A Stable Programming Language

Luca Paolini [1]

Dipartimento di Informatica
Università degli Studi di Torino
Corso Svizzera, 185
10149 Torino - Italy

E-MAIL: paolini@di.unito.it

Abstract

It is well-known that stable models (as dI-domains, qualitative domains and coherence spaces) are not fully abstract for the language $\mathcal{PCF}$. This fact is related to the existence of stable parallel functions and of stable functions that are not monotone with respect to the extensional order, which cannot be defined by programs of $\mathcal{PCF}$.

In this paper, a paradigmatic programming language named $\mathcal{StPCF}$ is proposed, which extends the language $\mathcal{PCF}$ with two additional operators. The operational description of the extended language is presented in an effective way, although the evaluation of one of the new operators cannot be formalized in a PCF-like rewrite system.

Since $\mathcal{StPCF}$ can define all finite cliques of coherence spaces the above gap with stable models is filled, consequently stable models are fully abstract for the extended language.

1 Introduction

$\mathcal{PCF}$ is a paradigmatic example of a typed functional programming language, which arose from the language LCF introduced by Dana Scott as a "calculus or algebra" for the purpose of studying logical properties of programs [1]. In time, $\mathcal{PCF}$ has become the most popular language investigated in the field of semantics of programming languages. In fact many kinds of mathematical structures have been related to it (examples are in [2,3,4,5,6,7,8,9,10,11,12,13,14,15]).

Much investigation effort has been devoted to the full abstraction problem (a key notion introduced by Robin Milner in [16]). Two programs are operationally equivalent whenever they are interchangeable "in all contexts" without affecting the ob-

[1] Paper partially supported by IST-2001-33477 DART Project, MIUR-Cofin'02 PROTOCOLLO Project, MIUR-Cofin'04 FOLLIA Project.

servable outcome of the computation (this equivalence is also called contextual or observational). In contrast, according to a denotational semantics the meaning of a program lies in its denotation; hence, two programs are denotationally equivalent in a given model only when they have the same denotation in the model itself. If the denotational equivalence implies the operational one, then the model is *correct*. If the reverse implication holds then the model is *complete*. If the equivalences coincide then the model is *fully abstract*.

Independently, Gordon Plotkin [15] and Vladmir Sazonov [17] have shown that the standard (with respect to the interpretation) Scott-continuous model [18] is not fully abstract for $\mathcal{PCF}$. In a nutshell, the mismatch may be explained by the fact that there is a function called *parallel-or* which is Scott-continuous but cannot be defined (i.e. programmed) in $\mathcal{PCF}$. In particular, Plotkin extended $\mathcal{PCF}$ with a *parallel-if* operator and shown that the Scott-continuous model is fully abstract for this extended language. Note that parallel-if and parallel-or are interdefinable [19]. The problem of finding fully abstract models of unextended $\mathcal{PCF}$ has been resolved in [2,10,16,12,13]. On the other hand, many models have been proved to be fully abstract with respect to languages derived from $\mathcal{PCF}$, as in [20,8,21,11,22,15]. Furthermore, the investigations on $\mathcal{PCF}$ have been fruitfully related to many other studies, for instance to works on higher-type computability, on sequential functions and degrees of parallelism [23,5,24,25,26,27,9,28,11,29,30,31,32].

The notions of stability and dI-domains have been defined by Gérard Berry in [33]. dI-domains are Scott-domains satisfying two additional axioms; stable functions produce some amount of "output information" only when a minimum amount of information is incoming. dI-domains and stable functions form a cartesian closed category. The theory of stable functions has been rediscovered, independently, by Jean-Yves Girard as a semantic counterpart of his theory of dilators [34] and he used stability in order to provide a model for second order polymorphic λ-calculus (the System F) [35]. Girard has also introduced qualitative domains [35] and coherence spaces [36], which are cartesian closed full subcategories of the category of dI-domains. All these categories contain the objects and morphisms in the range of the standard interpretation of $\mathcal{PCF}$, and without ambiguity they will be called "stable domains". Like the standard Scott-continuous model, the standard stable models are not fully abstract with respect to $\mathcal{PCF}$, because there exist stable functions with a finite domain of definition that cannot be programmed in $\mathcal{PCF}$. In particular, there exist stable functions which have some parallel flavour, like the *Gustave function* (Gustave is Berry's nickname), and there exist stable functions that are not monotone with respect to the extensional order [33].

A natural question is, how to extend $\mathcal{PCF}$ in such a way that the stable models are fully abstract for it? This question was already considered in many papers [37,11,14]. In this paper the answer is given. The language $\mathcal{StPCF}$ is obtained by extending $\mathcal{PCF}$ with two operators: `gor` and `strict`?. The `gor` operator corresponds to a Gustave-like `or` function, while the `strict`? operator corresponds to

a non extensional-monotone function. It is shown that the coherence space model is fully abstract with respect to $\mathcal{S}t\mathcal{P}CF$. In particular, each finite clique of a coherence space which is the interpretation of a $\mathcal{P}CF$-type is the denotation of a $\mathcal{S}t\mathcal{P}CF$-program. The results holds for the other stable domains considered above.

In particular, the above question was approached by Trevor Jim and Albert Meyer in [37]. They have shown some negative results. Let the contextual-preorder be the usual operational preorder defined by comparing the behaviour of terms in all contexts. On the other hand, let the applicative-preorder be defined by observing only the behaviour of terms applied to sequences of terms. It is well-known that the previous preorder relations coincide for $\mathcal{P}CF$. First of all, Jim and Meyer define in a denotational way the *true-separator* function which is a stable function that corresponds to a boolean version of `strict`?. Hence, they show that the true-separator breaks down the coincidence between the applicative-preorder on terms and the contextual-preorder. Finally, they show that with the class of "linear ground operational rules" (defining $\mathcal{P}CF$-like rewrite systems) the coincidence mentioned before cannot be broken. Therefore, a fully abstract extension of $\mathcal{P}CF$ using only operators having a "linear ground operational description" does not exist. Jim and Meyer[2] state,

> "However, one important result about cpos is not known for stable domains, namely, full abstraction with respect to some extension of PCF analogous to the parallel-or extension which Plotkin and Sazonov provided for the cpo model. What might a symbolic-evaluator for an extended PCF look like if it was well matched– fully abstract–with the stable model? We conclude that such an evaluator will have to be unusual looking: it cannot be specified by the kind of term-rewriting based evaluation rules known for PCF and its extensions.
>
> The significance of this negative result hinges heavily on how drastic we judge it to go beyond the scope of PCF-like rules. It is of course possible that some operational behavior that we declare to be non-PCF-like, in our technical sense, will nevertheless offer a useful extension of PCF for which stable domains are fully abstract. ..... (The general benefits of structured approaches to operational semantics and connection to full abstraction are discussed in [38,22])".

Their paper gives a sufficient motivation for the study of the effective operational description, given in this paper, of `strict`?. But `strict`? is also a strongly stable operator (in the sense of Antonio Bucciarelli and Thomas Ehrhard [25,39,40]) that can be defined in $\mathcal{P}CF$ extended either with control operators [41,42] or with Longley's `H` operator [11]. Thus such extensions cannot be evaluated through a $\mathcal{P}CF$-like rewrite system. Informally, the language $\mathcal{P}CF$+`H` provides an answer to the question of how far one can travel in languages endowed with control operators without sacrificing the functional nature (i.e. extensionality) of programs. Presently no operational semantics has been given for `H` in a direct way, albeit `H` can be defined

[2] [37], page 664.

in actual programming languages [43]. Hence, the given evaluation of strict? is related to the interest for the operational description of the H operator.

In conclusion, its effective evaluation makes $\mathcal{S}t\mathcal{P}CF$ an interesting paradigmatic purely functional typed programming language that can be used as a core for the development of real functional languages. The equivalence between $\mathcal{S}t\mathcal{P}CF$ programs can be tackled by the elegant mathematical tools provided by stable models.

1.1 Outline of the paper

After an informal presentation the language $\mathcal{S}t\mathcal{P}CF$ is formalized in Section 2. In Section 3 an effective operational semantics is given using a straightforward inductive closure of schematic big-step operational rules. This section ends with some discussions about $\mathcal{S}t\mathcal{P}CF$, in particular on the question of relations between strict? and $\mathcal{P}CF$-like rewrite system. Section 4 contains some cumbersome technical details needed for the proof of Theorem 3.3. In Section 5 the basic notions on coherence spaces are stated. The interpretation of $\mathcal{S}t\mathcal{P}CF$ on coherence spaces is given and its adequacy and correctness are proved in Sections 6 and 7 respectively, by quite standard proofs. Section 8 is devoted to the definability of finite cliques and to the full-abstraction result. In this section many examples have been presented. Conclusions, open questions and future works are presented in Section 9.

2 Syntax of $\mathcal{S}t\mathcal{P}CF$

$\mathcal{S}t\mathcal{P}CF$ is an extension of a $\mathcal{P}CF$-like language without explicit truth-values which are coded on integers (zero means "true" while any other numeral stands for "false").

Definition 2.1 ($\mathcal{S}t\mathcal{P}CF$-Types) *Let ι be the only* ground *type.*
Types *of $\mathcal{S}t\mathcal{P}CF$ are generated by the following grammar:*

$$\sigma ::= \iota \mid (\sigma \rightarrowtail \tau)$$

where $\sigma, \tau, ...$ are metavariables ranging over types of $\mathcal{S}t\mathcal{P}CF$.

As customary, $\rightarrowtail$ associates to right. Hence $\sigma_1 \rightarrowtail \sigma_2 \rightarrowtail \sigma_3$ is an abbreviation for $\sigma_1 \rightarrowtail (\sigma_2 \rightarrowtail \sigma_3)$. Furthermore, it is easy to see that all types τ have the shape $\tau_1 \rightarrowtail ... \rightarrowtail \tau_n \rightarrowtail \iota$, for some type $\tau_1, ..., \tau_n$ where $n \geq 0$.

Definition 2.2 ($\mathcal{S}t\mathcal{P}CF$-Words) *Let Var be a denumerable set of variables.*
Words of $\mathcal{S}t\mathcal{P}CF$ are produced by the following grammar:

$$\begin{aligned} \mathtt{M} ::= \ & \mathtt{x} \mid (\lambda \mathtt{x}^{\sigma}.\mathtt{N}) \mid (\mathtt{PQ}) \mid \mathtt{Y}_{\sigma} \\ & \mid \mathtt{if} \mid \mathtt{succ} \mid \mathtt{pred} \mid \tilde{\mathtt{n}} \mid \mathtt{strict?} \mid \mathtt{gor} \end{aligned}$$

where $\mathtt{x} \in Var$ *and* σ *is a type, while* $\mathtt{M}, \mathtt{N}, \mathtt{P}, \mathtt{Q}, \ldots$ *are metavariables ranging over the words of* $\mathcal{S}t\mathcal{P}CF$ *and* $\tilde{\mathtt{n}}, \tilde{\mathtt{m}}, \ldots$ *are metavariables ranging over numerals, namely the denumerable constants* $\tilde{\mathtt{0}}, \tilde{\mathtt{1}}, \tilde{\mathtt{2}}, \ldots\ldots$

As customary, $\mathtt{MNP}$ will be used as an abbreviation for $(\mathtt{MN})\mathtt{P}$ while $\lambda\mathtt{x}^{\sigma}\mathtt{y}^{\tau}.\mathtt{P}$ is an abbreviation for $(\lambda\mathtt{x}^{\sigma}.(\lambda\mathtt{y}^{\tau}.\mathtt{P}))$. The set of *free variables* of a term $\mathtt{M}$ is denoted by $\mathrm{FV}(\mathtt{M})$ and it is defined as for $\mathcal{P}CF$ extended with $\mathrm{FV}(\mathtt{gor}) = \mathrm{FV}(\mathtt{strict?}) = \emptyset$. A term $\mathtt{M}$ is *closed* if and only if $\mathrm{FV}(\mathtt{M}) = \emptyset$, otherwise M is said to be *open*. Words are considered up to α-equivalence (denoted $\equiv$ in the following), namely a bound variable can be renamed provided no free variable is captured. Moreover, as customary, $\mathtt{M}[\mathtt{N}/\mathtt{x}]$ denotes the capture-free substitution of all free occurrences of $\mathtt{x}$ in $\mathtt{M}$ by $\mathtt{N}$.

The λ-abstraction is the only binder as customary in λ-calculi, $\mathtt{Y}_{\sigma}$ is the recursion operator of type $(\sigma \rightarrowtail \sigma) \rightarrowtail \sigma$ for each type σ, numerals represent natural numbers having type ι, while $\mathtt{succ}$ and $\mathtt{pred}$ are successor and predecessor operators having type $\iota \rightarrowtail \iota$ (for us $\mathtt{pred}\,\tilde{\mathtt{0}}$ will be undefined). Moreover, $\mathtt{if}$ is a conditional operator having type $\iota \rightarrowtail \iota \rightarrowtail \iota \rightarrowtail \iota$; it checks if the first argument is zero or not, in order to choose how to forward the evaluation. In order to fill the gap between $\mathcal{P}CF$ functions and stable morphisms, the operators $\mathtt{gor}$ and $\mathtt{strict?}$ are introduced. They have respectively type $\iota \rightarrowtail \iota \rightarrowtail \iota \rightarrowtail \iota$ and $(\iota \rightarrowtail \iota) \rightarrowtail \iota$.

The operator $\mathtt{gor}$ corresponds essentially to a parallel Gustave-like "logical or". This kind of function was introduced independently by Kleene[3], by Berry [4,33] and by Coppo, Dezani-Ciancaglini, Ronchi Della Rocca [44]. Let $\mathtt{R} \equiv \mathtt{gor}\,\mathtt{P}_0\,\mathtt{P}_1\,\mathtt{P}_2$ be a "well-typed" term and let eval be the evaluation procedure. In an informal way, the evaluation of $\mathtt{R}$ can be described as follows:

- if $\mathsf{eval}(\mathtt{P}_0) = \tilde{\mathtt{0}}$ and $\mathsf{eval}(\mathtt{P}_1) = \tilde{\mathtt{n}} \neq \tilde{\mathtt{0}}$ then $\mathsf{eval}(R) = \tilde{\mathtt{0}}$,
- if $\mathsf{eval}(\mathtt{P}_1) = \tilde{\mathtt{0}}$ and $\mathsf{eval}(\mathtt{P}_2) = \tilde{\mathtt{n}} \neq \tilde{\mathtt{0}}$ then $\mathsf{eval}(R) = \tilde{\mathtt{1}}$,
- if $\mathsf{eval}(\mathtt{P}_2) = \tilde{\mathtt{0}}$ and $\mathsf{eval}(\mathtt{P}_0) = \tilde{\mathtt{n}} \neq \tilde{\mathtt{0}}$ then $\mathsf{eval}(R) = \tilde{\mathtt{2}}$,
- undefined otherwise.

The evaluation of $\mathtt{strict?}$ is subtler. This kind of operator was first considered by Berry in [33], and its use is crucial in the paper of Jim and Meyer [37] (in fact, their "true-separator" corresponds straightforwardly to a boolean version of $\mathtt{strict?}$). Let $\mathtt{strict?}\,\mathtt{M}$ be a "well-typed" term, let $\uparrow$ and $\downarrow$ denote respectively "divergence" and "convergence" of the evaluation (being a partial function) and let Ω_{ι} denote a divergent term of type ι. In an informal way, a nonconstructive description of the evaluation of $\mathtt{strict?}\,\mathtt{M}$ is

- if $\mathsf{eval}(\mathtt{M}\tilde{\mathtt{0}}) \downarrow$ and $\mathsf{eval}(\mathtt{M}\Omega_{\iota}) \uparrow$ then $\mathsf{eval}(\mathtt{strict?}\,\mathtt{M}) = \tilde{\mathtt{0}}$,
- if $\mathsf{eval}(\mathtt{M}\tilde{\mathtt{0}}) \downarrow$ and $\mathsf{eval}(\mathtt{M}\Omega_{\iota}) \downarrow$ then $\mathsf{eval}(\mathtt{strict?}\,\mathtt{M}) = \tilde{\mathtt{1}}$,
- undefined otherwise.

[3] See [3,25] for references.

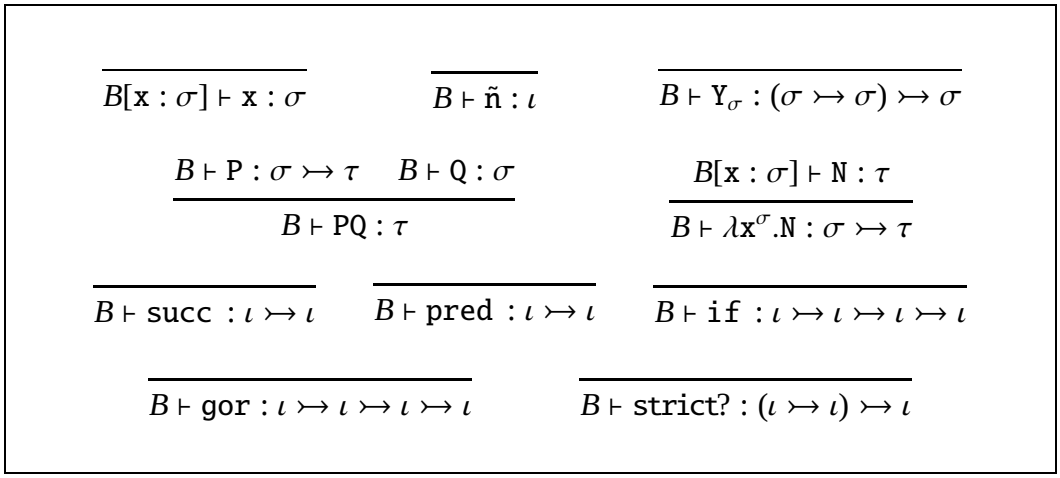


Figure 1. Typing Rules

Note that the expected type for strict? implies that $\iota \rightarrowtail \iota$ is the type for M; thus, if the evaluation of $\mathtt{M}\tilde{\mathtt{0}}$ converges (to a numeral) then strict? tells us whether M uses $\tilde{\mathtt{0}}$ or not. Note that an operator const? corresponding to $\lambda\mathtt{f}^{\iota\rightarrowtail\iota}.\mathtt{if}\,(\mathtt{strict?f})\,\tilde{\mathtt{1}}\,\tilde{\mathtt{0}}$ could be used in place of strict?. Clearly $\lambda\mathtt{x}^\iota.\tilde{\mathtt{9}}$ is (extensionally) more defined than $\lambda\mathtt{x}^\iota.\mathtt{if}\,\mathtt{x}\,\tilde{\mathtt{9}}\,\tilde{\mathtt{9}}$. Thus strict? and const? are not monotone with respect to the extensional order, in fact $\mathtt{strict?}(\lambda\mathtt{x}^\iota.\tilde{\mathtt{9}}) \equiv \tilde{\mathtt{1}}$ while $\mathtt{strict?}(\lambda\mathtt{x}^\iota.\mathtt{if}\,\mathtt{x}\,\tilde{\mathtt{9}}\,\tilde{\mathtt{9}}) \equiv \tilde{\mathtt{0}}$. On the other hand, $\mathtt{const?}(\lambda\mathtt{x}^\iota.\tilde{\mathtt{9}}) \equiv \tilde{\mathtt{0}}$ while $\mathtt{const?}(\lambda\mathtt{x}^\iota.\mathtt{if}\,\mathtt{x}\,\tilde{\mathtt{9}}\,\tilde{\mathtt{9}}) \equiv \tilde{\mathtt{1}}$.

Definition 2.3 ($\mathcal{S}t\mathcal{P}CF$-Terms and $\mathcal{S}t\mathcal{P}CF$-Programs)
A basis *B is a partial function from Var to types of $\mathcal{S}t\mathcal{P}CF$ with a finite domain of definition. If B is a basis then $B[\mathtt{x}:\sigma]$ denotes the basis such that*

$$B[\mathtt{x}:\sigma](\mathtt{y}) = \begin{cases} \sigma & \text{if } \mathtt{y} \equiv \mathtt{x}, \\ B(\mathtt{y}) & \text{otherwise.} \end{cases}$$

Moreover, the basis B such that dom$(B) = \{\mathtt{x}_1, ..., \mathtt{x}_n\}$ $(n \in \mathbb{N})$ and $B(x_i) = \sigma_i$, for $1 \leq i \leq n$ can be denoted by $\mathtt{x}_1 : \sigma_1, ..., \mathtt{x}_n : \sigma_n$ without repetition of variables.
A word M *of $\mathcal{S}t\mathcal{P}CF$ is a (well-typed)* term *when it is the subject of a* typing judgment *(often simply* typing*) of the shape $B \vdash \mathtt{M} : \sigma$ which is the conclusion of a derivation built by the rules of Figure 1. A* program *is a closed well-typed term.*

As usual, we write $B \vdash M : \sigma$ when the typing is a conclusion of a derivation built using the rules of Figure 1, while we write $B \nvdash M : \sigma$ when such derivation does not exist. If the basis of a typing is empty then we simply write $\vdash M : \sigma$.

Definition 2.4 (B^σ-Contexts)
Let σ be a type and $[^\sigma]$ be a new symbol, called the σ-hole. If P *is a $\mathcal{S}t\mathcal{P}CF$-word then $C[^\sigma], D[^\sigma], ...$ will be used in the following as metavariables, ranging over words produced by the following grammar:*

$$C[^\sigma] ::= \mathtt{P} \mid [^\sigma] \mid (\lambda\mathtt{x}^\tau.C[^\sigma]) \mid (C[^\sigma]D[^\sigma])$$

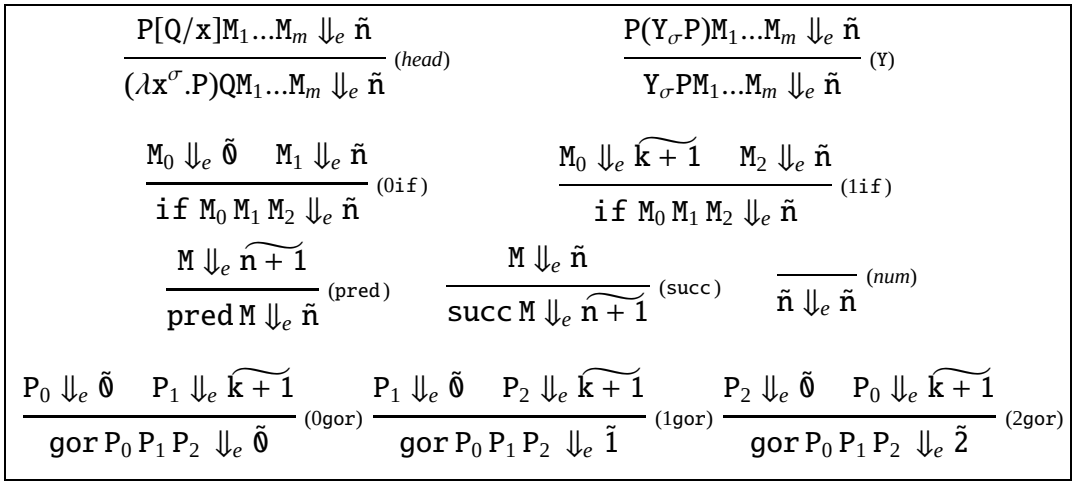


Figure 2. Operational Evaluation, Part I

If $B \vdash M : \sigma$ for some basis B, then $C[M]$ denotes the word obtained by replacing all occurrences of holes in $C[\sigma]$ by M. A word $C[\sigma]$ is called B^{σ}-context, *if there is a basis B' such that $B' \vdash C[M] : \iota$ whenever $B \vdash M : \sigma$ holds.*

It is useful to name some terms. In particular, Ω_{σ} will denote the term defined by induction σ as follows:

$$\Omega_{\iota} \equiv Y_{\iota}(\lambda x^{\iota}.x)\,, \qquad \Omega_{\mu\to\tau} \equiv \lambda x^{\mu}.\Omega_{\tau}\,.$$

By using Ω_{σ}, it is possible to define terms Y_{σ}^{k} ($k \in \mathbb{N}$) in the following way:

$$Y_{\sigma}^{0} \equiv \Omega_{(\sigma\to\sigma)\to\sigma}\,, \qquad Y_{\sigma}^{k+1} \equiv \lambda x^{\sigma\to\sigma}.x(Y_{\sigma}^{k}\, x)\,.$$

3 Structured Operational Semantics

The operational evaluation of $\mathcal{S}t\mathcal{P}CF$ will be given in an effective way, by a structured operational semantics [45,46].

Definition 3.1 *Let $\Downarrow_e$ be the* evaluation relation *associating a program M to a numeral $\tilde{n}$ whenever a judgment of the shape*

$$M \Downarrow_e \tilde{n}$$

can be proved by rules of the formal system defined in Figures 2 [4] and 3.
If there is a numeral $\tilde{n}$ such that $M \Downarrow_e \tilde{n}$ then we write $M \Downarrow_e$, otherwise we write $M \Uparrow_e$.

[4] Figure 2 is sufficient for the evaluation of $\mathcal{S}t\mathcal{P}CF$-programs without occurrences of strict?. The operational behavior of strict?M is a little more complex than the other operators. A constructive operational description for strict? is given by rules in Figure 3.

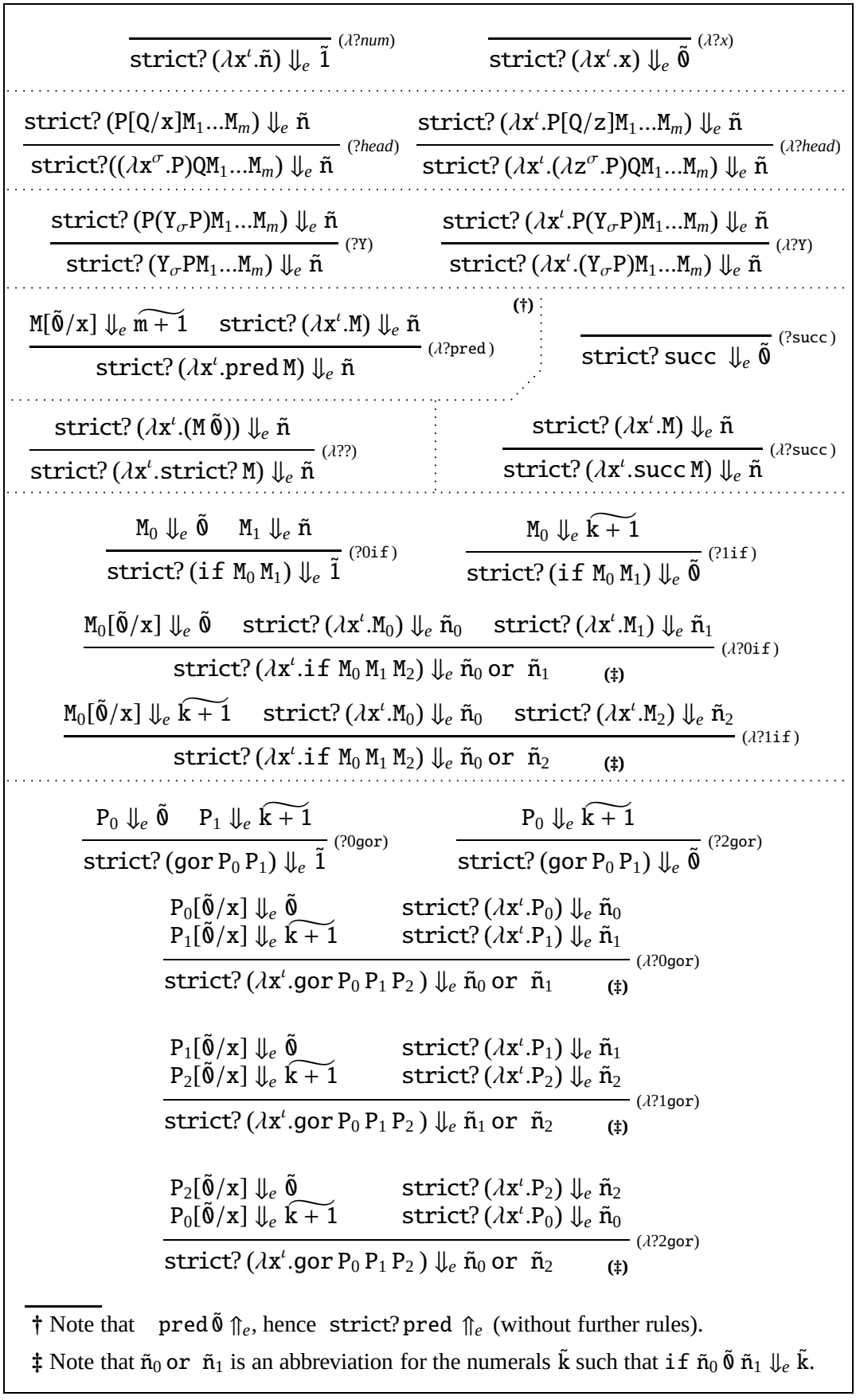


Figure 3. Operational Evaluation, Part II

The relation $\Downarrow_e$ implements a *call-by-name* parameter passing policy, since the arguments of abstractions are substituted without being evaluated. It does not implement a *lazy* (or *weak*) evaluation strategy, since reductions under λ-abstractions are taken into account (for example in the (λ?*head*) rule).

Since terms are only of interest as they are part of programs, we can regard terms with the same type as operationally equivalent if they can be freely substituted for each other in a program without affecting the behavior of the program itself.

Definition 3.2 (Operational Equivalence) *Suppose* $B \vdash \mathtt{M} : \sigma$, $B \vdash \mathtt{N} : \sigma$.

(i) $\mathtt{M} \lessapprox_\sigma \mathtt{N}$ *whenever* $C[\mathtt{M}] \Downarrow_e \tilde{\mathtt{n}}$ *for some numeral* $\tilde{\mathtt{n}}$ *implies that* $C[\mathtt{N}] \Downarrow_e \tilde{\mathtt{n}}$, *for all* B^σ*-contexts* $C[\sigma]$ *such that* $FV(C[\mathtt{M}]) = FV(C[\mathtt{N}]) = \emptyset$.
(ii) $\mathtt{M} \approx_\sigma \mathtt{N}$ *if and only if* $\mathtt{M} \lessapprox_\sigma \mathtt{N}$ *and* $\mathtt{N} \lessapprox_\sigma \mathtt{M}$.

It is easy to check that $\approx_\sigma$ is a congruence relation, i.e. an equivalence relation closed under contexts. Sometimes $\approx_\sigma$ is called observational or contextual equivalence.

Theorem 3.3 and Theorem 3.4 formalize our intuition on the operational behaviour of $\mathtt{strict?}$ and of the terms Ω_σ and $\mathtt{Y}^k_\sigma$ defined at the end of the Section 2. They will be useful in order to decrease the complexity of the proof of Lemma 7.3.

Theorem 3.3 *Let* $\vdash \mathtt{M} : \iota \rightarrowtail \iota$.

(i) $\mathtt{strict?}\,\mathtt{M} \Downarrow_e \tilde{\mathtt{0}}$ *if and only if* $\mathtt{M}\tilde{\mathtt{0}} \Downarrow_e$ *and* $\mathtt{M}\Omega_\iota \Uparrow_e$.
(ii) $\mathtt{strict?}\,\mathtt{M} \Downarrow_e \tilde{\mathtt{1}}$ *if and only if* $\mathtt{M}\tilde{\mathtt{0}} \Downarrow_e$ *and* $\mathtt{M}\Omega_\iota \Downarrow_e$.

Proof. The proof follows by Lemmas 4.1, 4.2 and 4.3 (Section 4). □

Theorem 3.4 *Let* $\mathtt{M}_0, ..., \mathtt{M}_m$ *be a sequence of terms* ($m \geq 0$).

(i) *If* $\Omega_\sigma \mathtt{M}_0 ... \mathtt{M}_m$ *is a program then* $\Omega_\sigma \mathtt{M}_0 ... \mathtt{M}_m \Uparrow_e$.
(ii) *Let* $\mathtt{Y}_\sigma \mathtt{M}_0 ... \mathtt{M}_m$ *be a program.*
$\mathtt{Y}_\sigma \mathtt{M}_0 ... \mathtt{M}_m \Downarrow_e \tilde{\mathtt{n}}$ *if and only if* $\mathtt{Y}^k_\sigma \mathtt{M}_0 ... \mathtt{M}_m \Downarrow_e \tilde{\mathtt{n}}$, *for some* $k \in \mathbb{N}$.

Proof. (i) The proof can be done by induction on m.
(ii) Both implications can be proved by induction on derivations proving the hypothesis.
□

3.1 Some Remarks

In the literature $\mathcal{PCF}$ is often presented with booleans and some operator on them. Only integers have been used here, since the differences between the two formalizations are irrelevant for our purposes. Thus, without loss of generality, some notions

formalized by Jim and Meyer[5] will be adapted to this setting in a natural way, in order to explicitly relate this paper to their one.

An extension of $\mathcal{PCF}$ is *conservative*[5] when it contains all programs of $\mathcal{PCF}$ and moreover, if $\mathtt{M}$ is one of such programs then the outcomes of the evaluation of $\mathtt{M}$ in both $\mathcal{PCF}$ and its extension coincide (either diverging or converging on the same numeral $\tilde{\mathtt{n}}$). $\mathcal{StPCF}$ is clearly conservative!
Stable models will be introduced in Section 5. A stable model is *preorder-adequate*[5] for an extension of $\mathcal{PCF}$ whenever a term $\mathtt{M}$ is "less or equal (in the model)" of a term $\mathtt{N}$ then $\mathtt{M}$ is "operationally (contextually) less or equal" of $\mathtt{N}$.
In [37] a family of small-step operational rules for conservative extensions of $\mathcal{PCF}$ is studied, as a kind of "rewriting system" [47]. Jim and Meyer remark that almost all the reduction rules considered in literature for extensions of $\mathcal{PCF}$ (as in case of the join operator [20], the parallel-or and the existential operators [15,17]) are instances of an abstract shape of rule. A *linear ground δ-rule*[5] is a rewrite rule of the shape

$$\delta \mathtt{m}_0 ... \mathtt{m}_n \rightarrow \mathtt{P}$$

where δ is a constant of the language, $\mathtt{m}_i$ is either a numeral or a variable $\mathtt{x}_i$ and $\mathtt{P}$ is a term where $\mathtt{m}_0, ..., \mathtt{m}_n$ can occur. The variables must be pairwise distinct, hence linear ground δ-rules are "driven" by the simple observation of some numerals.
If $\mathbf{s}$ is a substitution then the term $\mathbf{s}(\delta \mathtt{m}_0 ... \mathtt{m}_n)$ can be reduced to $\mathbf{s}(\mathtt{P})$ by the corresponding δ-rule.

Example 3.5 [6] *Let* $\mathtt{por}$ *be a δ-constant with type $\iota \rightarrowtail \iota \rightarrowtail \iota$. Its operational behaviour may be formalized with the following linear ground δ-rules:*

$$\mathtt{por}\ \tilde{\mathtt{0}}\ \mathtt{x} \rightarrow \tilde{\mathtt{0}} \qquad \mathtt{por}\ \mathtt{x}\ \tilde{\mathtt{0}} \rightarrow \tilde{\mathtt{0}} \qquad \mathtt{por}\ \widetilde{\mathtt{n}+\mathtt{1}}\ \widetilde{\mathtt{m}+\mathtt{1}} \rightarrow \tilde{\mathtt{1}}$$

Clearly $\mathtt{por}$ *corresponds to a parallel-or operator.*

A *$\mathcal{PCF}$-like rewrite system*[5] is a language $\mathcal{L}$ together with a set Θ of linear ground δ-rules on the constants of $\mathcal{L}$. The crucial statement[5] of Jim and Meyer is:

> "every stable model (interpreted in standard way) that is preorder-adequate for a conservative extension $\mathcal{L}$ of $\mathcal{PCF}$ obtained by a $\mathcal{PCF}$-like rewrite system is not fully abstract for $\mathcal{L}$".

Actually, the proof of the previous statement is developed by reasoning on the true-separator function (corresponding to a boolean version of $\mathtt{strict?}$) which is not extensionally-monotone. They show that $\mathcal{PCF}$-like rewrite systems can only describe extensionally-monotone operators. Hence, they conclude that a $\mathcal{PCF}$-like rewrite system cannot describe the operational behaviour of languages analogous to $\mathcal{StPCF}$.

---

5 [37] Definitions 2.4, 2.8 (page 667), Definition 4.1 (page 672), Theorem 5.5 (page 676).
6 Other examples can be found in [37], in particular the rewrite rules for (a full version of) $\mathcal{PCF}$ are presented in Figure 2 (page 678).

It is an easy exercise to give an operational description of the $\mathcal{S}t\mathcal{P}CF$ by a small-step operational rules less restrictive than the linear ground operational rules. A careful treatment of contextual closures for the reduction rules must be given.

In the literature, many extensions of $\mathcal{P}CF$ in which strict? can be defined have been proposed; in particular $S\mathcal{P}CF$ [8], $\mathcal{P}CF$ extended with the Longley's H operator [11] (denoted as $\mathcal{P}CF$+H in the follows) and $\mu\mathcal{P}CF$ [42,21]. All these languages are related to the study of concrete data structures [5,48], strongly stable functions [6] and sequentially realizable functionals [11]. It is possible to write a program simulating strict? in the language $S\mathcal{P}CF$ by using the catch operator. First of all, a variant of catch is presented informally. Without loss of generality, assume

$$\frac{B[\mathtt{x}_1 : \iota, ..., \mathtt{x}_k : \iota] \vdash \mathtt{M} : \iota}{B \vdash \mathtt{catch}\,\mathtt{x}_1...\mathtt{x}_k\ \mathtt{in}\,\mathtt{M} : \iota}$$

be the typing rule of catch and note that FV(catch $\mathtt{x}_1...\mathtt{x}_k$ in M) = FV(M) − $\{\mathtt{x}_1...\mathtt{x}_k\}$, i.e. catch is a binder. The evaluation of catch $\mathtt{x}_1...\mathtt{x}_k$ in M asks the evaluation of M, if the computation of M asks the evaluation of the variable $\mathtt{x}_i$ then the computation of catch $\mathtt{x}_1...\mathtt{x}_k$ in M terminates, returning $\widetilde{\mathtt{i}-\mathtt{1}}$. Otherwise, if the computation of M terminates on a numeral $\tilde{\mathtt{n}}$ without using any of the $\mathtt{x}_i$, then catch $\mathtt{x}_1...\mathtt{x}_k$ in M returns $\widetilde{\mathtt{n}+\mathtt{k}}$.
Note that the evaluation of catch $\mathtt{x}^\iota$ in (if $\mathtt{x}\,\Omega_\iota\,\Omega_\iota$) returns $\tilde{\mathtt{0}}$, while the evaluation of catch $\mathtt{x}^\iota$ in Ω_ι diverges. But strict?($\lambda\mathtt{x}^\iota$.if $\mathtt{x}\,\Omega_\iota\,\Omega_\iota$) diverges and it is easy to understand that catch cannot be defined by strict?. On the other hand, if(if(M$\tilde{0}$)(catch x in Mx)(catch x in Mx))$\tilde{0}\tilde{1}$ is a term with the same behaviour of strict?(M).

Moreover, $\mathcal{S}t\mathcal{P}CF$ can be studied from a true functional programming point of view. In this perspective to compare strict? with the Longley's H operator [11] appears to be interesting. Reasonably the "computational cost" of strict? is lower than that of H (see [49]), but strict? is again sufficient in order to express many meaningful applications that cannot be expressed in $\mathcal{P}CF$. Longley noted that in $\mathcal{P}CF$+H interesting applications like the *modulus* [43] can be programmed, as an example a similar application will be implemented here.
Let F be a term of type $(\iota \rightarrowtail \iota) \rightarrowtail \iota$ and g a term of type $\iota \rightarrowtail \iota$ such that Fg $\Downarrow_e$. Informally, in the course of the evaluation of Fg, the term F can learn informations about g by applying it to various arguments. When the computation of Fg finishes (i.e. a result is returned), F has learnt finite informations about g. Such finite information can be expressed by a term g′ corresponding to the minimum *restriction* of g such that Fg′ $\Downarrow_e$.
If Fg $\Downarrow_e$ then $\mathtt{T}_0 \equiv \lambda\mathtt{F}^{(\iota\rightarrowtail\iota)\rightarrowtail\iota}\mathtt{g}^{\iota\rightarrowtail\iota}$.if strict?($\lambda\mathtt{y}^\iota$.F($\lambda\mathtt{z}^\iota$.if y g(z) Ω_ι)) $\tilde{1}$ $\tilde{0}$ returns $\tilde{0}$ in case F is constant and returns $\tilde{1}$ otherwise. Let $\doteq$ be defined as in Page 25. If Fg $\Downarrow_e$ then $\mathtt{T}_1 \equiv \lambda\mathtt{F}^{(\iota\rightarrowtail\iota)\rightarrowtail\iota}\mathtt{g}^{\iota\rightarrowtail\iota}\mathtt{x}^\iota$.strict?($\lambda\mathtt{y}^\iota$.F($\lambda\mathtt{z}^\iota$.g(if(x $\doteq$ z)(if yzΩ_ι)z))) returns $\tilde{1}$ when either g is constant or the behaviour of g on x is not observed from F, otherwise $\tilde{0}$ is returned. Thus

$$\lambda\mathtt{F}^{(\iota\rightarrowtail\iota)\rightarrowtail\iota}\mathtt{g}^{\iota\rightarrowtail\iota}\mathtt{x}^\iota.\mathtt{if}\ (\mathtt{T}_0\,\mathtt{F}\,\mathtt{g})\ \Omega_\iota\ (\mathtt{if}\,(\mathtt{T}_1\,\mathtt{F}\,\mathtt{g}\,\mathtt{x})\ \mathtt{g}(\mathtt{x})\ \mathtt{g}(\Omega_\iota))$$

is an implementation of the restriction in $\mathcal{S}t\mathcal{P}CF$. Note that the restriction respect the stable order (not the extensional one), in the sense that if $\mathtt{g}$ is a constant function then its restriction is $\mathtt{g}$ itself. Clearly the above term is not equivalent to a $\mathcal{P}CF$ one.

4 Technical characterizations of strict?

The following three lemmas give some technical characterizations of the operational behaviour of strict?. They are useful in order to prove Theorem 3.3.

Lemma 4.1 *If* $\mathtt{z} : \iota \vdash \mathtt{M} : \iota$ *and* $\mathtt{M}[\tilde{\mathtt{0}}/\mathtt{z}] \Downarrow_e \tilde{\mathtt{n}}$ *then* $\mathtt{strict?}(\lambda \mathtt{z}^{\iota}.\mathtt{M}) \Downarrow_e \tilde{\mathtt{k}}$ *where* $\tilde{\mathtt{k}} \in \{\tilde{\mathtt{0}}, \tilde{\mathtt{1}}\}$*; moreover, if* $\mathtt{M}[\Omega_{\iota}/\mathtt{z}] \Downarrow_e \tilde{\mathtt{n}}'$ *then* $\tilde{\mathtt{k}} \equiv \tilde{\mathtt{1}}$ *and* $\tilde{\mathtt{n}}' \equiv \tilde{\mathtt{n}}$.

Proof. The proof is given by induction on the derivation proving $\mathtt{M}[\tilde{\mathtt{0}}/\mathtt{z}] \Downarrow_e \tilde{\mathtt{n}}$.

- If the derivation ends with $\dfrac{((\mathtt{P}[\mathtt{Q}/\mathtt{x}])\mathtt{M}_1 \ldots \mathtt{M}_m)[\tilde{\mathtt{0}}/\mathtt{z}] \Downarrow_e \tilde{\mathtt{n}}}{((\lambda \mathtt{x}^{\sigma}.\mathtt{P})\mathtt{Q}\mathtt{M}_1 \ldots \mathtt{M}_m)[\tilde{\mathtt{0}}/\mathtt{z}] \Downarrow_e \tilde{\mathtt{n}}}\ (\textit{head})$
  then $\mathtt{strict?}(\lambda \mathtt{z}^{\iota}.\mathtt{P}[\mathtt{Q}/\mathtt{x}]\mathtt{M}_1 \ldots \mathtt{M}_m) \Downarrow_e \tilde{\mathtt{k}}$ where $\tilde{\mathtt{k}} \in \{\tilde{\mathtt{0}}, \tilde{\mathtt{1}}\}$, by induction.
  Thus $\mathtt{strict?}(\lambda \mathtt{z}^{\iota}.((\lambda \mathtt{x}^{\sigma}.\mathtt{P})\mathtt{Q}\mathtt{M}_1 \ldots \mathtt{M}_m)) \Downarrow_e \tilde{\mathtt{k}}$, by rule ($\lambda$?*head*).
  If $((\lambda \mathtt{x}^{\sigma}.\mathtt{P})\mathtt{Q}\mathtt{M}_1 \ldots \mathtt{M}_m)[\Omega_{\iota}/\mathtt{z}] \Downarrow_e \tilde{\mathtt{n}}'$ then the last applied rule must be (*head*), so $((\mathtt{P}[\mathtt{Q}/\mathtt{x}])\mathtt{M}_1 \ldots \mathtt{M}_m)[\Omega_{\iota}/\mathtt{z}] \Downarrow_e \tilde{\mathtt{n}}'$ too. Hence $\tilde{\mathtt{k}} \equiv \tilde{\mathtt{1}}$ and $\tilde{\mathtt{n}}' \equiv \tilde{\mathtt{n}}$ by induction.
- If (Y) is the last applied rule then the proof is similar to that of the previous case, where rules (λ?Y) and (Y) are used in place of (λ?*head*) and (*head*).
- If the derivation ends with $\dfrac{\mathtt{M}_0[\tilde{\mathtt{0}}/\mathtt{z}] \Downarrow_e \tilde{\mathtt{0}} \quad \mathtt{M}_1[\tilde{\mathtt{0}}/\mathtt{z}] \Downarrow_e \tilde{\mathtt{n}}}{(\mathtt{if}\ \mathtt{M}_0\, \mathtt{M}_1\, \mathtt{M}_2)[\tilde{\mathtt{0}}/\mathtt{z}] \Downarrow_e \tilde{\mathtt{n}}}\ (\mathtt{0if})$ then by induction
  $\mathtt{strict?}(\lambda \mathtt{z}^{\iota}.\mathtt{M}_0) \Downarrow_e \tilde{\mathtt{k}}_0$ and $\mathtt{strict?}(\lambda \mathtt{z}^{\iota}.\mathtt{M}_1) \Downarrow_e \tilde{\mathtt{k}}_1$ where $\tilde{\mathtt{k}}_0, \tilde{\mathtt{k}}_1 \in \{\tilde{\mathtt{0}}, \tilde{\mathtt{1}}\}$.
  Thus $\mathtt{strict?}(\lambda \mathtt{z}^{\iota}.\mathtt{if}\ \mathtt{M}_0\, \mathtt{M}_1\, \mathtt{M}_2) \Downarrow_e \tilde{\mathtt{k}}$ where $\tilde{\mathtt{k}} \in \{\tilde{\mathtt{0}}, \tilde{\mathtt{1}}\}$ by rule (λ?0if).
  If $(\mathtt{if}\ \mathtt{M}_0\, \mathtt{M}_1\, \mathtt{M}_2)[\Omega_{\iota}/\mathtt{z}] \Downarrow_e \tilde{\mathtt{n}}'$ then the last applied rule must be either (0if) or (1if), so $\mathtt{M}_0[\Omega_{\iota}/\mathtt{z}] \Downarrow_e$. Hence $\mathtt{M}_0[\Omega_{\iota}/\mathtt{z}] \Downarrow_e \tilde{\mathtt{0}}$ and $\tilde{\mathtt{k}}_0 \equiv \tilde{\mathtt{1}}$ by induction.
  Thus the last applied rule must be (0if) and $\mathtt{M}_1[\Omega_{\iota}/\mathtt{z}] \Downarrow_e \tilde{\mathtt{n}}'$. Therefore $\tilde{\mathtt{n}}' \equiv \tilde{\mathtt{n}}$ and $\tilde{\mathtt{k}}_1 \equiv \tilde{\mathtt{1}}$ by induction. But $\mathtt{if}\ \tilde{\mathtt{1}}\, \tilde{\mathtt{0}}\, \tilde{\mathtt{1}} \Downarrow_e \tilde{\mathtt{1}}$ implies $\tilde{\mathtt{k}} \equiv \tilde{\mathtt{1}}$.
- If (1if) is the last applied rule then the proof is similar to the previous case.
- If the derivation ends with $\dfrac{\mathtt{M}[\tilde{\mathtt{0}}/\mathtt{z}] \Downarrow_e \widetilde{\mathtt{n}+1}}{\mathtt{pred}\,\mathtt{M}[\tilde{\mathtt{0}}/\mathtt{z}] \Downarrow_e \tilde{\mathtt{n}}}\ (\mathtt{pred})$ then $\mathtt{strict?}(\lambda \mathtt{z}^{\iota}.\mathtt{M}) \Downarrow_e \tilde{\mathtt{k}}$ where $\tilde{\mathtt{k}} \in \{\tilde{\mathtt{0}}, \tilde{\mathtt{1}}\}$, by induction; so $\mathtt{strict?}(\lambda \mathtt{z}^{\iota}.\mathtt{pred}\,\mathtt{M}) \Downarrow_e \tilde{\mathtt{k}}$ by rule (λ?pred).
  If $\mathtt{pred}\,\mathtt{M}[\Omega_{\iota}/\mathtt{z}] \Downarrow_e \tilde{\mathtt{n}}'$ then $\mathtt{M}[\Omega_{\iota}/\mathtt{z}] \Downarrow_e \widetilde{\mathtt{n}'+1}$, since the last applied rule must be (pred). The proof follows by induction.
- If (succ) is the last applied rule then the proof is similar to that of the previous case, by rule (λ?succ).
- Let (*num*) be the last applied rule; since $\mathtt{M}[\tilde{\mathtt{0}}/\mathtt{z}]$ is a numeral then either $\mathtt{M} \equiv \mathtt{z}$ or $\mathtt{M} \equiv \tilde{\mathtt{m}}$, for some numerals $\tilde{\mathtt{m}}$. In the first case $\mathtt{z}[\tilde{\mathtt{0}}/\mathtt{z}] \Downarrow_e$ and $\mathtt{z}[\Omega_{\iota}/\mathtt{z}] \Uparrow_e$, but $\mathtt{strict?}(\lambda \mathtt{z}^{\iota}.\mathtt{z}) \Downarrow_e \tilde{\mathtt{0}}$ by rule (λ?*x*). In the other case $\tilde{\mathtt{m}}[\tilde{\mathtt{0}}/\mathtt{z}] \Downarrow_e \tilde{\mathtt{m}}$ and $\tilde{\mathtt{m}}[\Omega_{\iota}/\mathtt{z}] \Downarrow_e \tilde{\mathtt{m}}$, but $\mathtt{strict?}(\lambda \mathtt{z}^{\iota}.\tilde{\mathtt{m}}) \Downarrow_e \tilde{\mathtt{1}}$ by rule (λ?*num*).

- If (0gor), (1gor) or (2gor) is the last applied rule then the proof is similar to that of (0if), where (λ?0gor), (λ?1gor) and (λ?2gor) are used respectively in place of (λ?0if).
- If (λ?*num*) is the last applied rule then two are three cases: $\mathtt{M} \equiv \mathtt{strict?}\,(\lambda\mathtt{x}^{\iota}.\mathtt{z})$ and $\mathtt{M} \equiv \mathtt{strict?}\,(\lambda\mathtt{x}^{\iota}.\tilde{\mathtt{n}})$. The proofs are:

$$\dfrac{\dfrac{\dfrac{}{\mathtt{strict?}\,(\lambda\mathtt{z}^{\iota}.\mathtt{z}) \Downarrow_e \tilde{\mathtt{0}}}\,(\lambda?num)}{\mathtt{strict?}\,(\lambda\mathtt{z}^{\iota}.(\lambda\mathtt{x}^{\iota}.\mathtt{z})\tilde{\mathtt{0}}) \Downarrow_e \tilde{\mathtt{0}}}\,(\lambda?head)}{\mathtt{strict?}\,(\lambda\mathtt{z}^{\iota}.\mathtt{strict?}(\lambda\mathtt{x}^{\iota}.\mathtt{z})) \Downarrow_e \tilde{\mathtt{0}}}\,(\lambda??) \qquad \dfrac{\dfrac{\dfrac{}{\mathtt{strict?}\,(\lambda\mathtt{z}^{\iota}.\tilde{\mathtt{0}}) \Downarrow_e \tilde{\mathtt{1}}}\,(\lambda?num)}{\mathtt{strict?}\,(\lambda\mathtt{z}^{\iota}.(\lambda\mathtt{x}^{\iota}.\mathtt{x})\tilde{\mathtt{0}}) \Downarrow_e \tilde{\mathtt{1}}}\,(\lambda?head)}{\mathtt{strict?}\,(\lambda\mathtt{z}^{\iota}.\mathtt{strict?}(\lambda\mathtt{x}^{\iota}.\mathtt{x})) \Downarrow_e \tilde{\mathtt{1}}}\,(\lambda??)$$

- If (λ?x) is the last applied rule then the proof is similar to the previous one.
- If the derivation ends with $\dfrac{(\mathtt{strict?}\,((\mathtt{P}[\mathtt{Q}/\mathtt{x}])\mathtt{M}_1...\mathtt{M}_m))[\tilde{\mathtt{0}}/\mathtt{z}] \Downarrow_e \tilde{\mathtt{n}} \quad (m \geq 0)}{(\mathtt{strict?}((\lambda\mathtt{x}^{\sigma}.\mathtt{P})\mathtt{Q}\mathtt{M}_1...\mathtt{M}_m))[\tilde{\mathtt{0}}/\mathtt{z}] \Downarrow_e \tilde{\mathtt{n}}}\,(?head)$

  then $\mathtt{strict?}(\lambda\mathtt{z}^{\iota}.\mathtt{strict?}\,(\mathtt{P}[\mathtt{Q}/\mathtt{x}]\mathtt{M}_1...\mathtt{M}_m)) \Downarrow_e \tilde{\mathtt{k}}$ where $\tilde{\mathtt{k}} \in \{\tilde{\mathtt{0}}, \tilde{\mathtt{1}}\}$, by induction. But the last rule of the derivation proving $\mathtt{strict?}(\lambda\mathtt{z}^{\iota}.\mathtt{strict?}\,(\mathtt{P}[\mathtt{Q}/\mathtt{x}]\mathtt{M}_1...\mathtt{M}_m)) \Downarrow_e \tilde{\mathtt{k}}$ must be (λ??), therefore $\mathtt{strict?}(\lambda\mathtt{z}^{\iota}.\mathtt{P}[\mathtt{Q}/\mathtt{x}]\mathtt{M}_1...\mathtt{M}_m\tilde{\mathtt{0}}) \Downarrow_e \tilde{\mathtt{k}}$.
  Thus $\mathtt{strict?}(\lambda\mathtt{z}^{\iota}.(\lambda\mathtt{x}^{\sigma}.\mathtt{P})\mathtt{Q}\mathtt{M}_1...\mathtt{M}_m\tilde{\mathtt{0}}) \Downarrow_e \tilde{\mathtt{k}}$ by rule (λ?*head*) and, by rule (λ??), $\mathtt{strict?}(\lambda\mathtt{z}^{\iota}.\mathtt{strict?}((\lambda\mathtt{x}^{\sigma}.\mathtt{P})\mathtt{Q}\mathtt{M}_1...\mathtt{M}_m)) \Downarrow_e \tilde{\mathtt{k}}$.
  If $(\mathtt{strict?}((\lambda\mathtt{x}^{\sigma}.\mathtt{P})\mathtt{Q}\mathtt{M}_1...\mathtt{M}_m))[\Omega_{\iota}/\mathtt{z}] \Downarrow_e \tilde{\mathtt{n}}'$ then the last applied rule must be (?*head*), so $(\mathtt{strict?}((\mathtt{P}[\mathtt{Q}/\mathtt{x}])\mathtt{M}_1...\mathtt{M}_m))[\Omega_{\iota}/\mathtt{z}] \Downarrow_e \tilde{\mathtt{n}}'$ too. So $\tilde{\mathtt{k}} \equiv \tilde{\mathtt{1}}$ and $\tilde{\mathtt{n}}' \equiv \tilde{\mathtt{n}}$ by induction.
- If the derivation ends with $\dfrac{(\mathtt{strict?}\,(\lambda\mathtt{x}^{\iota}.(\mathtt{P}[\mathtt{Q}/\mathtt{y}])\mathtt{M}_1...\mathtt{M}_m))[\tilde{\mathtt{0}}/\mathtt{z}] \Downarrow_e \tilde{\mathtt{n}} \quad (m \geq 0)}{(\mathtt{strict?}\,(\lambda\mathtt{x}^{\iota}.(\lambda\mathtt{y}^{\sigma}.\mathtt{P})\mathtt{Q}\mathtt{M}_1...\mathtt{M}_m))[\tilde{\mathtt{0}}/\mathtt{z}] \Downarrow_e \tilde{\mathtt{n}}}\,(\lambda?head)$

  then $\mathtt{strict?}(\lambda\mathtt{z}^{\iota}.\mathtt{strict?}\,(\lambda\mathtt{x}^{\iota}.(\mathtt{P}[\mathtt{Q}/\mathtt{y}])\mathtt{M}_1...\mathtt{M}_m)) \Downarrow_e \tilde{\mathtt{k}}$ where $\tilde{\mathtt{k}} \in \{\tilde{\mathtt{0}}, \tilde{\mathtt{1}}\}$, by induction. The last applied rule proving $\mathtt{strict?}(\lambda\mathtt{z}^{\iota}.\mathtt{strict?}\,(\lambda\mathtt{x}^{\iota}.(\mathtt{P}[\mathtt{Q}/\mathtt{y}])\mathtt{M}_1...\mathtt{M}_m)) \Downarrow_e \tilde{\mathtt{k}}$ must be (λ??), so $\mathtt{strict?}(\lambda\mathtt{z}^{\iota}.\mathbf{(}\lambda\mathtt{x}^{\iota}.(\mathtt{P}[\mathtt{Q}/\mathtt{y}])\mathtt{M}_1...\mathtt{M}_m\mathbf{)}\tilde{\mathtt{0}}) \Downarrow_e \tilde{\mathtt{k}}$ where the last applied rule must be (λ?*head*), thus $\mathtt{strict?}(\lambda\mathtt{z}^{\iota}.\mathbf{(}(\mathtt{P}[\mathtt{Q}/\mathtt{y}])\mathtt{M}_1...\mathtt{M}_m\mathbf{)}[\tilde{\mathtt{0}}/\mathtt{x}]) \Downarrow_e \tilde{\mathtt{k}}$. Now $\mathtt{strict?}(\lambda\mathtt{z}^{\iota}.\mathbf{(}(\lambda\mathtt{y}^{\sigma}.\mathtt{P})\mathtt{Q}\mathtt{M}_1...\mathtt{M}_m\mathbf{)}[\tilde{\mathtt{0}}/\mathtt{x}]) \Downarrow_e \tilde{\mathtt{k}}$ and $\mathtt{strict?}(\lambda\mathtt{z}^{\iota}.\mathbf{(}\lambda\mathtt{x}^{\iota}.(\lambda\mathtt{y}^{\sigma}.\mathtt{P})\mathtt{Q}\mathtt{M}_1...\mathtt{M}_m\mathbf{)}\tilde{\mathtt{0}}) \Downarrow_e \tilde{\mathtt{k}}$, by rule ($\lambda$?*head*). So $\mathtt{strict?}(\lambda\mathtt{z}^{\iota}.\mathtt{strict?}\mathbf{(}\lambda\mathtt{x}^{\iota}.(\lambda\mathtt{y}^{\sigma}.\mathtt{P})\mathtt{Q}\mathtt{M}_1...\mathtt{M}_m\mathbf{)}) \Downarrow_e \tilde{\mathtt{k}}$ by rule (λ??). Moreover, if $\mathtt{strict?}\mathbf{(}\lambda\mathtt{x}^{\iota}.(\lambda\mathtt{y}^{\sigma}.\mathtt{P})\mathtt{Q}\mathtt{M}_1...\mathtt{M}_m\mathbf{)}[\Omega_{\iota}/\mathtt{z}] \Downarrow_e \tilde{\mathtt{n}}'$ then the last applied rule must be (λ?*head*), so $\mathtt{strict?}(\lambda\mathtt{x}^{\iota}.(\mathtt{P}[\mathtt{Q}/\mathtt{y}])\mathtt{M}_1...\mathtt{M}_m))[\Omega_{\iota}/\mathtt{z}] \Downarrow_e \tilde{\mathtt{n}}'$ too. Hence $\tilde{\mathtt{k}} \equiv \tilde{\mathtt{1}}$ and $\tilde{\mathtt{n}}' \equiv \tilde{\mathtt{n}}$ by induction.
- Cases (?Y) and (λ?Y) are respectively similar to (?*head*) and (λ?*head*).
- If the derivation ends with $\dfrac{\mathtt{M}[\tilde{\mathtt{0}}/\mathtt{x}, \tilde{\mathtt{0}}/\mathtt{z}] \Downarrow_e \widetilde{\mathtt{n}+\mathtt{1}} \quad (\mathtt{strict?}\,(\lambda\mathtt{x}^{\iota}.\mathtt{M}))[\tilde{\mathtt{0}}/\mathtt{z}] \Downarrow_e \tilde{\mathtt{n}}}{(\mathtt{strict?}\,(\lambda\mathtt{x}^{\iota}.\mathtt{pred}\,\mathtt{M}))[\tilde{\mathtt{0}}/\mathtt{z}] \Downarrow_e \tilde{\mathtt{n}}}\,(\lambda?\mathtt{pred})$

  then $\mathtt{strict?}\,(\lambda\mathtt{z}^{\iota}.\mathtt{strict?}\,(\lambda\mathtt{x}^{\iota}.\mathtt{M})) \Downarrow_e \tilde{\mathtt{k}}$ where $\tilde{\mathtt{k}} \in \{\tilde{\mathtt{0}}, \tilde{\mathtt{1}}\}$, by induction. But the last rule of the derivation must be (λ??), so $\mathtt{strict?}\,(\lambda\mathtt{z}^{\iota}.(\lambda\mathtt{x}^{\iota}.\mathtt{M})\tilde{\mathtt{0}}) \Downarrow_e \tilde{\mathtt{k}}$, where the last rule of the derivation must be (λ?*head*), so $\mathtt{strict?}\,(\lambda\mathtt{z}^{\iota}.\mathtt{M}[\tilde{\mathtt{0}}/\mathtt{x}]) \Downarrow_e \tilde{\mathtt{k}}$. But by hypothesis $\mathtt{M}[\tilde{\mathtt{0}}/\mathtt{x}, \tilde{\mathtt{0}}/\mathtt{z}] \Downarrow_e \widetilde{\mathtt{n}+\mathtt{1}}$, so $\mathtt{strict?}\,(\lambda\mathtt{z}^{\iota}.\mathtt{pred}\,\mathtt{M}[\tilde{\mathtt{0}}/\mathtt{x}]) \Downarrow_e \tilde{\mathtt{k}}$ by rule (λ?pred). Thus $\mathtt{strict?}\,(\lambda\mathtt{z}^{\iota}.(\lambda\mathtt{x}^{\iota}.\mathtt{pred}\,\mathtt{M})\tilde{\mathtt{0}}) \Downarrow_e \tilde{\mathtt{k}}$ by rule (λ?*head*). The proof follows by rule (λ??).
  If $(\mathtt{strict?}\,(\lambda\mathtt{x}^{\iota}.\mathtt{pred}\,\mathtt{M}))[\Omega_{\iota}/\mathtt{z}] \Downarrow_e \tilde{\mathtt{n}}'$ then the proof is immediate by induction.

- Let (?succ) be the last applied rule. Clearly $\mathtt{strict?\ succ}\,[\Omega_\iota/\mathtt{z}] \Downarrow_e$, so

$$\dfrac{\dfrac{\dfrac{}{\mathtt{strict?}\,(\lambda\mathtt{z}^\iota.\tilde{\mathtt{0}}) \Downarrow_e \tilde{\mathtt{1}}}\,(\lambda?num)}{\mathtt{strict?}\,(\lambda\mathtt{z}^\iota.\mathtt{succ}\,\tilde{\mathtt{0}}) \Downarrow_e \tilde{\mathtt{1}}}\,(\lambda?\mathtt{succ})}{\mathtt{strict?}\,(\lambda\mathtt{z}^\iota.\mathtt{strict?\ succ}) \Downarrow_e \tilde{\mathtt{1}}}\,(\lambda??)$$

- If (λ?succ) is the last applied rule then the proof is easier than that for (λ?pred).
- If the derivation ends with $\dfrac{(\mathtt{strict?}\,(\lambda\mathtt{x}^\iota.(\mathtt{M}\,\tilde{\mathtt{0}})))[\tilde{\mathtt{0}}/\mathtt{z}] \Downarrow_e \tilde{\mathtt{n}}}{(\mathtt{strict?}\,(\lambda\mathtt{x}^\iota.\mathtt{strict?\ M}))[\tilde{\mathtt{0}}/\mathtt{z}] \Downarrow_e \tilde{\mathtt{n}}}\,(\lambda??)$

  then $\mathtt{strict?}\,(\lambda\mathtt{z}^\iota.\mathtt{strict?}\,(\lambda\mathtt{x}^\iota.(\mathtt{M}\,\tilde{\mathtt{0}}))) \Downarrow_e \tilde{\mathtt{k}}$ where $\tilde{\mathtt{k}} \in \{\tilde{\mathtt{0}}, \tilde{\mathtt{1}}\}$, by induction. But the last rule of the derivation must be (λ??), hence $\mathtt{strict?}\,(\lambda\mathtt{z}^\iota.(\lambda\mathtt{x}^\iota.(\mathtt{M}\,\tilde{\mathtt{0}}))\tilde{\mathtt{0}}) \Downarrow_e \tilde{\mathtt{k}}$ where the last rule must be (λ?*head*), thus $\mathtt{strict?}\,(\lambda\mathtt{z}^\iota.(\mathtt{M}\tilde{\mathtt{0}})[\tilde{\mathtt{0}}/\mathtt{x}]) \Downarrow_e \tilde{\mathtt{k}}$. Then $\mathtt{strict?}\,(\lambda\mathtt{z}^\iota.\mathtt{strict?}(\mathtt{M}[\tilde{\mathtt{0}}/\mathtt{x}])) \Downarrow_e \tilde{\mathtt{k}}$ by rule (λ??), so $\mathtt{strict?}\,(\lambda\mathtt{z}^\iota.(\lambda\mathtt{x}^\iota.\mathtt{strict?M})\tilde{\mathtt{0}}) \Downarrow_e \tilde{\mathtt{k}}$ by rule (λ?*head*). So $\mathtt{strict?}\,(\lambda\mathtt{z}^\iota.\mathtt{strict?}(\lambda\mathtt{x}^\iota.\mathtt{strict?M})) \Downarrow_e \tilde{\mathtt{k}}$ by rule (λ??).
  If $(\mathtt{strict?}\,(\lambda\mathtt{x}^\iota.\mathtt{strict?\ M}))[\Omega_\iota/\mathtt{z}] \Downarrow_e \tilde{\mathtt{n}}'$ then the last applied rule must be (λ??), therefore $(\mathtt{strict?}\,(\lambda\mathtt{x}^\iota.(\mathtt{M}\,\tilde{\mathtt{0}})))[\Omega_\iota/\mathtt{z}] \Downarrow_e \tilde{\mathtt{n}}'$ too. Hence $\tilde{\mathtt{k}} \equiv \tilde{\mathtt{1}}$ and $\tilde{\mathtt{n}}' \equiv \tilde{\mathtt{n}}$ by induction, so the proof is immediate.
- If (?0if) is the last applied rule then the proof is easier than that for (?1if).
- If the derivation ends with $\dfrac{\mathtt{M}_0[\tilde{\mathtt{0}}/\mathtt{z}] \Downarrow_e \widetilde{\mathtt{m}+\mathtt{1}}}{(\mathtt{strict?}\,(\mathtt{if\ M}_0\,\mathtt{M}_1))[\tilde{\mathtt{0}}/\mathtt{z}] \Downarrow_e \tilde{\mathtt{n}}}\,(?\mathtt{1if})$ then

  $\mathtt{strict?}(\lambda\mathtt{z}^\iota.\mathtt{M}_0) \Downarrow_e \tilde{\mathtt{k}}$ where $\tilde{\mathtt{k}} \in \{\tilde{\mathtt{0}}, \tilde{\mathtt{1}}\}$, by induction.

$$\dfrac{\dfrac{\dfrac{\vdots}{\mathtt{M}_0[\tilde{\mathtt{0}}/\mathtt{z}] \Downarrow_e \widetilde{\mathtt{m}+\mathtt{1}}}\,(\ldots) \quad \dfrac{\vdots}{\mathtt{strict?}(\lambda\mathtt{z}^\iota.\mathtt{M}_0) \Downarrow_e \tilde{\mathtt{k}}}\,(\ldots) \quad \dfrac{}{\mathtt{strict?}(\lambda\mathtt{z}^\iota.\tilde{\mathtt{0}}) \Downarrow_e \tilde{\mathtt{1}}}\,(\lambda?num)}{\mathtt{strict?}(\lambda\mathtt{z}^\iota.\mathtt{if\ M}_0\,\mathtt{M}_1\tilde{\mathtt{0}}) \Downarrow_e \tilde{\mathtt{k}}\,\mathtt{or}\ \tilde{\mathtt{1}}}\,(\lambda?\mathtt{1if})}{\mathtt{strict?}(\lambda\mathtt{z}^\iota.\mathtt{strict?}\,(\mathtt{if\ M}_0\,\mathtt{M}_1)) \Downarrow_e \tilde{\mathtt{k}}\,\mathtt{or}\ \tilde{\mathtt{1}}}\,(\lambda??)$$

  If $(\mathtt{strict?}(\mathtt{if\ M}_0\,\mathtt{M}_1))[\Omega_\iota/\mathtt{x}] \Downarrow_e \tilde{\mathtt{n}}'$ then the last applied rule must be (?1if), so the proof follows by induction.
- If the derivation ends with

$$\dfrac{\mathtt{M}_0[\tilde{\mathtt{0}}/\mathtt{x},\tilde{\mathtt{0}}/\mathtt{z}] \Downarrow_e \tilde{\mathtt{0}} \quad (\mathtt{strict?}\,(\lambda\mathtt{x}^\iota.\mathtt{M}_0))[\tilde{\mathtt{0}}/\mathtt{z}] \Downarrow_e \tilde{\mathtt{n}}_0 \quad (\mathtt{strict?}\,(\lambda\mathtt{x}^\iota.\mathtt{M}_1))[\tilde{\mathtt{0}}/\mathtt{z}] \Downarrow_e \tilde{\mathtt{n}}_1}{(\mathtt{strict?}\,(\lambda\mathtt{x}^\iota.\mathtt{if\ M}_0\,\mathtt{M}_1\,\mathtt{M}_2))[\tilde{\mathtt{0}}/\mathtt{z}] \Downarrow_e \tilde{\mathtt{n}}_0\,\mathtt{or}\ \tilde{\mathtt{n}}_1}\,(\lambda?\mathtt{0if})$$

  then $\mathtt{strict?}(\lambda\mathtt{z}^\iota.\mathtt{strict?}\,(\lambda\mathtt{x}^\iota.\mathtt{M}_0)) \Downarrow_e \tilde{\mathtt{k}}_0$ and $\mathtt{strict?}(\lambda\mathtt{z}^\iota.\mathtt{strict?}\,(\lambda\mathtt{x}^\iota.\mathtt{M}_1)) \Downarrow_e \tilde{\mathtt{k}}_1$ where $\tilde{\mathtt{k}}_0, \tilde{\mathtt{k}}_1 \in \{\tilde{\mathtt{0}}, \tilde{\mathtt{1}}\}$, by induction. In both those derivations the last applied rule must be (λ??), so $\mathtt{strict?}(\lambda\mathtt{z}^\iota.(\lambda\mathtt{x}^\iota.\mathtt{M}_0)\tilde{\mathtt{0}}) \Downarrow_e \tilde{\mathtt{k}}_0$ and $\mathtt{strict?}(\lambda\mathtt{z}^\iota.(\lambda\mathtt{x}^\iota.\mathtt{M}_1)\tilde{\mathtt{0}}) \Downarrow_e \tilde{\mathtt{k}}_1$ and yet $(\mathtt{strict?}(\lambda\mathtt{z}^\iota.\mathtt{M}_0))[\tilde{\mathtt{0}}/\mathtt{x}] \Downarrow_e \tilde{\mathtt{k}}_0$ and $(\mathtt{strict?}(\lambda\mathtt{z}^\iota.\mathtt{M}_1))[\tilde{\mathtt{0}}/\mathtt{x}] \Downarrow_e \tilde{\mathtt{k}}_1$ by rule (λ?*head*). So $\mathtt{strict?}\,(\lambda\mathtt{z}^\iota.(\mathtt{if\ M}_0\,\mathtt{M}_1\,\mathtt{M}_2)[\tilde{\mathtt{0}}/\mathtt{x}]) \Downarrow_e \tilde{\mathtt{k}}_0\,\mathtt{or}\ \tilde{\mathtt{k}}_1$ by rule (λ?0if) and, moreover, $\mathtt{strict?}\,(\lambda\mathtt{z}^\iota.(\lambda\mathtt{x}^\iota.\mathtt{if\ M}_0\,\mathtt{M}_1\,\mathtt{M}_2)\tilde{\mathtt{0}}) \Downarrow_e \tilde{\mathtt{k}}_0\,\mathtt{or}\ \tilde{\mathtt{k}}_1$ by rule (λ?*head*).
  Therefore $\mathtt{strict?}\,(\lambda\mathtt{z}^\iota.\mathtt{strict?}(\lambda\mathtt{x}^\iota.\mathtt{if\ M}_0\,\mathtt{M}_1\,\mathtt{M}_2)) \Downarrow_e \tilde{\mathtt{k}}_0\,\mathtt{or}\ \tilde{\mathtt{k}}_1$ by rule (λ??).
  If $\mathtt{strict?}((\lambda\mathtt{x}^\iota.\mathtt{if\ M}_0\,\mathtt{M}_1\,\mathtt{M}_2)[\Omega_\iota/\mathtt{z}] \Downarrow_e \tilde{\mathtt{n}}'$ then the proof follows by induction.
- If (λ?1if) is the last applied rule then the proof is similar to that of (λ?0if).
- Cases (?0gor) and (?2gor) are respectively similar to cases (?0if) and (?1if). If (λ?0gor), (λ?1gor) or (λ?2gor) is the last applied rule then the proof is simi-

lar to the case (λ?0if).

□

Lemma 4.2 *If* $\mathtt{M}\tilde{\mathtt{0}} \Downarrow_e \mathtt{n}$ *is a program and* $\mathtt{M}\tilde{\mathtt{0}} \Downarrow_e \mathtt{n}$ *then* $\mathtt{strict?\,M} \Downarrow_e \tilde{\mathtt{k}}$ *where* $\tilde{\mathtt{k}} \in \{\tilde{\mathtt{0}}, \tilde{\mathtt{1}}\}$*; moreover, if* $\mathtt{M}\,\Omega_\iota \Downarrow_e \mathtt{n}'$ *then* $\tilde{\mathtt{k}} \equiv \tilde{\mathtt{1}}$ *and* $\mathtt{n}' \equiv \mathtt{n}$.

Proof. The proof is given by induction on the derivation proving $\mathtt{M}\tilde{\mathtt{0}} \Downarrow_e \mathtt{n}$.
Note that no rule of Figure 3 can conclude the derivation $\mathtt{M}\tilde{\mathtt{0}} \Downarrow_e \mathtt{n}$.

- If the last applied rule is (*head*) then there are two cases.
  - If the derivation ends with $\dfrac{\mathtt{P[Q/x]M_1...M_m}\tilde{\mathtt{0}} \Downarrow_e \mathtt{n} \quad (m \geq 1)}{(\lambda \mathtt{x}^\sigma.\mathtt{P})\mathtt{QM_1...M_m}\tilde{\mathtt{0}} \Downarrow_e \mathtt{n}}\ (\textit{head})$
    then $\mathtt{strict?(P[Q/x]M_1...M_m)} \Downarrow_e \tilde{\mathtt{k}}$ where $\tilde{\mathtt{k}} \in \{\tilde{\mathtt{0}}, \tilde{\mathtt{1}}\}$, by induction.
    Thus $\mathtt{strict?}((\lambda \mathtt{x}^\sigma.\mathtt{P})\mathtt{QM_1...M_m}) \Downarrow_e \tilde{\mathtt{k}}$ by rule (?*head*).
    If $(\lambda \mathtt{x}^\sigma.\mathtt{P})\mathtt{QM_1...M_m}\Omega_\iota \Downarrow_e \mathtt{n}'$ then the proof follows by induction.
  - In case $\dfrac{\mathtt{P}[\tilde{\mathtt{0}}/\mathtt{x}] \Downarrow_e \mathtt{n}}{(\lambda \mathtt{x}^\sigma.\mathtt{P})\tilde{\mathtt{0}} \Downarrow_e \mathtt{n}}\ (\textit{head})$ then the proof follows by Lemma 4.1.
- Let (Y) be the last applied rule. It easy to see that a word $Y_\sigma\tilde{\mathtt{0}}$ cannot be a program, for each type σ. Thus the proof is similar to that of the first subcase of rule (*head*).
- If the derivation ends with (0if) then the proof is easy by rule (?0if). Since $\mathtt{if\ M_0\,M_1\,}\Omega_\iota \Downarrow_e$ the proof is done. If the derivation ends with (1if) then the proof is easy by rule (?1if). Note that $\mathtt{if\ M_0\,M_1\,}\Omega_\iota \Uparrow_e$.
- If (succ) is the last applied rule then the proof is trivial, by rule (?succ). Note that $\mathtt{succ}\,\Omega_\iota \Uparrow_e$. The last applied rule cannot be (pred), since $\mathtt{pred}\,\tilde{\mathtt{0}} \Uparrow_e$. Also the cases (*num*) and (1gor) are not possible.
- If (0gor) or (2gor) is the last applied rule then the proof follows respectively by rule (?0gor), (?2gor).

□

Lemma 4.3 *If* $\mathtt{strict?\,M} \Downarrow_e \tilde{\mathtt{k}}$ *then* $\tilde{\mathtt{k}} \in \{\tilde{\mathtt{0}}, \tilde{\mathtt{1}}\}$ *and there is a numeral* $\mathtt{n}$ *such that* $\mathtt{M}\tilde{\mathtt{0}} \Downarrow_e \mathtt{n}$*; moreover, if* $\tilde{\mathtt{k}} \equiv \tilde{\mathtt{1}}$ *then* $\mathtt{M}\Omega_\iota \Downarrow_e \mathtt{n}$.

Proof. The proof is given by induction on the derivation proving $\mathtt{strict?M} \Downarrow_e \tilde{\mathtt{k}}$.

- The proof is trivial if the derivation is one of the following

$$\frac{}{\mathtt{strict?\ succ} \Downarrow_e \tilde{\mathtt{0}}}\ (\text{?succ}) \qquad \frac{}{\mathtt{strict?}\,(\lambda \mathtt{x}^\iota.\mathtt{n}) \Downarrow_e \tilde{\mathtt{1}}}\ (\lambda\textit{?num}) \qquad \frac{}{\mathtt{strict?}\,(\lambda \mathtt{x}^\iota.\mathtt{x}) \Downarrow_e \tilde{\mathtt{0}}}\ (\lambda\text{?}x)$$

- If the derivation ends with $\dfrac{\mathtt{strict?}\,(\mathtt{P[Q/x]M_1...M_m}) \Downarrow_e \tilde{\mathtt{k}} \quad (m \geq 0)}{\mathtt{strict?}((\lambda \mathtt{x}^\sigma.\mathtt{P})\mathtt{QM_1...M_m}) \Downarrow_e \tilde{\mathtt{k}}}\ (\textit{?head})$
  then $\tilde{\mathtt{k}} \in \{\tilde{\mathtt{0}}, \tilde{\mathtt{1}}\}$ and $\mathtt{P[Q/x]M_1...M_m}\tilde{\mathtt{0}} \Downarrow_e \mathtt{n}$ by induction, so $(\lambda \mathtt{x}^\sigma.\mathtt{P})\mathtt{QM_1...M_m}\tilde{\mathtt{0}} \Downarrow_e \mathtt{n}$ by rule (*head*). If $\tilde{\mathtt{k}} \equiv \tilde{\mathtt{1}}$ then $\mathtt{P[Q/x]M_1...M_m}\Omega_\iota \Downarrow_e \mathtt{n}$ by induction, so the proof is trivial by rule (*head*).
- If (?Y) is the last applied rule then the proof is similar to that of the previous case.
- If the derivation ends with $\dfrac{\mathtt{strict?}\,(\lambda \mathtt{x}^\iota.\mathtt{P[Q/z]M_1...M_m}) \Downarrow_e \tilde{\mathtt{k}}}{\mathtt{strict?}\,(\lambda \mathtt{x}^\iota.(\lambda \mathtt{z}^\sigma.\mathtt{P})\mathtt{QM_1...M_m}) \Downarrow_e \tilde{\mathtt{k}}}\ (\lambda\textit{?head})$

then $\tilde{\mathtt{k}} \in \{\tilde{\mathtt{0}}, \tilde{\mathtt{1}}\}$ and $(\lambda \mathtt{x}^{\iota}.\mathtt{P}[\mathtt{Q}/\mathtt{z}]\mathtt{M}_1 ... \mathtt{M}_m)\tilde{\mathtt{0}} \Downarrow_e \tilde{\mathtt{n}}$ by induction, for some $\tilde{\mathtt{n}}$. But the last applied rule in the derivation proving $(\lambda \mathtt{x}^{\iota}.\mathtt{P}[\mathtt{Q}/\mathtt{z}]\mathtt{M}_1 ... \mathtt{M}_m)\tilde{\mathtt{0}} \Downarrow_e \tilde{\mathtt{n}}$ must be (*head*), thus $(\mathtt{P}[\mathtt{Q}/\mathtt{z}]\mathtt{M}_1 ... \mathtt{M}_m)[\tilde{\mathtt{0}}/\mathtt{x}] \Downarrow_e \tilde{\mathtt{n}}$. Therefore, both $((\lambda \mathtt{z}^{\sigma}.\mathtt{P})\mathtt{Q}\mathtt{M}_1 ... \mathtt{M}_m)[\tilde{\mathtt{0}}/\mathtt{x}] \Downarrow_e$ and $(\lambda \mathtt{x}^{\iota}.(\lambda \mathtt{z}^{\sigma}.\mathtt{P})\mathtt{Q}\mathtt{M}_1 ... \mathtt{M}_m)\tilde{\mathtt{0}} \Downarrow_e$ by rule (*head*).
If $\tilde{\mathtt{k}} \equiv \tilde{\mathtt{1}}$ then $(\lambda \mathtt{x}^{\iota}.\mathtt{P}[\mathtt{Q}/\mathtt{z}]\mathtt{M}_1 ... \mathtt{M}_m)\Omega_{\iota} \Downarrow_e \tilde{\mathtt{n}}$ by induction, but the last applied rule must be (*head*), having as premise $((\mathtt{P}[\mathtt{Q}/\mathtt{z}])\mathtt{M}_1 ... \mathtt{M}_m)[\Omega_{\iota}/\mathtt{x}] \Downarrow_e \tilde{\mathtt{n}}$. The proof follows by applying the rule (*head*) twice.

- If $(\lambda ?\mathtt{Y})$ is the last applied rule then the proof is similar to that of $(\lambda ?\mathit{head})$.
- If the derivation ends with $\dfrac{\mathtt{M}[\tilde{\mathtt{0}}/\mathtt{x}] \Downarrow_e \widetilde{\mathtt{m}+\mathtt{1}} \quad \mathtt{strict?}\,(\lambda \mathtt{x}^{\iota}.\mathtt{M}) \Downarrow_e \tilde{\mathtt{k}}}{\mathtt{strict?}\,(\lambda \mathtt{x}^{\iota}.\mathtt{pred}\,\mathtt{M}) \Downarrow_e \tilde{\mathtt{k}}}\,(\lambda ?\mathtt{pred})$ then

$\tilde{\mathtt{k}} \in \{\tilde{\mathtt{0}}, \tilde{\mathtt{1}}\}$ and $(\lambda \mathtt{x}^{\iota}.\mathtt{M})\tilde{\mathtt{0}} \Downarrow_e \tilde{\mathtt{n}}$ by induction. The last rule applied in the derivation proving $(\lambda \mathtt{x}^{\iota}.\mathtt{M})\tilde{\mathtt{0}} \Downarrow_e \tilde{\mathtt{n}}$ must be (*head*), thus $\mathtt{M}[\tilde{\mathtt{0}}/\mathtt{x}] \Downarrow_e \tilde{\mathtt{n}}$ and clearly $\widetilde{\mathtt{m}+\mathtt{1}} \equiv \tilde{\mathtt{n}}$. The proof follows by rules (pred) and (*head*).
If $\tilde{\mathtt{k}} \equiv \tilde{\mathtt{1}}$ then $(\lambda \mathtt{x}^{\iota}.\mathtt{M})\Omega_{\iota} \Downarrow_e \tilde{\mathtt{n}}$ by induction, but the last applied rule must be (*head*), having as premise $\mathtt{M}[\Omega_{\iota}/\mathtt{x}] \Downarrow_e \tilde{\mathtt{n}}$. The proof follows by Lemma 4.1, reasoning as before.

- If $(\lambda ?\mathtt{succ})$ is the last applied rule then the proof is similar to that of case $(\lambda ?\mathtt{pred})$.
- If the derivation ends with $\dfrac{\mathtt{M}_0 \Downarrow_e \tilde{\mathtt{0}} \quad \mathtt{M}_1 \Downarrow_e \tilde{\mathtt{n}}}{\mathtt{strict?}\,(\mathtt{if}\ \mathtt{M}_0\,\mathtt{M}_1) \Downarrow_e \tilde{\mathtt{1}}}\,(?\mathtt{0if})$ then the proof is

easy, since both $\mathtt{if}\,\mathtt{M}_0\mathtt{M}_1\tilde{\mathtt{0}} \Downarrow_e \tilde{\mathtt{n}}$ and $\mathtt{if}\,\mathtt{M}_0\mathtt{M}_1\Omega_{\iota} \Downarrow_e \tilde{\mathtt{n}}$ by hypothesis and by rule (0if).

- If (?1if) is the last used rule then the proof is easy, since $\tilde{\mathtt{0}} \Downarrow_e \tilde{\mathtt{0}}$ by rule (*num*). Thus $\mathtt{if}\,\mathtt{M}_0\mathtt{M}_1\tilde{\mathtt{0}} \Downarrow_e \tilde{\mathtt{0}}$ by hypothesis and by rule (1if). Note that $\mathtt{if}\,\mathtt{M}_1\mathtt{M}_2\Omega_{\iota} \Uparrow_e$.
- If the derivation ends with

$$\frac{\mathtt{M}_0[\tilde{\mathtt{0}}/\mathtt{x}] \Downarrow_e \tilde{\mathtt{0}} \quad \mathtt{strict?}\,(\lambda \mathtt{x}^{\iota}.\mathtt{M}_0) \Downarrow_e \tilde{\mathtt{k}}_0 \quad \mathtt{strict?}\,(\lambda \mathtt{x}^{\iota}.\mathtt{M}_1) \Downarrow_e \tilde{\mathtt{k}}_1}{\mathtt{strict?}\,(\lambda \mathtt{x}^{\iota}.\mathtt{if}\ \mathtt{M}_0\,\mathtt{M}_1\,\mathtt{M}_2) \Downarrow_e \tilde{\mathtt{k}}_0\ \mathtt{or}\ \tilde{\mathtt{k}}_1}\,(\lambda ?\mathtt{0if})$$

then $(\lambda \mathtt{x}^{\iota}.\mathtt{M}_0)\tilde{\mathtt{0}} \Downarrow_e \tilde{\mathtt{n}}_0$, $(\lambda \mathtt{x}^{\iota}.\mathtt{M}_1)\tilde{\mathtt{0}} \Downarrow_e \tilde{\mathtt{n}}_1$ and $\tilde{\mathtt{k}}_0, \tilde{\mathtt{k}}_1 \in \{\tilde{\mathtt{0}}, \tilde{\mathtt{1}}\}$ by induction.
Since $\mathtt{if}\ \tilde{\mathtt{k}}_0\ \tilde{\mathtt{0}}\ \tilde{\mathtt{k}}_1 \Downarrow_e \tilde{\mathtt{k}}_0\ \mathtt{or}\ \tilde{\mathtt{k}}_1$, it is easy to see that $\tilde{\mathtt{k}}_0\ \mathtt{or}\ \tilde{\mathtt{k}}_1 \in \{\tilde{\mathtt{0}}, \tilde{\mathtt{1}}\}$. But the last rule applied in the derivation proving $(\lambda \mathtt{x}^{\iota}.\mathtt{M}_1)\tilde{\mathtt{0}} \Downarrow_e \tilde{\mathtt{n}}_1$ must be (*head*), having as premise $\mathtt{M}_1[\tilde{\mathtt{0}}/\mathtt{x}] \Downarrow_e \tilde{\mathtt{n}}_1$. Note that $\mathtt{M}_0[\tilde{\mathtt{0}}/\mathtt{x}] \Downarrow_e \tilde{\mathtt{0}}$ by hypothesis; thus $(\mathtt{if}\ \mathtt{M}_0\,\mathtt{M}_1\,\mathtt{M}_2)[\tilde{\mathtt{0}}/\mathtt{x}] \Downarrow_e \tilde{\mathtt{n}}_1$ by rule (0if), so $(\lambda \mathtt{x}^{\iota}.\mathtt{if}\ \mathtt{M}_0\,\mathtt{M}_1\,\mathtt{M}_2)\tilde{\mathtt{0}} \Downarrow_e \tilde{\mathtt{n}}_1$ by rule (*head*).
Moreover, if $\mathtt{if}\ \tilde{\mathtt{k}}_0\ \tilde{\mathtt{0}}\ \tilde{\mathtt{k}}_1 \Downarrow_e \tilde{\mathtt{1}}$ then $\tilde{\mathtt{k}}_0 \equiv \tilde{\mathtt{k}}_1 \equiv \tilde{\mathtt{1}}$; thus, both $(\lambda \mathtt{x}^{\iota}.\mathtt{M}_0)\Omega_{\iota} \Downarrow_e \tilde{\mathtt{n}}_0$ and $(\lambda \mathtt{x}^{\iota}.\mathtt{M}_1)\Omega_{\iota} \Downarrow_e \tilde{\mathtt{n}}_1$. Hence $\mathtt{M}_1[\Omega_{\iota}/\mathtt{x}] \Downarrow_e \tilde{\mathtt{n}}_1$ by rule (*head*). Since $\mathtt{M}_0[\tilde{\mathtt{0}}/\mathtt{x}] \Downarrow_e \tilde{\mathtt{0}}$ by hypothesis, $\mathtt{M}_0[\Omega_{\iota}/\mathtt{x}] \Downarrow_e \tilde{\mathtt{0}}$ by Lemma 4.1. So $(\mathtt{if}\ \mathtt{M}_0\,\mathtt{M}_1\,\mathtt{M}_2)[\Omega_{\iota}/\mathtt{x}] \Downarrow_e \tilde{\mathtt{n}}_1$ by rule (0if), thus $(\lambda \mathtt{x}^{\iota}.\mathtt{if}\ \mathtt{M}_0\,\mathtt{M}_1\,\mathtt{M}_2)\Omega_{\iota} \Downarrow_e \tilde{\mathtt{n}}_1$ by rule (*head*).

- If $(\lambda ?\mathtt{1if})$ is the last applied rule then the proof is similar to that of $(\lambda ?\mathtt{0if})$.
- If the last applied rule is $\dfrac{\mathtt{strict?}\,(\lambda \mathtt{x}^{\iota}.\mathtt{M}\,\tilde{\mathtt{0}}) \Downarrow_e \tilde{\mathtt{n}}}{\mathtt{strict?}\,(\lambda \mathtt{x}^{\iota}.\mathtt{strict?}\,\mathtt{M}) \Downarrow_e \tilde{\mathtt{n}}}\,(\lambda ??)$ then $(\lambda \mathtt{x}^{\iota}.\mathtt{M}\,\tilde{\mathtt{0}})\tilde{\mathtt{0}} \Downarrow_e$

and $\tilde{\mathtt{n}} \in \{\tilde{\mathtt{0}}, \tilde{\mathtt{1}}\}$ by induction. Thus $\mathtt{M}[\tilde{\mathtt{0}}/\mathtt{x}]\tilde{\mathtt{0}} \Downarrow_e$ by rule (*head*), so by Lemma 4.1 $\mathtt{strict?}\mathtt{M}[\tilde{\mathtt{0}}/\mathtt{x}] \Downarrow_e \tilde{\mathtt{k}}$ where $\tilde{\mathtt{k}} \in \{\tilde{\mathtt{0}}, \tilde{\mathtt{1}}\}$; hence $(\lambda \mathtt{x}^{\iota}.\mathtt{strict?}\,\mathtt{M})\tilde{\mathtt{0}} \Downarrow_e$ by rule (*head*).

If $\tilde{\mathtt{k}} \equiv \tilde{1}$ then $(\lambda \mathtt{x}^{\iota}.\mathtt{M}\,\tilde{\mathtt{0}})\Omega_{\iota} \Downarrow_e$ by induction, but the last applied rule must be (*head*), having has premise $\mathtt{M}[\Omega_{\iota}/\mathtt{x}]\,\tilde{\mathtt{0}} \Downarrow_e \tilde{\mathtt{n}}$. The proof follows by Lemma 4.1, reasoning as before.

- If (?0gor) or (?2gor) is the last applied rule then the proof is similar to that of case (?0if). If (λ?0gor), (λ?1gor) or (λ?2gor) is the last applied rule then the proof is similar to that of case (λ?0if).

□

5 Coherence Spaces

Coherence spaces are a simple framework for Berry's stable functions [4,33], developed by Girard [50]; in this Section their basic definitions and properties are stated. Proof details can be found in [36].

First, some basic definitions are given. If X is a finite set then $\|X\|$ is the number of elements of X. A *partial order* or *poset* is a pair $(D, \sqsubseteq)$ where D is a set and $\sqsubseteq$ is an order relation, often noted simply as D. An element of D is *bottom* and denoted $\bot$ if and only if $\bot \sqsubseteq d$ for each $d \in D$. A partial order D is *flat* when, for all $x, y \in D$, if $x \sqsubseteq z$ then $x = \bot$ or $x = y$. A nonempty subset X of D is *directed* if $\forall x, x' \in X$ $\exists x'' \in X$ such that $x \sqsubseteq x''$ and $x' \sqsubseteq x''$, namely for each pair of elements of X there is an upper bound in X. A *cpo* is a poset D with bottom $\bot \in D$ such that if $X \subseteq D$ is directed then there is $\sqcup X \in D$ which is the least upper bound of X. Let A, B be cpos; a function $f : A \to B$ is *monotone* if and only if $\forall x, x' \in A$ if $x \sqsubseteq_A x'$ then $f(x) \sqsubseteq_B f(x')$.

Definition 5.1 *A* coherence space *X is a pair $(|X|, \frown_X)$ where $|X|$ is a set called the* web*, its elements are called* tokens *and $\frown_X$ is called* coherence relation *on X.*
$\frown_X$ is a binary reflexive and symmetric relation between tokens. The set of cliques *of X is $\mathsf{Cl}(X) = \{x \subseteq |X| \;/\; \forall a, b \in x \quad a \frown_X b\}$; moreover, $\mathsf{Cl}_{fin}(X)$ denotes the set of finite cliques of $\mathsf{Cl}(X)$.*
The strict incoherence $\smile_X$ is the complementary relation of $\frown_X$; the incoherence $\asymp_X$ is the union of relations $\smile_X$ and =; the strict coherence $\frown_X$ is the complementary relation of $\asymp_X$.

If X is a coherence space then $\mathsf{Cl}(X)$ is a poset with respect to the relation $\subseteq$.

Lemma 5.2 *Let X be a coherence space.*

(i) $\emptyset \in \mathsf{Cl}(X)$.
(ii) $\{a\} \in \mathsf{Cl}(X)$, *for each* $a \in |X|$.
(iii) *If* $y \subseteq x$ *and* $x \in \mathsf{Cl}(X)$ *then* $y \in \mathsf{Cl}(X)$.
(iv) *If* $D \subseteq \mathsf{Cl}(X)$ *is directed then* $\cup D \in \mathsf{Cl}(X)$.

Hence, cliques of a coherence space with set-inclusion form a cpo.
Let x, x' be sets; $x \subseteq_{fin} x'$ means that $x \subseteq x'$ and x is finite.

Definition 5.3 *Let X and Y be coherence spaces and $f : \mathsf{Cl}(X) \longrightarrow \mathsf{Cl}(Y)$ be a monotone function.*

- *f is* continuous *whenever $\forall x \in \mathsf{Cl}(X)\ \forall b \in f(x)\ \exists x_0 \subseteq_{fin} x$ such that $b \in f(x_0)$.*
- *f is* stable *whenever $\forall x \in \mathsf{Cl}(X)\ \forall b \in f(x)\ \exists x_0 \subseteq_{fin} x$ such that $b \in f(x_0)$ and $\forall x' \subseteq x$, if $b \in f(x')$ then $x_0 \subseteq x'$.*

Continuity asks for the existence of a finite amount of input for which some amount of output is produced, while stability asks for a minimum finite amount input for which some amount of output is produced. Equivalent formulations of continuity and stability are formalized in the following Lemmas.

Lemma 5.4 (i) *Let X and Y be coherence spaces and $f : \mathsf{Cl}(X) \longrightarrow \mathsf{Cl}(Y)$ be a monotone function. Then f is continuous if and only if $f(\cup D) = \cup\{f(x)/x \in D\}$, for each $D \subseteq \mathsf{Cl}(X)$ directed.*

(ii) *Let X and Y be coherence spaces and $f : \mathsf{Cl}(X) \longrightarrow \mathsf{Cl}(Y)$ be a continuous function. Then f is stable if and only if $\forall x, x' \in \mathsf{Cl}(X)$, $x \cup x' \in \mathsf{Cl}(X)$ implies $f(x \cap x') = f(x) \cap f(x')$.*

Stable functions can be represented as cliques.

Definition 5.5 *Let X and Y be coherence spaces.*
The trace $\mathsf{tr}(f)$ *of the stable function $f : \mathsf{Cl}(X) \longrightarrow \mathsf{Cl}(Y)$ is the set of pairs $(x_0, b) \in \mathsf{Cl}_{fin}(X) \times |Y|$ such that $b \in f(x_0)$ and $\forall x \subseteq x_0$, $b \in f(x)$ implies $x = x_0$.*

Stable functions can be represented as cliques of a coherence space.

Definition 5.6 *Let X and Y be coherence spaces.*
$X \Rightarrow Y$ is the coherence space having $|X \Rightarrow Y| = \mathsf{Cl}_{fin}(X) \times |Y|$ as web, while if $(x_0, b_0), (x_1, b_1) \in |X \Rightarrow Y|$, then $(x_0, b_0) \supset_{X \Rightarrow Y} (x_1, b_1)$ under the following conditions:

(i) *$x_0 \cup x_1 \in \mathsf{Cl}(X)$ implies $b_0 \supset_Y b_1$;*
(ii) *$x_0 \cup x_1 \in \mathsf{Cl}(X)$ and $b_0 = b_1$ imply $x_0 = x_1$.*

The bridge between stable functions and cliques follows.

Lemma 5.7 *If $f : \mathsf{Cl}(X) \longrightarrow \mathsf{Cl}(Y)$ is a stable function then $\mathsf{tr}(f) \in \mathsf{Cl}(X \Rightarrow Y)$.*

Let X, Y be coherence spaces and $t \in \mathsf{Cl}(X \Rightarrow Y)$ and $x \in \mathsf{Cl}(X)$. Let us define $\mathcal{F}(t) : \mathsf{Cl}(X) \longrightarrow \mathsf{Cl}(Y)$ be the function such that

$$\mathcal{F}(t)(x) = \{b \in |Y| \ / \ \exists x_0 \in \mathsf{Cl}(X) \quad (x_0, b) \in t \wedge x_0 \subseteq x\}.$$

Lemma 5.8 *If $t \in \mathsf{Cl}(X \Rightarrow Y)$ then $\mathcal{F}(t) : \mathsf{Cl}(X) \to \mathsf{Cl}(Y)$ is a stable function.*

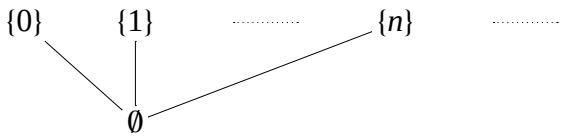
Coherence spaces and stable functions form a cartesian closed category which is a full subcategory of the categories of qualitative domains and dI-domains endowed with stable functions. All these categories contain objects and morphisms in the

range of the standard interpretation of $\mathcal{PCF}$, so without ambiguity they will be called stable models.

6 Interpretation

An interpretation of $\mathcal{PCF}$ is *standard* when ground types are interpreted on flat partial orders. Plotkin in [15] has shown how it is possible to interpret the $\mathcal{PCF}$ syntax on Scott's domains [51] by a set-theoretical standard interpretation. Although the same constraints can be formalized in a cleaner categorical style, for sake of simplicity, a set-theoretical interpretation is provided, since the proofs are developed by reasoning on cliques. Types will be mapped to coherence spaces and terms to cliques.

Definition 6.1 *Let* $\mathbf{N}$ *denote the space of natural numbers, namely* $(|\mathbf{N}|, \frown_{\mathbf{N}})$ *such that* $|\mathbf{N}| = \mathbb{N}$ *and* $m \frown_{\mathbf{N}} n$ *if and only if* $m = n$, *for all* $m, n \in |\mathbf{N}|$.
Thus $\mathsf{Cl}(\mathbf{N}) = \{\emptyset\} \cup \{\ \{n\} \mid n \in |\mathbf{N}|\ \}$ *is the following poset*



Note that $\mathsf{Cl}(\mathbf{N})$ endowed with the set theoretical inclusion forms a flat partial order. Emphatic brackets will be used as notation in order to formalize both the correspondence between types and coherence spaces and the correspondence between terms and cliques, in particular $[\![\iota]\!] = \mathbf{N}$ and $[\![\sigma \rightarrowtail \tau]\!] = [\![\sigma]\!] \Rightarrow [\![\tau]\!]$. If σ is the type of a $\mathcal{StPCF}$ program then $\sigma = \tau_1 \rightarrowtail ..... \rightarrowtail \tau_m \rightarrowtail \iota$, for some $m \geq 0$; if $[\![\sigma]\!]$ is the corresponding coherence space, in what follows for sake of simplicity its tokens will be wrote as $(x_1; ...; x_m; b)$ where $x_i \in \mathsf{Cl}_{fin}([\![\tau_i]\!])$, for all $i \leq m$, and $b \in |\mathbf{N}|$.

Lemma 6.2 *Let* $E = X_1 \Rightarrow ... \Rightarrow X_m \Rightarrow \mathbf{N}$ *be a coherence space* ($m \geq 1$) *and let* $(x_1; ...; x_m; b_x), (y_1; ...; y_m; b_y)$ *be distinct tokens of* $|E|$.
$(x_1; ...; x_m; b_x) \frown_E (y_1; ...; y_m; b_y)$ *if and only if* $\exists k \leq m$ *such that* $x_k \cup y_k \notin \mathsf{Cl}(X_k)$.

Proof. Both directions are proved by induction on m.

(⇐) If $m = 1$ then $x_1 \cup y_1 \notin \mathsf{Cl}(X_1)$, by hypotheses. Thus the proof is immediate, by coherence conditions. If $m \geq 2$ then there are two cases. If $x_1 \cup y_1 \notin \mathsf{Cl}(X_k)$ then again the proof is immediate. Otherwise, $x_1 \cup y_1 \in \mathsf{Cl}(X_1)$ implies $(x_2; ...; x_m; b_x) \neq (y_2; ...; y_m; b_y)$, since $\exists k \leq m$ such that $x_k \cup y_k \notin \mathsf{Cl}(X_k)$ by hypothesis. So $(x_2; ...; x_m; b_x) \frown_E (y_2; ...; y_m; b_y)$ by induction, and the proof follows by coherence conditions.

(⇒) Let $m = 1$ and $(x_1, b_x) \frown_E (y_1, b_y)$. There are two cases, since $\mathsf{Cl}(\mathbf{N})$ is flat. The case $b_x = b_y$ implies $x_1 \neq y_1$, since $(x_1, b_x) \neq (y_1, b_y)$ by hypothesis; therefore $x_1 \cup y_1 \notin \mathsf{Cl}(X_1)$, by Definition 5.6.(ii). In the second case $b_x \smile b_y$, therefore

$x_1 \cup y_1 \notin \mathsf{Cl}(X_1)$ by Definition 5.6.(i).
Let $(x_1; ...; x_m; b_x) \frown_E (y_1; ...; y_m; b_y)$. If $x_1 \cup y_1 \notin \mathsf{Cl}(X_1)$ then the proof is trivial. If $x_1 \cup y_1 \in \mathsf{Cl}(X_1)$ then $(x_2; ...; x_m; b_x) \supset_E (y_2; ...; y_m; b_y)$ by coherence conditions, thus there are two cases.
- $(x_2; ...; x_m; b_x) = (y_2; ...; y_m; b_y)$ would imply $x_1 = y_1$ by coherence conditions, and therefore $(x_1; ...; x_m; b_x) = (y_1; ...; y_m; b_y)$ against the hypothesis.
- The case $(x_2; ...; x_m; b_x) \frown_E (y_2; ...; y_m; b_y)$ follows by induction. □

The corollary below follows immediately.

Corollary 6.3 *Let $E = X_1 \Rightarrow ... \Rightarrow X_m \Rightarrow \mathbf{N}$ be a coherence space ($m \geq 0$).*
If $(x_1; ...; x_m; b_x), (y_1; ...; y_m; b_y) \in |E|$ then

$(x_1, ...x_m, b_x) \asymp_E (y_1, ...y_m, b_y)$ *if and only if* $\forall k \leq m \;\; x_k \cup y_k \in \mathsf{Cl}(X_k)$.

In order to give an interpretation to a $\mathcal{S}t\mathcal{P}CF$-term $\mathtt{M}$ we need to know its typing, therefore the interpretation will implicitly interpret typings rather than terms.
Let B be a basis; Env_B will denote the set of functions ρ such that, if $B(\mathtt{x}) = \sigma$ then $\rho(\mathtt{x})$ is a clique of $[\![\sigma]\!]$. Moreover, if $\rho \in \mathsf{Env}_B$, $B(\mathtt{x}) = \sigma$ and $x \in [\![\sigma]\!]$ then $\rho[x/\mathtt{x}] \in \mathsf{Env}_B$ is the *environment* such that, if $\mathtt{x} \equiv \mathtt{y}$ then $\rho[x/\mathtt{x}](\mathtt{y}) = x$, otherwise $\rho[x/\mathtt{x}](\mathtt{y}) = \rho(\mathtt{y})$. The interpretation mapping is presented in Figure 4. Please note that, sometimes some parts of a formula will be underlined in order to make it more readable (as in the interpretation of strict? in Figure 4).

The interpretation of $\mathtt{Y}_\tau$ is well defined (see [3] for instance) and $\mathcal{F}^n(x) \subseteq \mathcal{F}^{n+1}(x)$.
Let $E = X_1 \Rightarrow ... \Rightarrow X_{m+1}$ be a coherence space ($m \geq 0$) and let $t \in \mathsf{Cl}(E)$.
$\mathcal{F}_*(t) : \mathsf{Cl}(X_1) \longrightarrow ... \longrightarrow \mathsf{Cl}(X_{m+1})$ is the function such that $\forall x_i \in \mathsf{Cl}(X_i)$,

$$\mathcal{F}_*(t)x_1.....x_m = \{b \in |X_{m+1}| \mid \exists (y_1; ...y_m; b) \in t \text{ such that } \forall i \leq m, \;\; y_i \subseteq x_i\}.$$

Lemma 6.4 *Let $\mathtt{M}_0...\mathtt{M}_m$ be a term where $m \geq 1$.*
Thus $[\![\mathtt{M}_0...\mathtt{M}_m]\!]_\rho = \mathcal{F}_*([\![\mathtt{M}_0]\!]_\rho)[\![\mathtt{M}_1]\!]_\rho...[\![\mathtt{M}_m]\!]_\rho$.

Proof. The proof is easy, by induction on m. □

Clearly $\mathcal{F}_*$ extends the $\mathcal{F}$ used in Theorem 5.8.

Lemma 6.5 *The interpretation of* strict? *is actually a clique.*

Proof. Let $(\{(x_0, b_0)\}, c_0), (\{(x_1, b_1)\}, c_1) \in [\![\mathtt{strict?}]\!]$.
We will prove that $(\{(x_0, b_0)\}, c_0) \supset_{(\mathbf{N}\to\mathbf{N})\to\mathbf{N}} (\{(x_1, b_1)\}, c_1)$. Always $x_0 \cup x_1 \in \mathsf{Cl}_{fin}(\mathbf{N})$, thus the proof is immediate by Corollary 6.3 and Lemma 6.2. □

Theorem 6.6 and Theorem 6.7 formalize our intuition on the denotational meaning of the terms $\mathtt{Y}^k_\sigma$ (defined at the end of the Section 2) and of strict?. They will be useful in order to decrease the complexity of the proof of Lemma 7.3.

Theorem 6.6 *Let $B \vdash \mathtt{M} : \iota$ and $\rho \in Env_B$.*

Let $B \vdash \mathtt{M} : \sigma$ be a typing of $\mathcal{S}t\mathcal{P}CF$ and let $\rho \in \mathsf{Env}_B$.
The interpretation of $\mathtt{M}$ with respect to ρ is denoted $[\![\mathtt{M}]\!]_\rho$ (†) and it is a clique of $[\![\sigma]\!]$ obtained by induction on $\mathtt{M}$ in the following way:

- $[\![\mathtt{x}]\!]_\rho = \rho(\mathtt{x})$
- Let $\sigma = \mu \rightarrowtail \tau$ for some types μ, τ, thus
$$[\![\lambda \mathtt{x}^{\mu}.\mathtt{P}]\!]_\rho = \left\{ (x_0, b) \in \mathsf{Cl}([\![\mu]\!]) \times |[\![\tau]\!]| \;\middle|\; \begin{array}{l} b \in [\![\mathtt{P}]\!]_{\rho[x_0/\mathtt{x}]} \text{ and} \\ \forall y \subseteq x_0 \quad b \in [\![\mathtt{P}]\!]_{\rho[y/\mathtt{x}]} \text{ implies } y = x_0 \end{array} \right\}$$
- $[\![\mathtt{PQ}]\!]_\rho = \mathcal{F}([\![\mathtt{P}]\!]_\rho)[\![\mathtt{Q}]\!]_\rho$
- Let $\sigma = (\tau \rightarrowtail \tau) \rightarrowtail \tau$ and $x \in \mathsf{Cl}([\![\tau \rightarrowtail \tau]\!])$, thus
$$[\![\mathtt{Y}_\tau]\!](x) = \bigcup_{n \leq 0} \mathcal{F}^n(x) \text{ where } \mathcal{F}^n(x) = \begin{cases} \emptyset & \text{if } n = 0 \\ \mathcal{F}(x)(\mathcal{F}^{n-1}(x)) & \text{otherwise} \end{cases}$$
- $[\![\mathtt{if}]\!] = \{ (\{0\}; \{n\}; \emptyset; n) \mid n \in \mathbb{N} \} \cup \{ (\{m\}; \emptyset; \{n\}; n) \mid n \in \mathbb{N} \text{ and } m \neq 0 \}$.
- $[\![\tilde{\mathtt{n}}]\!] = \{n\}$, for each $n \in \mathbb{N}$.
- $$\begin{aligned} [\![\mathtt{gor}]\!] = & \left\{ \mathbf{(} \quad \{0\}; \{n+1\}; \quad \emptyset; \; 0 \; \mathbf{)} \;\middle|\; n \in \mathbb{N} \right\} \\ & \bigcup \left\{ \mathbf{(} \quad \emptyset; \quad \{0\}; \{n+1\}; \; 1 \; \mathbf{)} \;\middle|\; n \in \mathbb{N} \right\} \\ & \bigcup \left\{ \mathbf{(} \; \{n+1\}; \quad \emptyset; \quad \{0\}; \; 2 \; \mathbf{)} \;\middle|\; n \in \mathbb{N} \right\} \end{aligned}$$
- $[\![\mathtt{strict?}]\!] = \left\{ (\{\underline{(\{0\}, n)}\} \; ; \; 0) \;\middle|\; n \in |\mathbf{N}| \right\} \bigcup \left\{ (\{\underline{(\emptyset, n)}\} \; ; \; 1) \;\middle|\; n \in |\mathbf{N}| \right\}$.

† It would be clear that the interpretation of closed terms as constants is invariant with respect to environments, thus in such cases the environment indexing the interpretation mapping can be omitted.

Figure 4. Interpretation of $\mathcal{S}t\mathcal{P}CF$

$$\mathcal{F}([\![\mathtt{strict?}]\!])\,[\![\lambda \mathtt{x}^{\iota}.\mathtt{M}]\!]_\rho = \begin{cases} \{0\} & \textit{if } [\![\mathtt{M}]\!]_{\rho[\{0\}/\mathtt{x}]} \neq \emptyset \textit{ and } [\![\mathtt{M}]\!]_{\rho[\emptyset/\mathtt{x}]} = \emptyset\,, \\ \{1\} & \textit{if } [\![\mathtt{M}]\!]_{\rho[\emptyset/\mathtt{x}]} \neq \emptyset \textit{ (hence, } [\![\mathtt{M}]\!]_{\rho[\{0\}/\mathtt{x}]} \neq \emptyset)\,, \\ \emptyset & \textit{otherwise.} \end{cases}$$

Theorem 6.7 (i) $[\![\mathtt{Y}_\sigma^{(n)}]\!](x) = \mathcal{F}^n(x)$, *for all* $n \in \mathbb{N}$ *and type* σ.
(ii) $[\![\mathtt{Y}_\sigma]\!](x) = \bigcup_{n \leq 0} [\![\mathtt{Y}_\sigma^{(n)}]\!](x)$, *for all* $n \in \mathbb{N}$ *and type* σ.

The notion of denotational equivalence [16,15] can be formalized. Let $B \vdash \mathtt{M} : \sigma$ and $B \vdash \mathtt{N} : \sigma$. We write $\mathtt{M} \sim_\sigma \mathtt{N}$ if and only if $[\![\mathtt{M}]\!]_\rho = [\![\mathtt{N}]\!]_\rho$, for each $\rho \in \mathsf{Env}_B$.

If $\mathtt{M} \sim_\sigma \mathtt{N}$ implies $\mathtt{M} \approx_\sigma \mathtt{N}$ then the stable models are *correct* for $\mathcal{StPCF}$. If $\mathtt{M} \approx_\sigma \mathtt{N}$ implies $\mathtt{M} \sim_\sigma \mathtt{N}$ then the stable models are *complete* for $\mathcal{StPCF}$. The stable models are *fully-abstract* for $\mathcal{StPCF}$ if and only if it is both correct and complete for $\mathcal{StPCF}$.

Lemma 6.8 *Let $B \vdash \mathtt{M} : \sigma$ and $B \vdash \mathtt{N} : \tau$ and $\rho, \rho' \in Env_B$.*

(i) *If $\rho(\mathtt{x}) \subseteq \rho'(\mathtt{x})$, for all $FV(\mathtt{M})$, then $[\![\mathtt{M}]\!]_\rho \subseteq [\![\mathtt{M}]\!]_{\rho'}$.*
(ii) *If $x : \tau \in B$ then $[\![\mathtt{M}[\mathtt{N}/\mathtt{x}]]\!]_\rho = [\![\mathtt{M}]\!]_{\rho[[\![\mathtt{N}]\!]_\rho/\mathtt{x}]}$.*
(iii) *If $\tau = \sigma$, $[\![\mathtt{M}]\!]_\rho = [\![\mathtt{N}]\!]_\rho$ and $C[^\sigma]$ is a B^σ-context such that $FV(C[\mathtt{M}]) = FV(C[\mathtt{N}]) = \emptyset$ then $[\![C[\mathtt{M}]]\!] = [\![C[\mathtt{N}]]\!]$.*

It is easy to check that $\mathcal{F}([\![\lambda \mathtt{x}^\sigma.\mathtt{M}]\!]_\rho)(x) = [\![\mathtt{M}]\!]_{\rho[x/\mathtt{x}^\sigma]}$ where $x \in [\![\sigma]\!]$.
The interpretation is sound in the sense of the next Lemma.

Lemma 6.9 *Let $\mathtt{M}$ be a program. If $\mathtt{M} \Downarrow_e \tilde{\mathtt{n}}$ then $[\![\mathtt{M}]\!] = [\![\tilde{\mathtt{n}}]\!]$.*

Proof. The proof is done by induction on the derivation proving $\mathtt{M} \Downarrow_e \tilde{\mathtt{n}}$.

- If the last applied rule is (*head*), ($\mathtt{Y}$), ($0\mathtt{if}$), ($1\mathtt{if}$), ($\mathtt{pred}$), ($\mathtt{succ}$) or (*num*) then the proof is standard.
- If the last applied rule is either ($0\mathtt{gor}$), ($1\mathtt{gor}$) or ($2\mathtt{gor}$) then the proof is easy, by interpretation of $\mathtt{gor}$.
- If the derivation ends with $\dfrac{\mathtt{strict?}\,(\mathtt{P}[\mathtt{Q}/\mathtt{x}]\mathtt{M}_1 \ldots \mathtt{M}_m) \Downarrow_e \tilde{\mathtt{n}} \quad (m \in \mathbb{N})}{\mathtt{strict?}((\lambda \mathtt{x}^\sigma.\mathtt{P})\mathtt{Q}\mathtt{M}_1 \ldots \mathtt{M}_m) \Downarrow_e \tilde{\mathtt{n}}}\,(?head)$ then by induction $[\![\mathtt{strict?}(\mathtt{P}[\mathtt{Q}/\mathtt{x}]\mathtt{M}_1 \ldots \mathtt{M}_m)]\!] = [\![\tilde{\mathtt{n}}]\!]$. Since $[\![(\lambda \mathtt{x}^\sigma.\mathtt{P})\mathtt{Q}]\!]_\rho = [\![\mathtt{P}[\mathtt{Q}/\mathtt{x}]]\!]_\rho$, by Lemma 6.8.(ii) and the interpretation, the proof follows by Lemma 6.8.(iii).
- If the last applied rule is ($\lambda ?head$), ($?\mathtt{Y}$) or ($\lambda ?\mathtt{Y}$) then the proof is similar to that of the rule ($?head$).
- If the derivation ends with $\dfrac{\mathtt{M}_0 \Downarrow_e \tilde{\mathtt{0}} \quad \mathtt{M}_1 \Downarrow_e \tilde{\mathtt{k}}}{\mathtt{strict?}\,(\mathtt{if}\ \mathtt{M}_0\,\mathtt{M}_1) \Downarrow_e \tilde{\mathtt{1}}}\,(?0\mathtt{if})$ then $[\![\mathtt{M}_0]\!] = \{0\}$ and $[\![\mathtt{M}_1]\!] = \{k\}$ by induction. Hence $[\![\mathtt{if}\ \mathtt{M}_0\,\mathtt{M}_1]\!] = \mathcal{F}_*([\![\mathtt{if}\,]\!])[\![\mathtt{M}_0]\!]_\rho[\![\mathtt{M}_1]\!] = \{(\emptyset, k)\}$ and the proof follows by interpretation of $\mathtt{strict?}$.
- If ($?1\mathtt{if}$) is the last applied rule then the proof is similar to that of ($?0\mathtt{if}$).
- If the last applied rule is ($\lambda ?0\mathtt{if}$) or ($\lambda ?\mathtt{pred}$) then the proof is similar to that of ($\lambda ?1\mathtt{if}$).
- Let ρ be an environment. If the derivation ends with

$$\frac{\mathtt{M}_0[\tilde{\mathtt{0}}/\mathtt{x}] \Downarrow_e \widetilde{\mathtt{n}+\mathtt{1}} \quad \mathtt{strict?}\,(\lambda \mathtt{x}^\iota.\mathtt{M}_0) \Downarrow_e \tilde{\mathtt{n}}_0 \quad \mathtt{strict?}\,(\lambda \mathtt{x}^\iota.\mathtt{M}_2) \Downarrow_e \tilde{\mathtt{n}}_2}{\mathtt{strict?}\,(\lambda \mathtt{x}^\iota.\mathtt{if}\ \mathtt{M}_0\,\mathtt{M}_1\,\mathtt{M}_2) \Downarrow_e \tilde{\mathtt{n}}_0\ \mathtt{or}\ \tilde{\mathtt{n}}_2}\,(\lambda ?1\mathtt{if})$$

  then $[\![\mathtt{strict?}\,(\lambda \mathtt{x}^\iota.\mathtt{if}\ \mathtt{M}_0\,\mathtt{M}_1\,\mathtt{M}_2)]\!]_\rho = \mathcal{F}([\![\mathtt{strict?}]\!])[\![\lambda \mathtt{x}^\iota.\mathtt{if}\ \mathtt{M}_0\,\mathtt{M}_1\,\mathtt{M}_2]\!]_\rho = z$.
  Let $\tilde{\mathtt{k}} \equiv \tilde{\mathtt{n}}_0\ \mathtt{or}\ \tilde{\mathtt{n}}_2$, i.e. $\mathtt{if}\ \tilde{\mathtt{n}}_0\,\tilde{\mathtt{0}}\,\tilde{\mathtt{n}}_2 \Downarrow_e \tilde{\mathtt{k}}$; thus there are three cases.
  - If $\tilde{\mathtt{k}} \equiv \tilde{\mathtt{1}}$ then, both $\mathtt{strict?}\,(\lambda \mathtt{x}^\iota.\mathtt{M}_0) \Downarrow_e \tilde{\mathtt{1}}$ and $\mathtt{strict?}\,(\lambda \mathtt{x}^\iota.\mathtt{M}_2) \Downarrow_e \tilde{\mathtt{1}}$; therefore $[\![\mathtt{strict?}\,(\lambda \mathtt{x}^\iota.\mathtt{M}_0)]\!]_\rho = [\![\mathtt{strict?}\,(\lambda \mathtt{x}^\iota.\mathtt{M}_2)]\!]_\rho = \{1\}$ by induction.
    So $[\![\mathtt{M}_0]\!]_{\rho[\emptyset/\mathtt{x}]} \neq \emptyset \neq [\![\mathtt{M}_2]\!]_{\rho[\emptyset/\mathtt{x}]}$, by Theorem 6.6. But by induction $[\![\mathtt{M}_0[\tilde{\mathtt{0}}/\mathtt{x}]]\!]_\rho =$

$\{n+1\}$, thus both $[\![\mathtt{M}_0]\!]_{\rho[\{0\}/\mathtt{x}]} = \{n+1\}$ and $[\![\mathtt{M}_0]\!]_{\rho[\emptyset/\mathtt{x}]} = \{n+1\}$ too.
Therefore $[\![\mathtt{if}\ \mathtt{M}_0\,\mathtt{M}_1\,\mathtt{M}_2]\!]_{\rho[\emptyset/\mathtt{x}]} \neq \emptyset$ and the proof follows by Theorem 6.6.

- If $\tilde{\mathtt{n}}_0 \equiv \tilde{\mathtt{0}}$ then $\mathtt{strict?}\,(\lambda\mathtt{x}^\iota.\mathtt{M}_0) \Downarrow_e \tilde{\mathtt{0}}$, so $[\![\mathtt{M}_0]\!]_{\rho[\emptyset/\mathtt{x}]} = \emptyset$ while $[\![\mathtt{M}_0]\!]_{\rho[\{0\}/\mathtt{x}]} \neq \emptyset$. Thus $[\![\mathtt{if}\ \mathtt{M}_0\,\mathtt{M}_1\,\mathtt{M}_2]\!]_{\rho[\emptyset/\mathtt{x}]} = \emptyset$. By induction $[\![\mathtt{M}_0[\tilde{\mathtt{0}}/\mathtt{x}]]\!]_\rho = \{n+1\}$, so $[\![\mathtt{M}_0]\!]_{\rho[\{0\}/\mathtt{x}]} = \{n+1\}$. On the other hand $\mathtt{strict?}\,(\lambda\mathtt{x}^\iota.\mathtt{M}_2) \Downarrow_e \tilde{\mathtt{n}}_2$ implies $[\![\mathtt{M}_2]\!]_{\rho[\{0\}/\mathtt{x}]} \neq \emptyset$ so $[\![\mathtt{if}\ \mathtt{M}_0\,\mathtt{M}_1\,\mathtt{M}_2]\!]_{\rho[\{0\}/\mathtt{x}]} \neq \emptyset$. The proof follows by Theorem 6.6.
- If $\tilde{\mathtt{n}}_2 \equiv \tilde{\mathtt{0}}$ and $\tilde{\mathtt{n}}_0 \equiv \tilde{\mathtt{1}}$ then $\mathtt{strict?}\,(\lambda\mathtt{x}^\iota.\mathtt{M}_2) \Downarrow_e \tilde{\mathtt{0}}$, so $[\![\mathtt{M}_2]\!]_{\rho[\emptyset/\mathtt{x}]} = \emptyset$ while $[\![\mathtt{M}_2]\!]_{\rho[\{0\}/\mathtt{x}]} \neq \emptyset$. Moreover $[\![\mathtt{M}_0[\tilde{\mathtt{0}}/\mathtt{x}]]\!]_\rho = \{n+1\}$ implies $[\![\mathtt{M}_0]\!]_{\rho[\emptyset/\mathtt{x}]} = \emptyset$. Thus $[\![\mathtt{if}\ \mathtt{M}_0\,\mathtt{M}_1\,\mathtt{M}_2]\!]_{\rho[\emptyset/\mathtt{x}]} = \emptyset$. Since $[\![\mathtt{if}\ \mathtt{M}_0\,\mathtt{M}_1\,\mathtt{M}_2]\!]_{\rho[\{0\}/\mathtt{x}]} \neq \emptyset$ the proof follows by Theorem 6.6.

- The cases $(\lambda?\mathtt{succ})$, $(?\mathtt{succ})$, $(\lambda?num)$ or $(\lambda?x)$ are easy.
- If the derivation ends with $\dfrac{\mathtt{strict?}\,(\lambda\mathtt{x}^\iota.\mathtt{M}\,\tilde{\mathtt{0}}) \Downarrow_e \tilde{\mathtt{n}}}{\mathtt{strict?}\,(\lambda\mathtt{x}^\iota.\mathtt{strict?}\,\mathtt{M}) \Downarrow_e \tilde{\mathtt{n}}}\,(\lambda??)$, remark that $[\![\mathtt{M}\,\tilde{\mathtt{0}}]\!] \neq \emptyset$ if and only if $[\![\mathtt{strict?}\,\mathtt{M}]\!] \neq \emptyset$. By Theorem 6.6, the proof is easy.
- If the last applied rule is $(?0\mathtt{gor})$, $(?2\mathtt{gor})$, $(\lambda?0\mathtt{gor})$, $(\lambda?1\mathtt{gor})$ or $(\lambda?2\mathtt{gor})$ then the proof is similar to one of the previous cases.

□

7 Correctness

The operational behaviour may be related to the denotational model in a weaker sense than correctness. The denotational semantics is said to be *adequate* when $[\![\mathtt{M}]\!] = [\![\tilde{\mathtt{n}}]\!]$ and $\mathtt{M} \Downarrow_e \tilde{\mathtt{n}}$ are logically equivalent for any program $\mathtt{M}$, numeral $\tilde{\mathtt{n}}$.
The proof of adequacy is based on a computability argument in Tait style and it was used in [15] for Scott-continuous domains.

Definition 7.1 *The predicate $Comp(B, \mathtt{M}, \sigma)$ holds whenever $B \vdash \mathtt{M} : \sigma$ and one of the following cases is satisfied:*

(i) *$B = \emptyset$ and $\sigma = \iota$ if and only if $[\![\mathtt{M}]\!]_\rho = [\![\tilde{\mathtt{n}}]\!]_\rho$ implies $\mathtt{M} \Downarrow_e \tilde{\mathtt{n}}$, for each $\tilde{\mathtt{n}}$;*
(ii) *$B = \emptyset$ and $\sigma = \mu \rightarrowtail \tau$ if and only if $Comp(\emptyset, \mathtt{N}, \mu)$ implies $Comp(\emptyset, \mathtt{MN}, \tau)$;*
(iii) *$B = \{\mathtt{x}_0 : \nu_0, ..., \mathtt{x}_n\}$ for some $n \geq 1$, if and only if $Comp(\emptyset, \mathtt{N}_0, \nu_0)$ for all $i \leq n$ implies $Comp(\emptyset, \mathtt{M}[\mathtt{N}_0/\mathtt{x}_0, ..., \mathtt{N}_n/\mathtt{x}_n], \sigma)$.*

Note that $Comp(\emptyset, \mathtt{M}, \sigma \rightarrowtail \tau)$ and $Comp(\emptyset, \mathtt{N}, \sigma)$ imply $Comp(\emptyset, \mathtt{MN}, \tau)$.

Property 7.2 *$Comp(\{\mathtt{x}_0 : \nu_0, ... , \mathtt{x}_n : \nu_n\}, \mathtt{M}, \tau_1 \rightarrowtail ..... \rightarrowtail \tau_m \rightarrowtail \iota)$ if and only if, for all N_i and P_j such that $Comp(\emptyset, \mathtt{N}_i, \nu_i)$ and $Comp(\emptyset, \mathtt{P}_j, \tau_j)$ (where $i \leq n$, $j \leq m$) $[\![\mathtt{M}[\mathtt{N}_0/\mathtt{x}_0, ..., \mathtt{N}_n/\mathtt{x}_n]\mathtt{P}_1...\mathtt{P}_m]\!]_\rho = [\![\tilde{\mathtt{n}}]\!]_\rho$ implies $\mathtt{M}[\mathtt{N}_0/\mathtt{x}_0, ..., \mathtt{N}_n/\mathtt{x}_n]\mathtt{P}_1...\mathtt{P}_m \Downarrow_e \tilde{\mathtt{n}}$.*

The previous property will often be used implicitly in the next lemma.

Lemma 7.3 *If $B \vdash \mathtt{M} : \sigma$ then $Comp(B, \mathtt{M}, \sigma)$.*

Proof. The proof is given by induction on the derivation proving $B \vdash \mathtt{M} : \sigma$.

- Suppose $\sigma = \tau_1 \rightarrowtail ... \rightarrowtail \tau_m \rightarrowtail \iota$ ($m \in \mathbb{N}$) and $B[\mathtt{x} : \sigma] \vdash \mathtt{x} : \sigma$ and $Comp(\emptyset, \mathtt{P}, \sigma)$. If $Comp(\emptyset, \mathtt{N}_i, \tau_i)$ $(1 \leq i \leq m)$ and $[\![\mathtt{x}[\mathtt{P}/\mathtt{x}]\mathtt{N}_1...\mathtt{N}_m]\!]_\rho = [\![\tilde{\mathtt{n}}]\!]_\rho$ then $\mathtt{P}\mathtt{N}_1...\mathtt{N}_m \Downarrow_e \tilde{\mathtt{n}}$ since $Comp(\emptyset, \mathtt{P}, \sigma)$, thus $Comp(B, \mathtt{x}, \sigma)$, by Property 7.2.
- If $B \vdash \tilde{\mathtt{n}} : \iota$ then the proof is trivial.
- Suppose $B \vdash \mathtt{if} : \iota \rightarrowtail \iota \rightarrowtail \iota \rightarrowtail \iota$ and $Comp(\emptyset, \mathtt{N}_i, \iota)$ $(1 \leq i \leq 3)$.
  If $[\![\mathtt{if}\,\mathtt{N}_1\mathtt{N}_2\mathtt{N}_3]\!]_\rho = [\![\tilde{\mathtt{n}}]\!]_\rho$ then either $[\![\mathtt{N}_1]\!]_\rho = [\![\tilde{\mathtt{0}}]\!]_\rho$ or $[\![\mathtt{N}_1]\!]_\rho = [\![\widetilde{\mathtt{m}+1}]\!]_\rho$, by interpretation of $\mathtt{if}$. In the first case, clearly $[\![\mathtt{N}_2]\!]_\rho = [\![\tilde{\mathtt{n}}]\!]_\rho$ for some $\tilde{\mathtt{n}}$. Thus, both $\mathtt{N}_1 \Downarrow_e \tilde{\mathtt{0}}$ and $\mathtt{N}_2 \Downarrow_e \tilde{\mathtt{n}}$ by hypotheses $Comp(\emptyset, \mathtt{N}_1, \iota)$ and $Comp(\emptyset, \mathtt{N}_2, \iota)$, and the proof follows by applying the evaluation rules. The second case is similar.
- The cases $B \vdash \mathtt{succ} : \iota \rightarrowtail \iota$, $B \vdash \mathtt{pred} : \iota \rightarrowtail \iota$ are easier than the previous one.
- We will show that $Comp(B, \mathtt{P}, \mu \rightarrowtail \tau)$ and $Comp(B, \mathtt{Q}, \mu)$ imply $Comp(B, \mathtt{PQ}, \tau)$.
  Let $B = \mathtt{x}_1 : \nu_1, ..., \mathtt{x}_h : \nu_h$ ($h \in \mathbb{N}$) and $Comp(\emptyset, \mathtt{N}_i, \nu_i)$ $(1 \leq i \leq h)$.
  Let $\tau = \tau_1 \rightarrowtail ... \rightarrowtail \tau_m \rightarrowtail \iota$ ($m \in \mathbb{N}$) and $Comp(\emptyset, \mathtt{R}_i, \tau_i)$ $(1 \leq i \leq m)$.
  Thus $Comp(\emptyset, \mathtt{P}[\mathtt{N}_1/\mathtt{x}_1, ..., \mathtt{N}_h/\mathtt{x}_h], \mu \rightarrowtail \tau)$ and $Comp(\emptyset, \mathtt{Q}[\mathtt{N}_1/\mathtt{x}_1, ..., \mathtt{N}_h/\mathtt{x}_h], \mu)$ by hypotheses, so $Comp(\emptyset, \mathtt{P}[\mathtt{N}_1/\mathtt{x}_1, ..., \mathtt{N}_h/\mathtt{x}_h]\mathtt{Q}[\mathtt{N}_1/\mathtt{x}_1, ..., \mathtt{N}_h/\mathtt{x}_h]\mathtt{R}_1...\mathtt{R}_n, \iota)$.
  The proof follows by Definition 7.1.
- We show that $Comp(B[\mathtt{x} : \mu], \mathtt{P}, \tau)$ implies $Comp(B, \lambda\mathtt{x}^\sigma.\mathtt{P}, \mu \rightarrowtail \tau)$. Without loss of generality let $B = \mathtt{x}_1 : \nu_1, ..., \mathtt{x}_h : \nu_h$ ($h \in \mathbb{N}$) and $Comp(\emptyset, \mathtt{N}_i, \nu_i)$ $(1 \leq i \leq h)$.
  Let $\tau = \tau_1 \rightarrowtail ... \rightarrowtail \tau_m \rightarrowtail \iota$ ($m \in \mathbb{N}$) and $Comp(\emptyset, \mathtt{R}_i, \tau_i)$ $(1 \leq i \leq m)$.
  Let $Comp(\emptyset, \mathtt{Q}, \mu)$ and $[\![(\lambda\mathtt{x}^\sigma.\mathtt{P})[\mathtt{N}_1/\mathtt{x}_1, ..., \mathtt{N}_h/\mathtt{x}_h]\mathtt{Q}\mathtt{R}_1...\mathtt{R}_n]\!]_\rho = [\![\tilde{\mathtt{n}}]\!]_\rho$, for some $\tilde{\mathtt{n}}$; therefore, $[\![(\lambda\mathtt{x}^\sigma.\mathtt{P})[\mathtt{N}_1/\mathtt{x}_1, ..., \mathtt{N}_h/\mathtt{x}_h]\mathtt{Q}\mathtt{R}_1...\mathtt{R}_n]\!]_\rho = [\![\mathtt{P}[\mathtt{Q}/\mathtt{x}, \mathtt{N}_1/\mathtt{x}_1, ..., \mathtt{N}_h/\mathtt{x}_h]\mathtt{R}_1...\mathtt{R}_n]\!]_\rho$ by Lemmas 6.8.
  But $Comp(B[\mathtt{x} : \mu], \mathtt{P}, \tau)$ implies $Comp(\emptyset, \mathtt{P}[\mathtt{Q}/\mathtt{x}, \mathtt{N}_1/\mathtt{x}_1, ..., \mathtt{N}_h/\mathtt{x}_h]\mathtt{R}_1...\mathtt{R}_n, \iota)$, hence it follows that $\mathtt{P}[\mathtt{Q}/\mathtt{x}, \mathtt{N}_1/\mathtt{x}_1, ..., \mathtt{N}_h/\mathtt{x}_h]\mathtt{R}_1...\mathtt{R}_n \Downarrow_e \tilde{\mathtt{n}}$ by Definition 7.1. The proof follows by rule (*head*).
- Suppose $B \vdash \mathtt{gor} : \iota \rightarrowtail \iota \rightarrowtail \iota \rightarrowtail \iota$ and $Comp(\emptyset, \mathtt{N}_i, \iota)$ $(1 \leq i \leq 3)$.
  Let $[\![\mathtt{gor}\,\mathtt{N}_1\,\mathtt{N}_2\,\mathtt{N}_3]\!]_\rho = [\![\tilde{\mathtt{n}}]\!]_\rho$. There are 3 cases by $\mathtt{gor}$ interpretation. If $\tilde{\mathtt{n}} = \tilde{\mathtt{3}}$, $[\![\mathtt{N}_1]\!]_\rho = [\![\tilde{\mathtt{0}}]\!]_\rho$ and $[\![\mathtt{N}_2]\!]_\rho = [\![\widetilde{\mathtt{k}+1}]\!]_\rho$ then $\mathtt{N}_1 \Downarrow_e \tilde{\mathtt{0}}$ and $\mathtt{N}_2 \Downarrow_e \widetilde{\mathtt{k}+1}$ by hypotheses; thus the proof follows by rule ($0\mathtt{gor}$). The remaining cases are similar.
- Suppose $B \vdash \mathtt{strict?} : (\iota \rightarrowtail \iota) \rightarrowtail \iota$ and $Comp(\emptyset, \mathtt{N}, \iota \rightarrowtail \iota)$.
  We will show that, if $[\![\mathtt{strict?}\,\mathtt{N}]\!]_\rho = [\![\tilde{\mathtt{n}}]\!]_\rho$ then $\mathtt{strict?}\,\mathtt{N} \Downarrow_e \tilde{\mathtt{n}}$. It is easy to check that, both $Comp(\emptyset, \Omega_\iota, \iota)$ and $Comp(\emptyset, \tilde{\mathtt{0}}, \iota)$, so both $Comp(\emptyset, \mathtt{N}\Omega_\iota, \iota)$ and $Comp(\emptyset, \mathtt{N}\tilde{\mathtt{0}}, \iota)$ by hypothesis. By interpretation of $\mathtt{strict?}$ there are two cases.
  - If $\tilde{\mathtt{n}} \equiv \tilde{\mathtt{0}}$ then $[\![\mathtt{N}\Omega_\iota]\!]_\rho = \emptyset$ and $[\![\mathtt{N}\tilde{\mathtt{0}}]\!]_\rho = [\![\tilde{\mathtt{m}}]\!]_\rho$. Hence $\mathtt{N}\tilde{\mathtt{0}} \Downarrow_e \tilde{\mathtt{m}}$; moreover $\mathtt{strict?}\,\mathtt{N} \Downarrow_e \tilde{\mathtt{k}}$ where $\tilde{\mathtt{k}} \in \{\tilde{\mathtt{0}}, \tilde{\mathtt{1}}\}$ by Lemma 4.2. If $\tilde{\mathtt{k}} \not\equiv \tilde{\mathtt{1}}$ then $M\Omega_\iota \Downarrow_e \tilde{\mathtt{m}}$ by Lemma 4.3, thus $[\![\mathtt{N}\Omega_\iota]\!]_\rho = [\![\tilde{\mathtt{m}}]\!]_\rho \neq \emptyset$ by Lemma 6.9 against our hypothesis.
  - If $\tilde{\mathtt{n}} \not\equiv \tilde{\mathtt{0}}$ then $[\![\mathtt{N}\Omega_\iota]\!]_\rho = [\![\mathtt{N}\tilde{\mathtt{0}}]\!]_\rho = [\![\tilde{\mathtt{m}}]\!]_\rho$. Hence $\mathtt{N}\Omega_\iota \Downarrow_e \tilde{\mathtt{m}}$ and $\mathtt{N}\tilde{\mathtt{0}} \Downarrow_e \tilde{\mathtt{m}}$; thus the proof follows by Lemma 4.2.
- Let $B \vdash \mathtt{Y}_\sigma : (\sigma \rightarrowtail \sigma) \rightarrowtail \sigma$ where $\sigma = \tau_1 \rightarrowtail ... \rightarrowtail \tau_m \rightarrowtail \iota$ ($m \in \mathbb{N}$).
  The case $m = 0$ is trivial [7], so let $m \geq 1$. Without loss of generality assume $B = \emptyset$, $Comp(\emptyset, \mathtt{Q}, \sigma \rightarrowtail \sigma)$ and $Comp(\emptyset, \mathtt{R}_i, \tau_i)$ $(1 \leq i \leq m)$.
  We will prove that, if $[\![\mathtt{Y}_\sigma\mathtt{Q}\mathtt{R}_1...\mathtt{R}_n]\!]_\rho = [\![\tilde{\mathtt{n}}]\!]_\rho$ then $\mathtt{Y}_\sigma\mathtt{Q}\mathtt{R}_1...\mathtt{R}_n \Downarrow_e \tilde{\mathtt{n}}$. Note that there exists $k \in \mathbb{N}$ such that $[\![\mathtt{Y}^k_\sigma\mathtt{Q}\mathtt{R}_1...\mathtt{R}_n]\!]_\rho = [\![\mathtt{Y}_\sigma\mathtt{Q}\mathtt{R}_1...\mathtt{R}_n]\!]_\rho$ by Theorem 6.7. Thus

[7] Note that Ω_σ and $\mathtt{Y}^k_\sigma$ are defined using only $\mathtt{Y}_\iota$.

$\mathtt{Y}^k_\sigma \mathtt{QR}_1 ... \mathtt{R}_n \Downarrow_e \tilde{\mathtt{n}}$ by the previous points of this Lemma. The proof follows by Theorem 3.4. □

Corollary 7.4 *The stable models are adequate for $\mathcal{S}t\mathcal{P}CF$.*

Proof. Lemma 7.3 (together with Definition 7.1) and Lemma 6.9 imply that $[\![\mathtt{M}]\!] = [\![\tilde{\mathtt{n}}]\!]$ if and only if $\mathtt{M} \Downarrow_e \tilde{\mathtt{n}}$, for any program $\mathtt{M}$, numeral $\tilde{\mathtt{n}}$. □

Theorem 7.5 *The stable models are correct for $\mathcal{S}t\mathcal{P}CF$.*

Proof. Let $B \vdash \mathtt{M} : \sigma$ and $B \vdash \mathtt{N} : \sigma$ such that $[\![\mathtt{M}]\!]_\rho = [\![\mathtt{N}]\!]_\rho$, for each environment $\rho \in \mathsf{Env}_B$. If $C[.]$ is a B^σ-context such that both $C[\mathtt{M}]$ and $C[\mathtt{N}]$ are programs and $C[\mathtt{M}] \Downarrow_e \tilde{\mathtt{n}}$ for some value $\tilde{\mathtt{n}}$, then $[\![C[\mathtt{M}]]\!] = [\![\tilde{\mathtt{n}}]\!]$ by Lemma 6.9.
Since $[\![C[\mathtt{N}]]\!] = [\![C[\mathtt{M}]]\!] = [\![\tilde{\mathtt{n}}]\!]$ by Lemma 6.8, it follows that $C[\mathtt{N}] \Downarrow_e \tilde{\mathtt{n}}$ by adequacy. By definition of operational equivalence the proof is done. □

8 Definability and Full Abstraction

The proof of full abstraction is done like the one for $\mathcal{P}CF$ and Scott's domains [15]. If x_0 is a finite clique (in a coherence space interpretation of a type σ) then there exists a closed term $\mathtt{M}$ of $\mathcal{S}t\mathcal{P}CF$ such that $\vdash \mathtt{M} : \sigma$ and $[\![\mathtt{M}]\!] = x_0$. It follows that coherence spaces (and stable models) are fully abstract for $\mathcal{S}t\mathcal{P}CF$.

Definition 8.1 *Let x be a finite clique of a coherence space in the range of the interpretation of $\mathcal{S}t\mathcal{P}CF$-types. The class of closed terms having x as interpretation is denoted by $\wr x \wr$, hence $\wr x \wr = \{\mathtt{M} \mid [\![\mathtt{M}]\!] = x\}$.*

$\wr a_1, ..., a_k \wr$ is used as an abbreviation for $\wr \{a_1, ..., a_k\} \wr$ and $\wr x \wr = \mathtt{M}$ is used as an abbreviation for $\mathtt{M} \in \wr x \wr$.
If $B \vdash \mathtt{P}_i : \iota$, $B \vdash \mathtt{M}_i : \iota$ where $i \leq 2$ then $\mathtt{gif}\ \mathtt{P}_0\ \mathtt{P}_1\ \mathtt{P}_2\ \mathtt{M}_0\ \mathtt{M}_1\ \mathtt{M}_2$ is used as an abbreviation for the term $\mathtt{if}\ \underline{(\mathtt{gor}\,\mathtt{P}_0\,\mathtt{P}_1\,\mathtt{P}_2)}\ \mathtt{M}_0\ \big(\mathtt{if}\ \underline{(\mathtt{pred}\,(\mathtt{gor}\,\mathtt{P}_0\,\mathtt{P}_1\,\mathtt{P}_2))}\ \mathtt{M}_1\ \mathtt{M}_2\ \big)$.

$$\text{Clearly } [\![\mathtt{gif}\ \mathtt{P}_0\ \mathtt{P}_1\ \mathtt{P}_2\ \mathtt{M}_0\ \mathtt{M}_1\ \mathtt{M}_2\ ]\!]_\rho = \begin{cases} [\![\mathtt{M}_0]\!]_\rho & \text{if } [\![\mathtt{P}_0]\!]_\rho = \{0\},\ [\![\mathtt{P}_1]\!]_\rho = \{n+1\}, \\ [\![\mathtt{M}_1]\!]_\rho & \text{if } [\![\mathtt{P}_1]\!]_\rho = \{0\},\ [\![\mathtt{P}_2]\!]_\rho = \{n+1\}, \\ [\![\mathtt{M}_2]\!]_\rho & \text{if } [\![\mathtt{P}_2]\!]_\rho = \{0\},\ [\![\mathtt{P}_0]\!]_\rho = \{n+1\}, \\ \emptyset & \text{otherwise.} \end{cases}$$

An explicit operational description of $\mathtt{gif}$ is given in [52]. If $\mathtt{M}$ and $\mathtt{N}$ are programs then $\mathtt{M} \doteq \mathtt{N}$ is an abbreviation for the application of the following term to $\mathtt{M}$ and $\mathtt{N}$:

$$\mathtt{Y}_{\iota\to\iota\to\iota\to\iota}\Big(\lambda \mathtt{F}^{\iota\to\iota\to\iota\to\iota} \mathtt{x}^\iota \mathtt{y}^\iota . \mathtt{if}\ \mathtt{x}\ (\mathtt{if}\,\mathtt{y}\tilde{\mathtt{0}}\tilde{\mathtt{1}})\ (\mathtt{if}\ \mathtt{y}\ \tilde{\mathtt{1}}\ (\mathtt{F}(\mathtt{pred}\,\mathtt{x})(\mathtt{pred}\,\mathtt{y})))\Big).$$

It is easy to check that $[\![\mathtt{M} \doteq \mathtt{N}]\!] = \begin{cases} 0 & [\![\mathtt{M}]\!] = m = [\![\mathtt{N}]\!], \\ 1 & [\![\mathtt{M}]\!] = m \neq n = [\![\mathtt{N}]\!], \\ \emptyset & \text{otherwise.} \end{cases}$

Let $\mathtt{N}_0$ `or` $\mathtt{N}_1$ be an abbreviation for the term $\mathtt{if}\ \mathtt{N}_0\ (\mathtt{if}\ \mathtt{N}_1\ \tilde{\mathtt{0}}\ \tilde{\mathtt{0}})\ \mathtt{N}_1$ (being equivalent to $\mathtt{if}\ \mathtt{N}_0\ \tilde{\mathtt{0}}\ \mathtt{N}_1$, under the hypothesis that both $\mathtt{N}_0 \Downarrow_e$ and $\mathtt{N}_1 \Downarrow_e$). Let $\mathtt{N}_0$ `and` $\mathtt{N}_1$ be an abbreviation for the term $\mathtt{if}\ \mathtt{N}_0\ (\mathtt{if}\ \mathtt{N}_1\ \tilde{\mathtt{0}}\ \tilde{\mathtt{1}})\ (\mathtt{if}\ \mathtt{N}_1\ \tilde{\mathtt{1}}\ \tilde{\mathtt{1}})$. Let `not` $\mathtt{N}_0$ be an abbreviation for the term $\mathtt{if}\ \mathtt{N}_0\ \tilde{\mathtt{1}}\ \tilde{\mathtt{0}}$. It is easy to check that the operational behaviour of `and` , `or` and `not` is the expected one. Note that `and` , `or` and `not` are strict operators, in the sense that if one of their parameters diverges then their evaluation diverges. Last, let $\tilde{\mathtt{k}}$-`succ` $\mathtt{M}$ be an abbreviation for $\underbrace{(\mathtt{succ}\ .....(\mathtt{succ}}_{k}\ \mathtt{M})...)$ where $k \in \mathbb{N}$ and $\mathtt{M}$ is a term (possibly open) having type ι.

In order to help the reader, we will try to give an informal idea of the problems raised by definability proof by presenting some examples.

Example 8.2

a) *Consider* $(\{3\}, 4) \in |[\![\iota \rightarrowtail \iota]\!]|$*; clearly* $\wr(\{3\}, 4)\wr = \lambda \mathtt{x}^{\iota}.\mathtt{if}\,(\mathtt{x} \doteq \tilde{3})\tilde{4}\Omega_{\iota}$.

b) *Consider* $(\{(\{3\}, 4)\}, 5) \in |[\![(\iota \rightarrowtail \iota) \rightarrowtail \iota]\!]|$.
At a first sight, the term $\mathtt{M} \equiv \lambda \mathtt{f}^{\iota \rightarrowtail \iota}.\mathtt{if}\,(\mathtt{f}\tilde{3} \doteq \tilde{4})\tilde{5}\Omega_{\iota}$ *is a natural candidate for* $\wr(\{(\{3\}, 4)\}, 5)\wr$ *but unfortunately this impression is wrong. In fact,* $[\![\mathtt{M}]\!] = \{(\{(\{3\}, 4)\}, 5), (\{(\emptyset, 4)\}, 5)\}$. *It is easy to check that*

$$\wr(\{(\{3\}, 4)\}, 5)\wr = \lambda \mathtt{f}^{\iota \rightarrowtail \iota}.\mathtt{if} \begin{pmatrix} \mathtt{f}\tilde{3} \doteq \tilde{4} \quad \text{and} \\ \mathtt{strict?}(\lambda \mathtt{z}^{\iota}.\mathtt{f}(\tilde{3}\text{-}\mathtt{succ}\ \mathtt{z})) \end{pmatrix} \tilde{5}\ \Omega_{\iota}.$$

c) *Consider* $(\{(\underline{\{(\{3\}, 4)\}}, 5)\}, 6) \in |[\![((\iota \rightarrowtail \iota) \rightarrowtail \iota) \rightarrowtail \iota]\!]|$.
Thus $\mathtt{M} \equiv \lambda \mathtt{F}^{(\iota \rightarrowtail \iota) \rightarrowtail \iota}.\mathtt{if}\left(\mathtt{F}\underline{(\lambda \mathtt{x}^{\iota}.\mathtt{if}\,(\mathtt{x} \doteq \tilde{3})\tilde{4}\Omega_{\iota})} \doteq \tilde{5}\right)\tilde{6}\Omega_{\iota}$ *does not define the given token, in fact* $[\![\mathtt{M}]\!] = \{(\{(\underline{\{(\{3\}, 4)\}}, 5)\}, 6), (\{(\emptyset, 5)\}, 6)\}$.
It is easy to check that

$$\wr(\{(\{(\{3\}, 4)\}, 5)\}, 6)\wr = \lambda \mathtt{F}^{(\iota \rightarrowtail \iota) \rightarrowtail \iota}.\mathtt{if} \begin{pmatrix} \left(\mathtt{F}\,(\lambda \mathtt{x}^{\iota}.\mathtt{if}\,(\mathtt{x} \doteq \tilde{3})\tilde{4}\Omega_{\iota}) \doteq \tilde{5}\right)\ \text{and} \\ \mathtt{strict?}\left(\lambda \mathtt{z}^{\iota}.\mathtt{F}\,(\lambda \mathtt{x}^{\iota}.\mathtt{if}\,(\mathtt{x} \doteq \tilde{3})(\tilde{4}\text{-}\mathtt{succ}\ \mathtt{z})\Omega_{\iota})\right) \end{pmatrix} \tilde{6}\Omega_{\iota}.$$

d) *Let* $a = (\{(\underline{\{(\{(\{3\}, 4)\}, 5)\}}, 6)\}, 7) \in |[\![(((\iota \rightarrowtail \iota) \rightarrowtail \iota) \rightarrowtail \iota) \rightarrowtail \iota]\!]|$.
Note that the term $\mathtt{M} \equiv \lambda \mathtt{F}^{((\iota \rightarrowtail \iota) \rightarrowtail \iota) \rightarrowtail \iota}.\mathtt{if}\left(\mathtt{F}\underline{(\lambda \mathtt{f}^{\iota \rightarrowtail \iota}.\mathtt{if}\,((\mathtt{f}\tilde{3}) \doteq \tilde{4})\tilde{5}\Omega_{\iota})} \doteq \tilde{6}\right)\tilde{7}\Omega_{\iota}$ *does not define the given token, in fact*

$$[\![\mathtt{M}]\!] = \{(\{(\underline{\{(\{(\{3\}, 4)\}, 5)\}}, 6)\}, 7), (\{(\underline{\{(\{(\emptyset, 4)\}, 5)\}}, 6)\}, 7), (\{(\emptyset, 6)\}, 7)\}.$$

Let $\mathtt{N} \equiv \lambda \mathtt{F}^{((\iota \rightarrowtail \iota) \rightarrowtail \iota) \rightarrowtail \iota}.\mathtt{if}\left(\mathtt{F}\underline{\wr(\{(\{(\{3\}, 4)\}, 5)\}\wr} \doteq \tilde{6}\right)\tilde{7}\Omega_{\iota}$*, where* $\wr(\{(\{3\}, 4)\}, 5)\wr$ *is defined in* **b)**. *Again,* $\mathtt{N}$ *does not define the considered token. In fact, it is easy to*

check that $[\![\mathtt{M}]\!] = \{(\{(\{(\{(\{3\},4)\},5)\},6)\},7),(\{(\emptyset,6)\},7)\}$. *Finally,*

$$\wr a \wr = \lambda \mathtt{F}^{((\iota\rightarrowtail\iota)\rightarrowtail\iota)\rightarrowtail\iota}.\mathtt{if}\left(\begin{array}{l}\Big(\big(\mathtt{F}\ \wr(\{(\{(\{3\},4)\},5)\}\wr\big) \doteq \tilde{6}\Big)\ \mathtt{and}\\ \\ \mathtt{strict?}\Big(\lambda \mathtt{x}^{\iota}.\mathtt{F}\,(\lambda \mathtt{f}^{\iota\rightarrowtail\iota}.\mathtt{if}\,((\mathtt{f}\tilde{3}) \doteq \tilde{4})(\tilde{5}\text{-}\mathtt{succ}\ \mathtt{z})\Omega_{\iota})\Big)\end{array}\right)\tilde{7}\Omega_{\iota}.$$

e) *Let* $a = (\underbrace{\{(\{10\},11)\}}_{\iota\rightarrowtail\iota};\ \underbrace{\{(\overline{\{(\{3\},4)\}},5),(\overline{\{(\{3\},8)\}},9)\}}_{(\iota\rightarrowtail\iota)\rightarrowtail\iota};\ 6) \in |[\![(\iota \rightarrowtail \iota) \rightarrowtail ((\iota \rightarrowtail \iota) \rightarrowtail \iota) \rightarrowtail \iota]\!]|$.
Note that the term
$\mathtt{M} \equiv \lambda \mathtt{f}^{\iota\rightarrowtail\iota}\mathtt{F}^{(\iota\rightarrowtail\iota)\rightarrowtail\iota}.\mathtt{if}\Big(\mathtt{f}\tilde{1}\tilde{0} \doteq \tilde{1}\tilde{1}\ \mathtt{and}\ \ (\mathtt{F}\ \wr(\{(\{3\},4)\}\wr) \doteq \tilde{5}\ \mathtt{and}\ \ (\mathtt{F}\wr(\{(\{3\},8)\}\wr) \doteq \tilde{9}\Big)\tilde{6}\Omega_{\iota}$
does not define the given token a, in fact

$$[\![\mathtt{M}]\!] = \left\{\begin{array}{l}\big(\{(\{10\},11)\};\ \{(\underline{\{(\{3\},4)\}},5),(\underline{\{(\{3\},8)\}},9)\};\ 6\big)\\ \big(\{(\emptyset,11)\};\quad \{(\underline{\{(\{3\},4)\}},5),(\underline{\{(\{3\},8)\}},9)\};\ 6\big)\end{array}\right\}$$

It is easy to check that

$$\wr a \wr = \lambda \mathtt{f}^{\iota\rightarrowtail\iota}\mathtt{F}^{(\iota\rightarrowtail\iota)\rightarrowtail\iota}.\mathtt{if}\left(\begin{array}{l}\mathtt{f}\tilde{1}\tilde{0} \doteq \tilde{1}\tilde{1}\ \ \mathtt{and}\ \ \ \mathtt{strict?}(\lambda \mathtt{z}^{\iota}.\mathtt{f}(\tilde{1}\tilde{0}\text{-}\mathtt{succ}\ \mathtt{z}))\\ \\ (\mathtt{F}\ \wr(\{(\{3\},4)\}\wr) \doteq \tilde{5}\ \ \mathtt{and}\ \ \ (\mathtt{F}\wr(\{(\{3\},8)\}\wr) \doteq \tilde{9}\end{array}\right)\tilde{6}\Omega_{\iota}.$$

The following property is the crucial point enabling us to prove the definability. It is a formalization of the technique (illustrated by some of the previous examples) which allows us to check, syntactically, the "minimality" (with respect to the stable order [4]) of an input.

Property 8.3 *Let* $\vdash \mathtt{M} : \sigma \rightarrowtail \iota$ *and* $x \in \mathsf{Cl}_{fin}[\![\sigma]\!]$. *If* $x = \{a_0, ..., a_n\}$ *for some* $n \geq 1$, $x^{a_k} = \{(\emptyset, a_0), ..., (\emptyset, a_{k-1}), (\{0\}, a_k), (\emptyset, a_{k+1}), ..., (\emptyset, a_n)\} \in [\![\iota \rightarrowtail \sigma]\!]$ *for all* $k \leq n$ *and* $b \in \mathbf{N}$ *then the following conditions are equivalent:*

(i) $b \in \mathcal{F}([\![\mathtt{M}]\!])\,x$ *and* $\forall y \subseteq x,\ b \in \mathcal{F}([\![\mathtt{M}]\!])\,y$ *implies* $x = y$;
(ii) $b \in \mathcal{F}([\![\mathtt{M}]\!])(\mathcal{F}(\wr x^{a_k} \wr)\{0\})$ *while* $\mathcal{F}([\![\mathtt{M}]\!])(\mathcal{F}(\wr x^{a_k} \wr)\emptyset) = \emptyset$, *for all* $k \leq n$;
(iii) $b \in [\![\mathtt{M}\wr x \wr]\!]$ *and,* $\forall k \leq n$, $[\![\mathtt{strict?}(\lambda \mathtt{z}^{\iota}.\mathtt{M}(\wr x^{a_k} \wr \mathtt{z}))]\!] = \{0\}$.

Proof. Easy, by using Theorem 6.6. □

A last example may help the reader to understand a further problem arising from definability.

Example 8.4 *Let* $e = \left\{\begin{array}{ll}(\underline{\{(\{3\},30),(\{4\},41)\}},101), & (\underline{\{(\emptyset,90)\}},109),\\ (\underline{\{(\{3\},31),(\{5\},50)\}},102), & (\underline{\{(\{4\},40),(\{5\},51)\}},103)\end{array}\right\} \in \mathsf{Cl}\big([\![(\iota \rightarrowtail \iota) \rightarrowtail \iota]\!]\big)$.

Let $x_{101} = \{(\{3\},30),(\{4\},41)\}$, $x_{102} = \{(\{3\},31),(\{5\},50)\}$, $x_{103} = \{(\{4\},40),(\{5\},51)\}$, $x_{109} = \{(\emptyset,90)\}$ *and, note that they are pairwise incoherent. We will try to define the clique e in a compositional way, by using cliques defined in Fig. 5.*
Thus it is easy to check that:

$$\left\lceil \begin{pmatrix} (x_{101},0) \\ (x_{109},1) \end{pmatrix} \right\rceil = \lambda \mathtt{f}^{\iota\mapsto\iota}.\mathtt{if} \begin{pmatrix} \mathtt{f}\tilde{\mathtt{3}} \doteq \tilde{\mathtt{3}}\tilde{\mathtt{0}} \\ \text{and} \\ \mathtt{f}\tilde{\mathtt{4}} \doteq \tilde{\mathtt{4}}\tilde{\mathtt{1}} \end{pmatrix} \tilde{\mathtt{0}} \left(\mathtt{if}\,(\mathtt{f}\Omega_\iota \doteq \tilde{\mathtt{9}}\tilde{\mathtt{0}})\ \tilde{\mathtt{1}}\ \Omega_\iota\right) \qquad (\dagger)$$

$$\left\lceil \begin{pmatrix} (x_{101},1) \\ (x_{102},0) \\ (x_{103},0) \end{pmatrix} \right\rceil = \lambda \mathtt{f}^{\iota\mapsto\iota}.\mathtt{gif} \begin{pmatrix} \mathtt{if}\ (\mathtt{f}\tilde{\mathtt{3}} \doteq \tilde{\mathtt{3}}\tilde{\mathtt{0}})(\mathtt{if}\,(\mathtt{f}\tilde{\mathtt{4}} \doteq \tilde{\mathtt{4}}\tilde{\mathtt{1}})\tilde{\mathtt{0}}\Omega_\iota)(\mathtt{if} \begin{pmatrix} \mathtt{f}\tilde{\mathtt{3}} \doteq \tilde{\mathtt{3}}\tilde{\mathtt{1}} \\ \text{and} \\ \mathtt{f}\tilde{\mathtt{5}} \doteq \tilde{\mathtt{5}}\tilde{\mathtt{0}} \end{pmatrix} \tilde{\mathtt{1}}\Omega_\iota) \\ \mathtt{if}\ (\mathtt{f}\tilde{\mathtt{4}} \doteq \tilde{\mathtt{4}}\tilde{\mathtt{0}})(\mathtt{if}\,(\mathtt{f}\tilde{\mathtt{5}} \doteq \tilde{\mathtt{5}}\tilde{\mathtt{1}})\tilde{\mathtt{0}}\Omega_\iota)(\mathtt{if} \begin{pmatrix} \mathtt{f}\tilde{\mathtt{4}} \doteq \tilde{\mathtt{4}}\tilde{\mathtt{1}} \\ \text{and} \\ \mathtt{f}\tilde{\mathtt{3}} \doteq \tilde{\mathtt{3}}\tilde{\mathtt{0}} \end{pmatrix} \tilde{\mathtt{1}}\Omega_\iota) \\ \mathtt{if}\ (\mathtt{f}\tilde{\mathtt{5}} \doteq \tilde{\mathtt{5}}\tilde{\mathtt{0}})(\mathtt{if}\,(\mathtt{f}\tilde{\mathtt{3}} \doteq \tilde{\mathtt{3}}\tilde{\mathtt{1}})\tilde{\mathtt{0}}\Omega_\iota)(\mathtt{if} \begin{pmatrix} \mathtt{f}\tilde{\mathtt{5}} \doteq \tilde{\mathtt{5}}\tilde{\mathtt{1}} \\ \text{and} \\ \mathtt{f}\tilde{\mathtt{4}} \doteq \tilde{\mathtt{4}}\tilde{\mathtt{0}} \end{pmatrix} \tilde{\mathtt{1}}\Omega_\iota) \\ \tilde{\mathtt{1}} \\ \tilde{\mathtt{0}} \\ \tilde{\mathtt{0}} \end{pmatrix}$$

$$\left\lceil \begin{pmatrix} (x_{109},0) \\ (x_{102},1) \\ (x_{103},1) \end{pmatrix} \right\rceil = \lambda \mathtt{f}^{\iota\mapsto\iota}.\mathtt{if}\ (\mathtt{f}\tilde{\mathtt{5}} \doteq \tilde{\mathtt{9}}\tilde{\mathtt{0}})\left(\mathtt{if}\,(\mathtt{f}\Omega_\iota \doteq \tilde{\mathtt{9}}\tilde{\mathtt{0}})\ \tilde{\mathtt{0}}\ \Omega_\iota\right) \begin{pmatrix} \mathtt{if}(\mathtt{f}\tilde{\mathtt{5}} \doteq \tilde{\mathtt{5}}\tilde{\mathtt{0}})(\mathtt{if}\,(\mathtt{f}\tilde{\mathtt{3}} \doteq \tilde{\mathtt{3}}\tilde{\mathtt{1}})\tilde{\mathtt{1}}\Omega_\iota)(\mathtt{if} \begin{pmatrix} \mathtt{f}\tilde{\mathtt{5}} \doteq \tilde{\mathtt{5}}\tilde{\mathtt{1}} \\ \text{and} \\ \mathtt{f}\tilde{\mathtt{4}} \doteq \tilde{\mathtt{4}}\tilde{\mathtt{0}} \end{pmatrix} \tilde{\mathtt{1}}\Omega_\iota) \end{pmatrix}$$

† It would be clear that, in case $\mathtt{f}\tilde{\mathtt{3}} \doteq \tilde{\mathtt{3}}\tilde{\mathtt{0}}$ and $\mathtt{f}\tilde{\mathtt{4}} \doteq \tilde{\mathtt{4}}\tilde{\mathtt{1}}$ there is no need for checking the minimality, since it must be $[\![\mathtt{f}\Omega_\iota]\!] = \emptyset$ by monotonicity and correctness.

Figure 5. Examples of Clique Definitions

$$\wr e \wr = \lambda \mathtt{f}^{\iota\mapsto\iota}.\mathtt{gif}\left(\left\lceil \begin{pmatrix} (x_{101},0) \\ (x_{109},1) \end{pmatrix} \right\rceil\ \mathtt{f}\right)\ \left(\left\lceil \begin{pmatrix} (x_{101},1) \\ (x_{102},0) \\ (x_{103},0) \end{pmatrix} \right\rceil\ \mathtt{f}\right)\ \left(\left\lceil \begin{pmatrix} (x_{109},0) \\ (x_{102},1) \\ (x_{103},1) \end{pmatrix} \right\rceil\ \mathtt{f}\right)$$

$$\widetilde{\mathtt{101}}\ \begin{pmatrix} \mathtt{if}\ (\mathtt{f}\tilde{\mathtt{5}} \doteq \tilde{\mathtt{5}}\tilde{\mathtt{0}})(\mathtt{if}\,(\mathtt{f}\tilde{\mathtt{3}} \doteq \tilde{\mathtt{3}}\tilde{\mathtt{1}})\ \widetilde{\mathtt{102}}\,\Omega_\iota)(\mathtt{if} \begin{pmatrix} \mathtt{f}\tilde{\mathtt{5}} \doteq \tilde{\mathtt{5}}\tilde{\mathtt{1}} \\ \text{and} \\ \mathtt{f}\tilde{\mathtt{4}} \doteq \tilde{\mathtt{4}}\tilde{\mathtt{0}} \end{pmatrix} \widetilde{\mathtt{103}}\,\Omega_\iota) \end{pmatrix}\ \widetilde{\mathtt{109}}$$

Clearly, one can find simpler terms defining e.

A non standard measure on types will be useful in the proof of the Lemma 8.5. The RANK of a type is defined inductively as follows:

- $\text{RANK}(\iota) = 0$
- $\text{RANK}(\sigma \rightarrowtail \tau) = 1 + \text{RANK}(\sigma) + \text{RANK}(\tau)$.

It is easy to check that $\text{RANK}(\mu_1 \rightarrowtail ..... \rightarrowtail \mu_m \rightarrowtail \iota) = m + \sum_{j=1}^{m} \text{RANK}(\mu_i)$.

Lemma 8.5 (Definability)
If $\sigma = \tau_1 \rightarrowtail ..... \rightarrowtail \tau_k \rightarrowtail \iota$ for some $k \geq 0$ and $u \in \mathsf{Cl}_{fin}(\llbracket\sigma\rrbracket)$ then u is definable.

Proof. The proof is given by induction on the pair $\langle\ \text{RANK}(\sigma),\ ||u||\ \rangle$ ordered in a lexicographic way.

♦ If $\text{RANK}(\sigma) = 0$ then $\llbracket\sigma\rrbracket = \mathbf{N}$ and $\sigma = \iota$. Thus Ω_ι and numerals define all possible finite cliques, since $\mathsf{Cl}_{fin}(\mathbf{N}) = \{\emptyset\} \cup \{\{n\} \,/\, n \in |\mathbb{N}|\}$.

♦ If $\text{RANK}(\sigma) = 1$ then $\llbracket\sigma\rrbracket = \mathbf{N} \Rightarrow \mathbf{N}$ and $\sigma = \iota \rightarrowtail \iota$.

- If $||u|| = 0$ then $u = \emptyset$ is defined by $\Omega_{\iota\rightarrowtail\iota}$.
- Let $||u|| = 1$ and $u = \{(y^0, d^0)\}$ such that $y^0 \in \mathsf{Cl}_{fin}(\mathbf{N})$ and $d^0 \in \mathbb{N}$.
  If $y^0 = \emptyset$ then $\wr u \wr = \lambda \mathtt{z}^\iota.\wr d^0 \wr$. If $y^0 \neq \emptyset$ then $||y^0|| = 1$, i.e. it contains a numeral since $\mathsf{Cl}(\mathbf{N})$ is a flat cpo. If $y^0 = \{n\}$ then the program defining the clique has the following shape: $\lambda \mathtt{z}^\iota.\mathtt{if}\ (\mathtt{z} \doteq \tilde{\mathtt{n}})\, \wr d^0 \wr\, \Omega_\iota$.
- Let $||u|| > 1$ and $(y^0, d^0) \in u$ where $y^0 \in \mathsf{Cl}_{fin}(\mathbf{N})$ and $d^0 \in \mathbb{N}$.
  Clearly $y^0 \neq \emptyset$ by Lemma 6.2. If $u' = u - \{(y^0, d^0)\}$ and $y^0 = \{n\}$ then the program defining the clique has the following shape: $\lambda \mathtt{z}^\iota.(\mathtt{z} \doteq^\iota \wr n \wr)\, \wr d^0 \wr\, (\wr u' \wr \mathtt{z})$ where $\wr u' \wr$ is well defined by induction, since $||u'|| < ||u||$.

♦ Suppose $\text{RANK}(\sigma) \geq 2$ and $k = 1$; so $\sigma = \tau_1 \rightarrowtail \iota$ and $\tau_1 = \mu_1 \rightarrowtail ..... \rightarrowtail \mu_m \rightarrowtail \iota$, for some $m \in \mathbb{N}$. Clearly $\text{RANK}(\sigma) = 1 + \text{RANK}(\tau_1) = 1 + m + \sum_{j=1}^{m} \text{RANK}(\mu_i)$.

- If $||u|| = 0$ then $u = \emptyset$ is defined by $\Omega_{\tau_1\rightarrowtail\iota}$.
- Suppose $||u|| = 1$ and $u = \{(y^0, d^0)\}$ where $y^0 \in \llbracket\mu_1 \rightarrowtail ..... \rightarrowtail \mu_m \rightarrowtail \iota\rrbracket$.
  At this crucial point the proof proceeds by induction on $||y^0||$ too.
  - If $y^0 = \emptyset$ then $\wr u \wr = \lambda \mathtt{F}^{\tau_1}.\wr d^0 \wr$.
  - Suppose $y^0 = \{a^0, ..., a^n\}$ $(n \geq 0)$ where $a^i = (x_1^i; ...; x_m^i; b^i)$ and $x_j^i \in \llbracket\mu_j\rrbracket$ $(i \leq n,\ 1 \leq j \leq m)$. If $y' = y^0 - \{a^0\}$ then $u' = \{(y', d^0)\}$ is a clique definable by induction, while $u^0 = \{(\{(x_2^0; ...; x_m^0; b^0)\}, 0)\}$ is a clique definable by induction on the RANK. Clearly $a^0 = (x_1^0; ...; x_m^0; b^0)$. [8]

[8] The term $\mathtt{M} \equiv \lambda \mathtt{F}^{\tau_1}.\mathtt{if}\ \left((\mathtt{F}\, \wr x_1^0 \wr ... \wr x_m^0 \wr) \doteq^\iota \wr b^0 \wr \text{ and } \wr u^0 \wr (\mathtt{F}\, \wr x_1^0 \wr))\right)\, (\wr u' \wr \mathtt{F})\, \Omega_\iota$ does not define u (see Examples 8.2.b/c/d/e).

If $x_1^0 = \{c_1^0, ..., c_{h_0}^0\}$ and $k \leq h_0$ then let $x_1^{c_k^0} = \{(\emptyset, c_1^0), ..., (\{0\}, c_k^0), ..., (\emptyset, c_{h_0}^0)\}$ so

$$\wr u \wr = \lambda \mathtt{F}^{\sigma}.\mathtt{if} \left(\begin{array}{l} (\mathtt{F}\ \wr x_1^0 \wr ... \wr x_m^0 \wr) \doteq^{\iota} \wr b^0 \wr \quad \text{and} \\ \mathtt{strict?}(\lambda \mathtt{x}^{\iota}.\mathtt{F}\ (\wr x_1^{c_1^0} \wr \mathtt{x})\ \wr x_2^0 \wr ... \wr x_m^0 \wr) \\ \quad \text{and} \qquad ..... \qquad \text{and} \\ \mathtt{strict?}(\lambda \mathtt{x}^{\iota}.\mathtt{F}\ (\wr x_1^{c_{h_0}^0} \wr \mathtt{x})\ \wr x_2^0 \wr ... \wr x_m^0 \wr) \\ \quad \text{and}\ \ \wr u^0 \wr (\mathtt{F} \wr x_1^0 \wr) \end{array} \right) (\wr u' \wr \mathtt{F})\ \ \Omega_{\iota}.$$

Now an informal justification of the previous reasoning is given. Essentially $y^0 = \{a^0, ..., a^n\}$ is a set of constraints on $\mathtt{F}^{\sigma}$. If $a^i = (x_1^i; ...; x_m^i; b^i)$ then we must check that $(\mathtt{F}\ \wr x_1^i \wr ... \wr x_m^i \wr) \doteq^{\iota} \wr b^i \wr)$ in a minimal way with respect to the stable order, thus $(x_1^i; ...; x_m^i; b^i) \in [\![\mathtt{F}]\!]$. A descriptive analysis of the three arguments of `if` in the term $\wr u \wr$ defined before can be done as follows.

1. The first argument verifies that the constraint a^0 is satisfied. More explicitly, it verifies both $(\mathtt{F}\ \wr x_1^0 \wr ... \wr x_m^0 \wr) \doteq^{\iota} \wr b^0 \wr$ and the stable minimality of x_1^0, while the further stable minimal constraints are verified inductively by the term $\wr u^0 \wr$.
2. The second argument inductively checks the constraints $\{a^1, ..., a^n\}$ and, in the affirmative case, gives out d^0 as the result.

The third argument "loops forever", therefore $[\![\wr u \wr]\!] = u$.

- Suppose $\|u\| \geq 2$ and $u = \{(y^0, d^0), ..., (y^p, d^p)\}$ for some $p \geq 1$. If $i, j \leq p$ and $i \neq j$ then $y^i \cup y^j \notin \mathsf{Cl}(U)$, by Lemma 6.2; thus $y^i \neq \emptyset$ for each $j \leq p$.

  Let $y^j = \{a^{(j,0)}, ..., a^{(j,n_j)}\}$ for some $n_j \geq 0$ and $a^{(j,i)} = (x_1^{(j,i)}; ...; x_m^{(j,i)}; b^{(j,i)})$ where $x_q^{(j,i)} \in [\![\mu_q]\!]$, $b^{(j,i)} \in \mathbb{N}$, for all $1 \leq q \leq m$, $i \leq n_j$ and $j \leq p$.
  There are $a^{(0,k_0)} \in y^0$ and $a^{(1,k_1)} \in y^1$ for some $k_0 \leq n_0$, $k_1 \leq n_1$, s.t. $a^{(0,k_0)} \smile a^{(1,k_1)}$. From $a^{(0,k_0)} = (x_1^{(0,k_0)}, ..., x_m^{(0,k_0)}, b^{(0,k_0)})$, $a^{(1,k_1)} = (x_1^{(1,k_1)}, ..., x_m^{(1,k_1)}, b^{(1,k_1)})$ and by Corollary 6.3, it follows that $z_i = x_i^{(0,k_0)} \cup x_i^{(1,k_1)} \in \mathsf{Cl}([\![\mu_i]\!])$ for all $1 \leq i \leq m$.
  The cliques z_i for all $1 \leq i \leq m$, and the following cliques (well-defined by Lemma 6.2) are definable by induction on the RANK,

$$v_1 = \{(y^i, d^i) \in u \ / \ a^{(0,k_0)} \in y^i\}$$

$$v_2 = \{(y^i, d^i) \in u \ / \ a^{(1,k_1)} \in y^i\}$$

$$v_3 = u - (v_1 \cup v_2) = \{(y^i, d^i) \in u \ / \ a^{(0,k_0)} \notin y^i \text{ and } a^{(1,k_1)} \notin y^i\}$$

$$w_2 = \{(y^i, \tilde{\mathbb{0}}) \ / \ (y^i, d^i) \in v_3\} \cup \{(y^i, \tilde{\mathbb{1}}) \ / \ (y^i, d^i) \in v_1\}$$

$$w_3 = \{(y^i, \tilde{\mathbb{0}}) \ / \ (y^i, d^i) \in v_2\} \cup \{(y^i, \tilde{\mathbb{1}}) \ / \ (y^i, d^i) \in v_3\}$$

Note that $y^0 \notin v_1$, $y^1 \notin v_2$ and v_3 can be empty. Clearly $a^{(0,k_0)} \neq a^{(1,k_1)}$ implies that $b^{(0,k_0)} \neq b^{(1,k_1)}$ or $\exists q,\ x_q^{(0,k_0)} \neq x_q^{(1,k_1)}$ $(1 \leq q \leq m)$. In both cases

$$\wr u \wr = \lambda \mathtt{F}^{\sigma}.\mathtt{gif}\ \mathtt{T}\ \left(\wr w_2 \wr \mathtt{F}\right)\ \left(\wr w_3 \wr \mathtt{F}\right)\ (\wr v_1 \wr \mathtt{F})\ (\wr v_3 \wr \mathtt{F})\ (\wr v_2 \wr \mathtt{F})$$

where $\mathtt{T}$ is the open term defined as follows.

(i) If $b^{(0,k_0)} \neq b^{(1,k_1)}$ then $w_\star = \{(b^{(0,k_0)}, \tilde{\mathtt{0}})\ ,\ (b^{(1,k_1)}, \tilde{\mathtt{1}})\}$ is definable, by induction on RANK. Let $\mathtt{T}$ be the open term $\left(\wr w_\star \wr(\mathtt{F}\ \wr z_1 \wr ... \wr z_m \wr)\right)$.

(ii) Otherwise $b^{(0,k_0)} = b^{(1,k_1)}$ and there is q such that $x_q^{(0,k_0)} \neq x_q^{(1,k_1)}$ $(1 \leq q \leq m)$. Without loss of generality, there is a token $c_q^0 \in x_q^{(0,k_0)}$ such that $c_q^0 \notin x_q^{(1,k_1)}$. If such a token does not exist, it is sufficient to exchange (y^0, d^0) and (y^1, d^1). Hence $z_q^{c_q^0} = \{(\{\tilde{\mathtt{0}}\}, c_q^0)\} \cup \{(\emptyset, c)\ /\ c \in z_q \text{ and } c \neq c_q^0\}$ is a clique definable by induction, so $\mathtt{T}$ is the open term

$$(\mathtt{F}\ \wr z_1 \wr ... \wr z_m \wr) \doteq^\iota \wr b^{(0,k_0)} \wr \ \mathtt{and}\ \ \mathtt{strict?}(\lambda \mathtt{x}^\iota.\mathtt{F} \wr z_1 \wr ... \wr z_{q-1} \wr \left(\wr z_q^{c_q^0} \wr \mathtt{x}\right) \wr z_{q+1} \wr ... \wr z_m \wr).$$

Informally, if the set of constraints $y^j = \{a^{(j,0)}, ..., a^{(j,n_j)}\}$ is satisfied by $\mathtt{F}^\sigma$ then $\wr u \wr$ must return d^j. The constraint $a^{(0,k_0)} = (x_1^{(0,k_0)}, ..., x_m^{(0,k_0)}, b^{(0,k_0)})$ means that we must check that $(\mathtt{F}\ \wr x_1^i \wr ... \wr x_m^i \wr) \doteq^\iota \wr b^i \wr)$ and fulfils some minimal conditions. The cliques y^j are pairwise incoherent by the Lemma 6.2, in fact given an input only one integer d^j can be the result of $\wr u \wr$. Without loss of generality, we can assume that y^0, y^1 contain respectively the incoherent tokens $a^{(0,k_0)}$ and $a^{(1,k_1)}$ having the shape $a^{(0,k_0)} = (x_1^{(0,k_0)}, ..., x_m^{(0,k_0)}, b^{(0,k_0)})$ and $a^{(1,k_1)} = (x_1^{(1,k_1)}, ..., x_m^{(1,k_1)}, b^{(1,k_1)})$. Since $x_i^{(0,k_0)}$ and $x_i^{(1,k_1)}$ are pairwise coherent by Corollary 6.3, the cliques $z_i = x_i^{(0,k_0)} \cup x_i^{(1,k_1)} \in \mathsf{Cl}(\llbracket \mu_i \rrbracket)$ are more defined than $x_i^{(0,k_0)}$ and $x_i^{(1,k_1)}$. Hence $(\mathtt{F}\ \wr z_1 \wr ... \wr z_m \wr)$ is defined, whenever either $(\mathtt{F}\ \wr x_1^{(1,k_1)} \wr ... \wr x_m^{(0,k_0)} \wr)$ is defined or $(\mathtt{F}\ \wr x_1^{(1,k_1)} \wr ... \wr x_m^{(1,k_1)} \wr)$ is defined. In case $b^{(0,k_0)} \neq b^{(1,k_1)}$ the evaluation of $\left(\wr (b^{(0,k_0)}, \tilde{\mathtt{0}}), (b^{(1,k_1)}, \tilde{\mathtt{1}}) \wr (\mathtt{F}\ \wr z_1 \wr ... \wr z_m \wr)\right)$ allows to discriminate between the constraints y^0, y^1. In case $b^{(0,k_0)} = b^{(1,k_1)}$ without loss of generality there is a token in $c_q^0 \in x_q^{(0,k_0)}$ which is not used by a $\mathtt{F}^\sigma$ satisfying the constraint y^1. The evaluation of $\mathtt{T}$ defined in (*ii*) allows to discriminate between the constraints y^0, y^1, in this latter case. Inductively, the cliques w_2, w_3 give to the `gif` operator sufficient information in order to choose the proper conditional-branches (between the three rightmost branch) on which to forward the evaluation. Each of those branches verifies its respective stable minimal constraints.

♦ Suppose $\mathrm{RANK}(\sigma) \geq 2$ and $k \geq 2$, thus $\sigma = \tau_1 \rightarrowtail ..... \rightarrowtail \tau_k \rightarrowtail \iota$.

- If $u = \emptyset$ then $\wr u \wr = \Omega_\sigma$.
- Suppose $u = \{(y_1; ...; y_k; d)\}$ where $y_j \in \llbracket \tau_j \rrbracket$ $(1 \leq j \leq k)$. Thus

$$\wr u \wr = \lambda \mathtt{z}_1^{\tau_1} ... \mathtt{z}_k^{\tau_k}.\mathtt{if}\left(\wr (y_1, 0) \wr \mathtt{z}_1 \ \mathtt{and}\ \ .... \ \mathtt{and}\ \ \wr (y_k, 0) \wr \mathtt{z}_k\right) \wr d \wr\ \Omega_\iota$$

since $(y_j, 0) \in \llbracket \tau_j \rightarrowtail \iota \rrbracket$ $(1 \leq j \leq k)$ is definable by induction on RANK.

- Suppose $u = \{e_0, ..., e_n\}$ where $n \geq 1$, $e_i = (y_1^i; ...; y_k^i; d^i)$ and $y_j^i \llbracket \tau_j \rrbracket$ $(i \leq n,$ $1 \leq j \leq k)$. There exists h such that $y_h^0 \cup y_h^1 \notin \mathsf{Cl}(\llbracket \tau_h \rrbracket)$ by Lemma 6.2 and, in

particular, there exist $a^{(0)} \in y_h^0$ and $a^{(1)} \in y_h^1$ such that $a^{(0)} \smile_{Y_h} a^{(1)}$. Therefore

$$u_1 = \{(y_1^i; ...; y_m^i; d^i) \in u \ / \ a^{(0)} \in y_h^i \in \mathsf{Cl}(Y_h)\}$$
$$u_2 = \{(y_1^i; ...; y_m^i; d^i) \in u \ / \ a^{(1)} \in y_h^i \in \mathsf{Cl}(Y_h)\}$$
$$u_3 = u - (u_1 \cup u_2) = \{(y_1^i; ...; y_m^i; d^i) \in u \ / \ a^{(0)}, a^{(1)} \notin y_h^i\}$$
$$w_1 = \{(\{a^{(0)}\}, \tilde{0})\} \cup \{(\{a^{(1)}\}, \tilde{1})\}$$
$$w_2 = \{(y_1; ...; y_m; \tilde{0}) \ / \ \exists (y_1; ...; y_m; b) \in u_3\} \cup \{(y_1; ..., y_m; \tilde{1}) \ / \ \exists (y_1; ...; y_m; b) \in u_1\}$$
$$w_3 = \{(y_1; ...; y_m, \tilde{0}) \ / \ \exists (y_1; ...; y_m; b) \in u_2\} \cup \{(y_1; ...; y_m; \tilde{1}) \ / \ \exists (y_1; ...; y_m; b) \in u_3\}$$

are cliques by Lemma 6.2, and they are definable by induction. Hence,

$$\wr u \int = \lambda \mathtt{z}_1^{\tau_1} ... \mathtt{z}_k^{\tau_k}.\mathtt{gif}\ (\wr w_1 \int \mathtt{z}_h)\,(\wr w_2 \int \mathtt{z}_1 ... \mathtt{z}_m)\,(\wr w_3 \int \mathtt{z}_1 ... \mathtt{z}_m)$$
$$(\wr u_1 \int \mathtt{z}_1 ... \mathtt{z}_m)\,(\wr u_3 \int \mathtt{z}_1 ... \mathtt{z}_m)\,(\wr u_2 \int \mathtt{z}_1 ... \mathtt{z}_m).$$

□

Note that at first order types (of the shape $\tau_1 \rightarrowtail ... \rightarrowtail \tau_n \rightarrowtail \iota$ where $\tau_k = \iota$ for all k), all finite elements are definable from `gor` alone (no need for `strict`?).

The definability implies the completeness as shown in the next theorem.

Theorem 8.6 *The stable models are complete for $\mathcal{S}t\mathcal{P}CF$.*

Proof. It is easy to see that, if $\mathtt{M}, \mathtt{N}$ are two open terms of $\mathcal{S}t\mathcal{P}CF$ such that $\mathtt{M} \nsim_\sigma \mathtt{N}$ and $\mathrm{FV}(\mathtt{M}) \cup \mathrm{FV}(\mathtt{N}) \subseteq \{\mathtt{x}_1, ..., \mathtt{x}_n\}$ then $\lambda \mathtt{x}_1 ... \mathtt{x}_n.\mathtt{M} \nsim_\tau \lambda \mathtt{x}_1 ... \mathtt{x}_n.\mathtt{N}$ for some τ. Thus without loss of generality only closed terms will be considered. Let $\mathtt{M}, \mathtt{N}$ be two closed terms of $\mathcal{S}t\mathcal{P}CF$ such that $\vdash M : \sigma$ and $\vdash N : \sigma$, while $\mathtt{M} \nsim_\sigma \mathtt{N}$.
Let $\sigma = \tau_1 \rightarrowtail ..... \rightarrowtail \tau_m \rightarrowtail \iota$ for some $m \geq 0$ and without loss of generality assume that there is $a = (x_1; ...; x_m; b)$ where $x_j \in [\![\tau_j]\!]$ for all j, such that $a \in [\![\mathtt{M}]\!]$ but $a \notin [\![\mathtt{N}]\!]$. There are closed terms $\wr x_j \int$ having x_j as interpretation for all j by Lemma 8.5. Hence by interpretation $[\![\mathtt{M} \wr x_1 \int ..... \wr x_m \int]\!] = [\![\wr b \int]\!]$ while, on the other hand, $[\![\mathtt{N} \wr x_1 \int ..... \wr x_m \int]\!] = \emptyset \neq [\![\wr b \int]\!]$, for some $b \in \mathbb{N}$. Therefore $M \not\approx N$, since by Corollary 7.4, both $\mathtt{M} \wr x_1 \int ..... \wr x_m \int \Downarrow_e \wr b \int$ and $\mathtt{N} \wr x_1 \int ..... \wr x_m \int \Uparrow_e$, and the proof is done. □

Corollary 8.7 *The stable models are fully abstract for $\mathcal{S}t\mathcal{P}CF$.*

Proof. By Theorems 7.5 and 8.6. □

Therefore $\sim_\sigma$ and $\approx_\sigma$ are the same relation on programs of $\mathcal{S}t\mathcal{P}CF$.

9 Conclusions, Open Questions and Future Works

First of all, note that the operator gor is Scott-continuous, already definable in the language $\mathcal{PCF}$+por (called both $\mathcal{PCF}^+$ and $\mathcal{PCFP}$ in literature). Without loss of generality, let por be the operator of Example 3.5, page 10. If not is the operator defined on page 26 and pand x y ≡ not (por (not x)(not y)) (i.e. the parallel-and) then it is easy to check that gor x y z can be defined as

if (pand x (not y)) $\tilde{0}$ (if (pand y (not z)) $\tilde{1}$ (if (pand z (not x)) $\tilde{2}$ Ω_ι)).

But strict? is not Scott-continuous, since it is not monotone with respect to the extensional order. Hence strict? cannot be defined in $\mathcal{PCF}$+gor which contains only Scott-continuous functions (which are closed under composition).
On the other hand strict? is strongly stable, while gor is not. Hence gor cannot be defined in $\mathcal{PCF}$+strict? that contains only strongly stable functions. The same conclusion can be obtained in a syntactical way. It is easy to give an operational semantic to $\mathcal{PCF}$+gor through a $\mathcal{PCF}$-like rewrite system. The results of [37] assure that no non-estensionally-monotone operator can live in such language.
Therefore gor and strict? are independent.

It is well-known that Scott-domains contain elements that do not correspond to effective operators; this question is tackled in [53,54] and overcome via the notion of effectively given domains. An element of a Scott-domain is *computable* whenever it is the least upper bound of a recursively enumerable set of finite elements of the considered domains. In order to define all computable elements of Scott-domains a further operator has been added to $\mathcal{PCF}$+por [15,55]. For the sake of simplicity, an existential operator ∃ of type $(\iota \rightarrowtail \iota) \rightarrowtail \iota$ will be considered here. Let ∃M be a "well-typed" term, and let Ω_ι denote a divergent term of type ι. In an informal way, the evaluation of ∃M is (by using the notation introduced in page 5)

- if eval(M$\tilde{n}$) = $\tilde{0}$ for some numerals $\tilde{n}$ then eval(∃M) = $\tilde{0}$,
- if eval(MΩ_ι) = $\widetilde{m+1}$ for some numerals $\tilde{m}$ then eval(∃M) = $\tilde{1}$,
- undefined otherwise.

A model is *universal* for a language when every computable element (in the interpretation of a type) is definable by a closed term of the language [22]. Scott-domains form a universal model (via the standard interpretation) for the language $\mathcal{PCF}$+por +∃ (also called $\mathcal{PCF}^{++}$). Similar results can be found in [56,22,57] for modifications of the language $\mathcal{PCF}$.
Notions of computable elements of stable models have been introduced in [58,59]. I find it plausible that stable models (via the standard interpretation) give a universal model of $\mathcal{StPCF}$, and I am working on a proof of this conjecture.

The question concerning the relationship (full abstraction and universality) between $\mathcal{StPCF}$ (or a variation of it) with the bidomains of Berry [4] is still open.

Further questions arise in the study of the *higher-type computability* [28,11].
Note that the operator strict? (with the informal meaning given in page 5) cannot be added in an effective way to $\mathcal{PCF}^+$ (as essentially proved in [11]). In fact, it is evident that if M is a program then

$$\texttt{strict?}(\ \lambda \texttt{x}^\iota.\texttt{por}\ (\texttt{if}\ \texttt{M}\ \tilde{\texttt{0}}\ \tilde{\texttt{0}})\ \texttt{x}\ ) = \begin{cases} \tilde{\texttt{0}} & \text{if } \texttt{M}\uparrow \\ \tilde{\texttt{1}} & \text{if } \texttt{M}\downarrow \end{cases}$$

where $\uparrow$ and $\downarrow$ denote respectively "divergence" and "convergence" of the evaluation. A simple transformation of the previous code-fragment, namely

$$\lambda \texttt{y}^\iota.\texttt{strict?}(\ \lambda \texttt{x}^\iota.\texttt{por}\ (\texttt{if}\ \texttt{y}\ \tilde{\texttt{0}}\ \tilde{\texttt{0}})\ \texttt{x}\ ),$$

is a "halting program" which decides when the evaluation of its argument converges. Therefore strict? and por cannot live together in the same effective programming language. Since $\lambda \texttt{y}^\iota.\texttt{strict?}(\ \lambda \texttt{x}^\iota.\exists(\lambda \texttt{z}^\iota.\texttt{if z x (if y}\ \tilde{\texttt{0}}\ \tilde{\texttt{0}}))\ )$ is another "halting program", the operator strict? cannot also be added (in an effective way) to $\mathcal{PCF}+\exists$. As suggested by a referee, it is conceptually interesting to explore the question of how much parallelism can coexist with strict?. It is plausible that gor represents the maximum effective degree of parallelism that can coexist with strict? (in $\mathcal{StPCF}$).

John Longley[9] noted that there are seemingly natural incomparable notions of higher-type computability. In contrast with the Church's thesis, there is no a maximum "higher-type computational formal system". Informally, $\mathcal{PCF}^{++}$ and $\mathcal{PCF}+\texttt{H}$ form different "higher-type computational formalisms" such that there does not exist a more generous "higher-type computational formal system" that subsumes both of them. The results of this paper give us some further interesting pieces of this jigsaw puzzle.
A *partial type structure*[9] (PTS) $\mathcal{T}$ consists of:

- a set $\mathcal{T}^\sigma$ for each type σ and in particular T^ι is the flat poset of natural numbers,
- for each σ, τ a total "application function" $\cdot^{\sigma\to\tau} : \mathcal{T}^{\sigma\to\tau} \times \mathcal{T}^\sigma \to \mathcal{T}^\tau$.

The partial type structure $\mathcal{T}$ is *extensional*[9] (EPTS) if, for all types σ, τ and all $f, g \in \mathcal{T}^{\sigma\to\tau}$,

$$\forall x \in \mathcal{T}^\sigma,\ \ f \cdot x = g \cdot x \ \ \text{implies}\ \ f = g.$$

Let $\mathcal{T}, \mathcal{U}$ be EPTSs. A *simulation*[9] $s : \mathcal{T} \to \mathcal{U}$ consists of a total relation $s^\sigma \subseteq \mathcal{T}^\sigma \times \mathcal{U}^\sigma$ for each type σ, such that s^ι is the identity relation on the flat poset of natural numbers and for any $f \in \mathcal{T}^{\sigma\to\tau}$, $g \in \mathcal{U}^{\sigma\to\tau}$, $x \in \mathcal{T}^\sigma$, $y \in \mathcal{U}^\sigma$ we have

$$s^{\sigma\to\tau}(f, g)\ \ \text{and}\ \ s^\sigma(x, y)\ \ \text{imply}\ \ s^\tau(f \cdot x, g \cdot y).$$

If there is a simulation $s : \mathcal{T} \to \mathcal{U}$ then we write $\mathcal{T} \leq \mathcal{U}$.

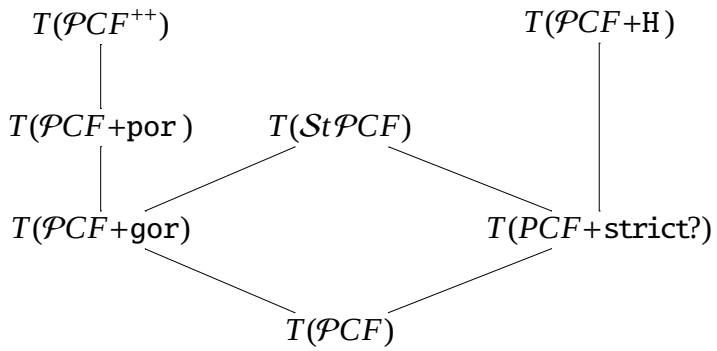
[9] [11], Section 11, Page 77.

It is clear that EPTSs and simulations form a category. It is also easy to see that the only simulation $\mathcal{T} \to \mathcal{T}$ is the identity; so the relation $\leq$ is a partial order on EPTSs. If $\mathcal{L}$ is a programming language then $T(\mathcal{L})$ denotes the type structure corresponding to the term-model of $\mathcal{L}$ built on its operational equivalence.

Proposition 9.1 (i) $T(PCF+\mathtt{gor}) \not\leq T(\mathcal{P}CF+\mathtt{strict?})$
(ii) $T(PCF+\mathtt{strict?}) \not\leq T(\mathcal{P}CF+\mathtt{gor})$

Proof. Minor modifications of the proof of Proposition 11.8 in [11]. □

A formalization of the notion of *effective type structure* corresponding to the expected one is given in Definition 11.2 of [11]. The relationships between some effective type structures may be depicted as follows.



Longley also showed that $T(\mathcal{P}CF^{++})$ is a maximal effective type structure and that $T(\mathcal{P}CF+\mathtt{H})$ is a maximal effectively sequential type structure. Therefore it is a natural question if $\mathcal{S}t\mathcal{P}CF$ is maximal (in some meaningful sense). Clearly this question is related to the previous conjecture that stable models are universal for $\mathcal{S}t\mathcal{P}CF$.

A further marginal question is related to the greatest lower bound for the type structures $T(\mathcal{P}CF^{++})$ and $T(\mathcal{P}CF+\mathtt{H})$. Longley[9] noted that Curien's Third counterexample [9] is an operator definable in $\mathcal{P}CF^{++}$ and $\mathcal{P}CF+\mathtt{H}$ but not in $\mathcal{P}CF$, therefore $T(\mathcal{P}CF)$ is strictly included in the greatest lower bound of $T(\mathcal{P}CF^{++})$ and $T(\mathcal{P}CF+\mathtt{H})$ in the poset of EPTSs. This counterexample can also be programmed in $\mathcal{S}t\mathcal{P}CF$. Therefore, it is natural to ask if the above greatest lower bound is strictly included in $T(\mathcal{S}t\mathcal{P}CF)$.

Acknowledgements

I wish to express my gratitude to Thomas Ehrhard, Simona Ronchi Della Rocca and Giuseppe Rosolini for some useful discussions about topics considered in this paper. I am also grateful to the anonymous referees for all the mistakes and the typos

they pointed out in a preliminary version of this paper and for all the suggestions they put forward.

References

[1] D. S. Scott, A type-theoretical alternative to ISWIM, CUCH, OWHY, Theoretical Computer Science 121 (1–2) (1993) 411–440, a Collection of Contributions in Honour of Corrado Böhm on the Occasion of his 70th Birthday. This paper widely circulated in upublished form since 1969.

[2] S. Abramsky, P. Malacaria, R. Jagadeesan, Full abstraction for PCF, Information and Computation 163 (2) (2000) 409–470, an extended abstract can be found in Theoretical Aspects of Computer Software (Sendai, 1994), Lecture Notes in Computer Science, 789:1-15, Springer-Verlag, Berlin, 1994.

[3] R. Amadio, P.-L. Curien, Domains and Lambda-Calculi, Vol. 46 of Cambridge Tracts in Theoretical Computer Science, Cambridge University Press, Cambridge, 1998.

[4] G. Berry, Stable models of typed λ-calculi, in: G. Ausiello, C. Böhm (Eds.), Fifth International Colloquium Automata, Languages and Programming - - ICALP'78, Udine, Italy, July 17-21, 1978, Vol. 62 of Lecture Notes in Computer Science, Springer-Verlag, 1978, pp. 72–89.

[5] G. Berry, P.-L. Curien, Sequential algorithms on concrete data structures, Theoretical Computer Science 20 (1982) 265–321.

[6] A. Bucciarelli, T. Ehrhard, Sequentiality and strong stability, in: Proceedings of the Symposium on Logic in Computer Science - LICS'91, 1991, pp. 138–145.

[7] A. Bucciarelli, T. Ehrhard, Sequentiality in an extensional framework, Information and Computation 110 (2) (1994) 265–296.

[8] R. Cartwright, P.-L. Curien, M. Felleisen, Fully abstract semantics for observably sequential languages, Information and Computation 111 (2) (1994) 297–401.

[9] P. L. Curien, Categorical Combinators, Sequential Algorithms and Functional Programming (2nd ed.), Birkhäuser, Basel, 1993.

[10] J. M. E. Hyland, L. C.-H. Ong, On full abstraction for PCF: I, II, and III, Information and Computation 163 (2) (2000) 285–408.

[11] J. R. Longley, The sequentially realizable functionals, Annals of Pure and Applied Logic 117 (2002) 1–93.

[12] K. Mulmuley, Full Abstraction and Semantics Equivalence, ACM Doctoral Dissertation Award, The MIT Press, 1987.

[13] P. W. O'Hearn, J. G. Riecke, Kripke logical relations and PCF, Information and Computation 120 (1) (1995) 107–116.

[14] C.-H. L. Ong, Correspondence between operational and denotational semantics: the full abstraction for PCF, in: S. Abramsky, D. Gabbay, T. S. E. Maibaum (Eds.), Handbook of Logic in Computer Science, Vol. 4, Oxford University Press, 1995, pp. 269–356.

[15] G. D. Plotkin, LCF considerd as a programming language, Theoretical Computer Science 5 (1977) 223–225.

[16] R. Milner, Fully abstract models of typed lambda-calculus, Theoretical Computer Science 4 (1977) 1–22.

[17] V. Y. Sazonov, Expressibility of functions in D. Scott's LCF language, Algebra i Logika 15 (3) (1976) 308–330, translation from Russian.

[18] D. S. Scott, Domains for denotational semantics, in: Automata, languages and programming (Aarhus, 1982), Springer-Verlag, Berlin, 1982, pp. 577–613.

[19] A. Stoughton, Interdefinability of parallel operations in PCF, Theoretical Computer Science 1979 (1991) 357–358.

[20] B. Bloom, Can LCF be topped? Flat lattice models of typed λ-calculus, Information and Computation 87 (1-2) (1990) 263–300, a preliminary version was appeared in the Proceedings of Third Annual Symposium on Logic in Computer Science, 5-8 July 1988, Edinburgh, UK.

[21] J. Laird, Full abstraction for functional languages with control, in: Proceedings of the 12th Annual IEEE Symposium on Logic in Computer Science – LICS, IEEE Computer Society Press, Warsaw, Poland, 1997, pp. 58–67.

[22] A. Meyer, Semantical paradigms, in: Proceedings of the Third Annual Symposium on Logic in Computer Science - LICS, 1988, pp. 236–253, notes for an invited lecture with two appendices by Stavros Cosmadakis.

[23] G. Berry, Séquentialité de l'evaluation formelle des λ-expressions, in: Proceedings of the 3rd International Colloquium on Programming, DUNOD, Paris, 1978, pp. 67–80.

[24] A. Bucciarelli, Sequential models of pcf: some contributions to the domain theoretic approach to full abstraction, Ph.D. thesis, Dipartimento di Informatica, Università di Pisa (1993).

[25] A. Bucciarelli, Another approach to sequentiality: Kleene's unimonotone functions., in: S. D. Brookes, M. G. Main, A. Melton, M. W. Mislove, D. A. Schmidt (Eds.), Proceedings of the 9th International Conference of Mathematical Foundations of Programming Semantics, New Orleans, USA, April 7-10, Vol. 802 of Lecture Notes in Computer Science, Springer-Verlag, 1993, pp. 333–358.

[26] A. Bucciarelli, Degrees of parallelism in the continuous type hierarchy, Theoretical Computer Science 177 (1) (1997) 59–71.

[27] A. Bucciarelli, B. Leperchey, Hypergraphs and degrees of parallelism: A completeness result, in: I. Walukiewicz (Ed.), Proceedings of the 7th International Conference of Foundations of Software Science and Computation Structures – FOSSACS 2004, Vol. 2987 of Lecture Notes in Computer Science, Springer-Verlag, Barcelona, Spain, 2004, pp. 58–71.

[28] J. R. Longley, Notions of computability at higher types I, in: R. Cori, A. Razborov, S. Todorcevic, C. Wood (Eds.), Proceedings of the Annual European Summer Meeting of the Association for Symbolic Logic – Logic Colloquium 2000, Vol. 19 of Lecture Notes in Logic, Association for Symbolic Logic, Paris, France, 2000, pp. 32–142.

[29] V. Y. Sazonov, Sequentially and parallelly computable functionals, in: Proceedings of Lambda-Calculus and Computer Science Theory, Vol. 37 of Lecture Notes in Computer Science, Springer-Verlag, Roma, Italia, 1975, pp. 312–318.

[30] M. B. Trakhtenbrot, On representation of sequential and parallel functions., in: J. Becvár (Ed.), Proceedings of the 4th Symposium Mathematical Foundations of Computer Science, Vol. 32 of Lecture Notes in Computer Science, Springer-Verlag, Mariánské Lázne, Czechoslovakia, 1975, pp. 411–417.

[31] M. B. Trakhtenbrot, Relationships between classes of monotonic functions., Theoretical Computer Science 2 (2) (1976) 225–247.

[32] J. Vuillemin, Proof techniques for recursive programs, Ph.D. thesis, Computer Science Department, Stanford University, USA (1973).

[33] G. Berry, Modèles completement adequats et stable du lambda-calcul typé, Ph.D. thesis, Thèse de Doctorat d'État, Université de Paris VII, France (1979).

[34] J.-Y. Girard, π_1^2-Logic: Part I, Dilators, Annals of Mathematical Logic (1981) 75–219.

[35] J.-Y. Girard, The system *F* of variable types, fifteen years later, Theoretical Computer Science 45 (2) (1986) 159–192.

[36] J.-Y. Girard, Y. Lafont, P. Taylor, Proofs and Types, Cambridge Tracts in Theoretical Computer Science, Cambridge University Press, 1989.

[37] T. Jim, A. R. Meyer, Full abstraction and context lemma, SIAM Journal of Computing 25 (3) (1996) 663–696.

[38] B. Bloom, S. Istrail, A. R. Meyer, Bisimulation can't be traced., Journal of the ACM 42 (1) (1995) 232–268.

[39] A. Bucciarelli, T. Ehrhard, A theory of sequentiality, Theoretical Computer Science 113 (2) (1993) 273–291.

[40] T. Ehrhard, Hypercoherence: A strongly stable model of linear logic, in: J.-Y. Girard, Y. Lafont, L. Regnier (Eds.), Proceedings of the Workshop on Advances in Linear Logic, no. 222 in London Mathematical Society Lecture Note Series, Cambridge University Press, Ithaca, New York, 1995, pp. 83–108.

[41] R. Cartwright, M. Felleisen, Observable sequentiality and full abstraction, in: Proceedings of 19th Symposium on Principles of Programming Languages - POPL'92, ACM Sigplan Notices, ACM Press, 2000, pp. 328–342.

[42] C.-H. L. Ong, C. A. Stewart, A curry-howard foundation for functional computation with control, in: Proceedings of the 24th SIGPLAN-SIGACT Symposium on Principles of Programming Languages – POPL, ACM Press, Paris, France, 1997, pp. 215–227.

[43] J. Longley, When is a functional program not a functional program?, in: Proceedings of the fourth International Conference on Functional Programming – ICFP, Vol. 34 of SIGPLAN Notices, ACM Press, Paris, France, 1999, pp. 1–7.

[44] M. Coppo, M. Dezani-Ciancaglini, S. Ronchi Della Rocca, (Semi)-separability of finite sets of terms in Scott's D_∞-models of the λ-calculus, in: G. Ausiello, C. Böhm (Eds.), Fifth International Colloquium on Automata, Languages and Programming - ICALP, Vol. 62 of Lecture Notes in Computer Science, Berlin, Springer-Verlag, Udine, Italy, 1978, pp. 142–164.

[45] G. Kahn, Natural semantics, in: Symposium on Theoretical Aspects of Computer Science - TACS, Vol. 247 of Lecture Notes in Computer Science, 1987, pp. 22–39.

[46] G. D. Plotkin, A structural approach to operational semantics, DAIMI FN-19, Aarhus University, Aarhus, Denmark (Sep. 1981).

[47] M. Bezem, J. W. Klop, R. de Vrijer (Eds.), Term Rewriting Systems – TERESE, Vol. 55 of Cambridge Tracts in Theoretical Computer Science, Cambridge University Press, 2003.

[48] G. Kahn, G. Plotkin, Concrete domains, Theoretical Computer Science 121 (1993) 187–277, first appeared in French as INRIA-LABORIA technical report, 1978.

[49] J. S. Royer, On the computational complexity of Longley's H functional., Theoretical Computer Science 318 (1-2) (2004) 225–241.

[50] J.-Y. Girard, Linear logic, Theoretical Computer Science 50 (1987) 1–102.

[51] D. S. Scott, Continuous lattices, in: F. W. Lawvere (Ed.), Toposes, Algebraic Geometry, and Logic, Vol. 274 of Lecture Notes in Mathematics, Springer-Verlag, Berlin, 1972, pp. 97–136.

[52] L. Paolini, Lambda-theories: some investigations, Ph.D. thesis, Università degli Studi di Genova and Université de la Méditerranée (Aix-Marseille II) (Jan. 2004).

[53] D. Scott, Lectures on a mathematical theory of computation, Technical Monograph PRG-19, Oxford University Computing Laboratory, Programming Research Group, appears in Theoretical foundations of programming methodology : lecture notes of an international summer school, directed by F.L. Bauer, E.W. Dijkstra, and C.A.R. Hoare (Ridel, 1982) (1981).

[54] M. B. Smyth, Effectively given domains., Theoretical Computer Science 5 (3) (1977) 257–274.

[55] V. Y. Sazonov, Degrees of parallelism in computations, in: A. W. Mazurkiewicz (Ed.), Proceedings of 5th Symposium of "Mathematical Foundations of Computer Science", Vol. 45 of Lecture Notes in Computer Science, Springer-Verlag, Gdansk, Poland, 1976, pp. 517–523.

[56] R. Kanneganti, R. Cartwright, M. Felleisen, SPCF: Its model, calculus, and computational power (preliminary version), in: J. W. de Bakker, W. P. de Roever, G. Rozenberg (Eds.), Proceedings of the REX Workshop (Sematics: Foundations and Applications), Lecture Notes in Computer Science, Springer-Verlag, Beekbergen, The Netherlands, 1992, pp. 318–347.

[57] T. Streicher, A universality theorem for PCF with recursive types, parallel-or and exists, Mathematical Structures in Computer Science 4 (1) (1994) 111–115.

[58] A. Asperti, Stability and computability in coherent domains, Information and Computation 86 (2) (1990) 115–139.

[59] A. Gruchalski, Computability on dI-domains, Information and Computation 124 (1) (1996) 7–19.