Realizability Models for a Linear Dependent PCF

Aloïs Brunel[a], Marco Gaboardi[b]

[a] *LIPN - UMR CNRS 7030 - Université Paris 13, Villetaneuse, France*
[b] *School of Computing - University of Dundee, Scotland*

Abstract

Recently, Dal Lago and Gaboardi have proposed a type system, named dℓPCF as a framework for implicit computational complexity. dℓPCF is a non-standard type system for PCF programs which is relatively complete with respect to quantitative properties thanks to the use of linear types inspired by Bounded linear logic and dependent types *à la* Dependent ML. In this work, we adapt the framework of quantitative realizability and obtain a model for dℓPCF. The quantitative realizability model aims at a better understanding of dℓPCF type decorations and at giving an abstract semantic proof of intensional soundness.

Keywords: Implicit Computational Complexity, Realizability model,

1. Introduction

Implicit computational complexity and resource consumption - The main goal of implicit computational complexity is to provide new characterizations of complexity classes that are abstract with respect to complexity measures and the underlying concrete models of computation. Besides, the tools developed in this area are useful to understand and analyze the resource consumption of programs.

Within this perspective, we can identify two approaches that have been proved fruitful so far.

(i) Within the first approach one identifies some *a priori restriction* on the shape of programs, in the form of a syntactic criterion or typing relation, that ensures programs respecting the restriction to be in a given complexity class $\mathcal{C}$.

(ii) Within the second approach one identifies some *a posteriori criteria*, in the form of a static program analysis, that ensures programs passing the criteria to be in a given complexity class $\mathcal{C}$.

The two approaches have their *pros* and *cons*. On the one hand, by following the first approach one aims at *designing* languages that characterize complexity classes. The positive aspect of this approach is that it is compositional and the generated constraints are usually easy to check [4, 22, 15, 3]; these constraints however cut off many interesting programs and so this approach gives systems suffering from poor intensional expressivity [15].

For the second approach, once one fixes the programming language (usually a first order language), one does not impose a priori restrictions on the shape of programs that can be considered. The restriction are *checked* on the specific programs by static analysis methods. This means that usually these methods give characterizations of complexity classes with a better intensional expressivity even if they too cut some interesting programs [19, 23, 25]. However, the analyses used in this setting are usually non-compositional and involve more complex verifications [1].

Relative completeness and $\mathsf{d}\ell\mathsf{PCF}$ **-** Recently, Dal Lago and Gaboardi in [8, 9] proposed a type system, named $\mathsf{d}\ell\mathsf{PCF}$, for a call-by-name version of PCF that combines the two approaches and pushes these ideas to the limit. $\mathsf{d}\ell\mathsf{PCF}$ can be considered as a non-standard type system that unifies type inference and program analysis. This type system explores ideas combining linear types inspired by Bounded linear logic (BLL) [13, 10] with dependent types *à la* Dependent ML [28]. That is, the types contain *index terms*—first-order terms drawn from a specific signature and whose meaning is given by means of an equational rewriting system—that are useful to reduce the problem of resource consumption of PCF terms to the problem of satisfying a set of inequalities. In order to achieve this goal, typing judgements in $\mathsf{d}\ell\mathsf{PCF}$ enrich PCF typing by means of additional information. An example of typing judgment in $\mathsf{d}\ell\mathsf{PCF}$ is the following:

$$\phi; \Phi; \Gamma \vdash_{\mathrm{I}}^{\mathcal{E}} t : [a < \mathrm{J}] \cdot \sigma \multimap \mathtt{Nat}[\mathrm{K}, \mathrm{H}]$$

In $\mathsf{d}\ell\mathsf{PCF}$, Φ is a set of inequalities over index terms which represent the side conditions under which the typing judgment is derivable, $\mathcal{E}$ is the equational theory giving meaning to index terms. The index term I describes the *weight* associated with the derivation of the typing judgment. The weight can be seen as an abstraction of the complexity of the program. The notation $[a < \mathrm{J}] \cdot \sigma$ is used instead of the more standard BLL notation $!_{a<\mathrm{J}}\sigma$ and K, H are index terms used to add some information to the base type `Nat` ensuring that the result of the computation is in the interval given by K and H. Finally, ϕ contains the free variables of all the index terms appearing in the typing judgement.

The main property of $\mathsf{d}\ell\mathsf{PCF}$ is *relative completeness*. This means that for *every* terminating PCF program there is a decoration in $\mathsf{d}\ell\mathsf{PCF}$ giving information about its complexity. So, the type system is able to analyze all the (terminating)

PCF terms with no a priori nor a posteriori restriction. The price to pay is that the typability is no longer decidable. More concretely, dℓPCF reduces typability to constraint satisfiability for constraints that are not in general decidable. This suggests that dℓPCF should not be considered a type system but instead it should be considered a general framework useful to study resource consumption and to compare different type systems for implicit computational complexity.

Besides being relative complete, dℓPCF is also intensionally sound. Anticipating on the next section, we can formulate intensional soundness as follows.

Theorem 1 (Intensional soundness)**.**
Let $\vdash_{\mathrm{I}} t : \mathtt{Nat}[\mathrm{J}, \mathrm{K}]$ *and suppose that* t *evaluates to the value* $\underline{\mathtt{m}}$ *in* n *steps. Then,* $n \leqslant |t| \cdot (\llbracket \mathrm{I} \rrbracket + 1)$ *and* $\llbracket \mathrm{J} \rrbracket \leqslant m \leqslant \llbracket \mathrm{K} \rrbracket$.

The theorem above relates typing judgments, and in particular index terms, with both the intensional and the extensional semantics of the program. That is, given a program t of PCF typable in dℓPCF with weight I and type $\mathtt{Nat}[\mathrm{J}, \mathrm{K}]$, the weight I gives a bound on the number n of evaluation steps while the index terms J and K give a bound on the value $\underline{\mathtt{m}}$ computed by t. The intensional soundness has been proved in [8, 9] by considering typed configurations of a call-by-name Krivine machine. More precisely, the proof in [8, 9] proceeds by assigning to every machine configuration C a weight I (corresponding to the weight of the type derivation of the configuration) and by showing that a reduction step transform a configuration C with weight I in a configuration D with weight J such that $\llbracket \mathrm{I} \rrbracket \geqslant \llbracket \mathrm{J} \rrbracket$. The intensional soundness then follows by noticing that substitution steps ensure moreover that $\llbracket \mathrm{I} \rrbracket > \llbracket \mathrm{J} \rrbracket$, and that configuration sizes provide a bound for the number of reduction steps between two substitutions.

Quantitative realizability models - Realizability, firstly introduced by Kleene, has been studied in different forms. In a series of recent works, Dal Lago and Hofmann have shown how to adapt Realizability to build *quantitative models* for subsystems of linear logic with restricted complexity. The models they propose contain a natural notion of resource (in the form of elements drawn from a resource monoid) that can be exploited to obtain semantic proofs of complexity soundness for these logics. Their models also permit to obtain a better understanding of the resource usage in different logics. Starting from Dal Lago and Hofmann results, the first author in [7] has extended the quantitative approach to Krivine's Realizability, a version of realizability introduced by Krivine that aims at extending the proofs-as-programs correspondence to classical logic and set theory. He has also shown how the quantitative aspects relates to the notion of *forcing* from set theory.

Contributions - In this work, we adapt the framework of *quantitative realizability* to dℓPCF. In particular, we follow the approach developed by Brunel in [7] and we

design a quantitative realizability model based on Krivine's Realizability.

In order to deal with the generality of $\mathsf{d}\ell\mathsf{PCF}$ typing judgments, we need to define the realizability interpretation $|\sigma|^{\mathcal{E}}_{\rho}$ of a type σ as *parametrized* over an equational program $\mathcal{E}$ and an assignment ρ of index term variables to natural numbers. In this way, we can internalize the generality of the type system in the model.

The interpretation we define is *sound* with respect to $\mathsf{d}\ell\mathsf{PCF}$ typing judgments. Moreover, we show two forms of completeness of our class of quantitative models with respect to $\mathsf{d}\ell\mathsf{PCF}$: *external completeness* and *internal completeness*. External completeness corresponds to a bounded-time termination property of realizers. Internal completeness gives quantitative information about the computed values and gives a concrete bound on the termination of realizers. Thanks to both external and internal completeness we give an abstract proof of $\mathsf{d}\ell\mathsf{PCF}$'s intensional soundness. Besides, using the relative completeness of $\mathsf{d}\ell\mathsf{PCF}$ we also prove that the type system and the realizability model carry exactly the same information when a universal equational program is considered. This last result represents a form of full abstraction and ensures that our model can be used in full generality to reason about $\mathsf{d}\ell\mathsf{PCF}$ programs.

The motivations for our study are twofold. On the one hand, we aim at better understanding the role of $\mathsf{d}\ell\mathsf{PCF}$ type decorations and the quantitative realizability model is the most natural candidate. On the other hand what we aim to obtain is a framework allowing us to extend by means of quantitative information some recent results obtained in the framework of complexity preserving certification [17].

Our study draws the attention on two roles played by index terms. First, the interpretation of modal types of the shape $[a < \mathrm{J}]\cdot\sigma$ closely resembles the standard interpretation of universal quantifiers in realizability models. This emphasizes the quantifier nature, on a bounded domain, of modal types with respect to index term variables. Second, in the soundness proof, the index term corresponding to the weight of a typing derivation can be used as a measure that is useful in proving the soundness of the fixpoint rule. This suggests that the weight plays a role similar to techniques like semantics approximants and step-indexing.

Outline - In Section 2 and Section 3, we recall respectively the system $\mathsf{d}\ell\mathsf{PCF}$ and its main properties. In Section 4, we introduce our quantitative realizability model and we show that it is sound with respect to $\mathsf{d}\ell\mathsf{PCF}$. In Section 5, we use the model defined in the previous section to give a semantic proof of $\mathsf{d}\ell\mathsf{PCF}$'s intensional soundness; moreover we show that when a universal equational program is considered, realizability and typing coincide. Finally, in Section 6, we recast some related works and conclude.

2. dℓPCF

The type system dℓPCF is a refinement of the type system for PCF by means of index terms whose semantics, in turn, is defined by an equational program over a signature Σ.

Formally, a *signature* Σ is a pair $(\mathcal{S}, \alpha)$ where $\mathcal{S}$ is a finite set of *function symbols* and $\alpha : \mathcal{S} \to \mathbb{N}$ assigns an *arity* to every function symbol. Index terms on a given signature $\Sigma = (\mathcal{S}, \alpha)$ are generated by the following grammar:

$$\mathrm{I}, \mathrm{J}, \mathrm{K} ::= a \mid \mathtt{f}(\mathrm{I}_1, \ldots, \mathrm{I}_{\alpha(\mathtt{f})}) \mid \sum_{a<\mathrm{I}} \mathrm{J} \mid \bigtriangleup_{a}^{\mathrm{I},\mathrm{J}} \mathrm{K}$$

where $\mathtt{f} \in \mathcal{S}$ and a is a variable drawn from a set $\mathcal{V}$ of *index variables*. We assume the symbols 0, 1 (with arity 0) and $+$, $\dot{-}$ (with arity 2) are always part of Σ. An index term in the form $\sum_{a<\mathrm{I}} \mathrm{J}$ is a *bounded sum*, while one in the form $\bigtriangleup_{a}^{\mathrm{I},\mathrm{J}} \mathrm{K}$ is a *forest cardinality*. .

Index terms are meant to denote natural numbers, possibly depending on the (unknown) values of variables. Variables can be instantiated with other index terms, e.g. $\mathrm{I}\{\mathrm{J}/a\}$. So, index terms can also act as first order functions. The meaning of the function symbols from Σ is induced by an equational program $\mathcal{E}$. Formally, an *equational program* $\mathcal{E}$ over a signature Σ is a set of equations in the form $t = s$ where both t and s are terms built from variables and the symbols in Σ. We are interested in equational programs guaranteeing that, whenever symbols in Σ are interpreted as partial functions over $\mathbb{N}$ and 0, 1, $+$ and $\dot{-}$ are interpreted in the usual way, the semantics of any function symbol $\mathtt{f}$ can be uniquely determined from $\mathcal{E}$. This can be guaranteed by, for example, taking $\mathcal{E}$ as an Herbrand-Gödel scheme or as an orthogonal constructor term rewriting system.

A bounded sum $\sum_{a<\mathrm{I}} \mathrm{J}$ is an index term whose value is simply the sum of all possible values of J with a taking the values from 0 up to I, excluded.

Forest cardinalities will be used to describe function call trees. Informally, $\bigtriangleup_{a}^{\mathrm{I},\mathrm{J}} \mathrm{K}$ is an index term denoting the number of nodes in a forest composed of J trees described using K. All the nodes in the forest are (uniquely) identified by natural numbers. These are obtained by consecutively visiting each tree in pre-order, starting from I. The term K has the role of describing the number of children of each forest node n by properly instantiating the variable a, e.g the number of children of the node $\mathtt{0}$ is $\mathrm{K}\{\mathtt{0}/a\}$. Formally, the meaning of a forest cardinality is defined by the following two equations:

$$\bigtriangleup_{a}^{\mathrm{I},0} \mathrm{K} = 0 \qquad \bigtriangleup_{a}^{\mathrm{I},\mathrm{J}+1} \mathrm{K} = \left(\bigtriangleup_{a}^{\mathrm{I},\mathrm{J}} \mathrm{K}\right) + 1 + \left(\bigtriangleup_{a}^{\mathrm{I}+1+\bigtriangleup_{a}^{\mathrm{I},\mathrm{J}} \mathrm{K},\mathrm{K}\{\mathrm{I}+\bigtriangleup_{a}^{\mathrm{I},\mathrm{J}} \mathrm{K}/a\}} \mathrm{K}\right)$$

and it represents the number (i.e. the cardinality) of nodes in the forest described by K.

The expression $[\![\mathrm{I}]\!]^{\mathcal{E}}_{\rho}$ denotes the meaning of I, defined by induction along the lines of the previous discussion, where $\rho : \mathcal{V} \to \mathbb{N}$ is an assignment and $\mathcal{E}$ is an equational program giving meaning to the function symbols in I. Since $\mathcal{E}$ does not necessarily interpret such symbols as *total* functions, and moreover, the value of a forest cardinality can be undefined, $[\![\mathrm{I}]\!]^{\mathcal{E}}_{\rho}$ can be undefined itself. A *constraint* is an inequality in the form $\mathrm{I} \leqslant \mathrm{J}$. A constraint is *true* in an assignment ρ if $[\![\mathrm{I}]\!]^{\mathcal{E}}_{\rho}$ and $[\![\mathrm{J}]\!]^{\mathcal{E}}_{\rho}$ are both defined and the first one is smaller or equal than the second one. Now, for a subset ϕ of $\mathcal{V}$, and for a set Φ of constraints involving variables in ϕ, the expression $\phi; \Phi \models^{\mathcal{E}} \mathrm{I} \leqslant \mathrm{J}$ denotes the fact that the truth of $\mathrm{I} \leqslant \mathrm{J}$ semantically follows from the truth of the constraints in Φ. The expression $\phi; \Phi \models^{\mathcal{E}} \mathrm{I} = \mathrm{I}$ indicates that (the semantics of) I is *defined* for the relevant values of the variables in ϕ under the constraints in Φ; this is usually written as $\phi; \Phi \models^{\mathcal{E}} \mathrm{I} \Downarrow$. Finally, we say that ρ *satisfies* $\phi; \Phi$ and we note $\rho \models^{\mathcal{E}} \phi; \Phi$ if $\phi \subseteq dom(\rho)$ and for each $\mathrm{I} \leqslant \mathrm{J} \in \Phi$, we have $[\![\mathrm{I}]\!]^{\mathcal{E}}_{\rho} \leqslant [\![\mathrm{J}]\!]^{\mathcal{E}}_{\rho}$.

From now on, all the definitions will be parametric on an equational program $\mathcal{E}$ over a signature Σ. For the sake of simplicity, we will often avoid to mention $\mathcal{E}$ explicitly. Terms are generated by the following grammar:

$$t, u, v ::= x \mid \underline{\mathtt{n}} \mid \mathtt{s}(t) \mid \mathtt{p}(t) \mid \lambda x.t \mid tu \mid \ \mathtt{ifz}\ t\ \mathtt{then}\ u\ \mathtt{else}\ v \mid \mathtt{fix}\ x.t$$

where $\mathtt{n}$ ranges over natural numbers and x ranges over a set of *variables*. The structure of terms should be familiar to the reader acquainted with PCF. A *w*eak-head reduction relation $\to$ on terms can be easily defined. A term t is said to be a *program* if it can be given the PCF type $\mathtt{Nat}$ in the empty context. A notion of *size* $|t|$ for a term t will be useful in the sequel. This can be defined as follows:

$$\begin{aligned}
|x| &= 1; & |\lambda x.t| &= |t| + 1;\\
|\underline{\mathtt{n}}| &= 1; & |tu| &= |t| + |u| + 1;\\
|\mathtt{s}(t)| &= |t| + 2; & |\ \mathtt{ifz}\ t\ \mathtt{then}\ u\ \mathtt{else}\ v| &= |t| + |u| + |v| + 1;\\
|\mathtt{p}(t)| &= |t| + 2; & |\mathtt{fix}\ x.t| &= |t| + 1.
\end{aligned}$$

Notice that for technical reasons size is defined in a slightly nonstandard way: every integer constant has size 1.

Lemma 1. *If t is a term and u is a subterm of t, then $|u| \leqslant |t|$.*

The language presented so far is the same as PCF. What distinguish dℓPCF from PCF is its type system. Basic and modal types are defined as follows:

$$\begin{aligned}
\sigma, \tau, \gamma &::= \mathtt{Nat}[\mathrm{I}, \mathrm{J}] \mid A \multimap \sigma && \text{basic types}\\
A, B &::= [a < \mathrm{I}] \cdot \sigma && \text{modal types}
\end{aligned}$$

$$\frac{\phi;\Phi \models^{\mathcal{E}} \mathrm{K} \leqslant \mathrm{I} \quad \phi;\Phi \models^{\mathcal{E}} \mathrm{J} \leqslant \mathrm{H}}{\phi;\Phi \vdash^{\mathcal{E}} \mathtt{Nat}[\mathrm{I},\mathrm{J}] \sqsubseteq \mathtt{Nat}[\mathrm{K},\mathrm{H}]}\ (\mathtt{Nat}.l) \qquad \frac{\phi;\Phi \vdash^{\mathcal{E}} B \sqsubseteq A \quad \phi;\Phi \vdash^{\mathcal{E}} \sigma \sqsubseteq \tau}{\phi;\Phi \vdash^{\mathcal{E}} A \multimap \sigma \sqsubseteq B \multimap \tau}\ (\multimap .l)$$

$$\frac{\phi, a;\Phi, a < \mathrm{J} \vdash^{\mathcal{E}} \sigma \sqsubseteq \tau \quad \phi;\Phi \models^{\mathcal{E}} \mathrm{J} \leqslant \mathrm{I}}{\phi;\Phi \vdash^{\mathcal{E}} [a < \mathrm{I}] \cdot \sigma \sqsubseteq [a < \mathrm{J}] \cdot \tau}\ ([-] \cdot .l)$$

Figure 1: The subtyping relation

where I, J range over index terms and a ranges over index variables. We will write the symbol ς when we want to talk about types without distinguishing between basic and modal types. $\mathtt{Nat}[\mathrm{I}]$ is syntactic sugar for $\mathtt{Nat}[\mathrm{I},\mathrm{I}]$. We will use the convention that $[a < \mathrm{I}] \cdot -$ has precedence over $\multimap$, e.g. $[a < \mathrm{I}] \cdot \sigma \multimap \tau$ stands for $([a < \mathrm{I}] \cdot \sigma) \multimap \tau$.

In the typing rules, modal types need to be manipulated in an algebraic way. For this reason, two operations on modal types need to be introduced. The first one is a binary operation $\uplus$ on modal types. Suppose that $A = [a < \mathrm{I}] \cdot \sigma\{a/c\}$ and that $B = [b < \mathrm{J}] \cdot \sigma\{\mathrm{I} + b/c\}$. In other words, A consists of the first I instances of σ, i.e. $\sigma\{0/c\}, \ldots, \sigma\{\mathrm{I} - \mathbf{1}/c\}$ while B consists of the next J instances of σ, i.e. $\sigma\{\mathrm{I} + 0/c\}, \ldots, \sigma\{\mathrm{I} + \mathrm{J} - \mathbf{1}/c\}$. Their *sum* $A \uplus B$ is naturally defined as a modal type consisting of the first $\mathrm{I} + \mathrm{J}$ instances of σ, i.e. $[c < \mathrm{I} + \mathrm{J}] \cdot \sigma$. An operation of bounded sum on modal types can be defined by generalizing the idea above: suppose that

$$A = [b < \mathrm{J}] \cdot \sigma\{\sum_{d<a} \mathrm{J}\{d/a\} + b/c\}.$$

Then its *bounded sum* $\sum_{a<\mathrm{I}} A$ is $[c < \sum_{a<\mathrm{I}} \mathrm{J}] \cdot \sigma$.

Central to dℓPCF is the notion of subtyping. An inequality relation $\sqsubseteq$ between (basic and modal) types can be defined using the formal system in Figure 1. This relation corresponds to lifting index inequalities to the type level. The equality $\phi;\Phi \vdash \sigma \cong \tau$ holds when both $\phi;\Phi \vdash \sigma \sqsubseteq \tau$ and $\phi;\Phi \vdash \tau \sqsubseteq \sigma$ can be derived from the rules in Figure 1.

Typing judgements of dℓPCF are expressions in the form

$$\phi;\Phi;\Gamma \vdash^{\mathcal{E}}_{\mathrm{I}} t : \sigma \tag{1}$$

where Γ is a *typing context*. That is, a set of term variable assignments of the shape $x : A$ where each variable x occurs at most once. The expression (1) can be informally read as follows: for every values of the index variables in ϕ satisfying

$$\frac{\phi;\Phi \models^{\mathcal{E}} \mathrm{J} \geqslant 0 \quad \phi;\Phi \vdash^{\mathcal{E}} [a<\mathrm{I}]\cdot\sigma \sqsubseteq [a<\mathbf{1}]\cdot\tau}{\phi;\Phi;\Gamma, x:[a<\mathrm{I}]\cdot\sigma \vdash^{\mathcal{E}}_{\mathrm{J}} x:\tau\{0/a\}}\ V \qquad \frac{\phi;\Phi;\Gamma, x:[a<\mathrm{I}]\cdot\sigma \vdash^{\mathcal{E}}_{\mathrm{J}} t:\tau}{\phi;\Phi;\Gamma \vdash^{\mathcal{E}}_{\mathrm{J}} \lambda x.t:[a<\mathrm{I}]\cdot\sigma \multimap \tau}\ L$$

$$\frac{\phi;\Phi;\Gamma \vdash^{\mathcal{E}}_{\mathrm{J}} t:[a<\mathrm{I}]\cdot\sigma \multimap \tau \quad \phi,a;\Phi,a<\mathrm{I};\Delta \vdash^{\mathcal{E}}_{\mathrm{K}} u:\sigma \quad \phi;\Phi \vdash^{\mathcal{E}} \Sigma \sqsubseteq \Gamma \uplus \sum_{a<\mathrm{I}} \Delta}{\phi;\Phi;\Sigma \vdash^{\mathcal{E}}_{\mathrm{J}+\sum_{a<\mathrm{I}}\mathrm{K}+\mathrm{I}} tu:\tau}\ A \qquad \frac{\phi;\Phi;\Gamma \vdash^{\mathcal{E}}_{\mathrm{K}} t:\mathtt{Nat}[\mathrm{I},\mathrm{J}] \quad \phi;\Phi,\mathrm{I}\leqslant 0;\Delta \vdash^{\mathcal{E}}_{\mathrm{H}} u:\sigma \quad \phi;\Phi,\mathrm{J}\geqslant 1;\Delta \vdash^{\mathcal{E}}_{\mathrm{H}} v:\sigma \quad \phi;\Phi \vdash^{\mathcal{E}} \Sigma \sqsubseteq \Gamma \uplus \Delta}{\phi;\Phi;\Sigma \vdash^{\mathcal{E}}_{\mathrm{K}+\mathrm{H}} \mathtt{ifz}\ t\ \mathtt{then}\ u\ \mathtt{else}\ v:\sigma}\ F$$

$$\frac{\phi;\Phi \vdash^{\mathcal{E}} \mathtt{Nat}[\mathrm{I}+1,\mathrm{J}+1] \sqsubseteq \mathtt{Nat}[\mathrm{K},\mathrm{H}] \quad \phi;\Phi;\Gamma \vdash^{\mathcal{E}}_{\mathrm{L}} t:\mathtt{Nat}[\mathrm{I},\mathrm{J}]}{\phi;\Phi;\Gamma \vdash^{\mathcal{E}}_{\mathrm{L}} \mathtt{s}(t):\mathtt{Nat}[\mathrm{K},\mathrm{H}]}\ S \qquad \frac{\phi;\Phi \vdash^{\mathcal{E}} \mathtt{Nat}[\mathrm{I}\dot{-}1,\mathrm{J}\dot{-}1] \sqsubseteq \mathtt{Nat}[\mathrm{K},\mathrm{H}] \quad \phi;\Phi;\Gamma \vdash^{\mathcal{E}}_{\mathrm{L}} t:\mathtt{Nat}[\mathrm{I},\mathrm{J}]}{\phi;\Phi;\Gamma \vdash^{\mathcal{E}}_{\mathrm{L}} \mathtt{p}(t):\mathtt{Nat}[\mathrm{K},\mathrm{H}]}\ P$$

$$\frac{\phi;\Phi \models^{\mathcal{E}} \mathrm{K} \geqslant 0 \quad \phi;\Phi \models^{\mathcal{E}} \mathrm{I} \leqslant n \quad \phi;\Phi \models^{\mathcal{E}} n \leqslant \mathrm{J}}{\phi;\Phi;\Gamma \vdash^{\mathcal{E}}_{\mathrm{K}} \underline{\mathtt{n}}:\mathtt{Nat}[\mathrm{I},\mathrm{J}]}\ N \qquad \frac{\begin{array}{c}\phi,b;\Phi,b<\mathrm{L};\Gamma,x:[a<\mathrm{I}]\cdot\sigma \vdash^{\mathcal{E}}_{\mathrm{K}} t:\tau \quad \phi;\Phi \vdash^{\mathcal{E}} \tau\{0/b\} \sqsubseteq \gamma \\ \phi,a,b;\Phi,a<\mathrm{I},b<\mathrm{L} \vdash^{\mathcal{E}} \tau\{\triangle_b^{b+1,a}\mathrm{I}+b+1/b\} \sqsubseteq \sigma \\ \phi;\Phi \vdash^{\mathcal{E}} \Sigma \sqsubseteq \sum_{b<\mathrm{L}} \Gamma \quad \phi;\Phi \models^{\mathcal{E}} \triangle_b^{0,1}\mathrm{I} \leqslant \mathrm{L},\mathrm{M}\end{array}}{\phi;\Phi;\Sigma \vdash^{\mathcal{E}}_{\mathrm{M}\dot{-}1+\sum_{b<\mathrm{L}}\mathrm{K}} \mathtt{fix}\ x.t:\gamma}\ R$$

Figure 2: Typing rules

Φ, t can be given type σ and *cost* I once its free term variables have types as in Γ. In proving this, equations from $\mathcal{E}$ can be used. Typing rules are in Figure 2, where binary and bounded sums are used in their natural generalization to contexts. A *type derivation* is nothing more than a tree built according to typing rules. A *precise type derivation* is a type derivation such that all premises of the form $\sigma \sqsubseteq \tau$ (respectively, in the form $\mathrm{I} \leqslant \mathrm{J}$) are required to be in the form $\sigma \cong \tau$ (respectively, $\mathrm{I} = \mathrm{J}$). As a last remark, note that each rule can be seen as a decoration of a rule of ordinary PCF.

Derivations in dℓPCF enjoy substitution properties both at the level of terms and at the level of index terms.

Lemma 2 (Substitution)**.** *Let* $\phi, a; \Phi, a < \mathrm{I}; \varnothing \vdash^{\mathcal{E}}_{\mathrm{J}} t : \sigma$ *and* $\phi; \Phi; x : [a < \mathrm{I}] \cdot \sigma, \Delta \vdash^{\mathcal{E}}_{\mathrm{K}} u : \tau$*. Then we have* $\phi; \Phi; \Delta \vdash^{\mathcal{E}}_{\mathrm{H}} u\{t/x\} : \tau$ *with* $\phi; \Phi \models^{\mathcal{E}} \mathrm{H} \leqslant \mathrm{K} + \mathrm{I} + \sum_{a<\mathrm{I}} \mathrm{J}$.

Thanks to the above Substitution Lemma we can prove, as expected, that types are preserved by reduction. In particular, this holds with respect to a weak reduction with which dℓPCF is equipped [8, 9].

Theorem 2 (Subject Reduction)**.** *Let* $\phi; \Phi; \varnothing \vdash_{\mathrm{I}} t : \sigma$ *and* $t \to u$*. Then,* $\phi; \Phi; \varnothing \vdash_{\mathrm{J}} u : \sigma$*, where* $\phi; \Phi \models \mathrm{J} \leqslant \mathrm{I}$.

Notice that the above theorem says something more than the usual type preservation theorems. Indeed, it says that the weight can change during the reduction but it also ensures that it cannot increase.

3. Index terms and Krivine's Machine

The main reason for introducing index terms is to obtain precise information about the reduction of $\mathsf{d}\ell\mathsf{PCF}$ programs by means of the abstract machine $\mathsf{K}_{\mathsf{PCF}}$ defined in Figure 3. This is an adaptation of Krivine's Machine to deal with all the PCF constructions.

The *configurations* of the machine $\mathsf{K}_{\mathsf{PCF}}$, ranged over by $C, D, \ldots$, are triples $C = (t, \mu, \xi)$ where μ and ξ are two additional constructions: μ is an *environment*, that is a partial function from variables to *closures*; while ξ is a (possibly empty) *stack* of *contexts*. Stacks are ranged over by $\xi, \theta, \ldots$. A *closure*, as usual, is a pair $c = (t, \mu)$ where t is a term and μ is an environment. A *context* is either a closure, a term $\mathtt{s}$, a term $\mathtt{p}$, or a triple (u, v, μ) where u, v are terms and μ is an environment. In the sequel we will consider configurations that can be typed by means of PCF types (a precise definition is given in [8, 9]) and we call the set of such configurations $\mathrm{Conf}_{\mathsf{PCF}}$.

As usual, the symbol $\rightarrow^*$ denotes the reflexive and transitive closure of the transition relation $\rightarrow$. The relation $\rightarrow^*$ implements weak-head reduction. Analogously, the symbol $\rightarrow^r$ denotes the relation obtained by considering r steps of the transition relation $\rightarrow$. Weak-head normal forms and the normal forms coincide for programs. So the machine $\mathsf{K}_{\mathsf{PCF}}$ is a correct and complete device to evaluate programs. For this reason, the notation $t \Downarrow \underline{\mathtt{n}}$ can be used as a shorthand for $(t, \epsilon, \epsilon) \rightarrow^* (\underline{\mathtt{n}}, \mu, \epsilon)$. Moreover, notations like $C \Downarrow^n$ will be used to stress that C reduces to an irreducible configuration in exactly n steps.

In the sequel we will also need to distinguish variable steps from the others. For this reason we write $\rightarrow_v$ for a reduction step obtained by applying the variable rule while we write $\rightarrow_{\overline{v}}$ for a reduction step obtained by applying any other rule except the variable rule. Finally, if C is a configuration, we use the notation $C \downarrow^n (v, \mu, \epsilon)$ to denote the fact that C reduces to the value v and uses during the reduction n variable steps. We define $C \downarrow^n$ as $\exists v, \mu$ such that $C \downarrow^n (v, \mu, \epsilon)$.

An important property that we inherit from the design of Krivine's machine is that in the evaluation of a program we need to record in environments and stacks only subterms of the initial term. Since we have some additional constructions that are not considered in the traditional definition of Krivine's machine, we need to slightly adapt this property. In particular, the property is valid for all the terms that are not numerals. This is expressed by the following property.

Lemma 3. *Let* $\varnothing;\varnothing;\varnothing \vdash_{\mathrm{I}} t : \mathtt{Nat}[\mathrm{J},\mathrm{K}]$ *such that* $(t,\epsilon,\epsilon) \to^r D$. *Then, every* $v \neq \underline{\mathtt{n}}$ *in* D *is a subterm of* t.

Proof. By induction on r. The base case is trivial. For the inductive case, a further case distinction on the step performed is needed. All the rules just move around pieces of terms that are either in head position or in the stack or in the environment. The only rule performing a substitution is the rule for variables, but this takes one term from the stack and place it in the head position. □

The above lemma justifies the following definition of size for $\mathsf{K}_{\mathsf{PCF}}$ configurations. The *size* of a configuration $C = (t,\mu,\xi)$, denoted $|C|$, is defined as $|(t,\mu,\xi)| = |t| + |\xi|$. The *size* of a stack ξ, denoted $|\xi|$, is defined as the sum of the sizes of its elements where the size of a context is defined as:

$$|(t,\mu)| = |t| \qquad |\mathtt{s}| = |\mathtt{p}| = 1 \qquad |(u,v,\mu)| = |u| + |v|$$

The above definition of size is useful to prove the following lemma.

Lemma 4. *Let* $\varnothing;\varnothing;\varnothing \vdash_{\mathrm{I}} t : \mathtt{Nat}[\mathrm{J},\mathrm{K}]$ *such that* $(t,\epsilon,\epsilon) \downarrow^r$. *If* $(t,\epsilon,\epsilon) \Downarrow^n$, *then* $n \leqslant |t| \cdot (r+1)$.

Proof. By induction on r, noticing that for each step $C' \to_{\overline{v}} C''$ we have $|C''| < |C'|$ while, by Lemma 3, for each step $C' \to_v C''$ we have $|C''| \leqslant |C'| + |t|$. □

Index terms ensure that typing judgements give precise information about the complexity of $\mathsf{d}\ell\mathsf{PCF}$ terms as stated by the following theorem.

Theorem 3 (Intensional soundness)**.** *Let* $\varnothing;\varnothing;\varnothing \vdash_{\mathrm{I}} t : \mathtt{Nat}[\mathrm{J},\mathrm{K}]$ *and* $t \Downarrow^n \underline{\mathtt{m}}$. *Then,* $n \leqslant |t| \cdot (\llbracket \mathrm{I} \rrbracket + 1)$ *and* $\llbracket \mathrm{J} \rrbracket \leqslant m \leqslant \llbracket \mathrm{K} \rrbracket$.

In [8, 9] the above result has been proved operationally. In particular, the idea of the proof given there is to show that by performing a variable step the weight of a typing judgement can decrease while every other step leaves the weight unchanged. This argument, combined with Lemma 4 above, provides a proof of Intensional Soundness. One of our goals in this work is to replace the above mentioned syntactic argument by means of a semantic argument with the aim of clarifying the nature of the $\mathsf{d}\ell\mathsf{PCF}$ index annotations.

The design of the index term decorations and of the $\mathsf{d}\ell\mathsf{PCF}$ type system has been motivated by the search for a *relatively complete* type system for complexity analysis. This property can be proved by considering a *universal equational program* $\mathcal{U}$ (i.e. an equational program able to simulate all the equational programs including itself), see [8, 9] for more details.

Term	Environment	Stack		Term	Environment	Stack
tu	μ	ξ	$\to$	t	μ	$(u,\mu)\cdot\xi$
$\lambda x.t$	μ	$c\cdot\xi$	$\to$	t	$c\cdot\mu$	ξ
x	$(t_0,\mu_0)\cdot\dots\cdot(t_n,\mu_n)$	ξ	$\to$	t_x	μ_x	ξ
$\mathtt{ifz}\ t\ \mathtt{then}\ u\ \mathtt{else}\ v$	μ	ξ	$\to$	t	μ	$(u,v,\mu)\cdot\xi$
$\mathtt{fix}\ x.t$	μ	ξ	$\to$	t	$(\mathtt{fix}\ x.t,\mu)\cdot\mu$	ξ
$\underline{\mathtt{n}}$	μ	$\mathtt{s}\cdot\xi$	$\to$	$\underline{\mathtt{n}+1}$	μ	ξ
$\underline{\mathtt{n}}$	μ	$\mathtt{p}\cdot\xi$	$\to$	$\underline{\mathtt{n}-1}$	μ	ξ
$\underline{0}$	μ	$(t,u,\nu)\cdot\xi$	$\to$	t	ν	ξ
$\underline{\mathtt{n}+1}$	μ	$(t,u,\nu)\cdot\xi$	$\to$	u	ν	ξ
$\mathtt{s}(t)$	μ	ξ	$\to$	t	μ	$\mathtt{s}\cdot\xi$
$\mathtt{p}(t)$	μ	ξ	$\to$	t	μ	$\mathtt{p}\cdot\xi$

Figure 3: The $\mathsf{K}_{\mathsf{PCF}}$ machine transition steps.

Theorem 4 (Relative Completeness for Programs)**.** *Let t be a* PCF *program such that $t \Downarrow^n \underline{\mathtt{m}}$. Then, there exist two index terms I and J such that $[\![\mathrm{I}]\!]^{\mathcal{U}} \leqslant n$ and $[\![\mathrm{J}]\!]^{\mathcal{U}} = m$ and such that the term t is typable in* dℓPCF *as $\varnothing;\varnothing;\varnothing \vdash^{\mathcal{U}}_{\mathrm{I}} t : \mathtt{Nat}[\mathrm{J}]$.*

Interestingly, this property does not only hold for programs but it also holds for functions.

Theorem 5 (Relative Completeness for Functions)**.** *Suppose that t is a* PCF *term such that $\vdash t : \mathtt{Nat} \to \mathtt{Nat}$. Moreover, suppose that there are two (total and computable) functions $f, g : \mathbb{N} \to \mathbb{N}$ such that $t\ \underline{\mathtt{n}} \Downarrow^{g(n)} \underline{\mathtt{f}(\mathtt{n})}$. Then, there exist three index terms $\mathrm{I},\mathrm{J},\mathrm{K}$ with $[\![\mathrm{I}+\mathrm{J}]\!] \leqslant g$ and $[\![\mathrm{K}]\!] = f$, such that*

$$a;\varnothing;\varnothing \vdash^{\mathcal{U}}_{\mathrm{I}} t : [b < \mathrm{J}]\cdot\mathtt{Nat}[a] \multimap \mathtt{Nat}[\mathrm{K}].$$

We conclude this section by stating two other properties of dℓPCF that we will tacitly use in the sequel.

Lemma 5 (Subtyping)**.** *Suppose $\phi;\Phi;x_1 : A_1,\dots,x_n : A_n \vdash_{\mathrm{I}} t : \sigma$ and $\phi;\Phi \vdash B_i \sqsubseteq A_i$ for $1 \leqslant i \leqslant n$ and $\phi;\Phi \vdash \sigma \sqsubseteq \tau$. Then, $\phi;\Phi;x_1 : B_1,\dots,x_n : B_n \vdash_{\mathrm{I}} t : \tau$.*

Lemma 6 (Index Term Substitution)**.** *Let $\phi,a;\Phi;\Gamma \vdash_{\mathrm{I}} t : \sigma$. Then we have*

$$\phi;\Phi\{\mathrm{J}/a\},\Psi;\Gamma\{\mathrm{J}/a\} \vdash_{\mathrm{I}\{\mathrm{J}/a\}} t : \sigma\{\mathrm{J}/a\}$$

for every J such that $\phi,\Psi \models^{\mathcal{E}} \mathrm{J} \Downarrow$.

Notice that the type system of dℓPCF is syntax-directed, so the two lemmas above, besides establishing correspondences between the two typing judgments, establish correspondences between the structure of the two type derivations.

4. Quantitative realizability model for dℓPCF

The realizability model we present in this section is inspired by Krivine's realizability [20]. As in Krivine's realizability, we build the model around the Krivine machine. Hence the machine $\mathsf{K}_{\mathsf{PCF}}$ presented in the previous section becomes at the same time the evaluation medium for dℓPCF terms and the basis of the realizability machinery.

To define the interpretation and derive the properties we are interested in, we first need to extend the machine $\mathsf{K}_{\mathsf{PCF}}$. We add to the set of closures a special closure ✠, named *daimon*, well typed for every PCF type and which has no corresponding reduction rule in the machine $\mathsf{K}_{\mathsf{PCF}}$. Hence, once in head position, the daimon blocks the computation. It is in a sense the dual of the empty stack[1]. Even if ✠ has no computational behavior, it will permit to evaluate under a λ binder. This is what the following lemma says.

Lemma 7. *Let t be a term, μ an environment and $m, n \in \mathbb{N}$. If $(t, \mu, ✠.\epsilon) \downarrow^m$, then $(t, \mu, \epsilon) \downarrow^n$ with $n \leqslant m$.*

Proof. Easy by induction on n and inspection of the $\mathsf{K}_{\mathsf{PCF}}$ machine rules. □

We can now start to define the realizability model. The core of biorthogonality-based models is the notion of orthogonality between closures and stacks. Here, we pair closures and stacks with index terms. Index terms are used in a similar way to monoid elements in [11] and [7].

Definition 1.

- *A* weighted closure *is a pair (c, I) where c is a closure and* I *is an index term. The set of weighted closures is denoted by* Λ.
- *A* weighted stack *is a pair (ξ, J) where ξ is a stack and* J *an index term. The set of weighted stacks is denoted by* Π.

Remark 1.
1. *The index terms associated with closures and stacks informally represent a bound on the resources (here, the time) used by these closures (resp. stacks) when they interact with stacks (resp. closures).*
2. *Notice that in the previous definition, the index terms are possibly open.*

[1] For the ones familiar with ludics [14], it is worth to stress that we use the diamon ✠ in a slightly different way. Whereas in ludics the daimon empties the context, here we choose it not to do so. This is only for convenience and it is not required for the soundness of our model.

Krivine style realizability is usually parametric over a subset $\perp\!\!\!\perp$ of machine's configurations that is closed under anti-evaluation: that is, if $C \in \perp\!\!\!\perp$ and $C' \to C$, then $C' \in \perp\!\!\!\perp$. Different sets $\perp\!\!\!\perp'$ and $\perp\!\!\!\perp''$ represent different notions of computational correctness. Configurations alone are not sufficient to track quantitative information. For this reason we extend the definition of $\perp\!\!\!\perp$ to include also an additional information represented by a natural number.

Definition 2. *A* quantitative pole *is a set* $\perp\!\!\!\perp \subseteq \mathit{Conf}_{\mathsf{PCF}} \times \mathbb{N}$ *such that:*

- *If* $n \leqslant m$ *and* $(C, n) \in \perp\!\!\!\perp$*, then* $(C, m) \in \perp\!\!\!\perp$.
- *If* $C \to_v C'$ *and* $(C', n) \in \perp\!\!\!\perp$*, then* $(C, n+1) \in \perp\!\!\!\perp$.
- *If* $C \to_{\overline{v}} C'$ *and* $(C', n) \in \perp\!\!\!\perp$*, then* $(C, n) \in \perp\!\!\!\perp$.

Remark 2. *As in the standard definition of realizability [20], we require that* $\perp\!\!\!\perp$ *is closed under anti-evaluation. In our case a anti-evaluation step can influence the quantitative information represented by the index term. However, notice that only the variable reduction steps* $\to_v$ *makes this parameter bigger. Indeed, we only want to count variable steps, as this information is sufficient to determine the total number of steps needed by a configuration to normalize, as stressed by Lemma 4.*

In the rest of this section, the definitions are parametric in a quantitative pole. In the next section, we will fix a particular quantitative pole $\perp\!\!\!\perp$ in order to derive the complexity results about typable $\mathsf{d}\ell\mathsf{PCF}$ terms.

A choice of quantitative pole induces a notion of orthogonality. In order to define it, we need to be able to obtain a natural number from an index term. This is achieved by using the interpretation function $[\![-]\!]^{\mathcal{E}}_{\rho}$. For this reason, the quantitative notion of orthogonality is naturally parametrized over an equational program $\mathcal{E}$ and an assignment ρ.

Definition 3. *A weighted closure* (c, I) *is* $\mathcal{E}$-ρ-orthogonal *to a weighted stack* (ξ, J) *iff:*

- $\mathtt{FV}(\mathrm{I}) \cup \mathtt{FV}(\mathrm{J}) \subseteq dom(\rho)$
- $((c, \xi), [\![\mathrm{I} + \mathrm{J}]\!]^{\mathcal{E}}_{\rho}) \in \perp\!\!\!\perp$

We use the notation $(c, \mathrm{I}) \perp^{\mathcal{E}}_{\rho} (\xi, \mathrm{J})$ *to indicate that* (c, I) *and* (ξ, J) *are* $\mathcal{E}$-ρ-orthogonal.

Notice that the notion of $\mathcal{E}$-ρ-orthogonality makes sense only when $[\![\mathrm{I} + \mathrm{J}]\!]^{\mathcal{E}}_{\rho}$ is defined. This is the case also for all the other notions that we introduce in the sequel of this section and the next one.

In what follows, we will assume that the equational program $\mathcal{E}$ is given so we will simply write $\perp_{\rho}$ for ρ-orthogonality. Substitution behaves well with respect to ρ-orthogonality.

Lemma 8. *The following two properties are equivalent:*

- $(c, I) \bot_{\rho\{a \leftarrow n\}} (\xi, J)$
- $(c, I\{n/a\}) \bot_{\rho} (\xi, J\{n/a\})$

Proof. It follows easily from the fact that for every index term I such that $\mathtt{FV}(\mathrm{I}) \subseteq dom(\rho) \cup \{a\}$ we have $[\![\mathrm{I}]\!]_{\rho\{a \leftarrow n\}} = [\![\mathrm{I}\{n/a\}]\!]_{\rho}$.

□

In order to define our quantitative realizability model we need to interpret types by sets of weighted closures. We then need additional operations for sets of weighted closures and weighted stacks.

Definition 4. *If X is a set of weighted closures, we define its upward closure $\overline{X}^{\rho}$ as*

$$\overline{X}^{\rho} = \{\ (c, \mathrm{I}) \mid \exists \mathrm{J}, (c, \mathrm{J}) \in X \wedge [\![\mathrm{J}]\!]_{\rho} \leqslant [\![\mathrm{I}]\!]_{\rho}\ \}$$

Notice that the upward closure operator is *monotonic*, i.e. $X \subseteq Y$ implies $\overline{X}^{\rho} \subseteq \overline{Y}^{\rho}$ and *idempotent*, i.e. $\overline{X}^{\rho} = \overline{\overline{X}^{\rho}}^{\rho}$. As usual, the orthogonality relation can be extended to sets of weighted closures and weighted stacks.

Definition 5.

- *If X is a set of weighted closures, then its ρ-orthogonal set $X^{\bot_{\rho}}$ is defined as:*
$$X^{\bot_{\rho}} = \{(\xi, \mathrm{J}) \mid \forall (c, \mathrm{I}) \in X, (c, \mathrm{I}) \bot_{\rho} (\xi, \mathrm{J})\}$$
- *If X is a set of weighted stacks, then its ρ-orthogonal set $X^{\bot_{\rho}}$ is defined as:*
$$X^{\bot_{\rho}} = \{(c, \mathrm{I}) \mid \forall (\xi, \mathrm{J}) \in X, (c, \mathrm{I}) \bot_{\rho} (\xi, \mathrm{J})\}$$
- *A set of weighted closures X is a* ρ-behavior *if $X^{\bot_{\rho}\bot_{\rho}} = X$.*

The extensions of ρ-orthogonality to sets of weighted closures and weighted stacks enjoy the usual properties of orthogonality. In particular we know that each set Y, which is the orthogonal of a set X, i.e. $Y = X^{\bot_{\rho}}$, is also a ρ-behavior. Moreover, we have the following important property.

Lemma 9. *Let ρ be a substitution and X be a ρ-behavior. If $(c, \mathrm{I}) \in X$ and J is an index term such that $\mathtt{FV}(\mathrm{J}) \subseteq dom(\rho)$ and $[\![\mathrm{I}]\!]_{\rho} \leqslant [\![\mathrm{J}]\!]_{\rho}$, then $(c, \mathrm{J}) \in X$.*

Proof. It follows easily from the definition of ρ-behavior (Definition 5) and by the definition of quantitative pole (Definition 2). □

$$
\begin{aligned}
|\mathtt{Nat}[\mathrm{I},\mathrm{J}]|_\rho &= \{\ (\bar{n},0) \mid [\![\mathrm{I}]\!]_\rho \leqslant n \leqslant [\![\mathrm{J}]\!]_\rho\ \}^{\perp_\rho\perp_\rho} \\
|A \multimap \sigma|_\rho &= \{\ (c.\xi, \mathrm{I}+\mathrm{J}) \mid (c,\mathrm{I}) \in |A|_\rho \wedge (\xi,\mathrm{J}) \in |\sigma|_\rho{}^{\perp_\rho}\ \}^{\perp_\rho} \\
|[a<\mathrm{I}].\sigma|_\rho &= \overline{\{\ (c, \sum_{a<\mathrm{I}} \mathrm{K}+\mathrm{I}) \mid (c,\mathrm{K}) \in \bigcap_{n<[\![\mathrm{I}]\!]_\rho} |\sigma|_{\rho\{a\leftarrow n\}}\ \} \cup \{(✠,0)\}}^{\rho}
\end{aligned}
$$

Figure 4: quantitative realizability Interpretation of $\mathsf{d}\ell\mathsf{PCF}$ Types.

Notice that the above Lemma says that behaviors are equal to their upward closure, i.e. for every ρ-behavior: $\overline{X}^\rho = X$.

Now, we have all the components to define an interpretation of $\mathsf{d}\ell\mathsf{PCF}$ basic and modal types.

Definition 6 (Interpreting Types)**.** *Given a type ς and an assignment ρ such that $\mathtt{FV}(\varsigma) \subseteq dom(\rho)$, the interpretation $|\varsigma|_\rho$ of ς in ρ is defined by the rules in Figure 4.*

Before moving to the properties of our interpretation it is worth discussing some of our choices. This is the content of the next two remarks.

Remark 3. *Notice that we do not interpret modal types over ρ-behaviors but nonetheless when ς does not contain any positive occurrence of a modal type, $|\varsigma|$ is a ρ-behavior. The reason why we don't interpret modal types as behaviors (for instance by using a biorthogonality closure) is that it would make more difficult to prove the soundness in the case of rules that manipulate modal types in the typing context. This is an issue related to call-by-name, and more can be found about it in [7].*

Remark 4. *The interpretation of modal types recalls the usual interpretation of universal quantifiers in realizability models, which is usually given by an intersection of behaviors, indexed by the quantification domain. We can in fact decompose the bounded modality using more primitive connectives: a* first-order quantification *and an* inequational implication*.*

The following definition corresponds to the interpretation of a usual first-order quantification, ranging over the natural numbers. This quantification $\forall a.\sigma$ binds the variable a in the type σ (but it does not bind a in the weighted terms that belong to this interpretation):

$$|\forall a.\sigma|_\rho = \bigcap_{n\in\mathbb{N}} |\sigma|_{\rho[a\leftarrow n]}$$

We can also define a inequational implication $a < I \mapsto \sigma$*, similar to the* equational implication *introduced by Miquel in [24]. This implication is computationally transparent and means $a < I$ implies σ. It corresponds to the following interpretation:*

$$|a < I \mapsto \sigma|_\rho = \begin{cases} |\sigma|_\rho & \text{if } \rho(a) \leqslant [\![I]\!]_\rho \\ \Lambda & \text{otherwise} \end{cases}$$

Given these two connectives, it is possible to decompose the bounded modality as follows:

$$|[a < I].\sigma|_\rho = \{\ (c, \sum_{a<I} K + I)\ \mid\ (c, K) \in |\forall a.(a < I \mapsto \sigma)|_\rho\ \}$$

We leave the study of this decomposition for subsequent works.

The interpretation of types has some interesting properties with respect to the quantitative information. In particular, it inherits some properties of the orthogonality relation.

Lemma 10. *Given a type ς and an assignment ρ, if $(c, K) \in |\varsigma|_{\rho\{a \leftarrow n\}}$ then $(c, K\{n/a\}) \in |\varsigma\{n/a\}|_\rho$.*

The type system of $\mathsf{d}\ell\mathsf{PCF}$ uses the subtyping relation extensively. For this reason, an important milestone in showing that our realizability definition gives a model of $\mathsf{d}\ell\mathsf{PCF}$ is to show that it is sound with respect to the subtyping relation. In particular, as already stressed, we are interested in having this correspondence only when we are able to satisfy the given constraints. This is formally stated by requiring that the assignment ρ satisfies the constraints in Φ. The soundness with respect to the subtyping is given by the following theorem.

Theorem 6 (Subtyping soundness)**.** *Suppose $\phi; \Phi \vdash \sigma \sqsubseteq \tau$. Then $\rho \models \phi; \Phi$ implies $|\sigma|_\rho \subseteq |\tau|_\rho$.*

Proof. By induction on the derivation with conclusion $\phi; \Phi \vdash \sigma \sqsubseteq \tau$. The linear arrow case follows as usual by the ρ-orthogonality properties.

Case

$$\frac{\phi; \Phi \models \mathrm{K} \leqslant \mathrm{I} \quad \phi; \Phi \models \mathrm{J} \leqslant \mathrm{H}}{\phi; \Phi \vdash \mathtt{Nat}[\mathrm{I}, \mathrm{J}] \sqsubseteq \mathtt{Nat}[\mathrm{K}, \mathrm{H}]}$$

Because we have $[\![\mathrm{K}]\!]_\rho \leqslant [\![\mathrm{I}]\!]_\rho$ and $[\![\mathrm{J}]\!]_\rho \leqslant [\![\mathrm{H}]\!]_\rho$, it is immediate that if $[\![\mathrm{I}]\!]_\rho \leqslant n \leqslant [\![\mathrm{J}]\!]_\rho$, then $[\![\mathrm{K}]\!]_\rho \leqslant n \leqslant [\![\mathrm{H}]\!]_\rho$. By the orthogonality properties, we have $|\mathtt{Nat}[\mathrm{I}, \mathrm{J}]|_\rho \subseteq |\mathtt{Nat}[\mathrm{K}, \mathrm{H}]|_\rho$.

Case

$$\frac{\phi; \Phi \vdash^{\mathcal{E}} B \sqsubseteq A \quad \phi; \Phi \vdash^{\mathcal{E}} \sigma \sqsubseteq \tau}{\phi; \Phi \vdash^{\mathcal{E}} A \multimap \sigma \sqsubseteq B \multimap \tau}$$

We know by induction that $|B|_\rho \subseteq |A|_\rho$ and $|\sigma|_\rho \subseteq |\tau|_\rho$. Then

$$\begin{array}{rcl} |A \multimap \sigma|_\rho & = & (|A|_\rho . |\sigma|_\rho{}^\perp)^\perp \\ & \subseteq & (|B|_\rho . |\tau|_\rho{}^\perp)^\perp \\ & = & |B \multimap \tau|_\rho \end{array}$$

Case

$$\frac{\phi, a; \Phi, a < \mathrm{J} \vdash \sigma \sqsubseteq \tau \quad \phi; \Phi \models \mathrm{J} \leqslant \mathrm{I}}{\phi; \Phi \vdash [a < \mathrm{I}] \cdot \sigma \sqsubseteq [a < \mathrm{J}] \cdot \tau}$$

We want to prove that

$$\{\ (c, \sum_{a<\mathrm{I}} \mathrm{K} + \mathrm{I}) \mid (c, \mathrm{K}) \in \bigcap_{n<[\![\mathrm{I}]\!]_\rho} |\sigma|_{\rho\{a \leftarrow n\}}\ \} \subseteq \overline{\{\ (c, \sum_{a<\mathrm{J}} \mathrm{K} + \mathrm{J}) \mid (c, \mathrm{K}) \in \bigcap_{n<[\![\mathrm{J}]\!]_\rho} |\tau|_{\rho\{a \leftarrow n\}}\ \}}^{\rho}$$

Then, the conclusion follows by monotonicity and idempotency of the upward closure operator.

Consider $(c, \mathrm{K}) \in \bigcap_{n<[\![\mathrm{I}]\!]_\rho} |\sigma|_{\rho\{a \leftarrow n\}}$. By induction hypothesis we have $|\sigma|_{\rho\{x \leftarrow n\}} \subseteq |\tau|_{\rho\{x \leftarrow n\}}$ for each $n < [\![\mathrm{J}]\!]_\rho$. Moreover, $\rho \models \phi; \Phi$ clearly implies $[\![\mathrm{J}]\!]_\rho \leqslant [\![\mathrm{I}]\!]_\rho$. So, we have $(c, \mathrm{K}) \in \bigcap_{n<[\![\mathrm{J}]\!]_\rho} |\tau|_{\rho\{a \leftarrow n\}}$. Hence, $(c, \sum_{a<\mathrm{J}} \mathrm{K} + \mathrm{J}) \in |[a < \mathrm{J}] \cdot \tau|_\rho$. Since $\sum_{a<\mathrm{J}} \mathrm{K} + \mathrm{J} \leqslant \sum_{a<\mathrm{I}} \mathrm{K} + \mathrm{I}$, by monotonicity and idempotency of the upward closure operator we can conclude $(c, \sum_{a<\mathrm{I}} \mathrm{K} + \mathrm{I}) \in |[a < \mathrm{J}] \cdot \tau|_\rho$.

□

Thanks to the soundness of the subtyping relation we can now prove that our realizability model is sound also with respect to the $\mathsf{d}\ell\mathsf{PCF}$ type system.

Theorem 7 (Soundness)**.** *Suppose* $\phi; \Phi; x_1 : A_1, \ldots, x_n : A_n \vdash^{\mathcal{E}}_K t : \sigma$. *Let* $\rho \models \phi; \Phi$ *and* $(c_i, J_i) \in |A_i|_\rho$. *Then*

$$(t, [x_1 := c_1, \ldots, x_n := c_n], K + \sum_{1 \leqslant i \leqslant n} J_i) \in |\sigma|_\rho$$

Proof. The proof is by generalized induction on the value of $[\![K]\!]_\rho$ with further induction on the derivation proving $\phi; \Phi; x_1 : A_1, \ldots, x_n : A_n \vdash^{\mathcal{E}}_K t : \sigma$. All the cases requires some manipulations of the index terms. We show few representative cases.
Let us consider first the case $[\![K]\!]_\rho = 0$:

Subcase

$$\frac{\phi; \Phi \vdash^{\mathcal{E}} [a < \mathrm{I}] \cdot \sigma \sqsubseteq [a < \mathbf{1}] \cdot \tau}{\phi; \Phi; x_1 : A_1, \ldots, x_n : A_n, x : [a < \mathrm{I}] \cdot \sigma \vdash^{\mathcal{E}}_{\mathsf{0}} x : \tau\{\mathsf{0}/a\}}$$

For simplicity, suppose $n = 0$. Consider $(c, \mathrm{H}) \in |[a < \mathrm{I}].\sigma|_\rho$. The case $c = \maltese$ is easy. So, consider the case $c \neq \maltese$. By assumption and by Subtyping soundness Theorem 6, we know that $(c, \mathrm{H}) \in |[a < \mathbf{1}].\tau|_\rho$. This means that there exists some K such that $\sum_{a<1} \mathrm{K} + \mathbf{1} = \mathrm{K}\{0/a\} + \mathbf{1} \leqslant \mathrm{H}$ and $(c, \mathrm{K}) \in |\tau|_{\rho\{a \leftarrow 0\}}$. But by Lemma 10, that means $(c, \mathrm{K}\{0/a\}) \in |\tau\{0/a\}|_\rho$. Hence, by anti-reduction, that means $(x, [x := c], \mathrm{K}\{0/a\} + \mathbf{1}) \in |\tau\{0/a\}|_\rho$. But because $\mathrm{K}\{0/a\} + 1 \leqslant \mathrm{H}$, we finally obtain $(x, [x := c], \mathrm{H}) \in |\tau\{0/a\}|_\rho$.

Subcase

$$\frac{\begin{array}{c}\phi, b; \Phi, b < \mathrm{L}; \Gamma, x : [a < \mathrm{P}] \cdot \sigma \vdash^{\mathcal{E}}_{\mathrm{K}} t : \tau \\ \phi; \Phi \vdash^{\mathcal{E}} \tau\{0/b\} \sqsubseteq \gamma \\ \phi, a, b; \Phi, a < \mathrm{P}, b < \mathrm{L} \vdash^{\mathcal{E}} \tau\{\textcircled{\triangle}_b^{b+1,a} \mathrm{P} + b + 1/b\} \sqsubseteq \sigma \\ \phi; \Phi \vdash^{\mathcal{E}} \Sigma \sqsubseteq \sum_{b<\mathrm{L}} \Gamma \\ \phi; \Phi \models^{\mathcal{E}} \textcircled{\triangle}_b^{0,1} \mathrm{P} \leqslant \mathrm{L}, \mathrm{R}\end{array}}{\phi; \Phi; \Sigma \vdash^{\mathcal{E}}_{\mathrm{R} \dot{-} 1 + \sum_{b<\mathrm{L}} \mathrm{K}} \mathtt{fix}\, x.t : \gamma}\ R$$

Without loss of generality and by definition of bounded sum we can consider the case where $\Gamma = y : [a < \mathrm{H}] \cdot \delta\{a + \sum_{d<b} \mathrm{H}\{d/a\}/a\}$ and $\Sigma = y : [a < \sum_{b<\mathrm{L}} \mathrm{H}] \cdot \delta$. Consider $(c, \mathrm{N}) \in |[a < \sum_{b<\mathrm{L}} \mathrm{H}] \cdot \delta|_\rho$. By the interpretation definition and by some manipulation of the index terms, we have that for every $n < [\![\mathrm{L}]\!]_\rho$:

$$(c, \sum_{a<\mathrm{H}\{n/b\}} \mathrm{M}\{n/b\} + \mathrm{H}\{n/b\}) \in |[a < \mathrm{H}] \cdot \delta\{a + \sum_{d<b} \mathrm{H}\{d/a\}/a\}|_{\rho\{b \leftarrow n\}}$$

for some M such that $\sum_{b<\mathrm{L}} (\sum_{a<\mathrm{H}} \mathrm{M} + \mathrm{H}) \cong \mathrm{N}$.

By using some manipulations of the indices, we can derive:

$$\phi; \Phi; y : [a < \mathrm{H}\{0/b\}] \cdot \delta\{0/b\}, x : [a < \mathrm{P}\{0/b\}] \cdot \sigma\{0/b\} \vdash^{\mathcal{E}}_{\mathrm{K}\{0/b\}} t : \gamma$$

Moreover, by assumption we have $[\![\mathrm{R} \dot{-} 1 + \sum_{b<\mathrm{L}} \mathrm{K}]\!]_\rho = 0$ and by definition of forest cardinality this implies both that $[\![\mathrm{P}\{0/b\}]\!]_\rho = 0$ and $[\![\mathrm{K}]\!]_\rho = 0$.

By definition of interpretation, this implies that we have $(\mathtt{fix}\ x.t, [y := c], 0) \in |[a < \mathrm{P}\{0/b\}] \cdot \sigma\{0/b\}|_\rho$. So, by induction hypothesis we have

$$(t, [y := c, x := (\mathtt{fix}\, x.t, [y := c])], \sum_{a<\mathrm{H}\{0/b\}} \mathrm{M}\{0/b\}+\mathrm{H}\{0/b\})+\mathrm{K}\{0/b\}) \in |\gamma|_\rho$$

and by antireduction we obtain:

$$(\mathtt{fix}\, x.t, [y := c], \sum_{a<\mathrm{H}\{0/b\}} \mathrm{M}\{0/b\} + \mathrm{H}\{0/b\}) + \mathrm{K}\{0/b\}) \in |\gamma|_\rho$$

and by Lemma 9 since clearly

$$[\![\sum_{a<\mathrm{H}\{0/b\}} \mathrm{M}\{0/b\} + \mathrm{H}\{0/b\}) + \mathrm{K}\{0/b\}]\!]_\rho \leqslant [\![\mathrm{N} + \mathrm{K}]\!]_\rho$$

we have

$$(\mathtt{fix}\, x.t, [y := c], \mathrm{N} + \mathrm{K}) \in |\gamma|_\rho$$

That is what we need since by assumption $[\![\mathrm{R} \dot{-} 1 + \sum_{b<\mathrm{L}} \mathrm{K}]\!]_\rho = 0$ and $[\![\mathrm{K}]\!]_\rho = 0$.

Let us consider now the case $[\![K]\!]_\rho = n + 1$:

Subcase

$$\frac{\begin{array}{c}\phi; \Phi; \Gamma \vdash_{\mathrm{J}} t : [a < \mathrm{I}] \cdot \sigma \multimap \tau \\ \phi, a; \Phi, a < \mathrm{I}; \Delta \vdash_{\mathrm{K}} u : \sigma \\ \phi; \Phi \vdash \Sigma \sqsubseteq \Gamma \uplus \sum_{a<\mathrm{I}} \Delta\end{array}}{\phi; \Phi; \Sigma \vdash_{\mathrm{J}+\sum_{a<\mathrm{I}} \mathrm{K}+\mathrm{I}} tu : \tau}\ A$$

Without loss of generality and by definition of bounded sum we consider the case where $\Gamma = x : [b < \mathrm{M}] \cdot \gamma$, and $\Delta = x : [b < \mathrm{H}] \cdot \gamma\{\mathrm{M} + b + \sum_{d<a} \mathrm{H}\{d/b\}/b\}$ and $\Sigma = x : [b < \mathrm{M} + \sum_{a<\mathrm{I}} \mathrm{H}] \cdot \gamma$. Consider $(c, \mathrm{N}) \in |[b < \mathrm{M} + \sum_{a<\mathrm{I}} \mathrm{H}] \cdot \gamma|_\rho$. By the interpretation definition and by some manipulation of the index terms we have that $(c, \sum_{b<\mathrm{M}} \mathrm{L} + \mathrm{M}) \in |[b < \mathrm{M}] \cdot \gamma|_\rho$ and for every $n \leqslant [\![\mathrm{I}]\!]_\rho$ we also have

$$(c, \sum_{b<\mathrm{H}} \mathrm{L} + \mathrm{H}) \in |[b < \mathrm{H}] \cdot \gamma\{\mathrm{M} + b + \sum_{d<a} \mathrm{H}\{d/b\}/b\}|_{\rho\{a \leftarrow\!\!\shortmid n\}}$$

for some L such that $\sum_{b<\mathrm{M}} \mathrm{L} + \mathrm{M} + \sum_{a<\mathrm{I}}(\sum_{b<\mathrm{H}} \mathrm{L} + \mathrm{H}) \cong \mathrm{N}$.

So, by induction hypothesis we have $(t, [x := c], \mathrm{J} + \sum_{b<\mathrm{M}} \mathrm{L} + \mathrm{M}) \in |[a < \mathrm{I}] \cdot \sigma \multimap \tau|_\rho$. Also, by induction hypothesis and some transformation we have $(u, [x := c], \sum_{a<\mathrm{I}}(\mathrm{K} + \sum_{b<\mathrm{H}} \mathrm{L} + \mathrm{H}) + \mathrm{I}) \in |[a < \mathrm{I}] \cdot \sigma|_\rho$. So by anti-reduction we have:

$$((tu, [x := c]), \mathrm{J} + \sum_{b<\mathrm{M}} \mathrm{L} + \mathrm{M} + (\sum_{a<\mathrm{I}}(\mathrm{K} + \sum_{b<\mathrm{H}} \mathrm{L} + \mathrm{H}) + \mathrm{I})) \in |\tau|_\rho$$

and since

$$\mathrm{J} + \sum_{b<\mathrm{M}} \mathrm{L} + \mathrm{M} + (\sum_{a<\mathrm{I}}(\mathrm{K} + \sum_{b<\mathrm{H}} \mathrm{L} + \mathrm{H}) + \mathrm{I}) = \mathrm{J} + \sum_{a<\mathrm{I}} \mathrm{K} + \mathrm{I} + \mathrm{N}$$

the conclusion follows.

Subcase

$$\frac{\begin{array}{c}\phi, b; \Phi, b < \mathrm{L}; \Gamma, x : [a < \mathrm{P}] \cdot \sigma \vdash^{\mathcal{E}}_{\mathrm{K}} t : \tau \\ \phi; \Phi \vdash^{\mathcal{E}} \tau\{0/b\} \sqsubseteq \gamma \\ \phi, a, b; \Phi, a < \mathrm{P}, b < \mathrm{L} \vdash^{\mathcal{E}} \tau\{\textcircled{\triangle}_b^{b+1,a} \mathrm{P} + b + 1/b\} \sqsubseteq \sigma \\ \phi; \Phi \vdash^{\mathcal{E}} \Sigma \sqsubseteq \sum_{b<\mathrm{L}} \Gamma \\ \phi; \Phi \models^{\mathcal{E}} \textcircled{\triangle}_b^{0,1} \mathrm{P} \leqslant \mathrm{L}, \mathrm{R}\end{array}}{\phi; \Phi; \Sigma \vdash^{\mathcal{E}}_{\mathrm{R} \dot{-} 1 + \sum_{b<\mathrm{L}} \mathrm{K}} \mathtt{fix}\, x.t : \gamma}\ R$$

Without loss of generality and by definition of bounded sum we can consider the case where $\Gamma = y : [a < \mathrm{H}] \cdot \delta\{a + \sum_{d<b} \mathrm{H}\{d/a\}/a\}$ and $\Sigma = y : [a < \sum_{b<\mathrm{L}} \mathrm{H}] \cdot \delta$. Consider $(c, \mathrm{N}) \in |[a < \sum_{b<\mathrm{L}} \mathrm{H}] \cdot \delta|_\rho$. By the interpretation definition and by some manipulation of the index terms we have that for every $n < [\![\mathrm{L}]\!]_\rho$:

$$(c, \sum_{a<\mathrm{H}\{n/b\}} \mathrm{M}\{n/b\} + \mathrm{H}\{n/b\}) \in |[a < \mathrm{H}] \cdot \delta\{a + \sum_{d<b} \mathrm{H}\{d/a\}/a\}|_{\rho\{b \leftarrow n\}}$$

for some M such that $\sum_{b<\mathrm{L}}(\sum_{a<\mathrm{H}} \mathrm{M} + \mathrm{H}) \cong \mathrm{N}$.

By using some manipulations of the indices, we can derive:

$$\phi; \Phi; \Gamma_1, x : [a < \mathrm{P}\{0/b\}] \cdot \sigma\{0/b\} \vdash^{\mathcal{E}}_{\mathrm{K}\{0/b\}} t : \gamma$$

Without loss of generality we can assume that $[\![\mathrm{P}\{0/b\}]\!] > 0$. The case where $[\![\mathrm{P}\{0/b\}]\!] = 0$ is similar to the base case. By further manipulation of the indices, we also have:

$$\phi; \Phi, a < \mathrm{P}\{0/b\}; \Sigma_1 \vdash^{\mathcal{E}}_{\mathrm{Q}\{a/c\} \dot{-} 1 + \sum_{b<\mathrm{Q}\{a/c\}} \mathrm{K}\{\mathrm{U}/b\}} \mathtt{fix}\, x.t : \sigma\{0/b\}$$

for $\mathrm{Q} = \circledtriangle_b^{0,1} \mathrm{P}\{b+1+\circledtriangle_b^{0,c} \mathrm{P}/b\}$ and $\mathrm{U} = 1+b+\sum_{c<a} \mathrm{Q}$ and $\phi; \Phi, a < \mathrm{P}\{0/b\} \vdash \Sigma_1 \sqsubseteq \sum_{b<\mathrm{Q}\{a/c\}} \Gamma\{\mathrm{U}/b\}$. In particular, we can choose Γ_1 and Σ_1 such that:

$$\phi; \Phi \vdash \Sigma \cong \sum_{a<\mathrm{P}\{0/b\}} \Sigma_1 + \Gamma_1 \sqsubseteq \sum_{a<\mathrm{P}\{0/b\}} \sum_{b<\mathrm{Q}\{a/c\}} \Gamma\{\mathrm{U}/b\} + \Gamma\{0/b\} \cong \sum_{b<\mathrm{L}} \Gamma$$

Since $[\![\mathrm{P}\{0/b\}]\!] > 0$ for every $\rho' = \rho\{a := k\}$ with $k < [\![\mathrm{P}\{0/b\}]\!]$ we have that

$$[\![\mathrm{Q}\{a/c\} \dot{-} 1 + \sum_{b<\mathrm{Q}\{a/c\}} \mathrm{K}\{\mathrm{U}/b\}]\!]_{\rho'} < [\![\mathrm{R} \dot{-} 1 + \sum_{b<\mathrm{L}} \mathrm{K}]\!]_{\rho}$$

So by generalized induction hypothesis we have that

$$(\mathtt{fix}\ x.t, [y := c], \mathrm{J}) \in |\gamma|_{\rho'}$$

for

$$\mathrm{J} = \mathrm{Q}\{a/c\} \dot{-} 1 + \sum_{b<\mathrm{Q}\{a/c\}} \mathrm{K}\{\mathrm{U}/b\} + \sum_{b<\mathrm{Q}\{a/c\}} (\sum_{a<\mathrm{H}\{\mathrm{U}/b\}} \mathrm{M}\{\mathrm{U}/b\} + \mathrm{H}\{\mathrm{U}/b\})$$

and so

$$(\mathtt{fix}\ x.t, [y := c], \sum_{a<\mathrm{P}\{0/b\}} \mathrm{J}) \in |[a < [\![\mathrm{P}\{0/b\}]\!]] \cdot \gamma|_{\rho}$$

So, by induction hypothesis we also have

$$(t, [y := c, x := (\mathtt{fix}\ x.t, [y := c])], \mathrm{K}_2) \in |\gamma|_{\rho}$$

where

$$\begin{aligned} \mathrm{K}_2 = {} & \mathrm{K}\{0/b\} + \mathrm{P}\{0/b\} + \\ & \sum_{a<\mathrm{P}\{0/b\}} \Big(\mathrm{Q}\{a/c\} \dot{-} 1 + \sum_{b<\mathrm{Q}\{a/c\}} \mathrm{K}\{\mathrm{U}/b\} + \sum_{b<\mathrm{Q}\{a/c\}} (\sum_{a<\mathrm{H}\{\mathrm{U}/b\}} \mathrm{M}\{\mathrm{U}/b\} + \mathrm{H}\{\mathrm{U}/b\}) \Big) \\ & + \sum_{a<\mathrm{H}\{0/b\}} \mathrm{M}\{0/b\} + \mathrm{H}\{0/b\} \\ & = \mathrm{R} \dot{-} 1 + \sum_{b<\mathrm{L}} \mathrm{K} + \mathrm{N} \end{aligned}$$

So, by antireduction we obtain

$$(\mathtt{fix}\ x.t, [y := c], \mathrm{K}_2) \in |\sigma|_{\mu})$$

that is what we need to prove.

□

We conclude this section with a remark about the above proof of soundness.

Remark 5. *The Soundness theorem states the correctness of the model with respect to the typing information. Usually, in order to prove this kind of theorems in the context of* PCF *one needs some technique to ensure that the interpretation of the rule for typing fixpoints is well-defined. In denotational models it is common practice to use the semantics approximants of fixpoints while in syntactic or term models it is common practice to use techniques like step-indexing [2]. Our proof does not need these techniques because we can use directly the information provided by the weight.*

5. Quantitative reducibility candidates and dℓPCF properties

In this section, we are interested in using the quantitative realizability model introduced in the previous section to prove properties about dℓPCF. In particular, we show how it can be used to give a new proof of the dℓPCF intensional soundness theorem. Moreover, when a universal equational program is considered we can also show a correspondance between our model and the type system.

All along this section we will consider a fixed quantitative pole $\perp\!\!\!\perp$ defined as follows:

$$\perp\!\!\!\perp = \{\ (C, n) \mid C \downarrow^m\ \wedge\ m \leqslant n\ \}$$

Checking that $\perp\!\!\!\perp$ is indeed a quantitative pole is easy since we take into account exactly the number of variable reduction steps. The choice of this quantitative pole ensures that we have some additional properties on ρ-behaviors. In particular, the following lemma shows that each behavior contains the weighted closure $(\maltese, 0)$.

Lemma 11. *For each ρ-behavior X, we have $(\maltese, 0) \in X$.*

Proof. If $(\xi, I) \in X^{\perp_\rho}$, then it is immediate that $(\maltese, \xi) \downarrow^0$. Hence $(\maltese, 0) \perp_\rho (\xi, I)$ since $0 \leqslant [\![I]\!]_\rho$. So, $(\maltese, 0) \in X^{\perp_\rho\perp_\rho} = X$.

□

The quantitative pole $\perp\!\!\!\perp$ also permits to consider a natural quantitative extension of the usual notion of reducibility candidates.

Definition 7 (Quantitative reducibility candidates)**.** *The set of ρ-quantitative reducibility candidates, denoted by $\mathtt{QCR}_\rho$ is the set of all the ρ-behaviors X such that $X \subseteq \{(\epsilon, 0)\}^{\perp_\rho}$.*

Remark 6. *Informally, a quantitative reducibility candidate is a set X that only contains bounded-time closures. Indeed, suppose X is a ρ-behavior and (c, I) is a bounded closure such that $(c, I) \in X$. Then, by definition we have $(c, \epsilon) \downarrow^m$ with $m \leqslant [\![\mathrm{I}]\!]_\rho$.*

The notion of a quantitative reducibility candidate helps us to prove a first form of completeness for our model. We use the terminology completeness for this first theorem, since it is similar to the notion of completeness of phase semantics with respect to linear logic or of Kripke models with respect to intuitionistic logic for instance, but where termination is considered instead of typability.

Theorem 8 (Completeness)**.** *For every ρ and all basic types σ we have $|\sigma|_\rho \in \mathtt{QCR}_\rho$.*

Proof. By induction on the structure of the basic type σ.

Case $\mathtt{Nat}[\mathrm{I}, \mathrm{J}]$. It is easy to check that for each integer $\underline{\mathtt{n}}$ and each environment ν, $(\underline{\mathtt{n}}, \nu, \mathtt{0}) \in \{(\epsilon, \mathtt{0})\}^{\perp_\rho}$. Hence by monotonicity of the biorthogonality closure, we obtain $|\mathtt{Nat}[\mathrm{I}, \mathrm{J}]|_\rho \subseteq \{(\epsilon, \mathtt{0})\}^{\perp_\rho \perp_\rho \perp_\rho} = \{(\epsilon, \mathtt{0})\}^{\perp_\rho}$.

Case $[a < \mathrm{I}]\sigma \multimap \tau$. Suppose $(c, \mathrm{K}) \in |[a < \mathrm{I}]\sigma \multimap \tau|_\rho$. Then because $(\maltese, \mathtt{0}) \in |[a < \mathrm{I}]\sigma|_\rho$ and because $(\epsilon, \mathtt{0}) \in |\tau|_\rho$ (since $|\tau|_\rho \in \mathtt{QCR}_\rho$ by induction hypothesis), we obtain that $(c, \maltese.\epsilon, [\![\mathrm{K}]\!]_\rho) \in \perp\!\!\!\perp$.

Using Lemma 7, we obtain $(c, \epsilon, [\![\mathrm{K}]\!]_\rho) \in \perp\!\!\!\perp$.

□

Theorem 8 stated above can be considered as a bounded-time termination property of realizers. However, this property alone is not sufficient to prove the intensional soundness result for $\mathsf{d}\ell\mathsf{PCF}$.

Remark 7. *Theorem 8 does not hold for all choices of a pole. For instance, if we choose the pole $\perp\!\!\!\perp = \{\ (C, n) \mid C \text{ diverges }\}$, for each integer $\underline{\mathtt{n}}$ and $k \in \mathbb{N}$, the configuration $(\underline{\mathtt{n}}, \epsilon, k) \notin \perp\!\!\!\perp$. This shows that $|\mathtt{Nat}[0, 0]|_\rho \notin \mathtt{QCR}_\rho$.*

To be able to prove intensional soundness of $\mathsf{d}\ell\mathsf{PCF}$, we first need to prove an internal completeness result for the type $\mathtt{Nat}[\mathrm{J}, \mathrm{K}]$. This result characterizes the elements of $|\mathtt{Nat}[\mathrm{J}, \mathrm{K}]|_\rho$. The proof mainly uses the fact that $\perp\!\!\!\perp$ only contains *safe* configurations, and that stacks can discriminate two different natural numbers. It is very similar to the internal completeness property of ludics [14, 27], hence the name internal completeness.

Theorem 9 (Internal completeness)**.** *Suppose $[\![\mathrm{J}]\!]_\rho \leqslant [\![\mathrm{K}]\!]_\rho$ and $\models \mathrm{I} \Downarrow$. If $(c, \mathrm{I}) \in |\mathtt{Nat}[\mathrm{J}, \mathrm{K}]|_\rho$ then $(c, \epsilon) \downarrow^m (\underline{\mathtt{n}}, \nu, \epsilon)$ with $m \leqslant [\![\mathrm{I}]\!]_\rho$ and $[\![\mathrm{J}]\!]_\rho \leqslant n \leqslant [\![\mathrm{K}]\!]_\rho$.*

Proof. Let Ω be a PCF diverging term. In order to discriminate integers we will use the following families of stacks indexed by $k \in \mathbb{N}$:

$$\xi_k = \mathsf{s} \cdot \underbrace{\mathsf{p} \cdot \mathsf{p} \ldots \mathsf{p}}_{k \text{ times}} \cdot (\Omega, \underline{0}, \epsilon) \cdot \epsilon \qquad \theta_k = \underbrace{\mathsf{p} \cdot \mathsf{p} \ldots \mathsf{p}}_{k \text{ times}} \cdot (\underline{0}, \Omega, \epsilon) \cdot \epsilon$$

These families of stacks have the following properties:

- $\forall k \leqslant n$, $(\underline{\mathtt{n}}, \epsilon, \xi_k) \to_{\overline{v}}^{k+1} (\underline{\mathtt{n}' + \mathtt{1}}, \epsilon, (\Omega, \underline{0}, \epsilon) \cdot \epsilon) \to_{\overline{v}} (\underline{0}, \epsilon, \epsilon)$,
- $\forall k > n$, $(\underline{\mathtt{n}}, \epsilon, \xi_k)$ diverges,
- $\forall k \geqslant n$, $(\underline{\mathtt{n}}, \epsilon, \theta_k) \to_{\overline{v}}^{k} (\underline{0}, \epsilon, (\underline{0}, \Omega, \epsilon) \cdot \epsilon) \to_{\overline{v}} (\underline{0}, \epsilon, \epsilon)$,
- $\forall k < n$, $(\underline{\mathtt{n}}, \epsilon, \theta_k)$ diverges.

Thanks to these properties we have:

$$\{\ (\xi_k, \mathtt{0}) \mid k \leqslant [\![\mathrm{J}]\!]_\rho\ \} \cup \{\ (\theta_k, \mathtt{0}) \mid [\![\mathrm{K}]\!]_\rho \leqslant k\ \} \subseteq \{\ (\underline{\mathtt{n}}, 0) \mid [\![\mathrm{J}]\!]_\rho \leqslant n \leqslant [\![\mathrm{K}]\!]_\rho\ \}^{\perp_\rho} = |\mathtt{Nat}[\mathrm{J}, \mathrm{K}]|_\rho^{\perp_\rho}$$

Now, suppose $(c, \mathrm{I}) \in |\mathtt{Nat}[\mathrm{J}, \mathrm{K}]|_\rho$. We have $(c, I) \in \{(\xi_{[\![\mathrm{J}]\!]_\rho}, 0), (\theta_{[\![\mathrm{K}]\!]_\rho}, 0)\}^{\perp_\rho}$. But this says that there exists some integer $n \in \mathbb{N}$ such that $(c, \epsilon) \downarrow^m \underline{\mathtt{n}}$ (by the safety property of the configurations in $\perp\!\!\!\perp$) and by Theorem 8 that $m \leqslant [\![I]\!]_\rho$. Moreover, it tells us also that $[\![\mathrm{J}]\!]_\rho \leqslant \underline{\mathtt{n}} \leqslant [\![\mathrm{K}]\!]_\rho$ otherwise we would have a diverging computation, hence the result. □

The internal completeness result can also be extended to first order functions.

Corollary 1 (Internal completeness for functions)**.** *Suppose* $a; \varnothing \models \mathrm{I} \Downarrow$*,* $a; \varnothing \models \mathrm{J} \Downarrow$*, and* $a; \varnothing \models \mathrm{K} \Downarrow$*. If* $((t, \epsilon), \mathrm{I}) \in |[b < \mathrm{J}] \cdot \mathtt{Nat}[a] \multimap \mathtt{Nat}[\mathrm{K}]|_{\{a \leftarrow n\}}$ *then* $(t\, \underline{\mathtt{n}}, \epsilon, \epsilon) \downarrow^m (\underline{\mathtt{k}}, \nu, \epsilon)$ *with* $m \leqslant [\![\mathrm{I} + \mathrm{J}]\!]_{\{a \leftarrow n\}}$ *and* $[\![\mathrm{K}]\!]_{\{a \leftarrow n\}} = \underline{\mathtt{k}}$*.*

Proof. Let $\rho = \{a \leftarrow n\}$. We need to show:

$$((t\, \underline{\mathtt{n}}, \epsilon), \mathrm{I} + \mathrm{J}) \in |\mathtt{Nat}[\mathrm{K}]|_{\{a \leftarrow n\}} \tag{2}$$

Then, we can conclude by Theorem 9 that $(t\, \underline{\mathtt{n}}, \epsilon, \epsilon) \downarrow^m (\underline{\mathtt{k}}, \nu, \epsilon)$ with $m \leqslant [\![\mathrm{I} + \mathrm{J}]\!]_{\{a \leftarrow n\}}$ and $\underline{\mathtt{k}} = [\![\mathtt{Nat}[\mathrm{K}]]\!]_{\{a \leftarrow n\}}$.

In order to prove 2, we prove first an intermediate lemma:

$$((\underline{\mathtt{n}}, \epsilon), \mathrm{J}) \in |[b < \mathrm{J}].\mathtt{Nat}[a]|_\rho \tag{3}$$

We have immediately that $((\underline{\mathtt{n}}, \epsilon), 0) \in |\mathtt{Nat}[a]|_\rho$. Hence for any $p \in \mathbb{N}$, we also have that $((\underline{\mathtt{n}}, \epsilon), 0) \in |\mathtt{Nat}[a]|_{\rho\{b \leftarrow p\}}$. Therefore we obtain

$$((\underline{\mathtt{n}}, \epsilon), \underbrace{\sum_{b<\mathrm{J}} 0}_{=0} + \mathrm{J}) \in |[b < \mathrm{J}].\mathtt{Nat}[a]|_\rho$$

Now we prove (2). Let $(\xi, \mathrm{I}') \in |\mathtt{Nat}[\mathrm{K}]|^{\perp_\rho}_{\{a \leftarrow n\}}$. We want to show that

$$((t\,\underline{\mathtt{n}}, \epsilon, \xi), [\![\mathrm{I} + \mathrm{J} + \mathrm{I}']\!]_\rho) \in \perp\!\!\!\perp$$

But, by (3), we have that:

$$(\underline{\mathtt{n}}.\xi, \mathrm{J} + \mathrm{I}') \in |[b < \mathrm{J}] \cdot \mathtt{Nat}[a] \multimap \mathtt{Nat}[\mathrm{K}]|^{\perp_\rho}_{\{a \leftarrow n\}}$$

Hence, by hypothesis we have

$$((t, \epsilon, \underline{\mathtt{n}}.\xi), [\![\mathrm{I} + \mathrm{J} + \mathrm{I}']\!]_\rho) \in \perp\!\!\!\perp$$

Which proves (2). □

The internal completeness theorem above is the last ingredient we need to prove the intensional soundness.

Theorem 10 (Intensional soundness)**.** *Let* $\varnothing; \varnothing; \varnothing \vdash_{\mathrm{I}} t : \mathtt{Nat}[\mathrm{J}, \mathrm{K}]$. *Then, we have* $t \Downarrow^n \underline{\mathtt{m}}$ *with* $n \leqslant |t| \cdot ([\![\mathrm{I}]\!] + 1)$ *and* $[\![\mathrm{J}]\!] \leqslant m \leqslant [\![\mathrm{K}]\!]$.

Proof. By assumption $\varnothing; \varnothing; \varnothing \vdash_{\mathrm{I}} t : \mathtt{Nat}[\mathrm{J}, \mathrm{K}]$. Hence, by Theorem 7

$$((t, \epsilon), \mathrm{I}) \in [\![\mathtt{Nat}[\mathrm{J}, \mathrm{K}]]\!]$$

Moreover, by the fact that t is well typed we have $[\![\mathrm{J}]\!] \leqslant [\![\mathrm{K}]\!]$ and that $\models \mathrm{I} \Downarrow$. So, by Internal Completeness we have $n, m \in \mathbb{N}$ such that $n \leqslant [\![\mathrm{I}]\!]$ and $[\![\mathrm{J}]\!] \leqslant m \leqslant [\![\mathrm{K}]\!]$ and:

$$(t, \epsilon, \epsilon) \downarrow^n (\underline{\mathtt{m}}, \mu, \epsilon)$$

This and Lemma 4 show then that $t \Downarrow^n \underline{\mathtt{m}}$ and $n \leqslant |t| \cdot ([\![\mathrm{I}]\!] + 1)$. □

The type system of dℓPCF has been designed to be relatively complete with respect to the evaluation on Krivine's machine. Relative completeness is obtained by considering a universal equational program $\mathcal{U}$. Using relative completeness for programs and for functions respectively, we can show that on programs and functions realizability and typability are equivalent.

Theorem 11 (Coincidence)**.**
Let $\rho \models^{\mathcal{U}} \phi; \Phi$*. Then,*

1. $\phi; \Phi; \varnothing \vdash_{\mathrm{I}}^{\mathcal{U}} t : \mathtt{Nat}[\mathrm{J}, \mathrm{K}] \Longleftrightarrow ((t, \epsilon), \mathrm{I}) \in |\mathtt{Nat}[\mathrm{J}, \mathrm{K}]|_{\rho}^{\mathcal{U}}$.
2. *Moreover, if* $\rho \models^{\mathcal{U}} \phi, a; \Phi$*, we have* $\phi, a; \Phi; \varnothing \vdash_{\mathrm{I}}^{\mathcal{U}} t : [b < \mathrm{J}] \cdot \mathtt{Nat}[a] \multimap \mathtt{Nat}[\mathrm{K}] \Longleftrightarrow ((t, \epsilon), \mathrm{I}) \in |[b < \mathrm{J}] \cdot \mathtt{Nat}[a] \multimap \mathtt{Nat}[\mathrm{K}]|_{\rho}^{\mathcal{U}}$

Proof. For both the points, the direction $\Rightarrow$ follows directly by Theorem 7. The direction $\Leftarrow$ follows instead by:

- The internal completeness Theorem 9 and by Relative Completeness for programs (Theorem 4) for the first point.
- The internal completeness for functions (Corollary 1) and Relative Completeness for functions (Theorem 5).

□

The above theorem ensures that we can reason about the type system in an abstract way by using the realizability model.

6. Conclusions and Related works

Realizability techniques are nowadays a standard tool to reason about program behavior [26, 6, 5, 17]. The gain in using such techniques is an approach to formal reasoning about programs that abstracts the language properties from the concrete syntax. An example is the semantic soundness proof we presented here. However, classical works on program behavior do not consider the quantitative aspects of programs.

Quantitative realizability models have been studied before in the context of linear logic. Hofmann and Scott in [16] have studied a realizability model for Bounded linear logic. This model has then been further revisited by Dal Lago and Hofmann in [11] and by Brunel in [7]. The main technical difference between our work and the previous ones is that our model, analogously to $\mathsf{d}\ell\mathsf{PCF}$, is parametrized on an equational program and on an assignment. In this respect, our work can be understood not as a unique model but as a set of models that can be instantiated as needed. Furthermore, ours is also the first quantitative realizability model that can be used to reason about the full PCF language. It is interesting to note that the information given by $\mathsf{d}\ell\mathsf{PCF}$ types is enough to reason about all the terminating recursive programs. This permits to avoid the use of extra-techniques like step-indexing [2] usually employed in this setting. An obvious limitation of

our approach is that index terms, differently from step-indexing, cannot be used to reason about non-terminating programs. Further comparisons of the two approaches are left for future investigations.

Another motivation for our study is the possibility to internalize the notion of *forcing* in classical realizability models as shown by Krivine in [21] and Miquel in [24]. These works established connections between forcing conditions of logical principles and program transformations. Starting from these works, Brunel in [7] has shown that the quantitative part of a quantitative realizability model can be seen as the result of the internalization of a specific forcing model inside a simple (non-quantitative) realizability model. Forcing can also be useful in an intuitionistic framework as shown in [18], where it is used to generalize step-indexing. Since they also use forcing to account for term fixpoints, it would be interesting to explore the link between these frameworks. We intend to explore the formal links between $\mathsf{d}\ell\mathsf{PCF}$ and forcing.

Recently, Dal Lago and Petit in [12] have proposed a type system similar to $\mathsf{d}\ell\mathsf{PCF}$ but related to call-by-value rather than call-by-name evaluation. The grammar of types they use is slightly different from the one of $\mathsf{d}\ell\mathsf{PCF}$. So, our model cannot be applied straightforwardly to their framework. We leave for further research the adaptation of our technique.

[1] Amadio, R., 2005. Synthesis of max-plus quasi-interpretations. Fund. Inform. 65, 29–60.

[2] Appel, A. W., McAllester, D., Sep. 2001. An indexed model of recursive types for foundational proof-carrying code. ACM Transactions on Programming Languages and Systems 23 (5), 657–683.

[3] Baillot, P., Terui, K., 2004. Light types for polynomial time computation in lambda-calculus. In: IEEE LICS. pp. 266–275.

[4] Bellantoni, S., Cook, S., 1992. A new recursion-theoretic characterization of the polytime functions. Computational Complexity 2 (2), 97–110.

[5] Benton, N., Hur, C.-K., 2009. Biorthogonality, step-indexing and compiler correctness. In: ICFP '09. ACM, New York, NY, USA, pp. 97–108.

[6] Birkedal, L., Støvring, K., Thamsborg, J., 2010. Realisability semantics of parametric polymorphism, general references and recursive types. MSCS 20 (4), 655–703.

[7] Brunel, A., 2012. Quantitative classical realizability. `http://arxiv.org/abs/1201.4307`.

[8] Dal Lago, U., Gaboardi, M., 2011. Linear dependent types and relative completeness. In: IEEE LICS '11. pp. 133–142.

[9] Dal Lago, U., Gaboardi, M., 2012. Linear dependent types and relative completeness. Logical Methods in Computer Science 8.

[10] Dal Lago, U., Hofmann, M., 2009. Bounded linear logic, revisited. In: TLCA. Vol. 5608 of LNCS. Springer, pp. 80–94.

[11] Dal Lago, U., Hofmann, M., 2011. Realizability models and implicit complexity. Theoretical Computer Science 412 (20), 2029 – 2047.
URL http://dx.doi.org/10.1016/j.tcs.2010.12.025

[12] Dal Lago, U., Petit, B., 2012. Linear dependent types in a call-by-value scenario. In: PPDP '12. ACM, New York, NY, USA, pp. 115–126.

[13] Girard, J., Scedrov, A., Scott, P., 1992. Bounded linear logic. TCS 97 (1), 1–66.

[14] Girard, J.-Y., 2001. Locus solum: From the rules of logic to the logic of rules. Mathematical Structures in Computer Science 11 (03), 301–506.

[15] Hofmann, M., 2000. Programming languages capturing complexity classes. ACM SIGACT News 31, 31–42.

[16] Hofmann, M., Scott, P. J., 2004. Realizability models for bll-like languages. TCS 318 (1-2), 121–137.

[17] Jaber, G., Tabareau, N., 2010. Krivine realizability for compiler correctness. LOLA.
URL http://hal.archives-ouvertes.fr/hal-00475210/

[18] Jaber, G., Tabareau, N., Sozeau, M., Jun. 2012. Extending Type Theory with Forcing. In: Proceedings of LICS'12.

[19] Kristiansen, L., Jones, N., 2005. The flow of data and the complexity of algorithms. In: Cie: New Computational Paradigms. Vol. 3526 of LNCS. Springer, pp. 289–304.
URL http://dx.doi.org/10.1007/11494645_33

[20] Krivine, J.-L., 2009. Realizability in classical logic. Panoramas et synthèses 27, 197–229.
URL http://hal.archives-ouvertes.fr/hal-00154500

[21] Krivine, J.-L., 2011. Realizability algebras: a program to well order R. LMCS 7 (3).

[22] Leivant, D., Marion, J.-Y., 1993. Lambda calculus characterizations of poly-time. In: TLCA '93. Vol. 664 of LNCS. Springer, pp. 274–288.

[23] Marion, J.-Y., Moyen, J.-Y., 2000. Efficient first order functional program interpreter with time bound certifications. In: LPAR. Vol. 1955. Springer, pp. 25–42.

[24] Miquel, A., 2011. Forcing as a program transformation. In: IEEE LICS. pp. 197–206.

[25] Moyen, J.-Y., Aug. 2009. Resource control graphs. ACM TOCL 10 (4), 29:1–29:44.

[26] Pitts, A., 2000. Parametric polymorphism and operational equivalence. MSCS 10, 321–359.

[27] Terui, K., 2011. Computational ludics. Theoretical Computer Science 412 (20), 2048–2071.

[28] Xi, H., 2007. Dependent ML: An approach to practical programming with dependent types. J. of Funct. Progr. 17 (2), 215–286.