Denotational testing semantics in coinductive form*

Michele Boreale[1] and Fabio Gadducci[2]

[1] Dipartimento di Sistemi e Informatica, Università di Firenze
Via Lombroso 6/17, 50134 Firenze, Italia. `boreale@dsi.unifi.it`
[2] Dipartimento di Informatica, Università di Pisa
via Buonarroti 2, 56125 Pisa, Italia. `gadducci@di.unipi.it`

Abstract. Building on recent work by Rutten on coinduction and formal power series, we define a denotational semantics for the CSP calculus and prove it fully abstract for testing equivalence. The proposed methodology allows for abstract definition of operators in terms of *behavioural differential equations* and for coinductive reasoning on them, additionally dispensing with continous order-theoretic structures.

Keywords: process calculi, coinduction, formal power series, testing equivalence.

1 Introduction

Testing equivalence [4, 5] and bisimilarity [7] are two classical proposals for process calculi semantics. They offer different tradeoffs between mathematical tractability and accuracy of process description. Bisimilarity comes equipped with with a nice coinductive proof technique. However, it lacks a natural denotational model, and is often blamed of being over-discriminating. Testing equivalence offers perhaps a more faithful picture of reality, with e.g. a proper distinction between termination and divergence, and comes equipped with a fully abstract denotational model. Unfortunately, it lacks tractable proof techniques.

In this paper we make an attempt at reconciling testing and bisimulation, while keeping the benefits of both. The key to reconciliation are *formal power series* (FPS's) over a semiring, and the related finality and coinduction principle, as presented in recent work by Rutten [9, 10].

A formal power series is a function from the set of words over an alphabet A to a generic semiring $\mathcal{K}$. The set of such objects, denoted $\mathcal{K}\langle A\rangle$, can be given an automaton structure, with inputs in A and outputs in $\mathcal{K}$. This particular automaton is *final*, in the sense that there is a unique homomorphism onto it from every automaton on $\mathcal{K}$, and enjoys moreover a *coinduction principle*, by which the unique homomorphism maps two bisimilar states into the same FPS.

* Research partially supported by the the EU within the FET – Global Computing Initiative, project MIKADO, and by the Italian MIUR projects COMETA and NAPOLI.

In this paper, we consider a simple process calculus and introduce a semiring for testing, $\mathcal{K}_T$. Next, we turn the calculus into an automaton $\mathcal{A}ut$ on $\mathcal{K}_T$ and show that bisimulation over $\mathcal{A}ut$ coincide with (must) testing equivalence. Hence, the unique homomorphism from $\mathcal{A}ut$ to $\mathcal{K}_T\langle A\rangle$ yields a fully abstract semantics for testing equivalence. Finally, we define a set of operators on $\mathcal{K}_T\langle A\rangle$ and show that the final homorphism is a morphism with respect to these operators, meaning that the semantics is a denotational one. One nontrivial point of this construction is the treatment of divergence – the possibility for a process of getting engaged in an infinite sequence of internal actions – that is not easily dealt with via bisimulation. In fact, we found it convenient to introduce *partial* formal power series, and to modify the notion of homomorphism accordingly.

The benefits of the above methodology can be summarized as follows.

- Simplicity of the semantic domain. In particular, we dispense with continous (order-theoretic, metric, topological,...) structures, and the existence of least fixpoints relies solely on the automaton structure of formal power series.
- Uniform and abstract definitions of operators on the semantic domain, via *behavioural differential equations* (BDE's, [9]). This benefit shows up clearly upon comparison of BDE's with the somehow intricate formulations often found in a standard denotational settings (see e.g. [5]).
- Coinductive reasoning. Proofs by coinduction, which amount to exhibiting apropriate (bi)simulation relations, are used to show existence of least fixpoints, full abstraction and compositionality of the semantics.

The rest of the paper is organized as follows. In Section 2 we introduce (a substantial fragment of) Hoare's CSP [6], and present testing equivalence for this calculus. In Section 3, building on [9], we introduce partial formal power series and the related coinduction and finality principles. The subsequent Section 4 introduces the semiring for testing $\mathcal{K}_T$ and the automaton $\mathcal{A}ut$ on CSP processes, then showing that the resulting final homomorphism is fully abstract for testing equivalence. In Section 5 we equip the set of FPS's with an algebraic structure and prove that the final homomorphism is compositional with respect to this structure. Section 6 discusses related works and further directions for research.

2 CSP and testing semantics

We introduce a simple process calculus, essentially Hoare's CSP [6], and then present the associated testing equivalence semantics.

2.1 Syntax and operational semantics

We assume a countable set of *visible actions*, denoted by A and ranged over by $a, b, \ldots$; a set X of *agent variables*, ranged over by $x, y, \ldots$; and an *invisible action* $\tau \notin A$, with the set $A \cup \{\tau\}$ and $\wp_f(A)$ (the finite subsets of A) ranged over by μ and L, respectively. Terms are built according to the following syntax

$$P ::= x \mid \mathsf{nil} \mid a.P \mid P \oplus P \mid P + P \mid P[a] \mid P \parallel_L P \mid rec_x.P\,.$$

A term is *closed* if each occurrence of a variable x is in the scope of a rec_x operator. The set of closed terms, or *processes*, is denoted by $\mathcal{P}$ and ranged over by $P, Q, R, \ldots$.

The constant nil represents the terminated process. The action prefix $a.P$ performs an atomic action a and evolves to P. The operator $\oplus$ describes non-deterministic *internal* choice: $P \oplus Q$ may evolve via an invisible action either to P or to Q. Summation $+$ denotes non-deterministic *external* choice: $P + Q$ behaves either as P or as Q, the choice being triggered by the execution of a visible action. In the parallel composition $P \parallel_L Q$, processes P and Q must evolve sinchronously with respect to each action $a \in L$, while they may evolve independently with any $\mu \notin L$. The process $P[a]$ behaves like P, except that any execution of action a is "hidden" and turned into τ. The intended meaning of the *recursion* operator $rec_x.P$ is the process defined by the equation $x = P$.

The operational semantics of this language is described by a *labeled transition system* defined in the so-called SOS style by the set of axioms and inference rules of Table 1 (where we have omitted the symmetric rules for the parallel composition and the two choice operators).

$$act : \frac{-}{a.P \xrightarrow{a} P} \qquad rec : \frac{-}{rec_x.P \xrightarrow{\tau} P[{}^{rec_x.P}/_x]} \qquad plus : \frac{-}{P \oplus Q \xrightarrow{\tau} P}$$

$$sum_a : \frac{P \xrightarrow{a} P'}{P + Q \xrightarrow{a} P'} \qquad sum_\tau : \frac{P \xrightarrow{\tau} P'}{P + Q \xrightarrow{\tau} P' + Q}$$

$$hide_a : \frac{P \xrightarrow{a} P'}{P[a] \xrightarrow{\tau} P'[a]} \qquad hide_\mu : \frac{P \xrightarrow{\mu} P'}{P[a] \xrightarrow{\mu} P'[a]} \; \mu \neq a$$

$$par_L : \frac{P \xrightarrow{a} P', \; Q \xrightarrow{a} Q'}{P \parallel_L Q \xrightarrow{a} P' \parallel_L Q'} \; a \in L \quad par_\mu : \frac{P \xrightarrow{\mu} P'}{P \parallel_L Q \xrightarrow{\mu} P' \parallel_L Q} \; \mu \notin L$$

Table 1. The operational semantics for $\mathcal{P}$.

2.2 Testing semantics

The idea underlying *testing* semantics is that of equating two processes whenever no external observer can tell them apart. By "external observer" we mean any process, running in parallel with the observed process (see [4, 5]). Within the testing approach one distinguishes between *may* and *must* semantics: informally, the former is meant to preserve safety properties of a process, while the latter is meant to preserve liveness ones. In the sequel we shall consider the must variant, which is by far more challenging, both because it takes into account the branching structure of processes, and because of the special role played by *divergence* in this semantics. For technical convenience, we shall rely on an alternative, observer-independent characterization of this equivalence, given below. This definition is easily proved to coincide with the original one (see e.g. [3, 5]).

Definition 1 (testing equivalence [4, 5]). *Let* A^* *denote the set of* finite words *on* A*, ranged over by* ω*. Let us consider the relations and predicates*

- $\Rightarrow \stackrel{def}{=} (\stackrel{\tau}{\longrightarrow})^*$;
- $\stackrel{a}{\Longrightarrow} \stackrel{def}{=} \Rightarrow \stackrel{a}{\longrightarrow} \Rightarrow$ *for any* $a \in A$;
- $\stackrel{\omega}{\Longrightarrow} \stackrel{def}{=} \stackrel{a_1}{\Longrightarrow} \dots \stackrel{a_n}{\Longrightarrow}$ *for any* $\omega = a_1 \cdots a_n \in A^*$;
- $I(P) \stackrel{def}{=} \{a \mid \exists Q : P \stackrel{a}{\Longrightarrow} Q\}$;
- $P \Downarrow$ *holds true iff there is no infinite sequence of* τ *actions* $P \stackrel{\tau}{\longrightarrow} \stackrel{\tau}{\longrightarrow} \cdots$ *starting from* P *(otherwise* $P \Uparrow$ *holds);*
- $P \Downarrow \omega$ *(read* P converges on ω*) holds true iff for each prefix* ω' *of* ω*, whenever* $P \stackrel{\omega'}{\Longrightarrow} P'$ *then* $P' \Downarrow$ *(otherwise* $P \Uparrow \omega$ *holds).*

Given two processes P *and* Q*, we say that they are* testing equivalent*, and write* $P \simeq Q$*, if for each* $\omega \in A^*$*, whenever* $P \Downarrow \omega$*, it holds that*

(a) $Q \Downarrow \omega$*, and*
(b) if $Q \stackrel{\omega}{\Longrightarrow} Q'$ *then there is* P' *such that* $P \stackrel{\omega}{\Longrightarrow} P'$ *and* $I(P') \subseteq I(Q')$*,*

and vice-versa for Q *and* P*.*

3 Coinduction on partial power series

This section presents a few definitions and results on formal power series, Moore automata and the related coinduction principles. They are directly inspired by [9]. However, Rutten's treatment is extended in order to take into account *partially defined* formal power series.

3.1 Formal power series and Moore automata

Definition 2 (semirings). *A (commutative, unitary) semiring is a five-tuple* $\mathcal{K} = \langle K, \oplus_{\mathcal{K}}, \otimes_{\mathcal{K}}, 0_{\mathcal{K}}, 1_{\mathcal{K}} \rangle$ *such that* K *is a set,* $0_{\mathcal{K}}, 1_{\mathcal{K}} \in K$*, and* $\oplus_{\mathcal{K}}, \otimes_{\mathcal{K}} : K \times K \to K$ *are binary operators on* K *making the triples* $\langle K, \oplus_{\mathcal{K}}, 0_{\mathcal{K}} \rangle$ *and* $\langle K, \otimes_{\mathcal{K}}, 1_{\mathcal{K}} \rangle$ *commutative monoids, additionally satisfying*

- $x \otimes_{\mathcal{K}} (y \oplus_{\mathcal{K}} z) = (x \otimes_{\mathcal{K}} y) \oplus (x \otimes_{\mathcal{K}} z)$ *for all* $x, y, z \in K$;
- $0_{\mathcal{K}} \otimes_{\mathcal{K}} x = 0_{\mathcal{K}}$ *for all* $x \in K$.

In the sequel, we shall drop the subscript $_{\mathcal{K}}$ from the components of a semiring, if no confusion can arise.

Let A, B be sets. We denote by $B_\perp$ the extension of B with a new element $\perp$, and we model a *partial* function from A to B as a total function $f : A \to B_\perp$, writing $f(a)\uparrow$ if $f(a) = \perp$, and $f(a)\downarrow$ otherwise. A function $f : A^1_\perp \times \cdots \times A^n_\perp \to B_\perp$ is *strict* if f yields $\perp$ whenever one of its argument is $\perp$ and for any function $f : A^1 \times \cdots \times A^n \to B_\perp$, we let $f_\perp : A^1_\perp \times \cdots \times A^n_\perp \to B_\perp$ denote the *strict extension* of f, defined as expected. For any relation $R \subseteq A \times B$, we denote by $R_\perp$ the relation obtained by adding the pair $\langle \perp, \perp \rangle$ to R. Finally, given a semiring $\mathcal{K}$, by abuse of notation we shall use symbols $\oplus$ and $\otimes$ to denote also the strict extensions of the corresponding operators on $\mathcal{K}_\perp$.

Definition 3 (partial formal power series). *Let A be a set, and let $\mathcal{K}$ be a semiring. A* partial formal power series *(also partial* FPS*) on $\mathcal{K}$ and A is a function $\sigma : A^* \to K_\perp$, such that for all words $\omega \in A^*$, if $\sigma(\omega) \uparrow$ then $\sigma(\omega a) \uparrow$, for each $a \in A$. The set of all series on $\mathcal{K}$ and A is denoted by $\mathcal{K}\langle A \rangle$.*

Let σ be a series in $\mathcal{K}\langle A \rangle$, and let $\omega \in A^*$. The ω-input derivative of σ, *written σ_ω, is the series defined by $\sigma_\omega(\omega') = \sigma(\omega\omega')$ for all words $\omega' \in A^*$.*

Let $\sigma \in \mathcal{K}\langle A \rangle$, and let $\omega \in A^*$: the *coefficient* of ω with respect to σ is the value $\sigma(\omega)$, and $\sigma(\epsilon)$ is called the *constant* of the series. Moreover, a series σ is *total* if $\sigma(\omega) \downarrow$ for each $\omega \in A^*$. Of special interest are the following FPS's: $\mathbf{0}$ is the FPS yielding 0 for all coefficients, $\mathbf{1}$ is the FPS with constant 1 and 0 elsewhere, $\mathbf{\Omega}$ is the FPS yielding $\perp$ for all coefficients (everywhere undefined).

Coefficients bear different interpretations on different semirings. For example, let $A = \{X\}$ and let $\mathcal{K}$ be the semiring of real numbers: any total FPS represents a power series in the usual sense (interpreting the word $X \cdots X$, the element X replicated n times, as X^n). If A is any set, and $\mathcal{K}$ is the semiring of boolean values (i.e., true and false), a total FPS just represents a subset of A^*, hence, a language over A. There is no obvious interpretation for partial FPS's in these cases. As we shall see, partiality is well suited to represent divergence in processes.

In the rest of the section we let A denote a set, and $\mathcal{K}$ a semiring.

Definition 4 (Moore automaton). *A* (partial) Moore automaton *with inputs in A and outputs in $\mathcal{K}$ is a pair $\langle S, (o_S, \delta_S) \rangle$ consisting of a set S of* states, *and of a pair of functions: an* output function *$o_S : S \to K_\perp$ and a* transition function *$\delta_S : S \times A \to S_\perp$, such that if $o_S(s) \uparrow$, then $\delta_S(s, a) \uparrow$ for each $a \in A$.*

The output $o_S(s)$ yields an observation at state s, while the transition δ_S specifies the state $\delta_S(s, a)$ reached from s after the consumption of an input a.

3.2 Finality and coinduction

Definition 5 (homomorphism and bisimulation). *Let S, T be automata with inputs in A and outputs in $\mathcal{K}$. A* homomorphism *between S and T is a function $f : S \to T$ preserving input and output, i.e., such that $o_S(s) = o_T(f(s))$ and $f_\perp(\delta_S(s, a)) = \delta_T(f(s), a)$ for all $s \in S$ and $a \in A$.*

A bisimulation *is a relation $R \subseteq S \times T$ preserving input and output, i.e., such that if $\langle s, t \rangle \in R$ then $o_S(s) = o_T(t)$ and $\langle \delta_S(s, a), \delta_T(t, a) \rangle \in R_\perp$ for all $a \in A$.*

Let S be an automaton, and let $s, s' \in S$: s and s' are *bisimilar* (denoted by $s \sim s'$) if there exists a bisimulation R between S and itself, such that R contains $\langle s, s' \rangle$. It is immediate to show that $\sim$ is a bisimulation, and that in fact it is an equivalence relation on S.

We now recall how to turn the set of FPS's into an automaton, using input derivatives in order to define both input and transition functions. The resulting automaton has the property of being final in the class of automata on A and $\mathcal{K}$, and additionally, it satisfies a coinduction principle, in the sense that over this automaton bisimilarity coincides with the identity relation.

Definition 6 (the automaton of partial FPS**'s).** *The automaton* $\mathcal{M}\langle\mathcal{K}\langle A\rangle\rangle$ *on* A *and* $\mathcal{K}$ *is the pair* $\langle\mathcal{K}\langle A\rangle, (o_{\mathcal{M}}, \delta_{\mathcal{M}})\rangle$*, where the output and transition functions are defined by* $o_{\mathcal{M}}(\sigma) = \sigma(\epsilon)$ *and* $\delta_{\mathcal{M}}(\sigma, a) = \sigma_a$ *for all series* σ *and* $a \in A$*.*

The proof of the following proposition goes along the same lines of the corresponding one in [9].

Proposition 1 (finality and coinduction). *The automaton* $\mathcal{M}\langle\mathcal{K}\langle A\rangle\rangle$ *satisfies the* coinduction principle*: for all series* $\sigma, \sigma' \in \mathcal{K}\langle A\rangle$*, if* $\sigma \sim \sigma'$ *then* $\sigma = \sigma'$*.*

Moreover, $\mathcal{M}\langle\mathcal{K}\langle A\rangle\rangle$ *is* final*: for any automaton* S *with inputs in* A *and outputs in* $\mathcal{K}$ *there exists a unique homomorphism* $l : S \to \mathcal{K}\langle A\rangle$*, additionally satisfying* $s \sim s'$ *in* S *iff* $l(s) = l(s')$*.*

4 A semiring for testing equivalence

The main result of this section is that, for an appropriate choice of the semiring, the set of CSP processes can be turned into a Moore automaton , in such a way that testing equivalence on the original transition system corresponds to bisimulation on that automaton. This will yield a fully abstract semantics for testing equivalence, in terms of the unique mapping into the final automaton.

4.1 The semiring $\mathcal{K}_T$

We start by presenting a concept routinely used in characterisations of testing equivalences, that is, *saturated sets* [4, 5].

Definition 7 (saturated sets). *Let* $F \subseteq_{\text{fin}} \wp_f(A)$*. We say that* F *is* saturated *if for all* $X \in F$*, whenever there exists* Y *such that* $X \subseteq Y \subseteq (\bigcup_{Z\in F} Z)$*, then* $Y \in F$*. The* saturation *of* F*, written* $\mathcal{S}(F)$*, is the smallest saturated family which contains* F*. The set of all saturated families on* A *is denoted by* $\mathcal{F}(A)$*.*

Lemma 1. *Let* $F, G \subseteq_{\text{fin}} \wp_f(A)$*. Let us assume (a)* $(\bigcup_{X\in F} X) = (\bigcup_{Y\in G} Y)$ *and (b) for each* $X \in F$ *there exists* $Y \in G$ *such that* $Y \subseteq X$ *(and vice-versa). Then,* $\mathcal{S}(F) = \mathcal{S}(G)$*.*

Let us now denote by $\mathcal{K}_T$ the five-tuple $\langle\mathcal{F}(A), \oplus_T, \otimes_T, \emptyset, \{\emptyset\}\rangle$, such that $F \oplus_T G = \mathcal{S}(F \cup G)$, and $F \otimes_T G = \{X \cup Y \mid X \in F, Y \in G\}$. It is an easy consequence of Lemma 1 that $\mathcal{K}_T$ is a semiring. Note in particular that $F \otimes_T G$ is obviously a saturated set, if so are F and G.

Our aim now is to individuate a subset of $\mathcal{K}_T\langle A\rangle$ which is apt for interpreting CSP processes. We shall do so by imposing a consistency requirement between output values and transitions at each state. First of all, we want to regard those derivatives σ_a leading to a state with 0 as output value as being *null*. So, let us define the *support of* σ, written $\mathrm{supp}(\sigma)$, as the set $\{a \mid \sigma(a) \neq 0\}$. Definition 8 below basically requires that all and only actions in the support contribute to the output value, at convergent states. This condition implies that, in those states: (a) the support is finite, and (b) if the output value is 0 (i.e. $\emptyset$) then the support is empty, which means that that state is actually $\mathbf{0}$.

Definition 8 (consistent partial FPS's). *The set of* consistent FPS*'s, denoted by* $\mathcal{K}^c_T\langle A\rangle$*, is the largest set of* FPS*'s, not containing* $\mathbf{0}$ *and closed under non null derivatives, such that if* $\sigma \in \mathcal{K}^c_T\langle A\rangle$ *and* $\sigma(\epsilon)\downarrow$ *then* $\mathrm{supp}(\sigma) = \bigcup_{Z\in\sigma(\epsilon)} Z$*.*

4.2 A Moore automaton for testing

The next step is turning the set of processes into an automaton on $\mathcal{K}_T$. In view of dealing with the interpretation of open terms (Section 5), we find it convenient to consider a larger set, which also includes constants for all consistent FPS's.

Let $\mathcal{P}^{\mathrm{e}}$ be the set of all closed terms built from the CSP operators, plus a set of distinct constants $\underline{\sigma}$, one for each $\sigma \in \mathcal{K}^c_T\langle A\rangle$. In order to extend the transition system of $\mathcal{P}$ to $\mathcal{P}^{\mathrm{e}}$, we introduce some notation. Given $D = \{P_1, P_2, \ldots, P_n\} \subseteq_{\mathrm{fin}} \mathcal{P}^{\mathrm{e}}$, we let $\bigoplus_{P\in D} P$ denote the process $P_1 \oplus P_2 \oplus \cdots \oplus P_n$, with summands P_i arranged in some fixed order and with the proviso that if $D = \emptyset$ then this term denotes the constant $\underline{\mathbf{0}} \notin \mathcal{P}^{\mathrm{e}}$. Similarly, we let $\Sigma_{P\in D} P$ denote the process $P_1 + P_2 + \cdots + P_n$, with the proviso that when $D = \emptyset$ then this term denotes the process nil. The transition system defined in Table 1 is extended to $\mathcal{P}^{\mathrm{e}}$ via the following (CCS-style) recursive definitions, one for each constant $\underline{\sigma} \in \mathcal{P}^{\mathrm{e}}$

$$\underline{\sigma} = \begin{cases} \bigoplus_{L\in\sigma(\epsilon)} \Sigma_{a\in L} a.\underline{\sigma_a} & \text{if } \sigma(\epsilon)\downarrow \\ rec_x.x & \text{if } \sigma(\epsilon)\uparrow . \end{cases}$$

The extended transition systems is still finitely branching, since consistent FPS's have finite support. Also note that testing equivalence on $\mathcal{P}^{\mathrm{e}}$ conservatively extends testing equivalence on $\mathcal{P}$.

We build now a Moore automaton out of $\mathcal{P}^{\mathrm{e}}$. We first need a technical lemma, standard from the theory of testing equivalence (see e.g. [5]).

Lemma 2. *Let* $P \in \mathcal{P}^{\mathrm{e}}$*. Then, the set* $I(P)$ *is finite and, for each* ω *such that* $P \Downarrow \omega$*, the set* $\{P' | \, P \stackrel{\omega}{\Longrightarrow} P'\}$ *is finite.*

Definition 9 ($\mathcal{P}^{\mathrm{e}}$ as a Moore automaton). *The Moore automaton* $\mathcal{A}ut$ *is the pair* $\langle \mathcal{P}^{\mathrm{e}} \cup \{\underline{\mathbf{0}}\}, (o, \delta)\rangle$*, and* δ *and* o *are defined as follows (assuming* $o(\underline{\mathbf{0}}) = \emptyset$ *and* $\delta(\underline{\mathbf{0}}, a) = \underline{\mathbf{0}}$ *for each* $a \in A$*)*

$$o(P) \stackrel{def}{=} \begin{cases} \mathcal{S}(\{I(Q) | \, P \Longrightarrow Q\}) & \text{if } P \Downarrow \\ \bot & \text{if } P \Uparrow \end{cases}$$
$$\delta(P, a) \stackrel{def}{=} \begin{cases} \bigoplus\{Q | \, P \stackrel{a}{\Longrightarrow} Q\} & \text{if } P \Downarrow a \\ \bot & \text{if } P \Uparrow a . \end{cases}$$

Proposition 2 (testing vs. bisimulation). *Let* $P, Q \in \mathcal{P}^{\mathrm{e}}$*. Then,* $P \simeq Q$ *if and only if* $P \sim Q$ *in* $\mathcal{A}ut$*.*

PROOF: We only show that $P \simeq Q$ implies $P \sim Q$ in $\mathcal{A}ut$, as the other direction is easier. Hence, it suffices to prove that $\simeq_{\underline{\mathbf{0}}} = \simeq \cup \{\langle \underline{\mathbf{0}}, \underline{\mathbf{0}}\rangle\}$ is a bisimulation.

Let R_a be a shorthand for $\delta(R, a)$, for each R in $\mathcal{A}ut$ and $a \in A$. Let P and Q be processes, such that $P \simeq Q$: it suffices to show that (a) $o(P) = o(Q)$ and that (b) if $o(P)\downarrow$ then $P_a \simeq_{\underline{\mathbf{0}}} Q_a$, for each $a \in A$ (the vice-versa for Q and P follows by symmetry). The proof is trivial if $P \Uparrow$, so let $P \Downarrow$. Fact (a) is a direct consequence of the definitions of o and δ, and of Lemma 1. Concerning fact (b), if $P_a = \underline{\mathbf{0}}$ then it must be $Q_a = \underline{\mathbf{0}}$, and vice-versa (i.e., $a \notin I(P) = I(Q)$), and the thesis follows trivially. Thus, let $P_a,\ Q_a \neq \underline{\mathbf{0}}$. Now, $o(P_a) = o(Q_a)$ is again a consequence of the definitions and of Lemma 1. Let us suppose that $o(P_a)\downarrow$, i.e. $P_a \Downarrow$ and that $P_a \Downarrow \omega$ for some ω. We have to show that (i) $Q_a \Downarrow \omega$ and that (ii) if $P_a \stackrel{\omega}{\Longrightarrow} P'$ then there is Q' such that $Q_a \stackrel{\omega}{\Longrightarrow} Q'$ and $I(Q') \subseteq I(P')$.

Now, $P_a \Downarrow \omega$ and $P \Downarrow$ imply $P \Downarrow a\omega$, this and $P \simeq Q$ in turn imply $Q \Downarrow a\omega$, hence $Q_a \Downarrow \omega$. Furthermore, by definition of P_a, if $P_a \stackrel{\omega}{\Longrightarrow} P'$ then $P \stackrel{a.\omega}{\Longrightarrow} P'$. This fact and $P \simeq Q$ imply that there is Q' such that $Q \stackrel{a.\omega}{\Longrightarrow} Q'$ and $I(Q') \subseteq I(P')$. By definition of Q_a, we have $Q_a \stackrel{\omega}{\Longrightarrow} Q'$, hence (i) and (ii) are proven. $\square$

Let l_T be the unique homomorphism $\mathcal{A}ut \to \mathcal{M}\langle\mathcal{K}_T\langle A\rangle\rangle$ induced by finality (Proposition1) and let $[\![\cdot]\!]$ denote its restriction to $\mathcal{P}^{\mathrm{e}}$.

Corollary 1 (full abstraction). *The mapping $[\![\cdot]\!] : \mathcal{P}^{\mathrm{e}} \to K_T\langle A\rangle$ is fully abstract for testing equivalence on $\mathcal{P}^{\mathrm{e}}$, i.e., $P \simeq Q$ if and only if $[\![P]\!] = [\![Q]\!]$.*

Proof: Note that $\mathcal{A}ut \supseteq \mathcal{P}^{\mathrm{e}}$ and that, by the definition of homomorphism) $l_T(\mathcal{P}^{\mathrm{e}}) \subseteq \mathcal{K}^{\mathrm{c}}_T\langle A\rangle$ Then, the coinduction principle (Proposition 1) and Proposition 2 ensure that $[\![\cdot]\!]$ is fully abstract for testing equivalence, hence the thesis. $\square$

5 Compositionality

The next step is to show that the final automaton can be equipped with an algebraic structure which is well suited to interpret processes. In fact, this structure will be used to prove that the mapping $[\![\cdot]\!]$ defined in the previous section also preserves the process operators, that is, $[\![\cdot]\!]$ yields a truly denotational semantics.

5.1 Algebraic operators via BDE's

A *behavioural differential equation* (BDE, [9]) can be used to define a FPS σ by specifying an initial condition (the value of $\sigma(\epsilon)$) and a condition on its input derivatives (the FPS's σ_a, for all $a \in A$). For example, the conditions $\sigma(\epsilon) = 1$ and $\sigma_a = \sigma$ (for each a) form a BDE whose unique solution is the FPS that associates 1 to every word in A^*. As we shall see, the coinduction principle on a final automaton $\mathcal{M}\langle\mathcal{K}\langle A\rangle\rangle$ allows one to show the existence and uniqueness of the solution of BDE's. In particular, BDE's can be used as a means to defining operators on $\mathcal{K}_T\langle A\rangle$ in a simple and uniform fashion.

Table 2 contains a set of BDE's, defining a few operators on $\mathcal{K}^{\mathrm{c}}_T\langle A\rangle$ clearly inspired by those of CSP. In fact, by abuse of notation, we shall use symbols drawn from CSP syntax to denote some of these operators. The equations in Table 2 deserve some explanation, but let us introduce the relevant notation first. For a finite set $D = \{\sigma_1, \ldots, \sigma_n\}$ of FPS's, $\bigoplus D$ denotes $\sigma_1 \oplus \cdots \oplus \sigma_n$, with the summands arranged in some fixed order, with the convention that this term denotes $\mathbf{0}$ if $D = \emptyset$. Finally, for any saturated set F and $L \subseteq_{\mathrm{fin}} A$, let $F \backslash\!\!\backslash L$ denote the obviously saturated set $\{X \backslash L |\, X \in F\}$.

In the equation for hiding $[b](\sigma)$, the undefined series $\mathbf{\Omega}$ models divergence, which may arise either because there is a finite sequence of b actions leading σ to a divergent state, or because σ has an infinite sequence of b actions. By the conditions imposed on consistent FPS's, either of these two cases arises iff there is no sequence of b's leading to $\mathbf{0}$. The constant coefficient of the parallel composition $\sigma \parallel_L \rho$ is the product of σ and ρ's constants, but synchronized actions (in L) that are not in the support of both are subtracted away. Finally note the form of the derivative for the $\otimes$ operator, intended to model external nondeterminism $+$, which is reminiscent of the CSP law $a.P + a.Q \;\simeq\; a.(P \oplus Q)$.

$$
\begin{array}{lcll}
(b.(\sigma))_a & = & \begin{cases} \sigma & \text{if } a = b \\ \mathbf{0} & \text{otherwise} \end{cases} & \text{with} \quad (b.(\sigma))(\epsilon) = \{\{\, b\,\}\} \\
(\sigma \oplus \rho)_a & = & \sigma_a \oplus \rho_a & \text{with} \quad (\sigma \oplus \rho)(\epsilon) = \sigma(\epsilon) \oplus_{\mathcal{K}} \rho(\epsilon) \\
(\sigma \otimes \rho)_a & = & \sigma_a \oplus \rho_a & \text{with} \quad (\sigma \otimes \rho)(\epsilon) = \sigma(\epsilon) \otimes_{\mathcal{K}} \rho(\epsilon) \\
([b](\sigma))_a & = & \begin{cases} \mathbf{0} & \text{if } a = b \text{ and } \exists j : \sigma_{b^j} = \mathbf{0} \\ [b](\bigoplus_{i \in \mathbb{N}} \sigma_{b^i a}) & \text{if } a \neq b \text{ and } \exists j : \sigma_{b^j} = \mathbf{0} \\ \mathbf{\Omega} & \text{otherwise} \end{cases} & \\
 & & \multicolumn{2}{l}{\text{with} \quad ([b](\sigma))(\epsilon) = \begin{cases} \bigoplus_{\mathcal{K}_T,\, i \in \mathbb{N}} \sigma(b^i) \backslash\!\!\backslash \{b\} & \text{if } \exists j : \sigma_{b^j} = \mathbf{0} \\ \perp & \text{otherwise} \end{cases}} \\
(\sigma \parallel_L \rho)_a & = & \begin{cases} \sigma_a \parallel_L \rho_a & \text{if } a \in L \\ (\sigma_a \parallel_L \rho) \oplus (\sigma \parallel_L \rho_a) & \text{otherwise} \end{cases} & \\
 & & \multicolumn{2}{l}{\text{with} \quad (\sigma \parallel_L \rho)(\epsilon) = (\sigma(\epsilon) \otimes_{\mathcal{K}} \rho(\epsilon)) \backslash\!\!\backslash (L \backslash (\mathrm{supp}(\sigma) \cap \mathrm{supp}(\rho))}
\end{array}
$$

Table 2. behavioural differential equations on $\mathcal{K}^{\mathrm{c}}_T\langle A\rangle$.

Proposition 3 (operators on $\mathcal{K}^{\mathrm{c}}_T\langle A\rangle$ via BDE's). *There exist unique* strict *binary operators $\oplus$, $\otimes$, $\parallel_L$ (for each $L \subseteq_{\mathrm{fin}} A$), unique* strict *unary operators $[b]$ and unique unary operators $b.$ (for each $b \in A$) on $\mathcal{K}^{\mathrm{c}}_T\langle A\rangle$ satisfying the BDE's of Table 2, for all $\sigma, \rho \in \mathcal{K}^{\mathrm{c}}_T\langle A\rangle$ and $a \in A$.*

PROOF: Let σ, ρ range over consistent series. We define our operators as follows: $b.(\sigma) \stackrel{def}{=} l_T(b.(\underline{\sigma}))$, $\sigma \oplus \rho \stackrel{def}{=} l_T(\underline{\sigma} \oplus \underline{\rho})$, $\sigma \otimes \rho \stackrel{def}{=} l_T(\underline{\sigma} \otimes \underline{\rho})$, and so on (note that the same symbol may denote an operator on series on the left-hand side of the definition, and a syntactic symbol on the right-hand side). It is an easy consequence of the homomorphism property of l_T that these operators satisfy the given BDE's, and that $l_T(\mathcal{P}^{\mathrm{e}}) \subseteq \mathcal{K}^{\mathrm{c}}_T\langle A\rangle$.

Concerning uniqueness, suppose that there are operators $b'.(\cdot)$, $\oplus'$, $\otimes'$ and so on satisfying the given BDE's. It is easy to prove, by exhibiting a suitable bisimulation relation, that $b.(\sigma) \sim b'.(\sigma)$, $\sigma \otimes \rho \sim \sigma \otimes' \rho$, $\sigma \parallel_L \rho \sim \sigma \parallel'_L \rho$, and so on, for each $\sigma, \rho \in \mathcal{K}^{\mathrm{c}}_T\langle A\rangle \cup \{\underline{\mathbf{0}}\}$. Hence, by the coinduction principle (see Proposition 1), it follows that $b'. = b.$, $\parallel'_L = \parallel_L$, $\otimes' = \otimes$ and so on. $\square$

5.2 The denotational mapping

In order to model recursion, we have to prove the existence of fixpoints in $\mathcal{K}^{\mathrm{c}}_T\langle A\rangle$. First, consider the partial ordering on $\mathcal{K}^{\mathrm{c}}_T\langle A\rangle$ given by inclusion, i.e., $\sigma \subseteq \tau$ if for each $\omega \in A^*$, $\sigma(\omega)\downarrow$ implies $\sigma(\omega) = \tau(\omega)$. We use $\subseteq$ as a criterion to select minimal solutions to recursive equations on $\mathcal{K}^{\mathrm{c}}_T\langle A\rangle$. In order to do so, all we need is a coinductive notion of *simulation.*

Definition 10 (simulation). *Let S an automaton with inputs in A and outputs in $\mathcal{K}$. A* simulation *on S is a relation $R \subseteq S \times S$ such that if $\langle s,t\rangle \in R$ and $o_S(s)\downarrow$ then $o_S(s) = o_S(t)$ and $\langle \delta_S(s,a), \delta_S(t,a)\rangle \in R_\perp$ for each $a \in A$.*

Let $\preceq$ denote the greatest simulation relation over S: it is a preorder on S and the coinduction principle carries over to simulation. More precisely, the unique homomorphism l from S to the final automaton on $\mathcal{K}\langle A\rangle$ given by Proposition 1 maps $\preceq$ into $\subseteq$, that is, $s \preceq t$ in S iff $l(s) \subseteq l(t)$ in $\mathcal{K}\langle A\rangle$. This holds for arbitrary semiring $\mathcal{K}$ and Moore automaton S on $\mathcal{K}$.

The semantics for recursive terms is usually obtained by taking open terms into account via environments. Altough this would be possible in our case, we prefer to rely on the extended syntax of $\mathcal{P}^{\mathrm{e}}$ and dispense with environments. Informally, we internalize enviroments by regarding $[\![P]\!]_E$, where E is the enviroment mapping $\tilde{x}$ to $\tilde{\sigma}$, as $[\![P[\underline{\tilde{\sigma}}/\tilde{x}]]\!]$. In the sequel, $P[x]$ denotes an open $\mathcal{P}^{\mathrm{e}}$ term where only variable x may occur free, and write $P[Q]$ for $P[Q/x]$, for any Q.

Proposition 4 (fixpoints). *Let $P[x]$ be an open term in $\mathcal{P}^{\mathrm{e}}$, and moreover let $F : \mathcal{K}^{\mathrm{c}}_T\langle A\rangle \to \mathcal{K}^{\mathrm{c}}_T\langle A\rangle$ be defined as $F(\sigma) = [\![P[\underline{\sigma}]]\!]$ for each $\sigma \in \mathcal{K}^{\mathrm{c}}_T\langle A\rangle$. Then F has a least (with respect to $\subseteq$) fixpoint σ_0, and it holds that $\sigma_0 = [\![rec_x.P]\!]$.*

Proof: Let σ_0 denote $[\![rec_x.P]\!]$. It is easy to see that σ_0 is indeed a fixpoint. Using $\underline{\sigma_0} \sim rec_x.P$ (a consequence of $[\![\underline{\sigma}]\!] = \sigma$ for each σ, and of the full abstraction of $[\![\cdot]\!]$), the fact that $P[rec_x.P] \sim P$ and congruence properties of testing equivalence one has: $F(\sigma_0) = [\![P[\underline{\sigma_0}]]\!] = [\![P[rec_x.P]]\!] = [\![rec_x.P]\!] = \sigma_0$.

It is a bit more difficult to show that σ_0 is the least fixpoint. Given a fixpoint σ of F, one first shows that $rec_x.P \preceq \underline{\sigma}$, from which the thesis follows by the coinduction principle extended to simulations, as discussed above. In order to show $rec_x.P \preceq \underline{\sigma}$, one proves that the binary relation over $\mathcal{A}ut$ defined by

$$R \;=\; \{\langle rec_x.P,\, \underline{\sigma}\rangle\} \cup \{\langle \underline{\mathbf{\Omega}},\, \underline{\sigma_a}\rangle \mid a \in A\}$$

is a simulation *up-to bisimulation and up-to context*, meaning that one can match a-derivatives of processess up to $\sim$ and cutting away common contexts (a variation on the technique presented in [11]): this in turn implies that R is included in $\preceq$. The most delicate point in proving that R is a simulation up-to is matching the a-derivatives in the case $rec_x.P \Downarrow a$. Let us write $P^{(k+1)}[x]$ for the unfolding $P[P[\cdots]]$, $k+1$ times, where $k = \max\{|s| \,:\, rec_x.P \xrightarrow{s} \text{ with } s \in \tau^* a \tau^*\}$ (this k exists finite, by $rec_x.P \Downarrow a$ and by König's Lemma). Obviously, $rec_x.P \sim P^{(k+1)}[rec_x.P]$ and $\underline{\sigma} \sim P^{(k+1)}[\underline{\sigma}]$. Moreover, the unfolding $P^{(k+1)}[rec_x.P]$ cannot "have more τ actions than" $rec_x.P$. Since in $P^{(k+1)}[x]$ each x is underneath at least $k+1$ action prefixes (indeed x is guarded in $P[x]$), the a-derivatives of the two unfoldings must be of the form $Q[rec_x.P]$ and $Q[\underline{\sigma}]$, respectively, for one and the same context $Q[\cdot]$; this pair of terms belong to R, up to context. □

We denote by $\mathrm{fix}(F)$ the least fixpoint of any $F : \mathcal{K}^{\mathrm{c}}_T\langle A\rangle \to \mathcal{K}^{\mathrm{c}}_T\langle A\rangle$, whenever this fixpoint exists.

Theorem 1 (compositionality). *The mapping $[\![\cdot]\!] : \mathcal{P}^{\mathrm{e}} \to \mathcal{K}^{\mathrm{c}}_T\langle A\rangle$ is a homomorphism with respect to the operators of* CSP *syntax and the operators on $\mathcal{K}^{\mathrm{c}}_T\langle A\rangle$ defined in Table 2. In particular, the equalities below hold*

$$\begin{aligned}
[\![\underline{\sigma}]\!] &= \sigma\\
[\![\mathsf{nil}]\!] &= \mathbf{1}\\
[\![a.P]\!] &= a.([\![P]\!])\\
[\![P \oplus Q]\!] &= [\![P]\!] \oplus [\![Q]\!]\\
[\![P + Q]\!] &= [\![P]\!] \otimes [\![Q]\!]\\
[\![P[a]]\!] &= [a]([\![P]\!])\\
[\![P \parallel_L Q]\!] &= [\![P]\!] \parallel_L [\![Q]\!]\\
[\![rec_x.P]\!] &= \mathrm{fix}(\lambda\sigma.[\![P[\underline{\sigma}/x]]\!])\,.
\end{aligned}$$

Proof: The last equation follows from Proposition 4. The proof for the remaining equations is a consequence of three facts: first, (a) $\underline{[\![P]\!]} \sim P$, for each $P \in \mathcal{P}^{\mathrm{e}}$; second, (b) the definition of $[\![\cdot]\!]$, and of the operators given in the proof of Proposition 3; and third, (c) the coinduction principle.

An alternative proof, not relying on the definition of the operators, is based on showing that the left-hand side of each equation is bisimilar to its right-hand side, and then the result follows by coinduction (Proposition 1). For each case, one exhibits an appropriate bisimulation relation. As an example, for the case of hiding, it is tedious but straightforward to check that the relation

$$R = \{\langle \bigoplus_{P\in X} [a]([\![P]\!]),\ \bigoplus_{P\in X} [\![P[a]]\!]\rangle \mid X \subseteq_{\mathrm{fin}} \mathcal{P}^{\mathrm{e}}\}$$

is a bisimulation on $\mathcal{K}_T\langle A\rangle$, by relying on the BDE for hiding on consistent series (Table 2), on the definition for o and δ on $\mathcal{A}ut$ given in the previous section, and on the fact that $[\![\cdot]\!]$ acts as a homomorphism. □

Example 1. Let us consider the processes $P = a.b.\mathsf{nil} + a.c.\mathsf{nil}$ and $Q = a.(b.\mathsf{nil} \oplus c.\mathsf{nil})$. It is immediate to see that they are testing equivalent (while they are not bisimilar in the sense of Milner [7]).

Let us turn our attention to the denotations of P and Q in $\mathcal{K}^{\mathrm{c}}_T\langle A\rangle$. Dropping a few parentheses, we have $[\![P]\!] = a.b.\mathbf{1} \otimes a.c.\mathbf{1}$ and $[\![Q]\!] = a.(b.\mathbf{1} \oplus c.\mathbf{1})$. Note now that the following principle can be proven by coinduction: if $\sigma \in \mathcal{K}^{\mathrm{c}}_T\langle A\rangle$ then $\sigma = \langle\sigma(\epsilon)\rangle \oplus \bigotimes_{a\in\mathrm{supp}(\sigma)} a.\sigma_a$ (where $\langle k\rangle$ denotes the FPS with constant k and 0 elsewhere). By applying this principle, and using the BDE's for $\oplus$ and $\otimes$, it is immediate to check that $[\![P]\!] = \langle\{\{a\}\}\rangle \oplus a.(b.\mathbf{1} \oplus c.\mathbf{1}) = [\![Q]\!]$.

Remark 1 (Acceptance trees and FPS*'s).* Among the various proposals for a fully abstract denotational model of must testing, a classical solution is described in [5] in terms of *acceptance trees*. There is a close analogy between acceptance trees and formal power series. Essentially, a finite acceptance tree is a tree with arcs labeled by actions, and nodes labeled by elements of $\mathcal{F}(A)$ or $\perp$.

In [5], the set of finite acceptance trees is turned into an algebraic CPO by an "ideal completion" construction. The resulting domain is used to interpret the operators of a calculus, in particular, the existence of least upper bounds of directed sets and continuity arguments are used to prove existence of least fixpoints when assigning meaning to recursive terms. Within our approach, we need no such order-theoretic constructions, as the existence of a denotational mapping is guaranteed by the automaton structure of FPS's. This happens by no chance, since after Barr it is well-known that final coalgebras can often be characterised as Cauchy completions of the corresponding initial algebras.

We could trace a more specific correspondence between completions of finite acceptance trees and FPS's following [1]. A Moore automaton can be considered as a coalgebra of the functor $S \to \{\perp\} + K \times (A \to S_\perp)$, which is ω-continuous. The initial algebra for the functor is the set of finite, non-empty trees, with leaves labelled by $\perp$, nodes labeled by elements in $\mathcal{F}(A)$, and edges labeled by elements in A. According to [1], the corresponding final coalgebra is then the set of all infinite trees, equipped with a partial order structure induced by the operation of replacing subtrees with $\perp$-leaves. We believe that this partial order reflects the simulation preorder we used in our definition of the fix-point operator, but leave for future work the task of making this correspondence precise.

6 Conclusions and related work

The paper proposes a denotational semantics for testing equivalence, building on recent work by Rutten on the coalgebraic presentation of formal power series. While the coalgebraic characterisation of processes up-to bisimulation is by now well-known, final semantics has seldom been used for capturing different behavioural equivalences. Related to ours, we are aware of Power and Turi's work on trace semantics [8]. The present paper is in the same spirit as theirs, but differently from them we work within the concrete, set-theoretic setting of FPS's, and deal with must testing, which is more challenging than trace equivalence.

A formulation of must testing in terms of bisimulation, at least for strongly convergent processes, has been given by Cleveland and Hennessy [2], but no denotational setting was worked out starting from that. We believe instead that our characterisation of testing equivalence by means of FPS's suggests a methodology – whose core lies in the definition of operators via BDE's and in the choice of an appropriate semiring – for different equivalences and/or process calculi, e.g. with name mobility. At the same time, alternative semantics or language extensions may be suggested by the domain of FPS's, for appropriate choices of the semiring. As an example, with our semiring $\mathcal{K}_T$, the constant series $\mathbf{0}$ really asks for including a *deadlock* operator in the language.

References

1. J. Adámek. Final coalgebras are ideal completions of initial algebras. *Journal of Logic and Computation*, 12:217–242, 2002.
2. R. Cleaveland and M. Hennessy. Testing equivalence as a bisimulation equivalence. *Formal Aspects of Computing*, 5:1–20, 1993.
3. R. De Nicola. Extensional equivalences for transition systems. *Acta Informatica*, 24:211–237, 1987.
4. R. De Nicola and M. Hennessy. Testing equivalences for processes. *Theoret. Comput. Sci.*, 34:83–133, 1984.
5. M. Hennessy. *Algebraic Theory of Processes*. MIT Press, 1988.
6. C.A.R. Hoare. *Communicating Sequential Processes*. Prentice Hall, 1985.
7. R. Milner. *Communication and Concurrency*. Prentice Hall, 1989.
8. J. Power and D. Turi. A coalgebraic foundation for linear time semantics. In M. Hofmann, G. Rosolini, and D. Pavlovic, editors, *Category Theory and Computer Science*, volume 29 of *El. Notes in Theoret. Comput. Sci.*. Elsevier Science, 1999.
9. J.J.M.M. Rutten. Automata, power series, and coinduction: taking input derivatives seriously. Technical Report SEN-R9901, CWI, 1999. Also in J. Wiedermann, P. van Emde Boas and M. Nielsen, editors, *Automata, Languages and Programming*, volume 1644 of *Lect. Notes in Comput. Sci.*, pages 645–654. Springer, 1999.
10. J.J.M.M. Rutten. Universal coalgebra: a theory of systems. *Theoret. Comput. Sci.*, 249:3–80, 2000.
11. D. Sangiorgi. On the bisimulation proof method, *Mathematical Structures in Computer Science*, 8:447–479. Cambridge University Press, 1998.