A Uniform Syntactical Method for Proving Coinduction Principles in λ-calculi*

Marina Lenisa

Dipartimento di Matematica e Informatica, Università di Udine, Italy.
lenisa@dimi.uniud.it

Abstract. *Coinductive* characterizations of various *observational congruences* which arise in the semantics of λ-calculus, when λ-terms are evaluated according to various reduction strategies, are discussed. We analyze and extend to *non-lazy strategies*, both deterministic and non-deterministic, Howe's *congruence candidate method* for proving the coincidence of the applicative (bisimulation) and the contextual equivalences. This *purely syntactical* method is based itself on a *coinductive* argument.

Introduction

This paper is part of a general project aiming at finding *elementary proof principles* for reasoning *rigorously* on *infinite* computational objects, see [4, 9] for the case of higher order functions, and [8] for the case of higher order processes. In this paper, as in [4, 9], we focus on the behaviour of λ-terms when these are evaluated according to various *reduction strategies*. We address the problem of showing the coincidence of the *applicative* (bisimulation) equivalence with the *observational* (operational, contextual) equivalence for various reduction strategies, thus deriving a *coinduction principle* for establishing obsevational equivalences. In particular, in this paper we analyze and generalize to *non-lazy* strategies the *purely syntactical* method originally introduced by Howe ([6, 7]) for lazy functional languages. We call this method *congruence candidate method*.

A *reduction strategy* is a procedure for determining, for each λ-term, a specific β-redex in it, to contract. Let $\Lambda(C)$ ($\Lambda^0(C)$) denote the set of (closed) λ-terms, where C is a set of base constants. When $C = \emptyset$, we write Λ (Λ^0). A (possibly non-deterministic) strategy can be formalized as a relation $\rightarrow_\sigma \subseteq \Lambda(C) \times \Lambda(C)$ ($\Lambda^0(C) \times \Lambda^0(C)$) such that, if $(M,N) \in \rightarrow_\sigma$ (also written infix as $M \rightarrow_\sigma N$), then N is a possible result of applying $\rightarrow_\sigma$ to M. The set of terms which do not belong to the domain of $\rightarrow_\sigma$ are partitioned into two disjoint sets: the set of *σ-values*, denoted by Val_σ, and the set of *σ-deadlocks*. Given $\rightarrow_\sigma$, we can define the *evaluation relation* $\Downarrow_\sigma \subseteq \Lambda(C) \times \Lambda(C)$ ($\Lambda^0(C) \times \Lambda^0(C)$), such that $M \Downarrow_\sigma N$ holds if and only if there exists a (possible empty) reduction path from M to a σ-value N. If there exists N such that $M \Downarrow_\sigma N$, then $\rightarrow_\sigma$ *halts successfully* on M and M converges ($M \Downarrow_\sigma$), otherwise $\rightarrow_\sigma$ does not terminate on M or reaches a *deadlock* from M, and M diverges ($M \not\Downarrow_\sigma$). Each reduction strategy induces an

* Work supported by HCM Contract No. CHRX-CT92.0046 Lambda Calcul Typé,

operational semantics, in that we can imagine a machine which evaluates terms by implementing the given strategy. The *observational equivalence* arises if we consider programs as *black boxes* and only observe their "halting properties".

Definition 1 (σ-observational Equivalence). Let $\rightarrow_\sigma$ be a reduction strategy and let $M, N \in \Lambda^0(C)$. The *observational equivalence* $\approx_\sigma$ is defined by
$M \approx_\sigma N$ iff $\forall C[\;].(C[M], C[N] \in \Lambda^0(C) \Rightarrow (C[M] \Downarrow_\sigma \Leftrightarrow C[N] \Downarrow_\sigma))$.

Showing σ-equivalences by induction on computation steps is difficult. Powerful proof-principles, allowing to factorize this difficult task, are precious. Coinduction principles for establishing $\approx_\sigma$ follow from the fact that $\approx_\sigma = \approx_\sigma^{app}$, where $\approx_\sigma^{app}$ denotes the *applicative equivalence* induced by $\rightarrow_\sigma$ (see Definition 2). It is interesting to notice that these two equivalences do not coincide for all strategies, see [9] for counterexamples.

The proof of $\approx_\sigma \supseteq \approx_\sigma^{app}$ can be factorized into two steps:
1. $\approx_\sigma^{app}$ is a congruence w.r.t. application;
2. $\approx_\sigma^{app}$ is a congruence w.r.t. λ-abstraction.

In many cases step 2 is not difficult to prove, while step 1 is in general problematic to show, and requires a specific technique. In this paper, we discuss the *congruence candidate method* for proving step 1. This method was originally introduced for the lazy call-by-name reduction strategy in [6], and later generalized to a class of *lazy* strategies by-name and by-value in [7]. Here we extend the method so as to deal with *non-lazy* strategies, both deterministic and non-deterministic, whose evaluation relation needs to be defined on the whole set of λ-terms and hence it has to deal also with reduction of open terms. The congruence candidate method is based on the definition of a "candidate relation", which is a congruence w.r.t. application, and which extends $\approx_\sigma^{app}$. Reasoning by coinduction, one shows that this relation coincides with $\approx_\sigma^{app}$; hence $\approx_\sigma^{app}$ is itself a congruence w.r.t. application. This method can be applied successfully to various reduction strategies in the literature, thus providing alternative proofs to those in [9], to the conjectures in [4].

In this paper we use λ-calculus concepts and notation as defined in [2, 4]. The paper is organized as follows. In Section 1 we introduce the problem of characterizing coinductively contextual equivalences via applicative equivalences. In Section 2 we present a list of strategies. In Section 3 we present in general the congruence candidate method, and we derive a proof of $\approx_\sigma = \approx_\sigma^{app}$ for all the strategies of Section 2. Final remarks appear in section 4.

The author is grateful to F. Honsell and A. Pitts for useful discussions.

1 Coinductive Characterizations via Applicative Equivalences

Given a reduction strategy $\rightarrow_\sigma$, the *σ-applicative equivalence*, $\approx_\sigma^{app}$, is defined by testing programs only on applicative (closed) contexts. It is reminiscent of bisimilarity in concurrent languages ([1]).

Definition 2. Let $\approx_\sigma^{app} \subseteq \Lambda^0(C) \times \Lambda^0(C)$ be the *applicative equivalence*:
$M \approx_\sigma^{app} N \;\Leftrightarrow\; \forall P_1, \ldots, P_n \in \Lambda^0(C),\ n \geq 0.\ (MP_1 \ldots P_n \Downarrow_\sigma \Leftrightarrow\ NP_1 \ldots P_n \Downarrow_\sigma)$.

The equivalence $\approx_\sigma^{app}$ has a coinductive characterization:

Lemma 1. *The applicative equivalence $\approx_\sigma^{app}$ can be viewed as the greatest fixed point of the monotone operator $\Psi_\sigma : \mathcal{P}(\Lambda^0(C) \times \Lambda^0(C)) \to \mathcal{P}(\Lambda^0(C) \times \Lambda^0(C))$*
$\Psi_\sigma(R) = \{(M, N) \mid (M \not\Downarrow_\sigma\ \wedge N \not\Downarrow_\sigma\ \wedge \forall P \in \Lambda^0(C).\ (MP, NP) \in R) \vee$
$(M \Downarrow_\sigma\ \wedge N \Downarrow_\sigma\ \wedge \forall P \in \Lambda^0(C).\ (MP, NP) \in R)\}$.

An immediate consequence is the validity of the *coinduction principle*:

$$\frac{(M, N) \in R \quad R \text{ is a } \Psi_\sigma\text{-bisimulation}}{M \approx_\sigma^{app} N} \qquad (1)$$

where a *Ψ_σ-bisimulation* is a relation $R \subseteq \Lambda^0(C) \times \Lambda^0(C)$ s.t. $R \subseteq \Psi_\sigma(R)$.

If $\approx_\sigma = \approx_\sigma^{app}$, then the coinduction principle above can be used to establish directly the observational equivalence. Hence the natural question arises: for which strategies σ's do the two equivalences coincide? Notice that there are σ's such that $\approx_\sigma \not\supseteq \approx_\sigma^{app}$, see [9] for counterexamples. However, for many interesting strategies in the literature, one can show that $\approx_\sigma = \approx_\sigma^{app}$, see e.g. [1, 3, 4, 7, 12, 10]. In general, proofs of the coincidence of the two equivalences are rather difficult and apply only to specific strategies. The technique discussed in Section 3 is rather general and it can be used for establishing the coincidence for all the strategies of Section 2.

2 A List of Strategies

In this section we present a list of reduction strategies, together with the corresponding evaluation relations.

$\to_l$ **strategy.** The *lazy call-by-name* strategy $\to_l \subseteq \Lambda^0 \times \Lambda^0$ reduces the *leftmost* β-redex not appearing in a λ-abstraction. $Val_l = \{\lambda x.M \mid M \in \Lambda\} \cap \Lambda^0$. The evaluation $\Downarrow_l$ is the least binary relation over $\Lambda^0 \times Val_l$ satisfying the rules:

$$\frac{}{\lambda x.M \Downarrow_l \lambda x.M} \qquad\qquad \frac{M \Downarrow_l \lambda x.P \quad P[N/x] \Downarrow_l Q}{MN \Downarrow_l Q}$$

Classical β-reduction is correct w.r.t. $\approx_l$[2] (see [1]).

$\to_v$ **strategy.** Plotkin's *lazy call-by-value* strategy $\to_v \subseteq \Lambda^0 \times \Lambda^0$ reduces the *leftmost* β-redex, not appearing within a λ-abstraction, whose argument is a λ-abstraction. $Val_v = \{\lambda x.M \mid M \in \Lambda\} \cap \Lambda^0$. The evaluation $\Downarrow_v$ is the least binary relation over $\Lambda^0 \times Val_v$ satisfying the following rules:

$$\frac{}{\lambda x.M \Downarrow_v \lambda x.M} \qquad\qquad \frac{M \Downarrow_v \lambda x.P \quad N \Downarrow_v Q \quad P[Q/x] \Downarrow_v U}{MN \Downarrow_v U}$$

[2] The β-reduction $\to_{\beta_r}$ is *correct* w.r.t. $\approx_\sigma$ if $M =_{\beta_r} N \implies M \approx_\sigma N$, where $=_{\beta_r}$ is the β_r-conversion.

The notion of β-reduction which is correct w.r.t. $\approx_v$ is the $\to_{\beta_v} \subseteq \Lambda \times \Lambda$, i.e.:
$(\lambda x.M)N \to_{\beta_v} M[N/x]$, if N is a variable or an abstraction.

$\to_o$ **strategy.** Let Ω be a new constant. The non-deterministic strategy $\to_o \subseteq \Lambda^0(\{\Omega\}) \times \Lambda^0(\{\Omega\})$ ([5]) rewrites λ-terms which contain occurrences of the constant Ω by reducing any β-redex. $Val_o = \Lambda^0$. Normal forms which are not in Val_o are the $\to_o$-deadlock terms. The evaluation relation $\Downarrow_o$ is the least binary relation over $\Lambda^0(\{\Omega\}) \times Val_o$ satisfying the following rules:

$$\frac{M \in Val_o}{M \Downarrow_o M} \qquad \frac{C[(\lambda x.M)N] \notin Val_o \quad C[M[N/x]] \Downarrow_o P}{C[(\lambda x.M)N] \Downarrow_o P}$$

β-reduction is trivially correct w.r.t. $\approx_o$.

$\to_h$ **strategy.** The *head call-by-name* strategy $\to_h \subseteq \Lambda \times \Lambda$ reduces the *leftmost* β-redex, if the term is not in head normal form. Val_h is the set of λ-terms in head normal form. The evaluation $\Downarrow_h$ is the least binary relation over $\Lambda \times Val_h$ satisfying the following rules, for $n \geq 0$:

$$\frac{}{xM_1 \ldots M_n \Downarrow_h xM_1 \ldots M_n} \qquad \frac{M \Downarrow_h N}{\lambda x.M \Downarrow_h \lambda x.N} \qquad \frac{M[N/x]M_1 \ldots M_n \Downarrow_h P}{(\lambda x.M)NM_1 \ldots M_n \Downarrow_h P}$$

β-reduction is correct w.r.t. $\approx_h$ (see e.g. [2]).

$\to_n$ **strategy.** The *normalizing* strategy $\to_n \subseteq \Lambda \times \Lambda$ reduces the leftmost β-redex. Val_n is the set of λ-terms in normal form. The evaluation $\Downarrow_n$ is the least binary relation over $\Lambda \times Val_n$ satisfying the following rules, for $n \geq 0$:

$$\frac{M_1 \Downarrow_n M_1' \ \ldots \ M_n \Downarrow_n M_n'}{xM_1 \ldots M_n \Downarrow_n xM_1' \ldots M_n'} \qquad \frac{M \Downarrow_n N}{\lambda x.M \Downarrow_n \lambda x.N} \qquad \frac{M[N/x]M_1 \ldots M_n \Downarrow_n P}{(\lambda x.M)NM_1 \ldots M_n \Downarrow_n P}$$

β-reduction is correct w.r.t. $\approx_n$.

$\to_p$ **strategy.** Barendregt's *perpetual* strategy $\to_p \subseteq \Lambda \times \Lambda$ reduces the leftmost β-redex not in the operator of a redex, which is either an $I\beta$-redex, or a $K\beta$-redex whose argument is a normal form. Val_p is the set of λ-terms in normal form. One can easily show that the evaluation $\Downarrow_p$ is the least binary relation over $\Lambda \times Val_p$ satisfying the following rules, for $n \geq 0$:

$$\frac{M_1 \Downarrow_p M_1' \ \ldots \ M_n \Downarrow_p M_n'}{xM_1 \ldots M_n \Downarrow_p xM_1' \ldots M_n'} \qquad \frac{M \Downarrow_p N}{\lambda x.M \Downarrow_p \lambda x.N} \qquad \frac{N \Downarrow_p \quad M[N/x]M_1 \ldots M_n \Downarrow_p V}{(\lambda x.M)NM_1 \ldots M_n \Downarrow_p V}$$

The reduction $\to_{\beta_{KN}}$, defined as follows, is correct w.r.t. $\approx_p$:
$(\lambda x.M)N \to_{\beta_{KN}} M[N/x]$, if $(\lambda x.M)N$ is either an $I\beta$-redex or a $K\beta$-redex whose argument is a closed normal form.

2.1 General Formats

The above strategies can be grouped under three general formats:

Lazy Strategies. $\to_l$, $\to_v$ can be viewed as special cases of the general format of lazy strategy on a λ-calculus with variables by name and by values (see [6, 7]).

Eager Leftmost Strategies. $\to_h$, $\to_n$, and $\to_p$ are eager in the sense that they reduce under the scope of a λ-abstraction. They can be viewed as special instances of the following general format:

$$\frac{M_{i_1} \Downarrow_\sigma M'_{i_1} \ \ldots \ M_{i_n} \Downarrow_\sigma M'_{i_n}}{xM_1 \ldots M_k \Downarrow_\sigma xM'_1 \ldots M'_k} \ i_1, \ldots, i_n \in \{1, \ldots, k\},\ n \geq 0 \qquad \frac{M \Downarrow_\sigma N}{\lambda x.M \Downarrow_\sigma \lambda x.N}$$

$$\frac{M[N/x]M_1 \ldots M_n \Downarrow_\sigma V \quad (N \Downarrow_\sigma)}{(\lambda x.M)NM_1 \ldots M_n \Downarrow_\sigma V} \ n \geq 0, \quad \text{where } (N \Downarrow_\sigma) \text{ can be omitted.}$$

Non-deterministic Strategies. $\to_o$ can be viewed as a special case of the following general format: let $\emptyset \subset Val \subset \Lambda(\{C\})$ be closed under β-reduction,

$$\frac{M \in Val}{M \Downarrow_\sigma M} \qquad \frac{C[(\lambda x.M)N] \notin Val \quad C[M[N/x]] \Downarrow_\sigma P}{C[(\lambda x.M)N] \Downarrow_\sigma P}$$

Notice that there are many ways to extend $\to_o$ on open terms in order to get a strategy of the above format; we will take the natural one.

3 Showing $\approx_\sigma^{app} = \approx_\sigma$

In this section, we present in detail the congruence candidate method for establishing $\approx_\sigma^{app} = \approx_\sigma$. A special instance of this method was first used by Howe in the case of the lazy call-by-name strategy $\to_l$ ([6]), and later generalized to a class of *lazy* strategies by-name and by-value, including $\to_v$ ([7]). Here we extend Howe's original method so as to deal with more complex strategies, like the eager leftmost strategies, whose evaluation relations cannot be axiomatized only on closed λ-terms, and non-deterministic strategies, such as $\to_o$. The congruence candidate method is used to show that $\approx_\sigma^{app}$ is a congruence w.r.t. application. In fact, in order to prove that $\approx_\sigma^{app} \subseteq \approx_\sigma$, it is sufficient to show (Theorem 4):
1. $\approx_\sigma^{app}$ is a congruence w.r.t. application, i.e. for all $M, N, P, Q \in \Lambda^0(C)$,

$$M \approx_\sigma^{app} N \ \wedge \ P \approx_\sigma^{app} Q \implies MP \approx_\sigma^{app} NQ;$$

2. $\approx_\sigma^{app}$ is a congruence w.r.t. λ-abstraction, i.e., $\forall M, N \in \Lambda(C)$ such that $FV(M, N) \subseteq \{x_1, \ldots, x_n\}$, $\forall P_1, \ldots, P_n \in \Lambda^0(C)$. $M[P_i/x_i] \approx_\sigma^{app} N[P_i/x_i] \implies \lambda x_1 \ldots . x_n.M \approx_\sigma^{app} \lambda x_1 \ldots . x_n.N$.
(In case the strategy is by-value, i.e. for $\sigma = v, p$, $P_1, \ldots, P_n$ are chosen to be convergent terms.)

The congruence of $\approx_\sigma^{app}$ w.r.t. λ-abstraction is immediate to show, once one has proved the Extensionality of $\approx_\sigma^{app}$ (see Theorem 2). This is really problematic only for $\sigma = n$; in this case one needs to exploit extensively the separability technique. For lack of space, we omit this proof.

Theorem 2 (Extensionality of $\approx_\sigma^{app}$). *i) Let $\sigma = l, v$. Let $M, N \in \Lambda^0$ be such that $M \Downarrow_\sigma$, $N \Downarrow_\sigma$. If, for all $P \in \Lambda^0$, $MP \approx_\sigma^{app} NP$, then $M \approx_\sigma^{app} N$.*

ii) Let $\sigma = o, h, p, n$. Let $M, N \in \Lambda^0(C)$. If, for all $P \in \Lambda^0(C)$, $MP \approx_\sigma^{app} NP$, then $M \approx_\sigma^{app} N$.

Using Theorem 2, one can easily show the following theorem:

Theorem 3. *$\approx_\sigma^{app}$ is a congruence w.r.t. λ-abstraction, for $\sigma \in \{l, v, o, h, n, p\}$.*

Proof. We show that, for $M, N \in \Lambda$ s.t. $FV(M,N) \subseteq \{x\}$,
$\forall P \in \Lambda^0(\textit{possibly convergent}).\ M[P/x] \approx_\sigma^{app} N[P/x] \ \Rightarrow\ \lambda x.M \approx_\sigma^{app} \lambda x.N$.
For $\sigma = l, v$ this is immediate. For $\sigma = o, h, n$ the proof follows from the Extensionality Theorem, using the fact that $(\lambda x.M)P \approx_\sigma^{app} M[P/x]$, which in turn follows from the correctness of the β-reduction w.r.t. $\approx_\sigma^{app}$. For $\sigma = p$, the proof follows from the fact that, for all $M \in \Lambda$, $(\exists P \in \Lambda^0.\ M[P/x] \Downarrow_p) \iff M \Downarrow_p$.
The implication ($\Rightarrow$) in this latter fact follows since $\rightarrow_p$ is perpetual. The other implication is proved by computation induction, choosing as P a suitable permutator. □

Theorem 4. *Suppose that $\approx_\sigma^{app}$ is a congruence w.r.t. λ-abstraction and application. Then $\approx_\sigma^{app} \subseteq \approx_\sigma$.*

Proof. We prove by induction on the context $C[\,]$ that:
$$M \approx_\sigma^{app} N \implies \forall C[\,].(C[M], C[N] \in \Lambda(C) \ \wedge\ FV(C[M], C[N]) \subseteq \{x_1, \ldots, x_n\} \Rightarrow \forall P_1 \ldots P_n \in \Lambda^0(C).\ C[M][P_i/x_i] \approx_\sigma^{app} C[N][P_i/x_i]).$$
(In case the strategy is by-value, i.e. $\sigma = v, p$, $P_1, \ldots, P_n$ must be convergent terms.)
□

3.1 The Congruence Candidate Method

The aim of the congruence candidate method is to show that $\approx_\sigma^{app}$ is a congruence w.r.t. application. The main difference between dealing with lazy strategies (whose evaluation relation is axiomatized only on closed λ-terms) and dealing with eager strategies, like $\rightarrow_h, \rightarrow_n, \rightarrow_p$, lies in the fact that, for eager strategies, in order to show that $\approx_\sigma^{app}$ is a congruence w.r.t. application, we need to assume that $\approx_\sigma^{app}$ is a congruence w.r.t. λ-abstraction. This hypothesis is not needed for the lazy strategies considered in [7]. A further special generalization of the proof is required for non deterministic strategies, like $\rightarrow_o$. In fact, the proof of the main proposition in Howe's method proceeds by induction on the length of the derivation of a suitable evaluation judgement, just as we do in the proof of the main proposition for the deterministic strategies in this paper (Propositions 8 and 9). The same result for non deterministic strategies, on the other hand, has to be obtained by induction on the minimal length of a converging path (Proposition 12).

The congruence candidate method is a syntactical method which nonetheless is quite uniform and modular. It makes essential use of the coinduction principle (1) of Section 1, and it is based on the definition of a *candidate relation*, which is a congruence w.r.t. application, and which extends $\approx_\sigma^{app}$. The aim is to show that the candidate relation is a Ψ_σ-bisimulation; hence the coinduction principle (1) guarantees that $\approx_\sigma^{app}$ itself is a congruence w.r.t. application. For the reader's convenience, we outline the:

General pattern of the congruence candidate method:

- Build a *candidate relation* $\widehat{\approx}_\sigma^{app} \subseteq \Lambda(C) \times \Lambda(C)$ s.t.
  1. $\widehat{\approx}_\sigma^{app} \supseteq \approx_\sigma^{app}$;
  2. $\widehat{\approx}_\sigma^{app}$ is a congruence w.r.t. application;
  3. $(\widehat{\approx}_\sigma^{app})_{|\Lambda^0(C)\times\Lambda^0(C)}$ is a Ψ_σ-bisimulation.
- Use the coinduction principle (1) to deduce that $\approx_\sigma^{app}$ is a congruence w.r.t. application.

More in detail, the congruence candidate method proceeds as follows. First of all, we have to explain how to build the *candidate relation* $\widehat{\approx}_\sigma^{app}$. Candidate relations are defined in terms of the extensions to open terms of Ψ_σ-bisimulations:

Definition 3. Let $\eta \subseteq \Lambda^0(C) \times \Lambda^0(C)$ be a Ψ_σ-bisimulation. Define $\eta^a \subseteq \Lambda(C) \times \Lambda(C)$ as follows: let $M, N \in \Lambda(C)$ be s.t. $FV(M,N) \subseteq \{x_1, \ldots, x_n\}$,

$$M \eta^a N \iff \forall P_1 \ldots P_n \in \Lambda^0(C).\ M[P_i/x_i] \eta N[P_i/x_i].$$

(In case the strategy is by-value, i.e. for $\sigma = v, p$, $P_1, \ldots, P_n$ are chosen to be convergent terms.)

Definition 4 (Candidate Relation). Let $\eta \subseteq \Lambda^0(C) \times \Lambda^0(C)$ be a reflexive and transitive Ψ_σ-bisimulation. Define the *candidate relation* $\widehat{\eta} \subseteq \Lambda(C) \times \Lambda(C)$ by induction on M as follows:

$$\frac{x\ \eta^a\ N}{x\ \widehat{\eta}\ N} \qquad \frac{M_1\ \widehat{\eta}\ M_1' \quad M_2\ \widehat{\eta}\ M_2' \quad M_1'M_2'\ \eta^a\ N}{M_1M_2\ \widehat{\eta}\ N} \qquad \frac{M\ \widehat{\eta}\ M' \quad \lambda x.M'\ \eta^a\ N}{\lambda x.M\ \widehat{\eta}\ N}$$

Notice that the candidate relation is not simply the contextual closure of η; this subtle definition of $\widehat{\eta}$ is necessary to guarantee the crucial Substitutivity Lemma 6. The following lemma is an easy consequence of the definition of $\widehat{\eta}$.

Lemma 5. *Let $\eta \subseteq \Lambda^0(C) \times \Lambda^0(C)$ be a reflexive and transitive Ψ_σ-bisimulation. Then: i) $\widehat{\eta}$ is reflexive. ii) $\eta^a \subseteq \widehat{\eta}$. iii) $\widehat{\eta}$ is a congruence w.r.t. application. iv) $M\widehat{\eta}M' \wedge M'\eta^a N \implies M\widehat{\eta}N$.*

Lemma 6 (Substitutivity). $M\widehat{\eta}M' \wedge N\widehat{\eta}N' \implies M[N/x]\widehat{\eta}M'[N'/x]$.
(In case the strategy is by-value, i.e. for $\sigma = v, p$, N, N' must be convergent terms.)

Proof. By induction on the structure of M.

- $M \equiv x$: $\quad \dfrac{x\ \eta^a\ M'}{x\ \widehat{\eta}\ M'}$

$x\eta^a M' \implies N'\eta^a M'[N'/x]$, from the definition of η^a.
$N\widehat{\eta}N' \wedge N'\eta^a M'[N'/x] \implies N\widehat{\eta}M'[N'/x]$, from item iv of Lemma 5.

- $M \equiv M_1M_2$: $\exists M_1', M_2'$ s.t. $\quad \dfrac{M_1\ \widehat{\eta}\ M_1' \quad M_2\ \widehat{\eta}\ M_2' \quad M_1'M_2'\ \eta^a\ M'}{M_1M_2\ \widehat{\eta}\ M'}$

By induction hypothesis, $M_1[N/x]\widehat{\eta}M_1'[N'/x]$ and $M_2[N/x]\widehat{\eta}M_2'[N'/x]$. Moreover, by definition of η^a, $M_1'M_2'[N'/x]\eta^a M'[N'/x]$. Hence:

$$\frac{M_1[N/x]\ \widehat{\eta}\ M_1'[N'/x] \quad M_2[N/x]\ \widehat{\eta}\ M_2'[N'/x] \quad M_1'M_2'[N'/x]\ \eta^a\ M'[N'/x]}{M_1M_2[N/x]\ \widehat{\eta}\ M'[N'/x]}\ .$$

- $M \equiv \lambda y.M_1$: $\exists M_1'$ s.t. $\dfrac{M_1\ \widehat{\eta}\ M_1' \quad \lambda y.M_1'\ \eta^a\ M'}{\lambda y.M_1\ \widehat{\eta}\ M'}$

By induction hypothesis, $M_1[N/x]\widehat{\eta}M_1'[N'/x]$. By definition of η^a, $(\lambda y.M_1')[N'/x]\eta^a M'[N'/x]$. Hence:

$$\frac{M_1[N/x]\ \widehat{\eta}\ M_1'[N'/x] \quad (\lambda y.M_1')[N'/x]\ \eta^a\ M'[N'/x]}{(\lambda y.M_1)[N/x]\ \widehat{\eta}\ M'[N'/x]}\ . \qquad \square$$

Thus, if we take η to be the equivalence $\approx_\sigma^{app}$, we get a relation $\widehat{\approx}_\sigma^{app}$, which, by item ii of Lemma 5, extends $\approx_\sigma^{app}$. Moreover, by item iii of the same lemma, it is a congruence w.r.t. application. In order to show that $\approx_\sigma^{app}$ is itself a congruence w.r.t. application, we prove that $(\widehat{\approx}_\sigma^{app})_{|\Lambda^0(C)\times\Lambda^0(C)} = \approx_\sigma^{app}$. This is done using the coinduction principle (1), by proving that $(\widehat{\approx}_\sigma^{app})_{|\Lambda^0(C)\times\Lambda^0(C)}$ is a Ψ_σ-bisimulation. Notice that this is the only part of the proof that depends on the reduction strategy $\to_\sigma$. We succeed in showing that $(\widehat{\approx}_\sigma^{app})_{|\Lambda^0(C)\times\Lambda^0(C)}$ is a Ψ_σ-bisimulation for all the strategies of Section 2. The proof of this fact makes an essential use of the Substitutivity Lemma, and moreover, it requires the validity of some further properties, depending on the strategy $\to_\sigma$. E.g. for eager leftmost strategies we have to assume that $\approx_\sigma^{app}$ is a congruence w.r.t. λ-abstraction; for $\to_v$, we need the technical property appearing in Lemma 7 below.

Congruence Candidate Technique for Lazy Strategies For the sake of completeness, we outline briefly the proof of the fact that $(\widehat{\approx}_\sigma^{app})_{|\Lambda^0(C)\times\Lambda^0(C)}$ is a Ψ_σ-bisimulation for $\sigma = l, v$. The strategies $\to_l, \to_v$ are special cases of Howe's general format of lazy strategies, see [6, 7] for more details.

Lemma 7. *For all $M, N \in \Lambda^0$,*
$(M \approx_v^{app} N \ \wedge\ M \Downarrow_v V) \implies \exists U.\ (N \Downarrow_v U \ \wedge\ V \approx_v^{app} U)$.

Proposition 8. $(\widehat{\approx}_\sigma^{app})_{|\Lambda^0\times\Lambda^0}$ *is a Ψ_σ-bisimulation, for $\sigma \in \{l, v\}$.*

Proof. (Sketch, see [6, 7] for more details.) Let $M(\widehat{\approx}_\sigma^{app})_{|\Lambda^0\times\Lambda^0}N$. From items i and iii of Lemma 5 it follows immediately that, for all $P \in \Lambda^0$, $MP(\widehat{\approx}_\sigma^{app})_{|\Lambda^0\times\Lambda^0}NP$. The difficult part of the proof consists in proving that $M(\widehat{\approx}_\sigma^{app})_{|\Lambda^0\times\Lambda^0}N \wedge M \Downarrow_\sigma \implies N \Downarrow_\sigma$. This can be shown by induction on the derivation of $M \Downarrow_\sigma$, using Lemmata 5, 6, and, for $\sigma = v$, also Lemma 7. $\square$

Congruence Candidate Technique for Eager Leftmost Strategies

Proposition 9. *Let $\to_\sigma$ be a eager leftmost strategy s.t. $\approx_\sigma^{app}$ is a congruence w.r.t. λ-abstraction. Then $(\widehat{\approx}_\sigma^{app})_{|\Lambda^0\times\Lambda^0}$ is a Ψ_σ-bisimulation.*

Proof. The only non trivial part of the proof consists in proving that

$$M(\widehat{\approx}_\sigma^{app})_{|\Lambda^0\times\Lambda^0}N \ \wedge\ M \Downarrow_\sigma \implies N \Downarrow_\sigma .$$

Since the evaluation relation is axiomatized on the whole Λ, the above fact cannot be proved simply by induction on the derivation of $M \Downarrow_\sigma$. However, it follows from the stronger result obtained by dropping the restriction on closed λ-terms, i.e.:

$$M \widehat{\approx}^{app}_\sigma N \ \wedge \ M \Downarrow_\sigma \implies N \Downarrow_\sigma .$$

To show this, we proceed by induction on the derivation of $M \Downarrow_\sigma$.

- $M \equiv x M_1 \dots M_k$: then, by hypothesis $\exists V_{i_1}, \dots, V_{i_n}$ s.t.

$$\frac{M_{i_1} \Downarrow_\sigma V_{i_1} \dots M_{i_n} \Downarrow_\sigma V_{i_n}}{x M_1 \dots M_k \Downarrow_\sigma x V_1 \dots V_k} \quad i_1, \dots, i_n \in \{1, \dots, k\}, \ n \geq 0$$

and $\exists N_1, \dots, N_k, N^0, \dots, N^{k-1}$ s.t.

$$\dfrac{\dfrac{x (\approx^{app}_\sigma)^a N^0}{x \widehat{\approx}^{app}_\sigma N^0} \quad M_1 \widehat{\approx}^{app}_\sigma N_1 \quad N^0 N_1 (\approx^{app}_\sigma)^a N^1}{x M_1 \widehat{\approx}^{app}_\sigma N^1}$$

$$\vdots$$

$$\frac{x M_1 \dots M_{k-1} \widehat{\approx}^{app}_\sigma N^{k-1} \qquad M_k \widehat{\approx}^{app}_\sigma N_k \qquad N^{k-1} N_k (\approx^{app}_\sigma)^a N}{x M_1 \dots M_k \widehat{\approx}^{app}_\sigma N}$$

Hence

$x (\approx^{app}_\sigma)^a N^0 \implies x N_1 (\approx^{app}_\sigma)^a N^0 N_1$

$x N_1 (\approx^{app}_\sigma)^a N^0 N_1 \ \wedge \ N^0 N_1 (\approx^{app}_\sigma)^a N^1 \implies x N_1 (\approx^{app}_\sigma)^a N^1$

$$\vdots$$

$x N_1 \dots N_k (\approx^{app}_\sigma)^a N^{k-1} N_k \ \wedge \ N^{k-1} N_k (\approx^{app}_\sigma)^a N \implies x N_1 \dots N_k (\approx^{app}_\sigma)^a N$.

By induction hypothesis, from $M_{i_1} \widehat{\approx}^{app}_\sigma N_{i_1}, \dots, M_{i_n} \widehat{\approx}^{app}_\sigma N_{i_n}$, it follows that $N_{i_1} \Downarrow_\sigma, \dots, N_{i_n} \Downarrow_\sigma$. Thus $x N_1 \dots N_k \Downarrow_\sigma$. Hence, from $x N_1 \dots N_k (\approx^{app}_\sigma)^a N$, using the fact that $\approx^{app}_\sigma$ is a congruence w.r.t. λ-abstraction, it follows that $N \Downarrow_\sigma$.

- $M \equiv \lambda x. M_1$: then, by hypothesis $\exists V_1$ s.t. $\dfrac{M_1 \Downarrow_\sigma V_1}{\lambda x. M_1 \Downarrow_\sigma \lambda x. V_1}$

and $\exists N_1$ s.t. $\dfrac{M_1 \widehat{\approx}^{app}_\sigma N_1 \quad \lambda x. N_1 (\approx^{app}_\sigma)^a N}{\lambda x. M_1 \widehat{\approx}^{app}_\sigma N}$.

By induction hypothesis $N_1 \Downarrow_\sigma$. Hence $\lambda x. N_1 \Downarrow_\sigma$. Thus, from $\lambda x. N_1 (\approx^{app}_\sigma)^a N$, using the fact that $\approx^{app}_\sigma$ is a congruence w.r.t. λ-abstraction, $N \Downarrow_\sigma$.

- $M \equiv (\lambda x. M_1) M_2 \dots M_k$: then, by hypothesis $\exists V$ s.t.

$$\frac{M_1[M_2/x] M_3 \dots M_k \Downarrow_\sigma V \quad (M_2 \Downarrow_\sigma)}{(\lambda x. M_1) M_2 \dots M_k \Downarrow_\sigma V} \quad k \geq 2$$

and $\exists N_1, \dots, N_k, N^1, \dots, N^{k-1}$ s.t.

$$\dfrac{\dfrac{M_1 \widehat{\approx}^{app}_\sigma N_1 \quad \lambda x. N_1 (\approx^{app}_\sigma)^a N^1}{\lambda x. M_1 \widehat{\approx}^{app}_\sigma N^1} \quad M_2 \widehat{\approx}^{app}_\sigma N_2 \quad N^1 N_2 (\approx^{app}_\sigma)^a N^2}{(\lambda x. M_1) M_2 \widehat{\approx}^{app}_\sigma N^2}$$

$$\vdots$$

$$\frac{(\lambda x. M_1) M_2 \dots M_{k-1} \widehat{\approx}^{app}_\sigma N^{k-1} \qquad M_k \widehat{\approx}^{app}_\sigma N_k \qquad N^{k-1} N_k (\approx^{app}_\sigma)^a N}{(\lambda x. M_1) M_2 \dots M_k \widehat{\approx}^{app}_\sigma N}$$

Hence

$N^{k-2}N_{k-1}(\approx_\sigma^{app})^a N^{k-1} \;\wedge\; N^{k-1}N_k(\approx_\sigma^{app})^a N \implies$
$\qquad\qquad N^{k-2}N_{k-1}N_k(\approx_\sigma^{app})^a N$
$N^{k-3}N_{k-2}(\approx_\sigma^{app})^a N^{k-2} \;\wedge\; N^{k-2}N_{k-1}N_k(\approx_\sigma^{app})^a N \implies$
$\qquad\qquad N^{k-3}N_{k-2}N_{k-1}N_k(\approx_\sigma^{app})^a N$

$\vdots$

$N^1N_2(\approx_\sigma^{app})^a N^2 \;\wedge\; N^2N_3\dots N_k(\approx_\sigma^{app})^a N \implies N^1N_2\dots N_k(\approx_\sigma^{app})^a N$
$\lambda x.N_1(\approx_\sigma^{app})^a N^1 \;\wedge\; N^1N_2\dots N_k(\approx_\sigma^{app})^a N \implies (\lambda x.N_1)N_2\dots N_k(\approx_\sigma^{app})^a N.$
To show that $N \Downarrow_\sigma$, it is sufficient to prove that $(\lambda x.N_1)N_2\dots N_k \Downarrow_\sigma$. Then, from the definition of $(\approx_\sigma^{app})^a$, since $\approx_\sigma^{app}$ is a congruence w.r.t. λ-abstraction, we get the thesis. To show that $(\lambda x.N_1)N_2\dots N_k \Downarrow_\sigma$, it is sufficient to prove that $N_1[N_2/x]N_3\dots N_k \Downarrow_\sigma$, and possibly also that $N_2 \Downarrow_\sigma$. This latter fact follows by induction hyp.. To show $N_1[N_2/x]N_3\dots N_k \Downarrow_\sigma$, we proceeds as follows. From $M_1 \widehat{\approx}_\sigma^{app} N_1, \dots, M_k \widehat{\approx}_\sigma^{app} N_k$, using the Substitutivity Lemma, we get $M_1[M_2/x]M_3\dots M_k \widehat{\approx}_\sigma^{app} N_1[N_2/x]N_3\dots N_k$. Hence, since $M_1[M_2/x]M_3\dots M_k \Downarrow_\sigma$, by induction hypothesis, $N_1[N_2/x]N_3\dots N_k \Downarrow_\sigma$. □

Congruence Candidate Technique for Non-deterministic Strategies
Using the fact that Val_σ is closed under β-reduction, for $\to_\sigma$ non-deterministic strategy of the format of Subsection 2.1, we immediately get

Lemma 10. *Let $\to_\sigma$ be a non-deterministic strategy. Then β-reduction is correct w.r.t. $\approx_\sigma$, i.e. for $M, N \in \Lambda(C)$, $M =_\beta N \implies M \approx_\sigma N$.*

Lemma 11. *Let $\to_\sigma$ be a non-deterministic strategy. For all contexts $C[\ ]$, if $C[(\lambda x.P)Q] \widehat{\approx}_\sigma^{app} N$, then $C[P[Q/x]] \widehat{\approx}_\sigma^{app} N$.*

Proof. The proof proceeds by induction on the structure of $C[\ ]$.
• $C[\ ] \in Var$: the thesis is immediate.
• $C[\ ] \equiv [\ ]$: from the hypothesis $(\lambda x.P)Q \widehat{\approx}_\sigma^{app} N$, $\exists N_1, N_2, N_3$ s.t.

$$\dfrac{\dfrac{P \widehat{\approx}_\sigma^{app} N_1 \qquad \lambda x.N_1(\approx_\sigma^{app})^a N_2}{\lambda x.P \widehat{\approx}_\sigma^{app} N_2} \quad Q \widehat{\approx}_\sigma^{app} N_3 \qquad N_2N_3(\approx_\sigma^{app})^a N}{(\lambda x.P)Q \widehat{\approx}_\sigma^{app} N}$$

$\lambda x.N_1(\approx_\sigma^{app})^a N_2 \;\wedge\; N_2N_3(\approx_\sigma^{app})^a N \implies (\lambda x.N_1)N_3(\approx_\sigma^{app})^a N$;
using Lemma 10, we get $N_1[N_3/x](\approx_\sigma^{app})^a N$; moreover, by the Substitutivity Lemma, $P \widehat{\approx}_\sigma^{app} N_1 \;\wedge\; Q \widehat{\approx}_\sigma^{app} N_3 \implies P[Q/x] \widehat{\approx}_\sigma^{app} N_1[N_3/x]$.
Hence, from $P[Q/x] \widehat{\approx}_\sigma^{app} N_1[N_3/x]$ and $N_1[N_3/x](\approx_\sigma^{app})^a N$, using item iv of Lemma 5, it follows that $P[Q/x] \widehat{\approx}_\sigma^{app} N$.
• $C[\ ] \equiv C_1[\ ]C_2[\ ]$: from the hyp. $C_1[(\lambda x.P)Q]C_2[(\lambda x.P)Q] \widehat{\approx}_\sigma^{app} N$, $\exists N_1, N_2$ s.t.

$$\frac{C_1[(\lambda x.P)Q] \widehat{\approx}_\sigma^{app} N_1 \quad C_2[(\lambda x.P)Q] \widehat{\approx}_\sigma^{app} N_2 \quad N_1N_2(\approx_\sigma^{app})^a N}{C_1[(\lambda x.P)Q]C_2[(\lambda x.P)Q] \widehat{\approx}_\sigma^{app} N}\ .$$

By induction hypothesis, $C_1[P[Q/x]] \widehat{\approx}_\sigma^{app} N_1$ and $C_2[P[Q/x]] \widehat{\approx}_\sigma^{app} N_2$, hence $C_1[P[Q/x]]C_2[P[Q/x]] \widehat{\approx}_\sigma^{app} N_1N_2$. Then, from $N_1N_2(\approx_\sigma^{app})^a N$, using item iv of Lemma 5, we get the thesis.
• $C[\ ] \equiv \lambda y.C_1[\ ]$: from the hypothesis $\lambda y.C_1[(\lambda x.P)Q] \widehat{\approx}_\sigma^{app} N$, $\exists N_1$ s.t.

$$\frac{C_1[(\lambda x.P)Q] \widehat{\approx}_\sigma^{app} N_1 \quad \lambda y.N_1(\approx_\sigma^{app})^a N}{\lambda y.C_1[(\lambda x.P)Q] \widehat{\approx}_\sigma^{app} N}\ .$$

By induction hypothesis, $C_1[P[Q/x]] \widehat{\approx}_\sigma^{app} N_1$, hence $\lambda y.C_1[P[Q/x]] \widehat{\approx}_\sigma^{app} \lambda y.N_1$. Then, from $\lambda y.N_1 (\approx_\sigma^{app})^a N$, using item iv of Lemma 5, we get the thesis. $\square$

As we remarked earlier, the proof of the fact that $(\widehat{\approx}_\sigma^{app})_{|\Lambda^0 \times \Lambda^0}$ is a Ψ_σ-bisimulation depends essentially on the strategy. The hypotheses of the proposition below have been tuned to the strategy $\to_o$. Different sets of hypotheses are probably necessary to deal with other non-deterministic strategies.

Proposition 12. *Let $\to_\sigma$ be a non-deterministic strategy s.t.:*
1. $\approx_\sigma^{app}$ is a congruence w.r.t. λ-abstraction;
2. for all $M \in \Lambda(C)$, i) $M \Downarrow_\sigma \iff \lambda x.M \Downarrow_\sigma$ and
ii) $\lambda x.M \in Val_\sigma \implies M \in Val_\sigma$;
3. for all $M_1, M_2 \in \Lambda(C)$, i) $(M_1 \Downarrow_\sigma \ \wedge \ M_2 \Downarrow_\sigma) \implies M_1 M_2 \Downarrow_\sigma$ and
ii) $M_1 M_2 \in Val_\sigma \implies (M_1 \in Val_\sigma \ \wedge \ M_2 \in Val_\sigma)$,
then $(\widehat{\approx}_\sigma^{app})_{|\Lambda^0 \times \Lambda^0}$ is a Ψ_σ-bisimulation.

Proof. We prove, by induction on the minimal length k of a convergent path from $M \in \Lambda(C)$, that: $M \widehat{\approx}_\sigma^{app} N \ \wedge \ M \Downarrow_\sigma \implies N \Downarrow_\sigma$.
• Suppose $k = 0$. Then we proceed by induction on the structure of M:

– $M \equiv x$: $\dfrac{x (\approx_\sigma^{app})^a N}{x \widehat{\approx}_\sigma^{app} N}$; from $x (\approx_\sigma^{app})^a N$, using hypotheses 1 and 2i), we get $N \Downarrow_\sigma$.

– $M \equiv \lambda x.M_1$: $\exists N_1$ s.t. $\dfrac{M_1 \widehat{\approx}_\sigma^{app} N_1 \quad \lambda x.N_1 (\approx_\sigma^{app})^a N}{\lambda x.M_1 \widehat{\approx}_\sigma^{app} N}$; hence, by hypothesis 2ii), $M_1 \in Val_\sigma$; from $M_1 \widehat{\approx}_\sigma^{app} N_1$, using the induction hypothesis, it follows that $N_1 \Downarrow_\sigma$. Hence by hypothesis 2i) $\lambda x.N_1 \Downarrow_\sigma$, and, by hypotheses 1 and 2i), $N \Downarrow_\sigma$.

– $M \equiv M_1 M_2$: $\exists N_1, N_2$ s.t. $\dfrac{M_1 \widehat{\approx}_\sigma^{app} N_1 \quad M_2 \widehat{\approx}_\sigma^{app} N_2 \quad N_1 N_2 (\approx_\sigma^{app})^a N}{M_1 M_2 \widehat{\approx}_\sigma^{app} N}$

Since, by hypothesis 3ii), $M_1, M_2 \in Val_\sigma$, by induction hypothesis, $N_1 \Downarrow_\sigma$ and $N_2 \Downarrow_\sigma$, i.e., by hypothesis 3i), $N_1 N_2 \Downarrow_\sigma$. Hence, by hypotheses 1 and 2i), $N \Downarrow_\sigma$.

• Suppose $k > 0$. $M \equiv C[(\lambda x.P)Q] \to_\sigma C[P[Q/x]] \Downarrow_\sigma$ (the length of a minimal convergent path from $C[P[Q/x]]$ is $k-1$). From $C[(\lambda x.P)Q] \widehat{\approx}_\sigma^{app} N$, by Lemma 11, it follows that $C[P[Q/x]] \widehat{\approx}_\sigma^{app} N$. Hence, by induction hypothesis, $N \Downarrow_\sigma$. $\square$

Corollary 13. $(\widehat{\approx}_o^{app})_{|\Lambda^0 \times \Lambda^0}$ *is a Ψ_o-bisimulation.*

Proof. We extend $\to_o$ on open terms in such a way that a (possibly open) λ-term converges if and only if there exists a β-reduction path to a (possibly open) λ-term not containing any occurrence of Ω. Then Proposition 12 is applicable. $\square$

4 Final Remarks

In this paper we introduce the *purely syntactical* congruence candidate method, but there are also other methods, both syntactical and semantical, for deriving a coinductive characterization of the observational equivalence. We mention the following:
1. *Plain induction* on *computation steps* of $\to_\sigma^*$. This, purely syntactical, direct approach, which can be traced back to the work of Berry, easily applies to $\to_l$ (see [1]). With suitable extensions in order to take care of open terms, it applies also to $\sigma = h, n$.

However, it is rather problematic for call-by-value strategies such as $\to_v$, $\to_p$, or non deterministic strategies like $\to_o$. For a subtle, but complex, proof by induction on computation for $\to_v$ see [11, 10].
2. Method based on a *Separability algorithm.* This method is based on the existence of an effective procedure (see e.g. [2]) which, given two non $\approx_\sigma$-equivalent terms, M, N, allows to define an applicative context $C[\ ]$ such that either $C[M] \Downarrow_\sigma$ and $C[M] \not\Downarrow_\sigma$, or viceversa. To our knowledge, this method works only for $\approx_h$, $\approx_n$.
3. Method based on the *Domain Logic* corresponding to the *intersection types* presentation of a suitable computationally adequate CPO-λ-model. This semantical method, introduced in [9], is the generalization of the technique originally used by Abramsky and Ong in [1] for the special case of $\to_l$. In [9], this method is applied to all the strategies of Section 2.
4. *Logical Relations* method based on a *mixed induction-coinduction principle.* This semantical method is introduced in [9]. It is the generalization of the technique originally introduced by Pitts ([12]) for $\to_l$ and $\to_v$. In [9], this method is applied to all the strategies of Section 2. The method in [3] for $\to_v$ can be viewed as a weaker variant.

In general, syntactical techniques are more elementary and conceptually simpler than semantical ones, but they are often "ad hoc". However, the general version of the congruence candidate method in this paper, still maintaining the low conceptual complexity of syntactical methods, is much more uniform than plain induction on computation steps. Moreover, it seems to be at least as powerful as the semantical methods in [9].

References

1. S.Abramsky, L.Ong, *Full Abstraction in the Lazy Lambda Calculus,* Information and Computation, 105(2):159–267, 1993.
2. H.Barendregt, *The Lambda Calculus, its Syntax and Semantics,* North Holland, Amsterdam, 1984.
3. L.Egidi, F.Honsell, S.Ronchi Della Rocca, *Operational, denotational and logical Descriptions: a Case Study,* Fundamenta Informaticae, 16(2):149–169, 1992.
4. F.Honsell, M.Lenisa, *Final Semantics for untyped λ-calculus,* M.Dezani et al. eds, TLCA'95 Springer LNCS, 902:249–265, Edinburgh, 1995.
5. F.Honsell, S.Ronchi Della Rocca, *An approximation theorem for topological lambda models ...*, J. of Computer and System Sciences, 45(1):49-75, 1992.
6. D.Howe, *Equality in Lazy Computation Systems*, 4th LICS Conference Proceedings, IEEE Computer Society Press, 198–203, 1989.
7. D.Howe, *Proving Congruence of Bisimulation in Functional Programming Languages*, Information and Computation, 124(2):103–112, 1996.
8. M.Lenisa, *Final Semantics for a Higher Order Concurrent Language,* CAAP'96 Conference Proceedings, H.Kirchner ed., Springer LNCS, 1059:102–118, 1996.
9. M.Lenisa, *Semantic Techniques for Deriving Coinductive Characterizations of Observational Equivalences for λ-calculi,* to appear in TLCA'97 Proc..
10. I.Mason, S.Smith, C.Talcott, *From Operational Semantics to Domain Theory,* Information and Computation to appear.
11. P.Pérez, *An extensional partial combinatory algebra based on λ-terms*, MFCS'91 Conference Proceedings,, Springer LNCS, 520, 1996.
12. A.M.Pitts, *Relational Properties of Domains*, Inf. and Comp., 127:66–90, 1996.

This article was processed using the LaTeX macro package with LLNCS style