A Fibrational Framework for Substructural and Modal Logics*

Daniel R. Licata[1], Michael Shulman[2], and Mitchell Riley[3]

1 **Wesleyan University, Middletown, CT, USA**
2 **University of San Diego, San Diego, CA, USA**
3 **Wesleyan University, Middletown, CT, USA**

Abstract

We define a general framework that abstracts the common features of many intuitionistic substructural and modal logics / type theories. The framework is a sequent calculus / normal-form type theory parametrized by a *mode theory*, which is used to describe the structure of contexts and the structural properties they obey. In this sequent calculus, the context itself obeys standard structural properties, while a term, drawn from the mode theory, constrains how the context can be used. Product types, implications, and modalities are defined as instances of two general connectives, one positive and one negative, that manipulate these terms. Specific mode theories can express a range of substructural and modal connectives, including non-associative, ordered, linear, affine, relevant, and cartesian products and implications; monoidal and non-monoidal functors, (co)monads and adjunctions; n-linear variables; and bunched implications. We prove cut (and identity) admissibility independently of the mode theory, obtaining it for many different logics at once. Further, we give a general equational theory on derivations / terms that, in addition to the usual $\beta\eta$-rules, characterizes when two derivations differ only by the placement of structural rules. Additionally, we give an equivalent semantic presentation of these ideas, in which a mode theory corresponds to a 2-dimensional cartesian multicategory, the framework corresponds to another such multicategory with a functor to the mode theory, and the logical connectives make this into a bifibration. Finally, we show how the framework can be used both to encode existing existing logics / type theories and to design new ones.

1998 ACM Subject Classification F.4.1 Mathematical Logic

Keywords and phrases type theory, modal logic, substructural logic, homotopy type theory

Digital Object Identifier 10.4230/LIPIcs.FSCD.2017.25

1 Introduction

In ordinary intuitionistic logic or λ-calculus, assumptions or variables can go unused (weakening), be used in any order (exchange), be used more than once (contraction), and be used in any position in a term. *Substructural* logics, such as linear logic, ordered logic, relevant logic, and affine logic, omit some of these structural properties of weakening, exchange, and contraction, while *modal logics* place restrictions on where variables may be used – e.g. a formula $\Box\, C$ can only be proved using assumptions of $\Box\, A$, while an assumption of $\Diamond\, A$

* This material is based on research sponsored by The United States Air Force Research Laboratory under agreement number FA9550-15-1-0053 and FA9550-16-1-0292. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the United States Air Force Research Laboratory, the U.S. Government, or Carnegie Mellon University.

can only be used when the conclusion is $\Diamond C$. Substructural and modal logics have had many applications to both functional and logic programming, modeling concepts such state, staging, distribution, and concurrency. They are also used as *internal languages* of categories, where one uses an appropriate logical language to do constructions "inside" a particular mathematical setting, which often results in shorter statements than working "externally". For example, to define a function externally in domains, one must first define the underlying set-theoretic function, and then prove that it is continuous; when using untyped λ-calculus as an internal language of domains, one writes what looks like only the function part, and continuity follows from a general theorem about the language itself. Substructural logics extend this idea to various forms of monoidal categories, while modal logics describe monads and comonads. Recent work [30, 31] proposed using modal operators to add a notion of *cohesion* to homotopy type theory/univalent foundations [33, 32]. Without going into the precise details, the idea is to add a triple $\int \dashv \flat \dashv \sharp$ of type operators, where for example $\sharp$ and $\int$ are monads (like a modal possibility $\Diamond$ or $\bigcirc$), $\flat$ is a comonad (like a modal necessity $\Box$), and there is an adjunction structure between them ($\flat A \to B$ is the same as $A \to \sharp B$). This raised the question of how to best add modalities with these properties to type theory.

Because other similar applications rely on functors with different properties, we would like general tools for going from a semantic situation of interest to a well-behaved logic/type theory for it – e.g. one with cut admissibility / normalization and identity admissibility / η-expansion. In previous work [15], we considered the special case of a single-assumption logic, building most directly on adjoint logic [5, 6, 26]. Here we extend this previous work to the multi-assumption case. The resulting framework is quite general and covers many existing intuitionistic substructural and modal connectives: non-associative, ordered, linear, affine, relevant, and cartesian products and implications; combinations thereof such as bunched logic [21] and resource separation [3]; n-linear variables [24, 1, 17]; the comonadic $\Box$ and linear exponential ! and subexponentials [20, 10]; monadic $\Diamond$ and $\bigcirc$ modalities; and adjoint logic F and G [5, 6, 26], including the single-assumption 2-categorical version from our previous work [15]. A central syntactic result is that cut and identity are admissible for our framework itself, and this implies cut admissibility for any logic that can be described in the framework, including all of the above, as well as any new logics that one designs using it. When we view the derivations in the framework as terms in a type theory, this gives an immediate normalization (and η-expansion) result. Our focus here is on propositional, single-conclusioned substructural and modal logics, leaving extensions to quantifiers, multi-conclusioned logics, and dependent types to future work.

At a high level, the framework makes use of the fact that all of the above logics / type theories are a restriction on how variables can be used in ordinary structural/cartesian proofs. We express these restrictions using a first layer, which is a simple type theory for what we will call *modes* and *context descriptors*. The modes are just a collection of base types, which we write as p, q, r, while a context descriptor α is a first-order term built from variables and function symbols. The next layer is the main logic. Each proposition/type is assigned a mode, and the basic sequent is $x_1 : A_1, \ldots, x_n : A_n \vdash_\alpha C$, where if A_i has mode p_i, and C has mode q, then $x_1 : p_1, \ldots, x_n : p_n \vdash \alpha : q$. We use a sequent calculus to concisely describe cut-free derivations/normal forms, but everything can be translated to natural deduction. We write Γ for $x_1 : A_1, \ldots, x_n : A_n$, and Γ itself behaves like an ordinary structural/cartesian context, while the substructural and modal aspects are enforced by the *term* α, which constrains how the resources from Γ may be used. For example, in linear logic/ordered logic/BI, the context is usually taken to be a multiset/list/tree. We represent this by a pair of an ordinary structural context Γ, together with a term α that describes

the multiset or list or tree structure, labeled with variables from the ordinary context at the leaves. We pronounce $\Gamma \vdash_\alpha A$ as "Γ proves A {along,over,using} α".

For example, if we have a mode n, together with a context descriptor constant $x : \mathsf{n}, y : \mathsf{n} \vdash x \odot y : \mathsf{n}$, then an example sequent $x : A, y : B, z : C, w : D \vdash_{(y \odot x) \odot z} E$ should be read as saying that we must prove E using the resources y and x and z (but not w) according to the particular tree structure $(y \odot x) \odot z$. If we say nothing else, the framework will treat $\odot$ as describing a non-associative, linear, ordered context [14]: if we have a product-like type $A \odot B$ internalizing this context operation,[1] then we will *not* be able to prove associativity $((A \odot B) \odot C \dashv\vdash A \odot (B \odot C))$ or exchange $(A \odot B \vdash B \odot A)$ etc. To get from this basic structure to a linear or affine or relevant or cartesian system, we provide a way to add structural properties governing the context descriptor term α. We analyze structural properties as *equations*, or more generally *directed transformations*, on such terms. For example, to specify linear logic, we will add a unit element $1 : \mathsf{n}$ together with equations making $(\odot, 1)$ into a commutative monoid $(x \odot (y \odot z) = (x \odot y) \odot z$ and $x \odot 1 = x = 1 \odot x$ and $x \odot y = y \odot x)$ so that the context descriptors ignore associativity and order. To get BI, we add an additional commutative monoid $(\times, \top)$ (with weakening and contraction, as discussed below), so that a BI context tree $(x : A, y : B); (z : C, w : D)$ can be represented by the ordinary context $x : A, y : B, z : C, w : D$ with the term $(x \odot y) \times (z \odot w)$ describing the tree. Because the context descriptors are themselves ordinary structural/cartesian terms, the same variable can occur more than once or not at all. A descriptor such as $x \odot x$ captures the idea that we can use the *same* variable x twice, expressing n-linear types. Thus, we can express contraction for a particular context descriptor $\odot$ as a transformation $x \Rightarrow x \odot x$ (one use of x allows two). Weakening, on the other hand, is represented by a transformation $x \Rightarrow 1$, which is oriented to allow throwing away an allowed use of x, but not creating an allowed use from nothing. We refer to these as *structural transformations*, to evoke their use in representing the structural properties of object logics that are embedded in our framework. The main sequent $\Gamma \vdash_\alpha A$ respects the specified structural properties in the sense that when $\alpha = \beta$, we regard $\Gamma \vdash_\alpha A$ and $\Gamma \vdash_\beta A$ as the same sequent (so a derivation of one is a derivation of the other), while when $\alpha \Rightarrow \beta$, there will be an operation that takes a derivation of $\Gamma \vdash_\beta A$ to a derivation of $\Gamma \vdash_\alpha A$ – i.e. uses of transformations are explicitly marked in the term.

Modal logics will generally involve a mode theory with more than one mode. For example, a context descriptor $x : \mathsf{c} \vdash \mathsf{f}(x) : \mathsf{l}$ will generate an adjoint pair of functors between the two modes, as in the adjoint syntax for linear logic's ! [6] or other modal operators [26]. Using this, a context descriptor $\mathsf{f}(x) \odot y$ expresses permission to use x in a cartesian way and y in a linear way. Structural transformations are used to describe how these modal operators interact with each other and with the products, and for some systems [15] it is important that there can be more than one transformation between a given pair of context descriptors.

A guiding principle of the framework is a meta-level notion of *structurality over structurality*. For example, we always have *weakening over weakening*: if $\Gamma \vdash_\alpha A$ then $\Gamma, y : B \vdash_\alpha A$, where α itself is weakened with y. This does not prevent encodings of relevant logics: though we might weaken a derivation of $\Gamma \vdash_{x_1 \odot \ldots \odot x_n} A$ ("use x_1 through x_n") to a derivation of $\Gamma, y : B \vdash_{x_1 \odot \ldots \odot x_n} A$, the (weakened) context descriptor does not allow the use of y. Similarly, we have exchange over exchange and contraction over contraction. The *identity-over-identity* principle says that we should be able to prove A using exactly an assumption $x : A$ $(\Gamma, x : A \vdash_x A)$. The cut principle says that from $\Gamma, x : A \vdash_\beta B$ and $\Gamma \vdash_\alpha A$ we

[1] We overload binary operations to refer both to context descriptors and propositional connectives, relying on metavariables ($\alpha_1 \odot \alpha_2$ vs. $A_1 \odot A_2$) to distinguish them.

get $\Gamma \vdash_{\beta[\alpha/x]} B$ – the context descriptor for the result of the cut is the substitution of the context descriptor used to prove A into the one used to prove B. For example, together with weakening-over-weakening, this captures the usual cut principle of linear logic, which says that cutting $\Gamma, x : A \vdash B$ and $\Delta \vdash A$ yields $\Gamma, \Delta \vdash B$: if Γ binds $x_1, \ldots, x_n$ and Δ binds $y_1, \ldots, y_n$, then we will represent the two derivations to be cut together by sequents with $\beta = x_1 \odot \ldots \odot x_n \odot x$ and $\alpha = y_1 \odot \ldots \odot y_n$, so $\beta[\alpha/x] = x_1 \odot \ldots \odot x_n \odot y_1 \odot \ldots \odot y_n$ correctly deletes x and replaces it with the variables from Δ. In more subtle situations such as BI, the substitution will insert the resources used to prove the cut formula in the correct place in the tree. Our cut algorithm follows cut admissibility for structural (cartesian) intuitionistic logic [22], and applies the same cut reductions to all mode theories. In substructural situations, certain portions of this general algorithm are unnecessary, but not harmful. For example, in a setting without contraction, our algorithm will recursively cut into all premises of a rule, even though the variable can only occur in one premise – but the extra recursive cuts for variables that do not occur will leave the derivation unchanged, as desired.

The framework has two main logical connectives / type constructors. The first, $\mathsf{F}_\alpha(\Delta)$, generalizes the left-adjoint F of adjoint logic and the multiplicative products (e.g. $\otimes$ of linear logic). The second, $\mathsf{U}_{x.\alpha}(\Delta \mid A)$, generalizes the right-adjoint G/U of adjoint logic and implication (e.g. $A \multimap B$ in linear logic). Here Δ is a context of assumptions $x_i : A_i$, and trivializing the context descriptors (i.e. adding an equation $\alpha = \beta$ for all α and β) degenerates $\mathsf{F}_\alpha(\Delta)$ into the ordinary intuitionistic product $A_1 \times \ldots \times A_n$, while $\mathsf{U}_{x.\alpha}(\Delta \mid A)$ becomes $A_1 \to \ldots \to A_n \to A$. As one would expect, F is left-invertible and U is right-invertible. In linear logic terms, our F and U cover both the multiplicatives and exponentials; additives can be defined separately by the usual rules. Moreover, though F and U form an adjoint pair, the subset of derivations that use either F_α or U_α but not both (for a particular α) describe constructions on functors that do not have an adjoint. We discuss many examples of *logical adequacy* theorems, showing that a sequent can be proved in a standard sequent calculus for a logic iff its embedding using these connectives can be proved in the framework.

Being a very general theory, our framework treats the object-logic structural properties in a general but naïve way, allowing an arbitrary structural transformation to be applied at the non-invertible rules for F and U and at the leaves of a derivation. For specific embedded logics, there is often a more refined discipline that suffices – e.g. for cartesian logic, always contract all assumptions in all premises, and only weaken at the leaves. We view our framework as a tool for bridging the gap between an intended semantic situation (in the cohesion example mentioned, "a comonad and a monad which are themselves adjoint") and a proof theory: the framework gives *some* proof theory for the semantics, and the placement of structural rules can then be optimized purely in syntax. To support this mode of use, we give an equational theory on derivations/terms that identifies different placements of the same structural rules. This can be used to prove correctness of such optimizations not just at the level of provability, but also identity of derivations – which matters for our intended applications to internal languages. We discuss some preliminary work on *equational adequacy*, which extends the logical correspondence to isomorphisms of definitional-equality-classes of derivations.

Semantically, the logic corresponds to a functor between *2-dimensional cartesian multicategories* which is a fibration in various senses. Multicategories are a generalization of categories which allow more than one object in the domain of morphisms, and cartesianness means that the multiple domain objects are treated structurally. The 2-dimensionality supplies a notion of morphism between (multi)morphisms. A *mode theory* specifying context descriptors and structural properties is analyzed as a cartesian 2-multicategory, with the descriptors as 1-cells and the structural properties as 2-cells. The functor relates the sequent

judgement to the mode theory, specifying the mode of each proposition and the context descriptor of a sequent. The fibration conditions (similar to [12, 13]) give respect for the structural transformations and the presence of F and U types. We prove that the sequent calculus and the equational theory are sound and complete for this semantics: the syntax can be interpreted in any bifibration, and itself determines one. This semantics shows that an interesting class of type theories can be identified with a class of more mathematical objects, fibrations of cartesian 2-multicategories, thus providing some progress towards characterizing substructural and modal type theories in mathematical terms.

In Section 2, we present the syntax of the framework. In Section 3, we discuss how a number of logics are represented. In Section 4, we give the $\beta\eta$-equational theory on derivations. In Section 5, we discuss the framework's categorical semantics. Proofs (of cut and identity admissibility, soundness and completeness of the semantics, and adequacy of encodings) and additional examples (subexponentials, modal S4 $\Box$, and strong/$\Box$-strong monads) are available in an extended version of this paper [16].

2 Sequent Calculus

2.1 Mode Theories

The first layer of our framework is a type theory whose types we will call *modes*, and whose terms we will call *context descriptors* or *mode morphisms*. To a first approximation, context descriptors are multi-sorted first-order terms with equality and "less than or equal to" relations. The only modes are atomic/base types p. A term is either a variable (bound in a context ψ) or a typed n-ary constant (function symbol) c applied to terms of the appropriate types. Two terms may be equal, written $\alpha \equiv \beta$, or α may be stronger than β, written $\alpha \Rightarrow \beta$.

This is formalized in the notion of signature, or *mode theory*, defined in Figure 1. The judgement $\Sigma\,\mathsf{sig}$ means that Σ is a well-formed signature. The top line says that a signature is either empty, or a signature extended with a new mode declaration, or a signature extended with a typed constant/function symbol, all of whose modes are declared previously in the signature. The notation $p_1, \ldots, p_n \to q$ is not itself a mode, but notation for declaring a function symbol in the signature (it cannot occur on the right-hand side of a typing judgement). For example, the type and term constructors for a monoid $(\odot, 1)$ are represented by a signature $\mathsf{p}\,\mathsf{mode}, \odot : (\mathsf{p}, \mathsf{p} \to \mathsf{p}), 1 : (\to \mathsf{p})$.

We elide the rules for the judgement $\vdash_\Sigma \psi\,\mathsf{ctx}$, which simply says that each mode used in the context of variable declarations ψ is declared in Σ. The judgement $\psi \vdash_\Sigma \alpha : p$ defines well-typedness of context descriptor terms, which are either a variable declared in the context, or a constant declared in the signature applied to arguments of the correct types. The judgement $\psi \vdash_\Sigma \gamma : \psi'$ defines a substitution as a tuple of terms in the standard way. The context ψ in these judgements enjoys the cartesian structural properties (associativity, unit, weakening, exchange, contraction). Simultaneous substitution into terms and substitutions is defined as usual (e.g. $x[\gamma, \alpha/x] := \alpha$ and $\mathsf{c}(\vec{\alpha_i})[\gamma] := \mathsf{c}(\alpha_i[\vec{\gamma}])$).

Returning to the top of the figure, the final two rules of the judgement $\Sigma\,\mathsf{sig}$ permit two additional forms of signature declaration. The first of these extends a signature with an equational axiom between two terms α and α' that have the same mode p, in the same context ψ, relative to the prior signature Σ. These equational axioms will be used to encode reversible object language structural properties, such as associativity, commutativity, and unit laws. For example, to specify the right unit law for the above monoid $(\odot, 1)$, we add an axiom $(x \odot 1 \equiv x : (x : \mathsf{p}) \to \mathsf{p})$ to the signature, which can be read as "$x \odot 1$ is equal to x as a morphism from $(x : \mathsf{p})$ to p". The judgement $\psi \vdash_\Sigma \alpha \equiv \alpha' : p$ (omitted from the figure;

Signatures $\Sigma\,\mathsf{sig}$

$$\frac{}{\cdot\,\mathsf{sig}} \qquad \frac{\Sigma\,\mathsf{sig}}{(\Sigma, p\,\mathsf{mode})\,\mathsf{sig}} \qquad \frac{\Sigma\,\mathsf{sig} \quad (p_1\,\mathsf{mode}, \ldots, p_n\,\mathsf{mode}, q\,\mathsf{mode}) \in \Sigma}{(\Sigma, c :\ p_1, \ldots, p_n \to q)\,\mathsf{sig}}$$

$$\frac{\Sigma\,\mathsf{sig} \quad \vdash_\Sigma \psi\,\mathsf{ctx} \quad p\,\mathsf{mode} \in \Sigma \quad \psi \vdash_\Sigma \alpha : p \quad \psi \vdash_\Sigma \alpha' : p}{(\Sigma, (\alpha \equiv \alpha' : \psi \to p))\,\mathsf{sig}}$$

$$\frac{\Sigma\,\mathsf{sig} \quad \vdash_\Sigma \psi\,\mathsf{ctx} \quad p\,\mathsf{mode} \in \Sigma \quad \psi \vdash_\Sigma \alpha : p \quad \psi \vdash_\Sigma \alpha' : p}{(\Sigma, (\alpha \Rightarrow \alpha' : \psi \to p))\,\mathsf{sig}}$$

Context descriptors $\psi \vdash_\Sigma \alpha : p$, where $\vdash_\Sigma \psi\,\mathsf{ctx}$ and $p\,\mathsf{mode} \in \Sigma$

$$\frac{x : p \in \psi}{\psi \vdash_\Sigma x : p} \qquad \frac{(\mathsf{c} : p_1, \ldots, p_n \to q) \in \Sigma \quad \psi \vdash_\Sigma \alpha_i : p_i}{\psi \vdash_\Sigma \mathsf{c}(\alpha_1, \ldots, \alpha_n) : q}$$

Mode Substitutions $\psi \vdash_\Sigma \gamma : \psi'$, where $\vdash_\Sigma \psi\,\mathsf{ctx}$ and $\vdash_\Sigma \psi'\,\mathsf{ctx}$

$$\frac{}{\psi \vdash_\Sigma \cdot : \cdot} \qquad \frac{\psi \vdash_\Sigma \gamma : \psi' \quad \psi \vdash_\Sigma \alpha : p}{\psi \vdash_\Sigma \gamma, \alpha/x : \psi', x : p}$$

Structural transformations $\psi \vdash_\Sigma \alpha \Rightarrow_p \alpha'$, where $\psi \vdash_\Sigma \alpha : p$ and $\psi \vdash_\Sigma \alpha' : p$

$$\frac{}{\psi \vdash_\Sigma \alpha \Rightarrow_p \alpha} \qquad \frac{\psi \vdash_\Sigma \alpha_1 \Rightarrow_p \alpha_2 \quad \psi \vdash_\Sigma \alpha_2 \Rightarrow_p \alpha_3}{\psi \vdash_\Sigma \alpha_1 \Rightarrow_p \alpha_3}$$

$$\frac{\psi, x : p, \psi' \vdash_\Sigma \beta \Rightarrow_q \beta' \quad \psi, \psi' \vdash_\Sigma \alpha \Rightarrow_p \alpha'}{\psi, \psi' \vdash_\Sigma \beta[\alpha/x] \Rightarrow_q \beta'[\alpha'/x]} \qquad \frac{(\alpha \Rightarrow \alpha' : \psi \to p) \in \Sigma}{\psi \vdash_\Sigma \alpha \Rightarrow_p \alpha'}$$

Figure 1 Syntax for mode theories.

the rules are the same as for $\Rightarrow$ plus symmetry) is the least congruence closed under these axioms.

The second of these extends a signature with a directed structural transformation axiom between two terms α and α' that have the same mode p, in the same context ψ, relative to the prior signature Σ. As discussed above, these structural transformations will be used to represent object language structural properties such as weakening and contraction that are not invertible. The judgement $\psi \vdash_\Sigma \alpha \Rightarrow_p \alpha'$ defines these transformations: it is the least precongruence (preorder compatible with the term formers) closed under the axioms specified in the signature Σ. For example, to say that the above monoid $(\odot, 1)$ is affine, we add in Σ a transformation axiom $(x \Rightarrow 1 : (x : \mathsf{p}) \to \mathsf{p})$.

Because context descriptors α and their equality $\alpha_1 \equiv \alpha_2$ are defined prior to the subsequent judgements, we suppress this equality by using α to refer to a term-modulo-$\equiv$– that is, we assume a metatheory with quotient sets/types, and use meta-level equality for object-level equality [2]. For example, because the judgement $\psi \vdash \alpha \Rightarrow_p \beta$ is indexed by equivalence classes of context descriptions, the reflexivity rule above implicitly means $\alpha \equiv \beta$ implies $\alpha \Rightarrow \beta$. In examples, we will notate a signature declaration introducing a term constant/function symbol by showing the function symbol applied to variables, rather than

$$\boxed{\text{Types } A, B, C \quad A \ \mathsf{type}_p}$$

$$\frac{}{P \ \mathsf{type}_p} \qquad \frac{\psi \vdash \alpha : q \quad \Delta \ \mathsf{ctx}_\psi}{\mathsf{F}_\alpha(\Delta) \ \mathsf{type}_q} \qquad \frac{\psi, x : q \vdash \alpha : p \quad \Delta \ \mathsf{ctx}_\psi \quad A \ \mathsf{type}_p}{\mathsf{U}_{x.\alpha}(\Delta \mid A) \ \mathsf{type}_q}$$

$$\boxed{\text{Contexts } \Gamma, \Delta \quad \Gamma \ \mathsf{ctx}_\psi}$$

$$\frac{}{\cdot \ \mathsf{ctx}.} \qquad \frac{\Gamma \ \mathsf{ctx}_\psi \quad A \ \mathsf{type}_p}{\Gamma, x : A \ \mathsf{ctx}_{\psi, x:p}}$$

$$\boxed{\Gamma \vdash_\alpha A \text{ where } \Gamma \ \mathsf{ctx}_\psi \text{ and } A \ \mathsf{type}_q \text{ and } \psi \vdash \alpha : q}$$

$$\frac{\Gamma, \Gamma', \Delta \vdash_{\beta[\alpha/x]} C}{\Gamma, x : \mathsf{F}_\alpha(\Delta), \Gamma' \vdash_\beta C} \ \mathsf{FL} \qquad \frac{\beta \Rightarrow \alpha[\gamma] \quad \Gamma \vdash_\gamma \Delta}{\Gamma \vdash_\beta \mathsf{F}_\alpha(\Delta)} \ \mathsf{FR}$$

$$\frac{x : \mathsf{U}_{x.\alpha}(\Delta \mid A) \in \Gamma \quad \beta \Rightarrow \beta'[\alpha[\gamma]/z] \quad \Gamma \vdash_\gamma \Delta \quad \Gamma, z : A \vdash_{\beta'} C}{\Gamma \vdash_\beta C} \ \mathsf{UL}$$

$$\frac{\Gamma, \Delta \vdash_{\alpha[\beta/x]} A}{\Gamma \vdash_\beta \mathsf{U}_{x.\alpha}(\Delta \mid A)} \ \mathsf{UR} \qquad \frac{x : P \in \Gamma \quad \beta \Rightarrow x}{\Gamma \vdash_\beta P} \ \mathsf{v}$$

$$\boxed{\Gamma \vdash_\gamma \Delta \text{ where } \Gamma \ \mathsf{ctx}_\psi \text{ and } \Delta \ \mathsf{ctx}_{\psi'} \text{ and } \psi \vdash \gamma : \psi'}$$

$$\frac{}{\Gamma \vdash_\cdot \cdot} \ \cdot \qquad \frac{\Gamma \vdash_\gamma \Delta \quad \Gamma \vdash_\alpha A}{\Gamma \vdash_{\gamma, \alpha/x} \Delta, x : A} \ \_,\_$$

Figure 2 Sequent Calculus.

writing the formal $\mathsf{c} : p_1, \ldots, p_n \to q$. For example, we write $x : \mathsf{p}, y : \mathsf{p} \vdash x \odot y : \mathsf{p}$ for $\odot : \mathsf{p}, \mathsf{p} \to \mathsf{p}$. We also suppress the signature Σ.

2.2 Sequent Calculus Rules

For a fixed mode theory Σ, we define a second layer of judgements in Figure 2. The first judgement assigns each proposition/type A a mode p. Encodings of non-modal logics will generally only make use of one mode, while modal logics use different modes to represent different notions of truth, such as the linear and cartesian categories in the adjoint decomposition of linear logic [5, 6] and the true/valid/lax judgements in modal logic [23]. The next judgement assigns each context Γ a mode context ψ. Formally, we think of contexts as ordered: we do not regard $x : A, y : B$ and $y : B, x : A$ as the same context, though we will have an admissible exchange rule that passes between derivations in one and the other.

The sequent judgement $\Gamma \vdash_\alpha A$ relates a context $\Gamma \ \mathsf{ctx}_\psi$ and a type $A \ \mathsf{type}_p$ and context descriptor $\psi \vdash \alpha : p$, while the substitution judgement $\Gamma \vdash_\gamma \Delta$ relates $\Gamma \ \mathsf{ctx}_\psi$ and $\Delta \ \mathsf{ctx}_{\psi'}$ and $\psi \vdash \gamma : \psi'$. Because $\Gamma \ \mathsf{ctx}_\psi$ means that each variable in Γ is in ψ, where $x : A_i \in \Gamma$ implies $x : p_i$ in ψ with $A_i \ \mathsf{type}_{p_i}$, we think of Γ as binding variable names both in α and for use in the derivation.

We now explain the rules for the sequent calculus; the reader may wish to refer to the examples in Section 3 in parallel with this abstract description. We assume atomic propositions P are given a specified mode p, and state identity as a primitive rule only for them with the v rule. This says that $\Gamma, x : P \vdash_x P$, and additionally composes with a structural transformation $\beta \Rightarrow x$. Using a structural property at a leaf of a derivation is common in e.g. affine logic, where the derivation of $\beta \Rightarrow x$ would use weakening to forget any additional resources besides x.

Next, we consider the $\mathsf{F}_\alpha(\Delta)$ type, which "internalizes" the context operation α as a type/proposition. Syntactically, we view the context $\Delta = x_1 : A_1, \ldots, x_n : A_n$ where A_i type_{p_i} as binding the variables $x_i : p_i$ in α, so for example $\mathsf{F}_\alpha(x : A, y : B)$ and $\mathsf{F}_{\alpha[x \leftrightarrow x']}(x' : A, y : B)$ are α-equivalent types (in de Bruijn form we would write $\mathsf{F}_\alpha(A_1, \ldots, A_n)$ and use indices in α). The type formation rule says that F moves covariantly along a mode morphism α, representing a "product" (in a loose sense) of the types in Δ structured according to the context descriptor α. A typical binary instance of F is a multiplicative product ($A \otimes B$ in linear logic), which, given a binary context descriptor $\odot$ as in the introduction, is written $\mathsf{F}_{x \odot y}(x : A, y : B)$. A typical nullary instance is a unit (1 in linear logic), written $\mathsf{F}_1()$. A typical unary instance is the F connective of adjoint logic, which for a unary context descriptor constant $\mathsf{f} : \mathsf{p} \to \mathsf{q}$ is written $\mathsf{F}_{\mathsf{f}(x)}(x : A)$. We sometimes write $\mathsf{F}_\mathsf{f}(A)$ in this case, eliding the variable name, and similarly for a unary U.

The rules for our F connective capture a pattern common to all of these examples. The left FL rule says that $\mathsf{F}_\alpha(\Delta)$ "decays" into Δ, but structuring the uses of resources in Δ with α by the substitution $\beta[\alpha/x]$. We assume that Δ is α-renamed to avoid collision with Γ (the proof term here is a "split" that binds variables for each position in Δ). The placement of Δ at the right of the context is arbitrary (because we have exchange-over-exchange), but we follow the convention that new variables go on the right to emphasize that Γ behaves mostly as in ordinary cartesian logic. The right FR rule says that you must rewrite (using structural transformations) the context descriptor to have an α at the outside, with a mode substitution γ that divides the existing resources up between the positions in Δ, and then prove each formula in Δ using the specified resources. We leave the typing of γ implicit, though there is officially a requirement $\psi \vdash \gamma : \psi'$ where Γ ctx_ψ and Δ $\mathsf{ctx}_{\psi'}$, as required for the second premise to be a well-formed sequent. Another way to understand this rule is to begin with the "axiomatic FR" instance $\mathsf{FR}^* :: \Delta \vdash_\alpha \mathsf{F}_\alpha(\Delta)$ which says that there is a map from Δ to $\mathsf{F}_\alpha(\Delta)$ along α. Then, in the same way that a typical right rule for coproducts builds a precomposition into an "axiomatic injection" such as $\mathsf{inl} :: A \vdash A + B$, the FR rule builds a precomposition with $\Gamma \vdash_\gamma \Delta$ and then an application of a structural rule $\beta \Rightarrow \alpha[\gamma]$ into the "axiomatic" version, in order to make cut and respect for transformations admissible.

Next, we turn to $\mathsf{U}_{x.\alpha}(\Delta \mid A)$. As a first approximation, if we ignore the context descriptors and structural properties, $\mathsf{U}_\_(\Delta \mid A)$ behaves like $\Delta \to A$, and the UL and UR rules are an annotation of the usual structural/cartesian rules for implication. In a formula $\mathsf{U}_{x.\alpha}(\Delta \mid A)$, the context descriptor α has access to the variables from Δ as well as an extra variable x, whose mode is the same as the *overall mode of* $\mathsf{U}_{x.\alpha}(\Delta \mid A)$, while the mode of A itself is the mode of the conclusion of α – in terms of typing, U is contravariant where F is covariant. It is helpful to think of x as standing for the context that will be used to prove $\mathsf{U}_{x.\alpha}(\Delta \mid A)$. For example, a typical function type $A \multimap B$ is represented by $\mathsf{U}_{x.x \otimes y}(y : A \mid B)$, which says to extend the "current context" x with a resource y. In UR, the context descriptor β being used to prove the U is substituted *for* x in α (dual to FL, which substituted α into β). The "axiomatic" UL instance $\mathsf{UL}^* :: \Delta, x : \mathsf{U}_{x.\alpha}(\Delta \mid A) \vdash_\alpha A$ says that $\mathsf{U}_{x.\alpha}(\Delta \mid A)$ together with Δ has a map to A along α. (The bound x in $x.\alpha$ subscript is tacitly renamed

to match the name of the assumption in the context, in the same way that the typing rule for $\lambda x.e : \Pi x : A.B$ requires coordination between two variables in different scopes). The full rule builds in precomposition with $\Gamma \vdash_\gamma \Delta$, postcomposition with $\Gamma, z : A \vdash_{\beta'} C$, and precomposition with $\beta \Rightarrow \beta'[\alpha[\gamma]/z]$.

Finally, the rules for substitutions are pointwise. In examples, we will write the components of a substitution directly as multiple premises of FR and UL, rather than packaging them with $\_,\_$ and $\cdot$.

For additives, the context descriptor is not modified; for example, a coproduct/disjunction $A_p + B_p$ type_p for a mode p is given by the following rules:

$$\frac{\Gamma \vdash_\alpha A}{\Gamma \vdash_\alpha A + B} \qquad \frac{\Gamma \vdash_\alpha B}{\Gamma \vdash_\alpha A + B} \qquad \frac{\Gamma, \Gamma', y : A \vdash_{\beta[y/x]} C \quad \Gamma, \Gamma', z : B \vdash_{\beta[z/x]} C}{\Gamma, x : A + B, \Gamma' \vdash_\beta C}$$

Our framework enjoys the following admissible structural rules:

▶ **Theorem 2.1** (Admissibility of cut, identity, structurality-over-structurality, and respect for 2-cells)**.** *The following rules are admissible:*

$$\frac{\Gamma, x : A \vdash_\beta B \quad \Gamma \vdash_\alpha A}{\Gamma \vdash_{\beta[\alpha/x]} B} \qquad \frac{}{\Gamma, x : A \vdash_x A} \qquad \frac{\Gamma \vdash_\alpha C}{\Gamma, y : A \vdash_\alpha C}$$

$$\frac{\Gamma, x : A, y : B \vdash_\alpha C}{\Gamma, y : B, x : A \vdash_\alpha C} \qquad \frac{\alpha \Rightarrow \beta \quad \Gamma \vdash_\beta A}{\Gamma \vdash_\alpha A}$$

The following general constructions can be helpful for understanding how the types behave. We write $A \vdash B$ for $x : A \vdash_x B$. The three "fusion" rules on the left (which are type isomorphisms, not just interprovabilities) relate F and U. Special cases include: $A \times (B \times C)$ is isomorphic to a primitive triple product $\{x : A, y : B, z : C\}$; currying; and associativity of n-ary functions $(A_1, \ldots, A_n \to (B_1, \ldots, B_m \to C)$ is isomorphic to $A_1, \ldots, A_n, B_1, \ldots, B_m \to C)$. Second, the types respect a transformation covariantly for F and contravariantly for U.

▶ **Theorem 2.2** (Fusion and Respect Laws)**.**

$$\begin{array}{rcl} \mathsf{F}_\alpha(\Delta, x : \mathsf{F}_\beta(\Delta'), \Delta'') & \dashv\vdash & \mathsf{F}_{\alpha[\beta/x]}(\Delta, \Delta', \Delta'') \\ \mathsf{U}_{x.\alpha}(\Delta, y : \mathsf{F}_\beta(\Delta'), \Delta'' \mid A) & \dashv\vdash & \mathsf{U}_{x.\alpha[\beta/y]}(\Delta, \Delta', \Delta'' \mid A) \\ \mathsf{U}_{x.\alpha}(\Delta \mid \mathsf{U}_{y.\beta}(\Delta' \mid A)) & \dashv\vdash & \mathsf{U}_{x.\beta[\alpha/y]}(\Delta, \Delta' \mid A) \end{array}$$

$$\begin{array}{rcll} \mathsf{F}_\alpha(\Delta) & \vdash & \mathsf{F}_\beta(\Delta) & \textit{if } \alpha \Rightarrow \beta \\ \mathsf{U}_{x.\beta}(\Delta \mid A) & \vdash & \mathsf{U}_{x.\alpha}(\Delta \mid A) & \textit{if } \alpha \Rightarrow \beta \end{array}$$

3 Examples

3.1 Products and Implications

First, we show how to encode substructural products and implications with various structural properties. A mode theory with one mode m and a constant $x : \mathsf{m}, y : \mathsf{m} \vdash x \odot y : \mathsf{m}$ specifies a completely astructural context (no weakening, exchange, contraction, associativity), as in non-associative Lambek calculus [14]. To pass to *ordered logic* (associativity and unit laws but none of exchange, weakening, and contraction), we add a constant $1 : \mathsf{m}$ and equational axioms $x \odot (y \odot z) \equiv (x \odot y) \odot z$ and $x \odot 1 \equiv x \equiv 1 \odot x$ – i.e. $(\odot, 1)$ is a monoid. To

get linear logic, we additionally add commutativity $x \odot y \equiv y \odot x$. As a first example of using the sequent calculus, we show how commutativity of $\odot$ in the mode theory for linear logic generates commutativity of the corresponding $A \otimes B$ type, which is represented by $\mathsf{F}_{x \odot y}(x : A, y : B)$:

$$\dfrac{\dfrac{x \odot y \Rightarrow (z \odot w)[y/z, x/w] \quad x : A, y : B \vdash_y B \quad x : A, y : B \vdash_x A}{x : A, y : B \vdash_{x \odot y} \mathsf{F}_{z \odot w}(z : B, w : A)}\ \mathsf{FR}}{q : \mathsf{F}_{x \odot y}(x : A, y : B) \vdash_q \mathsf{F}_{z \odot w}(z : B, w : A)}\ \mathsf{FL}$$

First, we use FL to split the product type on the left up, obtaining permission to use its pieces by substituting $(x \odot y)$ for the variable q we began with. Next, to use FR, we must transform the current context descriptor $x \odot y$ into a substitution instance of the one from the type $z \odot w$ – dividing our resources in the form dictated by the type. We take $y/z, x/w$, which requires a transformation $x \odot y \Rightarrow y \odot x$, which is given by reflexivity because of the commutativity axiom in the mode theory. Then we can prove each of A and B by identity, because we have the correct resources in each branch. In the mode theory for ordered logic, without commutativity, the only possible division is $x/z, y/w$, and with permission only to use x the first premise and y in the second, the derivation fails.

Returning to the mode theory of a non-symmetric $\odot$, we show how the two implications/residuations of ordered logic are modeled by U-types; the expected rules are

$$\dfrac{\Gamma, A \vdash^\circ B}{\Gamma \vdash^\circ A \rightharpoonup B} \qquad \dfrac{\Delta \vdash^\circ A \quad \Gamma, B, \Gamma' \vdash^\circ C}{\Gamma, A \rightharpoonup B, \Delta, \Gamma' \vdash^\circ C} \qquad \dfrac{A, \Gamma \vdash^\circ B}{\Gamma \vdash^\circ A \leftharpoonup B} \qquad \dfrac{\Delta \vdash^\circ A \quad \Gamma, B, \Gamma' \vdash^\circ C}{\Gamma, \Delta, A \leftharpoonup B, \Gamma' \vdash^\circ C}$$

We represent these by the U-types $A \rightharpoonup B := \mathsf{U}_{c.c \odot x}(x : A \mid B)$ and $A \leftharpoonup B := \mathsf{U}_{c.x \odot c}(x : A \mid B)$. The UL and UR rules specialize as follows:

$$\dfrac{\Gamma, x : A \vdash_{\beta \odot x} B}{\Gamma \vdash_\beta \mathsf{U}_{c.c \odot x}(x : A \mid B)} \qquad \dfrac{\begin{array}{l} c : \mathsf{U}_{c.c \odot x}(x : A \mid B) \in \Gamma \\ \beta \Rightarrow \beta'[c \odot \alpha / z] \\ \Gamma \vdash_\alpha A \\ \Gamma, z : A \vdash_{\beta'} C \end{array}}{\Gamma \vdash_\beta C}$$

$$\dfrac{\Gamma, x : A \vdash_{x \odot \beta} B}{\Gamma \vdash_\beta \mathsf{U}_{c.x \odot c}(x : A \mid B)} \qquad \dfrac{\begin{array}{l} c : \mathsf{U}_{c.x \odot c}(x : A \mid B) \in \Gamma \\ \beta \Rightarrow \beta'[\alpha \odot c / z] \\ \Gamma \vdash_\alpha A \\ \Gamma, z : A \vdash_{\beta'} C \end{array}}{\Gamma \vdash_\beta C}$$

The UR instances put x on the left or right of the current context descriptor β, by the substitution β/c in UR. Consider the left rule for $\rightharpoonup / \mathsf{U}_{c.c \odot x}(x : A \mid B)$, and suppose that the β in the conclusion is of the form $x_1 \odot \ldots c \ldots \odot x_n$ for distinct variables x_i. Because the only structural transformations are the associativity and unit equations, the transformation must reassociate β as $\beta_1 \odot (c \odot \alpha) \odot \beta_2$, with $\beta' = \beta_1 \odot z \odot \beta_2$, for some β_1 and β_2. Here α plays the role of Δ in the ordered logic rule – the resources used to prove A, which occur to the right of the implication being eliminated. Reading the substitution backwards, the resources β' used for the continuation are "β with $c \odot \alpha$ replaced by the result of the implication," as desired. While c and any variables used in α are still in Γ, permission to use them has been removed from β' – and there is no way to restore such permissions in this mode theory. The rule for $\leftharpoonup$ is the same, but with α on the opposite side of c. For the linear logic mode theory, $\mathsf{U}_{c.c \odot x}(x : A \mid B)$ and $\mathsf{U}_{c.x \odot c}(x : A \mid B)$ are equal types (because commutativity is an equation, and types are parametrized by equivalence-classes of context descriptors), and both represent $A \multimap B$.

Weakening (affine logic) is modeled by adding a directed structural transformation $\mathsf{w} :: x \Rightarrow 1$, while contraction (relevant logic) is modeled by $\mathsf{c} :: x \Rightarrow x \odot x$. As an example

use of weakening, we can show $A \odot B \vdash A$ (formally $z : \mathsf{F}_{x \odot y}(x : A, y : B) \vdash_z A$); and as an example of contraction we can show $A \vdash A \odot A$ (formally $z : A \vdash_z \mathsf{F}_{x \odot y}(x : A, y : A)$):

$$\dfrac{\dfrac{\dfrac{\mathsf{w} :: y \Rightarrow 1}{x \odot y \Rightarrow x \odot 1 \equiv x} \quad \overline{x : A, y : B \vdash_x A}}{x : A, y : B \vdash_{x \odot y} A}}{z : \mathsf{F}_{x \odot y}(x : A, y : B) \vdash_z A}\ \mathsf{FL} \qquad \dfrac{\mathsf{c} :: z \Rightarrow (x \odot y)[z/x, z/y] \quad \overline{z : A \vdash_z A}}{z : A \vdash_z \mathsf{F}_{x \odot y}(x : A, y : A)}\ \mathsf{FR}$$

If we have both $\mathsf{w} :: x \Rightarrow 1$ and $\mathsf{c} :: x \Rightarrow x \odot x$ (with some equations relating them), then $x \odot y$ is a cartesian product in the mode theory, and consequently the type $\mathsf{F}_{x \odot y}(x : A, y : B)$ will behave like a cartesian product type $A \times B$, and $\mathsf{U}_{c.c \odot x}(x : A \mid B)$ like the usual structural $A \to B$. We refer to this mode theory as a *cartesian monoid* and write $(\times, \top)$ for it.

These encodings are adequate in the following sense:

▶ **Theorem 3.1** (Logical Adequacy for Products and Implications). *Write A^* for the encoding of a type as above and extend this pointwise to contexts Γ^*. Further, define $\overline{x_1 : A_1, \ldots, x_n : A_n} = x_1 \odot \ldots \odot x_n$. Then $\Gamma \vdash A$ in the standard sequent calculus iff $\Gamma^* \vdash_{\overline{\Gamma}} A^*$.*

Proof. Proofs for ordered logic (products), affine logic, and cartesian logic are in the extended version. Encoding an object-language derivation is straightforward, because the mode theory is chosen to make each rule derivable. The back-translation from the framework relies on cut-freeness (so that we only need to back-translate normal forms), and a lemma that, for these mode theories, left-rules on variables that are in the framework context Γ but do not occur in the context descriptor α can be strengthened away. ◀

This approach extends to contexts with more than one type of tree node, as in bunched implication [21], which has two context-forming operations Γ, Γ' and $\Gamma; \Gamma'$, along with corresponding products and implications. Both are associative, unital, and commutative, but ; has weakening and contraction while , does not. A context is represented by a tree such as $(x : A, y : B); (z : C, w : D)$ (considered modulo the laws), and the notation $\Gamma[\Delta]$ is used to refer to a tree with a hole $\Gamma[-]$ that has Δ as a subtree at the hole. In sequent calculus style, the rules for the product and implication corresponding to , are

$$\dfrac{\Gamma[A, B] \vdash C}{\Gamma[A * B] \vdash C} \quad \dfrac{\Gamma \vdash A \quad \Delta \vdash B}{\Gamma, \Delta \vdash A * B} \quad \dfrac{\Gamma, A \vdash B}{\Gamma \vdash A \mathrel{-\!\!*} B} \quad \dfrac{\Delta \vdash A \quad \Gamma[B] \vdash C}{\Gamma[A \mathrel{-\!\!*} B, \Delta] \vdash C}$$

We model BI by a mode m with both a commutative monoid $(*, I)$ and a cartesian monoid $(\times, \top)$. We define the BI products and implications using the monoids as above: $A * B := \mathsf{F}_{x * y}(x : A, y : B)$ and $A \times B := \mathsf{F}_{x \times y}(x : A, y : B)$ and $A \mathrel{-\!\!*} B := \mathsf{U}_{c.c * x}(x : A \mid B)$ and $A \to B := \mathsf{U}_{c.c \times x}(x : A \mid B)$. A context descriptor such as $(x \times y) * (z \times w)$ captures the "bunched" structure of a BI context, and substitution for a variable models the hole-filling operation $\Gamma[\Delta]$. The derived left rules for $*$ and $\mathrel{-\!\!*}$ are

$$\dfrac{\Gamma, \Gamma', x : A, y : B \vdash_{\beta[x * y/z]} C}{\Gamma, z : A * B, \Gamma' \vdash_\beta C} \qquad \dfrac{c : A \mathrel{-\!\!*} B \in \Gamma \quad \beta \Rightarrow \beta'[c * \alpha/z] \quad \Gamma \vdash_\alpha A \quad \Gamma, z : B \vdash_{\beta'} C}{\Gamma \vdash_\beta C}$$

The rule for $*$ (and similarly $\times$) acts on a leaf z and replaces the leaf where z occurs in the tree β with the correct bunch $x * y$. The left rule for $\mathrel{-\!\!*}$ (and similarly for $\to$) isolates a subtree containing the implication c and resources $*$'ed with it, uses those resources to prove A, and then replaces the subtree with the variable z standing for the result of the implication.

3.2 Multi-use variables

An n-use variable [24, 1, 17] is a variable that is used "exactly n times" (modulo additives), as expressed by the following sequent calculus rules for n-use functions

$$\frac{}{0 \cdot \Gamma, x :^1 P \vdash P} \qquad \frac{\Gamma, x :^n A \vdash B}{\Gamma \vdash A \to^n B} \qquad \frac{\Delta \vdash A \quad \Gamma, z :^k B \vdash C}{\Gamma + f :^k A \to^n B + (nk \cdot \Delta) \vdash C}$$

where $\Gamma + \Delta$ acts pointwise by $x :^n A + x :^m A = x :^{n+m} A$ and $n \cdot \Delta$ acts pointwise by $n \cdot x^m A = x :^{nm} A$. In the left rule, Γ and Δ have the same underlying variables and types (but potentially different counts), and $f :^k A \to^n B$ abbreviates a context with the same variables and types but 0's for all counts besides f's. The left rule says that if you spend k "uses" of a function that takes n uses of an argument, then you need nk uses of whatever you use to construct the argument, in order to get k uses of the result.

We can model this in the mode theory of a commutative monoid by using context descriptors that are themselves non-linear: we define $A \to^n B := \mathsf{U}_{c.c\odot(x^n)}(x : A \mid B)$ where $x^n := x \odot x \odot \ldots \odot x$ (n times). This has the following instances of UL and UR:

$$\frac{\Gamma, x : A \vdash_{\beta \odot x^n} B}{\Gamma \vdash_\beta A \to^n B} \qquad \frac{f : \mathsf{U}_{f.f\odot x^n}(x : A \mid B) \in \Gamma \quad \beta \Rightarrow \beta'[f \odot (\alpha)^n/z] \quad \Gamma \vdash_\alpha A \quad \Gamma, z : B \vdash_{\beta'} C}{\Gamma \vdash_\beta C}$$

In the left rule, β' must be equal to some term $\beta'' \odot z^k$ for some k and β'' not mentioning z (for this mode theory, any term is a polynomial of variables), and the only structural transformations are the commutative monoid equations, so the premise is $\beta \equiv (\beta'' \odot z^k)[f \odot (\alpha)^n/z] \equiv \beta'' \odot f^k \odot (\alpha)^{nk}$. Here β'' corresponds to the Γ in the above left rule (the resources of the continuation, besides z^k) and α corresponds to Δ. The full proof of adequacy is in the extended version:

▶ **Theorem 3.2** (Logical adequacy for n-use variables). *$x_1 :^{k_1} A_1, \ldots, x_n :^{k_n} A_n \vdash C$ iff $x_1 : A_1^*, \ldots, x_n : A_n^* \vdash_{x_1^{k_1} \odot \ldots \odot x_n^{k_n}} C^*$, where A^* translates $A \to^n B$ to $\mathsf{U}_{c.c\odot(x^n)}(x : A^* \mid B^*)$*

3.3 Comonads

Following linear-nonlinear logic [5, 6], we decompose the ! exponential of intuitionistic linear logic as the comonad of an adjunction between "linear" and "cartesian" categories. We start with two modes l (linear) and c (cartesian), along with a commutative monoid $(\otimes, 1)$ on l and a cartesian monoid $(\times, \top)$ on c. Next, we add a context descriptor from c to l $(x : \mathsf{c} \vdash \mathsf{f}(x) : \mathsf{l})$ that we think of as including a cartesian context in a linear context. This generates types $\mathsf{F}_{\mathsf{f}(x)}(x : A_\mathsf{c})\ \mathsf{type}_\mathsf{l}$ and $\mathsf{U}_{x.\mathsf{f}(x)}(\cdot \mid A_\mathsf{l})\ \mathsf{type}_\mathsf{c}$ which are adjoint $\mathsf{F}_{\mathsf{f}(x)}(x : -) \dashv \mathsf{U}_{x.\mathsf{f}(x)}(\cdot \mid -)$. The bijection on hom-sets is defined using FL and UR and their invertibility. The comonad of the adjunction $\mathsf{F}_{\mathsf{f}(x)}(x : \mathsf{U}_{c.\mathsf{f}(c)}(\cdot \mid A))$ is the linear logic $!A$.

In LNL [5], $F(A \times B) \cong F(A) \otimes F(B)$ and $F(\top) \cong 1$ (these properties of F are necessary to prove that $!A$ has weakening and contraction with respect to $\otimes$, for example), which we can add to the mode theory by equations $\mathsf{f}(x \times y) \equiv \mathsf{f}(x) \otimes \mathsf{f}(y)$ and $\mathsf{f}(\top) \equiv 1$. By Theorem 2.2, these equations induce type isomorphisms because all of $F, \otimes, \times$ are represented by F-types in our framework. For example, $F(A \times B) \vdash F(A) \otimes F(B)$ is derived as follows:

$$\dfrac{\dfrac{\mathsf{f}(y \times z) \equiv \mathsf{f}(y) \otimes \mathsf{f}(z) \quad \dfrac{}{y : A, z : B \vdash_{\mathsf{f}(y)} \mathsf{F}_{x.\mathsf{f}(x)}(x : A)}\ \mathsf{FR}^* \quad \dfrac{}{y : A, z : B \vdash_{\mathsf{f}(z)} \mathsf{F}_{x.\mathsf{f}(x)}(x : B)}\ \mathsf{FR}^*}{y : A, z : B \vdash_{\mathsf{f}(y \times z)} \mathsf{F}_{z \otimes w}(z : \mathsf{F}_{\mathsf{f}(x)}(x : A), w : \mathsf{F}_{\mathsf{f}(x)}(x : B))}\ \mathsf{FR}}{x : \mathsf{F}_{y \times z}(y : A, z : B) \vdash_{\mathsf{f}(x)} \mathsf{F}_{z \otimes w}(z : \mathsf{F}_{\mathsf{f}(x)}(x : A), w : \mathsf{F}_{\mathsf{f}(x)}(x : B))}\ \mathsf{FL}}{q : \mathsf{F}_{\mathsf{f}(\mathsf{x})}(x : \mathsf{F}_{y \times z}(y : A, z : B)) \vdash_q \mathsf{F}_{z \otimes w}(z : \mathsf{F}_{\mathsf{f}(x)}(x : A), w : \mathsf{F}_{\mathsf{f}(x)}(x : B))}\ \mathsf{FL}$$

Omitting these equations allows us to describe non-monoidal (or lax monoidal, if we add only one direction) left adjoints: in the extended version, we consider S4 $\Box$ [23, 7], and prove adequacy for it.

3.4 Monads

We model a $\Diamond\, A$ modality [7, 23] with rules

$$\frac{\Gamma \vdash A\ \mathsf{true}}{\Gamma \vdash A\ \mathsf{poss}} \qquad \frac{\Gamma \vdash A\ \mathsf{poss}}{\Gamma \vdash \Diamond\, A\ \mathsf{true}} \qquad \frac{A\ \mathsf{true} \vdash C\ \mathsf{poss}}{\Gamma, \Diamond\, A\ \mathsf{true} \vdash C\ \mathsf{poss}}$$

by a mode theory with two modes t and p and context descriptor $x : \mathsf{t} \vdash \mathsf{g}(x) : \mathsf{p}$; we define $\Diamond_{\mathsf{g}}\, A := \mathsf{U}_{c.\mathsf{g}(c)}(\cdot \mid \mathsf{F}_{\mathsf{g}(x)}(x : A))$. This is always a monad, but it does not automatically have a tensorial strength. For example, if we have a monoid $(\otimes, 1)$ on mode t and try to derive strength

$$\dfrac{\dfrac{\mathsf{g}(x \otimes y) \Rightarrow \beta'[\mathsf{g}(y)/z] \quad x : A, y : \Diamond_{\mathsf{g}}\, B, z : \mathsf{F}_{\mathsf{g}}(B) \vdash_{\beta'} \mathsf{F}_{\mathsf{g}}(A \otimes B)}{x : A, y : \Diamond_{\mathsf{g}}\, B \vdash_{\mathsf{g}(x \otimes y)} \mathsf{F}_{\mathsf{g}}(A \otimes B)}\ \mathsf{UL}}{x : A, y : \Diamond_{\mathsf{g}}\, B \vdash_{x \otimes y} \Diamond_{\mathsf{g}}\, (A \otimes B)}\ \mathsf{UR}$$

we are stuck, because there is no way to rewrite $\mathsf{g}(x \otimes y)$ as a term containing $\mathsf{g}(y)$. If $\otimes$ is affine, then we can weaken away x and take $\beta' = z$ – corresponding to the context-clearing in the left rule for $\Diamond\, A$ – but then in the right-hand premise we will only have access to z, not x, so $\Diamond$ correctly represents a non-strong monad in this setting. In the extended version, we prove adequacy for this and extend the mode theory to express strong monads.

▶ **Theorem 3.3** (Logical adequacy for a monad)**.** *We translate all types at mode* t*, representing* $\Diamond\, A$ *as above. Then* $A_1\ \mathsf{true}, \ldots, A_1\ \mathsf{true} \vdash C\ \mathsf{true}$ *iff* $x_1 : A_1^*, \ldots, x_1 : A_n^* \vdash_{x_1 \otimes \ldots \otimes x_n} C^*$*, and* $A_1\ \mathsf{true}, \ldots, A_n\ \mathsf{true} \vdash C\ \mathsf{poss}$ *iff* $x_1 : A_1^*, \ldots, x_1 : A_n^* \vdash_{\mathsf{g}(x_1 \otimes \ldots \otimes x_n)} \mathsf{F}_{\mathsf{g}}(C^*)$*. The three "native" rules above are* FR*,* UR*, and a composite of* UL *followed by* FL*, respectively.*

3.5 Spatial Type Theory

The spatial type theory for cohesion [31] which motivated this work has an adjoint pair $\flat \dashv \sharp$, where $\flat$ is a comonad and $\sharp$ is a monad, with some additional properties. In the one-variable case [15], we analyzed this as arising from an idempotent comonad[2] in the mode theory: we have a mode c with a cartesian monoid $(\times, \top)$ and a context descriptor $x : \mathsf{c} \vdash \mathsf{r}(x) : \mathsf{c}$ such that $\mathsf{r}(\mathsf{r}(x)) \equiv \mathsf{r}(x)$ and there is a directed transformation $\mathsf{r}(x) \Rightarrow x$. Then we define $\flat A := \mathsf{F}_{\mathsf{r}}(A)$ and $\sharp A := \mathsf{U}_{\mathsf{r}}(A)$. These are adjoint, and the transformation gives the counit $\mathsf{F}_{\mathsf{r}}(A) \vdash A$ and the unit $A \vdash \mathsf{U}_{\mathsf{r}}(A)$. Now that we have a multi-assumptioned logic, we can model the fact that $\flat A$ preserves products by the equational axiom $\mathsf{r}(x \times y) \equiv \mathsf{r}(x) \times \mathsf{r}(y)$. Overall, we encode a simply-typed spatial type theory judgement $x_1 : A_1\ \mathsf{crisp}, \ldots; y_1 : B_1\ \mathsf{coh}, \ldots \vdash C\ \mathsf{coh}$ as $x_1 : A_1, \ldots, y_1 : B_1, \ldots \vdash_{\mathsf{r}(x_1) \times \ldots \times y_1 \times \ldots} C$. As a sequent calculus, the rules from [31] are

$$\frac{A \in \Delta \quad \Delta; \Gamma, A \vdash C}{\Delta; \Gamma \vdash C} \qquad \frac{\Delta; \cdot \vdash A}{\Delta; \Gamma \vdash \flat\, A} \qquad \frac{\Delta, A; \Gamma \vdash C}{\Delta; \Gamma, \flat\, A \vdash C} \qquad \frac{\Delta, \Gamma; \cdot \vdash C}{\Delta; \Gamma \vdash \sharp C} \qquad \frac{\sharp\, A \in \Delta \quad \Delta; \Gamma, A \vdash C}{\Delta; \Gamma \vdash C}$$

In order, these correspond to (1) the action of the contraction and $\mathsf{r}(x) \Rightarrow x$ transformations; (2) FR with weakening, using monoidalness of r in one direction; (3) FL; (4) UR, using monoidalness of r in the other direction and idempotence; (5) UL, with contraction. This provides a satisfying explanation for the unusual features of these rules, such as promoting all cohesive variables to crisp in $\sharp$-right, and eliminating a crisp $\sharp$ in $\sharp$-left, and illustrates how our framework can be used in investigating extensions of homotopy type theory.

[2] In [15], the mode theory was actually an idempotent *monad*. The multicategorical generalization prompted changes in the variance of F and U; for example, $\mathsf{F}_\alpha(\Delta)$ must now be covariant in $\psi \vdash \alpha : r$ for the n-ary Δ to match ψ.

4 Equational Theory on Derivations

In this section we give an equational theory describing $\beta\eta$-equality of derivations. We use this equational theory in the categorical semantics below, and to reason about terms in encoded languages (for example, to prove that a pair of entailments is an isomorphism, we show that the maps compose to the identity up to these equations).

First, we need a notation for derivations of the $\alpha \Rightarrow \beta$ judgement in Figure 1. We assume names for constants are given in the signature Σ, and write 1_α for reflexivity, $s_1; s_2$ for transitivity (in diagrammatic order), and $s_1[s_2/x]$ for congruence. We extend the signature Σ to allow axioms for equality of transformations $s_1 \equiv s_2$ (for two derivations of the same judgement $s_1, s_2 :: \psi \vdash \alpha \Rightarrow_p \beta$), and define equality to be the least congruence closed under those axioms and some associativity, unit, and interchange laws, which are the 2-category axioms extended to the multicategorical case (see the extended version for details). As with equality of context descriptors, we think of all definitions as being parametrized by $\equiv$-equivalence-classes of transformations, not raw syntax.

To simplify the axiomatic description of equality, we use a notation for derivations where the admissible transformation, identity, and cut rules are internalized as explicit rules – so the calculus has the flavor of an explicit substitution one. We write proof terms for these plus the 4 U/F rules (the hypothesis rule for atoms is derivable from these) as follows:

$$\frac{}{\Gamma, x : A \vdash_x x : A} \qquad \frac{s :: \alpha \Rightarrow \beta \quad \Gamma \vdash_\beta d : A}{\Gamma \vdash_\alpha s_*(d) : A} \qquad \frac{\Gamma, x : A \vdash_\beta e : B \quad \Gamma \vdash_\alpha d : A}{\Gamma \vdash_{\beta[\alpha/x]} e[d/x] : B}$$

$$\frac{\Gamma, \Gamma', \Delta \vdash_{\beta[\alpha/x]} d : C}{\Gamma, x : \mathsf{F}_\alpha(\Delta), \Gamma' \vdash_\beta (\mathsf{split}\, \Delta \;=\; x \,\mathsf{in}\, d) : C} \qquad \frac{s :: \beta \Rightarrow \alpha[\gamma] \quad \Gamma \vdash_\gamma d_i \vec{/} x_i : \Delta}{\Gamma \vdash_\beta s_*(d_i \vec{/} x_i) : \mathsf{F}_\alpha(\Delta)}$$

$$\frac{x : \mathsf{U}_{x.\alpha}(\Delta \mid A) \in \Gamma \quad s :: \beta \Rightarrow \beta'[\alpha[\gamma]/z] \quad \Gamma \vdash_\gamma d_i \vec{/} x_i : \Delta \quad \Gamma, z : A \vdash_{\beta'} d' : C}{\Gamma \vdash_\beta s_*(\mathsf{let}\, z \;=\; x(d_i \vec{/} x_i) \,\mathsf{in}\, d) : C}$$

$$\frac{\Gamma, \Delta \vdash_{\alpha[\beta/x]} d : A}{\Gamma \vdash_\beta \lambda\Delta.d : \mathsf{U}_{x.\alpha}(\Delta \mid A)}$$

The equational theory of derivations is the least congruence containing the following equations.

$$\begin{array}{rcl} d[x/x] & \equiv & d \\ x[d/x] & \equiv & d \\ d_1[d_2/x] & \equiv & d_1 \text{ if } x \# d_1 \\ (d_1[d_2/x])[d_3/y] & \equiv & (d_1[d_3/y])[d_2[d_3/y]/x] \end{array} \qquad \begin{array}{rcl} 1_*(d) & \equiv & d \\ (s_1; s_2)_*(d) & \equiv & s_{1*}(s_{2*}(d)) \\ (s_2[s_1/x])_*(d_2[d_1/x]) & \equiv & s_{2*}(d_2)[s_{1*}(d_1)/x] \end{array}$$

$$\begin{array}{rcll} (\mathsf{split}\, \Delta \;=\; x_0 \,\mathsf{in}\, d)[s_*(d_i \vec{/} x_i)/x_0] & \equiv & (1_\beta[s/x_0])_*(d[d_i \vec{/} x_i]) & \mathsf{F}\beta \\ (s_*(\mathsf{let}\, z \;=\; x_0(\vec{d_i}/x_i) \,\mathsf{in}\, d'))[\lambda\Delta.d/x_0] & \equiv & (s[1_\alpha/x_0])_*(d'[(d[\vec{d_i}/x_i])/z]) & \mathsf{U}\beta \\ d :: \Gamma, x : \mathsf{F}_\alpha(\Delta), \Gamma' \vdash_\beta C & \equiv & \mathsf{split}\, \Delta \;=\; x \,\mathsf{in}\, d[1_*(\Delta/\Delta)/x] & \mathsf{F}\eta \\ d :: \Gamma \vdash_\beta \mathsf{U}_{x.\alpha}(\Delta \mid A) & \equiv & \lambda\Delta.(1_*(\mathsf{let}\, z \;=\; x(\Delta/\Delta) \,\mathsf{in}\, z))[d/x] & \mathsf{U}\eta \end{array}$$

In the top-left, the first two equations say that identity is a unit for cut. The third says that non-occurrence of a variable is a projection. The fourth is functoriality of cut. In the top-right, the first two rules say that the action of a transformation is functorial, and the third says that it commutes with cut. The typing in the third rule is $d_1 :: \Gamma \vdash_{\alpha'} A$ and $d_2 :: \Gamma, x : A \vdash_{\beta'} C$ and $s_1 :: \alpha \Rightarrow \alpha'$ and $s_2 :: \beta \Rightarrow \beta'$, so both sides are derivations of as derivations of $\Gamma \vdash_{\beta[\alpha/x]} C$. Finally, we have the $\beta\eta$-laws for F and U. The β laws are the principal cut cases from our cut admissibility proof. The η laws witness left-invertibility of F and right-invertibility of U.

5 Categorical Semantics

In this section, we give a category-theoretic structure corresponding to the above syntax. First, we define a cartesian 2-multicategory as a semantic analogue of the syntax in Figure 1.

▶ **Definition 5.1.** A **(strict) cartesian 2-multicategory** consists of

1. A set $\mathcal{M}_0$ of *objects*.
2. For every object B and every finite list of objects $(A_1, \ldots, A_n)$, a category $\mathcal{M}(A_1, \ldots, A_n; B)$. The objects of this category are *1-morphisms* and its morphisms are *2-morphisms*; we write composition of 2-morphisms as $s_1 \cdot s_2$.
3. For each object A, an identity arrow $1_A \in \mathcal{M}(A; A)$.
4. For any object C and lists of objects $(B_1, \ldots, B_m)$ and $(A_{i1}, \ldots, A_{in_i})$ for $1 \leq i \leq m$, a composition functor $(g, (f_1, \ldots, f_m)) \mapsto g \circ (f_1, \ldots, f_m) : \mathcal{M}(B_1, \ldots, B_m; C) \times \prod_{i=1}^{m} \mathcal{M}(A_{i1}, \ldots, A_{in_i}; B_i) \longrightarrow \mathcal{M}(A_{11}, \ldots, A_{mn_m}; C)$. We write the action of this functor on 2-cells as $d \circ_2 (e_1, \ldots, e_m)$.
5. For any function $\sigma : \{1, \ldots, m\} \to \{1, \ldots, n\}$ and objects $A_1, \ldots, A_n, B$, a *renaming* functor $f \mapsto f\sigma^* : \mathcal{M}(A_{\sigma 1}, \ldots, A_{\sigma m}; B) \to \mathcal{M}(A_1, \ldots, A_n; B)$
6. satisfying some equalities (see the extended version)

The next three definitions will be used to describe the $\Gamma \vdash_\alpha A$ judgement.

▶ **Definition 5.2.** A **functor of cartesian 2-multicategories** $\pi : \mathcal{D} \to \mathcal{M}$ consists of a function $\pi_0 : \mathcal{D}_0 \to \mathcal{M}_0$ and functors $\mathcal{D}(A_1, \ldots, A_n; B) \to \mathcal{M}(\pi_0(A_1), \ldots, \pi_0(A_n); \pi_0(B))$ such that the chosen identities, compositions, and renamings are preserved (strictly). Given a functor π, we write $\mathcal{D}_\alpha(A_1, \ldots, A_n; B)$ for the fiber over $\alpha \in \mathcal{M}(\pi A_1, \ldots, \pi A_n; \pi B)$.

▶ **Definition 5.3.** A functor of cartesian 2-multicategories $\pi : \mathcal{D} \to \mathcal{M}$ is a **local discrete fibration** if each induced functor of ordinary categories $\mathcal{D}(A_1, \ldots, A_n; B) \to \mathcal{M}(\pi A_1, \ldots, \pi A_n; \pi B)$ is a discrete fibration. When π is a local discrete fibration, each fiber is a discrete set.

▶ **Definition 5.4.** If $\pi : \mathcal{D} \to \mathcal{M}$ is a local discrete fibration, then a morphism $\xi \in \mathcal{D}(A_1, \ldots, A_n; B)$ is **opcartesian** if all diagrams of the left-hand form are pullbacks of categories, and a morphism $\xi \in \mathcal{D}(\vec{C}, B, \vec{D}; E)$ is **cartesian at** B if all diagrams of the right-hand form are pullbacks of categories:

$$\begin{array}{ccc} \mathcal{D}(\vec{C}, B, \vec{D}; E) & \xrightarrow{(-)\circ_B \xi} & \mathcal{D}(\vec{C}, \vec{A}, \vec{D}; E) \\ \downarrow{\scriptstyle \pi} & & \downarrow{\scriptstyle \pi} \\ \mathcal{M}(\pi\vec{C}, \pi B, \pi\vec{D}; \pi E) & \xrightarrow[(-)\circ_{\pi B} \pi\xi]{} & \mathcal{M}(\pi\vec{C}, \pi\vec{A}, \pi\vec{D}; \pi E) \end{array} \qquad \begin{array}{ccc} \mathcal{D}(\vec{A}; B) & \xrightarrow{\xi \circ_B (-)} & \mathcal{D}(\vec{C}, \vec{A}, \vec{D}; E) \\ \downarrow{\scriptstyle \pi} & & \downarrow{\scriptstyle \pi} \\ \mathcal{M}(\pi\vec{A}; \pi B) & \xrightarrow[\pi\xi \circ_{\pi B} (-)]{} & \mathcal{D}(\pi\vec{C}, \pi\vec{A}, \pi\vec{D}; \pi E) \end{array}$$

Given $\mu : (p_1, \ldots, p_n) \to q$ in $\mathcal{M}$, we say that π **has μ-products** if for any A_i with $\pi A_i = p_i$, there exists a B with $\pi B = q$ and an opcartesian morphism in $\mathcal{D}_\mu(A_1, \ldots, A_n; B)$. Dually, we say π **has μ-homs** if for any i, any B with $\pi B = q$, and any A_j with $\pi A_j = p_j$ for $j \neq i$, there exists an A_i with $\pi A_i = p_i$ and a cartesian morphism in $\mathcal{D}_\mu(A_1, \ldots, A_n; B)$. We say that π is an **opfibration** if it has μ-products for all μ, a **fibration** if it has μ-homs for all μ, and a **bifibration** if it is both an opfibration and a fibration.

The proofs of our soundness and completeness results are described in Appendix A:

▶ **Theorem 5.5** (Mode theory presents a multicategory)**.** *A mode theory Σ presents a cartesian 2-multicategory $\mathcal{M}$, where $\mathcal{M}_0$ is the set of modes, and an object of $\mathcal{M}(p_1, \ldots, p_n; q)$ is a term $x_1 : p_1, \ldots, x_n : p_n \vdash \alpha : q$ and a morphism of $\mathcal{M}(p_1, \ldots, p_n; q)$ is a structural transformation $s :: \psi \vdash \alpha \Rightarrow_q \beta$, both considered modulo $\equiv$.*

▶ **Theorem 5.6** (Completeness/Syntactic Bifibration). *For a fixed mode theory $\mathcal{M}$, the syntax presents a bifibration $\pi : \mathcal{D} \to \mathcal{M}$, where:*

- *Objects of $\mathcal{D}$ are pairs $(p, A\ \mathsf{type}_p)$;*
- *1-morphisms $\Gamma \to B$, i.e., objects of $\mathcal{D}(\Gamma; B)$, are pairs $(\alpha, d :: \Gamma \vdash_\alpha B)$ (up to $\equiv$);*
- *2-morphisms $(\alpha, d) \to (\alpha', d')$ are structural transformations $s :: \alpha \Rightarrow \alpha'$ such that $s_*(d') \equiv d$;*
- *the μ-products are F-types, and the μ-homs are U-types.*

The functor $\pi : \mathcal{D} \to \mathcal{M}$ is given by first projection on objects and 1-morphisms, and sends 2-morphisms to the underlying structural transformations.

▶ **Theorem 5.7** (Soundness/Interpretation in any bifibration). *Fix a bifibration $\pi : \mathcal{D} \to \mathcal{M}$. Then there is a function $[\![-]\!]$ from types $A\ \mathsf{type}_p$ to $[\![A]\!] \in \mathcal{D}_0$ with $\pi([\![A]\!]) = p$ and from $\equiv$-classes of derivations $x : A_1, \ldots, x_n : A_n \vdash_\alpha C$ to morphisms $d \in \mathcal{D}([\![A_1]\!], \ldots, [\![A_n]\!]; [\![C]\!])$, such that $\pi(d) = \alpha$.*

6 Related and Future Work

We have described a sequent calculus that can express a variety of substructural and modal logics through a suitable choice of mode theory. Our framework builds on many approaches to substructural and modal logic in the literature. Logical rules that act at a leaf of a tree-structured context go back to the Lambek calculus [14]. A rich collection of context structures that correspond to type constructors plays a central role in display logic [4]. The λ-calculus for resource separation [3] is similar to mode theories with one mode, where there is at most one 2-cell between a given pair of 1-cells; at the logical level, our calculus is a unification of this with multimodal adjoint logic [26]. Algebras of resources play a central role in semantics of substructural and modal logics [28] and in their encodings in first-order logic [27], and resources on variables are used to track modalities in Agda's implementation [1] and in linear dependent types [17]. LF representations of modal or substructural logics work by restricting the use of cartesian variables [9]. Relative to all of these approaches, we believe that the analysis of the context structures/resources as a *term* in a base type theory, and the fibrational structure of the derivations over them, is a new and useful observation. For example, rather than needing extra-logical conditions on proof rules to ensure cut admissibility, as in display logic, the conditions are encoded in the language of context descriptors and the definition of types from them. Moreover, none of these existing approaches allow for proof-relevant 2-cells/structural rules, and their presence (and the equational theory we give for them) is important for our applications to homotopy type theory.

A point of contrast with substructural logical frameworks [8, 34, 25] is that logics are "embedded" in our calculus (giving a type translation such that provability in the object logic corresponds to provability in ours), rather than "encoding" the structure of derivations. This way, we obtain cut elimination for object languages as a corollary of framework cut elimination.

Bifibrations have also been used recently to model refinement types and logical relations [18, 19, 11]. Superficially, this seems to be a different use of the same semantic structure, because the base and domain categories of the bifibration play a different role. Here, the base mode theory describes different categories of types and functors (type constructors) between between them, and the domain represents types and terms; whereas in refinement types/logical relations, the base category represents types and terms, and the domain represents a further

notion of predicate/specification. It would be interesting to investigate deeper connections and combine the two notions.

One direction for future work is to continue a preliminary investigation of equational adequacy that is discussed in the extended version, investigating whether the logical adequacy proofs are an isomorphism on $\beta\eta$-classes of derivations – or, phrased semantically, that a bifibration over the mode theory is the usual notion of categorical model for a particular logic (e.g. a bifibration over the ordered logic mode theory presents the free monoidal category). It is generally easy to show that object-language equations are true in the framework. We conjecture that the converse is true for the mode theories we have described here, which says that the "extra" types and judgements available in the framework do not add to the equations between terms in the image of encoded sequents. Proving this is challenging because the equational theory of Section 4 does not itself obviously have the subformula property. We have sketched a proof of equational adequacy for a simple case (ordered logic products), assuming a lemma that the equational theory from Section 4 can be characterized by permuting conversions on cut-free derivations. A related avenue for improvement is that our adequacy proofs require reasoning about the 1- and 2-cells in the mode theory, which we have currently done entirely naïvely, but could possibly benefit from higher-dimensional rewriting techniques.

Additionally, we plan to apply our framework to investigate more extensions of homotopy type theory, such as an internal language for parametrized spectra, and an extension of spatial type theory to differential cohesion [29]. We also plan to consider encodings of programming-focused type theories, such as specialized effect calculi.

A final direction for future work is to extend our framework with first-order quantifiers, structured conclusions (as in classical or display logic), and dependent types, which all seem possible but not obvious. Scaling to dependent types will require more worked examples to understand the patterns of substructural and modal type dependency that a framework should capture.

References

1. Andreas Abel. The next 700 modal type assignment systems. In *International Conference on Types for Proofs and Programs (TYPES)*, 2015.
2. Thorsten Altenkirch and Ambrus Kaposi. Type theory in type theory using quotient inductive types. In *ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, 2016.
3. Robert Atkey. A λ-calculus for resource separation. In *Automata, Languages and Programming: 31st International Colloquium, ICALP 2004*, volume 3142 of *Lecture Notes in Computer Science*, pages 158–170. Springer, 2004.
4. Nuel D. Belnap Jr. Display logic. *Journal of Philosophical Logic*, 11:375–417, 1982.
5. Nick Benton. A mixed linear and non-linear logic: Proofs, terms and models. In *Computer Science Logic*, volume 933 of *LNCS*. Springer-Verlag, 1995.
6. Nick Benton and Philip Wadler. Linear logic, monads and the lambda calculus. In *IEEE Symposium on Logic in Computer Science*. IEEE Computer Society Press, 1996.
7. G. Bierman and V.C.V. de Paiva. On an intuitionistic modal logic. *Studia Logica*, 65:383–416, 2000.
8. Iliano Cervesato and Frank Pfenning. A linear logical framework. *Information and Computation*, 179(1):19–75, 2002.
9. Karl Crary. Higher-order representation of substructural logics. In *ACM SIGPLAN International Conference on Functional Programming*, pages 131–142. ACM, 2010.

10 Vincent Danos, Jean-Baptiste Joinet, and Harold Schellinx. The structure of exponentials: Uncovering the dynamics of linear logic proofs. In *Kurt Godel Colloquium*, LNCS, pages 159–171. Springer, 1993.

11 Neil Ghani, Patricia Johann, Fredrik Nordvall Forsberg, Federico Orsanigo, and Tim Revell. Bifibrational functorial semantics for parametric polymorphism. In *Proceedings of Mathematical Foundations of Program Semantics*, 2015.

12 Claudio Hermida. Fibrations for abstract multicategories. Fields Institute Communications; available from `http://sqig.math.ist.utl.pt/pub/HermidaC/fib-mul.pdf`, 2002.

13 Fritz Hörmann. Fibered multiderivators and (co)homological descent. Available from `http://arxiv.org/abs/1505.00974`, 2015.

14 Joachim Lambek. The mathematics of sentence structure. *The American Mathematical Monthly*, 65:154–170, 1958.

15 Daniel R. Licata and Michael Shulman. Adjoint logic with a 2-category of modes. In *Logical Foundations of Computer Science*, 2016.

16 Daniel R. Licata, Michael Shulman, and Mitchell Riley. A fibrational framework for substructural and modal logics (extended version). Available from `http://dlicata.web.wesleyan.edu/`, 2017.

17 Conor McBride. I got plenty o' nuttin'. *A List of Successes That Can Change the World: Essays Dedicated to Philip Wadler on the Occasion of His 60th Birthday*, pages 207–233, 2016.

18 Paul-André Melliès and Noam Zeilberger. Functors are type refinement systems. In *ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, 2015.

19 Paul-André Melliès and Noam Zeilberger. A bifibrational reconstruction of lawvere's presheaf hyperdoctrine. In *IEEE Symposium on Logic in Computer Science*, 2016.

20 Vivek Nigam and Dale Miller. Algorithmic specifications in linear logic with subexponentials. In *Principles and Practice of Declarative Programming*, 2009.

21 Peter W. O'Hearn and David J. Pym. The logic of bunched implications. *Bulletin of Symbolic Logic*, 5(2):215–244, 1999.

22 Frank Pfenning. A structural proof of cut elimination and its representation in a logical framework. Technical Report CMU-CS-94-218, Department of Computer Science, Carnegie Mellon University, 1994.

23 Frank Pfenning and Rowan Davies. A judgmental reconstruction of modal logic. *Mathematical Structures in Computer Science*, 11:511–540, 2001.

24 Jason Reed. Names are (mostly) useless: Encoding nominal logic programming techniques with use-counting and dependent types. Talk at Working on Mechanizing Metatheory (WMM), 2008.

25 Jason Reed. *A Hybrid Logical Framework*. PhD thesis, Carnegie Mellon University, 2009.

26 Jason Reed. A judgemental deconstruction of modal logic. Available from `www.cs.cmu.edu/~jcreed/papers/jdml.pdf`, 2009.

27 Jason Reed and Frank Pfenning. A constructive approach to the resource semantics of substructural logics. Available from `http://www.cs.cmu.edu/~jcreed/papers/rp-substruct.pdf`, 2009.

28 Greg Restall. *An introduction to substructural logics*. Routledge, 2000.

29 Urs Schreiber. Differential cohomology in a cohesive infinity-topos. *arXiv*, 2013. `arXiv:1310.7930`.

30 Urs Schreiber and Michael Shulman. Quantum gauge field theory in cohesive homotopy type theory. In *Workshop on Quantum Physics and Logic*, 2012.

31 Michael Shulman. Brouwer's fixed-point theorem in real-cohesive homotopy type theory. arXiv:1509.07584, 2015.

32 Univalent Foundations Program. *Homotopy Type Theory: Univalent Foundations Of Mathematics*. Available from `homotopytypetheory.org/book`, 2013.

33 Vladimir Voevodsky. A very short note on homotopy λ-calculus. `http://www.math.ias.edu/vladimir/files/2006_09_Hlambda.pdf`, September 2006.

34 Kevin Watkins, Iliano Cervesato, Frank Pfenning, and David Walker. A concurrent logical framework I: Judgments and properties. Technical Report CMU-CS-02-101, Department of Computer Science, Carnegie Mellon University, 2002. Revised May 2003.

A Technical Appendix: Categorical Semantics

To illustrate the connection between the syntax and the semantics, we sketch the proofs of soundness and completeness; full details are available in the extended version.

Proof of Theorem 5.6. Our goal is to construct a bifibration $\pi : \mathcal{D} \to \mathcal{M}$ from the syntax.

First, a mode theory Σ presents a cartesian 2-multicategory $\mathcal{M}$, where $\mathcal{M}_0$ is the set of modes, and an object of $\mathcal{M}(p_1, \ldots, p_n; q)$ is a term $x_1 : p_1, \ldots, x_n : p_n \vdash \alpha : q$ and a morphism of $\mathcal{M}(p_1, \ldots, p_n; q)$ is a structural transformation $s :: \psi \vdash \alpha \Rightarrow_q \beta$, both considered modulo $\equiv$.

Next, we construct the domain category $\mathcal{D}$ as indicated in the theorem statement: an object is a pair (p, A) where A is a type of mode p; a 1-cell $(\psi, \Gamma) \to (p, A)$ is a pair (α, d) where $\psi \vdash \alpha : p$ and d is a derivation of $\Gamma \vdash_\alpha A$; and a 2-cell $(\alpha, d) \Rightarrow (\alpha', d')$ is a structural transformation $s :: \alpha \Rightarrow \alpha'$ such that $s_*(d') \equiv d$. We write just A and d and s for objects and 1-cells and 2-cells, leaving the underlying modes and mode morphisms implicit, and we also omit variable names from sequents.

Composition of 1-morphisms is defined by iterating cut: given $g :: (x_1 : B_1, \ldots, x_m : B_m \vdash_\alpha C)$ and $f_i :: (A_{i1}, \ldots, A_{in_i} \vdash_{\beta_i} B_i)$ we set

$$g \circ (f_1, \ldots, f_n) := (\alpha[\beta_1/x_1, \ldots, \beta_m], g[f_1/x_1, \ldots, f_m/x_m])$$

That the latter derivation lies over $\alpha[\beta_1/x_1, \ldots, \beta_m]$ follows from the cut and weakening principles.

For the action of these composition functors on 2-morphisms, suppose we are given 1-morphisms

$$\begin{aligned}
&d :: x_1 : B_1, \ldots, x_m : B_m \vdash_\alpha C \\
&d' :: x_1 : B_1, \ldots, x_m : B_m \vdash_{\alpha'} C \\
&e_i :: A_{i1}, \ldots, A_{in_i} \vdash_{\beta_i} B_i \\
&e'_i :: A_{i1}, \ldots, A_{in_i} \vdash_{\beta'_i} B_i
\end{aligned}$$

and 2-morphisms $S : (\alpha, d) \Rightarrow (\alpha', d')$ and $T_i : (\beta_i, e_i) \to (\beta'_i, e'_i)$ such that S has underlying transformation $s :: \alpha \Rightarrow \alpha'$ and the T_i have underlying transformations $t_i :: \beta_i \Rightarrow \beta'_i$ respectively. This means that $d \equiv s_*(d')$ and $e_i \equiv (t_i)_*(e'_i)$ for all i. The composite $S \circ_2 (T_1, \ldots, T_m)$ is the 2-morphism given by the underlying transformation $s[t_1/x_1, \ldots, t_m/x_m]$. This is a valid 2-morphism $d[e_1/x_1, \ldots, e_m/x_m] \Rightarrow d'[e'_1/x_1, \ldots, e'_m/x_m]$ because

$$\begin{aligned}
&(s[t_1/x_1, \ldots, t_m/x_m])_*(d'[e'_1/x_1, \ldots, e'_m/x_m]) \\
\equiv &(s[t_1/x_1, \ldots, t_{m-1}/x_{m-1}])_*(d'[e'_1/x_1, \ldots, e'_{m-1}m/x_{m-1}])[(t_m)_*(e'_m)/x_m] \\
&\vdots \\
\equiv &s_*(d')[(t_1)_*(e'_1)/x_1, \ldots, (t_m)_*(e'_m)/x_m] \\
\equiv &d[e_1/x_1, \ldots, e_m/x_m]
\end{aligned}$$

as required, where we have repeatedly applied the equation

$$(s_2[s_1/x])_*(d_2[d_1/x]) \equiv s_{2*}(d_2)[s_{1*}(d_1)/x]\,.$$

The unit and associativity laws for 1-morphisms in $\mathcal{D}$ follow from the first set of equations for derivations, and from the definition of multi-variable substitution as iterated cut. For 2-morphisms, they follow as composition of 2-morphisms is simply composition of the underlying transformations in the mode theory.

The cartesian structure in $\mathcal{D}$ is given by the admissible rules for weakening-over-weakening, exchange-over-exchange and contraction-over-contraction, from which all renamings can be made. These rules also all preserve the underlying mode morphisms in the correct way to make π functorial.

The next step is to show that π is a local discrete fibration. Suppose we have a context Γ and object B. We must show that the functor $\pi : \mathcal{D}(\Gamma; B) \to \mathcal{M}(\pi\Gamma; \pi B)$ is a discrete fibration. Let $\alpha, \alpha' \in \mathcal{M}(\pi\Gamma; \pi B)$ be mode morphisms and suppose we have a transformation $s :: \alpha \Rightarrow \alpha'$ between them. Any 2-morphism in $\mathcal{D}(\Gamma; B)$ lying over s must clearly have s as the underlying transformation. Given a lift $d' :: \Gamma \vdash_{\alpha'} B$ of α', then we can consider s as a 2-morphism $(\alpha, s_*(d')) \Rightarrow (\alpha', d')$ over s, whose domain is the action of s on d', $s_*(d')$, as expected. The equational condition $s_*(d) \equiv s_*(d)$ is trivially satisfied, and in fact forces $s_*(d)$ as the only possible choice of domain, so the lift is unique. So π is a local discrete fibration.

We now show that π is an opfibration, i.e., has α-homs for all mode morphisms $\psi \vdash \alpha : q$. Suppose we have lifts for the modes in ψ, i.e., a context Δ with $\pi\Delta = \psi$. We define the opcartesian lift of α to be $\mathsf{FR}^* :: \Delta \vdash_\alpha \mathsf{F}_\alpha(\Delta)$, the generating map from Δ to $\mathsf{F}_\alpha(\Delta)$ that lives over α, which is an instance of the F right rule where both premises are the identity. To verify that this is an opcartesian morphism, we must show that all squares of the form

$$\begin{array}{ccc} \mathcal{D}(\Gamma, \mathsf{F}_\alpha(\Delta), \Gamma'; C) & \xrightarrow{-[\mathsf{FR}^*/x_0]} & \mathcal{D}(\Gamma, \Delta, \Gamma'; C) \\ \downarrow{\scriptstyle \pi} & & \downarrow{\scriptstyle \pi} \\ \mathcal{M}(\pi\Gamma, q, \pi\Gamma'; \pi C) & \xrightarrow[-[\alpha/x_0]]{} & \mathcal{M}(\pi\Gamma, \psi, \pi\Gamma'; \pi C) \end{array}$$

are pullbacks of categories. For this we will use the following characterisation: a diagram of categories

$$\begin{array}{ccc} \mathcal{A} & \xrightarrow{H} & \mathcal{B} \\ {\scriptstyle K}\downarrow & & \downarrow{\scriptstyle F} \\ \mathcal{C} & \xrightarrow[G]{} & \mathcal{D} \end{array}$$

is a pullback diagram iff

- For every pair of objects $b \in \mathcal{B}$ and $c \in \mathcal{C}$ with $Fb = Gc$, there is a unique object $a \in \mathcal{A}$ such that $Ha = b$ and $Ka = c$; and,
- For every pair of morphisms $f \in \mathcal{B}(b, b')$ and $g \in \mathcal{C}(c, c')$ with $Fb = Gc$ and $Fb' = Gc'$ and $Ff = Gg$, there is a unique morphism $\theta \in \mathcal{A}$ such that $H\theta = f$ and $K\theta = g$. The domain and codomain of θ are fixed by the previous property.

First, we show the property for objects. Suppose we have an object $d \in \mathcal{D}(\Gamma, \Delta, \Gamma'; C)$ and $\beta \in \mathcal{M}(\pi\Gamma, q, \pi\Gamma'; \pi C)$ such that $\pi(d) = \beta[\alpha/x_0]$. This simply states that d is of the form $d :: \Gamma, \Delta, \Gamma' \vdash_{\beta[\alpha/x_0]} C$. We must produce a unique object $e \in \mathcal{D}(\Gamma, \mathsf{F}_\alpha(\Delta), \Gamma'; C)$ such

that $\pi(e) = \beta$ and $e[\mathsf{FR}^*/x_0] \equiv d$. We take as our e the derivation $\mathsf{split}\,\Delta \;=\; x_0 \,\mathsf{in}\, d$. This lies over β, and we calculate

$$\begin{aligned} e[\mathsf{FR}^*/x_0] &= (\mathsf{split}\,\Delta \;=\; x_0 \,\mathsf{in}\, d)[1_{\alpha *}(x/x)/x_0] \\ &\equiv (1_\beta[1_\alpha/x_0])_*(d[\vec{x/x}]) \\ &\equiv (1_{\beta[\alpha/x_0]})_*(d) \\ &\equiv d \end{aligned}$$

by the β-law for F and unit laws. It remains to show uniqueness. Suppose we have some derivation e' such that $\pi(e') = \beta$ and $e'[\mathsf{FR}^*/x_0] \equiv d$. By the η-law for F, we have

$$e' \equiv \mathsf{split}\,\Delta \;=\; x_0 \,\mathsf{in}\, e'[\mathsf{FR}^*/x_0] \equiv \mathsf{split}\,\Delta \;=\; x_0 \,\mathsf{in}\, d = e$$

as required.

We now turn to the pullback property for morphisms. Let $\beta, \beta' \in \mathcal{M}(\pi\Gamma, q, \pi\Gamma'; \pi C)$ and let $s :: \beta \Rightarrow \beta'$ be a morphism. Further suppose that we have derivations $d :: \Gamma, \Delta, \Gamma' \vdash_{\beta[\alpha/x_0]} C$ and $d' :: \Gamma, \Delta, \Gamma' \vdash_{\beta'[\alpha/x_0]} C$ such that $(s[1_\alpha/x_0])_*(d') \equiv d$. This describes a morphism $T : d \Rightarrow d'$ in $\mathcal{D}(\Gamma, \mathsf{F}_\alpha(\Delta), \Gamma'; C)$ that lies over $s[1_\alpha/x_0]$. This latter transformation is the result of applying the functor $-[\alpha/x_0]$ to s.

We now must find a morphism S in $\mathcal{D}(\Gamma, \Delta, \Gamma'; C)$ that lies over s, and such that the functor $-[\mathsf{FR}^*/x_0]$ applied to the morphism S yields T. We know that for S to lie over s, its underlying structural transformation must be s. The action of $-[\mathsf{FR}^*/x_0]$ on S then takes s to $s[1_\alpha/x_0]$ as expected.

By the previous argument for objects, we know that S must have domain $\mathsf{split}\,\Delta \;=\; x_0 \,\mathsf{in}\, d$ and codomain $\mathsf{split}\,\Delta \;=\; x_0 \,\mathsf{in}\, d'$. We can verify that choosing the underlying transformation s gives a well-defined 2-morphism $S : (\mathsf{split}\,\Delta \;=\; x_0 \,\mathsf{in}\, d) \Rightarrow (\mathsf{split}\,\Delta \;=\; x_0 \,\mathsf{in}\, d')$:

$$\begin{aligned} s_*(\mathsf{split}\,\Delta \;=\; x_0 \,\mathsf{in}\, d') &\equiv \mathsf{split}\,\Delta \;=\; x_0 \,\mathsf{in}\, s_*(\mathsf{split}\,\Delta \;=\; x_0 \,\mathsf{in}\, d')[\mathsf{FR}^*/x_0] \\ &\equiv \mathsf{split}\,\Delta \;=\; x_0 \,\mathsf{in}\, s_*(\mathsf{split}\,\Delta \;=\; x_0 \,\mathsf{in}\, d')[(1_\alpha)_*(\mathsf{FR}^*)/x_0] \\ &\equiv \mathsf{split}\,\Delta \;=\; x_0 \,\mathsf{in}\, (s[1_\alpha/x_0])_*(\mathsf{split}\,\Delta \;=\; x_0 \,\mathsf{in}\, d'[\mathsf{FR}^*/x_0]) \\ &\equiv \mathsf{split}\,\Delta \;=\; x_0 \,\mathsf{in}\, (s[1_\alpha/x_0])_*(d') \\ &\equiv \mathsf{split}\,\Delta \;=\; x_0 \,\mathsf{in}\, d \end{aligned}$$

where we have used the η-law followed by the β-law.

We conclude that all squares of the given form are pullback squares, and so every α has an opcartesian lift. Therefore π is an opfibration. The proof that π is also a fibration is very similar, using U types instead of F types. ◀

Proof of Theorem 5.7. Conversely, we show some cases of the interpretation of the syntax in any bifibration π over the 2-multicategory $\mathcal{M}$ determined by the mode theory.

Since π is a local discrete fibration, the 2-cells of $\mathcal{M}$ act on the fibers. Suppose $\psi \vdash \alpha, \beta : p$ and $s : \alpha \Rightarrow \beta$. We re-use the notation s_* for the induced function (of sets) $\mathcal{D}_\beta(\Gamma; A) \to \mathcal{D}_\alpha(\Gamma; A)$ that sends an object $d \in \mathcal{D}_\alpha(\Gamma; A)$ to the domain of the unique lift of s with codomain d.

The definition of an opfibration of 2-multicategories guarantees that, given a morphism in the mode category $\psi \vdash \alpha : q$ and a set of objects Δ that lies over ψ, there is an opcartesian morphism over α with domain Δ. For each α we choose one such lift and take the codomain of this morphism as our interpretation of $\mathsf{F}_\alpha(\Delta)$. Let us name this opcartesian lift $\zeta_{\alpha,\Delta} : \Delta \to \mathsf{F}_\alpha(\Delta)$. ζ corresponds to the axiomatic FR^*.

We assume a given interpretation of each atomic proposition $[\![P\ \mathsf{type}_p]\!]$ as an object of $\mathcal{D}$ that lies over p. The sequent calculus rules are interpreted as follows (we elide semantic brackets on objects):

- The identity derivation of a sequent $x :: \Gamma \vdash_x A$ is defined to be $[\![x]\!] = 1_A$.
- Given a derivation $d :: \Gamma \vdash_\beta A$ and transformation $s :: \beta' \Rightarrow \beta$, the respect-for-transformations derivation is interpreted as $[\![s_*(d)]\!] = s_*([\![d]\!])$.
- For $d_1 :: \Gamma, x : A, \Gamma' \vdash_\alpha B$ and $d_2 :: \Gamma, \Gamma' \vdash_\beta A$, cut is interpreted as $[\![d_1[d_2/x]]\!] = [\![d_1]\!] \circ_A [\![d_1]\!]$ (writing $e \circ_A f$ for a one-place composition derived from the n-place multicategory composition).
- For FL

$$\frac{\Gamma, \Gamma', \Delta \vdash_{\beta[\alpha/x]} C}{\Gamma, x : \mathsf{F}_\alpha(\Delta), \Gamma' \vdash_\beta M : C}\ \mathsf{FL}$$

  the inductive hypothesis (after an exchange, which preserves the size of the derivations) gives a morphism $[\![d]\!] \in \mathcal{D}_{\beta[\alpha/x]}(\Gamma, \Delta, \Gamma'; C)$ and we must produce a morphism $\mathcal{D}_\beta(\Gamma, \mathsf{F}_\alpha(\Delta), \Gamma'; C)$. By the opcartesian-ness of $\zeta_{\alpha,\Delta}$, the following square is a pullback:

$$\begin{array}{ccc} \mathcal{D}(\Gamma, \mathsf{F}_\alpha(\Delta), \Gamma'; C) & \xrightarrow{(-)\circ\zeta_{\alpha,\Delta}} & \mathcal{D}(\Gamma, \Delta, \Gamma'; C) \\ \downarrow{\scriptstyle \pi} & & \downarrow{\scriptstyle \pi} \\ \mathcal{M}(\pi\Gamma, \pi\mathsf{F}_\alpha(\Delta), \pi\Gamma'; \pi C) & \xrightarrow[(-)\circ\alpha]{} & \mathcal{M}(\pi\Gamma, \pi\Delta, \pi\Gamma'; \pi C) \end{array}$$

  We are given an object of the bottom left (β) and the top right ($[\![d]\!]$), with $\pi[\![d]\!] = \beta \circ_{\pi\mathsf{F}_\alpha(\Delta)} \alpha$. By the above characterization of pullbacks of categories, there is a unique object $[\![d]\!]^{\mathsf{F}}_{\alpha,\Delta} \in \mathcal{D}(\Gamma, \mathsf{F}_\alpha(\Delta), \Gamma'; C)$ so that $\pi([\![d]\!]^{\mathsf{F}}_{\alpha,\Delta}) = \beta$. We take this object to be our interpretation.
- For FR

$$\frac{s : \beta \Rightarrow \alpha[\gamma] \quad \Gamma \vdash_\gamma M : \Delta}{\Gamma \vdash_\beta \mathsf{F}_\alpha(\Delta)}\ \mathsf{FR}$$

  where $\gamma = (\alpha_1, \ldots, \alpha_n)$ and $\Delta = (C_1, \ldots, C_n)$, the first premise is a 2-cell $s : \beta \Rightarrow \alpha \circ (\alpha_1, \ldots, \alpha_n)$, and the second is interpreted as a set of morphisms $[\![d_i]\!] \in \mathcal{D}_{\alpha_i}(\Gamma; C_i)$. We take the interpretation of the conclusion to be $s_*(\zeta_{\alpha,\Delta} \circ ([\![d_1]\!], \ldots, [\![d_n]\!]))$

What remains is to check that the above interpretation function respects the equational theory on derivations (see the extended version). ◀