On the representation of McCarthy's *amb* in the π-calculus

Arnaud Carayol [a,b] Daniel Hirschkoff [b] Davide Sangiorgi [c]

[a] *IRISA, Campus de Beaulieu, 35042 Rennes, France*

[b] *LIP, ENS Lyon, 46 allée d'Italie, 69364 Lyon Cedex 7, France*

[c] *Dipartimento di Scienze dell'Informazione, Universita di Bologna, Via di Mura Anteo Zamboni 7, 40127 Bologna, Italy*

Abstract

We study the encoding of $\lambda^{\parallel}$, the call by name λ-calculus enriched with McCarthy's *amb* operator, into the π-calculus. Semantically, *amb* is a challenging operator, for the fairness constraints that it expresses. We prove that, under a certain interpretation of divergence in the λ-calculus (*weak divergence*), a faithful encoding is impossible. However, with a different interpretation of divergence (*strong divergence*), the encoding is possible, and for this case we derive results and coinductive proof methods to reason about $\lambda^{\parallel}$ that are similar to those for the encoding of pure λ-calculi. We then use these methods to derive the most important laws concerning *amb*. We take bisimilarity as behavioural equivalence on the π-calculus, which sheds some light on the relationship between fairness and bisimilarity.

1 Introduction

The operator of ambiguous choice, *amb*, was first introduced in [McC63], to describe a form of composition of (partial) functions that is liable to return one among several results. [McC63] describes *amb* by giving its main properties. The two most important properties have to do with fairness. One property says that *amb* is *bottom-avoiding*, meaning that the composition of a function with a function that is undefined should return the result of the former function. The other important property says that *amb* behaves as a non-deterministic choice whenever the results computed by the functions being composed are

Email addresses: Arnaud.Carayol@irisa.fr (Arnaud Carayol), Daniel.Hirschkoff@ens-lyon.fr (Daniel Hirschkoff), Davide.Sangiorgi@cs.unibo.it (Davide Sangiorgi).

both defined: either of them may be returned, in an unpredictable way. The usefulness for an operator having the properties of *amb* has come to light for the specification of systems, in particular operating systems, essentially because a form of fair non-determinism is required to merge incoming messages (see [Hen82,Tur90], and also [HO90], that studies *amb* and other nondeterministic operators with respect to this issue). The main reason, however, for our interest in *amb* is that, semantically, 40 years later, *amb* remains a very challenging operator [Las98,Mor98,LM99,Pit01,FK02].

The difficulties introduced by *amb* are clear in $\lambda^{[]}$, the call-by-name λ-calculus enriched with the binary operator [] that is a 'realisation' of McCarthy's *amb*. The two standard approaches to obtain semantics and analysis techniques for λ-calculi are the denotational and the operational ones. The former is based on domain theory; in the latter, applicative bisimilarity [Abr90] is exploited to reason about contextual equivalence. It would be very hard and tedious to prove the laws using a direct application of the definition of contextual equivalence, due to its heavy quantification on contexts. The problem for denotational analyses is that *amb* is not continuous (see [Mor98] for a discussion). The operational approach has been followed by Moran, Lassen and Pitcher, in a series of works [Las98,Mor98,LM99,Pit01]. The question of proving congruence of applicative bisimilarity (or a similar coinductively defined relation, that coincides with or at least gives a good approximation of contextual equivalence) is however still open for $\lambda^{[]}$. The usual technique for proving congruence of applicative bisimilarity in λ-calculi is Howe's [How96], but this technique does not seem to work in presence of *amb* (see [LM99]). Therefore, to prove a set of characteristic laws of *amb*, some 'partial' proof techniques have been developed, in particular in [Mor98,LM99] (these techniques are *partial* in the sense that, taken separately, none of them can be used to derive all the laws – see also Section 4).

In the present paper, we explore an alternative way to give the semantics of $\lambda^{[]}$, via an encoding into the (asynchronous) π-calculus. There were various reasons for carrying out this study. The first reason is the quest for proof methods to reason about languages like $\lambda^{[]}$ that contain operators expressing fairness constraints. The problem of encoding the λ-calculus (as well as parallel and nondeterministic extensions of it) into the π-calculus has been extensively studied – see e.g. [Mil90,San92,BL00,SW01]. In the case of the call-by-name λ-calculus, for example, the π-calculus semantics induces an equivalence on λ-terms that coincides with the classical Lévy-Longo Tree semantics [SW01], which shows an agreement between the π-calculus semantics and standard denotational analyses of the call-by-name λ-calculus. Moreover, bisimulation is the canonical equivalence in the π-calculus, and comes with a well-developed theory, as well as powerful proof techniques that alleviate the task of building bisimulation proofs. One can therefore hope that working in the π-calculus

can help in defining useful bisimulation-based techniques for $\lambda^{\|}$.

A second motivation for this study is expressiveness. The π-calculus has been shown to be a very powerful formalism. We want to understand whether, and under which conditions, the π-calculus can encode an operator as sophisticated as *amb*. We are not aware of other attempts at providing π-calculus encodings of operators that express fairness constraints.

Another motivation is the question of fairness in the π-calculus. While the standard SOS rules of the π-calculus make no reference to fairness, the use of bisimulation or of similar semantical equivalences introduces this kind of property. The definition of a semantics for a fair operator like *amb* is a way to gain a better understanding of this issue. To illustrate this point, consider the π-calculus term $\tau^\omega \,|\, \overline{a}$, where τ^ω represents a process that can perform infinitely many internal actions, $\overline{a}$ is an output at channel a without value exchange, and '|' is the operator of parallel composition. Under bisimulation equivalence, as opposed to, say, testing equivalence, this process is deemed the same as the process $\overline{a}$. One way of interpreting this equality is to say that bisimilarity ignores divergence. However, another way of looking at the equality is to say that bisimilarity encompasses some fairness: under a fair implementation of parallel composition, the left component τ^ω cannot always prevail, hence eventually the action $\overline{a}$ on the right-hand side will be executed. It is precisely this second – and usually neglected – interpretation of bisimilarity that we are addressing, trying to understand its significance on a non-trivial concrete example.

When studying non-deterministic operators like *amb*, contextual equivalence is defined by observing the ability for two terms, in any context, to exhibit convergences and divergences. Two kinds of divergence can be distinguished (see e.g. [NC95]): a computation in which convergence is impossible is a *strong divergence*, while a *weak divergence* corresponds to an infinite computation along which the possibility to converge to a value is never lost. Both forms of divergence arise in $\lambda^{\|}$: first notice that Ω, the usual always diverging term, is strongly divergent. To give an example of a weak divergence, we use the operator of *internal choice*, $\oplus$, that can be encoded in $\lambda^{\|}$ as follows:

$$M \oplus N \quad \stackrel{\mathsf{def}}{=} \quad (K\, M \,[\!]\, K\, N)\, I\,,$$

K and I being the usual combinators for selection and identity. By definition of $\lambda^{\|}$, $M \oplus N$ can nondeterministically evolve to M or N. Now consider the term

$$T \quad \stackrel{\mathsf{def}}{=} \quad \mathsf{Fix}\ \lambda x.\, (x \oplus I)$$

(where Fix is defined as $A\, A$, with $A \stackrel{\mathsf{def}}{=} \lambda xy.\, y\ (x\ x\ y)$). Because of the 'erratic' nature of internal choice, T exhibits a weak divergence, along which convergence to I is repeatedly discarded. In the operational studies of *amb* in the literature, strong and weak divergences are not distinguished.

In this paper, we prove that if we do not distinguish between the two kinds of divergences, there exists no faithful encoding of $\lambda^{\parallel}$ into the π-calculus. By 'faithful', we mean that the encoding should be sound and should mimic the behaviour of $\lambda^{\parallel}$ terms, at least as far as divergence and reduction to values is concerned. This basically means that when taking weak divergences into account, encoding $\lambda^{\parallel}$ in the π-calculus is not possible. This result holds for the π-calculus as well as for any extension of π-calculus with *finitary* operators.

We consequently adopt a contextual equivalence in which only strong divergences are observed, and weak divergences are neglected. This restriction makes sense from the semantical point of view because the difference between strong and weak divergence does not affect the *characteristic laws* of *amb*: we refer here to a set of laws that capture *amb*'s essential properties (these laws are studied for example in [Mor98] — as mentioned above, the original specification of *amb* [McC63] is given in a rather informal way by mentioning a set of behavioural properties). We also show that neglecting weak divergences makes sense from an operational point of view. This is achieved by defining an operational semantics for *amb* in which weakly divergent behaviours have a null probability. The intuition is that weak divergences are 'unlikely' to happen, and can therefore be neglected (a similar argument is already present in [NC95] in a slightly different setting).

Under the strong interpretation of divergence, we show that the encoding of $\lambda^{\parallel}$ into the π-calculus is possible, and we derive results and coinductive proof methods to reason about $\lambda^{\parallel}$ that are similar to those that have been developed for the encodings of pure λ-calculi (see [SW01]). We then use these methods to derive the characteristic laws of McCarthy's *amb*. Using π-calculus-specific proof techniques, the proofs for some of these laws are very simple, in particular those of the two key properties of *amb*, the bottom-avoidance law $M \parallel \Omega \cong_{\mathrm{M}} M$, and the law $V \parallel V' \cong_{\mathrm{M}} V \oplus V'$ (where V and V' are λ-abstractions). We also study the extension of $\lambda^{\parallel}$ with local call-by-value, again showing an encoding into the π-calculus and then using the encoding to derive algebraic laws in the source calculus.

A preliminary version of this work was presented in EXPRESS'03 [CHS03]. This presentation includes full proofs, that were not given in [CHS03], as well as some new material (in particular in 2.1.3 and 3.3.2).

Outline. We present $\lambda^{\parallel}$ and the π-calculus, and establish some preliminary results we need about these calculi in Section 2. In Section 3, we analyse the setting in which we study McCarthy's *amb*, and we give some results about this framework that motivate the study in the next section. In Section 4, we introduce our π-calculus encodings of $\lambda^{\parallel}$, and present a number of applications and developments. We conclude and discuss further research directions

in Section 5.

2 Calculi

This section contains background material. It does also contain some novel results: a new semantics for $\lambda^{\parallel}$ and some new up-to proof techniques for coupled simulation.

2.1 The λ-calculus with Ambiguous Choice

2.1.1 Definition of $\lambda^{\parallel}$

We recall here the definition of $\lambda^{\parallel}$, the call-by-name λ-calculus extended with *amb*.

We suppose we have an infinite set of *variables*, ranged over with $x, y, \ldots$. *Terms* of $\lambda^{\parallel}$, ranged over with $M, N, \ldots$, are given by the following grammar:

$$M \quad \stackrel{\mathsf{def}}{=} \quad x \mid \lambda x.M \mid M_1\, M_2 \mid M_1 \,[]\, M_2\,.$$

Bound and free variables are defined as usual, and we will sometimes write $\lambda x_1 \ldots x_k.\, M$ for $\lambda x_1.\ldots.\lambda x_k.\, M$. A *closed term* is a term that contains no free variable. Substitution (written $M[N/x]$) and α-conversion are defined as usual, and we will work up-to α-conversion. Closed values, ranged over with $V, V', \ldots$, are abstractions. A context, ranged over with $C, C', \ldots$ is a term containing occurrences of a *hole*, written $[\,\cdot\,]$, in it. Given a context C, $C[M]$ denotes the term obtained by replacing the hole with a term M in C. Given M, C is *closing* if $C[M]$ is closed, this terminology being extended to the case where C is closing for several terms.

The following $\lambda^{\parallel}$ terms will be useful below:

$$I \stackrel{\mathsf{def}}{=} \lambda x.\, x \qquad \Omega \stackrel{\mathsf{def}}{=} (\lambda x.\, x\, x)\ (\lambda x.\, x\, x)$$
$$K \stackrel{\mathsf{def}}{=} \lambda x\, y.\, x \qquad \mathsf{Fix} \stackrel{\mathsf{def}}{=} A\, A \ \text{ where } \ A \stackrel{\mathsf{def}}{=} \lambda x\, y.\, y\,(x\, x\, y)\,.$$

2.1.2 Lassen and Moran's operational semantics for $\lambda^{\parallel}$

In [LM99], the operational semantics of $\lambda^{\parallel}$ is defined on *decorated* $\lambda^{\parallel}$ *terms*. These are $\lambda^{\parallel}$ terms in which every occurrence of an *amb* is of the form $M\,{}^{k}[]^{k'}\,N$, where k and k' are natural numbers. [LM99] also defines an operation, written

$$\textsc{Beta}\ (\lambda x.\ M)\ N \rightarrowtail M[N/x] \qquad \textsc{Val}_L\ V\ {}^{m+1}[]^n\ N \rightarrowtail V$$

$$\textsc{Lazy}\ \frac{M \rightarrowtail M'}{M\ N \rightarrowtail M'\ N} \qquad \textsc{Red}_L\ \frac{M \rightarrowtail M'}{M\ {}^{m+1}[]^n\ N \rightarrowtail M'\ {}^{m}[]^n\ N}$$

$$\textsc{Sched}\ M\ {}^{0}[]^{0}\ N \rightarrowtail M\ {}^{m}[]^{n}\ N \quad \text{if } m > 0 \text{ and } n > 0$$

Fig. 1. Operational semantics for decorated $\lambda^{[]}$ terms

---

$\cdot^\sharp$, that decorates all *amb*s in a term with counters set to zero. A decorated term M is *initialised* if $M = M_0^\sharp$ for some non-decorated term M_0.

Definition 1 (Notations for relations) *If $\mathcal{R}$ is a binary relation over elements of a set $\mathbb{S}$, $\mathcal{R}^{-1}$ denotes the inverse of $\mathcal{R}$, while $\mathcal{R}^+$ and $\mathcal{R}^*$ denote the transitive (resp. transitive and reflexive) closures of $\mathcal{R}$. Composition of two relations $\mathcal{R}$ and $\mathcal{S}$ is written $\mathcal{RS}$, and $\mathcal{R}^n$, for $n \geq 1$, stands for the result of composing n times relation $\mathcal{R}$ with itself. $T\,\mathcal{R}$ means that there exists T' such that $T\,\mathcal{R}\,T'$, and $T\,\mathcal{R}^\omega$ stands for the existence of an infinite sequence of elements of $\mathbb{S}$, $T_0 = T, T_1, \ldots$ such that for all i, $T_i\,\mathcal{R}\,T_{i+1}$ (and similarly for $T\mathcal{R}^n$ in the case of finite computations).*

Definition 2 ($\rightsquigarrow$) *Relation $\rightarrowtail$, defined on decorated $\lambda^{[]}$ terms, is given by the rules of Fig. 1 (symmetrical versions of rules $\textsc{Val}_L$ and $\textsc{Red}_L$ are omitted). $\rightarrowtail$ induces a relation $\rightsquigarrow$ on pure $\lambda^{[]}$ terms by setting $M \rightsquigarrow N \stackrel{\mathsf{def}}{=} M^\sharp \rightarrowtail^+ N^\sharp$. We define, for any $n \geq 1$, $\rightsquigarrow^n$ (resp. $\rightsquigarrow^{<n}$) by $M \rightsquigarrow^n N \stackrel{\mathsf{def}}{=} M^\sharp \rightarrowtail^n N^\sharp$ (resp. $M \rightsquigarrow^{<n} N \stackrel{\mathsf{def}}{=} M^\sharp \rightarrowtail^k N^\sharp$ for some $0 < k < n$).*

Intuitively, the natural integers decorating *amb* compositions can be seen as counters that are used to schedule the execution of the terms being composed: in $M\ {}^{k}[]^{k'}\ N$, term M (resp. N) has the 'right' to perform k (resp. k') reduction steps. In order to avoid one of the two components to reduce ad infinitum without letting the other one proceed, a synchronisation happens when (and only when) both counters reach 0, at which time these are updated using *non-null* values (rule $\textsc{Sched}$).

To our knowledge, all existing operational semantics for $\lambda^{[]}$ exploit a form of resource such as these decorations to 'program' *amb*'s behaviour by the means of a scheduler. We propose in 2.1.3 a reduction relation that works directly on unannotated $\lambda^{[]}$ terms and that coincides with $\rightsquigarrow$ (Proposition 6). We first establish the following preliminary results about $\rightarrowtail$ and $\rightsquigarrow$, that will be useful for this characterisation.

Lemma 3 *For any decorated terms P and Q,*

(1) if $P^\sharp \rightarrowtail^+ (\lambda x.N)$ then $(\lambda x.N)$ is initialised;

(2) if $P^\sharp \rightarrowtail^+ (\lambda x.N)\, M$ *then* $(\lambda x.N)\, M$ *is initialised.*

Proof. We simultaneously prove both properties by induction on the length n of $P^\sharp \rightarrowtail^+ (\lambda x.N)$ and $P^\sharp \rightarrowtail^+ (\lambda x.N)\, M$.

Suppose first $n = 1$. In both cases, $P^\sharp = (\lambda x.K)K'$ and $Q = K[K'/x]$. As K and K' are initialised, so is Q.

Suppose now $n > 1$. We first consider the case where $P^\sharp \rightarrowtail^n \lambda x.N$. Let $(M_i)_{i\in[0,n]}$ be a sequence of terms such that $M_0 = P^\sharp$ and $M_n = \lambda x.N$ and for all $i \in [0, n-1]$, $M_i \rightarrowtail M_{i+1}$. We distinguish two cases:

- If for some $j \in [1, n-1]$, M_j is a β-redex then $M_0 \rightsquigarrow^{<n} M_j$ and $M_j \rightsquigarrow M_n$. By applying part 2 of the induction hypothesis to $M_0 \rightsquigarrow^{<n} M_j$, we obtain that M_j is initialised. Then we can apply part 1 of the induction hypothesis to $M_j \rightsquigarrow M_n$ to conclude.
- If none of the $(M_i)_{i\in[1,n-1]}$ is a β-redex then $P = P_1 \,[]\, P_2$ and either $P_1 = \lambda x.N$, or $P_2 = \lambda x.N$, or $P_1 \rightsquigarrow^{<n} \lambda x.N$, or $P_2 \rightsquigarrow^{<n} \lambda x.N$. By applying if necessary the induction hypothesis, we can conclude in all cases that $\lambda x.N$ is initialised.

The case $P^\sharp \rightarrowtail^n (\lambda x.N)M$ is similar to the previous one. □

Lemma 4 *Suppose* $P \rightsquigarrow^n Q$ *for some* $n \geq 2$. *Then:*

- *if* $P = P_1 \,[]\, P_2$ *then*
  - *either* $Q = Q_1 \,[]\, Q_2$, $P_1 \rightsquigarrow^{<n} Q_1$ *and* $P_2 \rightsquigarrow^{<n} Q_2$,
  - *or* $Q = \lambda x.M$ *and we have either* $P_1 \rightsquigarrow^{<n} Q$, *or* $P_2 \rightsquigarrow^{<n} Q$, *or* $P_1 = Q$, *or* $P_2 = Q$;
- *if* $P = MN$ *then*
  - *either* $M \rightsquigarrow^n M'$ *and* $Q = M'N$,
  - *or there exists* R *such that* $P \rightsquigarrow^{<n} R$ *and* $R \rightsquigarrow^{<n} Q$.

Proof. Let $(P_i)_{i\in[0,n]}$ be a sequence of terms such that $P_0 = P^\sharp$, $P_n = Q^\sharp$ and for all $i \in [0, n-1]$, $P_i \rightarrowtail P_{i+1}$. In the following, α_i will stand for the name of the last inference rule used to infer $P_i \rightarrowtail P_{i+1}$. All the cases of this lemma are obtained by examining the sequence $(\alpha_i)_{i\in[0,n-1]}$.

- Suppose that $P = P_1 \,[]\, P_2$. All the α_is are of type Red,Val or Sched.
  - If none of the α_is is of type Val then for each i, $P_i = P_i^1 \,[]\, P_i^2$. Moreover, we have the following equalities: $P_0^1 = P_1$,$P_0^2 = P_2$ and for all $i \in [0, n-1]$ and $j \in \{1, 2\}$, either $P_i^j = P_{i+1}^j$ or $P_i^j \rightarrowtail P_{i+1}^j$. It is straightforward to check that $P_0^1 \rightarrowtail^{n_L} P_n^1$ and $P_0^2 \rightarrowtail^{n_R} P_n^2$, where n_L (resp. n_R) is equal to the number of α_is of type Red_L (resp. Red_R). We claim that n_L and

$$\textsc{Beta}\ (\lambda x.\ M)\ N\ \twoheadrightarrow\ M[N/x]$$

$$\textsc{Lazy}\ \frac{M\ \twoheadrightarrow\ M'}{M\ N\ \twoheadrightarrow\ M'\ N} \qquad \textsc{Trans}\ \frac{M\ \twoheadrightarrow\ M' \qquad M'\ \twoheadrightarrow\ M''}{M\ \twoheadrightarrow\ M''}$$

$$\textsc{Par}\ \frac{M\ \twoheadrightarrow\ M' \qquad N\ \twoheadrightarrow\ N'}{M \parallel N\ \twoheadrightarrow\ M' \parallel N'} \qquad \textsc{Val}_L\ \frac{M\ \twoheadrightarrow\ V \qquad \textbf{or} \qquad M\ =\ V}{M \parallel N\ \twoheadrightarrow\ V}$$

Fig. 2. Operational semantics for $\lambda^{\parallel}$

n_R belong to $[1, n-1]$. Suppose that n_L is null. As P_0 is initialised, α_0 is of type SCHED and $P_1 = P_0^{1\ n+1}\|^{m+1}\ P_0^2$. As $n_L = 0$, we would have $P_n = P_0^{1\ n+1}\|^{k}\ P_n^2$ which contradicts the fact that P_n is initialised. As P_0^j and P_n^j are initialised, we can conclude that $P_0^j \rightsquigarrow^{<n} P_n^j$ for $j \in \{1, 2\}$.

- If one of the α_is is of an instance of rule VAL then it must be α_{n-1} and hence $Q = \lambda x.M$. Using a similar method as above, we can establish that either $P_1 \rightsquigarrow^{<n} Q$, or $P_2 \rightsquigarrow^{<n} Q$, or $P_1 = Q$, or $P_2 = Q$.

- Now suppose that $P = MN$.
  - If one of the α_is is an instance of rule BETA, then let j be the smallest j such that $\alpha_j = \textsc{Beta}$. If $j = 0$ then we take $R = P_1$ and $P \rightsquigarrow^1 R$ and $R \rightsquigarrow^{n-1} Q$. If $j > 0$ then according to Lem. 3, $P_j = P_j^\sharp$, $P_0 \rightsquigarrow^{<n} P_j$ and $P_j \rightsquigarrow^{<n} P_n$.
  - If none of the α_is is an instance of rule BETA then all the α_is are instances of the rule LAZY. For all $i \in [0, n-1]$, $P_i = M_i\, N$ and $M_i \rightarrowtail M_{i+1}$. We have $P_0 = M_0\, N$, $P_n = M_n\, N$ and $M_0 \rightsquigarrow^n M_n$. □

2.1.3 A characterisation of $\rightsquigarrow$

Before analysing the properties of computation in $\lambda^{\parallel}$, we start by characterising $\rightsquigarrow$ using a simpler reduction relation, written $\twoheadrightarrow$.

Definition 5 ($\twoheadrightarrow$) *Relation* $\twoheadrightarrow$ *is given by the rules of Figure 2, where the symmetrical version of* $\textsc{Val}_L$ *is omitted.*

Note that $\twoheadrightarrow$ is defined directly on $\lambda^{\parallel}$ terms. In defining $\twoheadrightarrow$, we capture the transitive, *non-reflexive* closure of the underlying reduction relation. In rule PAR both components of an *amb* are allowed to evolve. Rules $\textsc{Val}_L$, $\textsc{Val}_R$ make the choice between components of an *amb*, when one of the branches converges.

Proposition 6 $\twoheadrightarrow = \rightsquigarrow$.

Proof. We prove both inclusions.

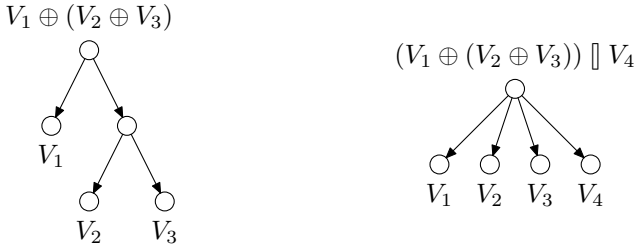

Fig. 3. Derivations trees of $V_1 \oplus (V_2 \oplus V_3)$ and $V_1 \oplus (V_2 \oplus V_3) \,[\!]\, V_4$ for $\twoheadrightarrow$

- From left to right: we prove by induction on the derivation tree of $P \twoheadrightarrow Q$ and by case analysis on the last rule being used that $P \twoheadrightarrow Q$ implies $P \rightsquigarrow Q$.

  We only present the most interesting case, when the last rule being used is PAR, to infer $P_1 \,[\!]\, P_2 \twoheadrightarrow Q_1 \,[\!]\, Q_2$. By applying the induction hypothesis to the premises $P_1 \twoheadrightarrow Q_1$ and $P_2 \twoheadrightarrow Q_2$, we have $P_1 \rightsquigarrow Q_1$ and $P_2 \rightsquigarrow Q_2$. Let n_1 and n_2 be the strictly positive integers such that $P_1 \rightsquigarrow^{n_1} Q_1$ and $P_2 \rightsquigarrow^{n_2} Q_2$. It is straightforward to check that:

$$P^\sharp = P_1^\sharp \,{}^0[\!]^0\, P_2^\sharp \;\rightarrowtail_{\text{SCHED}}\; P_1^\sharp \,{}^{n_1}[\!]^{n_2}\, P_2^\sharp \rightarrowtail^{n_1} Q_1^\sharp \,{}^0[\!]^{n_2}\, P_2^\sharp \rightarrowtail^{n_2} Q_1^\sharp \,{}^0[\!]^0\, Q_2^\sharp = Q^\sharp\,.$$

  Rule SCHED can be used because n_1 and n_2 are non zero.
- From right to left: we prove by induction on the length of the derivation $P \rightsquigarrow Q$ that $P \rightsquigarrow Q$ implies $P \twoheadrightarrow Q$.

  $n = 1$**.** In this case, we have $P^\sharp \rightarrowtail Q^\sharp$. By a straightforward induction on the structure of P, we can prove that $P^\sharp = (\lambda x.M)N_1 \ldots N_k$ for some $k \geq 1$ and $Q^\sharp = M[x/N_1]N_2 \ldots N_k$ (if $k \geq 2$) or $Q^\sharp = M[x/N_1]$ (if $k = 1$). In all cases, we have $P \twoheadrightarrow Q$.

  $n \geq 2$**.** We proceed by case analysis on the structure of P and we distinguish the same cases as in Lemma 4. All cases are trivial. □

2.1.4 Discussion about amb's properties

Let us make some observations about the operational semantics defined by $\twoheadrightarrow$. If we consider the terms given on Figure 3 (where the V_is are values), we see that, according to $\twoheadrightarrow$, *amb* composition makes trees degenerate and loose their branching structure. Thus, in some sense, $\twoheadrightarrow$ misses some choices along $\lambda^{[\![]}$ computations. This lack of precision can be seen as a drawback for defining a bisimulation-based equivalence for $\lambda^{[\![]}$, since such an equivalence usually exploits an accurate analysis of the decisions that are made along computation. Indeed, bisimulation equivalences are known to be more discriminating than trace equivalence, intuitively because they are based on trees and not on single executions (traces). In fact, on all terms of the form $M \,[\!]\, V$, $\twoheadrightarrow$ defines a big step semantics: such a term can only converge (immediately) to a value. Relation $\twoheadrightarrow$, together with the induced notions of convergence and divergence, thus appears to be too imprecise to allow one to derive a suitable notion of

bisimulation. We shall return on this observation below.

amb **vs. other operators.** The setting provided by $\twoheadrightarrow$ allows us to compare *amb* with other existing parallel or nondeterministic operators, and to illustrate *amb*'s expressiveness. The simplest form of choice is given by $\oplus$, the operator of *internal choice*. It can be defined in $\lambda^{[\!]}$ by

$$M \oplus N \stackrel{\mathsf{def}}{=} (K\, M \,[\!]\, K\, N)\; I\,.$$

We have that $M \oplus N \twoheadrightarrow M$ and $M \oplus N \twoheadrightarrow N$, which corresponds to the expected behaviour of internal choice, i.e., every branch of $\oplus$ may be selected, independently from other considerations.

Countable choice may be implemented in $\lambda^{[\!]}$ as a term that can nondeterministically reduce to $\lambda x_1 \ldots x_n.\, I$, for any $n \geq 0$. The simplest way to achieve this is by extending $\lambda^{[\!]}$ with a form of local call by value, brought by the traditional $\mathsf{let}\ldots\mathsf{in}$ construction (see 4.5 for a discussion on local call by value). The corresponding term is then the following:

$$R \stackrel{\mathsf{def}}{=} \mathsf{Fix}\,\lambda z.\; ((\mathsf{let}\; x = z\; \mathsf{in}\; \lambda y.x) \,[\!]\, I)\;.$$

We remark that $R \twoheadrightarrow (\mathsf{let}\; x = R\; \mathsf{in}\; \lambda y.x) \,[\!]\, I$, and R obviously cannot diverge (as a consequence of the definition of *amb*). We can then show by induction that $R \twoheadrightarrow \lambda x_1 \ldots x_n.\, I$ for any $n \geq 0$. We shall see a similar construction in the proof of Theorem 19. The term used in that proof shows that the definition of R could be adapted to a calculus without $\mathsf{let}\ldots\mathsf{in}$ construct, but we have preferred this presentation here for the sake of clarity.

The specification of the *parallel or* construct is based on a property of bottom avoidance, saying that if one of the two branches converges to the value true, then the whole term converges to true. In the case where the two branches converge to false, then the whole term does so, and otherwise the computation diverges. Considering that the possible outcomes of the computation of a boolean are true, false, or a divergence, there is no point in giving properties about fairness in the specification of parallel or. The following definition implements an operator having the requested properties in $\lambda^{[\!]}$, given an $\mathsf{if}\ldots\mathsf{then}\ldots\mathsf{else}$ construct for case analysis on booleans:

$$M\,\mathsf{por}\,N \stackrel{\mathsf{def}}{=} (\mathsf{if}\; M\; \mathsf{then\; true\; else}\; N) \,[\!]\, (\mathsf{if}\; N\; \mathsf{then\; true\; else}\; \Omega)\,.$$

This suggests that among existing concurrent and non-deterministic operators, *amb* is very expressive.

2.1.5 Observational equivalence in $\lambda^{\mathbb{I}}$

We now use $\twoheadrightarrow$ to define observational equivalence as in [LM99], by analysing the possibility for two terms to converge and to diverge.

Definition 7 ($\Downarrow$ and $\Uparrow$) *A term M is* convergent*, written $M \Downarrow$, if there exists a value V s.t. $M \twoheadrightarrow V$ or $M = V$. M is* divergent*, written $M \Uparrow$, if $M \twoheadrightarrow^{\omega}$.*

Definition 8 (Observational equivalence, using weak divergence) *Two terms M and N are* observationally equivalent*, written $M \cong_{M} N$, iff for any closing context C:*

$$(C[M] \Downarrow \iff C[N] \Downarrow) \quad \textit{and} \quad (C[M] \Uparrow \iff C[N] \Uparrow) .$$

2.2 The Asynchronous π-calculus

We suppose that we have an infinite set of *names*, also called *channels*, over which we range with small letters: $a, b, \ldots, x, y, \ldots$. For the sake of the Asynchronous π-calculus (in short, $A\pi$) encoding of Section 4, we shall translate a $\lambda^{\mathbb{I}}$ variable using a π-calculus name, and we suppose that there is an injection from variables to names so that we can keep letter x to refer to the encoding of a variable x. (Possibly empty) name tuples are ranged over with $\tilde{x}, \tilde{y}, \ldots$. $A\pi$ terms, to which we shall refer simply as *processes*, are ranged over using $P, Q, \ldots$, and are defined as follows:

$$P \quad \stackrel{\mathsf{def}}{=} \quad \mathbf{0} \;\Big|\; P_1 | P_2 \;\Big|\; !P \;\Big|\; \boldsymbol{\nu} x\ P \;\Big|\; x(\tilde{y}).P \;\Big|\; \bar{x}\langle \tilde{y} \rangle .$$

$\mathbf{0}$ is the inactive process, and $|$ is parallel composition. The replicated process $!P$ represents an unbounded number of copies of P put in parallel. The restriction operator $\boldsymbol{\nu}$ declares a name which is private to a process. $\bar{x}\langle \tilde{y} \rangle$ stands for the output particle resulting from the (asynchronous) emission of tuple $\tilde{y}$ on channel x, while $x(\tilde{y}).P$ is an input process listening on channel x, in which $\tilde{y}$ are parameters to be instantiated upon communication. We sometimes write $\boldsymbol{\nu} x, y\ P$ for $\boldsymbol{\nu} x\ \boldsymbol{\nu} y\ P$. Bound names in processes are defined by saying that the input and restriction operators are binding. Contexts in $A\pi$ are defined along the lines of $\lambda^{\mathbb{I}}$ contexts.

The operational semantics for $A\pi$ is defined by judgements of the form $P \xrightarrow{\mu} P'$, meaning that P is liable to evolve to P' by performing action μ. *Actions* are defined as follows (bound names in actions are defined by saying that restriction is binding):

$$\mu \quad \stackrel{\mathsf{def}}{=} \quad a(\tilde{x}) \;\Big|\; \boldsymbol{\nu} \widetilde{x}\ \bar{a}\langle \tilde{y} \rangle_{\widetilde{x} \subseteq \widetilde{y}} \;\Big|\; \tau .$$

$$\textsc{Out}\ \bar{a}\langle\tilde{x}\rangle \xrightarrow{\bar{a}\langle\tilde{x}\rangle} \mathbf{0} \qquad \textsc{In}\ a(\tilde{x}).P \xrightarrow{a(\tilde{x})} P$$

$$\textsc{Res}\ \frac{P \xrightarrow{\mu} P'}{\boldsymbol{\nu} a\ P \xrightarrow{\mu} \boldsymbol{\nu} a\ P'}\ a \notin \mathsf{n}(\mu) \qquad \textsc{Open}\ \frac{P \xrightarrow{\boldsymbol{\nu}\widetilde{d}\,\bar{a}\langle\tilde{b}\rangle} P'}{\boldsymbol{\nu} c\ P \xrightarrow{\boldsymbol{\nu} c,\tilde{d}\,\bar{a}\langle\tilde{b}\rangle} P'}\ c \in \tilde{b} \setminus \widetilde{d}\ \text{ and }\ a \neq c$$

$$\textsc{Rep}\ \frac{P \mid !\,P \xrightarrow{\mu} P'}{!\,P \xrightarrow{\mu} P'} \qquad \textsc{Par}_L\ \frac{P \xrightarrow{\mu} P'}{P \mid Q \xrightarrow{\mu} P' \mid Q}\ \mathsf{bn}(\mu) \cap \mathsf{fn}(Q) = \emptyset$$

$$\textsc{Close}_L\ \frac{P \xrightarrow{a(\tilde{c})} P' \qquad Q \xrightarrow{\boldsymbol{\nu}\widetilde{d}\,\bar{a}\langle\tilde{b}\rangle} Q'}{P \mid Q \xrightarrow{\tau} \boldsymbol{\nu}\tilde{d}\ (P'\{\tilde{b}/\tilde{c}\} \mid Q')}\ \tilde{d} \cap \mathsf{fn}(P) = \emptyset$$

Fig. 4. Asynchronous π-calculus – operational semantics

In a bound output action $\boldsymbol{\nu}\widetilde{x}\ \bar{a}\langle\tilde{y}\rangle$, $\widetilde{x}$ represents a *set* of names, i.e. we work modulo rearrangement of names. Similarly, a condition of the form $\widetilde{x} \subseteq \widetilde{y}$ should be understood as the inclusion between the corresponding name sets.

The rules for the labelled transition system are presented on Fig. 2.2 (symmetrical versions of rules $\textsc{Par}_L$ and $\textsc{Close}_L$ are omitted). We furthermore introduce the following notations: $\Rightarrow \stackrel{\mathsf{def}}{=} (\xrightarrow{\tau})^*$, $\xrightarrow{\hat{\mu}} \stackrel{\mathsf{def}}{=} \xrightarrow{\tau}$ or $=$ if $\mu = \tau$, $\xrightarrow{\hat{\mu}} \stackrel{\mathsf{def}}{=} \xrightarrow{\mu}$ otherwise, and $\stackrel{\hat{\mu}}{\Longrightarrow} \stackrel{\mathsf{def}}{=} \Rightarrow\xrightarrow{\hat{\mu}}\Rightarrow$.

Structural congruence, $\equiv$, is introduced to capture some basic structural properties of processes. It is defined by the following rules:

$$P|Q \equiv Q|P \qquad P|(Q|R) \equiv (P|Q)|R \qquad P|\mathbf{0} \equiv P \qquad \boldsymbol{\nu} a\ \mathbf{0} \equiv \mathbf{0}$$

$$\boldsymbol{\nu} a\ \boldsymbol{\nu} b\ P \equiv \boldsymbol{\nu} b\ \boldsymbol{\nu} a\ P \qquad P|\boldsymbol{\nu} a\ Q \equiv \boldsymbol{\nu} a\ (P|Q) \ \text{ if } a \notin \mathsf{fn}(P)$$

$$!P \equiv !P|P \qquad !!P \equiv !P \qquad !(P|Q) \equiv !P \mid !Q \qquad !\mathbf{0} \equiv \mathbf{0}$$

Structural congruence is needed in the statement of the following result, which will be useful for a proof below:

Proposition 9 ($\xrightarrow{\tau}/_{\equiv}$ is finitely branching, [SW01]) *Given a process P, there is, up to structural congruence, a finite number of processes P' such that $P \xrightarrow{\tau} P'$.*

2.2.1 Behavioural equivalences and preorders

We shall use a rather wide spectrum of equivalences and preorders in $A\pi$, according to the needs of our proofs about $\lambda^{\mathbb{I}}$. We define these below.

Definition 10 (Behavioural equivalences and preorders, $\approx, \leftrightarrows, \lesssim$)

- *A relation $\mathcal{R}$ on processes is a* weak simulation *if $P\,\mathcal{R}\,Q$ and $P \xrightarrow{\mu} P'$ imply that there exists Q' such that $Q \stackrel{\hat{\mu}}{\Longrightarrow} Q'$ and $P'\,\mathcal{R}\,Q'$.*
- *A weak bisimulation is a symmetric weak simulation.* Weak bisimilarity*, written $\approx$, is the greatest weak bisimulation.*
- *A* coupled bisimulation *is a pair of simulations $(\mathcal{S}_1, \mathcal{S}_2^{-1})$ such that:*
  - *$P\,\mathcal{S}_1\,Q$ then there exists Q' s.t. $Q \Rightarrow Q'$ and $P\,\mathcal{S}_2\,Q'$;*
  - *$P\,\mathcal{S}_2\,Q$ then there exists P' s.t. $P \Rightarrow P'$ and $P'\,\mathcal{S}_1\,Q$.*

  Two processes P and Q are coupled bisimilar*, written $P \leftrightarrows Q$, if there exists a coupled bisimulation $(\mathcal{S}_1, \mathcal{S}_2^{-1})$ such that $P\,\mathcal{S}_1\,Q$ and $P\,\mathcal{S}_2\,Q$.*
- *A relation $\mathcal{R}$ is an* expansion *if $P\,\mathcal{R}\,Q$ entails:*
  - *if $P \xrightarrow{\mu} P'$, then there exists Q' s.t. $Q \stackrel{\hat{\mu}}{\Longrightarrow} Q'$ and $P'\,\mathcal{R}\,Q'$;*
  - *if $Q \xrightarrow{\mu} Q'$, then there exists P' s.t. $P \stackrel{\hat{\mu}}{\Longrightarrow} P'$ and $P'\,\mathcal{R}\,Q'$.*

  The greatest expansion relation is written $\lesssim$, and $\gtrsim$ stands for $(\lesssim)^{-1}$.

Definition 11 ($\cong_\pi$) *Given a name p, $P \Downarrow_p$ stands for $P \Rightarrow \xrightarrow{\boldsymbol{\nu}\widetilde{x}\,\overline{p}\langle\tilde{y}\rangle}$ for some $\widetilde{x}$ and $\tilde{y}$. P and Q are* observationally equivalent*, written $P \cong_\pi Q$, iff*
(for all C and p, $C[P] \Downarrow_p \;\Leftrightarrow\; C[Q] \Downarrow_p$) and $(P \Rightarrow\approx \mathbf{0} \;\Leftrightarrow\; Q \Rightarrow\approx \mathbf{0})$.

The definition of $\cong_\pi$ follows the pattern of $\cong_{\mathrm{M}}$ in $\lambda^{\mathbb{I}}$ (Definition 8, see also Definition 22 below). In $A\pi$, observables are output particles, and visible (strong) divergences, arising from terms that are compelled to diverge, equate such terms with $\mathbf{0}$.

Proposition 12 (Congruence of $\approx$, [SW01]) *$\approx$ is a congruence in $A\pi$.*

We have $\approx \subseteq \leftrightarrows$. Moreover, $\approx \subseteq \cong_\pi$ and $\leftrightarrows \subseteq \cong_\pi$, and we shall use both $\approx$ and $\leftrightarrows$ to establish properties of $\cong_\pi$. This task will be made easy by the use of *up-to techniques*, essentially *up to context* and *up to expansion*. Such techniques are well-known for $\approx$ (see [SW01]). We establish similar results for $\leftrightarrows$, which is a coarser equivalence (to our knowledge, the results about up-to techniques for coupled bisimulation are not proved elsewhere, albeit they are not surprising).

2.2.2 Results about coupled bisimilarity

Our treatment of *coupled bisimulation* follows [NP96]. However, our definition is slightly different. We work in a polyadic version of $A\pi$ whereas Nestmann and Pierce consider the monadic version. Moreover, our definition of $\leftrightarrows$ is based on the notion of *weak simulation* and not on the notion of *weak asynchronous simulation* as in [NP96].

Following the lines of [NP96], we can prove the congruence of coupled bisimulation as defined in Def. 10.

Proposition 13 *In $A\pi$, $\leftrightarrows$ is a congruence.*

Proof. Along the lines of the proof of Prop. 2.4.4 in [NP96]. This proof relies on the fact that in monadic $A\pi$, a *weak asynchronous simulation* is a congruence. We can easily state the counterpart of this result in our setting: in the polyadic $A\pi$, a *weak simulation* is a congruence (see [SW01]). □

In order to simplify the proofs involving *coupled bisimulation*, we develop an *up to expansion* technique for $\leftrightarrows$. We start by recalling the *up to expansion* technique for weak simulation.

Definition 14 (Weak simulation *up to expansion*) *A weak simulation* up to expansion *is a relation $\mathcal{R}$ such that for any processes P, P', Q, if $P\ \mathcal{R}\ Q$ and $P \xrightarrow{\mu} P'$, then there exists Q' such that $Q \stackrel{\hat{\mu}}{\Longrightarrow} Q'$ and $P' \gtrsim \mathcal{R} \lesssim Q'$.*

Proposition 15 *If Q weakly simulates P* up to expansion *then Q weakly simulates P.*

Proof. Let $\mathcal{R}$ be a weak simulation *up to expansion* such that $P\,\mathcal{R}\,Q$. We check that $\mathcal{S} \stackrel{\mathsf{def}}{=} \gtrsim \mathcal{R} \lesssim$ is a weak simulation.

The proof is a simple diagram chasing displayed on Figure 5.

$$\begin{array}{ccccc} P_1 & \gtrsim & P_1' & \mathcal{R} & Q_1' \lesssim Q_1 \\ \downarrow_{\overline{n}} & & \downarrow_{\overline{n}} & & \Downarrow_{\overline{n}} \quad \Downarrow_{\overline{n}} \\ P_2 & \gtrsim & P_2' & \gtrsim \mathcal{R} \lesssim & Q_2' \lesssim Q_2 \end{array}$$

Fig. 5. Diagram for $\mathcal{S}$

Let P_1, Q_1 and P_2 be processes such that $P_1\ \mathcal{S}\ Q_1$ and $P_1 \xrightarrow{\mu} P_2$. We know from the definition of $\mathcal{S}$ that there exist P_1' and Q_1' such that $P_1 \gtrsim P_1'$, $Q_1 \gtrsim Q_1'$

and $P_1' \,\mathcal{R}\, Q_1'$. As $P_1 \gtrsim P_1'$ and $P_1 \xrightarrow{\mu} P_2$, we have by definition of $\gtrsim$ that there exists P_2' such that $P_1' \xrightarrow{\hat{\mu}} P_2'$ and $P_2 \gtrsim P_2'$. From Def. 14 and since $P_1' \,\mathcal{R}\, Q_1'$ and $P_1' \xrightarrow{\hat{\mu}} P_2'$, there exists Q_2' such that $Q_1' \stackrel{\hat{\mu}}{\Longrightarrow} Q_2'$ and $P_2' \gtrsim \mathcal{R} \lesssim Q_2'$. As $Q_1' \lesssim Q_1$ and $Q_1' \stackrel{\hat{\mu}}{\Longrightarrow} Q_2'$, there exists Q_2 such that $Q_1 \stackrel{\hat{\mu}}{\Longrightarrow} Q_2$ and $Q_2' \lesssim Q_2$. Using the transitivity of $\gtrsim$, we deduce that $P_2 \,\mathcal{S}\, Q_2$. □

The following result, which is quite similar to Proposition 15, will be useful below (the definition of weak bisimulation up to expansion should be clear):

Proposition 16 (Weak bisimulation up to expansion, [SM92]) *If $\mathcal{R}$ is a weak bisimulation up to expansion, then $\mathcal{R}$ is contained in weak bisimilarity.*

Definition 17 (Mutual simulation up to expansion) *A mutual simulation* up to expansion *is a pair $(\mathcal{S}_1, \mathcal{S}_2)$ where $\mathcal{S}_1$ and $\mathcal{S}_2^{-1}$ are weak simulations up to expansion such that:*

- *if $P \;\mathcal{S}_1\; Q$ then $Q \Rightarrow \gtrsim Q'$ and $P \;\gtrsim\; \mathcal{S}_2\; Q'$;*
- *if $P \;S_2\; Q'$ then $P \Rightarrow \gtrsim P'$ and $P' \;\mathcal{S}_1 \lesssim\; Q'$.*

Proposition 18 *If P and Q are mutually similar up to expansion, then $P \leftrightarrows Q$.*

Proof. We prove that $(\mathcal{D}_1, \mathcal{D}_2) = (\gtrsim \mathcal{S}_1 \lesssim, \gtrsim \mathcal{S}_2 \lesssim)$ is a mutual simulation.

From the proof of Prop. 15, we know that $\mathcal{D}_1$ and $\mathcal{D}_2$ are weak simulations. We just need to show that $(\mathcal{D}_1, \mathcal{D}_2)$ satisfies the coupling conditions. Again the proof is a simple diagram chasing summed up by Fig. 6.

$$\begin{array}{ccccc} P_1 & \gtrsim & P_1' & \mathcal{S}_1 & Q_1' \lesssim Q_1 \\ & & & & \Downarrow \quad \Downarrow \\ P_1 & \gtrsim & P_1' & \gtrsim \mathcal{S}_2 \lesssim & Q_2' \lesssim Q_2 \end{array}$$

Fig. 6. Diagram for $(\mathcal{D}_1, \mathcal{D}_2)$

Let P_1 and Q_1 be two processes such that $P_1 \,\mathcal{D}_1\, Q_1$. We want to prove that there exists a process Q_2 such that $Q_1 \Rightarrow Q_2$ and $P_1 \,\mathcal{D}_2\, Q_2$. As $P \,\mathcal{D}_1\, Q$, there exist P_1' and Q_1' such that $P_1 \gtrsim P_1' \,\mathcal{S}_1\, Q_1' \lesssim Q_1$. Using the coupling condition for $\mathcal{S}_1$, we obtain Q_2' such that $Q_1' \Rightarrow Q_2'$ and $P_1' \gtrsim \mathcal{S}_2 \lesssim Q_1'$. As $Q_1 \gtrsim Q_1'$ and $Q_1' \Rightarrow Q_2'$, we get Q_2 such that $Q_1 \Rightarrow Q_2$ and $Q_2 \gtrsim Q_2'$. Using the transitivity of $\gtrsim$, we get $P_1 \gtrsim \mathcal{S}_2 \lesssim Q_2$, which is by definition $P_1 \,\mathcal{D}_2\, Q_2$. □

3 Analysing the method

3.1 No divergence-faithful encoding

Our first result shows that the setting we have introduced in Subsection 2.2 is in some sense not amenable to an analysis in the π-calculus.

Theorem 19 (No divergence-faithful encoding) *Let $\precsim$ be an equivalence relation on π-calculus terms containing structural congruence. There does not exist an encoding $[\![\cdot]\!]$ of $\lambda^{\parallel}$ in $A\pi$ such that, for any closed term M:*

(i) $[\![M]\!] \precsim [\![N]\!] \Rightarrow M \cong_{M} N$ *(soundness w.r.t. $\cong_{M}$);*

(ii) $[\![M]\!] \xrightarrow{\tau}{}^{\omega} \Leftrightarrow M \twoheadrightarrow^{\omega}$ *(divergence faithfulness);*

(iii) $M \twoheadrightarrow V \Rightarrow [\![M]\!] \xrightarrow{\tau}{}^{+} \precsim [\![V]\!]$ *(value preservation).*

Proof. We reason by absurd and we suppose that there exists such an encoding named $[\![\cdot]\!]$.

Let us consider a term Z such that the set of values reachable from M is $\{\, \lambda x_1 \dots x_n.\, x_1 \mid n > 1 \,\}$ and such that Z cannot diverge. We can easily prove that $Z \stackrel{\mathsf{def}}{=} \mathsf{Fix}\, \lambda z.\, (I \,[]\, (\lambda x.\, z\ (\lambda y.\, x))$ satisfies these conditions. We study $\mathcal{T}_{/\equiv}$, the quotient w.r.t. $\equiv$ of the reduction tree $\mathcal{T}$ of $[\![Z]\!]$ in the π-calculus, and we prove that $\mathcal{T}_{/\equiv}$ has infinitely many nodes.

For each $n > 1$, we know from property *(iii)* that there exists a node T_n in $\mathcal{T}$ such that $T_n \precsim [\![\lambda x_1 \dots x_n.\, x_1]\!]$. From property *(i)*, we can deduce that for all $m, n > 1$, if $m \neq n$ then $[\![\lambda x_1 \dots x_m.\, x_1]\!] \not\precsim [\![\lambda x_1 \dots x_n.\, x_1]\!]$. In fact, $\lambda x_1 \dots x_m.\, x_1 \not\cong_{\mathrm{M}} \lambda x_1 \dots x_n.\, x_1$ implies that $[\![\lambda x_1 \dots x_m.\, x_1]\!] \not\precsim [\![\lambda x_1 \dots x_n.\, x_1]\!]$. From this, and since $\equiv \subset \precsim$, we can deduce that for all $m \neq n$, $T_n \not\equiv T_m$. So, finally $\mathcal{T}_{/\equiv}$ has infinitely many nodes.

According to Proposition 9, $\mathcal{T}_{/\equiv}$ is finitely branching, and we have proved that it has infinitely many nodes. Using König's lemma, we can deduce that $\mathcal{T}$ has an infinite branch. This means that $[\![Z]\!] \rightarrow^{\omega}$ and, from property *(ii)*, this would imply that Z may diverge. This is in contradiction with the fact that Z cannot diverge. □

Remark 20 *The previous result holds in any* finitary *(i.e. preserving Proposition 9) extension of $A\pi$. To our knowledge, all extensions of the π-calculus considered in the literature are finitely branching, except for the operator of infinite sum.*

Infinite sums could be used to implement the counters introduced by [LM99]. However, the resulting encoding would be intractable. Since finitary operators are not the main focus of our work, we do not study conditions on the format of the rules that ensure the 'finitary property' for an operator.

3.2 Distinguishing between strong and weak divergences

As illustrated in Section 1, working with bisimulation in $A\pi$ leads us to distinguish between strong and weak divergences, that are defined as follows:

Definition 21 (Strong and weak divergences) *Let M be a $\lambda^{\mathbb{I}}$ term.*
- M is strongly divergent*, written $M\uparrow$, whenever M can evolve into a term that cannot converge;*
- M is weakly divergent *if M exhibits an infinite computation along which it never loses the possibility to converge.*

A divergent term is either strongly or weakly divergent, or both, as is $T \oplus \Omega$, where $T \stackrel{\text{def}}{=} \mathsf{Fix}\, \lambda x.\,(x \oplus I)$ is the $\lambda^{\mathbb{I}}$ term defined in Section 1. This distinction between strong and weak divergences already appears in [NC95]; we analyse its meaning below. Note that the notions of weak and strong divergences defined in Definition 21 depend on the derivation relation we consider. In Proposition 24, we will implicitly employ the same kind of construction based on another reduction relation (we will otherwise refer to relation $\twoheadrightarrow$ when mentioning weak and strong divergences).

We now adapt Definition 8 to focus on strong divergences.

Definition 22 (Behavioural equivalence on $\lambda^{\mathbb{I}}$, $\cong_\lambda$) *For any M, N, we have $M \cong_\lambda N$ iff for any closing context C:*

$$(C[M] \Downarrow \;\Leftrightarrow\; C[N] \Downarrow) \quad \textit{and} \quad (C[M]\uparrow \;\Leftrightarrow\; C[N]\uparrow)\,.$$

We can observe that $\cong_\lambda$ and $\cong_{\mathrm{M}}$ (Def. 8) are incomparable: as $\cong_{\mathrm{M}}$ is sensitive to weak divergences, it separates terms that are equated by $\cong_\lambda$, hence $\cong_\lambda \not\subseteq \cong_{\mathrm{M}}$. Conversely, $\cong_\lambda \not\supseteq \cong_{\mathrm{M}}$ because $\cong_{\mathrm{M}}$ identifies weak and strong divergences. We have for instance:

$$I \;\begin{array}{c}\cong_\lambda \\ \not\cong_{\mathrm{M}}\end{array}\; \mathsf{Fix}\, \lambda x.\,(x \oplus I) \;\begin{array}{c}\not\cong_\lambda \\ \cong_{\mathrm{M}}\end{array}\; \Omega \oplus I\,.$$

This means in particular that the method we develop in this paper cannot be used to reason about $\lambda^{\mathbb{I}}$ as introduced in [LM99].

3.3 The relevance of strong divergences

As will be seen in Subsection 4.2, the desired properties for *amb* indeed hold in our setting, so that we may say that our presentation of $\lambda^{[\![]}$ which focuses on strong divergences agrees with *amb*'s specification. Before presenting the π-calculus' point of view on $\lambda^{[\![]}$, we examine the consequences brought by the observation of only strong divergences *within* $\lambda^{[\![]}$. We start by analysing $\cong_\lambda$ and its influence on the notion of divergence.

3.3.1 Robustness of strong divergences

Reasoning with $\cong_\lambda$ brings *de facto* a form of fairness. To illustrate this claim, we introduce a non fair operational semantics for *amb*:

Definition 23 ($\hookrightarrow$) *Relation $\hookrightarrow$ is defined by the following rules (rules* Beta*,* Lazy*, and symmetrical versions of rules* Amb_L *and* Imm_L *are omitted):*

$$\text{Imm}_L \; V \,[\!]\, N \;\hookrightarrow\; V \qquad\qquad \text{Amb}_L \; \frac{M \;\hookrightarrow\; M'}{M \,[\!]\, N \;\hookrightarrow\; M' \,[\!]\, N}$$

It can be remarked that $\hookrightarrow$ describes an operator similar to Boudol's [Bou94], where parallel composition has no fairness property. We have:

Proposition 24 (Fair and non fair operational semantics) *Relations $\twoheadrightarrow$ and $\hookrightarrow$ induce the same notions of convergence and strong divergence.*

Before going on with the proof, we establish some preliminary results on strongly divergent terms.

Definition 25 *A strongly divergent term M is said to strongly diverge at distance k for $\hookrightarrow$ if for some term R, $M \hookrightarrow^k R$ and R cannot converge.*

Lemma 26 *We write $\mathsf{Val}(M)$ for the set of values that are reachable from M. Consider a term M of the form $M = (P_1 \,[\!]\, P_2)\; N_1 \ldots N_n$. If M may strongly diverge at distance $k > 0$ for $\hookrightarrow$, then*

- *either $P_1\, N_1 \ldots N_n$ and $P_2\, N_1 \ldots N_n$ may strongly diverge at distance at most k for $\hookrightarrow$,*
- *or $V\; N_1 \ldots N_n$ is strongly divergent at distance strictly less than k for $\hookrightarrow$ where $V \in \mathsf{Val}(P_1) \cup \mathsf{Val}(P_2)$.*

Proof. By definition of the distance k, there exists a sequence of terms $(M_i)_{i\in[0,k]}$ such that M_k cannot converge and $M_i \hookrightarrow M_{i+1}$ for all $i \in [0, k-1]$.

We distinguish two cases.

- If for all $i \in [0,n]$, $M_i = (P_i^1 \,[\!]\, P_i^2)\ N_1 \dots N_n$, then we have $P_0^1 \hookrightarrow^{\leq k} P_k^1$ et $P_0^2 \hookrightarrow^{\leq k} P_k^2$. As $M_k = P_k^1 \,[\!]\, P_k^2\, N_1 \dots N_n$ cannot converge, it is also the case for $P_k^1 N_1 \dots N_n$ and $P_k^2 N_1 \dots N_n$. So, $P_1 N_1 \dots N_n$ and $P_2 N_1 \dots N_n$ may strongly diverge in the sense of $\hookrightarrow$ at distance at most k.
- If for some $j \in [0,n]$, M_j is not of the form $(P_i^1 \,[\!]\, P_i^2)\ N_1 \dots N_n$ then we call j_0 the smallest such integer. We have $M_{j_0} = V\ N_1 \dots N_n$ where $V \in \mathsf{Val}\,(P_1) \cup \mathsf{Val}\,(P_2)$. As $M_{j_0} \hookrightarrow^{<k} M_k$, $V N_1 \dots N_n$ may strongly diverge at distance less than k. □

Lemma 27 $\twoheadrightarrow \subseteq \hookrightarrow^+$.

Proof. We prove that for all P and Q, if $P \twoheadrightarrow Q$ then $P \hookrightarrow^+ Q$ by induction on the derivation tree of $P \twoheadrightarrow Q$. □

Proof of Proposition 24 By Lemma 27, we remark that we only need to prove that for any term M and value V:

(1) $M \hookrightarrow^+ V$ implies $M \twoheadrightarrow V$,
(2) If M may strongly diverge for $\hookrightarrow$, then M may strongly diverge for $\twoheadrightarrow$.

(1) We prove by induction on n that $M \hookrightarrow^n V$ implies $M \twoheadrightarrow V$ and $M \hookrightarrow^n (VN)$ implies $M \twoheadrightarrow (VN)$.
- Case $n = 1$. Immediate
- Case $n > 1$. Let us consider the sequence $(M_i)_{i\in[0,n]}$ associated to $M \hookrightarrow^n V$.

  If for some j, M_j is a β-redex, we distinguish two cases. If $j = 0$ then $M_0 = (\lambda x.K')K$ and $M_0 \hookrightarrow_{\text{Beta}} M_1$ thus $M_0 \twoheadrightarrow M_1$ and by induction hypothesis applied to $M_1 \hookrightarrow^{n-1} M_n$, we can conclude. If $j > 0$ then $M_0 \hookrightarrow^{<n} M_j$ and $M_j \hookrightarrow^{<n} M_n$, and we conclude by applying the induction hypothesis.

  If none of the M_is is a β-redex, then all the M_is are of the form $P_i \,[\!]\, Q_i$. We have either $P_0 \hookrightarrow^{<n} V$, or $Q_0 \hookrightarrow^{<n} V$, or $P_0 = V$, or $Q_0 = V$, and we easily conclude.

  Let us consider the sequence $(M_i)_{i\in[0,n]}$ associated to $M \hookrightarrow^n (VN)$.

  If for some j, M_j is a β-redex, we distinguish two cases. If $j = 0$ then $M_0 = (\lambda x.K')K$ and $M_0 \hookrightarrow_{\text{Beta}} M_1$, thus $M_0 \twoheadrightarrow M_1$ and, by applying the induction hypothesis to $M_1 \hookrightarrow^{n-1} M_n$, we can conclude. If $j > 0$ then $M_0 \hookrightarrow^{<n} M_j$ and $M_j \hookrightarrow^{<n} M_n$ and we conclude by applying the induction hypothesis.

  If none of the M_is is a β-redex, then all the M_is are of the form $P_i N$. All rules applied are of type Lazy and therefore $P_0 \hookrightarrow^n V$. We

already proved that this implies $P_0 \twoheadrightarrow V$, and hence $P_0 N \twoheadrightarrow VN$.

(2) We reason by absurd and we suppose that there exists a term M such that $M \downarrow$ for $\hookrightarrow$ but not for $\twoheadrightarrow$. Let k_0 be the smallest integer such that there exists a term M such that $M \downarrow$ for $\hookrightarrow$ at distance k_0 but not for $\twoheadrightarrow$. Let M_0 be the smallest term (w.r.t. the number of symbols used in its syntax) that may strongly diverge at distance k_0 for $\hookrightarrow$ but cannot strongly diverge for $\twoheadrightarrow$.

From the previous proof, we know that $k_0 > 0$. M_0 cannot be a β-redex because if $(\lambda x.K)N$ may strongly diverge at distance $k > 0$, then $K[N/x]$ may strongly diverge at distance $k-1$, and this would contradict the definition of k_0. Thus $M_0 = (P_1 \,[\!]\, P_2)\ N_1 \dots N_n$, and, by Lemma 26, we have:

either $P_1 N_1 \dots N_n$ and $P_2 N_1 \dots N_n$ strongly diverge at distance at most k_0: this contradicts the definition of M_0.

or there exists a value $V \in \mathsf{Val}(P_1) \cup \mathsf{Val}(P_2)$ such that $VN_1 \dots N_n$ may diverge at distance less than k_0. From the previous proof, we know that $M \twoheadrightarrow VN_1 \dots N_n$. As M cannot strongly diverge in the sense of $\twoheadrightarrow$, it is also the case for $VN_1 \dots N_n$. This contradicts the definition of k_0. □

This shows that all divergences added by $\hookrightarrow$ w.r.t. $\twoheadrightarrow$ are *weak*, and hence that from the point of view of $\cong_\lambda$, relation $\twoheadrightarrow$ or relation $\hookrightarrow$ can indifferently be used, fairness 'at an operational level' being somehow irrelevant in our setting.

3.3.2 An operational semantics that neglects weak divergences

Proposition 24 suggests that the characteristic properties of *amb* are guaranteed at the level of behavioural equivalence. It is thus natural to analyse the distinction between strong and weak divergences operationally, in order to see whether this distinction can be grasped at the level of execution.

We show in a rather general setting that it is possible to provide an operational semantics in which weak divergences always have a null probability, whereas convergences and strong divergences occur with a non-null probability. This suggests that the focus on strong divergences can be achieved (at least theoretically) by means of a particular evaluation strategy.

A probability measure for sets of computations

We now define a framework to compute probabilities over an arbitrary *finitely branching* relation $\rightarrow$, defined over a set of terms $\widetilde{M}$.

Definition 28 *A computation is a sequence $(c_i)_{i \in I}$ with either $I = \mathbb{N}$ or $I = [0, n]$ such that for all $i \in I \setminus \{0\}$, $c_{i-1} \rightarrow c_i$. When $c_0 = m$, we refer to a*

computation starting from m. *A computation is* maximal *if it is infinite or if it ends with a term that has no successor. A finite computation c is a* prefix *of a computation c' if $c' = cw$ for some possibly empty computation w.*

From now on, we fix a term $m_0 \in \widetilde{M}$ and we write $\mathcal{C}$ (resp. $\mathcal{C}^+$) for the set of all computations (resp. maximal computations) starting from m_0. If not stated otherwise, all computations are assumed to start from m_0.

Definition 29 (Intervals of $\mathcal{C}^+$) *For all finite computation $x \in \mathcal{C}$, the* interval rooted in x, *written I_x, is the set $\{c \in \mathcal{C}^+ \mid x \text{ is a prefix of } c\}$. We also define the set of all intervals of $\mathcal{C}^+$, written $\mathcal{I}$, as follows:*

$$\mathcal{I} \quad \stackrel{\text{def}}{=} \quad \{\emptyset\} \cup \{I_x, \mid x \in \mathcal{C}, x \text{ finite}\}\,.$$

The set $\mathcal{I}$ of all intervals of $\mathcal{C}^+$ enjoys some closure properties, that are expressed using the following definition:

Definition 30 (Semi-ring) *A semi-ring on a set Ω is a subset $\mathcal{S}$ of the power set 2^Ω of Ω, with the following properties:*

(1) $\emptyset \in \mathcal{S}$;
(2) for all $A, B \in \mathcal{S}$, $A \cap B \in \mathcal{S}$;
(3) for all $A, B \in \mathcal{S}$, there exists a finite sequence $(A_i)_{i\in[1,n]}$ of pairwise disjoint elements of $\mathcal{S}$ such that $A \setminus B = \cup_{i=1}^n A_i$.

Proposition 31 *The set $\mathcal{I}$ is a semi-ring on $\mathcal{C}^+$.*

Proof. (1) holds by definition, so we only have to check properties (2) and (3). Before proceeding, we remark that for any two intervals I_x and I_y, if x is a prefix of y, then I_y is included in I_x, and that if x and y are incomparable (for the prefix relation) then I_x and I_y are disjoint.

(2) Let A and B be two intervals in $\mathcal{I}$. It follows from the previous remark that $A \cap B$ is equal to either $\emptyset$, A or B. In all cases, $A \cap B$ belongs to $\mathcal{I}$.
(3) Given two intervals I_x and I_y in $\mathcal{I}$, we want to express $I_x \setminus I_y$ as a finite union of intervals. If I_x and I_y are disjoint or if I_x is included in I_y, it is immediate.

So we only need to consider the case where x is a prefix of y (i.e., $I_y \subset I_x$). We call D the set of computations of the form $zb \in \mathcal{C}$ such that $b \in \widetilde{M}$, x is a prefix of z, z is a prefix of y, zb is not a prefix of y and $z \neq y$. As the relation is finitely branching, D is finite. It is straightforward to show that a computation belongs to $I_x \setminus I_y$ if and only if it has a prefix in D. Hence $I_x \setminus I_y$ is equal to the finite union $\cup_{d\in D} I_d$. □

We now introduce the notion of probability measure, and define a 'natural' probability measure on $\mathcal{I}$.

Definition 32 (Probability measure) *A probability measure μ on a subset $\mathcal{S}$ of 2^Ω, which contains $\emptyset$ and Ω, is a mapping from $\mathcal{S}$ to $[0,1]$ such that:*

- $\mu(\Omega) = 1$;
- *for any sequence $(A_i)_{i\in\mathbb{N}}$ of pairwise disjoint elements of $\mathcal{S}$, if $\cup_{i\in\mathbb{N}} A_i$ belongs to $\mathcal{S}$ then $\mu(\cup_{i\in\mathbb{N}} A_i) = \sum_{i\in\mathbb{N}} \mu(A_i)$.*

We assume that a computation c starting from m_0 is randomly chosen as follows: if the kth term of the computation c_k is irreducible, then the process stops, otherwise the $(k+1)$th term is drawn from the set of successors of c_k (all successors have an equal probability to be chosen). The probability of obtaining a computation in the interval I_c, where $c = c_0 \dots c_n$ with $n \geq 1$, is given by

$$\mathcal{P}(I_c) = \prod_{i\in[0,n-1]} \frac{1}{|c_i^{\rightarrow}|},$$

where $|m^{\rightarrow}|$ stands for the cardinality of the set of successors of m. We also set $\mathcal{P}(I_{m_0}) = 1$ and $\mathcal{P}(\emptyset) = 0$.

Proposition 33 *$\mathcal{P}$ is a probability measure on $\mathcal{I}$.*

Proof. Since $\mathcal{C}^+ = I_{m_0}$, $\mathcal{P}(C^+) = \mathcal{P}(I_{m_0}) = 1$ and it is obvious that for any interval I, $\mathcal{P}(I)$ belongs to $[0,1]$. It remains to show that if an interval I_x is equal to $\cup_{i\in\mathbb{N}} A_i$ for some sequence $(A_i)_{i\in\mathbb{N}}$ of pairwise disjoint elements of $\mathcal{I}$ then $\mathcal{P}(I_x) = \sum_{i\in\mathbb{N}} \mathcal{P}(A_i)$.

- We first prove that there are only finitely many non-empty A_is.

  Suppose by absurd that there are infinitely many non-empty A_is. We construct an increasing sequence $(c_i)_{i\in\mathbb{N}}$ of finite computations in $\mathcal{C}$ such that $c_0 = x$ and for all $i \in \mathbb{N}$, the set $N_i = \{j \in \mathbb{N} \mid A_j \neq \emptyset \text{ and } A_j \subset I_{c_i}\}$ is infinite.

  The property holds for $c_0 = x$. Given c_i, we construct c_{i+1}. As the relation $\rightarrow$ is finitely branching, the set $S = \{s \in \mathcal{C} \mid s = c_i b \text{ and } b \in \widetilde{M}\}$ is finite and non-empty. For all $s \in S$, we call M_s the set $\{j \in \mathbb{N} \mid A_j \neq \emptyset \text{ and } A_j \subset I_s\}$. As N_i is equal to the finite union $\cup_{s\in S} M_s$, there exists at least one $s_0 \in S$ such that M_{s_0} is infinite. We take c_{i+1} equal to s_0.

  Let c be the limit of $(c_i)_{i\in\mathbb{N}}$, we claim that c does not belong to $\cup_{i\in\mathbb{N}} A_i$. In fact, if c belongs to some A_{i_0} then it implies that for some j, $I_{c_j} = A_{i_0}$. As all the A_is are pairwise disjoint, this would contradict the fact that I_{c_j} contains infinitely many non-empty A_is.

  We thus have a c which belongs to I_x but not to $\cup_{i\in\mathbb{N}} A_i$: this contradicts the fact that $I_x = \cup_{i\in\mathbb{N}} A_i$.

- It remains to prove that if $I_x = \cup_{i\in[1,n]} I_{a_i}$ where the I_{a_i}s are pairwise disjoint intervals, then $\mathcal{P}(I_x) = \sum_{i=1}^{n} \mathcal{P}(I_{a_i})$.

  We proceed by induction on n. The case $n = 1$ is immediate. Suppose that the property holds for some $n \geq 1$, we prove it for $n+1$. Let x_0 be the smallest computation such that x is a prefix of x_0 and x_0 ends with a term $m \in \widetilde{M}$ having more than one successor. The computation x_0 exists because I_x contains at least two disjoint intervals. Moreover, we have $I_x = I_{x_0}$ and, therefore, $I_{x_0} = \cup_{i\in[1,n+1]} A_i$. Let S be the set of computations defined by $S \stackrel{\mathsf{def}}{=} \{s \in \mathcal{C} \mid s = x_0 b \text{ and } b \in \widetilde{M}\}$. By definition of x_0, S contains at least two elements and $I_x = \cup_{s\in S} I_s$. For all $s \in S$, let us call R_s the set $\{i \in [1, n+1] \mid A_i \subset I_s\}$. As $I_x = \cup_{i\in[1,n+1]} A_i$, for all $s \in S$, $I_s = \cup_{i\in R_s} A_i$ with $|R_s| \leq n$. Therefore, by induction hypothesis, $\mathcal{P}(I_s) = \sum_{i\in R_s} \mathcal{P}(A_i)$. We finally have $\mathcal{P}(I_x) = \mathcal{P}(I_{x_0}) = \sum_{s\in S} \mathcal{P}(I_s) = \sum_{s\in S} \sum_{i\in R_s} \mathcal{P}(A_i) = \sum_{i\in[1,n+1]} \mathcal{P}(A_i)$. □

We want to measure the set of convergent (that is, maximal and finite) computations V, the set of strongly divergent computations S, and the set of weakly divergent computations W. In general, these sets do not belong to $\mathcal{I}$. We are therefore led to consider the closure under countable union and complement of $\mathcal{I}$, given by the following definition:

Definition 34 (σ-algebra) *Given a subset $\mathcal{S}$ of 2^Ω, the σ-algebra generated by $\mathcal{S}$, written $\sigma(\mathcal{S})$, is the smallest set containing $\mathcal{S}$ and closed under countable union and complement.*

The following classical theorem says that there exists a unique extension of $\mathcal{P}$ to $\sigma(\mathcal{I})$. In the following, we do not distinguish between $\mathcal{P}$ and its extension.

Theorem 35 (Caratheodory's extension, [Bil95]) *Let $\mathcal{S}$ be a semi-ring on Ω and μ a probability measure on $\mathcal{S}$, there exists a unique probability measure μ' on $\sigma(\mathcal{S})$ extending μ.*

It is fairly easy to check that V, S and W belong to $\sigma(\mathcal{I})$:

- As for each convergent computation v, we have $I_v = \{v\}$, and since V is a countable set, V is equal to the countable union $\cup_{v\in V} I_v$, and therefore $V \in \sigma(\mathcal{I})$.
- Let X be the set of all finite computations $c = c_0 \dots c_n$ in $\mathcal{C}$ such that c_n cannot converge and c_{n-1} (if it exists, i.e., when $n > 0$) can. It is straightforward to prove that X is countable and that a computation is strongly divergent if and only if it has a prefix in X. Hence, S coincides with the countable union $\cup_{x\in X} I_x$ and therefore $S \in \sigma(\mathcal{I})$.
- If the computation never reaches an irreducible term nor a term that cannot converge, then it is a weak divergence. Hence, W is equal to $\mathcal{C}^+ \setminus (V \cup S)$

and therefore $W \in \sigma(\mathcal{I})$.

Remark 36 *Since V and S can always be expressed as unions of intervals, if V (resp. S) is non-empty then $\mathcal{P}(V) > 0$ (resp. $\mathcal{P}(S) > 0$), because V (resp. S) contains at least one non-empty interval.*

Moreover, as $\mathcal{C}^+$ is equal to the disjoint union $V \cup S \cup W$, we have $\mathcal{P}(V) + \mathcal{P}(S) + \mathcal{P}(W) = 1$. As the following example shows, the probability of weak divergences is in general non-null.

Example 37 *Let $\widetilde{M}$ be the set of words over natural numbers $\{a_1 \dots a_n \mid a_i \in [0, 2^i - 1], n \in \mathbb{N}\} \cup \{\varepsilon\}$. Consider the relation $\not\mapsto$ on $\widetilde{M}$ defined by $\varepsilon \not\mapsto 0$, $\varepsilon \not\mapsto 1$ and for all wx and wxy in $\widetilde{M}$ with $x < 2^{|w|+1} - 1$, $wx \not\mapsto wxy$ (where $|w|$ stands for the length of the word w). The graph of $\not\mapsto$ starting from ε is given in Figure 7. There is no strong divergence starting from ε, hence $\mathcal{P}(S) = 0$*

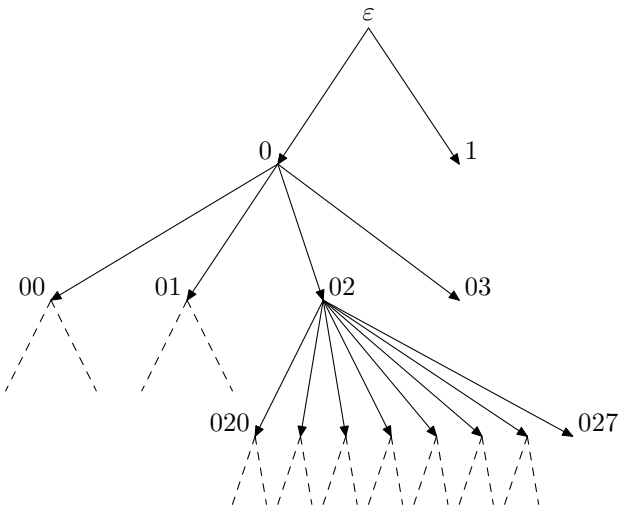

Fig. 7. An example where weak divergences occur with a non null probability.

(at each node in the tree, one can reach a blocked state in one step of $\not\mapsto$). The probability of convergence is given by:

$$\mathcal{P}(V) = \sum_{i=1}^{+\infty} \frac{1}{2^i} \prod_{k=1}^{i-1} \left(1 - \frac{1}{2^i}\right) = 1 - \underbrace{\prod_{i=1}^{+\infty} \left(1 - \frac{1}{2^i}\right)}_{\mathcal{P}(W)>0}$$

(the last equality can be proved by induction). Therefore, the probability of exhibiting a weak divergence starting from ε is non-null.

Weak divergence avoiding execution

We now define a relation $\rightrightarrows$ based on $\rightarrow$ which induces the same notions of convergence and strong divergence as $\rightarrow$. Moreover, $\rightrightarrows$ is such that weak

divergences have a null probability: for any starting term $m_0 \in \widetilde{M}$, $\mathcal{P}(W) = 0$.

Definition 38 (Weak divergence avoiding execution, $\rightrightarrows$) *Given a set of terms $\widetilde{M}$ and a finitely branching relation $\to$, the relation $\rightrightarrows \subseteq \left(2^{\widetilde{M}} \times \{+,-\}\right) \times \left(2^{\widetilde{M}} \times \{+,-\}\right)$ (where $2^{\widetilde{M}}$ stands for the power set of $\widetilde{M}$) is defined by the following rules, for $\widetilde{N} \in 2^{\widetilde{M}}$:*

- *If $\widetilde{N}$ does not contain values then $(\widetilde{N}, -) \rightrightarrows (\widetilde{N'}, -)$ where $\widetilde{N'}$ is equal to $\{n' \mid n \in \widetilde{N} \text{ and } n \to n'\}$.*
- *If $\widetilde{N}$ contains only reducible terms, then $(\widetilde{N}, +) \rightrightarrows (\{r\}, -)$ where $r \in \widetilde{N}$.*
- *If $\widetilde{N}$ can be written as the disjoint union $\widetilde{V} \cup \widetilde{R}$ for a set of values $\widetilde{V}$ and a set of reducible terms $\widetilde{R}$, then there are two transitions: $(\widetilde{N}, -) \rightrightarrows (\widetilde{V}, +)$ and $(\widetilde{N}, -) \rightrightarrows (\widetilde{R}, +)$.*
- *If $\widetilde{V}$ is a set of values not reduced to a singleton, then $(\widetilde{V}, +) \rightrightarrows (\{v\}, +)$ where $v \in \widetilde{V}$.*

Intuitively, when starting from $(\{m\}, +)$, relation $\rightrightarrows$ describes a particular strategy for the exploration of possible computations (for $\to$) issued from m. The *polarities* $+, -$ are introduced to 'program' an equiprobable choice between reaching a value or choosing not yet reduced branches in the third clause of the definition above. This way, the probability of weak divergences is brought to zero (since weak divergences lead to infinitely many such choices).

Example 39 *The relation $\rightrightarrows$ corresponding to the derivation $\Rightarrow$ of Example 37 is given in Figure 8. The resulting probability of exhibiting a weak divergence is thus $\mathcal{P}(W) = 1 - \sum_{i=1}^{+\infty} \frac{1}{2^i} = 0$.*

The following proposition states that $\rightrightarrows$ somehow preserves the behaviour expressed by $\to$, as far as convergences and strong divergences are concerned.

Proposition 40 *For any term $m \in \widetilde{M}$:*

(1) For any value $v \in \widetilde{M}$, $m \to^* v$ if and only if $(\{m\}, +) \rightrightarrows^* (\{v\}, +)$.
(2) The term m may strongly diverge w.r.t $\to$ if and only if $(\{m\}, +)$ may strongly diverge w.r.t $\rightrightarrows$.
(3) If $(\{m\}, +)$ may strongly diverge w.r.t $\rightrightarrows$, then the probability of exhibiting a strongly divergent computation w.r.t $\rightrightarrows$ (i.e. $\mathcal{P}(S)$) is non null.

Proof.

(1) A straightforward induction establishes that for all subsets $\widetilde{N}$ and $\widetilde{N'}$ of $\widetilde{M}$ and for all $\epsilon, \epsilon' \in \{-, +\}$ such that $(\widetilde{N}, \epsilon) \rightrightarrows^+ (\widetilde{N'}, \epsilon')$, we have that for all $n \in \widetilde{N}$ and $n' \in \widetilde{N'}$, $n \to^+ n'$. Conversely, if $n \to^+ n'$, then there exists $\widetilde{N'} \subset \widetilde{M}$ such that $n' \in \widetilde{N'}$ and $(\{n\}, +) \rightrightarrows^+ (\widetilde{N'}, -)$. It follows

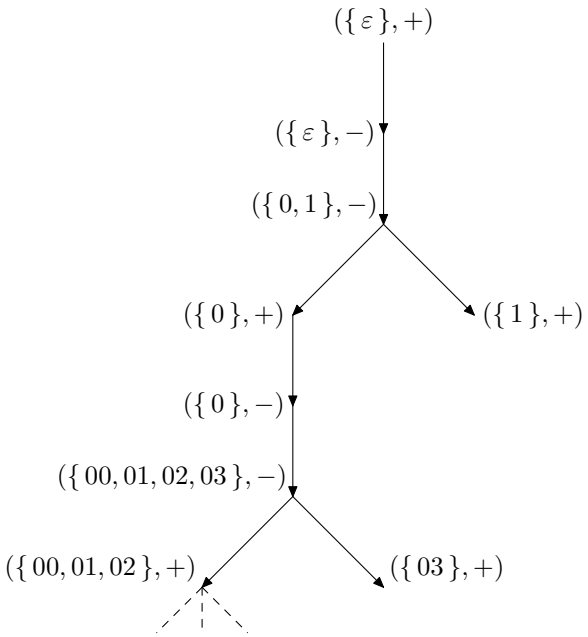

Fig. 8. The relation $\rightrightarrows$ based on the derivation of Example 37.

that m may converge to v w.r.t $\rightarrow$ if and only if $(\{m\}, +)$ may converge to $(\{v\}, +)$ w.r.t $\rightrightarrows$.

(2) If m may strongly diverge w.r.t $\rightarrow$, there exists a strongly divergent term $m' \in \widetilde{N}$ such that $m \rightarrow^+ m'$. There exists $\widetilde{N} \subset \widetilde{M}$ and $m' \in \widetilde{N}$ such that $(\{m\}, +) \rightrightarrows (\widetilde{N}, -)$. If $\widetilde{N}$ contains only strongly divergent terms, then $(\{m\}, +)$ is strongly divergent. Otherwise, $\widetilde{N}$ contains terms that may converge. Hence, there exists $\widetilde{N'} \subset \widetilde{M}$ such that $(\widetilde{N}, -) \rightrightarrows^* (\widetilde{N'}, -)$ and $\widetilde{N'}$ contains a value v and a term $m'' \in \widetilde{M}$ satisfying $m' \rightarrow^* m''$. It follows that $(\widetilde{N'}, -) \rightrightarrows (\{m''\}, -)$. As m'' is strongly divergent (being a reduct of m', which is strongly divergent) and $(\{m\}, +) \rightrightarrows (\{m''\}, +)$, $(\{m\}, +)$ is strongly divergent.

If $(\{m\}, +)$ may strongly diverge w.r.t. $\rightrightarrows$, then there exists $\widetilde{N} \subset \widetilde{M}$ which contains only strongly divergent terms such that $(\{m\}, +) \rightrightarrows (\widetilde{N}, +)$. Hence, there exists a strongly divergent term $m' \in \widetilde{N}$ such that $m \rightarrow^* m'$.

(3) This follows from Remark 36. □

We now show that $\rightrightarrows$ annihilates the probability of weak divergences:

Proposition 41 *For all $m \in \widetilde{M}$, the probability $\mathcal{P}(W)$ of weak divergences starting from $(\{ m \}, +)$ is equal to zero.*

Proof. If $(\{ m \}, +)$ cannot weakly diverge then the set W is empty and $\mathcal{P}(W) = \mathcal{P}(\emptyset) = 0$.

Suppose now that $(\{\, m \,\}, +)$ may weakly diverge. Unless stated otherwise, we suppose that all computations start with $(\{\, m \,\}, +)$. Let $c = c_0 \rightrightarrows \ldots \rightrightarrows c_k$ be a computation of $\rightrightarrows$; the *depth of* c is the number of positive polarities decorating the states in c minus one. Let W_n be the set of all computations of the form $c = c_0 \ldots c_n c_{n+1}$ such that c_n has a positive polarity, and c_{n+1} can converge and has a negative polarity. It is easy to check that the $(I_w)_{w \in W_n}$ are pairwise disjoint and that $W \subset \bigcup_{w \in W_n} I_w$.

We prove by induction on n that for all $n \geq 1$, $\mathcal{P}(\bigcup_{w \in W_n} I_w) \leq \frac{1}{2^n}$.

- For $n = 1$, it is enough to remark that there exists a computation c such that for all $w \in W_1$, $w = cb$ for some $b \in \widetilde{M}$. Moreover $\mathcal{P}(I_c) = \frac{1}{2}$ since all elements of $c = c_1 \ldots c_n$ have only one successor except for c_{n-1}, that has two. As $\bigcup_{w \in W_n} I_w \subset I_c$, we have $\mathcal{P}(\bigcup_{w \in W_n} I_w) \leq \frac{1}{2}$.
- Suppose that the property holds for $n \geq 1$; we prove it for $n + 1$. Each $x \in W_{n+1}$ admits a unique prefix in W_n. For each $w \in W_n$, we call S_w the set of elements from W_{n+1} admitting w as a prefix. We have $\bigcup_{x \in S_w} I_x \subset I_w$. By a reasoning similar to the case $n = 1$, we can prove that for all $w \in W_n$, $\mathcal{P}(\bigcup_{x \in S_w} I_x) \leq \frac{1}{2} I_w$. Therefore, $\mathcal{P}(\bigcup_{x \in W_{n+1}} I_x) = \sum_{w \in W_n} \mathcal{P}(\bigcup_{x \in S_w} I_x) \leq \frac{1}{2} \sum_{w \in W_n} I_w \leq \frac{1}{2^{n+1}}$.
  Since for all n, $W \subset \bigcup_{w \in W_n} I_w$, we have

$$\mathcal{P}(W) \leq \lim_{n \to +\infty} P(W_n) = 0\,,$$

  and, finally, the probability of exhibiting a weak divergence starting from $(\{\, m \,\}, +)$ is null. $\square$

Since $\hookrightarrow$ is finitely branching, Propositions 40 and 41 show that there is a way to execute $\lambda^{\mathbb{I}}$ terms so that weak divergences are unlikely to happen (and without introducing pathological behaviours).

This ends our presentation of $\lambda^{\mathbb{I}}$ and of the approach we shall adopt in the remainder of the paper.

4 Pi-calculus at work

We now turn to the π-calculus analysis of $\lambda^{\mathbb{I}}$. We start by presenting the corresponding translation.

$$[\![\lambda x.M]\!]_p \stackrel{\mathsf{def}}{=} \boldsymbol{\nu} l\ (\bar{p}\langle l\rangle \mid l(x,q).[\![M]\!]_q) \qquad [\![x]\!]_p \stackrel{\mathsf{def}}{=} \boldsymbol{\nu} p'\ (\bar{x}\langle p'\rangle \mid p' \rightarrowtriangle p)$$
$$[\![M\ N]\!]_p \stackrel{\mathsf{def}}{=} \boldsymbol{\nu} q\ \Big([\![M]\!]_q \mid q(l).\Big(\boldsymbol{\nu} x\ \Big(\bar{l}\langle x,p\rangle \mid\ !x(r).[\![N]\!]_r\Big)\Big)\Big)$$
$$[\![M [] N]\!]_p \stackrel{\mathsf{def}}{=} \boldsymbol{\nu} p'\ ([\![M]\!]_{p'} \mid [\![N]\!]_{p'} \mid p' \rightarrowtriangle p) \qquad \text{where } q \rightarrowtriangle p \stackrel{\mathsf{def}}{=} q(x).\bar{p}\langle x\rangle$$

Fig. 9. Encoding of $\lambda^{[]}$ in the π-calculus

4.1 Encoding and soundness

Our encoding of $\lambda^{[]}$ in $A\pi$, written $[\![\_]\!]$, is defined on Figure 9, and follows the usual encodings of the λ-calculus for the operators of application and abstraction. A $\lambda^{[]}$-term M is mapped to a process $[\![M]\!]_p$, p being a channel where the value of (the encoding of) M will be passed (cf. the clause for abstraction).

To take $[]$ into account, we run the encodings of M and N in parallel at a freshly created location q, and let the (ephemeral) link process $q \rightarrowtriangle p$ forward any successfully terminated evaluation on p. Once $q \rightarrowtriangle p$ has been triggered by one of the components, the other component is isolated from the context, either because it tries to interact on the private channel q, or because it diverges. Modulo $\cong_\pi$, this corresponds to what we expect from *amb*.

Note the extra indirection $p' \rightarrowtriangle p$ in the encoding of variables. A similar indirection is needed in the encoding of call-by-value into (untyped) π-calculus, and can be removed using capability types [SW01]. We do not know how to avoid the indirection in the encoding of Figure 9 using types or other means.

Example 42 *We illustrate the encoding on a simple example: consider the $\lambda^{[]}$ term $\lambda x\, y.\, (x[]y)$, that builds the* amb *composition of two terms. Its π-calculus encoding is given by:*

$$[\![\lambda x\, y.\, (x[]y)]\!]_p =$$
$$\boldsymbol{\nu} l, l'\ \Big(\bar{p}\langle l\rangle \mid l(x,q).\Big(\bar{q}\langle l'\rangle \mid l'(y,r).\,(\boldsymbol{\nu} n\ (\bar{x}\langle n\rangle \mid \bar{y}\langle n\rangle \mid n(z).\bar{r}\langle z\rangle))\Big)\Big)$$

Channels l and l' are the ports where the encoded $\lambda^{[]}$ function receives its arguments (these will be referred to using the channels instantiating x and y, respectively). Subterm $\boldsymbol{\nu} n\ (\bar{x}\langle n\rangle \mid \bar{y}\langle n\rangle \mid n(z).\bar{r}\langle z\rangle)$ is the most informative for our purposes: it shows that amb *is programmed via channel n, a resource that is used concurrently by the two agents that receive n. The important observation is that n is* linear*, in the sense that there is only one input at n — in the encoding, only the output capability on n is passed, so that processes receiving n are not allowed to perform an input on this channel. This way, the*

first agent that interacts at n consumes the input and prevents the other one from proceeding.

We now turn to the soundness proof for $[\![\_]\!]$, which is adapted from [San00]. The main property we need for this is given by Lemma 47, for which we first establish some auxiliary results in the π-calculus, that will allow us to prove operational correspondence for the encoding.

Lemma 43 *For any name x and any processes P, P_1, P_2 and Q such that x only appears as output object in P, P_1 and P_2 and $x \notin \mathsf{fn}(Q)$:*

(1) $\boldsymbol{\nu} x\ (P_1 \mid P_2 \mid !x(q).Q) \sim \boldsymbol{\nu} x\ (P_1 \mid !x(q).Q) \mid \boldsymbol{\nu} x\ (P_2 \mid !x(q).Q)$;
(2) For any context C such that $x \notin fn(C)$,

$$\boldsymbol{\nu} x\ (C[P] \mid !x(q).Q) \sim C[\boldsymbol{\nu} x\ (P \mid !x(q).Q)]\,.$$

Proof.

(1) See Lemma 3.14 in [San00].
(2) By structural induction on C and using Lemma 3.14 in [San00]. □

Lemma 44 (Validity of β-reduction) *For any M and N, $[\![(\lambda x.\, M)\ N]\!]_q \gtrsim [\![M[N/x]]\!]_q$.*

Proof. By remarking that the process $[\![(\lambda x.\ M)\ N]\!]_p$ deterministically reduces to $(\boldsymbol{\nu}\, x, l)\ ([\![M]\!]_p \mid !x(q).[\![N]\!]_q)$, we obtain:

$$[\![(\lambda x.\ M)\ N]\!]_p \gtrsim (\boldsymbol{\nu}\, x)\ ([\![M]\!]_p \mid !x(q).[\![N]\!]_q)\ . \tag{1}$$

We then prove by induction on M that:

$$\boldsymbol{\nu} x\ ([\![M]\!]_p \mid !x(q).[\![N]\!]_q) \gtrsim [\![M[N/x]]\!]_p\,. \tag{2}$$

We only consider the case where $M = P_1 \,[]\, P_2$ (the other cases are treated in [San00]). In this case, (2) becomes:

$$\boldsymbol{\nu} q\ \underbrace{\boldsymbol{\nu} x\ ([\![P_1]\!]_q \mid [\![P_2]\!]_q \mid q \rightarrowtriangle p \mid !x(n).[\![N]\!]_n)}_{\mathcal{E}} \gtrsim [\![P_1[N/x] \,[]\, P_2[N/x]]\!]_p\,. \tag{2'}$$

According to Lemma 43, we have:

$$\mathcal{E} \sim \boldsymbol{\nu} x\ ([\![P_1]\!]_q \mid !x(n).[\![N]\!]_n) \mid \boldsymbol{\nu} x\ ([\![P_2]\!]_q \mid !x(n).[\![N]\!]_n) \mid q \rightarrowtriangle p\,.$$

By applying the induction hypothesis to P_1 and P_2, we obtain:

$$\mathcal{E} \gtrsim [\![P_1[N/x]]\!]_q \mid [\![P_2[N/x]]\!]_q \mid q \rightarrowtriangle p\,.$$

Using the fact that $\gtrsim$ is a congruence, we can establish (2') from the above equation. Finally, combining (1) and (2), we obtain the desired result. $\square$

We now prove a *one step* operational correspondence property.

Proposition 45 *For any closed terms M and N,*

(1) if $M \hookrightarrow N$ then $[\![M]\!]_q \stackrel{\tau}{\rightarrow}^{\star} \gtrsim [\![N]\!]_q$;
(2) if $[\![M]\!]_q \stackrel{\tau}{\rightarrow} P$ then there exists N such that $M \hookrightarrow^{\star} N$ and $P \gtrsim [\![N]\!]_q$.

Proof.

(1) By induction on the derivation tree of $M \hookrightarrow N$.
(2) By induction on the structure of M. $\square$

From Proposition 45, we can easily derive operational correspondence.

Proposition 46 (Operational correspondence) *For any closed terms M and N,*

(1) if $M \hookrightarrow^{+} N$ then $[\![M]\!]_q \stackrel{\tau}{\rightarrow}^{+} \gtrsim [\![N]\!]_q$;
(2) if $[\![M]\!]_q \stackrel{\tau}{\rightarrow}^{+} P$ then there exists N such that $M \hookrightarrow^{\star} N$ and $P \gtrsim [\![N]\!]_q$.

Proof.

(1) By induction on the length of $M \hookrightarrow^{+} N$ and using Prop. 45.
(2) By induction on the length of $[\![M]\!]_q \stackrel{\tau}{\rightarrow}^{+} P$ and using Prop. 45. $\square$

From Proposition 46, we derive soundness of our encoding.

Lemma 47 (Soundness lemma) *For all closed term M,*

$$(M\Downarrow \;\Leftrightarrow\; [\![M]\!]_p \Rightarrow \xrightarrow{\boldsymbol{\nu} l\, \bar{p}\langle l\rangle}) \quad \textit{and} \quad (M\uparrow \;\Leftrightarrow\; [\![M]\!]_p \Rightarrow \approx \mathbf{0})\,.$$

From this result, and using the compositionality of our encoding, we deduce:

Theorem 48 (Soundness) *For all terms M and N, $[\![M]\!]_p \cong_{\pi} [\![N]\!]_p$ implies $M \cong_{\lambda} N$.*

$$M \,[\!]\, N \cong_\lambda N \,[\!]\, M \qquad (M \,[\!]\, N) \,[\!]\, P \cong_\lambda M \,[\!]\, (N \,[\!]\, P)$$

$$(\lambda x.\, M)\, N \cong_\lambda M[N/x] \qquad M \,[\!]\, \Omega \cong_\lambda M \qquad V \oplus V' \cong_\lambda V \,[\!]\, V'$$

$$(M \oplus N) \oplus P \cong_\lambda M \oplus (N \oplus P) \qquad M \,[\!]\, M \cong_\lambda M \text{ for } M \text{ closed}$$

Fig. 10. Some properties of *amb*

Proof. Suppose that $[\![M]\!]_p \cong_\pi [\![N]\!]_p$. As $[\![\cdot]\!]$ is compositional, for any closing $\lambda^{\Vdash}$context $C[]$ there exists a π-calculus context $C_\pi^{q,p}[\cdot]$ such that $[\![C[M]]\!]_q = C_\pi^q[[\![M]\!]_p]$ and $[\![C[N]]\!]_q = C_\pi^q[[\![N]\!]_p]$. As $\cong_\pi$ is a congruence, we have for any closing context C, $[\![C[M]]\!]_p \cong_\pi [\![C[N]]\!]_p$. From the definition of $\cong_\pi$, we have:

$$([\![C[M]]\!]_p \Rightarrow \xrightarrow{\boldsymbol{\nu} l\, \bar{p}\langle l\rangle} \;\Leftrightarrow\; [\![C[N]]\!]_p \Rightarrow \xrightarrow{\boldsymbol{\nu} l\, \bar{p}\langle l\rangle})$$
$$\text{and} \quad ([\![C[M]]\!]_p \Rightarrow \approx \mathbf{0} \;\Leftrightarrow\; [\![C[N]]\!]_p \Rightarrow \approx \mathbf{0})\,.$$

Using Lemma 47, we obtain that $C[M]\Downarrow \Leftrightarrow C[N]\Downarrow$ and $C[M]\!\uparrow \Leftrightarrow C[N]\!\uparrow$. Finally, $M \cong_\lambda N$. □

4.2 Deriving characteristic properties of amb

4.2.1 Some laws

Figure 10 presents a set of laws regarding *amb* that we have been able to establish. The proofs of these results are all based on the same method: we compute the $A\pi$ encoding of the two $\lambda^{\Vdash}$-terms being compared, construct a (weak or coupled) bisimulation to show (possibly using up-to techniques and algebraic laws for $A\pi$) that these processes are related by $\cong_\pi$, and conclude using Theorem 48.

We give an illustration of our method for the bottom avoidance property $M \,[\!]\, \Omega \cong_\lambda M$, one of the key fairness properties of *amb*. We first need a technical result:

Lemma 49 *For any M and p, $[\![M]\!]_p \approx \boldsymbol{\nu} q\ ([\![M]\!]_q \mid q \rightarrowtriangle p)$.*

Proof. See [MS98].

This is now how we show that for any term M, $M \parallel \Omega \cong_\lambda M$:

$$\begin{aligned}
[\![M \parallel \Omega]\!]_p &\stackrel{\mathsf{def}}{=} \boldsymbol{\nu} q\ ([\![M]\!]_q \mid [\![\Omega]\!]_q \mid q \rightarrowtriangle p) \\
&\approx \boldsymbol{\nu} q\ ([\![M]\!]_q \mid q \rightarrowtriangle p) && \text{because } [\![\Omega]\!]_q \approx \mathbf{0} \\
&\approx [\![M]\!]_p && \text{using Lemma 49}
\end{aligned}$$

The proofs of the other laws illustrate several variations on the method we just showed. Some of these require the introduction of new notions, presented below, and we therefore defer the explanation of these to the end of this subsection, where we also discuss how these laws compare to the existing work about $\lambda^{\parallel}$. It has to be stressed however that in exploiting these techniques, the general framework remains the same, which shows the uniformity of our approach.

4.3 Derived techniques

We present here three techniques that can in some cases simplify the proofs when reasoning about $A\pi$ processes resulting from the encoding of $\lambda^{\parallel}$ terms.

4.3.1 A relaxed encoding

The encoding we have presented uses a link process to embed the choice made by *amb* once a component has converged. A similar mechanism is at work in the encoding of application (cf. Fig. 9), where name q is used linearly to make the connection between a function and its argument. We may in certain cases use an alternative encoding, written $[\![\_]\!]'$, in which we exploit this observation and translate $\parallel$ simply as parallel composition in $A\pi$. $[\![\_]\!]'$ is defined like $[\![\_]\!]$, except for the following clause:

$$[\![M \parallel N]\!]'_q \quad \stackrel{\mathsf{def}}{=} \quad [\![M]\!]'_q \mid [\![N]\!]'_q \,.$$

We can prove the following law, that captures the behaviour discussed above:

$$[\![(M \parallel N)\ P]\!]'_p \approx$$

$$\boldsymbol{\nu} q, q'\ \left([\![M]\!]'_{q'} \mid [\![N]\!]'_{q'} \mid q' \rightarrowtriangle q \mid q(l).\left(\boldsymbol{\nu} x\ \left(\bar{l}\langle x, p\rangle \mid\ !x(r).[\![P]\!]'_r\right)\right)\right) .$$

The evaluation strategy we implement through this encoding can be described by a slight modification of relation $\hookrightarrow$ (Definition 23), in which rule IMM would be replaced by:

$$\textsc{Imm} \quad (V \parallel N)\ M \ \hookrightarrow\ V\, M \,.$$

Obviously, encoding $[\![\_]\!]'$ is not in general operationally faithful w.r.t. the definition of *amb*, since the translation of a term having *amb* as topmost constructor behaves like the parallel execution of the two components, with a total absence of choice. Still, when applicable, $[\![\_]\!]'$ defines a sound proof technique:

Proposition 50 *For any terms M, N, and for any p, if $[\![M]\!]'_p \cong_\pi [\![N]\!]'_p$, then $M \cong_\lambda N$.*

The proof of this result goes as for Theorem 48.

Due to its simplicity, encoding $[\![\_]\!]'$ sometimes leads to much simpler proofs, e.g. when establishing commutativity and associativity of *amb*, that follow directly from the corresponding properties of $|$ for $\equiv$.

4.3.2 'Kleene equivalence'

We can also use the π-calculus encoding to derive proof techniques similar to those used in the literature to establish the laws of $\lambda^{\|}$ [Mor98,LM99]. We start by introducing a technique that is similar to the *'Kleene equivalence'* technique of [LM99].

Definition 51 ($\asymp$) *For two $\lambda^{\|}$ terms M and N, $M \asymp N$ iff*

(i) if $M \hookrightarrow^+ V$, there exists V' s.t. $N \hookrightarrow^+ V'$ and, for any p, $[\![V]\!]_p \leftrightarrows [\![V']\!]_p$;
(ii) $M \downarrow$ iff $N \downarrow$.

Proposition 52 (Soundness of $\asymp$) *For any M and N, $M \asymp N$ implies $M \cong_\lambda N$.*

Proof. Let M and N be two terms such that $M \asymp N$, and let p be a π-calculus name. We call $\left(V_i^M\right)_{i \in I_M}$ (resp. $\left(V_i^N\right)_{i \in I_N}$) the values reachable from M (resp. N). By hypothesis, there exists a function ϕ_M from I_M to I_N such that $[\![V_i^M]\!]_p \leftrightarrows [\![V^N_{\phi^M(i)}]\!]_p$. Given i, let us call $(\mathcal{M}_1^i, \mathcal{M}_2^i)$ the corresponding coupled bisimulation. We define ϕ^N and the $(\mathcal{N}_1^i, \mathcal{N}_2^i)$s along the same lines.

We want to prove that $[\![M]\!]_p \leftrightarrows [\![N]\!]_p$. We apply Prop. 18 to show that $(\mathcal{S}_1, \mathcal{S}_2)$ is a coupled bisimilarity *up to expansion*, $\mathcal{S}_1$ and $\mathcal{S}_2$ being defined as follows:

$$
\begin{array}{lcll}
\mathcal{S}_1 & \stackrel{\mathsf{def}}{=} & \{(\mathbf{0},\mathbf{0})\} & (1)\\
& & \cup\ \bigcup_{i\in I_M} \mathcal{M}_1^i & (2)\\
& & \cup\ \bigcup_{i\in I_N} \mathcal{N}_1^i & (3)\\
& & \cup\ \{([\![R]\!]_p, [\![N]\!]_p)\ \mid M \hookrightarrow^* R\} & (4)\\
& & \cup\ \left\{\left([\![V_i^M]\!]_p, [\![K]\!]_p\right)\ \mid i\in I_M \text{ and } N \hookrightarrow^* K \hookrightarrow^* V^N_{\phi^M(i)}\right\} & (5)\\
\mathcal{S}_2 & \stackrel{\mathsf{def}}{=} & \{(\mathbf{0},\mathbf{0})\} & \\
& & \cup\ \bigcup_{i\in I_M} \mathcal{M}_2^i & \\
& & \cup\ \bigcup_{i\in I_N} \mathcal{N}_2^i & \\
& & \cup\ \{([\![M]\!]_p, [\![R]\!]_p)\ \mid N \hookrightarrow^* R\} & \\
& & \cup\ \left\{\left([\![K]\!]_p, [\![V_i^N]\!]_p\right)\ \mid i\in I_N \text{ and } M \hookrightarrow^* K \hookrightarrow^* V^M_{\phi^N(i)}\right\} &
\end{array}
$$

We now study relations $\mathcal{S}_1$ and $\mathcal{S}_2$, in order to establish that $[\![M]\!]_p \leftrightarrows [\![N]\!]_p$.

$\mathcal{S}_1$ is a weak simulation up to expansion. Let P and Q be two processes such that $P\,\mathcal{S}_1\,Q$. If this follows from clause (2) or (3), the simulation property immediately follows from the fact that the $\mathcal{M}_1^i$s and the $\mathcal{N}_1^i$s are simulations. The result is also immediate in case (1).

Suppose now that, according to clause (4), $P = [\![R]\!]_p$, $Q = [\![N]\!]_p$, and $M \hookrightarrow^* R$. If $P \xrightarrow{\tau} P'$ then by operational correspondence there exists R' such that $R \hookrightarrow^* R'$ and $P' \gtrsim [\![R']\!]_p$. We may then take $Q' = Q$, and since the pair $([\![R']\!]_p, [\![N]\!]_p)$ belongs to $\mathcal{S}_1$, we can conclude.

If $P \xrightarrow{\boldsymbol{\nu} x\,\bar{p}(x)} P'$, then for some $i \in I_M$, $P = [\![V_i^M]\!]_p$ (only the encoding of a value can perform a free output). By operational correspondence, as $N \hookrightarrow^* V^N_{\phi^M(i)}$, we have $Q \Rightarrow\gtrsim [\![V^N_{\phi^M(i)}]\!]_p$. Since $([\![V_i^M]\!]_p, [\![V^N_{\phi^M(i)}]\!]_p)$ belongs to $\mathcal{M}_1^i$ and $\mathcal{M}_1^i$ is a weak simulation, there exists Q' such that $P'\mathcal{M}_1^i Q'$ and $[\![V^N_{\phi^M(i)}]\!]_p \stackrel{\boldsymbol{\nu} x\,\bar{p}\langle x\rangle}{\Longrightarrow} Q'$. Now $Q \Rightarrow\gtrsim [\![V^N_{\phi^M(i)}]\!]_p$ and $[\![V^N_{\phi^M(i)}]\!]_p \stackrel{\boldsymbol{\nu} x\,\bar{p}\langle x\rangle}{\Longrightarrow} Q'$ entail that $Q \stackrel{\boldsymbol{\nu} x\,\bar{p}\langle x\rangle}{\Longrightarrow}\gtrsim Q'$. We thus have $Q \stackrel{\boldsymbol{\nu} x\,\bar{p}\langle x\rangle}{\Longrightarrow}\gtrsim Q'$ and $P'\,\mathcal{S}_1\,Q'$, which is enough to conclude.

Clause (5) can be treated similarly.

$\mathcal{S}_1$ satisfies the coupling condition with $\mathcal{S}_2$. Let P and Q be two processes such that $P\,\mathcal{S}_1\,Q$. As above, clauses (1), (2) and (3) are immediate.

Let us now examine clause (4): $Q = [\![N]\!]_p$, $P = [\![R]\!]_p$ and $M \hookrightarrow^* R$.

If $P \gtrsim \mathbf{0}$, then M may strongly diverge. By definition of $\asymp$, it is also the case for N. Therefore, using operational correspondence, $Q = [\![N]\!]_p \Rightarrow\gtrsim \mathbf{0}$. As $\mathbf{0}\,\mathcal{S}_2\,\mathbf{0}$, we can conclude.

If $P \not\gtrsim \mathbf{0}$, then there exists $i \in I_M$ such that $R \hookrightarrow^* V_i^M$. We know that

$N \hookrightarrow^+ V^N_{\phi^M(i)}$ and by operational correspondence, $[\![N]\!]_p \Rightarrow\gtrsim [\![V^N_{\phi^M(i)}]\!]_p$. We are through with this case, since $R\,\mathcal{S}_2\,[\![V^N_{\phi^M(i)}]\!]_p$.

We now examine clause (5): $P = [\![V^M_i]\!]_p$ and $Q = [\![K]\!]_p$ where $N \hookrightarrow^* K \hookrightarrow^* V^N_{\phi^M(i)}$. From operational correspondence, we know that $Q \Rightarrow\gtrsim [\![V^N_{\phi^M(i)}]\!]_p$. By definition of $\mathcal{M}^i_2$, $[\![V^M_i]\!]_p\,\mathcal{M}^i_2\,[\![V^N_{\phi^M(i)}]\!]_p$, which allows us to conclude since $\mathcal{M}^i_2 \subseteq \mathcal{S}_2$.

The definitions of $\mathcal{S}_1$ and $\mathcal{S}_2$ being symmetric, the properties we have just established for $\mathcal{S}_1$ hold for $\mathcal{S}_2$, and $(\mathcal{S}_1, \mathcal{S}_2)$ is a coupled bisimulation up to expansion, and hence, finally, $[\![M]\!]_p \leftrightarrows [\![N]\!]_p$. $\square$

Aside the use of the π-calculus, the main difference with 'Kleene equivalence' as in [LM99] is that, in clause (i), the latter uses syntactic equality to compare V and V', while we can rely on behavioural equivalences (since $\approx \subseteq \leftrightarrows$, we can also use $\approx$ to compare $[\![V]\!]_p$ and $[\![V']\!]_p$ when treating clause (i) above). As an illustration of this difference, Proposition 52 allows us to show that $\lambda x.\,(x\,[\!]\,\Omega) \cong_\lambda I$, which cannot be proved using the technique of [LM99]. More interestingly, perhaps, if we let $C \stackrel{\mathsf{def}}{=} \lambda xy.\,(x\,x\,x)$, and define Ξ' as $C\,C$, the law $\Xi \cong_\lambda \Xi'$ can be proved using $[\![\_]\!]$ together with $\approx$, and cannot be proved using Kleene equivalence. Note that these equalities are not really 'characteristic *amb* laws' – their role is rather to illustrate our point in contrasting the proof techniques.

4.3.3 Unique solution of inequations

The second method defined in [LM99] is based on an adaptation of *cost equivalence* (written $\doteq$, see [San95]) to the setting of $\lambda^{[\![]}$, and introduces a *unique fixpoint induction* principle, expressed by the following inference rule:

$$\frac{M \doteq {}^{\surd}C[M] \quad N \doteq {}^{\surd}C[N]}{M \doteq N} \quad \text{where } C \text{ is a } \lambda^{[\![]} \text{ context.}$$

Here ${}^{\surd}M$ (*"tick M"*) is a term which makes one step of reduction before behaving like M. Cost equivalence is a very fine-grained relation, and this method involves an accurate insertion of 'ticks' in processes, which intuitively amounts to transform a weak bisimulation into a cost-sensitive one. In $A\pi$, we may reason using coarser equivalences, thanks the following principle (we say that a context C is *guarded* when the hole always occurs under at least one prefix in C):

Proposition 53 (Unique solution of inequations) *Let C be a guarded $A\pi$ context. For any P, Q, if $P \gtrsim C[P]$ and $Q \gtrsim C[Q]$, then $P \approx Q$.*

Proof. We are going to show that:

$$S \;=\; \{(D[M], D[N]) \mid D \text{ is an garded context}\}$$

is a weak bisimulation up to expansion.

Let D be a guarded context, if $D[M] \stackrel{\mu}{\to} Q$ then there exists a context D' (not necessarily guarded) such that $Q = D'[M]$. As $\gtrsim$ is a congruence on $A\pi$, $D'[M] \;\gtrsim\; D'[C[M]]$. The context $D''[\cdot] = D'[C[\cdot]]$ is guarded and we have $D[M] \stackrel{\mu}{\to} \gtrsim D''[M]$.

Reasoning along the same lines with $D[N]$, we obtain $D[N] \stackrel{\mu}{\to} \gtrsim D''[N]$. According to Proposition 16, we have that $S \subset \approx$. In particular, it holds that $\boldsymbol{\nu} c\ (\bar{c}\langle\rangle \mid c().M) \;\approx\; \boldsymbol{\nu} c\ (\bar{c}\langle\rangle \mid c().N)$. This easily implies $M \;\approx\; N$. $\square$

To our knowledge, this proof method is new in the setting of the π-calculus. It may be used to reason about functions defined by a fixpoint operator, since it allows one to consider one-step unfoldings of the corresponding terms, and validate a form of induction principle. This kind of reasoning is at work in [LM99] on several examples, that can all be revisited in our setting. The fact that we adopt weaker equivalences than in [LM99] suggests that Proposition 53 provides a more powerful proof principle.

We now explain how the laws of Figure 10 are established, using in each case the technique that gives the simplest proof.

- $(M \,[\!]\, N) \,[\!]\, P \cong_\lambda M \,[\!]\, (N \,[\!]\, P)$ and $M \,[\!]\, N \cong_\lambda N \,[\!]\, M$.
  We know that $\equiv$ validates the corresponding laws for the 'relaxed' encoding of these terms. This is enough to conclude using Proposition 50.
- $(\lambda x.\, M)\, N \cong_\lambda M[N/x]$.
  This law is obtained by combining Lemma 44 and Theorem 48.
- $I \;\cong_\lambda\; \mathsf{Fix}\ (\lambda x.\ (I \,[\!]\, x)) \cong_\lambda \mathsf{Fix}\ (\lambda x.\ (I \oplus x))$.
  We easily verify that $I \asymp \mathsf{Fix}\ (\lambda x.\ (I \,[\!]\, x)) \asymp \mathsf{Fix}\ (\lambda x.\ (I \oplus x))$. We conclude using Proposition 52.
- $M \,[\!]\, M \;\cong_\lambda\; M$, for M closed.
  We verify that $M \,[\!]\, M \;\asymp\; M$ and we conclude using Prop. 52.
- $(M \oplus N) \oplus P \;\cong_\lambda\; M \oplus (N \oplus P)$.
  For this proof, we are compelled to reason with $\leftrightarrows$ (and not $\approx$), because of the presence of 'partially committed states' in the execution of choices.

We comment on the laws we prove and compare our setting with related works. As mentioned in Section 1, *amb* has been originally introduced to reason over partial functions. In that setting, the distinction between strong and weak divergences does not really make sense, and the characteristic laws

of *amb* are thus just $M \,[\!]\, \Omega = M$ (bottom avoidance), $V \,[\!]\, V' = V \oplus V'$, and $M \,[\!]\, N = N \,[\!]\, M$ (to express the fact that no branch of an *amb* has priority w.r.t. the other).

When we move to the more accurate description given by $\lambda^{[\!]}$, and refer to the fair operational semantics proposed by Lassen and Moran in [LM99] (cf. Definition 2), we can express more precise properties about computation using *amb*. In particular, we can consider that a law like $\mathsf{Fix}\ (\lambda x.(x \,[\!]\, I)) = I$ belongs to *amb*'s specification in that framework. This is also the case for $\mathsf{Fix}\ (\lambda x.(x \oplus I)) = I \oplus \Omega$.

While we can prove the former law in our framework, the latter stresses the difference between our approach and the setting of [LM99]. When focusing on strong divergences, we have $\mathsf{Fix}\ (\lambda x.(x \oplus I)) = I$ (and, of course, $I \neq I \oplus \Omega$), intuitively because by neglecting weak divergences, we impose more fairness than [LM99]'s operational semantics. Our semantics, while remaining operationally sound w.r.t. the existing descriptions of $\lambda^{[\!]}$, can be deemed as 'avoiding more divergences' than Lassen and Moran's.

4.4 Full abstraction.

The method we exploit in the π-calculus is not fully abstract with respect to $\cong_\lambda$. To understand why, we discuss the treatment of open terms in our setting.

From this point of view, the law $M \,[\!]\, M \cong_\lambda M$ in Figure 10 deserves further attention. It says that *amb* is somehow insensitive to the replication of M, a kind of property that usually does not hold for bisimulation. This is the reason why we have been able to establish this result for M *closed* only. Indeed, when M reduces to a term having a variable x in head position, the encodings of M and $M \,[\!]\, M$ are able to make respectively one and two emissions on x, and are thus separated by (weak or coupled) bisimulation. We have not been able to find an extension of our methods in order to tackle this question in a simple way. However, note that this problem is related to the difficulty of handling multiplicities using bisimulation, which is well-known, rather than to the specific treatment of open terms (as a matter of fact, the two resulting $A\pi$ processes do not even simulate each other). In [Mor98,LM99], open terms are dealt with by applying closing substitutions, while we exploit the compositionality of our techniques, which allows us for instance to compare directly M and N when we have to test equivalence between $\lambda x.\, M$ and $\lambda x.\, N$.

Due to this problem with the analysis of open terms, our method is not fully-abstract with respect to $\cong_\lambda$. We can however derive a *partial full-abstraction* result (partial in the sense that we only compare pure λ-terms – see Theorem 55 below), for the 'open' version of applicative bisimilarity (see [SW01,

$$[\![\lambda x.M]\!]_p \stackrel{\mathsf{def}}{=} \boldsymbol{\nu} l\ (\bar{p}\langle l\rangle \mid !l(x,q).[\![M]\!]_q)$$
$$[\![\mathsf{let}\ x = M\ \mathsf{in}\ N]\!]_p \stackrel{\mathsf{def}}{=} \boldsymbol{\nu} q\ ([\![M]\!]_q \mid q(v).([\![N]\!]_p \mid !x(r).\bar{r}\langle v\rangle))$$

Fig. 11. π-calculus encoding of $\lambda^{[\!]}$ with local call-by-value

Part VI]). This relation, written $\cong_\lambda^{\mathsf{op}}$, is defined by extending relation $\twoheadrightarrow$ to open terms, and by saying that a term having a free variable in head position is stuck (for example, $x\,[\!]\,\Omega$ cannot diverge). In the following definition, we keep the same notation $\twoheadrightarrow$ for the extended version of the operational semantics.

Definition 54 (Open applicative bisimilarity) *$\cong_\lambda^{op}$ is the largest symmetric relation on $\lambda^{[\!]}$ such that, whenever $M \cong_\lambda^{op} N$,*

(i) $M \twoheadrightarrow \lambda x.\, M'$ implies $N \twoheadrightarrow \lambda x.\, N'$ with $M' \cong_\lambda^{op} N'$;
(ii) $M \twoheadrightarrow x\, M_1 \dots M_n$ with $n \geq 0$ implies $N \twoheadrightarrow x\, N_1 \dots N_n$ and $M_i \cong_\lambda^{op} N_i$ for all $1 \leq i \leq n$.

Theorem 55 (Partial full abstraction) *Let M, N be two $\lambda^{[\!]}$ terms with no occurrence of $[\!]$, and let p be a name. Then*

$$[\![M]\!]_p \approx [\![N]\!]_p \qquad \text{iff} \qquad M \cong_\lambda^{op} N\,.$$

Proof. Along the lines of [SW01]. □

It can be noted that for the λ-calculus extended with internal choice, the problem of full abstraction on the whole calculus (i.e., whether the π-calculus encoding is fully abstract w.r.t. open applicative bisimilarity) is still open. The same question in $\lambda^{[\!]}$ seems at least as difficult.

4.5 Local call by value

An important enrichment of $\lambda^{[\!]}$ is that with the familiar $\mathsf{let}\ldots\mathsf{in}$ construction, that introduces a form of *local call by value* in the language. The corresponding additional reduction rule is:

$$\textsc{Let}\ \frac{M \hookrightarrow V}{\mathsf{let}\ x = M\ \mathsf{in}\ N \hookrightarrow N[V/x]}$$

The encoding of the resulting calculus is obtained by a modification of the encoding presented above, as shown on Figure 4.5 (clauses that are left unchanged are not mentioned). The translation of a $\mathsf{let}\ldots\mathsf{in}$ construct consists

in the evaluation of the locally declared term, *followed by* the evaluation of the term after the 'in' in which the bound variable is replaced by the computed value. We also add *persistence*, using replication, in the encoding of abstractions, since in presence of let...in, several copies of a function may be triggered along a computation.

The correspondence proved in Proposition 24 between $\twoheadrightarrow$ and $\hookrightarrow$ is still valid with the addition of rule LET to the calculus. We can therefore use our encoding to reason about $\twoheadrightarrow$ in the calculus extended with local call-by-value. The results presented in previous sections also hold on $\lambda^{[\![}$ with let. In particular, soundness becomes:

Theorem 56 *For any terms M, N of $\lambda^{[\![}$ enriched with local call-by-value, and for any name p, $[\![M]\!]_p \cong_\pi [\![N]\!]_p$ implies $M \cong_\lambda N$.*

Again, using simple bisimulation reasoning, we are able to derive the following example laws for $\lambda^{[\![}$ with let:

$$\mathsf{let}\ x{=}V\ \mathsf{in}\ M \ \cong_\lambda\ (\lambda x.\, M)\, V \qquad \mathsf{let}\ x{=}M\ \mathsf{in}\ x \,[\!]\, x \ \cong_\lambda\ M \ \text{ for } M \text{ closed}$$

$$\mathsf{let}\ x{=}\Omega\ \mathsf{in}\ M \ \cong_\lambda\ \Omega \qquad \mathsf{let}\ x{=}M\ \mathsf{in}\ N \ \cong_\lambda\ N \ \text{ if } M \Downarrow \text{ and } x \notin \mathsf{fn}(N)$$

5 Conclusion

In the present work, we have distinguished strong and weak divergences, and shown that only strong divergences should be considered in order to define a semantics for $\lambda^{[\![}$ using the π-calculus. We think that both resulting semantics – the one where both strong and weak divergences are observed, and the one where only strong divergences are relevant – are interesting. However, one may argue that in languages with operators like *amb*, a general fairness requirement that a computation should not 'always miss a reachable value' – obtained by taking only strong divergences into account – appears more reasonable (for instance, a computation starting from the term T in Section 1 should not keep discarding the value I).

Existing extensions of the λ-calculus with parallel operators include [DCdP98] and [Bou94]. These works are concerned with 'parallel', rather than 'choice' operators, and do not address the issues related to fairness brought up in the study of *amb*. Indeed, semantically, the operators of [DCdP98,Bou94] are much simpler than *amb* (their encoding into the π-calculus is straightforward, see [SW01]).

Some of the laws we have established express *amb*'s fairness in $A\pi$, and are derived in our setting by exploiting bisimulation. It would be interesting to go

further in this direction in order to gain a better understanding of the fairness brought by bisimulation. A way to do this is to study the π-calculus semantics of other fair operators, like e.g. *fair merge*, which is more expressive than *amb* [PS88,FK02]. This operator computes the merge of two (finite or infinite) lists in a fair fashion, also in the case when the lists contain divergences. It is possible to adapt an argument of [PS88] to prove that one cannot represent fair merge into the π-calculus at an operational level. An interesting question is the definability of fair merge *modulo bisimulation*, i.e., at a behavioural level.

Acknowledgements. We would like to thank Mariangiola Dezani, Emmanuel Jeandel and Pascal Koiran for useful discussions. The help of anonymous referees is also acknowledged.

References

[Abr90] S. Abramsky. The lazy lambda calculus. In D.A. Turner, editor, *Research Topics in Functional Programming*, pages 65–116. Addison-Wesley, 1990.

[Bil95] P. Billingsley. *Probability and Measure*. J. Wiley & Sons, 1995.

[BL00] G. Boudol and C. Laneve. Lambda-calculus, multiplicities and the pi-calculus. In *Proof, Language, and Interaction, Essays in Honour of Robin Milner*. MIT Press, 2000.

[Bou94] G. Boudol. Lambda-calculi for (strict) parallel functions. In *Information and Computation*, volume 108, pages 51–127, 1994.

[CHS03] A. Carayol, D. Hirschkoff, and D. Sangiorgi. On the representation of McCarthy's *amb* in the π-calculus. In *Proc. EXPRESS'03*, volume 96 of *Electronic Notes in Theoretical Computer Sciences*, pages 73–89. Elsevier, 2003.

[DCdP98] M. Dezani-Ciancaglini, U. de'Liguoro, and A. Piperno. A filter model for concurrent λ-calculus. *SIAM J. Comput.*, 27(5):1376–1419, 1998.

[FK02] M. Fernandez and L. Khalil. Interaction Nets with McCarthy's amb. In *Proc. EXPRESS'02*, volume 68(2) of *Electronic Notes in Theoretical Computer Science*. Elsevier, 2002.

[Hen82] P. Henderson. Purely functional operating systems. In *Functional Programming and its Applications*, pages 177–192. Cambridge Univ. Press, 1982.

[HO90] J. Hughes and J. O'Donnell. Expressing and reasoning about non-deterministic functional programs. In *Proc. Glasgow 1989 Workshop on Functional Programming*, Workshops in Computing. Springer Verlag, 1990.

[How96] D. J. Howe. Proving congruence of bisimulation in functional programming languages. *Information and Computation*, 124(2):103–112, 1996.

[Las98] S. Lassen. *Relational reasoning about functions and nondeterminism.* PhD thesis, Aarhus University, 1998.

[LM99] S. Lassen and A. Moran. Unique fixed point induction for McCarthy's Amb. In *Proc. of MFCS'99*, volume 1672 of *Lecture Notes in Computer Science*, pages 198–208. Springer Verlag, 1999.

[McC63] J. McCarthy. A basis for a mathematical theory of computation. In *Computer Programming and Formal Systems*, pages 33–70. North-Holland, 1963.

[Mil90] R. Milner. Functions as processes. Technical Report 1154, INRIA Sophia Antipolis, 1990.

[Mor98] A. Moran. *Call-by-name, Call-by-need, and McCarthy's Amb.* PhD thesis, Chalmers Univ. of Technology and Univ. of Gothenburg, 1998.

[MS98] M. Merro and D. Sangiorgi. On asynchrony in name-passing calculi. In *Proc. ICALP'98*, volume 1443 of *Lecture Notes in Computer Science*, pages 856–867. Springer Verlag, 1998.

[NC95] V. Natarajan and R. Cleaveland. Divergence and fair testing. In *Proc. ICALP 95*, volume 944 of *Lecture Notes in Computer Science*, pages 648–659. Springer Verlag, 1995.

[NP96] U. Nestmann and B. Pierce. Decoding choice encodings. In *International Conference on Concurrency Theory*, pages 179–194, 1996.

[Pit01] C. Pitcher. *Functional programming and erratic non-determinism.* PhD thesis, Oxford University, 2001.

[PS88] P. Panangaden and V. Shanbhogue. McCarthy's amb cannot implement fair merge. In *Proc. of FST-TCS*, volume 338 of *Lecture Notes in Computer Science*, pages 348–363. Springer Verlag, 1988.

[San92] D. Sangiorgi. The lazy lambda calculus in a concurrency scenario. In *Proc. 7th LICS Conf.*, pages 102–109. IEEE Computer Society Press, 1992.

[San95] D. Sands. A naïve time analysis and its theory of cost equivalence. *Journal of Logic and Computation*, 5(4):495–541, 1995.

[San00] D. Sangiorgi. Lazy functions and mobile processes. In *Proof, Language, and Interaction, Essays in Honour of Robin Milner.* MIT Press, 2000.

[SM92] D. Sangiorgi and R. Milner. The problem of "Weak Bisimulation up to". In *Proc. CONCUR '92*, volume 630 of *Lecture Notes in Computer Science*, pages 32–46. Springer Verlag, 1992.

[SW01] D. Sangiorgi and D. Walker. *The π-calculus: a Theory of Mobile Processes*. Cambridge University Press, 2001.

[Tur90] D. Turner. An approach to functional operating systems. In *Research Topics in Functional Programming*. Addison Wesley, 1990.