$
$$(\mathfrak{s} \prec \Xi\{\downarrow(\mathfrak{t} \prec \Phi)^{?\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket}\})^{s_L \llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket} \mid (\Gamma, \Gamma' \llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket)^{s_L \llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket}$$
$$\vdash f \llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket : \left(\textcircled{\prod}_{(x^{\mathfrak{r}} : A)}{}^{\mathfrak{p}} B \right) \llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket$$
$$(\mathfrak{s} \prec \Xi\{\downarrow(\mathfrak{t} \prec \Phi)^{?\llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket}\})^{s_R \llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket} \mid (\Gamma, \Gamma' \llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket)^{s_R \llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket}$$
$$\vdash a \llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket : A \llbracket \mathfrak{t} \prec t_L \boxtimes t_R / \mathfrak{t}' \rrbracket$$

By definition,

$$\left(\textcircled{\textstyle\prod}_{(x^{\mathfrak{r}}:A)}{}^{\mathfrak{p}}B\right)[\![\mathfrak{t} \prec t_L \boxtimes t_R/\mathfrak{t}']\!] \equiv \textcircled{\textstyle\prod}_{(x^{\mathfrak{r}}:A[\![\mathfrak{t}\prec t_L\boxtimes t_R/\mathfrak{t}']\!])}{}^{\mathfrak{p}}B[\![\mathfrak{t} \prec t_L \boxtimes t_R/\mathfrak{t}']\!]$$

and so reapplying the rule gives:

$$\begin{aligned}
&\mathfrak{s} \prec \Xi\{\downarrow(\mathfrak{t} \prec \Phi)^{?[\![\mathfrak{t}\prec t_L\boxtimes t_R/\mathfrak{t}']\!]}\}\\
&\mid \Gamma, \Gamma'[\![\mathfrak{t} \prec t_L \boxtimes t_R/\mathfrak{t}']\!]\\
&\vdash f[\![\mathfrak{t} \prec t_L \boxtimes t_R/\mathfrak{t}']\!]\langle a[\![\mathfrak{t} \prec t_L \boxtimes t_R/\mathfrak{t}']\!]\rangle\\
&: B[\![\mathfrak{t} \prec t_L \boxtimes t_R/\mathfrak{t}']\!][a[\![\mathfrak{t} \prec t_L \boxtimes t_R/\mathfrak{t}']\!]/x][\![\mathfrak{s} \prec s_L[\![\mathfrak{t} \prec t_L \boxtimes t_R/\mathfrak{t}']\!] \boxtimes s_R[\![\mathfrak{t} \prec t_L \boxtimes t_R/\mathfrak{t}']\!]/\mathfrak{p}]\!]
\end{aligned}$$

This type is correct, by Lemma B.1.7 and Lemma B.1.16.

□

Lemma B.3.25.

$$\frac{\Phi \mid \Gamma, \underline{\Gamma'[\![\mathfrak{t} \prec s_L \boxtimes s_R/\mathfrak{t}']\!]} \vdash \mathcal{J}}{\Phi \mid \Gamma, \underline{\Gamma'} \vdash \mathcal{J}}$$

Proof. Repeated application of Lemma B.3.6, together with Lemma B.1.20. □