On Intersection Types and Probabilistic Lambda Calculi

Flavien Breuvart
LIPN, Université Paris-Nord
flavien.breuvart@lipn.univ-paris13.fr

Ugo Dal Lago
University of Bologna & INRIA Sophia Antipolis
ugo.dallago@unibo.it

Abstract

We define two intersection type systems for the pure, untyped, probabilistic λ-calculus, and prove that type derivations precisely reflect the probability of convergence of the underlying term. We first define a simple system of *oracle* intersection types in which derivations are annotated by binary strings and the probability of termination can be computed by combining *all* the different possible annotations. Although inevitable due to recursion theoretic limitations, the fact that (potentially) infinitely many derivations need to be considered is of course an issue when seeing types as a verification methodology. We thus develop a more complex system: the *monadic* intersection type system. In this second system, the probability of termination of a term is shown to be the *least upper bound* of the weights of its type derivations.

ACM Reference Format:
Flavien Breuvart and Ugo Dal Lago. 2018. On Intersection Types and Probabilistic Lambda Calculi. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 22 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 Introduction

Interactions between computer science and probability theory are pervasive and fruitful. Probability theory offers models that enable system abstraction, but it also suggests a new model *of computation*, like in randomised computation or cryptography [16]. All this has stimulated the study of probabilistic computational models and programming languages: probabilistic variations on automata [11, 30], Turing machines [14, 32], and the λ-calculus [31], have indeed been introduced and studied since the early days of computer science.

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

Among the many ways probabilistic choice can be captured in programming, the simplest one consists in endowing the language of programs with an operator modelling sampling from (one or many) distributions. Fair, binary, probabilistic choice is for example perfectly sufficient to get universality if the underlying programming language is itself universal [8, 32]. This is precisely what happened in the realm of the λ-calculus, from the pioneering works by Saheb-Djahromi dating back to the seventies [31] to the most recent contributions (e.g., [9, 13, 19, 27]), through the seminal work, e.g., of Jones and Plotkin [21].

Termination is a crucial property of programs, and remains desirable in a probabilistic setting, e.g., in probabilistic programming [17] where inference algorithms often rely on the underlying program to terminate. However, one needs first of all to understand *what it means* for a probabilistic computation to terminate, i.e., how termination should be defined in the first place. If one wants to stick to a *qualitative* definition, almost-sure termination is a well-known answer: a probabilistic computation is said to almost-surely terminate if divergence, although possible, has null probability. One could even go beyond and require *positive* almost-sure termination, which asks the average time to termination to be finite. This is well-known to be stronger than almost sure termination. Recursion-theoretically, checking (positive) almost-sure termination is harder than checking termination in deterministic computation, where termination is at least recursively enumerable, although undecidable: in a universal probabilistic imperative programming language, almost sure termination is Π^0_2 complete, while positive almost-sure termination is Σ^0_2 complete [23].

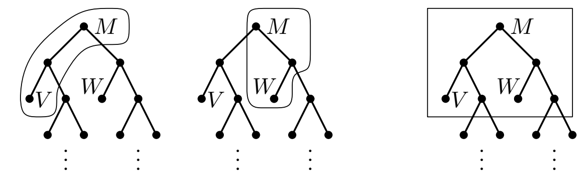
In the λ-calculus, termination is one of the best-studied verification problems [20, 22], and intersection types are well-known to be able not only to guarantee but also to *characterise* various notions of termination for λ-terms, including weak and strong normalisation [2, 6, 25]. Can all this be generalised to probabilistic λ-calculi? This paper gives a first complete, positive, answer to this question. This is however bound to be challenging: due to the aforementioned recursion-theoretic limitations, we cannot hope to get a system of intersection types in which termination is witnessed by the existence of *one* type derivation, even if we are prepared—as in the deterministic setting—to drop decidability of type inference. If we could find one, that would contradict the Π^0_2 completeness of the underlying verification problem, since type derivations can, at least, be enumerated. In other words, the price of being higher in the arithmetical hierarchy needs to be paid, somehow.

In this paper, we attack this difficult problem from two unusual routes. We design a type system by observing that binary probabilistic choice can be seen as a form of reading operation: a read-only, use-once, Boolean is taken in input from the memory and execution proceeds dependently on the read value. This suggests a way to define a system of intersection types, which we call *oracle* types. The second route we follow consists in lifting types to distributions, thus switching to a *monadic* form of type. In both cases, and again due to the aforementioned recursion-theoretic limitations, the probability of convergence cannot be read from a *single* derivation, but from *countably many of them.* While for oracle types the probability of convergence is obtained as the *sum* of the weights of *all* type derivations, monadic types allow for the weight of single type derivations to *converge* to the target probability. As a consequence, they are arguably better tailored as a verification methodology, and we will indeed define a decidable approximation of them.

After presenting the termination problem informally (Section 2) and giving the necessary preliminaries about distributions and probabilistic λ-calculi (Section 3), this paper's main contributions are presented:

- First, a system of *oracle* intersection types is introduced. The syntax of oracle intersection types explicitly mentions the bits the typed term is supposed to read from the oracle. In other words, choices performed along execution are "hard-wired" into types, and each type derivation by construction talks about *one* probabilistic branch. One then only has to sum the *weights* of derivations representing different choice sequences. Soundness and completeness of typability with respect to the probability of convergence are proved by reducibility, and by subject *expansion.* This is the topic of Section 5.
- The second type system we present is motivated by termination as a verification problem, and can be seen as being designed[1] from *prevision spaces* [18]. Probability distributions of types become first class objects, allowing for a new completeness theorem: the probability of termination of a term is *the least upper bound* of the norms of all type distributions for it. Moreover, with verification in mind, we refine our type system to suppress redundancies, and we show that there is at least one interesting fragment of it for which type inference is decidable. All this is in Section 6.

The differences between the two proposed type systems can be depicted as in Figure 1. The evaluation of a term M can be seen as being modeled as a (possibly infinite) tree whose leaves are values. While each oracle intersection type derivation can be seen as capturing *one* (finite) path leading M to a value (see Figure 1(a)), a monadic intersection type derivation captures a finite portion of the tree rooted at M obtained by pruning some (possibly infinite) subtrees (see Figure 1(b)).



(a) Oracle Intersection Types (b) Monadic Intersection Types

Figure 1. Intersection Types and the Probabilistic Evolution of Terms.

2 On Types and Termination in a Probabilistic Setting

The untyped λ-calculus [3] is a minimalist formal system enjoying some remarkable properties like confluence and standardisation, which make it an ideal candidate for a reference model of pure functional programming. Key properties of terms, like being strongly or weakly normalisable, can be characterised (somehow "compositionally") by way of intersection-type disciplines [2, 6]. This is possible not only because normalisation has a semantic counterpart which can be captured by way of types but also, more fundamentally, because being normalisable is a recursively enumerable property: otherwise, one could not hope to get a type-based characterisation of it, unless type derivation checking (which is easier than type inference) turns out to be undecidable.

We can intuitively describe an intersection type derivation as a faithful, compositional, but "optimised" description of the evaluation of the underlying typed term. As an example, consider the term $M = (\lambda x.xx)\mathbf{I}$, where $\mathbf{I}$ is the identity $\lambda y.y$. It is clear that the variable x cannot be given just *one* simple type in the term above, and intersections are there precisely to account for the multiple uses *the same* subterm can be subject to. In the end, functions are not considered in their entirety but only on a finite number of arguments, namely those it will be fed with. In the example above, $\mathbf{I}$ needs to get *both* a type in the form $\beta = \alpha{\to}\alpha$ *and* the type $\beta{\to}\beta$, and M would thus have type β. The power of intersection types comes from the fact that any type derivation built according to them can be seen as a witness to termination, and that no information is lost this way.

In a probabilistic λ-calculus, this simple and beautiful picture is simply not there anymore. First of all, confluence does *not* hold, and this is not the mere consequence of the presence of probabilistic choice: it fails even if *all probabilistic execution branches* are taken into account, i.e. if one

[1]With a lot of simplification for our much simpler framework.

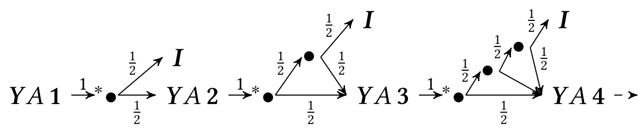
works with *distributions*. Consider, as an example, the term[2] $(\lambda x.x(x\mathbf{0}))\ (\boldsymbol{S} \oplus \boldsymbol{I})$, where $\oplus$ is an operator for fair, binary, probabilistic choice. When evaluated call-by-value (CBV for short), this term reduces to $\mathbf{0}$ with probability $\frac{1}{2}$ or to $\mathbf{2}$ with probability $\frac{1}{2}$. In call-by-name order (CBN for short), however, it reduces to $\mathbf{0}$ with probability $\frac{1}{4}$, to $\mathbf{2}$ with probability $\frac{1}{4}$ or to $\mathbf{1}$ with probability $\frac{1}{2}$. Simply put, duplication and probabilistic sampling do not commute. One avoid those issues by considering a fixed reduction strategy, which in this section will be, indeed, CBN. All the results we will give in this paper, however, hold both for CBN and CBV reduction. We will be keen to present both formulations, in order to highlight the (sometime subtle) differences.

The presence of probabilistic choice, together with the necessity of precisely capturing the behaviour of terms—we are aiming at completeness, after all—pose other challenges. Take, as an example, the term $(\lambda x.\mathbf{A} \oplus x)\ (\mathbf{B} \oplus \mathbf{C})$, where $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$ are closed terms of types α, β and γ respectively. This term call-by-name reduces to $\mathbf{A}$ with probability $\frac{1}{2}$, to $\mathbf{B}$ with probability $\frac{1}{4}$ and to $\mathbf{C}$ with probability $\frac{1}{4}$. Thus, naively, we would like this term to somehow have the type $\frac{1}{2}\alpha + \frac{1}{4}\beta + \frac{1}{4}\gamma$. This requires types to include distributions on the left of the arrow, but this is not enough. Since typing needs to be somehow compositional, one would like to type separately $(\lambda x.\mathbf{A} \oplus x)$ and $(\mathbf{B} \oplus \mathbf{C})$, and the former has to be typed *without* knowing its arguments; ideally, it would be typed without even knowing the type of its arguments, but this is known to be incompatible with completeness [1]. Summing up, the language of types and typing rules are bound to be quite complex.

In addition to keeping track of the probability of certain events, we also have to deal with the multiple uses of the same variable by a term. E.g., in a term such as $\lambda x.xx$ that uses its argument twice and very differently, we need to require the variable x to have *two* different types, say α and β. Such an intersection $\alpha \wedge \beta$ thus means that the argument have to be of type α and of type β at the same time. Somehow, this issue is orthogonal to the issue discussed above, and will thus have to be captured separately. Dealing with both of them *compositionally* is the main challenge we face, technically speaking.

As a slightly more complicated example, consider the term $(\lambda x.x\ (x\ x))\ (\lambda y.y \oplus \Omega)$. The three occurrences of x have very different roles: the rightmost one is only used as an argument, and there is no need for it to have a functional type; the one in the middle must have a functional type that takes a value (obtained with probability one) and returns something with probability $\frac{1}{2}$; the leftmost one takes an argument that may or may not converge, which means that the way it uses its argument matters a lot in the resulting probability. As a result, the management of the probabilistic behaviours we were talking about is different for each of the three occurrences of x.

Finally, it is worthwhile to notice that even if one forgets the technicalities related to the underlying type system, termination becomes itself an intrinsically more complex verification problem: a term (or, more generally, a probabilistic computation) does not merely *converge* to a normal form, but it does so *with a certain probability*. Moreover, the probability of convergence of a term is infinitary in nature. Let us consider, as an example, a term A such as $\lambda ux.x\ (\lambda y.(u\ (\boldsymbol{S}x) \oplus y))\ I$, where $\boldsymbol{S}$ is a combinator computing the successor on natural numbers. Let $\boldsymbol{Y}$ be Curry's fixpoint combinator. Then, the probabilistic evolution of the program $\boldsymbol{Y}A\mathbf{1}$ can be described by the following *infinite* tree:



In other words, $\boldsymbol{Y}A\mathbf{1}$ converges with probability 1, but this is witnessed by infinitely many (finite) probabilistic branches. Indeed, the already mentioned well-known results on the difficulty of verifying termination for probabilistic computations [23] can be rephrased as follows in any universal model of probabilistic computation like the untyped probabilistic λ-calculus [8] (and for any computable $0 < p \leq 1$):

- the lower bound problem "$\mathtt{Prob}(M\Downarrow) > p$" is Σ^0_1-complete, *i.e.*, non decidable but recursively enumerable,
- the upper bound problem "$\mathtt{Prob}(M\Downarrow) < p$" is Σ^0_2-complete, thus not recursively enumerable,
- the exact bound problem "$\mathtt{Prob}(M\Downarrow) = p$" is Π^0_2 complete, thus not recursively enumerable.

All this implies, in particular, that *it is not possible* to give a type system where the exact (or even a upper) bound for the probability of termination of a term M is always proven by a finite derivation, unless checking the correctness of a derivation becomes undecidable.

Summing up, someone looking for a complete (intersection) type system for probabilistic λ-calculi would face two major challenges:

1. The first is, as we have seen, a severe constraint from recursion theory: almost sure termination is not a recursively enumerable property, and thus cannot be captured by a simple, recursively enumerable type system.
2. The second comes from the intrinsic complexity of dealing with probabilistic effects in a compositional way, and of keeping track of the dependencies between probabilistic choices, which themselves forces the type system to be complicated, formally.

The first difficulty will be overcome by considering only the lower bound problem (which is recursively enumerable): a derivation computes an approximant of the probability of termination and the full probability of termination is obtained by considering *all* possible derivations. In other words, we

[2] where $\mathbf{0}$ is a numeral for 0, and S is the successor function.

will initially consider completely independent derivations corresponding to distinct possible executions of the term. We still have to be sure that distinct probabilistic branches correspond to type derivations which can somehow be themselves told apart. Moreover, the complex internal dependencies will be flatten so that the term $(\lambda x.C \oplus x)\ (D \oplus E)$ will get four different derivations, which can be distinguished by the derived typing judgements, one for each of the 2^2 binary strings of length 2.

Unfortunately, the aforementioned type system is not well suited as a verification methodology. Indeed, we would like to trade completeness for a Σ^0_1 and thus approximable type-checking and type-inference without too much loss. However, a type derivation in the aforementioned system only gives information on *one* possible execution. In particular, we would like a system where the completeness is not obtained by *adding* the weights of different derivations, but as the *least upper bound* of those. With this goal in mind, we define a monadic intersection type system, which will be given in Section 6 below. A judgement in this system assigns a *distribution* of types rather than a type and its contexts associate to each variable a *set* of type distributions. This time, the execution tree is not explored depth-first, but breadth-first. A derivation corresponds to a finite portion of the execution tree. In this system, Figure 2(a) is a derivation for $(\lambda x.\mathbf{A} \oplus x)\ (\mathbf{B} \oplus \mathbf{C})$. Figure 2(b) is a type derivation for $(\lambda x.x\ (x\ x))\ (\lambda y.y \oplus \Omega)$. There, we assume a distribution of type $\mathcal{A}$ such that $\mathcal{B}$ is the Dirac distribution for $\mathcal{A} \to \frac{1}{2}\mathcal{A}$, so that the last deduction comes from the fact that C can be either $\mathcal{A}$, $\mathcal{B}$, or $\frac{1}{2}\mathcal{B}$. Notice the different "style" of those two derivations.

3 A Probabilistic λ-Calculus and Its Operational Semantics

3.1 Preliminaries

Let S be any countable set. We indicate the powerset of S as $\mathfrak{P}(S)$, and its restriction to finite sets as $\mathfrak{P}_f(S)$. We often describe sets using braces $\{-\}$. Similarly, we indicate as $\mathfrak{M}(S)$ the set of multisets of S, and as $\mathfrak{M}_f(S)$ its restriction to finite multisets. To distinguish multisets from sets, we describe multisets using square brackets $[-]$. Both finite sets and finite multisets are denoted with metavariables ranging over the lowercase alphabet letters $a, b, c, \ldots$.

We denote $\mathfrak{D}(S)$ the set of *probabilistic (sub)distributions over* S:

$$\mathfrak{D}(S) := \left\{ \mathcal{M} : S \to \mathbb{R}_{[0,1]} \;\middle|\; \sum_{M \in S} \mathcal{M}(M) \leq 1 \right\}.$$

Probabilistic distributions[3] form a ω**CPO** when endowed with the pointwise order. We write $SUPP\,(\mathcal{M})$ for the *support* of the distribution over $\mathcal{M}$, namely the set $\{M \in S \mid \mathcal{M}(M) > 0\}$. In particular, we denote $\mathfrak{D}_f(S)$ the set of finitely supported distributions over S. Distributions are denoted with metavariable ranging over mathcal alphabet: $\mathcal{A}, \mathcal{B}, C, \ldots$

Distributions are always represented using brackets $\langle - \rangle$. In most cases, we use the $\mapsto$ arrow to represent the non-null coefficient, e.g., $\left\langle \begin{matrix} M \mapsto \frac{1}{2} \\ N \mapsto \frac{1}{2} \end{matrix} \right\rangle$ is the uniform distribution over the 2-element set $\{M, N\}$, which will also be denoted as $\langle \frac{1}{2}M, \frac{1}{2}N \rangle$. We sometime indicate Dirac distributions $\langle M \mapsto 1 \rangle$ by $\langle M \rangle$, while the *null* distribution is denoted $\langle \rangle$.

The set of functions (on the same set) whose codomain is the field of real numbers can be given themselves the status of a vector space. As a consequence, given two distributions $\mathcal{M}$ and $\mathcal{N}$ and a real number $p \leq 1$, we can indeed define the following:

$$\mathcal{M} + \mathcal{N} = \langle M \mapsto \mathcal{M}(M) + \mathcal{N}(M) \mid M \in SUPP\,(\mathcal{M}) \cup SUPP\,(\mathcal{N}) \rangle\,,$$
$$p\mathcal{M} = \langle M \mapsto p \cdot \mathcal{M}(M) \mid M \in SUPP\,(\mathcal{M}) \rangle\,.$$

Observe, however, that $\mathcal{M} + \mathcal{N}$ is not necessarily a *distribution*: its sum can in general be more than 1.

3.2 Probabilistic Abstract Reduction Systems

A *(fully) probabilistic abstract reduction system* (a *PARS*, for short) on S is a partial function from S to $\mathfrak{D}(S)$. Given a PARS $\to$, the fact that $(M, \mathcal{M}) \in \to$ is, as usual, indicated with $M \to \mathcal{M}$, and any $V \in S$ that cannot be reduced (i.e., $\to$ is undefined on V) is called *irreducible* or a *normal form*. The set of normal forms is denoted as S_V, while S_R is $S - S_V$. We also define the *reducible support* and *irreducible support* of $\mathcal{M} \in \mathfrak{D}(S)$ as $SUPP_R(\mathcal{M}) := SUPP\,(\mathcal{M}) \cap S_R$ and $SUPP_V(\mathcal{M}) := SUPP\,(\mathcal{M}) \cap S_V$, respectively. Any distribution $\mathcal{M}$ with empty reducible support is called itself *irreducible* and is indicated with metavariables like $\mathcal{V}$ or $\mathcal{W}$. $\mathcal{M}_V$ stands for the restriction of $\mathcal{M}$ to elements in S_V. Similarly for $\mathcal{M}_R$.

The function $\to$ can be generalised into a binary relation on $\mathfrak{D}(S)$ by taking it as the smallest such relation closed under the following rule:

$$\frac{\forall M \in SUPP_R(\mathcal{M}),\ M \to \mathcal{N}_M \qquad \forall V \in SUPP_V(\mathcal{M}),\ \mathcal{N}_M := \langle V \rangle}{\mathcal{M} \to \sum_M \mathcal{M}(M) \cdot \mathcal{N}_M}\ (r\text{-}\in)$$

Proposition 3.1. *If* $\mathcal{M} \to \mathcal{N}$ *then* $\mathcal{M}_V \leq \mathcal{N}_V$.

Actually, $\to$ as we have just defined it is a (total) function. For $n \in \mathbb{N}$, we also define the n-step reduction $\to^n$ as the n-th iteration of $\to$: $\to^0$ is the identity on $\mathfrak{D}(T)$, while $\to^{n+1} := \to^n \circ \to$ for every $n \in \mathbb{N}$. The reflexive transitive closure $\bigcup_n \to^n$ of $\to$ is, as usual, indicated as $\to^*$. Please observe that $\to^n$ is again a total function for every $n \in \mathbb{N}$. For any term M and for every $n \in \mathbb{N}$, we can thus define $\mathcal{M}_n$ to be the unique distribution such that $M \to^n \mathcal{M}_n$. Due to the increasing character of $(\mathcal{M}_n)_V$, and to the fact that distributions form

[3] In the following "distribution" always stands for "subdistribution".

$$
\dfrac{
  \dfrac{
    \vdash \boldsymbol{A} : \alpha
    \qquad
    \dfrac{}{x : \left\langle \begin{array}{l} \beta \mapsto \frac{1}{2} \\ \gamma \mapsto \frac{1}{2} \end{array} \right\rangle \vdash x : \left\langle \begin{array}{l} \beta \mapsto \frac{1}{2} \\ \gamma \mapsto \frac{1}{2} \end{array} \right\rangle}
  }{
    \dfrac{
      x : \left\langle \begin{array}{l} \beta \mapsto \frac{1}{2} \\ \gamma \mapsto \frac{1}{2} \end{array} \right\rangle \vdash \boldsymbol{A} \oplus x : \left\langle \begin{array}{l} \alpha \mapsto \frac{1}{2} \\ \beta \mapsto \frac{1}{4} \\ \gamma \mapsto \frac{1}{4} \end{array} \right\rangle
    }{
      \vdash \lambda x.\boldsymbol{A} \oplus x : \left\langle \begin{array}{l} \beta \mapsto \frac{1}{2} \\ \gamma \mapsto \frac{1}{2} \end{array} \right\rangle \to \left\langle \begin{array}{l} \alpha \mapsto \frac{1}{2} \\ \beta \mapsto \frac{1}{4} \\ \gamma \mapsto \frac{1}{4} \end{array} \right\rangle
    }
  }
  \qquad
  \dfrac{\vdash \boldsymbol{B} : \beta \qquad \vdash \boldsymbol{C} : \gamma}{\vdash \boldsymbol{B} \oplus \boldsymbol{C} : \left\langle \begin{array}{l} \beta \mapsto \frac{1}{2} \\ \gamma \mapsto \frac{1}{2} \end{array} \right\rangle}
}{
  \vdash (\lambda x.\boldsymbol{A} \oplus x)\ (\boldsymbol{B} \oplus \boldsymbol{C}) : \left\langle \begin{array}{l} \alpha \mapsto \frac{1}{2} \\ \beta \mapsto \frac{1}{4} \\ \gamma \mapsto \frac{1}{4} \end{array} \right\rangle
}
$$

(a) A Derivation Using Probabilities in CBN where $\boldsymbol{A} : \alpha$, $\boldsymbol{B} : \beta$ and $\boldsymbol{C} : \gamma$

$$
\dfrac{
  \dfrac{
    \dfrac{}{x : \tfrac{1}{2}\mathcal{B} \to \tfrac{1}{4}\mathcal{B} \vdash x : \tfrac{1}{2}\mathcal{B} \to \tfrac{1}{4}\mathcal{B}}
    \qquad
    \dfrac{
      \dfrac{}{x : \mathcal{B} \to \tfrac{1}{2}\mathcal{B} \vdash x : \mathcal{B} \to \tfrac{1}{2}\mathcal{B}}
      \qquad
      \dfrac{}{x : \mathcal{B} \vdash x : \mathcal{B}}
    }{
      x : \left\{ \begin{array}{c} \mathcal{B} \\ \mathcal{B} \to \tfrac{1}{2}\mathcal{B} \end{array} \right\} \vdash x\ x : \tfrac{1}{2}\mathcal{B}
    }
  }{
    \dfrac{
      x : \left\{ \begin{array}{c} \mathcal{B} \\ \mathcal{B} \to \tfrac{1}{2}\mathcal{B} \\ \tfrac{1}{2}\mathcal{B} \to \tfrac{1}{4}\mathcal{B} \end{array} \right\} \vdash x\ (x\ x) : \tfrac{1}{4}\mathcal{B}
    }{
      \vdash \lambda x.x\ (x\ x) : \left\{ \begin{array}{c} \mathcal{B} \\ \mathcal{B} \to \tfrac{1}{2}\mathcal{B} \\ \tfrac{1}{2}\mathcal{B} \to \tfrac{1}{4}\mathcal{B} \end{array} \right\} \to \tfrac{1}{4}\mathcal{B}
    }
  }
  \qquad
  \forall C,\ \vdash \lambda y.y \oplus \Omega : C \to \tfrac{1}{2}C
}{
  \vdash (\lambda x.x\ (x\ x))\ (\lambda y.y \oplus \Omega) : \tfrac{1}{4}\mathcal{B}
}
$$

(b) A Derivation Using Intersections in CBN

Figure 2. Examples of Derivations. For simplicity, singletons $\{\mathcal{A}\}$ are identified with there sole element $\mathcal{A}$ and Diracs $\langle \alpha \rangle$ are identified with there sole element α.

an ω**CPO**, we can define:

$$[\![M]\!] := \mathsf{sup}_n (\mathcal{M}_n)_V.$$

The distribution $[\![M]\!]$ is thus the distribution to which M converges *at the limit*. As such, it can be seen as the result of evaluating M.

3.3 Terms and Their Operational Semantics

The calculus we are interested in here, called $\Lambda_\oplus$, can be seen as obtained by endowing the pure, untyped, λ-calculus with an operator for binary probabilistic choice. This makes it a PARS, which will be the subject of this section. To be precise, this will be done in *two* different ways, corresponding to two notions of evaluation for λ-terms: call-by-name reduction and call-by-value reduction.

The set of *terms* is defined as follows:

$$\Lambda_\oplus : \qquad L, M, N ::= x \mid \lambda x.M \mid M\ N \mid M \oplus N.$$

Terms are taken modulo α-equivalence, which allows the definition of $M[N/x]$, the capture avoiding substitution of the term N for all free occurrences of x in M. The set of *closed terms* (i.e., terms in which no variable occurs free) is denoted as $\Lambda_\oplus^{\mathcal{C}}$. A *term distribution* (a *closed term distribution*, respectively) is a probabilistic subdistribution over terms (over closed terms, respectively), i.e. an element of $\Lambda_\oplus^{\mathcal{D}} = \mathfrak{D}(\Lambda_\oplus)$ (of $\Lambda_\oplus^{\mathcal{DC}} = \mathfrak{D}(\Lambda_\oplus^{\mathcal{C}})$, respectively), which is ranged over by metavariables like $\mathcal{L}$ and $\mathcal{N}$. Whenever this does not cause ambiguity, we generalise syntactic operators to operators on (closed) term distributions, e.g. $M\mathcal{N}$ stands for the distribution assigning probability $\mathcal{N}(L)$ to ML.

As already mentioned, the operational semantics takes the form of a probabilistic abstract reduction system over terms.

Definition 3.2. Rules for call-by-name (also known as weak-head) and call-by-value reduction are given in Figures 3(a)

$$\frac{}{(\lambda x.M)\;N \to_N \left\langle M[N/x]\right\rangle}\,(r_N\text{-}\beta) \qquad \frac{M \to_N \mathcal{M}}{M\;N \to_N \mathcal{M}\;N}\,(r_N\text{-}@)$$

$$\frac{}{M \oplus N \to_N \left\langle \begin{matrix} M \mapsto \frac{1}{2} \\ N \mapsto \frac{1}{2} \end{matrix} \right\rangle}\,(r_N\text{-}\oplus)$$

(a) Call-by-Name Operational Semantics

$$\frac{}{(\lambda x.M)\;V \to_V \left\langle M[V/x]\right\rangle}\,(r_V\text{-}\beta) \qquad \frac{M \to_V \mathcal{M}}{M\;V \to_V \mathcal{M}\;V}\,(r_V\text{-}@L)$$

$$\frac{N \to_V \mathcal{N}}{M\;N \to_V M\;\mathcal{N}}\,(r_V\text{-}@R) \qquad \frac{}{M \oplus N \to_V \left\langle \begin{matrix} M \mapsto \frac{1}{2} \\ N \mapsto \frac{1}{2} \end{matrix} \right\rangle}\,(r_V\text{-}\oplus)$$

(b) Call-by-Value Operational Semantics

Figure 3. Operational Semantics.

and 3(b), respectively. This defines two PARSs $\to_N$ and $\to_V$ on $\Lambda^C_\oplus$.

A quick inspection at the rules of Figure 3(a) and Figure 3(b) reveals that (closed) normal forms of $\to_N$ and $\to_V$ are *closed values*, i.e. closed abstractions. Moreover, $\to_N$ and $\to_V$ are indeed partial functions as required by the definition of a PARS: all the rules are syntax-directed.

Following the development from Section 3.2 above, we can define the semantics (or evaluation) $[\![M]\!]_N$ and $[\![M]\!]_V$ as distributions over closed, irreducible terms. The *call-by-name probability of convergence* of any term M, then, is $\sum [\![M]\!]_N := \sum_V [\![M]\!]_N(V)$. Similarly for $\sum [\![M]\!]_V := \sum_V [\![M]\!]_V(V)$, the *call-by-value probability of convergence*. These notions of evaluation satisfy some interesting properties, such as the following continuity lemma:

Lemma 3.3. *For any pair of terms M, N, the following hold:*

$$[\![[\![M]\!]_N\;\;N]\!] = [\![M\;N]\!]_N\;, \qquad [\![[\![M]\!]_V\;\;[\![N]\!]_V]\!] = [\![M\;N]\!]_V\;.$$

4 Intersection Types: A Naïve Attempt

Let us start with two remarks. First, our type system should subsume an existing intersection type system on terms not containing the binary choice operator $\oplus$. Secondly, handling duplication requires some care, since duplicating an $\oplus$ operator *before* or *after* its evaluation is drastically different (see the discussion on Section 2). It is thus natural to try to extend De Carvalho's non-idempotent intersection type system [10], which is known for being resource aware. In doing so, let us consider call-by-name evaluation.

We present a naïve intersection type system which is simply a decoration of De Carvalho's system to a probabilistic information p called "weight". A judgment $\Gamma \vdash M : p{\cdot}\alpha$

$$\frac{}{x : [\alpha] \vdash x : 1{\cdot}\alpha}\,(N\text{-}x) \quad \frac{}{\Gamma \vdash \lambda x.M : \star}\,(N\text{-}\star) \quad \frac{\Gamma \vdash M : p{\cdot}\alpha}{\Gamma \vdash M \oplus N : \frac{p}{2}{\cdot}\alpha}\,(N\text{-}\oplus L)$$

$$\frac{\Gamma; x : a \vdash M : p{\cdot}\alpha}{\Gamma \vdash \lambda x.M : 1{\cdot}(a {\to} p{\cdot}\alpha)}\,(N\text{-}\lambda) \qquad \frac{\Gamma \vdash N : p{\cdot}\alpha}{\Gamma \vdash M \oplus N : \frac{p}{2}{\cdot}\alpha}\,(N\text{-}\oplus R)$$

$$\frac{\Gamma \vdash M : p{\cdot}([\alpha_1, \ldots, \alpha_n] {\to} q{\cdot}\beta) \qquad \forall i \leq n,\ \Delta_i \vdash N : r_i{\cdot}\alpha_i}{\Gamma + \sum_i \Delta_i \vdash M\;N : p{\cdot}q{\cdot}(\prod_i r_i){\cdot}\beta}\,(N\text{-}@)$$

Figure 4. The Naive Intersection Type System.

should be read as "there is an execution of the term M converging into a normal form of type α and this execution has probability p of occurring".

The *naive intersection types* for the probabilistic lambda calculus are either a *base type* (or *type variable*) $\star$, or the *arrow type* $a{\to}p{\cdot}\alpha$, where α is itself an intersection type, $p \in [0,1]$ is the weight of the function and a is a finite multiset of intersection types:

$$\begin{array}{lrrcl} \text{(Weights)} & & p & \in & [0,1] \\ \text{(Types)} & \mathbb{T}_E & \alpha, \beta \ldots & := & \star \mid a {\to} p{\cdot}\alpha \\ \text{(Intersections)} & \mathfrak{M}_f(\mathbb{T}_E) & a, b \ldots & := & [\alpha_1, \ldots, \alpha_n] \end{array}$$

An *environment* $\Gamma : \mathbb{V} \to \mathfrak{M}_f(\mathbb{T}_E)$ is a function from variables to finite multisets of types. We define a (commutative) sum of contexts as the pointwise sum $\Gamma + \Delta := (x \mapsto \Gamma(x) \uplus \Delta(x))$. Moreover we use the following syntactic sugar: $(x : a)$ is the context that associates a to x and $[]$ to every other variable and $(\Gamma; x : a) := \Gamma + (x : a)$ is defined only whenever $\Gamma(x) = []$. A *judgment* is of the form $\Gamma \vdash M : p{\cdot}\alpha$ for $p \in \mathbb{R}_{[0,1]}$, Γ a context, M a term and α a type. The real number p is called the *weight* of the judgment. The naive intersection type system for the probabilistic λ-calculus is given in Figure 4.

The rules $(N\text{-}x)$ and $(N\text{-}\lambda)$ are similar to De Carvalho's system, with a trivial weight 1. Notice that, since we are considering a weak notion of reduction, a λ-abstraction is irreducible, thus having maximal probability of normalisation. We need, however, to keep track of the probability of convergence of the content of those λ-abstractions *once applied to an argument*: this is the purpose of the weight p in the arrow type $a{\to}p{\cdot}\alpha$. The rules $(N\text{-}\oplus L)$ and $(N\text{-}\oplus R)$ are very natural probabilistic adaptations of the classical rules used for the non-deterministic operators. The rule $(N\text{-}@)$ seems complex, but it corresponds to the rule of De Carvalho's system with the intuitive probabilistic annotations.

Our hope, at this point, is to prove that derivability of $\vdash M : p{\cdot}\alpha$ and $\sum [\![M]\!] = p$ are equivalence. Unfortunately, this is *not* the case, and for very good reasons which have to do with the recursion-theoretic difficulty of proving a probabilistic computation to be, e.g., almost surely terminating. Indeed, the set of type derivations proving the judgement above is recursively enumerable, so the "existence of a type derivation of some $\vdash M : p{\cdot}\alpha$" is a Σ^0_1 property, which is incompatible with the Π^0_2-completeness of the set of almost-surely terminating terms [23].

In fact, a derivation of $\vdash M : p{\cdot}\alpha$ means that there is *one* possible normalising execution of M which happens with probability p. If we want to fully characterise the probability of normalisation, we need to consider derivations for *each* of the potential normalising executions of M. The probability of convergence becomes the sum of the weights of each derivation. This is not as easy as one could imagine, as the following example shows.

Take the term $(\lambda x.x\ (x\ I))\ (I \oplus I)$ where $I := \lambda y.y$ is the identity. This term evaluates to $\langle I \rangle$. However, there are four possible normalising executions—each copy of $I \oplus I$ results in a probabilistic choice. This means that we are expecting four type derivations, each with a weight of $\frac{1}{4}$. But, two of them *cannot* be really distinguished. In order to describe them, let us introduce the following shortcuts:

$$\alpha := [\star] \to 1{\cdot}\star\,, \quad \beta := [\alpha] \to 1{\cdot}\alpha\,, \quad \tau := [\beta,\beta] \to 1{\cdot}\alpha\,,$$

and define a type derivation π as follows

$$\dfrac{\dfrac{\dfrac{}{x:[\beta]\vdash x:1{\cdot}\beta} \qquad \dfrac{\dfrac{}{x:[\beta]\vdash x:1{\cdot}\beta} \quad \dfrac{\dfrac{}{y:[\star]\vdash y:1{\cdot}\star}}{\vdash I:1{\cdot}\alpha}}{x:[\beta]\vdash x\ I:1{\cdot}\alpha}}{x:[\beta]\vdash x\ (x\ I):1{\cdot}\alpha}\,(N\text{-}@)}{\vdash \lambda x.x\ (x\ I):1{\cdot}\tau}\,(N\text{-}\lambda)$$

and another type derivation ρ as follows:

$$\dfrac{\dfrac{}{x:[\alpha]\vdash x:1{\cdot}\alpha}\,(N\text{-}x)}{\vdash I:1{\cdot}\beta}\,(N\text{-}\lambda).$$

The different derivations are the following ones, differing only by the uses of rules $(N\text{-}\oplus L)$ and $(N\text{-}\oplus R)$:

$$\dfrac{\dfrac{\pi}{\vdash \lambda x.x\ (x\ I):\tau} \qquad \dfrac{\dfrac{\rho}{\vdash I:1{\cdot}\beta}}{\vdash I\oplus I:\frac{1}{2}{\cdot}\beta}\,(N\text{-}\oplus L) \qquad \dfrac{\dfrac{\rho}{\vdash I:1{\cdot}\beta}}{\vdash I\oplus I:\frac{1}{2}{\cdot}\beta}\,(N\text{-}\oplus L)}{\vdash (\lambda x.x\ (x\ I))\ (I\oplus I):\frac{1}{4}{\cdot}\alpha}\,(N\text{-}@)$$

$$\dfrac{\dfrac{\pi}{\vdash \lambda x.x\ (x\ I):\tau} \qquad \dfrac{\dfrac{\rho}{\vdash I:1{\cdot}\beta}}{\vdash I\oplus I:\frac{1}{2}{\cdot}\beta}\,(N\text{-}\oplus R) \qquad \dfrac{\dfrac{\rho}{\vdash I:1{\cdot}\beta}}{\vdash I\oplus I:\frac{1}{2}{\cdot}\beta}\,(N\text{-}\oplus R)}{\vdash (\lambda x.x\ (x\ I))\ (I\oplus I):\frac{1}{4}{\cdot}\alpha}\,(N\text{-}@)$$

$$\dfrac{\dfrac{\pi}{\vdash \lambda x.x\ (x\ I):\tau} \qquad \dfrac{\dfrac{\rho}{\vdash I:1{\cdot}\beta}}{\vdash I\oplus I:\frac{1}{2}{\cdot}\beta}\,(N\text{-}\oplus L) \qquad \dfrac{\dfrac{\rho}{\vdash I:1{\cdot}\beta}}{\vdash I\oplus I:\frac{1}{2}{\cdot}\beta}\,(N\text{-}\oplus R)}{\vdash (\lambda x.x\ (x\ I))\ (I\oplus I):\frac{1}{4}{\cdot}\alpha}\,(N\text{-}@)$$

Essentially, we would like to get the following fourth derivation

$$\dfrac{\dfrac{\pi}{\vdash \lambda x.x\ (x\ I):\tau} \qquad \dfrac{\dfrac{\rho}{\vdash I:1{\cdot}\beta}}{\vdash I\oplus I:\frac{1}{2}{\cdot}\beta}\,(N\text{-}\oplus R) \qquad \dfrac{\dfrac{\rho}{\vdash I:1{\cdot}\beta}}{\vdash I\oplus I:\frac{1}{2}{\cdot}\beta}\,(N\text{-}\oplus L)}{\vdash (\lambda x.x\ (x\ I))\ (I\oplus I):\frac{1}{4}{\cdot}\alpha}\,(N\text{-}@)$$

but this derivation is in fact the same as the former one (the order between the two rightmost assumptions of the applicative rule does not matter and *cannot* matter). The issue comes from τ having twice the same type, so that we cannot distinguish the two different uses.

The naive approach would be to use ordered sequences in place of multisets, leading to non-commutative intersection types. In fact, intersection types are not able to distinguish any non-commutative effect. This is because they do not track sequentiality of the execution. That is why we rather opt for a solution preserving subject reduction where the *effect* of flipping a fair coin is kept track in the judgment, and thus in the type, as we will detail in the following section. Nevertheless, we believe that there is a way to turn probabilistic coherent spaces [13] to a system of non-idempotent intersection types, but that there is no immediate way to extract the probability of termination of a term M from the set of type derivations for M. We attack this problem in a forthcoming paper.

It is dificult to state this naive approach without making a comparison with the only other mension of probabilistic intersection types in the litterature [13**?**]. There, Ehrhard, Pagani and Tasson's use an intersection type system extracted from probabilistic coherent spaces. This system was a source of inspiration for us, but is vastly different nontheless for two important reasons:

- They choose to consider head reduction rather than weak head reduction. As result, the probabilities associated with higher order terms hardly make any sense (they can change after instanciating a polymorphic type variable). Since they consider PCF, they bypass this issue by using first order observations, but in the untyped λ-calculus this is not possible.
- The issue stated with the naive type system was resolved in a rather crude way : they anotated each sub derivation with its interpretation as coherent space in order to diferentiate fundamentally diferent one. We are looking for more resonable resolutions.

5 Achieving Completeness: Oracle Intersection Types

How should we alter the naïve intersection type system we introduce in the last section, so as to make it capable of capturing distinct probabilistic evolutions of a term by distinct type derivations? Ultimately, what made our type system *not* capable of distinguishing two essentially different sequences of probabilistic choices is that only their *probability* is taken into account, while the outcome of the choice is simply discarded. In other words, there is nothing in the type of $M \oplus N$ telling us that M and all its reducts will be reached only if the outcome of the first probabilistic choice we perform is, say, 0 rather than 1: the only thing we remember is that this path(s) have a (combined) probability of at most $\frac{1}{2}$.

The basic idea behind oracle intersection types consists in substituting real numbers in naïve intersection types with strings, this way allowing to switch to a simple type system in which intersection becomes idempotent. This works both when call-by-name and call-by-value evaluation is considered, although the type systems needs to be tuned. This is what we are going to do in this section.

It is worth noticing that our approach, in the spirit, follows the ideas underlying Goubault-Larrecq and Varacca's "Continuous Random Variables" [**?**]. However, further study would be needed in order to formalise this link.

5.1 System $\mathbf{I}_N$

Let us first describe a system of oracle intersection types fitted for the PARS $\to_N$. Since, in call-by-name, arguments are passed to functions *unevaluated*, we are bound to work with types in which intersections appear in negative positions, i.e., to the left of the arrow. But how about a fuction's *result*? Actually, an *oracle intersection types* is either the base type or an arrow type $a{\to}(s{\cdot}\alpha)$, where $(s{\cdot}\alpha)$ is a *weighted* intersection type, i.e., a pair of a string $s \in \{0,1\}^*$ and an intersection type α, and a is a set of weighted intersection types. Formally:

$$\begin{array}{rclcl} \text{(Weights)} & \mathbb{B} & s & \in & \{0,1\}^* \\ \text{(Types)} & \mathbb{O}_N & \alpha,\beta\ldots & := & \star \ \mid\ a{\to}(s{\cdot}\alpha) \\ \text{(Intersections)} & \mathbb{O}_N^{(\mathbb{B})} & a,b\ldots & := & \{(s_1{\cdot}\alpha_1),\ldots,(s_n{\cdot}\alpha_n)\} \end{array}$$

For example, a term like $\lambda x.x \oplus \Omega$ can be given the type $\{1{\cdot}\star\} \to 01{\cdot}\star$, meaning that on an input which evaluates to a value after following the right branch of a probabilistic choice (and thus having type $1{\cdot}\star$), the function results in a term which evaluates to a value after performing *two* probabilistic choices, the first one taking the left branch, and the second one taking the right branch (and thus having type $01{\cdot}\star$).

An *environment* $\Gamma : \mathbb{V} \to \mathbb{O}_N^{(\mathbb{B})}$ is a function from variables to intersections. We define a commutative sum of environments as $\Gamma \cup \Delta := (x \mapsto \Gamma(x) \cup \Delta(x))$. Moreover we use the following syntactic sugar: $(x : a)$ is the environment associating a to x and $\emptyset$ to every other variable, and $(\Gamma; x : a) := \Gamma \cup (x : a)$ is defined only whenever $\Gamma(x) = \emptyset$. A *sequent* is of the form $\Gamma \vdash M : s{\cdot}\alpha$, for $s \in \{0,1\}^*$, Γ an environment, M a term and α a type. The binary string s is called the *weight* of the sequent.

The intersection type system $\mathbf{I}_N$ is given in Figure 5. First of all, please observe how values can be typed in two different ways, namely by an arrow type and by $\star$. In both cases, the underlying binary string is ε, since values are irreducible and thus cannot be fired. Consider now rule (I_N-@): the string pq in the conclusion is the concatenation of two strings in the function M, while the strings s_k in the arguments are simply not taken into account. This corresponds to the fact that arguments are passed to functions *unevaluated*.

The following is the analogue of the classic Subject Reduction Theorem:

Proposition 5.1 (Subject Reduction). *If* $\vdash M : s{\cdot}\alpha$ *and* $M \to_N \langle N_i \mapsto p_i\rangle_{1\le i\le n}$ *then:*

- *Either* $n = 1$ *and* $\vdash N_1 : s{\cdot}\alpha$*;*
- *Or* $n = 2$*,* $s = br$ *and there is* $i \in \{1,2\}$ *such that* $\vdash N_i : r{\cdot}\alpha$*.*

Given a string $s \in \{0,1\}^*$, *the probability of* s is naturally defined as $2^{-|s|}$. Given a set of binary strings $S \subseteq \{0,1\}^*$, the expression $2^{-|S|}$ stands for the sum $\sum_{s\in S} 2^{-|s|}$, also called *the probability of* S. Given a term M, the set $\mathcal{E}_N(M) \subseteq \{0,1\}$ of *binary strings for* M is defined as follows:

$$\mathcal{E}_N(M) := \{s \in \{0,1\}^* \mid \exists\alpha.\ \vdash M : s{\cdot}\alpha\}.$$

We are now in a position to state completeness of $\mathbf{I}_N$ as a verification methodology for almost sure termination, and more generally as an inference methodology for the probability of termination:

Theorem 5.2. *Let* M *be any closed term. The probability of convergence of* M *is the sum of the probabilities of the binary strings for* M*:*

$$\sum [\![M]\!] = 2^{-|\mathcal{E}_N(M)|}.$$

Proof. We consider a λ-calculus with a linear read-only binary stream as a state, defined with a read operator $\boxplus$ such that $(0s, M \boxplus N) \to (s, M)$ and $(1s, M \boxplus N) \to (s, N)$. We consider a weak-head convergence to a term with an empty stream, so that $(01, \lambda x.x)$ is a diverging term. A reducibility argument, together with Subject Expansion, guarantees that (s, M) converges iff $\vdash M : s.\alpha$ for some α. What remains to be done, then, is to prove the following two implications:

- If $M \to_N^* \mathcal{M}$ then there is a set X of binary strings such that $\sum_{s\in X} 2^{-|s|} \geq \sum \mathcal{M}$ and for every $s \in X$ it holds that $(s, M^{\oplus 2\boxplus})$ converges.
- Let X be any set of binary strings such that for every $s \in X$ it holds that (s, M) converges. Then there is a distribution such that $(M)^{\boxplus 2\oplus} \to_N^* \mathcal{M}$ and $\sum \mathcal{M} \geq \sum_{s\in X} 2^{-|s|}$.

where $(\cdot)^{\oplus 2\boxplus}$ and $(\cdot)^{\boxplus 2\oplus}$ are simple translations that switches between the operators $\oplus$ and $\boxplus$. ◻

5.2 System $\mathbf{I}_V$

Is there a way to fit $\mathbf{I}_N$ to call-by-*value* evaluation? That is a very relevant question, given that most effectful languages indeed adopt eager evaluation—otherwise there would be no way of re-using the outcome of probabilistic sampling. The system $\mathbf{I}_V$ can be seen as designed from Girard's "boring" translation [15, 28] in the same way $\mathbf{I}_N$ is designed from the standard encoding of intuitionistic logic. Apart from that, there is no other essential difference between the two systems, with the exception of string annotations, which are placed only at the right of the arrow in $\mathbf{I}_V$, following the lifting of monads in CBV.

Intersection types take here the form $a{\to}s{\cdot}b$, where $s \in \{0,1\}^*$ is the weight of the function and where both a and b are *sets* of intersection types. Formally:

$$\begin{array}{rclcl} \text{(Weights)} & & s & \in & \{0,1\}^* \\ \text{(Types)} & \mathbb{O}_V & \alpha,\beta\ldots & := & a{\to}s{\cdot}b \\ \text{(Intersections)} & \mathbb{O}_V^{(\mathbb{B})} & a,b\ldots & := & \{\alpha_1,\ldots,\alpha_n\} \end{array}$$

$$\frac{s{\cdot}\alpha \in a}{\Gamma, x : a \vdash x : s{\cdot}\alpha}\ (I_N\text{-}x) \qquad \frac{\Gamma; x : a \vdash M : s{\cdot}\alpha}{\Gamma \vdash \lambda x.M : \varepsilon{\cdot}(a{\to}s{\cdot}\alpha)}\ (I_N\text{-}\lambda) \qquad \frac{}{\Gamma \vdash \lambda x.M : \varepsilon{\cdot}\star}\ (I_N\text{-}\star) \qquad \frac{\Gamma \vdash M : s{\cdot}\alpha}{\Gamma \vdash M \oplus N : 0s{\cdot}\alpha}\ (I_N\text{-}\oplus L)$$

$$\frac{\Gamma \vdash M : rp{\cdot}(\{s_k{\cdot}\alpha_k\}_{k\in K}{\to}(t{\cdot}\beta)) \qquad \Big\{\Gamma \vdash N : s_k{\cdot}\alpha_k\Big\}_{k\in K}}{\Gamma \vdash M\,N : rt{\cdot}\beta}\ (I_N\text{-}@) \qquad \frac{\Gamma \vdash N : s{\cdot}\alpha}{\Gamma \vdash M \oplus N : 1s{\cdot}\alpha}\ (I_N\text{-}\oplus R)$$

Figure 5. The Intersection Type System $\mathbf{I}_N$.

Environments are defined in the natural way, *sequents* are of the form $\Gamma \vdash M : s{\cdot}a$ for $s \in \{0,1\}^*$, Γ a context, M a term and a an intersection type.

The intersection type system $\mathbf{I}_V$ is given in Figure 6. Please observe how (closed) values continue to be annotated with the empty string. On the other hand, there is a striking difference in the way applications are treated: in CBN the probabilistic choices an application MN performs are those produced by M, followed by those produced by the λ-abstraction to which M evaluates *when applied* to N. In CBV, terms are passed to functions *evaluated*, and this must be taken into account.

Given a term M, the set $\mathcal{E}_V(M) \subseteq \{0,1\}$ of *binary strings for* M is defined as follows:

$$\mathcal{E}_V(M) := \{s \in \{0,1\}^* \mid \exists a.\ \vdash M : s{\cdot}a\}.$$

This results in a theorem analogous to Theorem 5.2:

Theorem 5.3. *Let M be any closed term. The probability of convergence of M is the sum of the probabilities of the binary strings for M:*

$$\sum [\![M]\!] \;=\; 2^{-|\mathcal{E}_V(M)|}.$$

The proof of Theorem 5.3 is very similar, almost identical in structure, to the one of Theorem 5.2.

5.3 On Recursion Theory

Despite their simplicity, the type systems $\mathbf{I}_V$ and $\mathbf{I}_N$ are optimal, recursion theoretically. Indeed, consider the following formula

$$F(M) = \forall n \in \mathbb{N}.\exists X \subseteq_f \{0,1\}^*.\exists Y \subseteq_f \{0,1\}^*.P(M,n,X,Y)$$

where $P(M,n,X,Y)$ encodes the fact that for any $x \in X$ there (an encoding of) a type derivation $y \in Y$ whose conclusion is $\vdash M : x \cdot \alpha$ *and* $\sum_{x\in X} 2^{-|x|} \geq 1 - 2^{-n}$. Observe that the truth value of $P(M,n,X,Y)$ can be computed in elementary time. The fact that $F(M)$ exactly captures almost surely termination of M is a consequence of soundness and completeness. Finally, notice that F is a Π^0_2-formula.

6 Trading Elegance for Tractability: Monadic Intersection Types

Until now, we have seen that usual intersection type systems can indeed been refined to a degree of resource awareness allowing to track *all* the probabilistic choices performed along a computation, by embedding string annotations into types. The completeness of oracle intersection types is certainly of theoretical importance: it tells you that there is *a* way to adapt intersection type disciplines to probabilistic λ-calculi which is optimal recursion-theoretically. However, oracle intersection types are lacking as a practical verification methodology. And this is for several reasons:

- Having to look for several "independent" derivations before obtaining a sufficiently precise result is a computationally heavy process.
- There is no easy way to turn the type system into a methodology for inferring lower bounds on the probability of termination of a given term.

This section is devoted to introducing two type systems which go beyond oracle intersection types and towards a more tractable type system. However, the resulting system is bound to be complicated.

From a monadic point of view, being capable of computing type distributions means that we want to switch to a proper probabilistic monad $\mathfrak{D}$. However, $\mathfrak{D}$ is infamous for not being distributive with respect to the powerset comonad $\mathfrak{B}$. Much work has been devoted to trying to reconcile probabilistic distributions and powersets (e.g., [34]). However, the results are only partial. The more convincing advancements are to consider convex sets of distributions or multidistributions, which have their own drawbacks.

Fortunately, we will not have to deal with this issue directly. Indeed, we can define a system where we do not need any distribution rule. The reason for that is that CBV and CBN are somehow symmetric: in CBV, we can use Girard "boring translation" [15] which does not fully use the comonadic part, while in CBN, we use a monadic version of this translation [29] as a degenerate way to treat monads in CBN.

6.1 A New Paradigm: Monadic Intersection Type Systems

Morally, what we have to do is to systematically superpose different oracle type derivations. This transformation alone, however, would not lead to a correct and complete type system. In fact, one more modification is needed: adopting sets *of distributions* over types rather that sets of types as intersections.

The *monadic intersection types* for the probabilistic lambda calculus are all *arrow types* $a{\to}\mathcal{A}$, where $\mathcal{A}$ is a distribution

$$\frac{}{x : a \vdash x : \varepsilon{\cdot}a}\ (I_V\text{-}x) \qquad \frac{\left\{\Gamma_k ; x : a_k \vdash M : s_k{\cdot}b_k\ \right\}_{k\in K}}{\cup_{k\in K}\Gamma_k \vdash \lambda x.M : \varepsilon{\cdot}\{a_k {\to} s_k{\cdot}b_k\}_{k\in K}}\ (I_V\text{-}\lambda)$$

$$\frac{\Gamma \vdash M : r{\cdot}[a{\to}t{\cdot}b] \qquad \Delta \vdash N : s{\cdot}a}{\Gamma \cup \Delta \vdash M\,N : (rst){\cdot}b}\ (I_V\text{-}@) \qquad \frac{\Gamma \vdash M : s{\cdot}a}{\Gamma \vdash M \oplus N : (0s){\cdot}a}\ (I_V\text{-}{\oplus}L) \qquad \frac{\Gamma \vdash N : s{\cdot}a}{\Gamma \vdash M \oplus N : (1s){\cdot}a}\ (I_V\text{-}{\oplus}R)$$

Figure 6. The Intersection Type System $\mathbf{I}_V$. In rule ($\mathbf{I}_V$-λ), the set of sequents means that a proof has to be given for each of these sequents.

$$\frac{\mathcal{A} \in a}{\Gamma; x : a \vdash x : \mathcal{A}}\ (MI_N\text{-}x) \qquad \frac{\Gamma \vdash M : \mathcal{A} \qquad \Gamma \vdash N : \mathcal{B}}{\Gamma \vdash M \oplus N : \frac{1}{2}\mathcal{A} + \frac{1}{2}\mathcal{B}}\ (MI_N\text{-}\oplus) \qquad \frac{}{\Gamma \vdash M : \langle\rangle}\ (MI_N\text{-}\langle\rangle)$$

$$\frac{\Gamma; x : a \vdash M : \mathcal{A}}{\Gamma \vdash \lambda x.M : \langle a{\to}\mathcal{A}\rangle}\ (MI_N\text{-}\lambda) \qquad \frac{\Gamma \vdash M : \mathcal{A} \qquad \left\{\Gamma \vdash N : \mathcal{B} \,\middle|\, \forall (a{\to}C) \in SUPP(\mathcal{A}),\ \forall \mathcal{B} \in a\right\}}{\Gamma \vdash M\,N : \sum_{a\to C} \mathcal{A}(a{\to}C).C}\ (MI_N\text{-}@)$$

Figure 7. The Call-by-Name Monadic Intersection Type System $\mathbf{MI}_N$.

over intersection types with a finite support[4] and a is a set of distributions over intersection types. This means that we are using the informal identity:

$$\mathbb{M}_N \ \simeq\ \mathfrak{P}_f(\mathfrak{D}_f(\mathbb{M}_N)) \to \mathfrak{D}_f(\mathbb{M}_N)\ .$$

More formally, we are using the following grammar:

(Types)	$\mathbb{M}_N$	$\alpha,\beta,\gamma\ldots$	$:=$	$a{\to}\mathcal{A}$
(Distributions)	$\mathfrak{D}_f(\mathbb{M}_N)$	$\mathcal{A},\mathcal{B},\mathcal{C}\ldots$	$:=$	$\langle\alpha_i \mapsto p_i\rangle_{i\le n}$
(Intersections)	$\mathfrak{P}_f(\mathfrak{D}_f(\mathbb{M}_N))$	$a,b,c\ldots$	$:=$	$\{\mathcal{A}_1,\cdots,\mathcal{A}_n\}$

Notice that a type $\star$ is not necessary anymore, as we can use $\star := \emptyset{\to}\langle\rangle$.

The *environments* $\Gamma : \mathbb{V} \to \mathfrak{P}_f(\mathfrak{D}_f(\mathbb{M}_N))$ are similar to those of the previously introduced intersection type systems. A *judgment* is of the form $\Gamma \vdash M : \mathcal{A}$ for Γ an environment, M a term and $\mathcal{A}$ distribution over types.

The monadic intersection type system $\mathbf{MI}_N$ is given in Figure 7. Notice how the rule ($\mathbf{MI}_N$-$\oplus$) explores both probabilistic branches, while in oracle intersection types only one was considered in any type derivation. This allows us to get more refined derivations, but requires another rule, namely ($\mathbf{MI}_N$-$\langle\rangle$), to cut off infinite branches in the execution. This rule is very similar to the ω rule in usual intersection types, since the empty type plays the role of ω.

Theorem 6.1 (Subject Reduction/Expansion). *If $M \to \mathcal{N}$ then $\vdash M : \mathcal{A}$ iff $\vdash \mathcal{N} : \mathcal{A}$ where $\vdash \mathcal{N} : \mathcal{A}$ means that $(\vdash N : \mathcal{A}_N)_N$ for some decomposition $\mathcal{A} = \sum_N \mathcal{N}(N)\mathcal{A}_N$.*

The *weight* of a derivation π of $\Gamma \vdash M : \mathcal{A}$ is the norm $\sum \mathcal{A}$ of the type distribution. With such a definition, we get a correctness and completeness theorem that we claim more satisfactory than Theorem 5.2, as far as verification is concerned:

Theorem 6.2. *Let M be any closed term. The probability of CbN-convergence of M is the sup of the weights of its derivations:*

$$\sum [\![M]\!] \ =\ \bigvee_{\vdash M:\mathcal{A}} \sum \mathcal{A}$$

Proof. Soundness is proved by the usual reducibility argument. Reducibility candidates are not defined as sets but by the following relations

$V \vDash_V a{\to}\mathcal{A}$	iff	$\forall M \vDash_S a,\ (V\ M) \vDash_T \mathcal{A}$
$M \vDash_S a$	iff	$\forall \mathcal{A} \in a,\ M \vDash_T \mathcal{A}$
$\mathcal{V} \vDash_D \mathcal{A}$	iff	$\mathcal{V}\ \widetilde{\vDash_V}\ \mathcal{A}$
$M \vDash_T \mathcal{A}$	iff	$[\![M]\!] \vDash_D \mathcal{A}$
$\mathcal{M} \vDash_T \mathcal{A}$	iff	$[\![\mathcal{M}]\!] \vDash_D \mathcal{A}$

where $\widetilde{\vDash_V}$ is the right-lax coupling relation over $\vDash_V$ (see [4]); *i.e*, $\mathcal{V}\ \widetilde{\vDash_V}\ \mathcal{A}$ if there is $\sigma \in \mathfrak{D}(\vDash_V)$ such that $\mathcal{V}(V) = \sum_\alpha \sigma(V,\alpha)$ and $\mathcal{A}(\alpha) \ \le\ \sum_V \sigma(V,\mathcal{A})$. The proof that $\vdash M{:}\mathcal{A}$ implies $M{\vDash_T}\mathcal{A}$ follows the usual induction over the derivation of $\vdash M{:}\mathcal{A}$, using a saturation theorem; we only need to be careful while manipulating distributions. Completeness is proved by subject expansion as usual. □

Remark that the rule (MI_N-$\langle\rangle$) can be replaced by a more expressive one allowing to combine different sub-derivations:

$$\frac{\left\{\Gamma \vdash M : \mathcal{A}_u \,\middle|\, u \in U\right\} \qquad \mathcal{U} \in \mathfrak{D}(U)}{\Gamma \vdash M : \sum_u \mathcal{U}(u)\mathcal{A}_u}\ (MI_N\text{-}D)$$

[4] Infinite distribution would make sense if one adds some special derivation for fixedpoint. But this would first require to formalise the inductive creation of types.

This rule is correct, but is not necessary to get completeness of $\mathbf{MI}_N$. However, it is necessary for the completeness of the call-by-value version.

6.2 On Call-by-Value Evaluation

The same ideas we used to build $\mathbf{I}_V$ from $\mathbf{I}_N$ can be used to turn monadic intersection types into their call-by-value counterparts. However, monadic intersection types are notationally less elegant in CBV.

The first compromise we need to make is the target calculus: for the sake of simplicity we will not consider the full calculus $\Lambda_\oplus$, but the sub-calculus containing only let-normal-forms:

Definition 6.3. A *let-normal form* is a term of the form:

$$M, N \ := \ V \ \mid \ V\,M \qquad V, W \ := \ x \ \mid \ \lambda x.M.$$

Any term can be turned into a let-normal form by eta-expending any subterm $(M\,N)$ into $(\lambda x.(\lambda y.y\,x)\,M)\,N$.

Another source of complexity comes from the type system itself: contrary to CBN, we have to add the convexity rule (MI_V-D) to get completeness. Since we only need it when typing abstractions, we merge this rule into (MI_V-λ). In addition, the rule for applications is more complex in order to preserve this convexity.

The *call by value monadic intersection types* for the probabilistic lambda calculus are *arrow types* $a \to \mathcal{A}$, where a is a set of intersection type and $\mathcal{B}$ is a distribution over sets of intersection types. This means that we are using the informal identity:

$$\mathbb{M}_V \ \simeq \ \mathfrak{P}_f(\mathbb{M}_V) \to \mathfrak{D}_f(\mathfrak{P}_f(\mathbb{M}_V)) \ .$$

More formally, we are using the following grammar:

(Types)	$\mathbb{M}_V$	$\alpha, \beta, \gamma\ldots$	$:=$	$a \to \mathcal{A}$
(Distributions)	$\mathfrak{D}_f(\mathfrak{P}_f(\mathbb{M}_V))$	$\mathcal{A}, \mathcal{B}, C\ldots$	$:=$	$\langle a_i \mapsto p_i \rangle_{i \leq n}$
(Intersections)	$\mathfrak{P}_f(\mathbb{M}_V)$	$a, b, c\ldots$	$:=$	$\{\alpha_1, \cdots, \alpha_n\}$

In addition, we define a stability condition on intersections that is not to be always respected but is only necessary in the left arguments of applications:

Definition 6.4. A set $b \in \mathfrak{P}_f(\mathbb{M}_V)$ is *stable* if any two different arrows $(a_1 \to \mathcal{A}_1), (a_2 \to \mathcal{A}_2) \in b$, can be subsumed by a third one $((a_1 \cup a_2) \to \mathcal{B}) \in b$ such that $\mathcal{B} \geq \mathcal{A}_1, \mathcal{A}_2$ with the pointwise order. The interest of stable sets is that any stable a can be lifted into a function $\hat{a} : \mathfrak{P}_f(\mathbb{M}_V) \mapsto \mathfrak{D}_f(\mathfrak{P}_f(\mathbb{M}_V))$ by:

$$\hat{a}(b) := \bigvee \{\mathcal{A} \mid \exists c \subseteq b, (c \to \mathcal{A}) \in a\} \,.$$

We also define, for all set a, the powerset $\downarrow_s a := \{c \subseteq a \mid c \text{ is stable}\}$ of its stable subsets.

The *environments* $\Gamma : \mathbb{V} \to \mathfrak{P}_f(\mathbb{M}_V)$ are defined similarly to the ones for $\mathbf{MI}_N$. Notice that types of *values* and variable are necessarily Diracs. A *judgment* is of the form $\Gamma \vdash M : \mathcal{A}$ for Γ an environment, M a term and $\mathcal{A}$ distribution over types. The call-by-value monadic intersection type system $\mathbf{MI}_V$ is given in Figure 8.

Lemma 6.5 (Subject Reduction)**.** *For any term M in let-normal form, if $M \to \mathcal{N}$ then $\vdash M : \mathcal{A}$ implies $\vdash \mathcal{N} : \mathcal{A}$ where $\vdash \mathcal{N} : \mathcal{A}$ means that $(\vdash N : \mathcal{A}_N)_N$ for some decomposition $\mathcal{A} = \sum_N \mathcal{N}(N)\mathcal{A}_N$.*

Remark that we do not claim Subject Expansion, as we did in $\mathbf{MI}_N$. In fact, we did prove the subject expension, but for a richer system whose other properties are equivalent.

As for $\mathbf{MI}_N$, the *weight* of a $\mathbf{MI}_V$ derivation of the judgement $\Gamma \vdash M : \mathcal{A}$ is the norm $\sum \mathcal{A}$ of the type distribution.

Theorem 6.6. *Let M be any closed term in let-normal form. The probability of CbV-convergence of M is the least upper bound of the weights of its derivations:*

$$\sum [\![M]\!] \ = \ \bigvee_{\vdash M:\mathcal{A}} \sum \mathcal{A}$$

6.3 Trading Completeness for Decidability

Intersection type systems are well-known to have undecidable (but semi-decidable) type inference problems, due to their completeness. This also holds for all the systems we have introduced in this paper. A natural way to get a type system with decidable type inference out of an intersection type system is to get rid of intersections, thus collapsing down to a propositional type system. In our case, we also have to suppress distributions and only considers sub-Diracs. This means that we are only considering arrow types of the form $\{p\langle\alpha\rangle\} \to q\langle\beta\rangle$, that we denote $p{\cdot}\alpha \to q{\cdot}\beta$.

We call this restriction the system of *probabilistic simple types*, or $\mathbf{PST}_N$. Its rules are given in Figure 9(a). As a subsystem of $\mathbf{MI}_N$, the system $\mathbf{PST}_N$ is sound, which is an easy consequence of the following theorem:

Theorem 6.7. *For any derivation $\Gamma \vdash M : p{\cdot}\alpha$ and any reduction $M \to^* \mathcal{M}$, there is a family of provable judgements $\left(\Gamma \vdash N : q_N{\cdot}\alpha\right)_{N \in SUPP(\mathcal{M})}$ such that*

$$p \leq \sum_N q_N{\cdot}\mathcal{M}(N) \ .$$

As announced, this system is inferenciable:

Theorem 6.8. *The type inference problem for $\mathbf{PST}_N$ is decidable.*

Proof. First, we infer a derivation without probabilistic annotations. Then, we use the inference algorithm for simply typed λ-calculus except that the algorithm use a non-deterministic oracle[5] that can stop the inference of a branch by the rule ($\mathbf{MI}_N$-$\langle\rangle$). Then we add the probabilistic annotations in a top-down manner. □

However, this system is far from being complete. For example, the following term is not typable (with a probability above 0): $(\lambda f.f\,(f\,\boldsymbol{I}))\,(\boldsymbol{I} \oplus \boldsymbol{\Omega})$.

[5]Notice that the size of the derivation is bounded by the size of the term so that there is a finite number of choices made by the oracle.

$$\frac{a \supseteq b}{\Gamma; x : a \vdash x : \langle b \rangle}\ (MI_V\text{-}x) \qquad \frac{\Gamma \vdash M : \mathcal{A} \qquad \Gamma \vdash N : \mathcal{B}}{\Gamma \vdash M \oplus N : \frac{1}{2}\mathcal{A} + \frac{1}{2}\mathcal{B}}\ (MI_V\text{-}\oplus)$$

$$\frac{\Gamma \vdash V : \langle a \rangle \qquad \Gamma \vdash N : \sum_{c \in \downarrow_s a} \mathcal{B}_c}{\Gamma \vdash V\ N : \sum_{b,c} \mathcal{B}_c(b)\hat{c}(b)}\ (MI_V\text{-}@) \qquad \frac{\left\{\Gamma; x : a_i \vdash M : \mathcal{B}_{i,u} \;\middle|\; i \le n, u \in U_i\right\} \qquad \forall i, \mathcal{U}_i \in \mathfrak{D}(U_i)}{\Gamma \vdash \lambda x.M : \left\langle\left\{ a_i \to \sum_u \mathcal{U}_i(u)\mathcal{B}_{i,u} \;\middle|\; i \le n\right\}\right\rangle}\ (MI_V\text{-}\lambda)$$

Figure 8. The Call-by-Value Monadic Intersection Type System $\mathbf{MI}_V$.

$$\frac{}{\Gamma; x : p{\cdot}\alpha \vdash x : p{\cdot}\alpha}\ (S_N\text{-}x) \qquad \frac{\Gamma \vdash M : p{\cdot}\alpha \qquad \Gamma \vdash N : q{\cdot}\alpha}{\Gamma \vdash M \oplus N : \frac{p+q}{2}{\cdot}\alpha}\ (S_N\text{-}\oplus) \qquad \frac{}{\Gamma \vdash M : 0{\cdot}\alpha}\ (S_N\text{-}\langle\rangle)$$

$$\frac{\Gamma; x : p{\cdot}\alpha \vdash M : q{\cdot}\beta}{\Gamma \vdash \lambda x.M : 1{\cdot}(p{\cdot}\alpha \to q{\cdot}\beta)}\ (S_N\text{-}\lambda) \qquad \frac{\Gamma \vdash M : p{\cdot}(r{\cdot}\alpha \to q{\cdot}\beta) \qquad \Gamma \vdash N : r{\cdot}\alpha}{\Gamma \vdash M\ N : (p.q){\cdot}\beta}\ (S_N\text{-}@)$$

(a) The Call-by-Name Probabilistic Simple Type System $\mathbf{PST}_N$

$$\frac{}{\Gamma; x : \alpha \vdash x : 1{\cdot}\alpha}\ (S_V\text{-}x) \qquad \frac{\Gamma \vdash M : p{\cdot}\alpha \qquad \Gamma \vdash N : q{\cdot}\alpha}{\Gamma \vdash M \oplus N : \frac{p+q}{2}{\cdot}\alpha}\ (S_V\text{-}\oplus) \qquad \frac{}{\Gamma \vdash M : 0{\cdot}\alpha}\ (S_V\text{-}\langle\rangle)$$

$$\frac{\Gamma; x : \alpha \vdash M : p{\cdot}\beta}{\Gamma \vdash \lambda x.M : 1{\cdot}(\alpha \to p{\cdot}\beta)}\ (S_V\text{-}\lambda) \qquad \frac{\Gamma \vdash M : q{\cdot}(\alpha \to r{\cdot}\beta) \qquad \Gamma \vdash N : p{\cdot}\alpha}{\Gamma \vdash M\ N : (p.q.r){\cdot}\beta}\ (S_V\text{-}@)$$

(b) The Call-by-Value Probabilistic Simple Type System $\mathbf{PST}_V$

Figure 9. The Probabilistic Simple Type Systems

As usual, the same ideas can be turned into a type system for CBV evaluation, called $\mathbf{PST}_V$, which is described in Figure 9(b).

7 Related Work

Intersection types appeared for the first time in the classic work by Coppo and Dezani [7]. Although the reason for their inception was mainly semantical, they have been immediately recognised as a means to characterise various notions of normalisation in the λ-calculus. As such, they have later been the object of many studies. In particular, if type intersection is *not* assumed to be idempotent, intersection types are able to capture also quantitative properties, like the number of reduction steps to normal form [5]. Recently, intersection types have been proved to be useful in synthesis [12], but also in verification [24, 33]. Noticeably, none of the work above deal with probabilistic effects.

The denotational semantics of probabilistic higher-order programming languages have been studied since the eighties, with the pioneering work of Sahjeb-Djaromi [31], followed by many others, and in particular by Jones and Plotkin [21]. Particularly relevant to our work are probabilistic coherent spaces [13**?**].

In [13], the authors are deriving intersection types derived from probabilistic coherent spaces. Those are fundamentally different from ours, for different reason we discussed in section **??**. In addition, in order to solve the problem of the naive IT system, they add historic labels to the rules, seemingly breaking the locality of typing discipline.

Recently, probabilistic higher-order computation has received a lot of attention from the research community, given the appearance of programming languages like `Church` or `Anglican`, in which probabilistic graphical models can be specified as higher-order functional programs. Those programming languages provides not only primitives for sampling from continuous distributions, but also operators for conditioning. As such, they cannot be subjected to the analysis we do here.

Designing powerful type systems for probabilistic programming languages, and for higher-order ones in particular, has remained an elusive research direction until very recently. The only publication dealing with it is due to Dal Lago and Grellois [26], and introduces a system of sized types ensuring almost sure termination for programs in a probabilistic variation of PCF. There is however a fundamental difference: while sized types are by definition incomplete, intersection types are designed in such a way as to reflect the underlying dynamic process very precisely.

8 Conclusion

In this paper, we have showed that probabilistic higher-order languages can indeed be subject to the intersection type systems, obtaining results similar to those one gets in the usual, deterministic, setting. The price to pay for capturing a class of terms which stands very high in the arithmetical hierarchy is the fact that not *one*, but *countably many* derivations need to be analysed for completeness to hold.The main result then trades completeness and elegance for tractability, providing a type system which is complete only up-to approximations.

Topics for further work certainly includes the study of more tractable version of our type system. Applications of (variations of) our type disciplines to verification and synthesis, possibly following the works by Rehof, Kobayashi and others [12, 24, 33] are also very intriguing research directions, which however lie outside the scope of this paper.

But, for now, we intend to explore underlying denotational semantics. Indeed, it is folklore that intersection type systems correspond to specific kind of denotational semantics. In the case of the monadic system, the unusual asymmetry between CbN and CbV indicates that the underlying semantics is highly non-standard and thus an interesting subject of study.

References

[1] Samson Abramsky. 1990. Abstract Interpretation, Logical Relations and Kan Extensions. *J. Log. Comput.* 1, 1 (1990), 5–40. https://doi.org/10.1093/logcom/1.1.5

[2] Henk Barendregt, Mario Coppo, and Mariangiola Dezani-Ciancaglini. 1983. A Filter Lambda Model and the Completeness of Type Assignment. *J. Symb. Log.* 48, 4 (1983), 931–940.

[3] Henk P. Barendregt. 1984. *The Lambda Calculus, Its Syntax and Semantics.*

[4] Gilles Barthe, Benjamin Grégoire, Justin Hsu, and Pierre-Yves Strub. 2017. Coupling proofs are probabilistic product programs. In *Proceedings of the 44th ACM SIGPLAN Symposium on Principles of Programming Languages, POPL 2017, Paris, France, January 18-20, 2017.* 161–174. http://dl.acm.org/citation.cfm?id=3009896

[5] Antonio Bucciarelli, Delia Kesner, and Daniel Ventura. 2017. Non-idempotent intersection types for the Lambda-Calculus. *Logic Journal of the IGPL* 25, 4 (2017), 431–464. https://doi.org/10.1093/jigpal/jzx018

[6] Mario Coppo and Mariangiola Dezani-Ciancaglini. 1978. A new type assignment for λ-terms. *Archiv für mathematische Logik und Grundlagenforschung* 19, 1 (1978), 139–156.

[7] Mario Coppo and Mariangiola Dezani-Ciancaglini. 1980. An extension of the basic functionality theory for the λ-calculus. *Notre Dame Journal of Formal Logic* 21, 4 (1980), 685–693. https://doi.org/10.1305/ndjfl/1093883253

[8] Ugo Dal Lago and Margherita Zorzi. 2012. Probabilistic operational semantics for the lambda calculus. *RAIRO - Theor. Inf. and Applic.* 46, 3 (2012), 413–450.

[9] V. Danos and R. Harmer. 2002. Probabilistic game semantics. *ACM Trans. Comput. Log.* 3, 3 (2002), 359–382.

[10] Daniel de Carvalho. 2009. Execution Time of lambda-Terms via Denotational Semantics and Intersection Types. *CoRR* abs/0905.4251 (2009).

[11] Karel De Leeuw, Edward F Moore, Claude E Shannon, and Norman Shapiro. 1956. Computability by probabilistic machines. *Automata studies* 34 (1956), 183–198.

[12] Andrej Dudenhefner and Jakob Rehof. 2017. Intersection type calculi of bounded dimension. In *Proceedings of the 44th ACM SIGPLAN Symposium on Principles of Programming Languages, POPL 2017, Paris, France, January 18-20, 2017*, Giuseppe Castagna and Andrew D. Gordon (Eds.). ACM, 653–665. https://doi.org/10.1145/3009837

[13] Thomas Ehrhard, Michele Pagani, and Christine Tasson. 2014. Probabilistic Coherence Spaces are Fully Abstract for Probabilistic PCF. In *POPL*, P. Sewell (Ed.). ACM.

[14] John Gill. 1977. Computational complexity of probabilistic Turing machines. *SIAM J. Comput.* 6, 4 (1977), 675–695.

[15] Jean-Yves Girard. 1987. Linear Logic. *Theor. Comput. Sci.* 50 (1987), 1–102. https://doi.org/10.1016/0304-3975(87)90045-4

[16] Shafi Goldwasser and Silvio Micali. 1984. Probabilistic encryption. *Journal of computer and system sciences* 28, 2 (1984), 270–299.

[17] Noah D. Goodman, Vikash K. Mansinghka, Daniel M. Roy, Keith Bonawitz, and Joshua B. Tenenbaum. 2008. Church: a language for generative models. In *UAI*. 220–229.

[18] Jean Goubault-Larrecq. 2015. Full Abstraction for Non-Deterministic and Probabilistic Extensions of PCF I: the Angelic Cases. *Journal of Logic and Algebraic Methods in Programming* 84, 1 (Jan. 2015), 155–184. https://doi.org/10.1016/j.jlamp.2014.09.003

[19] Chris Heunen, Ohad Kammar, Sam Staton, and Hongseok Yang. 2017. A convenient category for higher-order probability theory. In *32nd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2017, Reykjavik, Iceland, June 20-23, 2017.* 1–12.

[20] John Hughes, Lars Pareto, and Amr Sabry. 1996. Proving the Correctness of Reactive Systems Using Sized Types. In *POPL*. 410–423.

[21] C. Jones and Gordon D. Plotkin. 1989. A Probabilistic Powerdomain of Evaluations. In *LICS*. 186–195.

[22] Neil D. Jones and Nina Bohr. 2008. Call-by-Value Termination in the Untyped lambda-Calculus. *Logical Methods in Computer Science* 4, 1 (2008).

[23] Benjamin Lucien Kaminski and Joost-Pieter Katoen. 2015. On the Hardness of Almost-Sure Termination. In *MFCS (LNCS)*, Vol. 9234. 307–318.

[24] Naoki Kobayashi and C.-H. Luke Ong. 2009. A Type System Equivalent to the Modal Mu-Calculus Model Checking of Higher-Order Recursion Schemes. In *Proceedings of the 24th Annual IEEE Symposium on Logic in Computer Science, LICS 2009, 11-14 August 2009, Los Angeles, CA, USA*. IEEE Computer Society, 179–188. https://doi.org/10.1109/LICS.2009.29

[25] Jean-Louis Krivine. 1993. *Lambda-calculus, types and models*. Masson.

[26] Ugo Dal Lago and Charles Grellois. 2017. Probabilistic Termination by Monadic Affine Sized Typing. In *Programming Languages and Systems - 26th European Symposium on Programming, ESOP 2017, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2017, Uppsala, Sweden, April 22-29, 2017, Proceedings (Lecture Notes in Computer Science)*, Hongseok Yang (Ed.), Vol. 10201. Springer, 393–419. https://doi.org/10.1007/978-3-662-54434-1_15

[27] Ugo Dal Lago, Davide Sangiorgi, and Michele Alberti. 2014. On coinductive equivalences for higher-order probabilistic functional programs. In *The 41st Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL '14, San Diego, CA, USA, January 20-21, 2014*. 297–308.

[28] John Maraist, Martin Odersky, David N. Turner, and Philip Wadler. 1999. Call-by-name, Call-by-value, Call-by-need and the Linear lambda Calculus. *Theor. Comput. Sci.* 228, 1-2 (1999), 175–210.

[29] Eugenio Moggi. 1991. Notions of computation and monads. *Information and computation* 93, 1 (1991), 55–92.

[30] Michael O Rabin. 1963. Probabilistic automata. *Information and control* 6, 3 (1963), 230–245.

[31] N. Saheb-Djahromi. 1978. Probabilistic LCF. In *MFCS*. 442–451.

[32] Eugene S Santos. 1969. Probabilistic Turing machines and computability. *Proc. Amer. Math. Soc.* 22, 3 (1969), 704–710.

[33] Takeshi Tsukada and Naoki Kobayashi. 2012. An Intersection Type System for Deterministic Pushdown Automata. In *Theoretical Computer Science - 7th IFIP TC 1/WG 2.2 International Conference, TCS 2012, Amsterdam, The Netherlands, September 26-28, 2012. Proceedings (Lecture Notes in Computer Science)*, Jos C. M. Baeten, Thomas Ball, and Frank S. de Boer (Eds.), Vol. 7604. Springer, 357–371. https://doi.org/10.1007/978-3-642-33475-7_25

[34] Daniele Varacca and Glynn Winskel. 2006. Distributing probability over non-determinism. *Mathematical Structures in Computer Science* 16, 1 (2006), 87–113.

9 Annex A: Notations and Coupling

For readability, we introduce the following integral notation (even so we only speak of finite sums):

For any distribution $\mathcal{M} \in \mathfrak{D}(S)$ and for any sequence $(\mathcal{X}_M)_{M \in SUPP(\mathcal{M})} \in \mathfrak{D}(X)^{SUPP(\mathcal{M})}$, we define

$$\int_M \mathcal{A}_M d\mathcal{M} = \sum_M \mathcal{M}(M)\mathcal{A}_M.$$

Similarly, for any distribution $\mathcal{R} \in \mathfrak{D}(S \times T)$ and for sequence $(\mathcal{A}_{s,t})_{(s,t) \in SUPP(\mathcal{M})} \in \mathfrak{D}(A)^{SUPP(\mathcal{M})}$, we define

$$\int_{(s,t)} \mathcal{A}_{s,t} d\mathcal{R} = \sum_d \sum_t \mathcal{R}(s,t)\mathcal{A}_{s,t}.$$

The integral can be manipulated as usual integrals which respect the less usual equation:

$$\int_N \mathcal{L}_N d\left(\int_M \mathcal{N}_M d\mathcal{M}\right) \quad = \quad \int_M \left(\int_N \mathcal{L}_N d\mathcal{N}_M\right) d\mathcal{M} \quad (1)$$

For example, rule ($\mathbf{MI}_N$-@) can be rewritten:

$$\frac{\Gamma \vdash M : \mathcal{A} \qquad \left\{\Gamma \vdash N : \mathcal{B} \;\middle|\; \forall (a \to C) \in SUPP\,(\mathcal{A})\,,\ \forall \mathcal{B} \in a\right\}}{\Gamma \vdash M\,N : \int_{(a \to C)} C d\mathcal{A}}$$

Similarly, rule (MI_V-λ) can be rewritten:

$$\frac{\left\{\Gamma; x : a_i \vdash M : \mathcal{B}_{i,u} \;\middle|\; i \leq n, u \in U_i\right\} \qquad \forall i, \mathcal{U}_i \in \mathfrak{D}(U_i)}{\Gamma \vdash \lambda x.M : \left\langle \left\{ a_i \to \int \mathcal{U}_i \mathcal{B}_{i,u} u \;\middle|\; i \leq n \right\} \right\rangle}$$

It is now well known [4] that, given a relation between sets, we can define its coupling relation between their sets of distribution. Here, we are using the "right-lax" variant that is similar but fits our situation where types are only approximations.

Definition 9.1 (Coupling)**.**
For any relation $\triangle \subseteq S \times T$, we use $\widetilde{\triangle} \in \mathfrak{D}(S) \times \mathfrak{D}(T)$ to denote the *right-lax coupling relation* defined by $\mathcal{M}\,\widetilde{\triangle}\,\mathcal{A}$ whenever there exists a *coupling distribution* $\mathcal{R} \in \mathfrak{D}(\triangle)$ such that:

$$\forall s \in S, \quad \mathcal{M}(s) = \int_{(s,')} \delta_{s,s'} d\mathcal{R}t$$

$$\forall t \in T, \quad \mathcal{A}(t) \leq \int_{(,t,t')} \delta d\mathcal{R}st',$$

where δ is the Kronecker function

Intuitively, imagine that $\mathcal{M}$ and $\mathcal{A}$ are sets containers of volumes $(\mathcal{M}(s))_{s \in S}$ and $(\mathcal{A}(t))_{t \in T}$. If, by filling $\mathcal{M}$ with water, and by putting canals between s and t whenever $s\Delta t$, we can transfer the water and fill $\mathcal{A}$, then we say that $\mathcal{M}\,\widetilde{\triangle}\,\mathcal{A}$.

For example, we will later use the ordering ($\sqsubseteq$) over distributions of sets which be redefined as the lifting of the inclusion: $(\sqsupseteq) := (\widetilde{\supseteq})$.

The coupling have many definitions and many wonderful properties. One of its main interests is that it will lift the relation to a linear relation:

Lemma 9.2 (Linearity of coupling)**.**
For any relation $\triangle \subseteq S \times T$, the right-lax coupling relation is linear in the sens that for any instances $\mathcal{M} \,\widetilde{\triangle}\, \mathcal{A}$ and $\mathcal{N} \,\widetilde{\triangle}\, \mathcal{B}$ and for any $p + q \leq 1$:

$$p\mathcal{M} + q\mathcal{N} \quad \widetilde{\triangle} \quad p\mathcal{A} + q\mathcal{B}.$$

Lemma 9.3 (Sublinearity of coupling)**.**
For any relation $\triangle \subseteq S \times T$, the right-lax coupling relation is right-sub-linear in the sens that for any instances $(\mathcal{M} \,\widetilde{\triangle}\, \mathcal{A}_u)_{u \in U}$ and any distribution $\mathcal{U} \in \mathfrak{D}(U)$:

$$\mathcal{M} \quad \widetilde{\triangle} \quad \int_u \mathcal{A}_u d\mathcal{U}.$$

In the meantime, it conserves (or decrease) the total weight of the distributions:

Lemma 9.4 (weight-conservation of coupling)**.**
For any relation $\triangle \subseteq S \times T$, the right-lax coupling relation is size decreasing in the sens that for any instances
$$\mathcal{M} \,\widetilde{\triangle}\, \mathcal{A}, \sum \mathcal{M} \geq \sum \mathcal{A}.$$

An other interest is that coupling distributions can project into one axis or the other:

Lemma 9.5 (Projections)**.**
For any functions $f : S \to \mathfrak{D}(L)$ (resp. $g : T \to \mathfrak{D}(L)$) and any coupling , $\mathcal{M} \,\widetilde{\triangle}\, \mathcal{A}$ given by a coupling distribution $\mathcal{R} \in \mathfrak{D}(\triangle)$ we have:

$$\int_{(s,t)} f(s) d\mathcal{R} = \int_s f(s) d\mathcal{M}$$
$$\left(resp. \int_{(s,t)} g(t) d\mathcal{R} \geq \int_t g(t) d\mathcal{A} \right)$$

In fact coupling distributions are equivalently characterised as the $\mathcal{R} \in \mathfrak{D}(\triangle)$ that verifies these equations for all f and g.

10 Annex B: full proofs of CbN Saturation and Theorem 6.2

In this annex, we are proving the saturation theorem (Theorem 10.3), as well as the completeness (Theorem 10.5) and soundness (Theorem 10.9) of $\mathbf{MI}_N$; which corresponds to Theorem 6.1 for the saturation and Theorem 6.2 for the soundness/completeness.

10.1 Saturation: Subject Reduction and Subject Expansion

First, we need a weakening lemma:

Lemma 10.1 (Weakening)**.** *For any derivation $\Gamma, x{:}a \vdash M : \mathcal{A}$, and any $b \supseteq a$, we have a derivation $\Gamma, x{:}b \vdash M : \mathcal{A}$.*

Proof. By a trivial induction of the type derivation. □

As usual, we prove a substitution lemma:

Lemma 10.2 (Substitution lemma)**.** *For any Γ, M, N and $\mathcal{A}$, the two are equivalent:*

1. $\Gamma \vdash M[N/x] : \mathcal{A}$
2. *there is a such that $\Gamma, x{:}a \vdash M : \mathcal{A}$ and for all for all $\mathcal{B} \in a, \Gamma \vdash N : \mathcal{B}$.*

Proof. (1) ⇒ (2) By induction on $\Gamma \vdash M[N/x] : \mathcal{A}$

- If $M = x$ then $a = \{\mathcal{A}\}$,
- If $M = y \neq x$ then $a = \emptyset$,
- If $\mathcal{A} = \langle\rangle$, then $a = \emptyset$,
- We assume that $M = M_1 \oplus M_2$ and $\mathcal{A} = \mathcal{A}_1 + \mathcal{A}_2$ with
$$\Gamma \vdash M_1[N/x] : \mathcal{A}_1 \quad \text{and} \quad \Gamma \vdash M_2[N/x] : \mathcal{A}_2.$$
Then by induction hypothesis there is a_1, a_2 such that
$$\Gamma, x{:}a_1 \vdash M_1 : \mathcal{A}_1, \quad \Gamma, x{:}a_2 \vdash M_2 : \mathcal{A}_2$$
$$\text{and} \ \ \forall \mathcal{B} \in a_1 \cup a_2, \quad \Gamma \vdash N : \mathcal{B}.$$
We can thus set $a = a_1 \cup a_2$.
Then, by Lemma 10.1, we have
$$\Gamma, x{:}a \vdash M_1 : \mathcal{A}_1 \quad \text{and} \quad \Gamma, x{:}a \vdash M_2 : \mathcal{A}_2$$
so that $\Gamma, x{:}a \vdash M : \mathcal{A}$.
- We assume that $M = \lambda y.M'$ and $\mathcal{A} = \langle b {\to} \mathcal{A}' \rangle$ with
$$\Gamma, y{:}b \vdash M'[N/x] : \mathcal{A}'.$$
By induction hypothesis there is a such that
$$\Gamma, x{:}a \vdash M' : \mathcal{A}' \quad \text{and} \quad \forall \mathcal{B} \in a, \Gamma \vdash N : \mathcal{B}.$$
We conclude by applying rule ($\mathbf{MI}_N$-λ) back.
- We assume that $M = M_1 M_2$ and $\mathcal{A} = \int_{(c \to C)} C d\mathcal{A}'$ with $\Gamma \vdash M_1[N/x] : \mathcal{A}'$ and with
$$\forall (c {\to} C) \in SUPP(\mathcal{A}), \forall \mathcal{B} \in c, \quad \Gamma \vdash M_2[N/x] : \mathcal{B}.$$
then by induction hypothesis, there is a' and $(a_{\mathcal{B}})_{\mathcal{B}}$ such that $\Gamma, x{:}a' \vdash M_1 : \mathcal{A}'$, such that
$$\forall (c {\to} C) \in SUPP(\mathcal{A}), \forall \mathcal{B} \in c, \quad \Gamma, x{:}a_{\mathcal{B}} \vdash M_2 : \mathcal{B}, \text{t}$$
and such that
$$\forall \mathcal{B} \in a' \cup \bigcup a_{\mathcal{B}}, \quad \Gamma \vdash N : \mathcal{B}.$$
If we set $a = a' \cup \bigcup a_{\mathcal{B}}$, we can conclude by applying rule ($\mathbf{MI}_N$-@) back.

(2) ⇒ (1): By trivial induction on $\Gamma, x : a \vdash M : \mathcal{A}$

- If $M = x$ and $\mathcal{A} \in a$, then $\Gamma \vdash x[N/x] = N : \mathcal{A}$ by definition.
- If $M = y \neq x$ and $\Gamma = \Gamma', y : b$ with $\mathcal{A} \in b$, then $\Gamma \vdash y[N/x] = y : \mathcal{A}$.
- If $\mathcal{A} = \langle\rangle$ then it is trivial,
- We assume that $M = \lambda y.M'$ and $\mathcal{A} = \langle b \to \mathcal{A}' \rangle$ with
$$\Gamma, x{:}a, y{:}b \vdash M' : \mathcal{A}'.$$
Then by induction hypothesis $\Gamma, y{:}b \vdash M'[N/x] : \mathcal{A}'$, so that we can conclude by applying rule ($\mathbf{MI}_N$-λ) back.
- We assume that $M = M_1\, M_2$ and $\mathcal{A} = \int_{(c \to C)} C d\mathcal{A}'$ with $\Gamma, x{:}a \vdash M_1 : \mathcal{A}'$ and with
$$\forall (c \to C) \in SUPP\,(\mathcal{A}),\ \forall \mathcal{B} \in c\Gamma, \quad x{:}a \vdash M_2 : \mathcal{B}.$$
Then by induction hypothesis $\Gamma \vdash M_1[N/x] : \mathcal{A}'$ and
$$\forall (c \to C) \in SUPP\,(\mathcal{A}),\ \forall \mathcal{B} \in c, \quad \Gamma \vdash M_2[N/x] : \mathcal{B},$$
we can conclude by applying rule ($\mathbf{MI}_N$-@) back.

□

This allows to prove the saturation theorem (Theorem 6.1), which is the subject reduction and expansion together:

Theorem 10.3 (Saturation). *If $M \to \mathcal{N}$ then*
$$\vdash M : \mathcal{A} \qquad \textit{iff} \qquad \vdash \mathcal{N} : \mathcal{A}$$
where $\vdash \mathcal{N} : \mathcal{A}$ means that $(\vdash N : \mathcal{A}_N)_N$ for some decomposition $\mathcal{A} = \int_N \mathcal{A}_N d\mathcal{N}$.

Proof. By induction on $M \to \mathcal{N}$. The difficult case is for $(\lambda x.M)N \to \langle M[N/x] \rangle$, where we use Lemma 10.2.

- Suppose that $M = (\lambda x.M_1)M_2 \to \langle M[M_2/x] \rangle = \mathcal{N}$. Then $\vdash (\lambda x.M)M_2 : \mathcal{A}$ means that either:
$$\dfrac{\dfrac{x : a \vdash M_1 : \mathcal{A}}{\vdash \lambda x.M : \langle a \to \mathcal{A} \rangle}\,(\mathbf{MI}_N\text{-}\lambda) \qquad \left\{ \vdash M_2 : \mathcal{B} \,\middle|\, \forall \mathcal{B} \in a \right\}}{\vdash (\lambda x.M)\, M_2 : \mathcal{A}}$$
or $\mathcal{A} = \emptyset$ which is a subcase of the previous one (for $a = \emptyset$ and using ($\mathbf{MI}_N$-$\langle\rangle$) over M_1).
By Lemma 10.2, the conditions are equivalent to
$$\vdash \langle M[N/x] \rangle : \mathcal{A}.$$
- Suppose that $M = M_1 \oplus M_2 \to \frac{1}{2}\langle M_1 \rangle + \frac{1}{2}\langle M_2 \rangle = \mathcal{N}$. Then $\vdash M : \mathcal{A}$ means either that:
$$\dfrac{\vdash M_1 : \mathcal{A}_1 \qquad \vdash M_2 : \mathcal{A}_2}{\vdash M_1 \oplus M_2 : \frac{1}{2}\mathcal{A}_1 + \frac{1}{2}\mathcal{A}_2 = \mathcal{A}}\,(MI_V\text{-}\oplus)$$
or that $\mathcal{A} = \langle\rangle$, but the first subsumes the second (for $\mathcal{A}_1 = \mathcal{A}_2 = \langle\rangle$).
Thus we can conclude since
$$\mathcal{A}_{M_i} = \mathcal{A}_i \ \text{ and } \ \mathcal{A} = \int_N \mathcal{A}_N d\mathcal{N} = \tfrac{1}{2}\mathcal{A}_1 + \tfrac{1}{2}\mathcal{A}_2.$$
- Suppose that $M = M_1\, M_2$ and $M_1 \to \mathcal{N}'$ with $\mathcal{N} = \mathcal{N}'\, M_2$.
  - If $\vdash M : \mathcal{A}$ then either
$$\dfrac{\vdash M_1 : \mathcal{A}' \quad \left\{ \vdash M_2 : \mathcal{B} \,\middle|\, \forall (a \to C) \in SUPP\,(\mathcal{A}'),\ \forall \mathcal{B} \in a \right\}}{\vdash M_1\, M_2 : \displaystyle\int_{(a \to C)} C d\mathcal{A}'}$$
with $\mathcal{A} = \int_{(a \to C)} C d\mathcal{A}'$; or $\mathcal{A} = \langle\rangle$, but the first subsumes the second (for $\mathcal{A}' = \langle\star\rangle$).
By induction hypothesis, we have $\vdash \mathcal{N}' : \mathcal{A}'$ so that there is $(\vdash N : \mathcal{A}'_N)_N$ for some decomposition
$$\mathcal{A}' = \int_N \mathcal{A}'_N d\mathcal{N}'.$$
By applying ($\mathbf{MI}_N$-@) for any N, we get that
$$\vdash N\, M_2 : \int_{(a \to C)} C d\mathcal{A}'_N.$$
We set $\mathcal{A}_{(N M_2)} := \int_{(a \to C)} C d\mathcal{A}_N$ and $\mathcal{A}_N = \langle\rangle$ otherwise, so that $\vdash N : \mathcal{A}_N$ for all N.
We can conclud since:
$$\begin{aligned} \int_N \mathcal{A}_N d\mathcal{N} &= \int_N \left(\int_{(a \to C)} C d\mathcal{A}'_N \right) d\mathcal{N}' && \text{def } \mathcal{A}'_N \\ &= \int_{(a \to C)} C d \left(\int_N \mathcal{A}'_N d\mathcal{N}' \right) && \text{Eq (1)} \\ &= \mathcal{A} \end{aligned}$$
  - If $\vdash \mathcal{N} : \mathcal{A}$, then there is a decomposition
$$\mathcal{A} = \int_N \mathcal{A}_N d\mathcal{N}$$
such that for all N:
$$\dfrac{\vdash N : \mathcal{A}'_N \quad \left\{ \vdash M_2 : \mathcal{B} \,\middle|\, \forall (a \to C) \in SUPP\left(\mathcal{A}'_N\right),\ \forall \mathcal{B} \in a \right\}}{\vdash N\, M_2 : \displaystyle\int_{(a \to C)} C d\mathcal{A}'_N}$$
with $\mathcal{A}_N = \int_{(a \to C)} C d\mathcal{A}'_N$.
We set $\mathcal{A}' := \int_N \mathcal{A}'_N d\mathcal{N}$.
We have $SUPP\,(\mathcal{A}') \subseteq_f \bigcup_N SUPP\left(\mathcal{A}'_N\right)$, thus
$$\forall (a \to C) \in SUPP\,(\mathcal{A}')\,, \forall \mathcal{B} \in a, \quad \vdash M_2 : \mathcal{B}.$$
Moreover, by induction hypothesis $\vdash M'_1 : \mathcal{A}'$, thus we can apply rule ($\mathbf{MI}_N$-@) to get
$$\vdash M_1\, M_2 : \int_{(a \to C)} C d\mathcal{A}'.$$
we can conclude since:
$$\begin{aligned} \int_{(a \to C)} C d\mathcal{A}' &= \int_d C d \left(\int_N \mathcal{A}'_N d\mathcal{N} \right) (a \to C) \\ &= \int_N \left(\int_{(a \to C)} C d\mathcal{A}'_N \right) d\mathcal{N} && \text{Eq (1)} \\ &= \mathcal{A} \end{aligned}$$

□

10.2 Completeness

The $\mathbf{MI}_N$ system is only complete in the sens that the norm of the deviated type can approach the probability of termination with arbitrary precision. This is, however, the best notion of completion we can hope for a finite type system.

Recall that $\star := \emptyset \to \langle\rangle$.

Lemma 10.4 (Trivial derivation of values). *For any distribution $\mathcal{V}$ of values, we have $\vdash \mathcal{V} : (\sum \mathcal{V})\langle\star\rangle$.*

Proof. Using rules ($\mathbf{MI}_N$-λ) and ($\mathbf{MI}_N$-D), we get that

$$\forall V \in \mathit{SUPP}(\mathcal{V}), \quad \vdash V : \langle\star\rangle.$$

Then we just have to sum. □

Theorem 10.5 (Completeness). *For any term M, there is a sequence $(p_i)_{i\geq 0}$ such that*

$$\vdash M : p_i\langle\star\rangle \quad \text{and} \quad \bigvee_i p_i = \sum [\![M]\!].$$

Proof. By definition of the evaluation, there is $(\mathcal{M})_{i\geq 0}$ st

$$M \to^* \mathcal{M}_i \text{ and } [\![M]\!] = \textstyle\bigvee_i \mathcal{M}_i^V.$$

Let $p_i := \sum \mathcal{M}_i^V$, so that thus $\bigvee_i p_i = \sum [\![M]\!]$.
Since it is composed of values, we have $\vdash \mathcal{M}_i^V : p_i\langle\star\rangle$.
Moreover, we can use the rule (MI_N-D) to get

$$\vdash \mathcal{M}_i^R : \langle\rangle \quad \text{so that} \quad \vdash \mathcal{M}_i : p_i\langle\star\rangle.$$

By Theorem 10.3 we get $\vdash M : p_i\langle\star\rangle$ for all i. □

10.3 Soundness

Readability candidates are not defined as sets but by the following relations

Definition 10.6 (Reducibility candidates).

$$\begin{array}{lll}
V \vDash_V a \to \mathcal{A} & \text{iff} & \forall M \vDash_S a,\ (V\ M) \vDash_T \mathcal{A} \\
M \vDash_S a & \text{iff} & \forall \mathcal{A} \in a,\ M \vDash_T \mathcal{A} \\
\mathcal{V} \vDash_D \mathcal{A} & \text{iff} & \mathcal{V}\ \widetilde{\vDash_V}\ \mathcal{A} \\
M \vDash_T \mathcal{A} & \text{iff} & [\![M]\!] \vDash_D \mathcal{A} \\
\mathcal{M} \vDash_T \mathcal{A} & \text{iff} & [\![\mathcal{M}]\!] \vDash_D \mathcal{A}
\end{array}$$

We have to use this relation in place of sets

$$\mathsf{Red}_{\mathcal{A}} := \{M \mid M \vDash_T \mathcal{A}\}$$

in order to define $\vDash_D$ as the right-lax coupling of $\vDash_V$.

First, we give two trivial lemmas:

Lemma 10.7. *If $M \to \mathcal{M}$ then*

$$M \vDash_T \mathcal{A} \quad \textit{iff} \quad \mathcal{M} \vDash_T \mathcal{A}$$

Proof. Trivial since $[\![M]\!] = [\![\mathcal{M}]\!]$. □

Lemma 10.8. *If $V \vDash_V \alpha$, then $V \vDash_T \langle\alpha\rangle$.*

Proof. We have $\langle V\rangle \vDash_D \langle\alpha\rangle$ using the Dirac coupling

$$\mathcal{R} := \langle(V, \alpha)\rangle.$$

We conclude since $[\![V]\!] = \langle V\rangle$. □

Then, we can go with the soundness.

Theorem 10.9 (Soundness). *For any valid sequent $x_1 : a_1, \ldots, x_n : a_n \vdash M : \mathcal{A}$, and for any sequence $(L_i)_{i\leq n}$ such that for all $i \leq n, L_i \vDash_S a_i$, we have $M[\vec{L}/\vec{x}] \vDash_T \mathcal{A}$.*
In particular, if $\vdash M : \mathcal{A}$, then $\sum [\![M]\!] \geq \sum \mathcal{A}$.

Proof. The second assertion stands from the first using Lemma 9.4.
Let $\vec{L}$ such that for all $i \leq n$, $L_i \vDash_S a_i$.
We prove the first assertion by induction on the type derivation:

- ($\mathbf{MI}_N$-x):
  We assume that
  $$\Gamma, x : a_i \vdash x : \mathcal{A}, \quad \mathcal{A} \in a_i.$$
  Then by definition of $\vDash_S$, we have $x[L/x] = L \vDash_T \mathcal{A}$.
- ($\mathbf{MI}_N$-$\oplus$):
  We assume that
  $$\vec{x} : \vec{a} \vdash M : \mathcal{A}, \quad \vec{x} : \vec{a} \vdash N : \mathcal{B}.$$
  Let us write $M' := M[\vec{L}/\vec{x}]$ and $N' := N[\vec{L}/\vec{x}]$.
  By induction hypothesis, we know that
  $$M' \vDash_T \mathcal{A} \quad \text{and} \quad N' \vDash_T \mathcal{B}.$$
  This means that $[\![M']\!] \vDash_D \mathcal{A}$ and $[\![N']\!] \vDash_D \mathcal{B}$.
  By Lemma 9.2 we have
  $$(\frac{1}{2}[\![M']\!] + \frac{1}{2}[\![N']\!]) \vDash_D (\frac{1}{2}\mathcal{A} + \frac{1}{2}\mathcal{B}).$$
  By Lemma 10.7 we then have $M' \oplus N' \vDash_T (\frac{1}{2}\mathcal{A} + \frac{1}{2}\mathcal{B})$.
- ($\mathbf{MI}_N$-λ):
  We assume that
  $$\vec{x} : \vec{a}, y : b \vdash M : \mathcal{B}.$$
  We set $M' := M[\vec{L}/\vec{x}]$.
  By induction hypothesis, we know that
  $$\forall N \vDash_S b, \quad M'[N/y] \vDash \mathcal{B}.$$
  Thus, by Lemma 10.7,
  $$\forall N \vDash_S b, \quad ((\lambda y.M')\ N) \vDash_T \mathcal{B}.$$
  By definition of $\vDash_V$, this means that $\lambda y.M' \vDash_V b \to \mathcal{B}$, we conclude by Lemma 10.8.
- ($\mathbf{MI}_N$-@):
  We assume that $\vec{x} : \vec{a} \vdash M : \mathcal{A}$ and that
  $$\forall (b \to C) \in \mathit{SUPP}(\mathcal{A}), \forall \mathcal{B} \in b, \vec{x} : \vec{a} \vdash N' : \mathcal{B}.$$
  Let $(L_i \vDash_S a_i)_{i\leq n}$, $M' := M[\vec{L}/\vec{x}]$ and $N' := N'[\vec{L}/\vec{x}]$.
  By induction hypothesis, $M' \vDash_T \mathcal{A}$ and for all $(b \to C) \in \mathit{SUPP}(\mathcal{A})$ and all $\mathcal{B} \in b$, $N' \vDash_T \mathcal{B}$. By definition of $\vDash_S$, we have for all $(b \to C) \in \mathit{SUPP}(\mathcal{A})$, that $N' \vDash_S b$.
  From $M' \vDash_T \mathcal{A}$, we have that $[\![M']\!]\ \widetilde{\vDash_V}\ \mathcal{A}$. Let $\mathcal{R}$ the associated coupling.
  For any $(V, b \to C) \in \mathit{SUPP}(\mathcal{R})$, we have $V \vDash_V b \to C$, and thus $V\ N' \vDash_T C$, or equivalently $[\![V\ N']\!] \vDash_D C$.
  By summing over all $\mathcal{R}$, we have:
  $$\int_{(V,(b\to C))} [\![V\ N']\!]\, d\mathcal{R} \vDash_D \int_{(V,(b\to C))} C\, d\mathcal{R}$$

By Lemma 9.5, we can simplify this equation:

$$[\![[\![M']\!]\ N']\!] = \int_V [\![V\ N']\!]\, d\, [\![M']\!] \vDash_D \int_{(b\to C)} C d\mathcal{A}$$

We conclude by Lemma 3.3. □

11 Annex C: full proof of Theorem 6.6

11.1 Full system and single threaded version

First, let extend our type system with the rules of Figure 10.

There, $\sqsupseteq$ is the lifting $\widetilde{\supseteq}$ of the inclusion by a right-lax coupling, i.e., $\mathcal{A} \sqsubseteq \mathcal{B}$ if there exists a distribution $\mathcal{R} \in \mathfrak{D}(\sqsubseteq)$ such that $\mathcal{A}(a) \leq \sum_b \mathcal{R}(a,b)$ and $\mathcal{B}(b) = \sum_a \mathcal{R}(a,b)$.

We also extends the notion of stability to distributions of sets by saying that a distribution $\mathcal{A} \in \mathfrak{D}_f(\mathfrak{P}_f(\mathbb{M}_V))$ is stable if every set of its support is stable.

In this extension we did two main generalisation:

- We have separated $(MI_v\text{-}\lambda)$ into the two rules $(MI_v\text{-}\lambda')$ and $(MI_v\text{-}D)$, which basically means that rule $(MI_v\text{-}D)$ can now be applied anywhere, not just above abstractions.
- We have generalise $(MI_v\text{-}@)$ into $(MI_v\text{-}D@)$ so that the left part can accept distributions, this is mandatory when we consider that rule $(MI_v\text{-}D)$ could have been applied on the left hypothesis.

This extension has the particularity that any $(MI_v\text{-}D)$ can be pushed downward so that we recover the previous definition:

Definition 11.1. A type derivation in the system $\mathbf{MI}_V$ of Figure 8 is called single threaded.
A type derivation in the system $\mathbf{MI}_V$ of Figure 10 with a unique application of the rule $(MI_V\text{-}D)$ at the root followed by single-threaded derivations is called a distributive normal form.

Lemma 11.2. *Any type derivation in the system $\mathbf{MI}_V$ of Figure 8 can be rewritten into a distributive normal form.*

Proof. The idea is to propagate downward the extensions so that we only get rule $(MI_V\text{-}D)$ at the roots of the derivation or above occurrences of rule $(MI_v\text{-}\lambda')$ so that they enter into a rule $(MI_v\text{-}\lambda)$.
Excepts for $(MI_v\text{-}\lambda')$, rules are all linear, which means that rule $(MI_v\text{-}D)$ can be pushed downward until reaching the root or a rule $(MI_v\text{-}\lambda)$.
Remains to show that rules $(MI_v\text{-}D@)$ can be eliminated. Since there is no occurrence of rule $(MI_v\text{-}D)$, each occurrence of rule $(MI_v\text{-}D@)$ has the form:

$$\dfrac{\Gamma \vdash V : \langle a\rangle \qquad \Gamma \vdash N : \sum_i \mathcal{B}_i \qquad \mathcal{A}_i \in \mathfrak{D}(\downarrow_s a)}{\Gamma \vdash V\ N : \sum_i \iint_{c,b} \hat{c}(b) da_i d\mathcal{B}_i}\ (MI_V\text{-}D@)$$

Let $\mathcal{B}_c = \sum_i \mathcal{A}_i(c)\mathcal{B}_i$ and $\mathcal{B}_\emptyset = \sum_i(\mathcal{A}_i(\emptyset)+1-\sum_c \mathcal{A}_i(c))\mathcal{B}_i$, so that $\mathcal{B} = \sum_{c\in\downarrow_s a} \mathcal{B}_{i,c}$ and

$$\dfrac{\Gamma \vdash V : \langle a\rangle \qquad \Gamma \vdash N : \sum_{c\in\downarrow_s(a)} \mathcal{B}_c}{\Gamma \vdash V\ N : \sum_{c\in\downarrow_s(a)} \int_b \hat{c}(b) d\mathcal{B}_c}\ (MI_V\text{-}@)$$

$$\frac{a \supseteq b}{\Gamma; x : a \vdash x : \langle b \rangle}\ (MI_V\text{-}x) \qquad \frac{\Gamma \vdash M : \mathcal{A} \qquad \Gamma \vdash N : \mathcal{B}}{\Gamma \vdash M \oplus N : \frac{1}{2}\mathcal{A} + \frac{1}{2}\mathcal{B}}\ (MI_V\text{-}\oplus) \qquad \frac{\left\{\Gamma \vdash M : \mathcal{A}_u \,\middle|\, u \in U\right\} \qquad \mathcal{U} \in \mathfrak{D}(U)}{\Gamma \vdash M : \sum_u \mathcal{U}(u)\mathcal{A}'_u}\ (MI_V\text{-}D)$$

$$\frac{\left\{\Gamma; x : a_i \vdash M : \mathcal{B}_i \,\middle|\, i \leq n\right\}}{\Gamma \vdash \lambda x.M : \langle\{a_i \to \mathcal{B}_i \mid i \leq n\}\rangle}\ (MI_V\text{-}\lambda') \qquad \frac{\Gamma \vdash V : \mathcal{A} \qquad \Gamma \vdash N : \sum_i \mathcal{B}_i \qquad \mathcal{A}_i \sqsubseteq \mathcal{A} \text{ are stable}}{\Gamma \vdash V\, N : \sum_i \iint_{a,b} \hat{a}(b) d\mathcal{A}_i d\mathcal{B}_i}\ (MI_V\text{-}D@)$$

Figure 10. The Full Call-by-Value Monadic Intersection Type System $\mathbf{MI}_V^F$. In rule ($\mathbf{MI}_V$-λ) and (MI_V-D), the sets of sequents means that a proof has to be given for each of these sequents.

We conclude since :

$$\begin{aligned}
\sum_{c \in \downarrow_s(a)} \int_b \hat{c}(b) d\mathcal{B}_c &= \sum_{c \in \downarrow_s(a)} \int_b \hat{c}(b) d\left(\sum_{i \leq n} \mathcal{A}_i(c)\mathcal{B}_i\right) \\
&\quad + \int_b \hat{\emptyset}(b) d\left(\sum_i \left(1 - \sum_c \mathcal{A}_i(c)\right)\mathcal{B}_i\right) \\
&= \sum_{c \in \downarrow_s(a)} \int_b \hat{c}(b) d\left(\sum_{i \leq n} \mathcal{A}_i(c)\mathcal{B}_i\right) \\
&= \sum_{i \leq n} \sum_{c \in \downarrow_s(a)} \mathcal{A}_i(c) \int_b \hat{c}(b) d\mathcal{B}_i \\
&= \sum_{i \leq n} \iint_{c,b} \hat{c}(b) d\mathcal{A}_i d\mathcal{B}_i
\end{aligned}$$

□

In the following, we use (MI_v-@) when considering distributive normal forms and rule (MI_v-D@) when considering arbitrary derivation.

11.2 Saturation: Subject Reduction and Subject Expansion

First, we need a weakening lemma:

Lemma 11.3 (Weakening). *For any derivation $\Gamma, x{:}a \vdash M : \mathcal{A}$, and any $b \supseteq a$, we have a derivation $\Gamma, x{:}b \vdash M : \mathcal{A}$.*

Proof. By a trivial induction of the type derivation. □

We also use the fact that the types of values form an ideal:

Lemma 11.4 (Value-intersection). *For any two derivations $\Gamma \vdash V : \langle a \rangle$ and $\Gamma \vdash V : \langle b \rangle$ of a same value, we can perform the intersection $\Gamma \vdash V : \langle a \cup b \rangle$. Similarly, for any derivations $\Gamma \vdash V : \langle a \rangle$ and and $b \subseteq a$, there is a derivation of $\Gamma \vdash V : \langle b \rangle$.*

Proof. Trivial since values are all λ-abstractions. □

As usual, we prove a substitution lemma:

Lemma 11.5 (Substitution lemma). *For any Γ, M, V and $\mathcal{A}$, the two are equivalent:*

1. $\Gamma \vdash M[V/x] : \mathcal{A}$
2. *there is a such that $\Gamma, x : a \vdash M : \mathcal{A}$ and $\Gamma \vdash V : \langle a \rangle$.*

Proof. (1) ⇒ (2) : By induction on $\Gamma \vdash M[V/x] : \mathcal{A}$

- If $M = x$ then $\Gamma \vdash V : \mathcal{A}$, but since V is a value, we have $\mathcal{A} = \langle a \rangle$, which concludes.
- If $M = y$ for $x \neq y$ and $\Gamma \vdash y : b$ for some $(y : b') \in \Gamma$ and $b' \subseteq b$, then we just have to set $a = \langle\rangle$.
- If $M = M_1 \oplus M_2$ with $\Gamma \vdash M_i[V/x] : \mathcal{A}_i$ for all $i \leq 2$ and $\mathcal{A} = \frac{1}{2}\mathcal{A}_1 + \frac{1}{2}\mathcal{A}_2$; then by induction hypothesis there is a_1 and a_2 such that $\Gamma, x : a_1 \vdash M_1 : \mathcal{A}_1$ and $\Gamma, x : a_2 \vdash M_2 : \mathcal{A}_2$, with $\Gamma \vdash V : \langle a_1 \rangle$ and $\Gamma \vdash V : \langle a_2 \rangle$. We set $a = a_1 \cup a_2$. Then by Lemma 11.3, $\Gamma, x : a \vdash M_1 : \mathcal{A}_1$ and $\Gamma, x : a \vdash M_2 : \mathcal{A}_2$, and by Lemma 11.4, we have $\Gamma \vdash V : \langle a \rangle$. We conclude by using rule (MI_V-$\oplus$).
- If $\mathcal{A} = \int_u \mathcal{A}_u d\mathcal{U}$, and if there is a such that $\Gamma, x : a_u \vdash M : \mathcal{A}_u$ and $\Gamma \vdash V : \langle a_u \rangle$; then by Lemma 11.4, $\Gamma \vdash V : \langle a \rangle$ for $a = \bigcup_u a_u$ and by Lemma 11.4, we have $\Gamma, x : a \vdash M : \mathcal{A}_u$. Thus $\Gamma, x : a \vdash M : \mathcal{A}$ by rule (MI_V-D).
- If $M = \lambda y.M'$ with $\Gamma, y : b_i \vdash M'[V/x] : \mathcal{B}_i$ for all $i \leq n$ and $\mathcal{A} = \langle\{b_i \to \mathcal{B}_i\}\rangle$; then by induction hypothesis there is a_i such that $\Gamma, y : b_i, x : a_i \vdash M' : \mathcal{B}_i$ and $\Gamma \vdash V : \langle a_i \rangle$. Let $a = \bigcup_i a_i$. Then by Lemma 11.3, $\Gamma, x : a \vdash M' : \mathcal{B}_i$ for all $i \leq n$, and by Lemma 11.4, we have $\Gamma \vdash V : \langle a \rangle$. We conclude by using rule (MI_V-λ).
- If $M = M_1\ M_2$ with $\Gamma \vdash M_1[V/x] : \mathcal{B}_1$ and $\Gamma \vdash M_2[V/x] : \mathcal{B}_2$ such that $\mathcal{A} = \sum_i \iint_{b,b} \hat{a}(b) d\mathcal{B}_{1,i} d\mathcal{B}_{2,i}$ for some decomposition $\sum_i \mathcal{B}_{2,i} = \mathcal{B}_2$ and some $\mathcal{A}_i \sqsubseteq \mathcal{A}$; then by induction hypothesis there is a_1 and a_2 such that $\Gamma, x : a_1 \vdash M_1 : \mathcal{B}_1$ and $\Gamma, x : a_2 \vdash M_2 : \mathcal{B}_2$, with $\Gamma \vdash V : \langle a_1 \rangle$ and $\Gamma \vdash V : \langle a_2 \rangle$. We set $a = a_1 \cup a_2$. Then by Lemma 11.3, $\Gamma, x : a \vdash M_1 : \mathcal{B}_1$ and $\Gamma, x : a \vdash M_2 : \mathcal{B}_2$, and by Lemma 11.4, we have $\Gamma \vdash V : \langle a \rangle$. We conclude by using rule (MI_V-@).

(2) ⇒ (1) : By a trivial induction on $x : a \vdash M : \mathcal{A}$ □

This allows us to prove the saturation theorem, which is the subject reduction and expansion together:

Theorem 11.6 (Saturation). *For any M in Let-normal form, if $M \to \mathcal{N}$ then*

$$\vdash M : \mathcal{A} \qquad \textit{iff} \qquad \vdash \mathcal{N} : \mathcal{A}$$

Proof. By Lemma 11.2, we can consider that the starting derivation is single-threaded wlog.
By induction on $M \to \mathcal{N}$:

- Suppose that $M = (\lambda x.M')\, V$.
Then $\vdash M : \mathcal{A}$ means either that:

$$\dfrac{\dfrac{\left\{x : a_i \vdash M' : \mathcal{B}_i \;\middle|\; i \leq n\right\}}{\vdash \lambda x.M' : \langle\{a_i {\to} \mathcal{B}_i \mid i \leq n\}\rangle} \qquad \vdash V : \langle a\rangle}{\vdash (\lambda x.M')\, V : \sum_I p_I \bigvee_{i \in I \mid a_j \subseteq a} \mathcal{B}_i = \mathcal{A}}\,(MI_V\text{-}@)$$

where p_I's are summing bellow 1 and where the I's are ranging over all subsets $I \subseteq [0,n]$ such that $\{a_i {\to} \mathcal{B}_i \mid i \in I\}$ is stable.
Since the supps are finite, the results are reached, thus we get: $\mathcal{A} = \sum_i p_i \mathcal{B}_i$, where p_i's are summing bellow 1, so that we can write a distribution: $\mathcal{A} = \int_i \mathcal{B}_i d\mathcal{I}$.
Thus, using rule $(MI_V\text{-}D)$, we get that $x : a \vdash M' : \mathcal{A}$. And it is easy to see that it is an equivalence (by taking $n = 1$ and $p_n = 1$ for example). This shows that $\vdash (\lambda x.M')\, V : \mathcal{A}$ iff $x : a \vdash M' : \mathcal{A}$ and $\vdash V : \langle a\rangle$; by Lemma 11.5, this is equivalent to $\vdash M'[V/x] : \mathcal{A}$, i.e., to $\vdash \mathcal{N} : \mathcal{A}$.
- Suppose that $M = M_1 \oplus M_2 \to \frac{1}{2}\langle M_1\rangle + \frac{1}{2}\langle M_2\rangle = \mathcal{N}$.
Then $\vdash M : \mathcal{A}$ means either that:

$$\dfrac{\vdash M_1 : \mathcal{A}_1 \qquad \vdash M_2 : \mathcal{A}_2}{\vdash M_1 \oplus M_2 : \frac{1}{2}\mathcal{A}_1 + \frac{1}{2}\mathcal{A}_2 = \mathcal{A}}\,(MI_V\text{-}\oplus)$$

This is exactly equivalent to say that $\vdash \frac{1}{2}\langle M_1\rangle + \frac{1}{2}\langle M_2\rangle : \frac{1}{2}\mathcal{A}_1 + \frac{1}{2}\mathcal{A}_2 = \mathcal{A}$.
- Suppose that $M = V_1\, M_2$ and $M_2 \to \mathcal{N}'$ with $\mathcal{N} = V_1\, \mathcal{N}'$.
  - If $\vdash V_1\, M_2 : \mathcal{A}$, we can say that :

$$\dfrac{\vdash V_1 : \langle a\rangle \qquad \vdash M_2 : \sum_{c \in \downarrow_s a} \mathcal{B}_b}{\vdash V_1\, M_2 : \sum_{c \in \downarrow_s a} \int_b \hat{c}(b) d\mathcal{B}_c}\,(MI_V\text{-}@)$$

By induction hypothesis, $\vdash \mathcal{N}' : \sum_c \mathcal{B}_c$, which means that there is $(\mathcal{B}'_N)_N$ such that $\vdash N : \mathcal{B}'_N$ and $\sum_c \mathcal{B}_c = \int_N \mathcal{B}'_N d\mathcal{N}'$. We can then slice each distributions into $\mathcal{B}'_N = \sum_c \mathcal{B}_{N,c}$ such that $\mathcal{B}_c = \int_N \mathcal{B}_{N,c} d\mathcal{N}'$.
By applying $(MI_V\text{-}@)$, we get for all N:

$$\dfrac{\vdash V_1 : \langle a\rangle \qquad \vdash N : \sum_{c \in \downarrow_s a} \mathcal{B}_{N,c}}{\vdash V_1\, N : \sum_{c \in \downarrow_s a} \int_b \hat{c}(b) d\mathcal{B}_{N,c}}\,(MI_V\text{-}@)$$

so that we conclude since :

$$\begin{aligned}
&\int_N \left(\sum_{c \in \downarrow_s a} \int_b \hat{c}(b) d\mathcal{B}_{N,c}\right) d\mathcal{N}' \\
&= \sum_{c \in \downarrow_s a} \int_N \left(\int_b \hat{c}(b) d\mathcal{B}_{N,c}\right) d\mathcal{N}' \\
&= \sum_{c \in \downarrow_s a} \int_b \hat{c}(b) d\left(\int_N \mathcal{B}_{N,c} d\mathcal{N}'\right) \\
&= \sum_{c \in \downarrow_s a} \int_b \hat{c}(b) d\mathcal{B}_c\ .
\end{aligned}$$

  - Conversely, if $\vdash V_1\, \mathcal{N}' : \mathcal{A}$, then there is $(\vdash V_1\, N : \mathcal{A}_N)_N$ such that $\mathcal{A} = \int_N \mathcal{A}_N d\mathcal{N}'$. We can say that for all N:

$$\dfrac{\vdash V_1 : \langle a_N\rangle \qquad \vdash N : \sum_{c \in \downarrow_s a_N} \mathcal{B}_{N,c}}{\vdash V_1\, N : \sum_{c \in \downarrow_s a_N} \int_b \widehat{c}(b) d\mathcal{B}_{N,c} = \mathcal{A}_N}\,(MI_V\text{-}@)$$

By induction hypothesis, $\vdash M_2 : \int_N \sum_c \mathcal{B}_{N,c} d\mathcal{N}'$.
Moreover, we got $\vdash V_1 : \langle a_N\rangle$ for all N, thus, by Lemma 11.4, there is $\vdash V_1 : \langle a\rangle$ such that $a_N \subseteq a$ for all $N \in SUPP(\mathcal{N})$.
Then we get:

$$\dfrac{\vdash V_1 : \langle a\rangle \qquad \vdash N : \sum_{C \in \downarrow_s a} \int_N \mathcal{B}_{N,c} d\mathcal{N}'}{\vdash V_1\, N : \sum_{C \in \downarrow_s a} \int_b \widehat{c}(b) d\left(\int_N \mathcal{B}_{N,C} d\mathcal{N}'\right)}\,(MI_V\text{-}@)$$

which is equal to $\mathcal{A}$:

$$\begin{aligned}
&\sum_{C \in \downarrow_s a} \int_b \widehat{c}(b) d\left(\int_N \mathcal{B}_{N,C} d\mathcal{N}'\right) \\
&= \sum_{C \in \downarrow_s a} \int_N \left(\int_b \widehat{c}(b) d\mathcal{B}_{N,C}\right) d\mathcal{N}' \\
&= \int_N \left(\sum_{C \in \downarrow_s a} \int_b \widehat{c}(b) d\mathcal{B}_{N,C}\right) d\mathcal{N}' \\
&= \int_N \mathcal{A}_N d\mathcal{N}' \ = \mathcal{A}
\end{aligned}$$

$\square$

11.3 Soundness

Definition 11.7 (Reducibility candidates).

$$\begin{array}{lll} V \vDash_V a \to \mathcal{A} & \text{iff} & \forall W \vDash_S a,\ (V\ W) \vDash_T \mathcal{A} \\ V \vDash_S a & \text{iff} & \forall \alpha \in a,\ V \vDash_V \alpha \\ \mathcal{V} \vDash_D \mathcal{A} & \text{iff} & \mathcal{V}\ \widetilde{\vDash_S}\ \mathcal{A} \\ M \vDash_T \mathcal{A} & \text{iff} & [\![M]\!] \vDash_D \mathcal{A} \\ \mathcal{M} \vDash_T \mathcal{A} & \text{iff} & [\![\mathcal{M}]\!] \vDash_D \mathcal{A} \end{array}$$

Lemma 11.8. *If $M \to \mathcal{M}$ then*

$$M \vDash_T \mathcal{A} \quad \textit{iff} \quad \mathcal{M} \vDash_T \mathcal{A}$$

Proof. Trivial since $[\![M]\!] = [\![\mathcal{M}]\!]$. □

Lemma 11.9. *If $V \vDash_S a$, then $V \vDash_T \langle a \rangle$.*

Proof. We have $\langle V \rangle \vDash_D \langle a \rangle$ using the Dirac coupling $\mathcal{R} = \langle (V, a) \rangle$. We conclude since $[\![V]\!] = \langle V \rangle$. □

Lemma 11.10. *Realizability candidate are downward close in the following sens:*

- *if $V \vDash_S a$ and $b \subseteq a$ then $V \vdash_S a$,*
- *if $V \vdash_D \mathcal{A}$ and $\mathcal{B} \sqsubseteq \mathcal{B}$ then $V \vdash \mathcal{B}$,*
- *if $M \vdash_T \mathcal{A}$ and $\mathcal{B} \sqsubseteq \mathcal{B}$ then $M \vdash \mathcal{B}$.*

Proof. The first is trivial and the third directly follows from the second. For the second we have to use that $(\Vdash_D \circ \sqsupseteq)) = (\widetilde{\Vdash_S} \circ \widetilde{\supseteq}) = (\widetilde{\Vdash_S \circ \supseteq}) = (\widetilde{\Vdash_S}) = (\Vdash_D)$ with the second-last equality coming from the first statement. □

Lemma 11.11. *If a is stable and $V \vDash_S a$, then for all $W \vDash_S b$ we have $(V\ W) \vDash_T \hat{a}(b)$.*

Proof. Since $\hat{a}(b) := \bigvee \{\mathcal{A} \mid \exists b' \subseteq b, (b' \to \mathcal{A}) \in a\}$, and since this set is finite, we only have to prove that
for any $b' \subseteq b$ such that $(b' \to \mathcal{A}) \in a$, we have $(V\ W) \vDash_T \mathcal{A}$.
By Lemma 11.10, if $b' \subseteq b$, then $W \vDash_S b'$. Moreover, by definition of $\vDash_S$, if $(b' \to \mathcal{A}) \in a$, then $V \vDash_V b' \to \mathcal{A}$, which means that $(V\ W) \vDash_T \mathcal{A}$. □

Theorem 11.12 (Soundness). *For any valid sequent $x_1 : a_1, \ldots, x_n : a_n \vdash M : \mathcal{A}$, and for any sequence $(V_i)_{i \le n}$ such that for all $i \le n, V_i \vDash_S a_i$, we have $M[\vec{V}/\vec{x}] \vDash_T \mathcal{A}$.*

Proof. By induction on the type derivation:

- $(MI_V\text{-}x)$:
  If $\Gamma, x : a \vdash x : \langle b \rangle$ for $b \subseteq a$ and $V \vDash_S a$ then we show that $x[V/x] = V \vDash_T \langle b \rangle$.
  By Lemma 11.9 we have $V \vDash_T \langle a \rangle$ and we conclude by lemma Lemma 11.4.
- $(MI_V\text{-}\oplus)$:
  We assume that $\vec{x} : \vec{a} \vdash M : \mathcal{A}$, that $\vec{x} : \vec{a} \vdash N : \mathcal{B}$ and that $V_i \vDash_S a_i$.
  We use $M' := M[\vec{V}/\vec{x}]$ and $N' := N[\vec{V}/\vec{x}]$.
  The induction hypothesis give $M' \vDash_T \mathcal{A}$ and $N' \vDash_T \mathcal{B}$.
  This means that $[\![M']\!] \vDash_D \mathcal{A}$ and $[\![N']\!] \vDash_D \mathcal{B}$.
  By Lemma 9.2 we have
  $$(\tfrac{1}{2}[\![M']\!] + \tfrac{1}{2}[\![N']\!]) \vDash_D (\tfrac{1}{2}\mathcal{A} + \tfrac{1}{2}\mathcal{B}).$$
  By definition of $\vDash_T$ we then have
  $$M' \oplus N' \vDash_T (\tfrac{1}{2}\mathcal{A} + \tfrac{1}{2}\mathcal{B}).$$
- $(MI_V\text{-}\lambda')$:
  We assume that for all $j \le n$ we have
  $$\vec{x} : \vec{a}, y : b_j \vdash M : \mathcal{B}_j \text{ and } \forall i \le n, V_i \vDash_S a_i.$$
  Then for all $j \le m$ and all $W_j \vDash_S a_j$, we have
  $$M[V/x, W/y] \vDash_T \mathcal{B}_j.$$
  Thus for any such j and $W_j \vDash_S a_j$, by Lemma 11.8 give
  $$((\lambda y.M)[V/x]\ W) \vDash_T \mathcal{B}_j.$$
  This means that for any $j \le m$, we have
  $$(\lambda y.M)[V/x] \vDash_V a_j \to \mathcal{B}_j.$$
  From there we get
  $$(\lambda y.M)[V/x] \vDash_S \{a_j \to \mathcal{B}_j \mid j \le m\}.$$
  By Lemma 11.9 we can conclude that
  $$(\lambda y.M)[V/x] \vDash_T \langle \{b_j \to \mathcal{B}_j \mid j \le n\} \rangle.$$
- $(MI_V\text{-}D)$:
  We assume that $M[\vec{V}/\vec{x}] \vDash_T \mathcal{B}_u$ for $\mathcal{A} = \int_u \mathcal{B}_u d\mathcal{U}$.
  Then $\left[\!\left[M[\vec{V}/\vec{x}]\right]\!\right] \vDash_D \mathcal{B}_u$ by definition of $\vDash_T$.
  Thus $\left[\!\left[M[\vec{V}/\vec{x}]\right]\!\right] \vDash_D \mathcal{A}$ by sub-linearity of the coupling relation (Lemma 9.3).
  We conclude $M[\vec{V}/\vec{x}] \vDash_D \mathcal{A}$ by definition of $\vDash_T$.
- $(MI_V\text{-}D@)$:
  We assume that $V_i \vDash_S a_i$ for all $i \le n$, and that
  $$\frac{\Gamma \vdash M : \mathcal{A} \qquad \Gamma \vdash N : \sum_i \mathcal{B}_i \qquad \mathcal{A}_i \sqsubseteq \mathcal{A} \text{ are stable}}{\Gamma \vdash M\ N : \sum_i \iint_{a,b} \hat{a}(b) d\mathcal{A}_i d\mathcal{B}_i}$$
  Let us write
  $$M' := M[\vec{x}/\vec{N}] \text{ and } N' := N[\vec{x}/\vec{N}].$$
  By induction hypothesis,
  $$M' \vDash_T \mathcal{A} \text{ and } N' \vDash_T \textstyle\sum_i \mathcal{B}_i.$$
  Using Lemma 11.10, we get
  $$M' \vDash_T \mathcal{A}_i \text{ for all } i.$$
  Let $\mathcal{R}$ be the coupling distribution of $[\![N']\!] \vDash_D \sum \mathcal{B}_i$.
  Let $\mathcal{N}_i := \int_\beta \mathcal{R}(\_, \beta) d\mathcal{B}_i$ so that
  $$\mathcal{N}_i \vDash_D \mathcal{B}_i \text{ and } [\![N']\!] = \textstyle\sum_i \mathcal{N}_i.$$
  By linearity of the lax coupling (Lemma 9.2) and by Lemma 11.11, we have for all i:
  $$[\![[\![M']\!]\ \mathcal{N}_i]\!] \vDash_D \iint_{a,b} \hat{a}(b) d\mathcal{A}_i d\mathcal{B}_i.$$
  By a second application of the linearity, we have:
  $$[\![[\![M']\!]\ [\![N']\!]]\!] \vDash_D \sum_i \iint_{a,b} \hat{a}(b) d\mathcal{A}_i d\mathcal{B}_i.$$
  We conclude by Lemma 3.3.

□

11.4 Completeness

The completeness of $\mathbf{MI}_V$ has to be proven after its soundness because we need the soundness of $\mathbf{MI}_V$ in order to reduce the completeness of $\mathbf{MI}_V^F$ to the completeness of $\mathbf{MI}_V$.

Once again, $\mathbf{MI}_V$ is only complete in the sens that the norm of the deviated type can approach the probability of termination with arbitrary precision.

Theorem 11.13 (Completeness). *For any term M in let-normal form, there is a sequence $(p_i)_{i\geq 0}$ such that $\vdash M : p_i\langle\emptyset\rangle$ and $\bigvee_i p_i = \sum [\![M]\!]$ in system $\mathbf{MI}_V$.*

Proof. First, we prove the statement in system $\mathbf{MI}_V^F$.

There is $(\mathcal{M})_{i\geq 0}$ such that $M \to^* \mathcal{M}_i$ and $[\![M]\!] = \bigvee_i \mathcal{M}_i^V$. Since it is composed of values, we have $\vdash \mathcal{M}_i^V : p_i\langle\emptyset\rangle$ for $p_i := \sum \mathcal{M}_i^V$; thus $\bigvee_i p_i = \sum [\![M]\!]$. Moreover, we can use the rule $(MI_V\text{-}\langle\rangle)$ to get $\vdash \mathcal{M}_i^R : \langle\rangle$ and $\vdash \mathcal{M}_i : p_i\langle\emptyset\rangle$. By Saturation (Theorem 11.6) we get $\vdash M : p_i\langle\emptyset\rangle$ for all i.

Now that we have a sequence of derivations in $\mathbf{MI}_V^F$, we can take their distributive normal forms. It is then easy to see that the rule $(MI_v\text{-}D)$ at the root has an argument verifying $\vdash M : q_i\langle\star\rangle$ for $p_i \leq q_i \leq [\![M]\!]$, i.e., $\bigvee_i p_i = \sum [\![M]\!]$:
We assume that

$$\frac{\left\{\vdash M : \mathcal{A}_u \;\middle|\; u \in U\right\} \qquad \mathcal{U} \in \mathfrak{D}(U)}{\vdash M : \int_u \mathcal{A}_u d\mathcal{U} = p_i\langle\emptyset\rangle}\ (MI_V\text{-} \geq)$$

Wlog, we can also assume that $U = SUPP\,(\mathcal{U})$.
Then for all $u \in U$, we have $\mathcal{A}_u = q_u\langle\star\rangle$ for a certain $(q_u)_{u\in U}$ such that $p_i = \int_u q_u d\mathcal{U}$.
Let $q_i = \bigvee_u q_u$, then

$$p_i = \int_u q_u d\mathcal{U} \leq \int_u q_i d\mathcal{U} \leq q_i\ .$$

We conclude since $q_i \geq [\![M]\!]$ by Theorem 11.12. □