Beyond Monads and Biproducts: A Uniform Interpretation of Parallelism in Intuitionistic Logic

Alejandro Díaz-Caro
Université de Lorraine, CNRS, Inria, LORIA, Nancy, France
Universidad Nacional de Quilmes, Buenos Aires, Argentina

Octavio Malherbe
IMERL, Facultad de Ingeniería, Universidad de la República, Montevideo, Uruguay

Abstract

Traditional approaches to modelling parallelism and algebraic structure in lambda calculi often rely on monads – as in Moggi's framework – or on rich categorical structures such as biproducts – as used in certain models of linear logic. In this work, we propose a minimal alternative that captures both parallelism and weighted parallelism (linear combinations) within the setting of intuitionistic propositional logic, without resorting to monads or assuming the existence of biproducts.

We introduce two lambda calculi: a parallel lambda calculus and an algebraic lambda calculus, both extending full propositional intuitionistic logic. Their semantics are given in two categories: $\mathbf{Mag}_{\mathbf{Set}}$, whose objects are magmas and arrows are functions in **Set**; and $\mathbf{AMag}^{\mathcal{S}}_{\mathbf{Set}}$, whose objects are action magmas.

The key technical challenge addressed is the interpretation of disjunction in the presence of parallel and algebraic operators. Since the usual coproduct structure is unavailable in our minimal setting, we propose a novel set-theoretic interpretation based on the union of the disjoint union and the Cartesian product. This allows for the construction of sound and adequate models for both calculi.

Our results offer a unified and structurally lightweight framework for modelling parallelism and algebraic effects in intuitionistic logic, opening the way to alternatives beyond the traditional monadic or linear logic approaches.

2012 ACM Subject Classification Theory of computation → Proof theory; Theory of computation → Type theory; Theory of computation → Categorical semantics; Theory of computation → Denotational semantics

Keywords and phrases Algebraic lambda calculus, Categorical semantics, Disjunction, Proof theory

Digital Object Identifier 10.4230/LIPIcs.FSTTCS.2025.28

Related Version *Full version with appendices*: `arXiv:2408.16102` [10]

Funding Supported by the European Union through the MSCA SE project QCOMICAL (Grant Agreement ID: 101182521) and by the Uruguayan CSIC grant 22520220100073UD.
Alejandro Díaz-Caro: Also supported by the Plan France 2030 through the PEPR integrated project EPiQ (ANR-21-PETQ-0007).

1 Introduction

Motivation. In his 1991 paper [18], Eugenio Moggi introduced a uniform framework to handle effectful computations in the lambda calculus, using monads to encapsulate various computational effects such as non-determinism, state, exceptions, and input/output. This monadic approach has since been widely adopted in functional programming languages, providing a powerful abstraction for reasoning about side effects. In the context of intuitionistic propositional logic – which corresponds categorically to cartesian closed categories – non-determinism can be captured, for instance, in the category **Set** by the map

$\chi : A \times A \to \mathcal{P}A$ defined as $(x, y) \mapsto \{x, y\}$. This non-deterministic operator is sometimes referred to as a *parallel operator*, as it can be seen as preserving all possible outputs in parallel.

While this monadic framework is highly versatile, it does not extend seamlessly to all logical systems. In particular, within Intuitionistic Linear Logic (ILL), the monadic encoding of non-determinism becomes problematic: the map χ is not linear. For instance, $\chi(x_1 + x_2, y_1 + y_2) = \{x_1 + y_1, x_2 + y_2\}$, whereas $\chi(x_1, y_1) + \chi(x_2, y_2) = \{x_1, y_1\} + \{x_2, y_2\}$, which does not yield the same set under any reasonable definition of set addition.

An alternative approach, suitable for linear settings, was developed through a line of work on relational models. In particular, in [4], building on previous work developed in Manzonetto's doctoral thesis [16] and in [3], and in the context of untyped λ-calculus models, the hom-sets of the category **MRel**, restricted to a reflexive object, were enriched with a commutative semiring structure in order to model a parallel and non-deterministic λ-calculus. In the typed setting, this approach was pursued in weighted relational models of typed λ-calculi [14], where the effect is captured by the map $\chi' : A \times A \to A$ defined by $(x, y) \mapsto x + y$. In [11], this idea was extended into a more general categorical characterisation, relying on the presence of biproducts to define the sum.

The guiding question of the present work is the following: *Can we construct an alternative to Moggi's monad, inspired by the use of biproducts, that works in the context of propositional logic, where biproducts are not available?* We show that the answer is affirmative by providing a concrete model that captures this effect without requiring the full structure of a biproduct.

Our proposal is a structurally simple alternative capable of capturing both parallelism and weighted parallelism (or linear combinations). We move from the category **Set** to a richer setting based on magmas, where we define an ad hoc operation that mimics – to some extent – the behaviour of a biproduct, though it is not a biproduct in the categorical sense.

More precisely, we introduce the category $\mathbf{Mag}_{\mathbf{Set}}$, whose objects are magmas with no required algebraic properties, and whose arrows are simply functions from the category **Set**. In addition, we define the category $\mathbf{AMag}^{\mathcal{S}}_{\mathbf{Set}}$, whose objects are *action magmas* – magmas equipped with an external scalar action – and whose arrows are again functions from **Set**. These constructions allow us to interpret the parallel lambda calculus and the algebraic lambda calculus, respectively, in a way reminiscent of the interpretation of the parallel linear lambda calculus in the category of vector spaces, but without requiring the existence of biproducts.

A key difficulty in adapting these techniques to intuitionistic propositional logic lies in the treatment of disjunction. In contrast to models of linear logic equipped with biproducts – where additive conjunction and disjunction are both interpreted via this common structure – the absence of a unified product-coproduct structure in propositional logic means that disjunction cannot be handled in the same way as conjunction. This asymmetry poses significant challenges in modelling the behaviour of parallel or algebraic combinations involving disjunctive terms. In this work, we address this difficulty directly by proposing a concrete interpretation of disjunction that preserves the intuition of additive structure, while remaining compatible with the categorical limitations of the setting.

Antecedents. Several foundational works have explored extensions of the lambda calculus to express superposition, linear combinations, and parallelism. The Lineal calculus [1] was introduced to represent quantum programs by incorporating the principle that, since data and programs are unified in the lambda calculus, the superposition of data implies the superposition of programs. This idea was formalised through a linear structure supporting algebraic combinations of terms.

An independent development is the Algebraic Lambda Calculus [19], a simplification of the Differential Lambda Calculus [13], which emphasises algebraic structure rather than operational semantics. Later, Lineal and the Algebraic Lambda Calculus were shown to be closely related [2].

Several semantic models have addressed non-determinism and parallelism. In [7], a filter model was proposed for a concurrent lambda calculus, where terms are interpreted as sets of possible outcomes. A relational model capturing non-deterministic and parallel behaviour was introduced in [5], and a general framework using weighted relations, allowing for deterministic, non-deterministic, and probabilistic computation, was proposed in [14].

More recently, the $\odot$-connective was introduced in [8] and extended in [9] to express quantum superpositions and measurements within a logical framework. These systems include additive structure in proof terms, such as sums and scalar products, aiming to represent quantum behaviour. A categorical interpretation for such calculi in a linear setting – using biproducts in the category of vector spaces – was developed in [11].

Our perspective. While Moggi's monadic approach provides a general and robust framework for handling computational effects in intuitionistic settings, our goal is to explore whether the structural techniques developed in the linear logic context – particularly those relying on biproducts – can be adapted to intuitionistic propositional logic. Instead of using a monad to encapsulate non-determinism or parallelism, we investigate how to reconstruct these effects using categorical structures that simulate the role of biproducts, even in their absence. In this way, we aim to propose a conceptually unified alternative for modelling such effects across both linear and non-linear logical frameworks.

Plan of the paper and contributions. In this paper, we introduce two lambda calculi for propositional logic – one modelling parallelism and the other algebraic combinations – based on minimal categorical structure inspired by biproducts, and avoiding the use of monads.

- In Section 2, we introduce the parallel lambda calculus $\lambda_{\parallel}$ for the full intuitionistic propositional logic. It is mostly based on the in-left-right-+-calculus (inlr-calculus for short) [12], with a slight modification. In the inlr-calculus the parallel of $\mathsf{inl}(t)$ and $\mathsf{inr}(u)$ reduces to a new term $\mathsf{inlr}(t,u)$. In our presentation we just keep $\mathsf{inl}(t) \parallel \mathsf{inr}(u)$, however, the two calculi are equivalent. This calculus follows the approach of [6] for the structure operator, where the parallel operator is neither idempotent, commutative, nor associative. Section 2.1 gives the syntax, deduction, and reduction rules. Section 2.2 restate the correctness properties of the calculus, proved in [12] and adapted to our setting in [10, Appendix A]. Section 2.3 introduces the category $\mathbf{Mag_{Set}}$, which is the category whose objects are magmas and whose arrows are functions from the **Set** category. We also introduce the product $\oplus$ and prove several necessary properties. Section 2.4 provides the interpretation of $\lambda_{\parallel}$ within the category $\mathbf{Mag_{Set}}$, and prove its soundness and adequacy results, that is, if one term reduces to another, they are interpreted by the same arrow, and if two closed proof-terms are interpreted by the same arrows, then they are computationally equivalent.
- In Section 3, we introduce the algebraic lambda calculus $\lambda_{\mathsf{alg}}^{\mathcal{S}}$ for the full intuitionistic propositional logic. This calculus is an extension of the $\lambda_{\parallel}$ calculus with scalars and follows the approach of the Algebraic lambda calculus [19].
  Section 3.1 gives the syntax, deduction, and reduction rules. Section 3.2 states the correctness properties of the calculus as straightforward adaptations of those from Section 2.2. Section 3.3 introduces the concept of action magmas and define the category

$$\frac{}{x:A,\Gamma\vdash x:A}\text{ax}\qquad\frac{\Gamma\vdash t:A\quad\Gamma\vdash u:A}{\Gamma\vdash t\parallel u:A}\text{par}\qquad\frac{}{\Gamma\vdash\star:\top}\top_i\qquad\frac{\Gamma\vdash t:\top\quad\Gamma\vdash u:C}{\Gamma\vdash\delta_\top(t,u):C}\top_e$$

$$\frac{\Gamma\vdash t:\bot}{\Gamma\vdash\delta_\bot(t):C}\bot_e\qquad\frac{x:A,\Gamma\vdash t:B}{\Gamma\vdash\lambda x.t:A\Rightarrow B}\Rightarrow_i\qquad\frac{\Gamma\vdash t:A\Rightarrow B\quad\Gamma\vdash u:A}{\Gamma\vdash t\ u:B}\Rightarrow_e$$

$$\frac{\Gamma\vdash t:A\quad\Gamma\vdash u:B}{\Gamma\vdash\langle t,u\rangle:A\wedge B}\wedge_i\qquad\frac{\Gamma\vdash t:A\wedge B}{\Gamma\vdash\pi_1(t):A}\wedge_{e1}\qquad\frac{\Gamma\vdash t:A\wedge B}{\Gamma\vdash\pi_2(t):B}\wedge_{e2}$$

$$\frac{\Gamma\vdash t:A}{\Gamma\vdash\mathsf{inl}(t):A\vee B}\vee_{i1}\qquad\frac{\Gamma\vdash t:B}{\Gamma\vdash\mathsf{inr}(t):A\vee B}\vee_{i2}$$

$$\frac{\Gamma\vdash t:A\vee B\quad x:A,\Gamma\vdash u:C\quad y:B,\Gamma\vdash v:C}{\Gamma\vdash\delta_\vee(t,x.u,y.v):C}\vee_e$$

Figure 1 The deduction rules of the $\lambda_\parallel$.

$\mathbf{AMag}^{\mathcal{S}}_{\mathbf{Set}}$, whose objects are action magmas, and whose arrows are functions from the **Set** category. We also extend the product $\oplus$ into an action magma. Section 3.4 provides the interpretation of the algebraic lambda calculus within the category $\mathbf{AMag}^{\mathcal{S}}_{\mathbf{Set}}$, and prove its soundness and adequacy.

- In Section 4, we present some concluding remarks.

2 The calculus $\lambda_\parallel$

2.1 Syntax, deduction, and reduction rules

The connectives in this logic are those found in standard intuitionistic propositional logic: $\top$, $\bot$, $\Rightarrow$, $\wedge$, and $\vee$. The syntax of the proof-terms are also standard, with the addition of the parallel construction as mentioned in the introduction.

$$\begin{array}{rll} t = & x \mid t \parallel t & \\ & \mid \star \mid \delta_\top(t,t) & (\top) \\ & \mid \delta_\bot(t) & (\bot) \\ & \mid \lambda x.t \mid t\ t & (\Rightarrow) \\ & \mid \langle t,t\rangle \mid \pi_1(t) \mid \pi_2(t) & (\wedge) \\ & \mid \mathsf{inl}(t) \mid \mathsf{inr}(t) \mid \delta_\vee(t,x.t,y.t) & (\vee) \end{array}$$

where x and y range over a countable set of variables.

The terms $\star$, $\lambda x.t$, $\langle t,u\rangle$, $\mathsf{inl}(t)$, and $\mathsf{inr}(t)$ are called introductions. The terms $\delta_\top(t,u)$, $\delta_\bot(t)$, $\pi_1(t)$, $\pi_2(t)$, and $\delta_\vee(t,x.u,y.v)$ are called eliminations. Variables and $t\parallel u$ are neither introductions nor eliminations, except for $\mathsf{inl}(t)\parallel\mathsf{inr}(u)$ which can be considered as an introduction. Free variables are defined as usual, and the substitution of x by u in t is written $(u/x)t$, with the usual renaming to avoid variable captures.

Our notation is inspired by logic. However, $\delta_\top(t,r)$ could also be written $t;r$ to denote sequencing, $\delta_\bot(t)$ as $\mathsf{error}(t)$ to denote raising an exception, $\delta_\vee(t,x.u,y.v)$ as $\mathsf{case}\ t\ \mathsf{of}\ \{\mathsf{inl}(x)\to u;\mathsf{inr}(y)\to v\}$ for pattern matching, and $\pi_1(t)$ and $\pi_2(t)$ as $\mathsf{fst}(t)$ and $\mathsf{snd}(t)$ for projections. From this viewpoint, the calculus can be seen as a programming language.

The also standard deduction rules are given in Figure 1.

The reduction relation is the contextual closure of the relation defined by the rules:

$$\delta_\top(\star, t) \longrightarrow t \qquad (1)$$
$$(\lambda x.t)\ u \longrightarrow (u/x)t \qquad (2)$$
$$\pi_1\langle t, u\rangle \longrightarrow t \qquad (3)$$
$$\pi_2\langle t, u\rangle \longrightarrow u \qquad (4)$$
$$\delta_\vee(\mathsf{inl}(t), x.v, y.w) \longrightarrow (t/x)v \qquad (5)$$
$$\delta_\vee(\mathsf{inr}(u), x.v, y.w) \longrightarrow (u/y)w \qquad (6)$$
$$\delta_\vee(\mathsf{inl}(t) \parallel \mathsf{inr}(u), x.v, y.w) \longrightarrow (t/x)v \parallel (u/y)w \qquad (7)$$

$$\star \parallel \star \longrightarrow \star \qquad (8)$$
$$(\lambda x.t) \parallel (\lambda x.u) \longrightarrow \lambda x.(t \parallel u) \qquad (9)$$
$$\langle t_1, u_1\rangle \parallel \langle t_2, u_2\rangle \longrightarrow \langle t_1 \parallel t_2, u_1 \parallel u_2\rangle \qquad (10)$$
$$\mathsf{inl}(t_1) \parallel \mathsf{inl}(t_2) \longrightarrow \mathsf{inl}(t_1 \parallel t_2) \qquad (11)$$
$$\mathsf{inl}(t_1) \parallel (\mathsf{inl}(t_2) \parallel \mathsf{inr}(u_1)) \longrightarrow \mathsf{inl}(t_1 \parallel t_2) \parallel \mathsf{inr}(u_1) \qquad (12)$$
$$\mathsf{inr}(u_1) \parallel \mathsf{inl}(t_1) \longrightarrow \mathsf{inl}(t_1) \parallel \mathsf{inr}(u_1) \qquad (13)$$
$$\mathsf{inr}(u_1) \parallel \mathsf{inr}(u_2) \longrightarrow \mathsf{inr}(u_1 \parallel u_2) \qquad (14)$$
$$\mathsf{inr}(u_1) \parallel (\mathsf{inl}(t_1) \parallel \mathsf{inr}(u_2)) \longrightarrow \mathsf{inl}(t_1) \parallel \mathsf{inr}(u_1 \parallel u_2) \qquad (15)$$
$$(\mathsf{inl}(t_1) \parallel \mathsf{inr}(u_1)) \parallel \mathsf{inl}(t_2) \longrightarrow \mathsf{inl}(t_1 \parallel t_2) \parallel \mathsf{inr}(u_1) \qquad (16)$$
$$(\mathsf{inl}(t_1) \parallel \mathsf{inr}(u_1)) \parallel \mathsf{inr}(u_2) \longrightarrow \mathsf{inl}(t_1) \parallel \mathsf{inr}(u_1 \parallel u_2) \qquad (17)$$
$$(\mathsf{inl}(t_1) \parallel \mathsf{inr}(u_1)) \parallel (\mathsf{inl}(t_2) \parallel \mathsf{inr}(u_2)) \longrightarrow \mathsf{inl}(t_1 \parallel t_2) \parallel \mathsf{inr}(u_1 \parallel u_2) \qquad (18)$$

The first group of reduction rules is standard, except for the last rule, which allows reducing inl and inr in parallel. The second group establishes the commutation of the parallel construction with all the connectives. The parallel construction commutes with introductions, as seen in the first three rules of the second group for the cases of $\top$, $\Rightarrow$, and $\wedge$. The case of $\vee$ is left to the next eight rules since multiple cases must be considered due to the fact that there are two normal forms for disjunctions, and disjunction is neither commutative nor associative in Natural Deduction.

Instead of commuting the parallel construction with the introduction of disjunction, we could have considered doing so with its elimination, using the rule $\delta_\vee(t \parallel u, x.v, y.w) \longrightarrow \delta_\vee(t, x.u, y.w) \parallel \delta_\vee(u, x.v, y.w)$, which can also be proven valid in our model (the proof of soundness for this rule is given in [10, Appendix C]). However, we chose to consider commutation with the introduction, as it leads to a better introduction property: a closed irreducible proof of $\top$ is $\star$, a closed irreducible proof of an implication is a lambda abstraction, a closed irreducible proof of a conjunction is a pair, and a closed irreducible proof of a disjunction is either inl, inr, or the parallel of both (see Theorem 2.4).

Along this paper if R is a relation, we write R^* for its transitive-reflexive closure.

2.2 Correctness

The correctness properties of this section have been proven in a preprint for the inlr calculus [12], and the proofs are the same for the $\lambda_\parallel$ calculus.

▶ **Theorem 2.1** (Subject reduction [12, Appendix A.1]). *If $\Gamma \vdash t : A$ and $t \longrightarrow u$, then $\Gamma \vdash u : A$.*

▶ **Theorem 2.2** (Strong normalisation [12, Appendix A.4]). *If $\Gamma \vdash t : A$ then t is strongly normalising.*

▶ **Theorem 2.3** (Confluence). *The $\lambda_\parallel$ calculus is confluent.*

Proof. The relation $\longrightarrow$ from Section 2.1 is left linear and has no critical pairs. And, as proved in [17, Theorem 6.8], higher-order left linear systems without critical pairs are confluent. ◀

▶ **Theorem 2.4** (Introduction property [12, Appendix A.3]). *Let $\vdash t : A$ be irreducible. Then: If $A = \top$, $t = \star$. A is not $\bot$. If $A = B \Rightarrow C$, $t = \lambda x.u$. If $A = B \wedge C$, $t = \langle u, v\rangle$. If $A = B \vee C$, $t = \mathsf{inl}(u)$, $t = \mathsf{inr}(v)$, or $t = \mathsf{inl}(u) \parallel \mathsf{inr}(v)$.*

2.3 The category $\mathbf{Mag}_{\mathbf{Set}}$

▶ **Definition 2.5** (The category $\mathbf{Mag}_{\mathbf{Set}}$). *The category $\mathbf{Mag}_{\mathbf{Set}}$ is determined by the following data: Objects are magmas $(A, *_A)$. Arrows are maps in $\mathbf{Set}$.*

▶ Remark 2.6. Note that $\mathbf{Mag}_{\mathbf{Set}}$ is not an algebraic category, in the sense that the natural arrows for the algebraic structure (such as homomorphisms) are not considered. Instead, the objects are plain magmas, without any algebraic property, and the arrows are simply functions in $\mathbf{Set}$, with composition and identities inherited from $\mathbf{Set}$. The reasons for this choice are discussed in detail in Section 4.

The next definitions give some objects and arrows used later.

▶ **Definition 2.7.** *Let A and B be objects in $\mathbf{Mag}_{\mathbf{Set}}$, we define the following objects.*

- *$\emptyset$ with the empty map as its operation.*
- *$\{\star\}$ with the operation defined as $\star *_{\{\star\}} \star = \star$.*
- *$A \times B$ with the operation defined as $(a_1, b_1) *_{A\times B} (a_2, b_2) = (a_1 *_A a_2, b_1 *_B b_2)$.*
- *$[A \to B]$ with the operation defined as $(f *_{[A\to B]} g)(a) = f(a) *_B g(a)$.*

▶ **Theorem 2.8** (CCC). *The category $\mathbf{Mag}_{\mathbf{Set}}$ is Cartesian closed.*

Proof. We need to prove that there is an adjunction $A \times \_ \dashv [A \to \_]$ in $\mathbf{Mag}_{\mathbf{Set}}$. For all A, B objects in $\mathbf{Mag}_{\mathbf{Set}}$, we have $Hom_{\mathbf{Mag}_{\mathbf{Set}}}(A, B) = Hom_{\mathbf{Set}}(A, B)$. Since, $A \times B$ and $[B \to C]$ are objects in $\mathbf{Mag}_{\mathbf{Set}}$, by the adjunction in $\mathbf{Set}$, we have $Hom_{\mathbf{Mag}_{\mathbf{Set}}}(A \times B, C) = Hom_{\mathbf{Set}}(A \times B, C) \simeq Hom_{\mathbf{Set}}(A, [B \to C]) = Hom_{\mathbf{Mag}_{\mathbf{Set}}}(A, [B \to C])$. ◀

As usual, the disjoint union of two sets A and B is denoted by $A \uplus B$ and is defined as $A \uplus B = (A \times \{0\}) \cup (B \times \{1\})$ where 0 and 1 are distinct elements that do not belong to either to A or B. If we want $A \uplus B$ to be an object in $\mathbf{Mag}_{\mathbf{Set}}$, we must define its magma operation. For example, we can choose $(a_1, 0) *_\uplus (a_2, 0) = (a_1 *_A a_2, 0)$, and similarly $(b_1, 1) *_\uplus (b_2, 1) = (b_1 *_B b_2, 1)$. However, it is not clear how to define $(a, 0) *_\uplus (b, 1)$. We could choose to define it as either $(a, 0)$ or $(b, 1)$, but this would break the symmetry. This is the same problem encountered in $\mathsf{inl}(t) \parallel \mathsf{inr}(u)$. Indeed, in the sequent $\vdash \mathsf{inl}(t) \parallel \mathsf{inr}(u) : A \vee B$, we are given both a proof of A and a proof of B. However, to produce a proof of $A \vee B$, we would typically need to discard one of them, whereas with the parallel operator, we can retain both. This is why we need to define a new operation, which we call pokeball product and denote by $\oplus$.

▶ **Definition 2.9** (The pokeball product). *We define $A \oplus B = (A \uplus B) \cup (A \times B)$.*

▶ **Definition 2.10** (The operation $*_\oplus$)**.** *Let* $(A, *_A), (B, *_B) \in \mathsf{Obj}(\mathbf{Mag_{Set}})$*. We define* $*_{A\oplus B}$ *(*$*_\oplus$ *for short) as* $(A \oplus B) \times (A \oplus B) \xrightarrow{*_\oplus} A \oplus B$ *given by*

$$(c_1, c_2) \mapsto \begin{cases} (a_1 *_A a_2, 0) & \textit{if } c_1 = (a_1, 0) \textit{ and } c_2 = (a_2, 0) \\ (a, b) & \textit{if } c_1 = (a, 0) \textit{ and } c_2 = (b, 1) \\ (a_1 *_A a_2, b) & \textit{if } c_1 = (a_1, 0) \textit{ and } c_2 = (a_2, b) \\ (a, b) & \textit{if } c_1 = (b, 1) \textit{ and } c_2 = (a, 0) \\ (b_1 *_B b_2, 1) & \textit{if } c_1 = (b_1, 1) \textit{ and } c_2 = (b_2, 1) \\ (a, b_1 *_B b_2) & \textit{if } c_1 = (b_1, 1) \textit{ and } c_2 = (a, b_2) \\ (a_1 *_A a_2, b) & \textit{if } c_1 = (a_1, b) \textit{ and } c_2 = (a_2, 0) \\ (a, b_1 *_B b_2) & \textit{if } c_1 = (a, b_1) \textit{ and } c_2 = (b_2, 1) \\ (a_1 *_A a_2, b_1 *_B b_2) & \textit{if } c_1 = (a_1, b_1) \textit{ and } c_2 = (a_2, b_2) \end{cases}$$

▶ **Definition 2.11.** *Let* $A \xrightarrow{f} A'$ *and* $B \xrightarrow{g} B'$*. The arrow* $A \oplus B \xrightarrow{f \oplus g} A' \oplus B'$ *is defined as follows.*

$$c \mapsto \begin{cases} (f(a), 0) & \textit{if } c = (a, 0) \\ (g(b), 1) & \textit{if } c = (b, 1) \\ (f(a), g(b)) & \textit{if } c = (a, b) \end{cases}$$

The pokeball product entails a bifunctorial functor.

▶ **Lemma 2.12** (Bifunctoriality)**.** *Let* $A \xrightarrow{f} A'$*,* $A \xrightarrow{f'} A'$*,* $B \xrightarrow{g} B'$*, and* $B \xrightarrow{g'} B'$*. Then* $A \oplus \_$ *is a functor, and it is bifunctorial, that is* $(f \oplus g) \circ (f' \oplus g') = (f \circ f') \oplus (g \circ g')$*.*

Proof.

- We prove that $A \oplus \_$ is a functor. That is, we need to prove the commutation of the following diagrams.

$$\begin{tikzcd} A \oplus B \arrow[r, "\mathsf{id} \oplus g"] \arrow[dr, "\mathsf{id} \oplus (h \circ g)"'] & A \oplus B' \arrow[d, "\mathsf{id} \oplus h"] \\ & A \oplus B'' \end{tikzcd} \qquad \begin{tikzcd} A \oplus B \arrow[r, shift left, "\mathsf{id}_A \oplus \mathsf{id}_B"] \arrow[r, shift right, "\mathsf{id}_{A \oplus B}"'] & A \oplus B \end{tikzcd}$$

We prove the diagrams by giving their evaluations.

$$\begin{tikzcd} (a,0) \arrow[r, mapsto, "\mathsf{id} \oplus g"] \arrow[dr, mapsto, "\mathsf{id} \oplus (h \circ g)"'] & (a,0) \arrow[d, mapsto, "\mathsf{id} \oplus h"] \\ & (a,0) \end{tikzcd} \qquad \begin{tikzcd} (b,1) \arrow[r, mapsto, "\mathsf{id} \oplus g"] \arrow[dr, mapsto, "\mathsf{id} \oplus (h \circ g)"'] & (g(b),1) \arrow[d, mapsto, "\mathsf{id} \oplus h"] \\ & (h(g(b)),1) \end{tikzcd} \qquad \begin{tikzcd} (a,b) \arrow[r, mapsto, "\mathsf{id} \oplus g"] \arrow[dr, mapsto, "\mathsf{id} \oplus (h \circ g)"'] & (a,g(b)) \arrow[d, mapsto, "\mathsf{id} \oplus h"] \\ & (a,h(g(b))) \end{tikzcd}$$

$$\begin{tikzcd} (a,0) \arrow[r, mapsto, shift left, "\mathsf{id}_A \oplus \mathsf{id}_B"] \arrow[r, mapsto, shift right, "\mathsf{id}_{A \oplus B}"'] & (a,0) \end{tikzcd} \qquad \begin{tikzcd} (b,1) \arrow[r, mapsto, shift left, "\mathsf{id}_A \oplus \mathsf{id}_B"] \arrow[r, mapsto, shift right, "\mathsf{id}_{A \oplus B}"'] & (b,1) \end{tikzcd} \qquad \begin{tikzcd} (a,b) \arrow[r, mapsto, shift left, "\mathsf{id}_A \oplus \mathsf{id}_B"] \arrow[r, mapsto, shift right, "\mathsf{id}_{A \oplus B}"'] & (a,b) \end{tikzcd}$$

The proof of $\_ \oplus B$ is also analogous.

- To prove that it is bifunctorial it suffices to prove that the following diagram commutes [15, Proposition 3.1].

$$\begin{tikzcd} A \oplus B \arrow[r, "\mathsf{id} \oplus g"] \arrow[d, "f \oplus \mathsf{id}"'] & A \oplus B' \arrow[d, "f \oplus \mathsf{id}"] \\ A' \oplus B \arrow[r, "\mathsf{id} \oplus g"'] & A' \oplus B' \end{tikzcd}$$

We prove this diagram by giving its possible evaluations.

$$\begin{tikzcd}
(a,0) \arrow[r, mapsto, "\mathsf{id}\oplus g"] \arrow[d, mapsto, "f\oplus\mathsf{id}"'] & (a,0) \arrow[d, mapsto, "f\oplus\mathsf{id}"] \\
(f(a),0) \arrow[r, mapsto, "\mathsf{id}\oplus g"'] & (f(a),0)
\end{tikzcd}
\qquad
\begin{tikzcd}
(b,1) \arrow[r, mapsto, "\mathsf{id}\oplus g"] \arrow[d, mapsto, "f\oplus\mathsf{id}"'] & (g(b),1) \arrow[d, mapsto, "f\oplus\mathsf{id}"] \\
(b,1) \arrow[r, mapsto, "\mathsf{id}\oplus g"'] & (g(b),1)
\end{tikzcd}
\qquad
\begin{tikzcd}
(a,b) \arrow[r, mapsto, "\mathsf{id}\oplus g"] \arrow[d, mapsto, "f\oplus\mathsf{id}"'] & (a,g(b)) \arrow[d, mapsto, "f\oplus\mathsf{id}"] \\
(f(a),b) \arrow[r, mapsto, "\mathsf{id}\oplus g"'] & (f(a),g(b))
\end{tikzcd}$$

◀

While the pokeball product will not be a coproduct in $\mathbf{Mag}_{\mathbf{Set}}$ (see Remark 2.17), we can define some maps that will allow us to interpret the disjunction with the pokeball product in a manner similar to a coproduct. First, we note that the usual injections i_1 and i_2 are magma homomorphisms.

▶ **Lemma 2.13.** *The maps $A \xrightarrow{i_1} A \oplus B$ and $B \xrightarrow{i_2} A \oplus B$ are homomorphisms.*

Proof.
- $i_1(a_1) *_{\oplus} i_1(a_2) = (a_1, 0) *_{\oplus} (a_2, 0) = (a_1 *_A a_2, 0) = i_1(a_1 *_A a_2)$
- $i_2(b_1) *_{\oplus} i_2(b_2) = (b_1, 1) *_{\oplus} (b_2, 1) = (b_1 *_B b_2, 1) = i_2(b_1 *_B b_2)$ ◀

We can define the inclusion $A \times B \xrightarrow{\text{inc}} A \oplus B$ by $(a, b) \mapsto (a, b)$. An interesting property is that after two injections and the operation $*_{\oplus}$, we can recover the original element, as stated by the following lemma.

▶ **Lemma 2.14.** $*_{\oplus} \circ (i_1 \times i_2) = \text{inc}$.

Proof. Let $a \in A$ and $b \in B$. Then $*_{\oplus} \circ (i_1 \times i_2)(a, b) = (a, 0) *_{\oplus} (b, 1) = (a, b)$. ◀

▶ **Definition 2.15.** *Let $A \xrightarrow{f} C$ and $B \xrightarrow{g} C$ be arrows in $\mathbf{Mag}_{\mathbf{Set}}$. We define the arrow $A \oplus B \xrightarrow{[f,g]} C$ as follows.*

$$c \mapsto \begin{cases} f(a) & \text{if } c = (a, 0) \\ g(b) & \text{if } c = (b, 1) \\ f(a) *_C g(b) & \text{if } c = (a, b) \end{cases}$$

▶ **Lemma 2.16** (Weak coproduct)**.** *Then map $[f, g]$ makes the following diagram commute*

$$\begin{tikzcd}
A \arrow[r, "i_1"] \arrow[dr, "f"'] & A \oplus B \arrow[d, "{[f,g]}"] & B \arrow[l, "i_2"'] \arrow[dl, "g"] \\
& C &
\end{tikzcd}$$

Proof.
- Left triangle: $[f, g](i_1(a)) = [f, g](a, 0) = f(a)$.
- Right triangle: $[f, g](i_2(b)) = [f, g](b, 1) = g(b)$. ◀

▶ Remark 2.17 (Not a coproduct). Lemma 2.16 does not imply that $A \oplus B$ is a coproduct in $\mathbf{Mag}_{\mathbf{Set}}$, since the map $[f, g]$ is clearly not unique.

However, in the category $\mathbf{Mag}$ of magmas, whose arrows are homomorphism (which is not Cartesian closed, and so it is not a good candidate for us), the map $[f, g]$ is indeed a coproduct, since it is easy to prove that it is an homomorphism, and its unicity can be proven using the homomorphism property as follows.

Let $h : A \oplus B \to C \in \mathsf{Arr}(\mathbf{Mag})$ such that $h(i_1(a)) = h(a, 0) = f(a)$ and $h(i_2(b)) = h(b, 1) = g(b)$. Then, $h(a, b) = h((a, 0) *_{\oplus} (b, 1)) = h(a, 0) *_C h(b, 1) = f(a) *_C g(b)$. Therefore, if arrows are homomorphisms, $h = [f, g]$.

▶ **Definition 2.18.** *Let A and B be objects in $\mathbf{Mag}_{\mathbf{Set}}$, we define the following arrows.*

- $A \xrightarrow{!} \{\star\}$ *defined by* $a \mapsto \star$.
- $A \xrightarrow{\Delta} A \times A$ *defined by* $a \mapsto (a, a)$.
- $(A \oplus B) \times C \xrightarrow{d} (A \times C) \oplus (B \times C)$ *defined by*

$$(e, c) \mapsto \begin{cases} ((a, c), 0) & \text{if } e = (a, 0) \\ ((b, c), 1) & \text{if } e = (b, 1) \\ ((a, c), (b, c)) & \text{if } e = (a, b) \end{cases}$$

- $A \times A \xrightarrow{*} A$ *defined by* $(a_1, a_2) \mapsto a_1 *_A a_2$.
- $\delta = (\mathsf{id} \times \sigma \times \mathsf{id}) \circ (\mathsf{id} \times \Delta)$. *That is,* $A \times B \times C \xrightarrow{\delta} A \times C \times B \times C$.

2.4 Model of the $\lambda_{\parallel}$ calculus

2.4.1 Interpretation

The interpretation of the propositions and contexts of the proof system is the following:

Interpretation of propositions	Interpretation of contexts
$[\![\top]\!] = \{\star\}$	$[\![\emptyset]\!] = \{\star\}$
$[\![\bot]\!] = \emptyset$	$[\![x : A, \Gamma]\!] = [\![A]\!] \times [\![\Gamma]\!]$
$[\![A \Rightarrow B]\!] = [[\![A]\!] \to [\![B]\!]]$	
$[\![A \wedge B]\!] = [\![A]\!] \times [\![B]\!]$	
$[\![A \vee B]\!] = [\![A]\!] \oplus [\![B]\!]$	

The interpretation of the deduction rules is the following We refer to each deduction rule by its last sequent, since the system is syntax directed.

$$[\![x : A, \Gamma \vdash x : A]\!] = [\![A]\!] \times [\![\Gamma]\!] \xrightarrow{\pi_1} [\![A]\!]$$

$$[\![\Gamma \vdash t \parallel u : A]\!] = [\![\Gamma]\!] \xrightarrow{\Delta} [\![\Gamma]\!] \times [\![\Gamma]\!] \xrightarrow{t \times u} [\![A]\!] \times [\![A]\!] \xrightarrow{*_{[\![A]\!]}} [\![A]\!]$$

$$[\![\Gamma \vdash \star : \top]\!] = [\![\Gamma]\!] \xrightarrow{!} \{\star\} = [\![\top]\!]$$

$$[\![\Gamma \vdash \delta_\top(t, u) : C]\!] = [\![\Gamma]\!] \xrightarrow{\Delta} [\![\Gamma]\!] \times [\![\Gamma]\!] \xrightarrow{t \times u} \{\star\} \times [\![C]\!] \xrightarrow{\pi_2} [\![C]\!]$$

$$[\![\Gamma \vdash \delta_\bot(t) : C]\!] = [\![\Gamma]\!] \xrightarrow{t} \emptyset \xrightarrow{\emptyset} [\![C]\!]$$

$$[\![\Gamma \vdash \lambda x.t : A \Rightarrow B]\!] = [\![\Gamma]\!] \xrightarrow{\eta^{[\![A]\!]}} [[\![A]\!] \to [\![A]\!] \times [\![\Gamma]\!]] \xrightarrow{[[\![A]\!] \to t]} [[\![A]\!] \to [\![B]\!]]$$

$$[\![\Gamma \vdash t\ u : B]\!] = [\![\Gamma]\!] \xrightarrow{\Delta} [\![\Gamma]\!] \times [\![\Gamma]\!] \xrightarrow{t \times u} [[\![A]\!] \to [\![B]\!]] \times [\![A]\!] \xrightarrow{\varepsilon} [\![B]\!]$$

$$[\![\Gamma \vdash \langle t, u\rangle : A \wedge B]\!] = [\![\Gamma]\!] \xrightarrow{\Delta} [\![\Gamma]\!] \times [\![\Gamma]\!] \xrightarrow{t \times u} [\![A]\!] \times [\![B]\!]$$

$$[\![\Gamma \vdash \pi_1(t) : A]\!] = [\![\Gamma]\!] \xrightarrow{t} [\![A \wedge B]\!] = [\![A]\!] \times [\![B]\!] \xrightarrow{\pi_1} [\![A]\!]$$

$$[\![\Gamma \vdash \pi_2(t) : A]\!] = [\![\Gamma]\!] \xrightarrow{t} [\![A \wedge B]\!] = [\![A]\!] \times [\![B]\!] \xrightarrow{\pi_2} [\![B]\!]$$

$$[\![\Gamma \vdash \mathsf{inl}(t) : A \vee B]\!] = [\![\Gamma]\!] \xrightarrow{t} [\![A]\!] \xrightarrow{i_1} [\![A]\!] \oplus [\![B]\!]$$

$$[\![\Gamma \vdash \mathsf{inr}(t) : A \vee B]\!] = [\![\Gamma]\!] \xrightarrow{t} [\![B]\!] \xrightarrow{i_2} [\![A]\!] \oplus [\![B]\!]$$

$$\begin{aligned}[\![\Gamma \vdash \delta_\vee(t, x.u, y.v) : C]\!] &= [\![\Gamma]\!] \xrightarrow{\Delta} [\![\Gamma]\!] \times [\![\Gamma]\!] \xrightarrow{t \times \mathsf{id}} ([\![A]\!] \oplus [\![B]\!]) \times [\![\Gamma]\!] \\ &\xrightarrow{d} ([\![A]\!] \times [\![\Gamma]\!]) \oplus ([\![B]\!] \times [\![\Gamma]\!]) \xrightarrow{[u,v]} [\![C]\!]\end{aligned}$$

In some diagrams we may write A instead of $[\![A]\!]$, to simplify the notation.

2.4.2 Soundness

The soundness of the model is proven in Theorem 2.20. As usual, we require the following substitution lemma. Its proof is done by induction on t and given in [10, Appendix B.1].

▶ **Lemma 2.19** (Substitution). *If* $x : A, \Gamma \vdash t : B$ *and* $\Gamma \vdash u : A$, *then* $[\![\Gamma \vdash (u/x)t : B]\!] = [\![\Gamma \vdash t : B]\!] \circ ([\![\Gamma \vdash u : A]\!] \times \mathsf{id}) \circ \Delta$.

▶ **Theorem 2.20** (Soundness). *If* $t \longrightarrow r$ *and* $\Gamma \vdash t : A$, *then* $[\![\Gamma \vdash t : A]\!] = [\![\Gamma \vdash r : A]\!]$.

Proof. By induction on the relation $\longrightarrow$. The full proof is given in [10, Appendix B.2]. We only show two interesting cases here, involving the parallel operator.

- Rule $\delta_\vee(\mathsf{inl}(t) \parallel \mathsf{inr}(u), x.v, y.w) \longrightarrow (t/x)v \parallel (u/y)w$. The commuting diagram is

$$\begin{tikzcd}
\Gamma \times \Gamma \arrow[rrrr, "\Delta \times \mathsf{id}"] \arrow[drr, dashed, "\Delta \times \Delta"] \arrow[dddd, "(t/x)v \times (u/y)w"'] & & & & (\Gamma \times \Gamma) \times \Gamma \arrow[dll, dashed, "\mathsf{id} \times \Delta"'] \arrow[dddd, "(t \times u) \times \mathsf{id}"] \\
& & (\Gamma \times \Gamma) \times (\Gamma \times \Gamma) \arrow[d, dashed, "\mathsf{id} \times \sigma \times \mathsf{id}"] \arrow[ddd, dashed, bend right=60, "(t \times \mathsf{id}) \times (u \times \mathsf{id})"'] & & \\
& & (\Gamma \times \Gamma) \times (\Gamma \times \Gamma) \arrow[dd, dashed, "(t \times \mathsf{id}) \times (u \times \mathsf{id})"] & & \\
& & & (A \times B) \times (\Gamma \times \Gamma) \arrow[dl, dashed, "\mathsf{id} \times \sigma \times \mathsf{id}"'] & \\
C \times C \arrow[d, "*_C"] & & (A \times \Gamma) \times (B \times \Gamma) \arrow[ll, dashed, "v \times w"] \arrow[ddll, dashed, "\mathrm{inc}"] & & (A \times B) \times \Gamma \arrow[ll, dashed, "\delta"] \arrow[ul, dashed, "\mathsf{id} \times \Delta"'] \arrow[d, "(i_1 \times i_2) \times \mathsf{id}"] \arrow[dd, dashed, bend right=60, "\mathrm{inc}"'] \\
C & & & & (A \oplus B) \times (A \oplus B) \times \Gamma \arrow[d, "*_{\oplus} \times \mathsf{id}"] \\
(A \times \Gamma) \oplus (B \times \Gamma) \arrow[u, "{[v,w]}"] & & & & (A \oplus B) \times \Gamma \arrow[llll, "d"]
\end{tikzcd}$$

  The top diagram commutes by the functoriality of $\times$. The upper-left diagram commutes by Lemma 2.19 and the functoriality of $\times$. The upper-middle diagram commutes by coherence, when precomposed with Δs. The upper-right-top diagram commutes by the naturality of $(\mathsf{id} \times \sigma \times \mathsf{id}) \circ (\mathsf{id} \times \Delta)$. The upper-right-bottom diagram is the definition of the map δ. The bottom-left diagram commutes by definition of $[v, w]$. The bottom-right diagram commutes by definition of $*_{\oplus}$. The bottom-middle diagram commutes since δ is a restriction of d.

- Rule $\mathsf{inr}(u_1) \parallel (\mathsf{inl}(t_1) \parallel \mathsf{inr}(u_2)) \longrightarrow \mathsf{inl}(t_1) \parallel \mathsf{inr}(u_1 \parallel u_2)$. The commuting diagram is

$$\begin{tikzcd}
\Gamma \times (\Gamma \times \Gamma) \arrow[r, "u_1 \times (t_1 \times u_2)"] \arrow[d, "t_1 \times (u_1 \times u_2)"'] & B \times (A \times B) \arrow[r, "i_2 \times (i_1 \times i_2)"] \arrow[dr, dashed, "i_2 \times \mathrm{inc}"'] & (A \oplus B) \times ((A \oplus B) \times (A \oplus B)) \arrow[d, "\mathsf{id} \times *_{\oplus}"] \\
A \times (B \times B) \arrow[ur, dashed, "\simeq"'] \arrow[d, "\mathsf{id} \times *_B"'] & & (A \oplus B) \times (A \oplus B) \arrow[d, "*_{\oplus}"] \\
A \times B \arrow[r, "i_1 \times i_2"'] \arrow[rr, dashed, bend left=20, "\mathrm{inc}"] & (A \oplus B) \times (A \oplus B) \arrow[r, "*_{\oplus}"'] & A \oplus B
\end{tikzcd}$$

  The upper-left diagram commutes by coherence. The upper-right diagram commutes by the functoriality of $\times$ and Lemma 2.14. The bottom diagram commutes by Lemma 2.14. The middle diagram commutes since $*_{\oplus} \circ (i_2 \times \mathrm{inc})(b_1, (a, b_2)) = (b_1, 1) *_{\oplus} (a, b_2) = (a, b_1 *_B b_2)$. ◀

▶ Remark 2.21.

- The case of the rule $\delta_\vee(\mathsf{inl}(t) \parallel \mathsf{inr}(u), x.v, y.w) \longrightarrow (t/x)v \parallel (u/y)w$ uses the fact that δ is a restriction of d. Indeed, $A \oplus B$ includes $A \times B$, and thus, $\mathsf{inl}(t) \parallel \mathsf{inr}(u)$ can be seen as a pair of terms.
- The case of the rule $\mathsf{inr}(u_1) \parallel (\mathsf{inl}(t_1) \parallel \mathsf{inr}(u_2)) \longrightarrow \mathsf{inl}(t_1) \parallel \mathsf{inr}(u_1 \parallel u_2)$ is solved by the definition of $*_\oplus$. Indeed, we do not ask all the operations to be commutative or associative, but we can still use an operation with these properties to solve the problem.

2.4.3 Adequacy

A proof of completeness would mean that if two closed proofs have the same interpretation, they both reduce to the same irreducible proof. However, this is too much to ask for in a calculus without η-reduction. Indeed, $\vdash \lambda x.tx : A \Rightarrow B$ and $\vdash t : A \Rightarrow B$ have the same interpretation, but they do not reduce to the same proof. We can only ask for the following weaker property, which is usually called adequacy. That is, if two proof-terms have the same interpretation, then they are computationally equivalent. The notion of computational equivalence is that no other program can distinguish between them. That is, putting them in any context will yield the same result. It is sufficient, however, to consider only elimination contexts of some proposition large enough to distinguish between two programs, such as $\top \vee \top$, the usual booleans.

Indeed, if K is one of these contexts, putting two terms t_1 and t_2 inside K is sufficient, and testing whether an introduction term such as $\langle K, u\rangle$ behaves the same when t_1 or t_2 are put inside K does not add any additional information. Only K, the elimination context, is sufficient to distinguish between the two terms.

The notion of an elimination context is given below. Without loss of generality, we consider the elimination context to always produce proof-terms for smaller propositions.

▶ **Definition 2.22** (Elimination context). *An elimination context is a proof-term produced by the following grammar, where $[\cdot]$ denotes a distinguished variable.*

$$K := [\cdot] \mid K\ t \mid \pi_1(K) \mid \pi_2(K) \mid \delta_\vee(K, x.t, y.u)$$

where in the proof $\delta_\vee(K, x.t, y.u)$, if $[\cdot] : C \vdash K : A \vee B$, then $x : A \vdash t : D$ and $y : B \vdash u : D$, with $|D| < |A \vee B|$. As usual, we write $K[t]$ for $(t/[\cdot])K$.

We define the computational equivalence as follows.

▶ **Definition 2.23** (Computational equivalence). *Two proofs $\vdash t : A$ and $\vdash u : A$ are computationally equivalent, written $t \sim u$, if for each elimination context $[\cdot] : A \vdash K : \top \vee \top$, there exists an irreducible proof $\vdash v : \top \vee \top$ such that $K[t] \longrightarrow^* v$ and $K[u] \longrightarrow^* v$.*

Finally, we can state and prove the adequacy of the model as follows.

▶ **Theorem 2.24** (Adequacy). *If $[\![\vdash t : A]\!] = [\![\vdash u : A]\!]$, then $t \sim u$.*

Proof. By induction on the structure of A. The full details are given in [10, Appendix B.3]. We only show one interesting case here. If $A = B \vee C$, then, by Theorems 2.1, 2.2, and 2.4, $t \longrightarrow^* \mathsf{inl}(t')$, $t \longrightarrow^* \mathsf{inr}(t')$, or $t \longrightarrow^* \mathsf{inl}(t') \parallel \mathsf{inr}(t'')$; and $u \longrightarrow^* \mathsf{inl}(u')$, $u \longrightarrow^* \mathsf{inr}(u')$, $u \longrightarrow^* \mathsf{inl}(u') \parallel \mathsf{inr}(u'')$.

However, by Theorem 2.20, and the fact that $[\![\vdash t : A]\!] = [\![\vdash u : A]\!]$, we have that if $t \longrightarrow^* \mathsf{inl}(t')$ then $u \longrightarrow^* \mathsf{inl}(u')$, if $t \longrightarrow^* \mathsf{inr}(t')$ then $u \longrightarrow^* \mathsf{inr}(u')$, and if $t \longrightarrow^* \mathsf{inl}(t') \parallel \mathsf{inr}(t'')$ then $u \longrightarrow^* \mathsf{inl}(u') \parallel \mathsf{inr}(u'')$. We must consider these three cases. We show here only the third case as example. We have

$$\begin{tikzcd} \{\star\} \arrow[r, "\Delta"] & \{\star\} \times \{\star\} \arrow[r, bend left, "t' \times t''"] \arrow[r, bend right, "u' \times u''"'] & A \times B \arrow[r, "i_1 \times i_2"] & (A \oplus B) \times (A \oplus B) \arrow[r, "*_\times"] & A \oplus B \end{tikzcd}$$

Thus, $[\![\vdash t' : A]\!] = [\![\vdash u' : A]\!]$ and $[\![\vdash t'' : B]\!] = [\![\vdash u'' : B]\!]$, and so, by the induction hypothesis $t' \sim u'$ and $t'' \sim u''$. Hence, $t \sim u$. ◀

3 The calculus $\lambda_{\mathsf{alg}}^{\mathcal{S}}$

3.1 Syntax, deduction, and reduction rules

In this section, we consider the $\lambda_{\mathsf{alg}}^{\mathcal{S}}$ calculus as an extension with scalars of the $\lambda_{\parallel}$ calculus. This calculus is similar to the Algebraic lambda calculus [19], but considering all the logical connectives, making it closer to the $\odot^{\mathcal{S}}$-calculus [8]. The idea is to consider linear combinations of terms, as in $\mathfrak{a} \cdot t \parallel \mathfrak{b} \cdot u$, where we continue writing $\parallel$ for the operation instead of the more common $+$ from algebraic calculi, to emphasise its connection with the language defined in the previous section.

In the Algebraic lambda calculus [19], the scalars belong to a semiring with zero and unity (i.e. a rig) and it is claimed not to use a ring since there are some confluence problems with negative scalars in its untyped setting. Indeed, a term $Y_t = (\lambda x.xx \parallel t)(\lambda x.xx \parallel t)$ reducing to $Y_t \parallel t$ is the source of non-confluence when considering $\mathfrak{a} \cdot Y_t \parallel -\mathfrak{a} \cdot Y_t$, which may reduce to $t \parallel \cdots \parallel t$ for any amount of t in parallel (even zero), since at any point $\mathfrak{a} \cdot Y_t$ may be cancelled with $-\mathfrak{a} \cdot Y_t$ on these calculi. There are several solutions to this problem. One is to consider only positive scalars, as done in the Algebraic lambda calculus. Another one is to consider a call-by-value beta reduction together with a linearity rule, as done in [1] (specifically, the solution involves restricting some rewriting rules to closed normal forms, which forces a call-by-value strategy). However, the simpler solution to this lack of confluence is to consider a typed setting ensuring strong normalisation, where the term Y_t is not typable. For an extensive discussion on the different design choices for algebraic calculi, see [2]. In our case, we relax all the restrictions on scalars and just require them to belong to a bi-magma, as defined next.

▶ **Definition 3.1** (Bi-magma). *A bi-magma is a set with two operations* $(\mathcal{S}, +, \cdot)$.

The syntax of the $\lambda_{\mathsf{alg}}^{\mathcal{S}}$ calculus extends the $\lambda_{\parallel}$ calculus introduced in Section 2.1 by incorporating scalars from a fixed bi-magma $\mathcal{S}$, as follows: The term $\star$ is replaced with $\star_{\mathfrak{s}}$, and a new term $\mathfrak{s} \bullet t$ is added, where $\mathfrak{s} \in \mathcal{S}$ and t is a term.

The deduction rules are also those given in Section 2.1, with the exception of rule $\top_i$, and the addition of new families of rules "$\top_i(\mathfrak{s})$" and "prod($\mathfrak{s}$)" as follows.

$$\frac{}{\Gamma \vdash \star_{\mathfrak{s}} : \top}\ \top_i(\mathfrak{s}) \qquad\qquad \frac{\vdash t : A}{\Gamma \vdash \mathfrak{s} \bullet t : A}\ \mathrm{prod}(\mathfrak{s})$$

As with the rule par, the rules prod($\mathfrak{s}$) do not introduce new provable formulas. They can be read as "If we have a proof of A, then we have a proof of A".

The reduction relation is similar to that of the $\lambda_{\parallel}$ calculus given in Section 2.1, with modifications to the elimination of $\top$ and the commutation of $\parallel$ with $\top_i(\mathfrak{s})$, since $\top_i$ now includes scalars:

$$\delta_{\top}(\star_{\mathfrak{s}}, t) \longrightarrow \mathfrak{s} \bullet t \qquad\qquad \star_{\mathfrak{s}_1} \parallel \star_{\mathfrak{s}_2} \longrightarrow \star_{\mathfrak{s}_1 + \mathfrak{s}_2}$$

Additionally, a third group is added for the commutation rules of $\mathrm{prod}(\mathfrak{s})$ with all the connectives. Indeed, just as with the rule par, the rule $\mathrm{prod}(\mathfrak{s})$ can appear between the introduction and the elimination of a connective. As in the case of the par rule, we must commute it with either the elimination or the introduction. Following the same convention, we always commute the $\mathrm{prod}(\mathfrak{s})$ rule with introductions:

$$\begin{aligned}
\mathfrak{s}_1 \bullet \star_{\mathfrak{s}_2} &\longrightarrow \star_{\mathfrak{s}_1 \cdot \mathfrak{s}_2} & \mathfrak{s} \bullet \mathsf{inl}(t) &\longrightarrow \mathsf{inl}(\mathfrak{s} \bullet t)\\
\mathfrak{s} \bullet \lambda x.t &\longrightarrow \lambda x.\mathfrak{s} \bullet t & \mathfrak{s} \bullet \mathsf{inr}(t) &\longrightarrow \mathsf{inr}(\mathfrak{s} \bullet t)\\
\mathfrak{s} \bullet \langle t, v\rangle &\longrightarrow \langle \mathfrak{s} \bullet t, \mathfrak{s} \bullet v\rangle & \mathfrak{s} \bullet (\mathsf{inl}(t) \parallel \mathsf{inr}(v)) &\longrightarrow \mathsf{inl}(\mathfrak{s} \bullet t) \parallel \mathsf{inr}(\mathfrak{s} \bullet v)
\end{aligned}$$

3.2 Correctness

The correctness properties are straightforward adaptations of the proofs of Theorems 2.1, 2.2, 2.3, and 2.4, and are therefore omitted. The only significant difference is that the introduction property now states that if $\vdash t : \top$ is irreducible, then $t = \star_{\mathfrak{s}}$ for some $\mathfrak{s} \in \mathcal{S}$.

3.3 The category $\mathbf{AMag}^{\mathcal{S}}_{\mathbf{Set}}$

▶ **Definition 3.2** (Action magma). *An action magma $(A, *_A, \bullet_A)$ over $\mathcal{S}$ consists of a magma $(A, *_A)$ and a bi-magma $(\mathcal{S}, +, \cdot)$, with a map $\mathcal{S} \times A \xrightarrow{\bullet_A} A$ such that it distributes with respect to $*_A$, that is $\mathfrak{s} \bullet_A (a_1 *_A a_2) = \mathfrak{s} \bullet_A a_1 *_A \mathfrak{s} \bullet_A a_2$.*

▶ **Definition 3.3** (The category $\mathbf{AMag}^{\mathcal{S}}_{\mathbf{Set}}$). *The category $\mathbf{AMag}^{\mathcal{S}}_{\mathbf{Set}}$ is determined by the following data: Objects are action magmas over the fixed bi-magma $\mathcal{S}$. Arrows are maps in* **Set**.

▶ **Theorem 3.4** (CCC). *The category $\mathbf{AMag}^{\mathcal{S}}_{\mathbf{Set}}$ is Cartesian closed.*

Proof. The argument is the same as the one given in Theorem 2.8. ◀

We now extend the pokeball to the Action Magmas by defining its action.

▶ **Definition 3.5** (The operation $\bullet_{\oplus}$). *Let $(A, *_A, \bullet_A), (B, *_B, \bullet_B) \in \mathsf{Obj}(\mathbf{AMag}^{\mathcal{S}}_{\mathbf{Set}})$. We define $\bullet_{A\oplus B}$ ($\bullet_{\oplus}$ for short) as $\mathcal{S} \times (A \oplus B) \xrightarrow{\bullet_{\oplus}} A \oplus B$ given by*

$$(\mathfrak{s}, c) \mapsto \begin{cases} (\mathfrak{s} \bullet_A a, 0) & \text{if } c = (a, 0)\\ (\mathfrak{s} \bullet_B b, 1) & \text{if } c = (b, 1)\\ (\mathfrak{s} \bullet_A a, \mathfrak{s} \bullet_B b) & \text{if } c = (a, b)\end{cases}$$

▶ **Definition 3.6.** *Let $A, B \in \mathsf{Obj}(\mathbf{AMag}^{\mathcal{S}}_{\mathbf{Set}})$. The following are also objects in $\mathbf{AMag}^{\mathcal{S}}_{\mathbf{Set}}$.*

- *$(A \oplus B, *_{\oplus}, \bullet_{\oplus})$ with the action given in Definition 3.5.*
- *$(\emptyset, *_{\emptyset}, \bullet_{\emptyset})$ with the action defined as the empty map.*
- *$(\{\star\}, *_{\{\star\}}, \bullet_{\{\star\}})$ with the action defined as $\mathfrak{s} \bullet_{\{\star\}} \star = \star$.*
- *A bi-magma $(\mathcal{S}, +, \cdot)$ with the magma operation defined by $+$ and the action defined by $\cdot$.*
- *$(A \times B, *_{A\times B}, \bullet_{A\times B})$ with the action defined by $\mathfrak{s} \bullet_{A\times B} (a, b) = (\mathfrak{s} \bullet_A a, \mathfrak{s} \bullet_B b)$.*
- *$([A \to B], *_{[A\to B]}, \bullet_{[A\to B]})$ with the action defined by $(\mathfrak{s} \bullet_{[A\to B]} f)(a) = \mathfrak{s} \bullet_B f(a)$.*

▶ **Definition 3.7.** *Let A be an object in $\mathbf{AMag}^{\mathcal{S}}_{\mathbf{Set}}$, we define the following arrows.*

- *$\{\star\} \xrightarrow{\mathfrak{s}} \mathcal{S}$ is defined as $\star \mapsto \mathfrak{s}$.*
- *$\mathcal{S} \times A \xrightarrow{\bullet_A} A$ is defined as $(\mathfrak{s}, a) \mapsto \mathfrak{s} \bullet_A a$.*
- *$A \xrightarrow{\hat{\mathfrak{s}}} A$ is defined as $A \xrightarrow{\rho} \{\star\} \times A \xrightarrow{\mathfrak{s}\times\mathsf{id}} \mathcal{S} \times A \xrightarrow{\bullet_A} A$.*

3.4 Model of the $\lambda^{\mathcal{S}}_{\mathsf{alg}}$ calculus

3.4.1 Interpretation

The interpretation of propositions and contexts from Section 2.4.1 remains the same, except that the interpretation of $\top$ is updated as $[\![\top]\!] = \mathcal{S}$.

The interpretation of the deduction rules is also the same as that from Section 2.4.1, except for the introduction and elimination of $\top$, and the extra rules $\mathrm{prod}(\mathfrak{s})$, which are the following:

$$
\begin{aligned}
[\![\Gamma \vdash \star_{\mathfrak{s}} : \top]\!] &= [\![\Gamma]\!] \xrightarrow{!} \{\star\} \xrightarrow{\mathfrak{s}} \mathcal{S} \\
[\![\Gamma \vdash \delta_\top(t,u) : A]\!] &= [\![\Gamma]\!] \xrightarrow{\Delta} [\![\Gamma]\!] \times [\![\Gamma]\!] \xrightarrow{t \times u} \mathcal{S} \times [\![A]\!] \xrightarrow{\bullet_A} [\![A]\!] \\
[\![\Gamma \vdash \mathfrak{s} \bullet t : A]\!] &= [\![\Gamma]\!] \xrightarrow{t} [\![A]\!] \xrightarrow{\hat{\mathfrak{s}}} [\![A]\!]
\end{aligned}
$$

3.4.2 Soundness

The proof of soundness is updated from the proof of soundness from Section 2.4.2.

As before, we require the substitution lemma (proved in [10, Appendix D.1]).

▶ **Lemma 3.8** (Substitution). *If $x : A, \Gamma \vdash t : B$ and $\Gamma \vdash u : A$, then $[\![\Gamma \vdash (u/x)t : B]\!] = [\![\Gamma \vdash t : B]\!] \circ ([\![\Gamma \vdash u : A]\!] \times \mathsf{id}) \circ \Delta$.*

▶ **Theorem 3.9** (Soundness). *If $t \longrightarrow r$ and $\Gamma \vdash t : A$, then $[\![\Gamma \vdash t : A]\!] = [\![\Gamma \vdash r : A]\!]$.*

Proof. By induction on the relation $\longrightarrow$. The full proof can be found in [10, Appendix D.2]. We only give a few illustrative examples.
Rule $\delta_\top(\star_{\mathfrak{s}}, t) \longrightarrow \mathfrak{s} \bullet t$. The commuting diagram is the following:

$$
\begin{tikzcd}
\Gamma \arrow[r, "\Delta"] \arrow[d, "t"'] & \Gamma \times \Gamma \arrow[r, "! \times t"] & \{\star\} \times A \arrow[d, "\mathfrak{s} \times \mathsf{id}"] \\
A \arrow[r, "\hat{\mathfrak{s}}"'] \arrow[urr, dashed, "\rho"] & A & \mathcal{S} \times A \arrow[l, "\bullet_A"]
\end{tikzcd}
$$

The commutation of the left diagram is trivial. The right diagram is the definition of $\hat{\mathfrak{s}}$. Rule $\star_{\mathfrak{s}_1} \parallel \star_{\mathfrak{s}_2} \longrightarrow \star_{\mathfrak{s}_1 + \mathfrak{s}_2}$. The commuting diagram is the following:

$$
\begin{tikzcd}
\Gamma \arrow[r, "!"] \arrow[d, "\Delta"'] & \{\star\} \arrow[r, "\mathfrak{s}_1 + \mathfrak{s}_2"] \arrow[d, dashed, "\Delta"'] & \mathcal{S} \\
\Gamma \times \Gamma \arrow[r, "! \times !"'] & \{\star\} \times \{\star\} \arrow[r, "\mathfrak{s}_1 \times \mathfrak{s}_2"'] & \mathcal{S} \times \mathcal{S} \arrow[u, "*_{\mathcal{S}}"']
\end{tikzcd}
$$

The left diagram commutes by naturality of Δ. The right diagram commutes since the magma operation of $\mathcal{S}$ is defined with respect to the bi-magma operation: $*_{\mathcal{S}} = +$. Rule $\mathfrak{s} \bullet \mathsf{inl}(t) \longrightarrow \mathsf{inl}(\mathfrak{s} \bullet t)$. The commuting diagram is

$$
\begin{tikzcd}
\Gamma \arrow[r, "t"] & A \arrow[r, "i_1"] \arrow[d, "\hat{\mathfrak{s}}"'] & A \oplus B \arrow[d, "\hat{\mathfrak{s}}"] \\
& A \arrow[r, "i_1"'] & A \oplus B
\end{tikzcd}
$$

The diagram commutes since $\mathfrak{s} \bullet_{\oplus} (a, 0) = (\mathfrak{s} \bullet_A a, 0)$ by definition. ◀

3.4.3 Adequacy

In this setting, the concept of computational equivalence is simpler than in the calculus $\lambda_{\parallel}$, since the proposition $\top \vee \top$ no longer represents the booleans but rather the action magma $\mathcal{S}^2$. Thus, to distinguish two programs, it is sufficient to use the proposition $\top$, as it contains as many scalars as $\mathcal{S}$ (unless $\mathcal{S} = \emptyset$, in which case the language becomes trivial, or $\mathcal{S} = \{\star\}$, in which case the language is analogous to $\lambda_{\parallel}$).

▶ **Theorem 3.10** (Adequacy). *If* $[\![\vdash t : A]\!] = [\![\vdash u : A]\!]$, *then* $t \sim u$.

Proof. By induction on the structure of A. See [10, Appendix D.3]. ◀

4 Conclusion

Summary. In this paper, we have proposed a categorical framework for modelling parallelism and algebraic combinations within intuitionistic propositional logic, without relying on monads or biproducts. Our approach is based on the minimal structure of magmas and action magmas, with arrows taken from the category **Set**, which allows us to interpret two lambda calculi: a parallel lambda calculus $\lambda_{\parallel}$ and its algebraic extension $\lambda^{\mathcal{S}}_{\mathsf{alg}}$.

A central challenge addressed in this work is the interpretation of disjunction in the presence of parallel and scalar operators. Unlike in certain models of linear logic – those equipped with biproducts – where it is possible to identify disjunction and conjunction via a common categorical structure, intuitionistic logic does not provide this feature. Our solution, based on a set-theoretic construction combining the disjoint union and the Cartesian product, allows us to model disjunction while preserving the intended behaviour of parallel composition. This highlights the disjunction as the most delicate connective to handle in this context, and its resolution was crucial to enabling a uniform interpretation of both calculi.

By showing that these effects can be captured without requiring monads or biproducts, we provide an alternative and structurally simpler foundation for reasoning about parallelism in logic. In particular, our work suggests that many of the insights from linear logic – including the idea of interpreting computational effects through categorical structure – can be translated to a broader logical setting, provided that we carefully reconsider the structural assumptions.

Homomorphisms. Neither $\mathbf{Mag}_{\mathbf{Set}}$ nor $\mathbf{AMag}^{\mathcal{S}}_{\mathbf{Set}}$ are algebraic categories, in the sense that the natural arrows for the given objects would be homomorphisms preserving the operations. The reason for this differs for each calculus and deserves some discussion.

In the case of the $\lambda_{\parallel}$ calculus, we could have opted for a category whose arrows preserve the operations of the objects. However, in such a case, it would not have been sufficient to work with magmas, since the arrow $*_{\oplus} : (A \oplus B) \times (A \oplus B) \longrightarrow A \oplus B$ is not a homomorphism unless it is associative and commutative, which is not the case unless $*_A$ and $*_B$ are. Indeed, if $f = *_{\oplus}$, then $f((c_1, c_2) *_{\times} (c_3, c_4)) = f(c_1 *_{\oplus} c_3, c_2 *_{\oplus} c_4) = (c_1 *_{\oplus} c_3) *_{\oplus} (c_2 *_{\oplus} c_4) \neq (c_1 *_{\oplus} c_2) *_{\oplus} (c_3 *_{\oplus} c_4) = f(c_1, c_2) *_{\oplus} f(c_3, c_4)$. Moreover, the category of commutative magmas is not Cartesian closed, and thus, we would not be able to interpret the $\lambda_{\parallel}$ calculus in it.

The advantage of using the category $\mathbf{Mag}_{\mathbf{Set}}$ is that it allows us to interpret the parallel operator in a more general setting, where the operator does not need to be associative or commutative. It also gives us a closer connection to the algebraic lambda calculus, where the interpretation cannot be done in a category with homomorphisms. Indeed, it is well known [2] that $(\lambda x.t)(\mathfrak{a} \bullet u)$ and $\mathfrak{a} \bullet (\lambda x.t)u$ do not yield the same result, unless the linearity is enforced by the reduction strategy as is the case of Lineal [1][1]. An easy counterexample, taking $\mathcal{S} = \mathbb{N}$, is the following: $(\lambda x.\star_1)(2 \bullet t) \longrightarrow \star_1 \neq \star_2 \longleftarrow 2 \bullet \star_1 \longleftarrow 2 \bullet (\lambda x.\star_1)t$.

The interpretation of disjunctions. As mentioned in the introduction, the fact that the parallel of disjunctions can be seen as a pair is evident. For example, compare $\mathsf{inl}(t) \parallel \mathsf{inr}(u)$ with $\langle t, u\rangle$, where t and u are terms of type A and B, respectively. This resemblance becomes even more apparent when considering the Linear Logic additive disjunction and conjunction, which are both interpreted as a biproduct.

The interpretation of disjunction that we gave for the non-linear case, namely $A \oplus B = (A \cup B) \cup (A \times B)$, attempts to mimic this biproduct-like behaviour, but without making disjunction and conjunction coincide. Instead, it establishes an inclusion. Indeed, when linearity is not considered, we cannot simply add a zero to a conjunction to obtain a disjunction. In a conjunction, both t and u are present, whereas in a disjunction with parallel, both t and u can be present – but it is still possible to have only one of them. This provides a more refined interpretation of disjunction.

We stress that our system does feature a proper disjunction: the only novelty is that, due to the presence of the parallel operator, terms such as $\mathsf{inl}(v) \parallel \mathsf{inr}(w)$ may also be viewed as conjunctive. This viewpoint is consistent with the recent preprint [12], where instead of a parallel operator an additional introduction rule $\mathsf{inlr}(v, w)$ is considered, which is equivalent to $\mathsf{inl}(v) \parallel \mathsf{inr}(w)$.

This perspective also highlights the added value of our model: the introduction of the pokeball product provides a semantic account of this phenomenon, which had previously been treated only at the syntactic level. In particular, it captures the situation where both sides of a disjunction are present simultaneously. The pokeball product shows that this behaviour can be modelled categorically in intuitionistic logic, without requiring biproducts, thereby revealing a subtle logical feature of disjunction in the presence of parallelism.

Future directions. The calculi we presented are strongly normalising. As future work, we plan to investigate the addition of fixpoint operators to extend their expressive power while ensuring that desirable semantic properties are preserved.

More broadly, we believe that the approach developed here could be extended to capture other computational effects beyond parallelism and algebraic combinations, in a spirit similar to Moggi's monadic framework. Identifying the appropriate categorical structures for such effects remains an open and promising direction for future work.

References

1 Pablo Arrighi and Gilles Dowek. Lineal: a linear algebraic λ-calculus. *Logical Methods in Computer Science*, 13(1:8):1–33, 2017. doi:10.23638/LMCS-13(1:8)2017.

2 Ali Assaf, Alejandro Díaz-Caro, Simon Perdrix, Christine Tasson, and Benoît Valiron. Call-by-value, call-by-name and the vectorial behaviour of the algebraic λ-calculus. *Logical Methods in Computer Science*, 10(4:8), 2014. doi:10.2168/LMCS-10(4:8)2014.

3 Antonio Bucciarelli, Thomas Ehrhard, and Giulio Manzonetto. Not enough points is enough. In Jacques Duparc and Thomas A. Henzinger, editors, *Computer Science Logic, 21st International Workshop, CSL 2007, 16th Annual Conference of the EACSL, Lausanne, Switzerland, September 11-15, 2007, Proceedings*, volume 4646 of *Lecture Notes in Computer Science*, pages 298–312. Springer, 2007. doi:10.1007/978-3-540-74915-8_24.

[1] In Lineal, $(\lambda x.t)(\mathfrak{a} \bullet u)$ does not beta-reduce, but first reduces to $\mathfrak{a} \bullet (\lambda x.t)u$ until u does not include any more scalars or sums, so, forcing the linearity by evaluation.

4 Antonio Bucciarelli, Thomas Ehrhard, and Giulio Manzonetto. A relational model of a parallel and non-deterministic lambda-calculus. In Sergei N. Artëmov and Anil Nerode, editors, *Logical Foundations of Computer Science, International Symposium, LFCS 2009, Deerfield Beach, FL, USA, January 3-6, 2009. Proceedings*, volume 5407 of *Lecture Notes in Computer Science*, pages 107–121. Springer, 2009. doi:10.1007/978-3-540-92687-0_8.

5 Antonio Bucciarelli, Thomas Ehrhard, and Giulio Manzonetto. A relational semantics for parallelism and non-determinism in a functional setting. *Annals of Pure and Applied Logic*, 163(7):918–934, 2012. doi:10.1016/j.apal.2011.09.008.

6 Horatiu Cirstea, Germain Faure, and Claude Kirchner. A ρ-calculus of explicit constraint application. *Higher-Order and Symbolic Computation*, 20(1-2):37–72, 2007. doi:10.1007/s10990-007-9004-2.

7 Mariangiola Dezani-Ciancaglini, Ugo de'Liguoro, and Adolfo Piperno. A filter model for concurrent λ-calculus. *SIAM Journal on Computing*, 27(5):1376–1419, 1998. doi:10.1137/S0097539794275860.

8 Alejandro Díaz-Caro and Gilles Dowek. A new connective in natural deduction, and its application to quantum computing. *Theoretical Computer Science*, 957:113840, 2023. doi:10.1016/j.tcs.2023.113840.

9 Alejandro Díaz-Caro and Gilles Dowek. A linear linear lambda-calculus. *Mathematical Structures in Computer Science*, 34(10):1103–1137, 2024. doi:10.1017/S0960129524000197.

10 Alejandro Díaz-Caro and Octavio Malherbe. Beyond monads and biproducts: A uniform interpretation of parallelism in intuitionistic logic. Extended version with appendix at arXiv:2408.16102, 2025. doi:10.48550/arXiv.2408.16102.

11 Alejandro Díaz-Caro and Octavio Malherbe. The sup connective in IMALL: A categorical semantics. Draft at arXiv:2205.02142, 2025. doi:10.48550/arXiv.2205.02142.

12 Alejandro Díaz-Caro and Gilles Dowek. A new introduction rule for disjunction. Draft at arXiv:2502.19172, 2025. doi:10.48550/arXiv.2502.19172.

13 Thomas Ehrhard and Laurent Regnier. The differential lambda-calculus. *Theoretical Computer Science*, 309(1–3):1–41, 2003. doi:10.1016/S0304-3975(03)00392-X.

14 Jim Laird, Giulio Manzonetto, Guy McCusker, and Michele Pagani. Weighted relational models of typed lambda-calculi. In *28th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2013, New Orleans, LA, USA, June 25-28, 2013*, pages 301–310. IEEE Computer Society, 2013. doi:10.1109/LICS.2013.36.

15 Saunders Mac Lane. *Categories for the Working Mathematician*. Springer, 2 edition, 1991. doi:10.1007/978-1-4757-4721-8.

16 Giulio Manzonetto. *Models and theories of lambda calculus*. PhD thesis, Univsità Ca'Foscari (Venice) and Université Paris Diderot (Paris 7), 2008. URL: https://tel.archives-ouvertes.fr/tel-00715207.

17 Richard Mayr and Tobias Nipkow. Higher-order rewrite systems and their confluence. *Theoretical Computer Science*, 192(1):3–29, 1998. doi:10.1016/S0304-3975(97)00143-6.

18 Eugenio Moggi. Notions of computation and monads. *Information and Computation*, 93(1):55–92, 1991. doi:10.1016/0890-5401(91)90052-4.

19 Lionel Vaux. The algebraic lambda calculus. *Mathematical Structures in Computer Science*, 19(5):1029–1059, 2009. doi:10.1017/S0960129509990089.