A Complete Coinductive Logical System for Bisimulation Equivalence on Circular Objects*

Marina Lenisa

Laboratory for the Foundations of Computer Science
University of Edinburgh, Scotland.
lenisa@dcs.ed.ac.uk

Abstract. We introduce a *coinductive* logical system *à la* Gentzen for establishing *bisimulation* equivalences on *circular non-wellfounded regular* objects, inspired by work of Coquand, and of Brandt and Henglein. In order to describe circular objects, we utilize a typed language, whose coinductive types involve disjoint sum, cartesian product, and finite powerset constructors. Our system is shown to be complete with respect to a maximal fixed point semantics. It is shown to be complete also with respect to an equivalent final semantics. In this latter semantics, terms are viewed as points of a coalgebra for a suitable endofunctor on the category Set^* of *non-wellfounded* sets. Our system subsumes an axiomatization of regular processes, alternative to the classical one given by Milner.

Introduction

In recent years, considerable energy has been devoted towards the development of simple principles and techniques for understanding, defining and reasoning on infinite and circular objects, such as streams, exact reals, processes, and other *lazy data types* ([Mil83, MPC86, Tal90, Coq94, Gim95, BM96, Fio96]). Structural induction trivially fails on infinite and non-wellfounded objects. It can be applied only in rather contrived ways, and always indirectly, often utilizing inefficient implementations of these objects, e.g. streams as inductively defined functions on natural numbers. Elaborate mathematical theories, such as domain theory ([Plo85]) and metric semantics ([BV96]), can be used, of course, to support rigorous treatment of such objects. But an ideal framework should allow to deal with infinite computational objects in a *natural*, *operationally based*, *implementation-independent* way, without requiring any heavy mathematical overhead.

Systems based on *coinductive definitions* and *coinduction proof principles* appear to be a good starting point for developing such a framework. See e.g. [Coq94, HL95, BM96, Fio96, Len96, Pit96, Rut96, Tur96, Len98] for various approaches to infinite objects based on coinduction. Coinductive techniques are natural, in that infinite and circular objects and concepts often arise in

* Work supported by Esprit Working Group "Types", MURST'97 Cofin. "Sistemi Formali..." grant, TMR Linear FMRX-CT98-0170.

connection with a *maximal fixed point* construction of some kind. Moreover, they can be justified often simply by elementary set-theoretical means, see e.g. [Acz88, Len98]. In many situations, simple categorical concepts, such as those of Final Semantics ([Acz88, RT93, Len96, Rut96, Len98]), are enough to achieve a substantial generality. In this context infinite objects are described as terms of F-coalgebras for suitable functors F's.

In this paper, inspired by the seminal work of Coquand ([Coq94, Gim94]), we make a first step towards the formulation of a *simple coinductive logical* system for reasoning on infinite circular objects, generalizing [BH97]. In particular, we present a system *à la* Gentzen $\mathcal{S}_{co}$ for capturing bisimulation equivalences on non-wellfounded *regular* (rational) objects, i.e. objects which have only a *finite* number of non-isomorphic subobjects. In order to describe the objects, we make use of an elementary typed language. Types are defined using the constructors $+$ (disjoint sum), $\times$ (cartesian product), $\mathcal{P}_f$ (finite powerset), and the higher order binding constructor ν (maximal fixed point). Objects are defined only by constructors and recursive definitions. Differently from Coquand, we do not consider functional types or term destructors. Many infinite recursive objetcs usually dealt with in lazy programming can be easily seen to have a formal counterpart in our typed language.

The crucial ingredient in the formulation of our logical system are rules whose conclusion can be used as auxiliary hypothesis in establishing the premises. In a sense, our system can be viewed as a system for infinitely regressive proofs. As remarked earlier, it is inspired by the technique for dealing with coinductive types in Intuitionistic Type Theories, introduced in [Coq94], where *infinitely proofs* are handled by means of the *guarded induction principle*. This technique, originally developed for predicative systems, was later extended by Giménez to impredicative systems, [Gim94, Gim95]. Our system can be seen as a partial attempt to an elementary reconstruction of that approach, in such a way that it can be reconciled with other, more classical, syntactical approaches to circular objects ([Mil84, Acz88, BH97]). Our work seems to be related in particular with [Gim94], where Coquand's principle of guarded induction is shown to be complete with respect to the traditional principle of coinduction, in a type theoretic setting.

This paper generalizes [BH97], where a coinductive axiomatization of the type (in)equality for a simple first order language of regular recursive types is provided. The types considered in [BH97] are terms for denoting regular binary trees.

In order to give external independent justifications to our system, we consider two different, but equivalent, semantics. The first is a fixed point semantics, the latter is based on the Final Semantics paradigm ([Acz88, RT93, Tur96, Len98]).

The fixed point semantics is defined by introducing, for each type σ, a corresponding bisimulation equivalence $\approx_\sigma$ on the set T^0_σ of closed terms typable with σ. This family of equivalences is defined as the greatest fixed point of a monotone operator Φ, and it can be viewed as the "intended semantics". One of the main technical results in this paper is the fact that the system $\mathcal{S}_{co}$ axiomatizes completely the bisimulation equivalences $\approx_\sigma$, for all type σ. The correctness of $\mathcal{S}_{co}$ is proved by coinduction, i.e. by showing that the family of relations axiomatized by $\mathcal{S}_{co}$ on closed terms typable with σ is a Φ-bisimulation. The completeness proof exploits the fact that the terms that we consider are regular.

In order to give the categorical semantics, we define a "universal" functor F, involving constructors corresponding to each of the type constructors. Then we show how to endow the family of closed typable terms $\{T^0_\sigma\}_{\sigma \in Type}$ with a structure of F-coalgebra, in such a way that the greatest F-bisimulation on the coalgebra of terms coincides with the family of bisimulation equivalences $\{\approx_\sigma\}_{\sigma \in Type}$. This yields a final semantics for our typed language. Another technical result of this paper is the fact that the categorical semantics coincides with the fixed point semantics. For simplicitly, we work in the category Set^* of *non-wellfounded sets* and set-theoretic functions. In this context final coalgebras of many functors are maximal fixpoints. Non-wellfounded sets are elements of a Universe *à la* Zermelo-Fraenkel in which the Foundation Axiom is replaced by the Antifoundation Axiom X_1 of Forti and Honsell [FH83] (or by the Antifoundation Axiom AFA of [Acz88]).

Our system, when restricted to the type of CCS-like processes, can be viewed as a logical system for establishing strong equivalence of processes, alternative to the classical axiomatic system of Milner, [Mil84].

The paper is organized as follows. In Section 1, we introduce the syntax for types and terms, and the system for establishing correct typing judgements. We introduce also the fixed point semantics as a family of bisimulation equivalences $\{\approx_\sigma\}_\sigma$. In Section 2, we introduce the coinductive formal system $\mathcal{S}_{co}$ *à la* Gentzen, and we show that, for all closed type σ, this system axiomatizes the bisimulation equivalence $\approx_\sigma$ on T^0_σ. In Section 3, we define a "universal" functor F on the category Set^*, and we endow the set of closed typable terms with a coalgebra structure for the functor F. Moreover, we show that the system $\mathcal{S}_{co}$ axiomatizes the largest F-bisimulation on the coalgebra of closed typable terms. Final remarks and directions for future work appear in Section 4.

The author is grateful to Peter Aczel, Furio Honsell, and the anonymous referees for useful comments.

1 Types and Terms

In this section we introduce a finite language for infinite objects.

Definition 1.1 (Types). Let $TVar$ be a set of type variables. The set of types $Type$ is defined by

$$\sigma \ ::= \ X \ \mid \ K_1 \ \mid \ldots \mid \ K_n \ \mid \ \sigma + \sigma \ \mid \ \sigma \times \sigma \ \mid \ \mathcal{P}_f(\sigma) \ \mid \ \nu X.\sigma \ ,$$

where $X \in TVar$, the symbols $K_1, \ldots, K_n$ denote constant types, $+$, $\times$, $\mathcal{P}_f(\,)$ are disjoint sum, cartesian product, and finite powerset type constructors. The coinductive type $\nu X.\sigma$ is considered always to be guarded, i.e. all the free occurrences of the variable X in σ are within the scope of a type constructor.

In the type $\nu X.\sigma$, the occurrences of the variable X in σ are *bound*. An occurrence of the variable X in σ is *free* if it is not bound.

Remark 1.2. For simplicitly, in the definition of types we have considered only binary product and binary disjoint sum, but we could have considered, more in general, n-ary products and n-ary disjoint sums, for $n \geq 0$.

Definition 1.3 (Terms). Let Var be a set of variables. The set of terms $Term$ is defined by

$$t \ ::= \ x \ \mid \ c^i_j \ \mid \ i_1(t) \ \mid \ i_2(t) \ \mid < t, t > \mid \ [t, \ldots, t] \ \mid \ rec\, x.t \ \mid \ in(t) \ ,$$

where $x \in Var$, $\{C_j \equiv \{c^i_j \mid i \in I_j\}\}_{j \leq n}$ are sets of constants, $[\ldots]$ denotes the multiset term constructor, $i_1(\ )$, $i_2(\ )$ are the left and right injections in the disjoint sum, $<\ ,\ >$ is the pairing constructor, $in(\ )$ is the unfolding constructor, and the term $rec\, x.t$ is required to be guarded, i.e. all the free occurrences of the variable x in t are within the scope of one of the following term constructors: $i_1(\ )$, $i_2(\ )$, $<\ ,\ >$, $[\ldots]$.
Let $Term^0$ denote the set of closed terms.

We take terms to be equal up to permutation of elements in multisets. The constructor $in(\ )$ is introduced in order to obtain a typing system in which the shape of the type determines the form of the terms typable with that type (see Definition 1.4 and Lemma 1.5 below).

In the syntax defined above, the non-deterministic process constructor $+$ of CCS-like concurrent languages ([Mil83]) is subsumed by the $[\ldots]$ constructor.

The terms which we are interested in are those typable as follows:

Definition 1.4. Let $\mathcal{S}_{type}$ be the following formal typing system for deriving judgements of the shape $\Delta \vdash t : \sigma$, where the environment Δ is a partial function from Var to $Type$.

$$\frac{}{\Delta, x : \sigma \vdash_{type} x : \sigma} \ (var)$$

$$\frac{}{\Delta \vdash_{type} c^i_j : K_j} \ (const)$$

$$\frac{\Delta \vdash_{type} t : \sigma_1}{\Delta \vdash_{type} i_1(t) : \sigma_1 + \sigma_2} \ (+_1)$$

$$\frac{\Delta \vdash_{type} t : \sigma_2}{\Delta \vdash_{type} i_2(t) : \sigma_1 + \sigma_2} \ (+_2)$$

$$\frac{\Delta \vdash_{type} t_1 : \sigma_1 \quad \Delta \vdash_{type} t_2 : \sigma_2}{\Delta \vdash_{type} < t_1, t_2 > : \sigma_1 \times \sigma_2} \ (\times)$$

$$\frac{\{\Delta \vdash_{type} t_i : \sigma\}_{i=1,\ldots,n}}{\Delta \vdash_{type} [t_1, \ldots, t_n] : \mathcal{P}_f(\sigma)} \ ([\,])$$

$$\frac{\Delta, x : \nu X.\sigma \vdash_{type} t : \nu X.\sigma \quad rec\, x.t \text{ guarded}}{\Delta \vdash_{type} rec\, x.t : \nu X.\sigma} \ (\nu)$$

$$\frac{\Delta \vdash_{type} t : \sigma[\nu X.\sigma/X]}{\Delta \vdash_{type} in(t) : \nu X.\sigma} \ (fold)$$

Lemma 1.5. *Let $t \in Term \setminus Var$ be such that $\Delta \vdash_{type} t : \sigma$. Then*

$$\begin{array}{ll}
\sigma \equiv K_j & \Longleftrightarrow t \in C_j \\
\sigma \equiv \sigma_1 + \sigma_2 & \Longleftrightarrow \exists j \in \{1,2\}.\ \exists t'.\ (t \equiv i_j(t')\ \&\ \Delta \vdash_{type} t' : \sigma_j) \\
\sigma \equiv \sigma_1 \times \sigma_2 & \Longleftrightarrow \exists t_1, t_2.\ (t \equiv < t_1, t_2 >\ \&\ \forall j = 1,2.\ \Delta \vdash_{type} t_j : \sigma_j) \\
\sigma \equiv \mathcal{P}_f(\sigma_1) & \Longleftrightarrow \exists n \geq 0.\ \exists t_1, \ldots, t_n.\ (t \equiv [t_1, \ldots, t_n]\ \& \\
& \quad \forall i = 1, \ldots, n.\ \Delta \vdash_{type} t_i : \sigma_1) \\
\sigma \equiv \nu X.\sigma_1 & \Longleftrightarrow \exists n \geq 0.\ \exists t'.\ (t \equiv rec\, x_1 \ldots rec\, x_n.\ in(t')\ \& \\
& \quad \Delta, x_1 : \nu X.\sigma_1, \ldots, x_n : \nu X.\sigma_1 \vdash_{type} t' : \sigma_1[\nu X.\sigma_1/X]).
\end{array}$$

The following Substitution Lemma can be easily proved by induction on derivations.

Lemma 1.6 (Substitution).

$$\Delta, x : \tau \vdash_{type} t : \sigma\ \&\ \ \Delta \vdash_{type} t' : \tau \implies \Delta \vdash_{type} t[t'/x] : \sigma\ .$$

The following notation will be useful in the sequel:

Notation Let $\sigma \in Type$.

- Let T_σ denote the set $\{t \in Term \mid \ \exists \Delta.\ \Delta \vdash_{type} t : \sigma\}$.
- Let T^0_σ denote the set $\{t \in Term^0 \mid \vdash_{type} t : \sigma\}$.

1.1 Bisimulation Equivalence on Closed Typable Terms

In this subsection we give the intended fixed point semantics of our typed language. This takes the form of a family of *bisimulation equivalences* $\{\approx_\sigma\}_{\sigma \in Type}$, where $\approx_\sigma$ is a relation on the set of closed terms T^0_σ. The family $\{\approx_\sigma\}_\sigma$ is characterized as the greatest fixed point of the following monotone operator, whose definition clearly reflects the intended meaning of the constructors:

Definition 1.7. Let $\Phi : \Pi_{\sigma \in Type} \mathcal{P}(T^0_\sigma \times T^0_\sigma) \to \Pi_{\sigma \in Type} \mathcal{P}(T^0_\sigma \times T^0_\sigma)$ be the operator[1] defined as follows

$$\Phi(\{\mathcal{R}_\sigma\}_{\sigma \in Type}) = \{\mathcal{R}^\Phi_\sigma\}_{\sigma \in Type}\ ,$$

where the the relation $\mathcal{R}^\Phi_\sigma \subseteq T^0_\sigma \times T^0_\sigma$ is defined by

$$\begin{array}{ll}
t\ \mathcal{R}^\Phi_{K_j}\ t' & \Longleftrightarrow t \equiv t' \\
t\ \mathcal{R}^\Phi_{\sigma_1+\sigma_2}\ t' & \Longleftrightarrow \exists j \in \{1,2\}. \exists t_1, t'_1.\ (t \equiv i_j(t_1)\ \&\ t' \equiv i_j(t'_1)\ \&\ t_1\ \mathcal{R}_{\sigma_j}\ t'_1) \\
t\ \mathcal{R}^\Phi_{\sigma_1 \times \sigma_2}\ t' & \Longleftrightarrow \exists t_1, t_2, t'_1, t'_2.\ (t \equiv < t_1, t_2 >\ \&\ t' \equiv < t'_1, t'_2 >\ \& \\
& \quad \forall j = 1,2.\ t_j\ \mathcal{R}_{\sigma_j}\ t'_j) \\
t\ \mathcal{R}^\Phi_{\mathcal{P}_f(\sigma_1)}\ t' & \Longleftrightarrow \exists m, n \geq 0. \exists t_1, \ldots, t_m, t'_1, \ldots, t'_n.\ (t \equiv [t_1, \ldots, t_m]\ \& \\
& \quad t' \equiv [t'_1, \ldots, t'_n]\ \& \\
& \quad \forall t_i \in [t_1, \ldots, t_m]\ \exists t'_j \in [t'_1, \ldots, t'_n].\ t_i\ \mathcal{R}_{\sigma_1}\ t'_j\ \& \\
& \quad \forall t'_j \in [t'_1, \ldots, t'_n]\ \exists t_i \in [t_1, \ldots, t_m].\ t_i\ \mathcal{R}_{\sigma_1}\ t'_j) \\
t\ \mathcal{R}^\Phi_{\nu X_1.\sigma_1}\ t' & \Longleftrightarrow \exists m, n \geq 0. \exists t_1, t'_1.\ (t \equiv rec\, x_1 \ldots rec\, x_m.in(t_1)\ \& \\
& \quad t' \equiv rec\, x_1 \ldots rec\, x_n.in(t'_1)\ \& \\
& \quad t_1[t/x_1, \ldots, t/x_m]\ \mathcal{R}_{\sigma_1[\nu X_1.\sigma_1/X_1]}\ t'_1[t'/x_1, \ldots, t'/x_n]).
\end{array}$$

[1] $\Pi_{i \in I} A_i$ denotes the infinite cartesian product of the A_i's, for $i \in I$.

The definition above can be viewed as the set-theoretical counterpart of the definition of relational structures on c.p.o.'s given by Pitts (see [Pit96]). Among the various differences between our approach and his, we point out that we allow for *nested recursion* directly at the outset in Definition 1.7, while Pitts deals with it separately.

Proposition 1.8. *The operator $\Phi : \Pi_{\sigma \in Type} \mathcal{P}(T^0_\sigma \times T^0_\sigma) \to \Pi_{\sigma \in Type} \mathcal{P}(T^0_\sigma \times T^0_\sigma)$ is monotone over the complete lattice $(\Pi_{\sigma \in Type} \mathcal{P}(T^0_\sigma \times T^0_\sigma), \Pi_{\sigma \in Type} \subseteq_\sigma)$, where $\forall \sigma.\ \subseteq_\sigma \equiv \subseteq$.*

Let us denote by $\{\approx_\sigma\}_{\sigma \in Type}$ the greatest fixed point of the operator Φ. This will be the family of *bisimulation equivalences* giving the intended semantics of our system.

The validity of the following coinduction principle follows immediately:

$$\frac{\forall \sigma \in Type.\ \mathcal{R}_\sigma \subseteq \mathcal{R}^\Phi_\sigma}{\forall \sigma \in Type.\ \mathcal{R}_\sigma \subseteq \approx_\sigma}$$

We call *Φ-bisimulation* a family $\{\mathcal{R}_\sigma\}_{\sigma \in Type}$ such that $\forall \sigma \in Type.\ \mathcal{R}_\sigma \subseteq \mathcal{R}^\Phi_\sigma$.

Notice that, using our language of types and the notion of bisimulation equivalences introduced above, we can recover the case of binary trees, and the case of non-deterministic processes with strong bisimulation equivalence. In fact, binary trees can be described as the set of terms $T^0_{\nu X.(X \times X) + \sigma_C}$, for σ_C constant type, while non-deterministic processes over a set of labels C of type σ_C can be described as the set of terms $T^0_{\nu X.\mathcal{P}_f(\sigma_C \times X)}$.

2 A Coinductive Logical System for Bisimulation Equivalence

In this section, we introduce the formal system $\mathcal{S}_{co}$, *à la* Gentzen, for proving $\sim$-equivalence between pairs of terms. We will show that $\mathcal{S}_{co}$ axiomatizes exactly, for all type σ, the bisimulation equivalence $\approx_\sigma$.

Definition 2.1. Let $\mathcal{S}_{co}$ be the following formal system for deriving judgements of the shape $< \Delta; \Gamma > \vdash_{co} t \sim t' : \sigma$, where $< \Delta; \Gamma >$ is the environment and

- Δ is a partial function from Var to $Type$;
- Γ is a multiset of the shape $[t_1 \sim t'_1 : \sigma_1, \ldots, t_n \sim t'_n : \sigma_n]$;
- Γ is *coherent* with Δ, i.e.
  $t_i \sim t'_i : \sigma_i \in \Gamma \ \Rightarrow\ (\Delta \vdash_{type} t_i : \sigma_i\ \&\ \Delta \vdash_{type} t'_i : \sigma_i)$;
- $\Delta \vdash_{type} t : \sigma\ \&\ \Delta \vdash_{type} t' : \sigma$.

The rules of $\mathcal{S}_{co}$ are the following:

$$\frac{\Delta \vdash_{type} t : \sigma \quad \Gamma \text{ coherent with } \Delta}{< \Delta; \Gamma > \vdash_{co} t \sim t : \sigma} \ (\mathit{refl})$$

$$\frac{< \Delta; \Gamma > \vdash_{co} t_1 \sim t_2 : \sigma}{< \Delta; \Gamma > \vdash_{co} t_2 \sim t_1 : \sigma} \ (\mathit{symm})$$

$$\frac{< \Delta; \Gamma >\vdash_{co} t_1 \sim t_2 : \sigma \quad < \Delta; \Gamma >\vdash_{co} t_2 \sim t_3 : \sigma}{< \Delta; \Gamma >\vdash_{co} t_1 \sim t_3 : \sigma} \; \textit{(trans)}$$

$$\frac{\Delta \vdash_{type} t : \sigma \quad \Delta \vdash_{type} t' : \sigma}{< \Delta; \Gamma, t \sim t' : \sigma >\vdash_{co} t \sim t' : \sigma} \; \textit{(hyp)}$$

$$\frac{\Delta \vdash_{type} rec\, x.t : \nu X.\sigma \quad \Gamma \text{ coherent with } \Delta}{< \Delta; \Gamma >\vdash_{co} rec\, x.t \sim t[rec\, x.t/x] : \nu X.\sigma} \; \textit{(rec)}$$

$$\frac{\{< \Delta; \Gamma >\vdash_{co} t_i \sim t_i' : \sigma_i\}_{i=1,2}}{< \Delta; \Gamma >\vdash_{co} < t_1, t_2 > \sim < t_1', t_2' >: \sigma_1 \times \sigma_2} \; (\times cong)$$

$$\frac{< \Delta; \Gamma >\vdash_{co} t \sim t' : \sigma_1}{< \Delta; \Gamma >\vdash_{co} i_1(t) \sim i_1(t') : \sigma_1 + \sigma_2} \; (+_1 cong)$$

$$\frac{< \Delta; \Gamma >\vdash_{co} t \sim t' : \sigma_2}{< \Delta; \Gamma >\vdash_{co} i_2(t) \sim i_2(t') : \sigma_1 + \sigma_2} \; (+_2 cong)$$

$$\frac{\{< \Delta; \Gamma >\vdash_{co} t_i \sim t_i' : \sigma\}_{i=1,\ldots,n}}{< \Delta; \Gamma >\vdash_{co} [t_1, \ldots, t_n] \sim [t_1', \ldots, t_n'] : \mathcal{P}_f(\sigma)} \; ([\,] cong)$$

$$\frac{\Delta \vdash_{type} [t_1, \ldots, t_n, t, t'] : \mathcal{P}_f(\sigma) \quad < \Delta; \Gamma >\vdash_{co} t \sim t' : \sigma}{< \Delta; \Gamma >\vdash_{co} [t_1, \ldots, t_n, t, t'] \sim [t_1, \ldots, t_n, t] : \mathcal{P}_f(\sigma)} \; \textit{(abs)}$$

$$\frac{< \Delta; \Gamma, in(t) \sim in(t') : \nu X.\sigma >\vdash_{co} t \sim t' : \sigma[\nu X.\sigma/X]}{< \Delta; \Gamma >\vdash_{co} in(t) \sim in(t') : \nu X.\sigma} \; \textit{(in)}$$

The names given to the rules above are suggestive. In particular, the rules *(cong)* are the *congruence rules*, while rule *(abs)* is the *absorption rule*, which embodies contraction for equal terms appearing in multisets.

One can easily check, by induction on derivations, using Lemma 1.6, that the definition above is well posed, i.e.

$$< \Delta; \Gamma >\vdash_{co} t \sim t' : \sigma \implies (\Gamma \text{ coherent with } \Delta \,\&\, \Delta \vdash_{type} t : \sigma \,\&\, \Delta \vdash_{type} t' : \sigma)\,.$$

Notice the "coinductive" nature of the rule (in): in order to establish the equivalence $\sim$ between terms of the shape $in(t)$ and $in(t')$, we can assume, in the premise of the rule (in), the judgement that we want to prove, i.e. $in(t) \sim in(t')$.

Remark 2.2. i) In place of rule (in) in the system $\mathcal{S}_{co}$ above, one could use equivalently the following two rules

$$\frac{< \Delta; \Gamma >\vdash_{co} t \sim t' : \sigma}{< \Delta; \Gamma >\vdash_{co} in(t) \sim in(t') : \nu X.\sigma}$$

$$\frac{< \Delta; \Gamma, rec\, x.t \sim rec\, y.t' : \nu X.\sigma >\vdash_{co} t \sim t' : \sigma}{< \Delta; \Gamma >\vdash_{co} rec\, x.t \sim rec\, y.t' : \nu X.\sigma}$$

This latter presentation would emphasize Coquand's correspondence between guarded infinite objects and guarded infinite proofs, but the presentation of the system of Definition 2.1 slightly simplifies the proof of Theorem 2.10 below.

ii) When specialized to the type $\nu X.\mathcal{P}_f(\sigma_C \times X)$ of CCS non-deterministic processes, our logical system provides an alternative axiomatization of Milner's strong bisimulation ([Mil84]). The crucial difference between our system and the classical system of Milner is the absence, in $\mathcal{S}_{co}$, of a counterpart to Milner's rule for recursion, *viz*: $\dfrac{t \sim t'[t/x]}{t \sim rec\, x.t'}$ *(uniqueness)* . This rule is recovered in $\mathcal{S}_{co}$ by the coinductive rule (in), which amounts to the coinductive version of the congruence rule for the *rec* operator in Milner's system. Milner's system is a Hilbert system. Hence top-down proof search can be rather unpractical. For instance, when confronted with two terms one of which is a *rec* term, one has to guess whether to unfold the term or to use rule *(uniqueness)*. On the contrary, in $\mathcal{S}_{co}$, one can capitalize on hypotheses, and hence the structure of terms determine essentially, i.e. up-to absorption and unfolding, the top-down search of a proof (see Example 2.4 below). This informal argument will be put to use in order to show the completeness of $\mathcal{S}_{co}$ (Theorem 2.13) and its decidability.

It is immediate to see, by induction on derivations, that the following Weakening Lemma holds:

Lemma 2.3 (Weakening). *If* $< \Delta; \Gamma >\vdash_{co} t \sim t' : \sigma$ *is derivable in* $\mathcal{S}_{co}$ *and* Γ' *is coherent with* Δ*, then also* $< \Delta; \Gamma, \Gamma' >\vdash_{co} t \sim t' : \sigma$ *is derivable in* $\mathcal{S}_{co}$.

We illustrate now the system $\mathcal{S}_{co}$ at work.

Example 2.4. Let $t_1 \equiv rec\, x.in(< c, x >)$ and $t_2 \equiv rec\, y.in(< c, in(< c, y >) >)$, where $c \in C_j$. Then one can easily check that $\vdash_{type} t_i : \nu X.K_j \times X$, for $i = 1, 2$. Moreover, using the system $\mathcal{S}_{co}$, one can show that the two terms t_1 and t_2 are bisimilar. In fact, up to applications of the rules *(rec), (symm), (trans)*, we can build a derivation of $\vdash_{co} t_1 \sim t_2 : \nu X.K_j \times X$ as follows:

$$\dfrac{\dfrac{< \Delta; \Gamma', \Gamma >\vdash_{co} c \sim c : K_j \quad < \Delta; \Gamma', \Gamma >\vdash_{co} t_1 \sim t_2 : \nu X.K_j \times X}{< \Delta; \Gamma', \Gamma >\vdash_{co} < c, t_1 > \sim < c, t_2 >: K_j \times \nu X.K_j \times X}\,(\times cong)}{< \Delta; \Gamma >\vdash_{co} in(< c, t_1 >) \sim in(< c, t_2 >) : \nu X.K_j \times X}\,(in)$$

and

$$\dfrac{\dfrac{< \Delta; \Gamma >\vdash_{co} c \sim c : K_j \quad < \Delta; \Gamma >\vdash_{co} t_1 \sim in(< c, t_2 >) : \nu X.K_j \times X}{< \Delta; \Gamma >\vdash_{co} < c, t_1 > \sim < c, in(< c, t_2 >) >: K_j \times \nu X.K_j \times X}\,(\times cong)}{\vdash_{co} in(< c, t_1 >) \sim in(< c, in(< c, t_2 >)) : \nu X.K_j \times X}\,(in)$$

where
$\Delta \equiv \emptyset$
$\Gamma \equiv [in(< c, t_1 >) \sim in(< c, in(< c, t_2 >) >) : \nu X.K_j \times X]$
$\Gamma' \equiv [in(< c, t_1 >) \sim in(< c, t_2 >) : \nu X.K_j \times X]$.

Example 2.5 (Conway Identity). A term with $n > 0$ rec's at the top is equivalent to a term with just one rec, i.e., any term $\overline{t} \equiv rec\, x_1 \ldots rec\, x_n.in(t)$, $n > 0$, typable with $\nu X.\sigma$, for some σ, is such that

$$\exists < \Delta; \Gamma > . \ < \Delta; \Gamma >\vdash_{co} \overline{t} \sim \overline{t'} : \nu X.\sigma \ ,$$

where $\overline{t'} \equiv rec\, x.in(t[x/x_1, \ldots, x/x_n])$, and the variables $x_1, \ldots, x_n$ are replaced by the variable x, which is new in $\overline{t}$.
By rules *(rec)*, *(symm)*, *(trans)*, *(in)* (using also the Weakening Lemma), it is sufficient to show that the two terms $t[\overline{t}/x_1, \ldots, \overline{t}/x_n]$ and $t[\overline{t'}/x_1, \ldots, \overline{t'}/x_n]$ are $\sim$-equivalent. More in general, we show, by structural induction on t, that, for all $n > 0$, for all $\overline{t}_1, \overline{t'}_1, \ldots, \overline{t}_n, \overline{t'}_n$ such that $\exists \Delta \exists \tau.\ \Delta \vdash_{type} t[\overline{t}_1/x_1, \ldots, \overline{t}_n/x_n] : \tau$ and $\Delta \vdash_{type} t[\overline{t'}_1/x_1, \ldots, \overline{t'}_n/x_n] : \tau$,

$$\exists \Gamma.\ < \Delta; \Gamma > \vdash_{co} t[\overline{t}_1/x_1, \ldots, \overline{t}_n/x_n] \sim t[\overline{t'}_1/x_1, \ldots, \overline{t'}_n/x_n] : \tau\ .$$

The only non trivial case is that of $t \equiv rec\, y_1 \ldots rec\, y_m.in(\widetilde{t})$, for $m \geq 0$. But, again by rules *(rec)*, *(symm)*, *(trans)*, *(in)*, it is sufficient to prove that
$< \Delta; \Gamma > \vdash_{co} \widetilde{t}^{\,\overline{t}_1 \ldots \overline{t}_n t \ldots t}_{x_1 \ldots x_n y_1 \ldots y_m} \sim \widetilde{t}^{\,\overline{t'}_1 \ldots \overline{t'}_n t \ldots t}_{x_1 \ldots x_n y_1 \ldots y_m} : \tau'$, for a suitable τ', where
$\widetilde{t}^{\,\overline{t}_1 \ldots \overline{t}_n t \ldots t}_{x_1 \ldots x_n y_1 \ldots y_m} \equiv \widetilde{t}[\overline{t}_1/x_1, \ldots, \overline{t}_n/x_n, t/y_1, \ldots, t/y_m]$ and
$\widetilde{t}^{\,\overline{t'}_1 \ldots \overline{t'}_n t \ldots t}_{x_1 \ldots x_n y_1 \ldots y_m} \equiv \widetilde{t}[\overline{t'}_1/x_1, \ldots, \overline{t'}_n/x_n, t/y_1, \ldots, t/y_m]$.
But this follows by induction hypothesis.

The rest of this section is devoted to the proof of the fact that the system $\mathcal{S}_{co}$ axiomatizes exactly, for all type σ, the bisimulation equivalence $\approx_\sigma$. More precisely, we will prove that, for all $\sigma \in Type$ and for all $t, t' \in T^0_\sigma$,

$$\vdash_{co} t \sim t' : \sigma \iff t \approx_\sigma t'\ .$$

We will refer to the implication ($\Rightarrow$) as the correctness of the system $\mathcal{S}_{co}$ w.r.t. $\approx_\sigma$, and to the implication ($\Leftarrow$) as the completeness of the system $\mathcal{S}_{co}$ w.r.t. $\approx_\sigma$.

2.1 Correctness of $\mathcal{S}_{co}$

First we need a technical definition.

Definition 2.6. A sequent $< \Delta; t_1 \sim t'_1 : \sigma_1, \ldots, t_n \sim t'_n : \sigma_n > \vdash_{co} t_{n+1} \sim t'_{n+1} : \sigma_{n+1}$ is *completely derivable* in $\mathcal{S}_{co}$ if there exist derivations in $\mathcal{S}_{co}$ of $< \Delta; t_1 \sim t'_1 : \sigma_1, \ldots, t_{i-1} \sim t'_{i-1} : \sigma_{i-1} > \vdash_{co} t_i \sim t'_i : \sigma_i$, for all $i = 1, \ldots, n+1$.

In order to show the correctness of $\mathcal{S}_{co}$, we will prove that the following family of relations is a Φ-bisimulation:

Definition 2.7. Let $\sigma \in Type$. We define

$$\mathcal{R}^{cd}_\sigma = \{(t, t') \in T^0_\sigma \times T^0_\sigma \mid \exists < \Delta; \Gamma > .\ < \Delta; \Gamma > \vdash_{co} t \sim t' : \sigma \text{ completely derivable}\}\ .$$

The following two lemmata are instrumental.

Lemma 2.8. *Let $< \Delta; \Gamma > \vdash_{co} t \sim t' : \sigma$ be a completely derivable sequent. Then*

1. *If* $\sigma \equiv \sigma_1 + \sigma_2$*,* $t \equiv i_j(\overline{t})$ *and* $t' \equiv i_j(\overline{t}')$*, for some* $j \in \{1,2\}$*, then also* $< \Delta; \Gamma > \vdash_{co} \overline{t} \sim \overline{t}' : \sigma_j$ *is a completely derivable sequent.*
2. *If* $\sigma \equiv \sigma_1 \times \sigma_2$ *and* $t \equiv < t_1, t_2 >$*, then also* $< \Delta; \Gamma > \vdash_{co} t_i \sim t'_i : \sigma_i$*, for all* $i = 1, 2$*, is a completely derivable sequent.*
3. *If* $\sigma \equiv \mathcal{P}_f(\sigma_1)$*,* $t \equiv [t_1, \ldots, t_m]$*,* $t' \equiv [t'_1, \ldots, t'_n]$*, then* $\forall i \in \{1, \ldots, m\}. \exists j \in \{1, \ldots, n\}$ *such that* $< \Delta; \Gamma > \vdash_{co} t_i \sim t'_j : \sigma_1$ *is a completely derivable sequent, and* $\forall j \in \{1, \ldots, n\}. \exists i \in \{1, \ldots, m\}$ *such that* $< \Delta; \Gamma > \vdash_{co} t_i \sim t'_j : \sigma_1$ *is a completely derivable sequent.*

Proof. The proof is by induction on the sum of the lengths of the derivations τ and τ_i's, where τ denotes the derivation of $< \Delta; t_1 \sim t'_1 : \sigma_1, \ldots, t_n \sim t'_n : \sigma_n > \vdash_{co} t \sim t' : \sigma$ and τ_i denotes the derivation of $< \Delta; t_1 \sim t'_1 : \sigma_1, \ldots, t_{i-1} \sim t'_{i-1} : \sigma_{i-1} > \vdash_{co} t_i \sim t'_i : \sigma_i$, for $i = 1, \ldots, n$. We work out in detail only the proof of item 3, the proofs of the other two items are similar.
Base Case: The only rule applied in τ is *(refl)*. The thesis follows using Lemma 1.5 and rule *(refl)*.
Induction Step: we proceed by analyzing the last rule applied in τ. If the last rule is *(refl)*, then again the thesis follows using Lemma 1.5 and rule *(refl)*. If the last rule is *(symm)* or *(hyp)*, the thesis follows immediately by induction hypothesis. If the last rule is *(trans)*, then the thesis follows by induction hypothesis, using Lemma 1.5. If the last rule is *(abs)*, the thesis follows using Lemma 1.5 and rule *(refl)*. Finally, if the last rule in τ is ([ ]*cong*), then the thesis is immediate. □

Lemma 2.9. *Let* $\sigma \in Type$*. Then*
i) For all $t \in T^0_\sigma$*,* $t(\mathcal{R}^{cd}_\sigma)^\Phi t$*.*
ii) For all $t_1, t_2 \in T^0_\sigma$*,* $t_1(\mathcal{R}^{cd}_\sigma)^\Phi t_2 \implies t_2(\mathcal{R}^{cd}_\sigma)^\Phi t_1$*.*
iii) For all $t_1, t_2, t_3 \in T^0_\sigma$ *such that, for some* Γ, Δ*, the sequents* $< \Delta; \Gamma > \vdash_{co} t_1 \sim t_2 : \sigma$ *and* $< \Delta; \Gamma' > \vdash_{co} t_2 \sim t_3 : \sigma$ *are completely derivable,*

$$[t_1(\mathcal{R}^{cd}_\sigma)^\Phi t_2 \ \&\ t_2(\mathcal{R}^{cd}_\sigma)^\Phi t_3] \implies t_1(\mathcal{R}^{cd}_\sigma)^\Phi t_3 \ .$$

Proof. Both items i) and ii) can be easily shown by case analysis on σ, using Lemma 1.5. Item iii) is shown by by case analysis on σ, using Lemmata 2.3 and 2.8. □

Theorem 2.10 (Correctness). *Let* $\sigma \in Type$*. For all* $t, t' \in T^0_\sigma$*,*

$$\vdash_{co} t \sim t' : \sigma \implies t \approx_\sigma t' \ .$$

Proof. We show that the family $\{\mathcal{R}^{cd}_\sigma\}_{\sigma \in Type}$ is a Φ-bisimulation, i.e. we have to show that $\forall \sigma.\ \mathcal{R}^{cd}_\sigma \subseteq (\mathcal{R}^{cd}_\sigma)^\Phi$. We prove this by induction on the sum of the lengths of the derivations τ and τ_i's, where τ denotes the derivation of $< \Delta; t_1 \sim t'_1 : \sigma_1, \ldots, t_n \sim t'_n : \sigma_n > \vdash_{co} t \sim t' : \nu X.\sigma$ and τ_i denotes the derivation of $< \Delta; t_1 \sim t'_1 : \sigma_1, \ldots, t_{i-1} \sim t'_{i-1} : \sigma_{i-1} > \vdash_{co} t_i \sim t'_i : \sigma_i$, for $i = 1, \ldots, n$.
Base Case: The only rule applied in τ is *(refl)* or *(rec)*. The thesis follows from item i) of Lemma 2.9.

Induction Step: We proceed by analyzing the last rule applied in τ. If the last rule is *(refl)* or *(rec)*, then again the thesis follows from item i) of Lemma 2.9. If the last rule is *(symm)*, then the thesis is immediate by induction hypothesis, using item ii) of Lemma 2.9. If the last rule is *(trans)*, then again the thesis is immediate by induction hypothesis, using item iii) of Lemma 2.9. If the last rule in τ is one of the following ($\times$*cong*), ($+_1$*cong*), ($+_2$*cong*), ([ ]*cong*), *(abs)*, then then the thesis is immediate. Finally, if the last rule in τ is *(hyp)* or *(in)*, then the thesis follows immediately from the induction hypothesis.

$\square$

2.2 Completeness of $\mathcal{S}_{co}$

In order to show the completeness of the system $\mathcal{S}_{co}$, we need to exploit the implicit regularity of the terms expressible in our language. Namely, we introduce the notion of set of subterms of a given term.

Definition 2.11. Let $t \in T_\sigma$. The set of subterms of t, $sub(t)$, is defined by induction on t as follows:

- if $t \equiv x \in Var$ or $t \equiv c \in C$, then $sub(t) = \{t\}$;
- if $t \equiv i_j(t')$, for some $j \in \{1, 2\}$, then $sub(t) = \{t\} \cup sub(t')$;
- if $t \equiv < t_1, t_2 >$, then $sub(t) = \{t\} \cup sub(t_1) \cup sub(t_2)$;
- if $t \equiv [t_1, \ldots, t_n]$, for some $n \geq 0$, then $sub(t) = \{t\} \cup \bigcup_{i=1,\ldots,n} sub(t_i)$;
- if $t \equiv in(t')$, then $sub(t) = \{t\} \cup sub(t')$;
- if $t \equiv rec\, x.t'$, then $sub(t) = \{t\} \cup \{t_1[t/x] \mid t_1 \in sub(t')\}$.

The following lemma can be immediately shown by induction on terms.

Lemma 2.12. *For all σ and for all $t \in T_\sigma$,*
i) the set $sub(t)$ is finite;
ii) $\forall t' \in sub(t).\ sub(t') \subseteq sub(t)$.

Now we are in the position of stating the Completeness Theorem for the system $\mathcal{S}_{co}$. The proof of this theorem consists in showing that, if two terms $t, t' \in T^0_\sigma$ are $\approx_\sigma$-bisimilar, then, since they have only a finite number of subterms, we can build a derivation of $\vdash_{co} t \sim t' : \sigma$ in a top-down fashion.

Theorem 2.13 (Completeness). *Let $\sigma \in Type$. For all $t, t' \in T^0_\sigma$,*

$$t \approx_\sigma t' \implies \vdash_{co} t \sim t' : \sigma \ .$$

Proof. We prove that, if $t \approx_\sigma t'$, then for all $t_1, \ldots, t_n, \bar{t} \in sub(t)$, $t'_1, \ldots, t'_n, \bar{t}' \in sub(t')$ such that $\forall i = 1, \ldots, n.\ t_i, t'_i \in T^0_{\sigma_i}$ & $t_i \approx_{\sigma_i} t'_i$, $\bar{t}, \bar{t}' \in T^0_{\bar{\sigma}}$ and $\bar{t} \approx_{\bar{\sigma}} \bar{t}'$, there exists a derivation of $t_1 \sim t'_1 : \sigma_1, \ldots, t_n \sim t'_n : \sigma_n \vdash_{co} \bar{t} \sim \bar{t}' : \bar{\sigma}$.
Suppose by contradiction that $t_1 \sim t'_1 : \sigma_1, \ldots, t_n \sim t'_n : \sigma_n \vdash_{co} \bar{t} \sim \bar{t}' : \bar{\sigma}$ is not derivable. Then we show that there exists an infinite sequence of distinct pairs of processes $t_i, t'_i \in T^0_{\sigma_i}$ such that $t_i \approx_{\sigma_i} t'_i$ and $t_i \in sub(t)$, $t'_i \in sub(t')$,

for $i = 1, \ldots, n$, which is clearly impossible because, by Lemma 2.12, $sub(t)$ and $sub(t')$ are finite. In fact, if $t_1 \sim t'_1 : \sigma_1, \ldots, t_n \sim t'_n : \sigma_n \vdash_{co} \overline{t} \sim \overline{t}' : \overline{\sigma}$ is not derivable, then we show that a sequent of the following shape is not derivable: $t_1 \sim t'_1 : \sigma_1, \ldots, t_n \sim t'_n : \sigma_n, t_{n+1} \sim t'_{n+1} : \sigma_{n+1} \vdash_{co} \widehat{t} \sim \widehat{t'} : \widehat{\sigma}$, for some $t_{n+1}, \widehat{t} \in sub(t)$, $t'_{n+1}, \widehat{t'} \in sub(t')$, such that $t_{n+1} \approx_{\sigma_{n+1}} t'_{n+1}$, $\widehat{t} \approx_{\widehat{\sigma}} \widehat{t}'$, and the hypothesis $t_{n+1} \sim t'_{n+1} : \sigma_{n+1}$ is new, in the sense that it does not appear among $t_1 \sim t'_1 : \sigma_1, \ldots, t_n \sim t'_n : \sigma_n$. This latter fact is proved by induction on the structure of $\overline{\sigma}$.
If $\overline{\sigma} \equiv K_j$, then the sequent $t_1 \sim t'_1 : \sigma_1, \ldots, t_n \sim t'_n : \sigma_n \vdash_{co} \overline{t} \sim \overline{t}' : K_j$ is immediately derivable, since $\overline{t} \approx_{K_j} \overline{t}' \;\Rightarrow\; t = t' \in C_j$.
If $\overline{\sigma} \equiv \nu X_1.\sigma_1$, then there exists $m, n \geq 0$ such that $\overline{t} \equiv rec\, x_1 \ldots rec\, x_m.in(\widetilde{t})$ and $\overline{t'} \equiv rec\, x_1 \ldots rec\, x_n.in(\widetilde{t'})$, for some terms $\widetilde{t}, \widetilde{t}'$. Then, by rule *(in)* (possibly using rules (rec), *(symm)*, and *(trans)*), also $t_1 \sim t'_1 : \sigma_1, \ldots, t_n \sim t'_n : \sigma_n, in(\widetilde{t})[\overline{t}/x_1, \ldots, \overline{t}/x_m] \sim in(\widetilde{t})[\overline{t}'/x_1, \ldots, \overline{t}'/x_n] : \nu X_1.\sigma_1 \vdash_{co} \widetilde{t}[\overline{t}/x_1, \ldots, \overline{t}/x_m] \sim \widetilde{t}[\overline{t}'/x_1, \ldots, \overline{t}'/x_n] : \nu X_1.\sigma_1$ is not derivable, and the pair $in(\widetilde{t})[\overline{t}/x_1, \ldots, \overline{t}/x_m] \sim in(\widetilde{t})[\overline{t}'/x_1, \ldots, \overline{t}'/x_n] : \nu X_1.\sigma_1$ is new among $t_1 \sim t'_1 : \sigma_1, \ldots, t_n \sim t'_n : \sigma_n$, otherwise we would already have a proof of the sequent $t_1 \sim t'_1 : \sigma_1, \ldots, t_n \sim t'_n : \sigma_n \vdash_{co} \overline{t} \sim \overline{t}' : \overline{\sigma}$.
If $\overline{\sigma} \equiv \sigma_1 + \sigma_2$, then $\overline{t} \equiv i_j(\overline{t}_j)$ and $\overline{t}' \equiv i_j(\overline{t}'_j)$, for some $j \in \{1, 2\}$ and, by rule $(+_j\, cong)$ (possibly using rules (rec), *(symm)*, and *(trans)*), also the sequent $t_1 \sim t'_1 : \sigma_1, \ldots, t_n \sim t'_n : \sigma_n \vdash_{co} \overline{t}_j \sim \overline{t_j}' : \sigma_j$ is not derivable. Hence we can apply the induction hypothesis to σ_j, since, by definition of $i_j(\overline{t}_j) \approx_{\sigma_1+\sigma_2} i_j(\overline{t}'_j)$, we have also $\overline{t}_j \approx_{\sigma_j} \overline{t}'_j$.
Finally, the cases $\overline{\sigma} \equiv \sigma_1 \times \sigma_2$ and $\overline{\sigma} \equiv \mathcal{P}_f(\sigma_1)$ are dealt with similarly to the previous case. $\square$

The proof of Theorem 2.13 above is given by contradiction just for the sake of conciseness. Clearly a *constructive* proof can be easily obtained from the proof above. As a side-remark, we point out that a proof of decidability of $\sim$-equivalence can be easily obtained using the argument of the above proof.

3 Categorical Semantics

In this section we give a categorical final semantics in the style of [Acz88, RT93, Len96, Rut96, Len98] (to which we refer for further details on this topic) to our language, and we show that it captures exactly the greatest fixed point semantics of Section 1.

The interest of this categorical semantics is that it achieves a significant degree of generality, in that it subsumes naturally a great number of concrete examples of infinite objects in programming. The significance of a final semantics for a language like ours is that, contrary to the fixed point semantics, it allows us to embody as a point of a final coalgebra a canonical "minimal" representative for each equivalence class of terms. These denotations are the mathematical counterparts of our intuitive circular objects. Notice that defining a final semantics for a language with a given notion of equivalence is not a mechanical task.

We work in the category Set^* of non-wellfounded sets and set-theoretic functions for simplicity, but we could have also worked in other categories based on sets. Denotations would have become rather obscure however. We proceed as follows. We define a "universal" endofunctor F, embodying constructors corresponding to the type constructors. Then we endow the set $\Sigma_{\sigma \in Type} T^0_\sigma$, i.e. the disjoint sum of all closed typable terms, with a structure of F-coalgebra. Finally, we show that the largest F-bisimulation on the coalgebra defined on $\Sigma_{\sigma \in Type} T^0_\sigma$ coincides with the family of bisimulation equivalences $\{\approx_\sigma\}_\sigma$ introduced in Section 1.

Our categorical semantics could be equivalently presented in the framework of categories indexed over types. But, for the sake of simplicitly, we prefer the set-theoretic setting.

For more informations on the Final Semantics paradigm see e.g. [Len98].

Definition 3.1. Let $F : Set^* \to Set^*$ be the functor defined by:

$$F(X) = \Sigma_{\sigma \in Type}(\Sigma_{j \leq n} C_j + (X + X) + (X \times X) + \mathcal{P}_f(X)) \ .$$

We endow the set $\Sigma_{\sigma \in Type} T^0_\sigma$ with a structure of F-coalgebra as follows:

Definition 3.2. Let $\alpha : \Sigma_{\sigma \in Type} T^0_\sigma \to F(\Sigma_{\sigma \in Type} T^0_\sigma)$ be the function defined by:

$$\alpha(t) = (\sigma, z) \ ,$$

where

$$z = \begin{cases} \mathrm{in}_{K_j}(c^i_j) & \text{if } A_{K_j} \\ \mathrm{in}_+(i_j(t_1)) & \text{if } A_{\sigma_1+\sigma_2} \\ \mathrm{in}_\times(t_1, t_2) & \text{if } A_{\sigma_1 \times \sigma_2} \\ \mathrm{in}_{\mathcal{P}_f}([t_1, \ldots, t_m]) & \text{if } A_{\mathcal{P}_f(\sigma_1)} \\ \mathrm{in}_{K_j}(c^i_j) & \text{if } A_{\nu X.K_j} \\ \mathrm{in}_+(i_j(t_1[t/x_1, \ldots, t/x_n])) & \text{if } A_{\nu X.\sigma_1+\sigma_2} \\ \mathrm{in}_\times(t_1[t/x_1, \ldots, t/x_n], t_2[t/x_1, \ldots, t/x_n]) & \text{if } A_{\nu X.\sigma_1 \times \sigma_2} \\ \mathrm{in}_{\mathcal{P}_f}([t_1[t/x_1, \ldots, t/x_n], \ldots, t_m[t/x_1, \ldots, t/x_n]]) & \text{if } A_{\nu X.\mathcal{P}_f(\sigma_1)} \end{cases}$$

where in_{K_j}, in_+, $\mathrm{in}_\times$, $\mathrm{in}_{\mathcal{P}_f}$ denote canonical injections into disjoint sum and

$$\begin{array}{ll} A_{K_j} & \equiv (\sigma \equiv K_j \wedge t \equiv c^i_j, i \in I_j) \\ A_{\sigma_1+\sigma_2} & \equiv (\sigma \equiv \sigma_1 + \sigma_2 \wedge t \equiv i_j(t_1)) \\ A_{\sigma_1 \times \sigma_2} & \equiv (\sigma \equiv \sigma_1 \times \sigma_2 \wedge t \equiv < t_1, t_2 >) \\ A_{\mathcal{P}_f(\sigma_1)} & \equiv (\sigma \equiv \mathcal{P}_f(\sigma_1) \wedge t \equiv [t_1, \ldots, t_m]) \\ A_{\nu X.K_j} & \equiv (\sigma \equiv \nu X.K_j \wedge t \equiv rec\, x_1 \ldots rec\, x_n.in(c^i_j), i \in I_j) \\ A_{\nu X.\sigma_1+\sigma_2} & \equiv (\sigma \equiv \nu X.\sigma_1 + \sigma_2 \wedge t \equiv rec\, x_1 \ldots rec\, x_n.in(i_j(t_1))) \\ A_{\nu X.\sigma_1 \times \sigma_2} & \equiv (\sigma \equiv \nu X.\sigma_1 \times \sigma_2 \wedge t \equiv rec\, x_1 \ldots rec\, x_n.in(< t_1, t_2 >)) \\ A_{\nu X.\mathcal{P}_f} & \equiv (\sigma \equiv \nu X.\mathcal{P}_f(\sigma_1) \wedge t \equiv rec\, x_1 \ldots rec\, x_n.in([t_1, \ldots, t_m])). \end{array}$$

Now, our goal is that of showing that the largest F-bisimulation on the coalgebra $(\Sigma_\sigma T^0_\sigma, \alpha)$, which we denote by $\sim$, coincides exactly with the family of bisimulation equivalences $\{\approx_\sigma\}_\sigma$ defined in Section 1. First of all, we recall the definition of categorical F-bisimulation:

Definition 3.3. Let $F : Set^* \to Set^*$. An F-bisimulation on the F-coalgebra (X, α_X) is a set-theoretic relation $R \subseteq X \times X$ such that there exists an arrow of Set^*, $\gamma : \mathcal{R} \to F(\mathcal{R})$, making the following diagram commute:

$$\begin{tikzcd}
X \arrow[d, "\alpha_X"'] & \mathcal{R} \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] \arrow[d, "\gamma"] & X \arrow[d, "\alpha_X"] \\
F(X) & F(\mathcal{R}) \arrow[l, "F(\pi_1)"] \arrow[r, "F(\pi_2)"'] & F(X)
\end{tikzcd}$$

The proof of the following proposition is routine:

Proposition 3.4. *The largest F-bisimulation on the coalgebra $(\Sigma_\sigma T^0_\sigma, \alpha)$ is the family $\{\approx_\sigma\}_\sigma$.*

4 Final Remarks and Directions for Future Work

In this paper, we have presented a "coinductive" axiomatization of the bisimulation equivalence on non-wellfounded regular objects. Moreover, we have shown that it is complete with respect to a maximal fixed point semantics and also to a categorical semantics. Our presentation makes use of a typed language for denoting circular terms.

We could generalize our language of terms so as to allow non-regular objects, still getting a sound axiomatization. In fact, the regularity property is crucial only for proving the completeness of our system.

There are various other promising directions for possible generalizations and extensions of the coinductive axiomatization presented in this paper.

- Categories other than the purely set-theoretical ones could be investigated. This would involve the use of a generalized notion of set-theoretic relation. In the case of c.p.o.'s, this should go in the direction of providing a formal system for expressing Pitts' relational structures ([Pit96]).
- A richer collection of types, including inductive types and the *mixed* covariant-contravariant $\to$ constructor could be considered, as well as destructors in terms.
- Coarser notions of bisimulations other than Milner's strong bisimulation could be considered, e.g. weak bisimulation and congruence, van Glabbeek-Weijland branching bisimulation, Montanari-Sassone dynamic bisimulation.
- Other coinductively defined equivalences, arising in different contexts, could be considered. E.g. equivalence of streams representing exact reals.
- Finally, it would be interesting to compare systems like $\mathcal{S}_{co}$ to other logics for bisimulations (see e.g. [Mos?]).

References

Acz88. P.Aczel. *Non-well-founded sets*, CSLI Lecture Notes **14**, Stanford 1988.

BV96. J.de Bakker, E.de Vink. *Control Flow Semantics*, Foundations of Computing Series, The MIT Press, Cambridge, 1996.

BM96. J.Barwise, L.Moss. *Vicious circles: On the mathematics of non-wellfounded phenomena*, CSLI Publications, Stanford, 1996.

BH97. M.Brandt, F.Henglein. *Coinductive axiomatization of recursive type equality and subtyping*, *TLCA'97* Conf. Proc., P.de Groote, R.Hindley, eds., Springer LNCS **1210**, 1997, 63–81.

Coq94. T.Coquand. Infinite Objects in Type Theory, Types for Proofs and Programs *TYPES-93*, Springer LNCS **806**, 1994, 62–78.

Fio96. M.Fiore. A Coinduction Principle for Recursive Data Types Based on Bisimulation, *Inf. & Comp.* **127**, 1996, 186–198.

FH83. M.Forti, F.Honsell. Set theory with free construction principles, *Ann. Scuola Norm. Sup. Pisa*, Cl. Sci. (4) **10**, 1983, 493– 522.

Gim94. E.Giménez. Codifying guarded definitions with recursive schemes, *Workshop on Types for Proofs and Programs*, P.Dybjer et al. eds., Springer LNCS **996**, 1994, 39–59.

Gim95. E.Giménez. An application of co-Inductive types in Coq: verification of the Alternating Bit Protocol, *Workshop Types Proofs and Programs*, 1995.

Gim96. E.Giménez. Un calcul de constructions infinies et son application a la verfication de systemes communicants, PhD Thesis, École normale supérieure de Lyon, December 1996.

HL95. F.Honsell, M.Lenisa. Final Semantics for Untyped Lambda Calculus, *TLCA'95* Conf. Proc., M.Dezani et al eds., Springer LNCS **902**, 1995, 249–265.

Len96. M.Lenisa. Final Semantics for a Higher Order Concurrent Language, *CAAP'96*, H.Kirchner et. al. eds., Springer LNCS **1059**, 1996, 102–118.

Len98. M.Lenisa. Themes in Final Semantics, Ph.D. Thesis **TD-6/98**, Dipartimento di Informatica, Università di Pisa, March 1998.

MPC86. N.P.Mendler, P.Panangaden, R.L.Constable. Infinite Objects in Type Theory, 1th *LICS* Conf. Proc., IEEE Computer Society Press, 1986, 249–255.

Mil83. R.Milner. Calculi for synchrony and asynchrony, *TCS* **25**, 1983, 267–310.

Mil84. R.Milner. A complete inference system for a class of regular behaviours, *J. of Computer and System Sciences* **28**, 1984, 39–466.

Mos?. L.Moss. Coalgebraic Logic, to appear in the *Annals of Pure and Applied Logic*.

Pit96. A.M.Pitts. Relational Properties of Domains, *Inf. & Comp.* **127**, 1996.

Plo85. G.Plotkin. Types and Partial Functions, Post-Graduate Lecture Notes, Department of Computer Science, University of Edinburgh, 1985.

Rut96. J.J.M.M.Rutten. Universal coalgebra: a theory of systems, Report **CS-R9652**, CWI, Amsterdam, 1996.

RT93. J.J.M.M.Rutten, D.Turi. On the Foundations of Final Semantics: Non-Standard Sets, Metric Spaces, Partial Orders, *REX* Conf. Proc., J.de Bakker et al. eds., Springer LNCS **666**, 1993, 477–530.

Tal90. C.Talcott. A Theory for Program and Data type Specification, *TCS*, Disco90 Special Issue, 1990.

Tur96. D.Turi. *Functorial Operational Semantics and its Denotational Dual*, PhD thesis, CWI, 1996.