Under consideration for publication in Math. Struct. in Comp. Science

Final semantics for decorated traces

F. Bonchi,[1] M. Bonsangue, [2,3] G. Caltais, [3,4] J. Rutten, [3,5] A. Silva, [3,5,6] †

[1] *ENS Lyon, Université de Lyon, LIP (UMR 5668 CNRS ENS Lyon UCBL INRIA)*
[2] *LIACS - Leiden University, The Netherlands*
[3] *Centrum voor Wiskunde en Informatica, The Netherlands*
[4] *School of Computer Science - Reykjavik University, Iceland*
[5] *Radboud University Nijmegen, The Netherlands*
[6] *HASLab / INESC TEC, Universidade do Minho, Braga, Portugal*

Received 30 October 2012

In concurrency theory, various semantic equivalences on labelled transition systems are based on traces enriched or *decorated* with some additional observations, generally referred to as *decorated traces*. Using the generalized powerset construction, recently introduced by a subset of the authors (Silva, Bonchi, Bonsangue & Rutten 2010), we give a coalgebraic presentation of decorated trace semantics. This yields a uniform notion of minimal representatives for the various decorated trace equivalences, in terms of final Moore automata. As a consequence, proofs of decorated trace equivalence can be given by coinduction, using different types of (Moore-) bisimulation (up-to), which is helpful for automation. The coalgebraic framework introduced in this paper handles ready, failure, (complete) trace, possible-futures, ready trace and failure trace semantics.

1. Introduction

The study of systems equivalence has been an interesting research topic for many years now. Several equivalences have been proposed throughout the years, each of which suitable for use in different contexts of application. Many of the equivalences that are important in the theory of concurrency were described in the well-known paper by van Glabbeek (van Glabbeek 2001).

Proof methods for the different equivalences are an important part of this research enterprise. In this paper, we propose *coinduction* as a general proof method for what van Glabbeek calls *decorated trace semantics*, which includes ready, failure, (complete) trace, possible-futures, ready trace and failure trace semantics.

Coinduction is a general proof principle which has been uniformly defined in the theory

† The work of Georgiana Caltais has been partially supported by a CWI Internship and by the project 'Meta-theory of Algebraic Process Theories' (nr. 100014021) of the Icelandic Research Fund. The work of Alexandra Silva is partially funded by the ERDF through the Programme COMPETE and by the Portuguese Government through FCT - Foundation for Science and Technology, project ref. `PTDC/EIA-CCO/122240/2010` and `SFRH/BPD/71956/2010`. We thank Luca Aceto and Anna Ingólfsdóttir for comments and references to the literature.

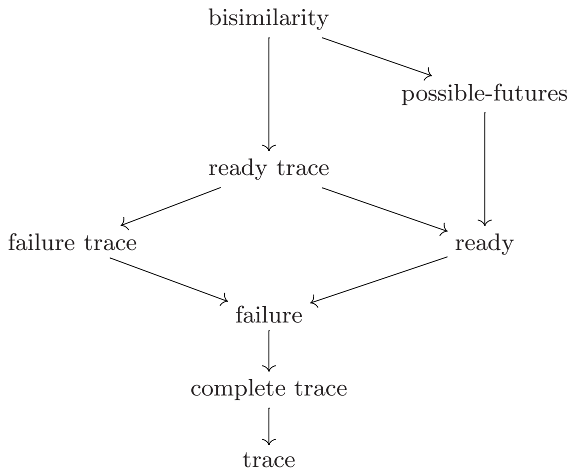


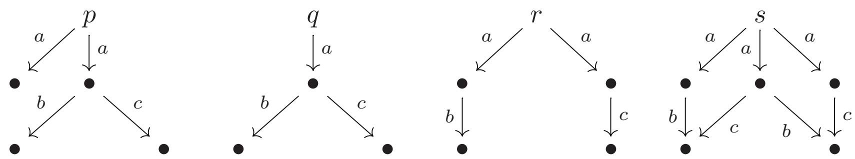
Fig. 1. Lattice of semantic equivalences.

of coalgebras for different types of state-based systems and infinite data types. Given a functor $\mathcal{F}\colon \mathbf{Set} \to \mathbf{Set}$, an $\mathcal{F}$-coalgebra is a pair (X, f) consisting of a set of states X and a function $f\colon X \to \mathcal{F}(X)$ defining the dynamics of the system. The functor $\mathcal{F}$ determines the type of the transition system or data type under study. For a large class of functors $\mathcal{F}$, there exists a *final coalgebra* into which every $\mathcal{F}$-coalgebra is mapped by a unique homomorphism. Intuitively, one can see the final coalgebra as the universe of all behaviours of systems and the unique morphism as the map assigning to each system its behaviour. This provides a standard notion of equivalence called *$\mathcal{F}$-behavioural equivalence*. Moreover, these canonical behaviours are minimal, by general coalgebraic considerations (Rutten 2000), in that no two different states are equivalent.

Labelled transition systems (LTS's) can be modelled as coalgebras for the functor $\mathcal{F}(X) = (\mathcal{P}_\omega X)^A$ and the canonical behavioural equivalence associated with $\mathcal{F}$ is precisely the finest equivalence of the spectrum in (van Glabbeek 2001). In the recent past, other equivalences of the spectrum have been also cast in the coalgebraic framework. Notably, trace semantics was widely studied (Hasuo, Jacobs & Sokolova 2007, Silva et al. 2010) and, more recently, decorated trace semantics was recovered via a coalgebraic generalization of the classical powerset construction (Silva, Bonchi, Bonsangue & Rutten 2011).

In Fig. 1 we illustrate the hierarchy (based on the coarseness level) among bisimilarity, ready, failure, (complete) trace, possible-futures, ready trace and failure trace semantics, as introduced in (van Glabbeek 2001). For example, bisimilarity (the standard behavioural equivalence on $\mathcal{F}$-coalgebras) is the finest of the aforementioned semantics, whereas trace is the coarsest one.

To get some intuition on the type of distinctions the equivalences above encompass, consider the following labelled transition systems over the alphabet $A = \{a, b, c\}$:



The semantic equivalences in Fig. 1 distinguish between p, q, r and s as summarized in the table below:

	p, q	p, r	p, s	q, r	q, s	r, s
bisimilarity	×	×	×	×	×	×
trace	✓	✓	✓	✓	✓	✓
complete trace	×	×	×	✓	✓	✓
ready	×	×	×	×	×	×
failure	×	×	×	×	×	✓
possible-futures	×	×	×	×	×	×
ready trace	×	×	×	×	×	×
failure trace	×	×	×	×	×	✓

where ✓ to stands for an "yes" answer w.r.t. the behavioural equivalence of two of the systems p, q, r and s, whereas × represents a "no" answer. More explicitly:

None of the systems above are bisimilar. Note that after executing action a from p a deadlock state can be reached. Obviously, q and r have a different branching structure, so they are not bisimilar. System s subsumes the (non-bisimilar) behaviours of both q and r.

The traces of the states p, q, r and s are $\{a, ab, ac\}$, and therefore they are all trace equivalent. Complete trace semantics identifies states that have the same set of complete traces, that is, traces that lead to states where no further action are possible. Of the four states above, q and r and s are complete trace equivalent, whereas p is the only state that has a as a complete trace.

Ready semantics identifies states according to the set of actions they can trigger immediately after a certain trace has been executed. None of the states above are ready equivalent. Observe that after the execution of action a: process p can reach a deadlock state, q has always to choose between actions b and c, process r can only do b or c, whereas s subsumes the (not equivalent) behaviours of both q and r.

Failure semantics takes into account the set of actions that cannot be fired immediately after the execution of a certain trace. Only r and s are failure equivalent. Note that after triggering action a, process p can fail executing $\{a, b, c\}$. Moreover, after firing a, process q only fails executing $\{a\}$.

Possible-futures semantics identifies states that can perform the same traces w and, moreover, the states reached by executing such w's are trace equivalent. None of the states above are possible-futures equivalent. After triggering action a: p can reach a deadlock state (with no further behaviour), q can exfecute the set of traces $\{b, c\}$, state r can trigger either trace b or c, whereas s inherits the behaviours of both q and r (which are not possible-futures equvalent).

Ready (respectively failure) trace semantics identifies states that can trigger the same traces w and the (pairwise-taken) intermediate states determined by such w's are ready (respectively refuse) to trigger the same sets of actions. None of the systems above is ready trace equivalent. p is the only one that after triggering a can reach a deadlock state. After preforming action a: process q reaches a state that is ready to trigger both b and c, whereas r cannot. Moreover, s subsumes the behaviour of both q and r (which are not ready trace equivalent). The analysis on failure trace equivalence follows a similar reasoning.

This paper is an extended version of the conference paper (Bonchi, Bonsangue, Caltais, Rutten & Silva 2012) where we a) proved that the coalgebraic ready, failure and (complete) trace semantics are equivalent to the corresponding set-theoretic notions from (van Glabbeek 2001), b) showed how the coalgebraic semantics lead to canonical representatives for the aforementioned decorated traces, and c) showed how to prove decorated trace equivalence using coinduction, by constructing bisimulations (up-to context) that witness the desired equivalence. The latter is interesting also from the point of view of tool development: construction of bisimulations is known to be particularly suitable for automation. Moreover, the up-to context technique also increases the efficiency of reasoning, as verifications are performed under certain closure properties, which means the bisimulations that are built are smaller (see Section 3, and Section 4 for examples). The techniques we used for up-to reasoning are an extension of the recent work in (Bonchi & Pous 2013).

In this paper we extend a), b) and c) above also for the case of possible-futures, ready trace and failure trace semantics. Moreover, we have included more details, proofs and examples on how to use the coalgebraic framework (summarized in Fig. 16) for reasoning on decorated trace equivalences.

The paper is organized as follows. In Section 2, we provide the basic notions from coalgebra and recall the generalized powerset construction. In Section 3, we show how the powerset construction can be applied for determinizing LTS's in terms of Moore automata $(X, f \colon X \to B \times X^A)$, in order to coalgebraically characterize decorated trace semantics. Detailed descriptions of coalgebraic decorated trace semantics are provided in Section 4. Here we also prove that the obtained coalgebraic models are equivalent to the original definitions, and illustrate how one can reason about decorated trace equivalence by constructing bisimulations up-to context. Section 5 discusses that the canonical representatives of LTS's we obtain coalgebraically coincide with the minimal LTS's one would obtain by identifying all states equivalent w.r.t. a particular decorated trace semantics. Section 6 contains concluding remarks and discusses future work.

2. Preliminaries

In this section, we briefly recall basic notions from coalgebra and the generalized powerset construction (Silva et al. 2010). We first introduce some notation on sets.

We denote sets by capital letters $X, Y, \dots$ and functions by lower case letters $f, g, \dots$. The *cartesian product* of two sets X and Y is denoted by $X \times Y$, and has the projection maps $X \xleftarrow{\pi_1} X \times Y \xrightarrow{\pi_2} Y$. By X^Y we represent the family of *functions* $f\colon Y \to X$, whereas the collection of *finite subsets* of X is denoted by $\mathcal{P}_\omega X$. For each of these operations defined on sets, there is an analogous one on functions (for details see for example (Awodey 2010)). This turns the operations above into (bi)functors, which we shall use throughout this paper.

For an alphabet A, we denote by A^* the set of all *words* over A and by ε the *empty word*. The *concatenation* of words $w_1, w_2 \in A^*$ is written $w_1 w_2$.

Coalgebras: We consider coalgebras of functors $\mathcal{F}$ defined on **Set** – the category of sets and functions. An $\mathcal{F}$-*coalgebra* (or *coalgebra*, when $\mathcal{F}$ is understood) is a pair $(X, c\colon X \to \mathcal{F}X)$, where $X \in \mathbf{Set}$. We call X the state space, and we say that $\mathcal{F}$ together with c determine the dynamics, or the transition structure of the $\mathcal{F}$-coalgebra.

An $\mathcal{F}$-*homomorphism* between two $\mathcal{F}$-coalgebras (X, f) and (Y, g), is a function $h\colon X \to Y$ preserving the transition structure, *i.e.*, $g \circ h = \mathcal{F}(h) \circ f$.

An $\mathcal{F}$-coalgebra (Ω, ω) is *final* if for any $\mathcal{F}$-coalgebra (X, f) there exists a unique $\mathcal{F}$-homomorphism $[\![-]\!]_X\colon X \to \Omega$. A final coalgebra represents the universe of all possible *behaviours* of $\mathcal{F}$-coalgebras. The unique morphism $[\![-]\!]_X\colon X \to \Omega$ maps each state in X to its behaviour. Using this mapping, behavioural equivalence can be defined as follows: for any two coalgebras (X, f) and (Y, g), the states $x \in X$ and $y \in Y$ are *behaviourally equivalent*, written $x \sim_{\mathcal{F}} y$, if and only if they have the same behaviour, that is

$$x \sim_{\mathcal{F}} y \text{ iff } [\![x]\!]_X = [\![y]\!]_Y. \tag{1}$$

We think of $[\![x]\!]_X$ as the *canonical representative* of the behaviour of x. The image of X under $[\![-]\!]_X$ can be viewed as the minimization of (X, f), since the final coalgebra contains no pairs of equivalent states.

For an example we consider deterministic automata (DA). A deterministic automaton over the input alphabet A is a pair $(X, \langle o, t\rangle)$, where X is a set of states and $\langle o, t\rangle\colon X \to 2 \times X^A$ is a function with two components: o, the output function, determines if a state x is final ($o(x) = 1$) or not ($o(x) = 0$); and t, the transition function, returns for each input letter a the next state. DA's are coalgebras for the functor $\mathcal{D}(X) = 2 \times X^A$. The final coalgebra of this functor is $(2^{A^*}, \langle \epsilon, (-)_a\rangle)$ where 2^{A^*} is the set of languages over A and $\langle \epsilon, (-)_a\rangle$, given a language L, determines whether or not the empty word ε is in the language ($\epsilon(L) = 1$ or $\epsilon(L) = 0$, resp.) and, for each input letter a, returns the *derivative* of L: $L_a = \{w \in A^* \mid aw \in L\}$. From any DA, there is a unique map $[\![-]\!]$ into 2^{A^*} which assigns to each state its behaviour (that is, the language that the state recognizes).

$$\begin{tikzcd} X \arrow[r, dashed, "\llbracket - \rrbracket_X"] \arrow[d, "\langle o,t\rangle"'] & 2^{A^*} \arrow[d, "\langle \epsilon,(-)_a\rangle"] \\ 2 \times X^A \arrow[r, dashed, "id\times \llbracket - \rrbracket_X^A"'] & 2 \times (2^{A^*})^A \end{tikzcd} \qquad \begin{aligned} \llbracket x \rrbracket_X(\varepsilon) &= o(x) \\ \llbracket x \rrbracket_X(aw) &= \llbracket t(x)(a) \rrbracket_X(w) \end{aligned}$$

Therefore, behavioural equivalence for the functor $\mathcal{D}$ coincides with the classical language equivalence of automata.

Another example (fundamental for the rest of the paper) is given by Moore automata. Moore automata with inputs in A and outputs in B are coalgebras for the functor $\mathcal{M}(X) = B \times X^A$, that is pairs $(X, \langle o,t \rangle)$ where X is a set, $t \colon X \to X^A$ is the transition function (like for DA) and $o \colon X \to B$ is the output function which maps every state in its output. Thus DA can be seen as a special case of Moore automata where $B = 2$. The final coalgebra for $\mathcal{M}$ is $(B^{A^*}, \langle \epsilon, (-)_a \rangle)$ where B^{A^*} is the set of all functions $\varphi \colon A^* \to B$, $\epsilon \colon B^{A^*} \to B$ maps each φ into $\varphi(\epsilon)$ and $(-)_a \colon B^{A^*} \to (B^{A^*})^A$ is defined for all $\varphi \in B^{A^*}$, $a \in A$ and $w \in A^*$ as $(\varphi)_a(w) = \varphi(aw)$.

$$\begin{tikzcd} X \arrow[r, dashed, "\llbracket - \rrbracket_X"] \arrow[d, "\langle o,t\rangle"'] & B^{A^*} \arrow[d, "\langle \epsilon,(-)_a\rangle"] \\ B \times X^A \arrow[r, dashed, "id\times \llbracket - \rrbracket_X^A"'] & B \times (B^{A^*})^A \end{tikzcd} \qquad \begin{aligned} \llbracket x \rrbracket_X(\varepsilon) &= o(x) \\ \llbracket x \rrbracket_X(aw) &= \llbracket t(x)(a) \rrbracket_X(w) \end{aligned}$$

Bisimulations: Coalgebras provide a useful technique for proving behavioural equivalence, namely, bisimulation. Let (X, f) and (Y, g) be two $\mathcal{F}$-coalgebras. A relation $R \subseteq X \times Y$ is a *bisimulation* if there exists a function $\alpha_R \colon R \to \mathcal{F}R$ such that $\pi_1 \colon R \to X$ and $\pi_2 \colon R \to Y$ are coalgebra homomorphisms. In (Rutten 2000), it is shown that under certain conditions on $\mathcal{F}$ (which are met by all the functors considered in this paper), bisimulations are a sound and complete proof technique for behavioural equivalence, namely,

$$x \sim_{\mathcal{F}} y \text{ iff there exists a bisimulation } R \text{ such that } xRy. \tag{2}$$

The generalized powerset construction: As shown above, every functor $\mathcal{F}$ induces both a notion of $\mathcal{F}$-coalgebra and a notion of behavioural equivalence $\sim_{\mathcal{F}}$. Sometimes, it is interesting to consider different equivalences than $\sim_{\mathcal{F}}$ for reasoning about $\mathcal{F}$-coalgebras. This is the case of labeled transition systems which are coalgebras for the functor $\mathcal{L}(X) = (\mathcal{P}_\omega X)^A$. The induced behavioural equivalence $\sim_{\mathcal{L}}$ coincides with the standard notion of bisimilarity by Milner and Park (Park 1981, Milner 1989). However, in concurrency theory, many other equivalences have been studied, notably, *decorated trace equivalences* (van Glabbeek 2001). Another example is given by non-deterministic automata which are coalgebras for the functor $\mathcal{N}(X) = 2 \times (\mathcal{P}_\omega X)^A$. The associated equivalence $\sim_{\mathcal{N}}$ strictly implies language equivalence, which is often taken as an intended semantics.

For this reason, a subset of the authors has introduced in (Silva et al. 2011) the *generalized powerset construction*, for coalgebras $f \colon X \to \mathcal{F}T(X)$ for a functor $\mathcal{F}$ and a monad T, with the proviso that that $\mathcal{F}T(X)$ is an algebra for the monad T. In (Silva et al. 2011), all the technical details are explored and many interesting instances of the construction are shown. In this paper, we will only be interested in the case where $T = \mathcal{P}_\omega$ and $\mathcal{M}(X) = B \times X^A$, for A an action alphabet and B a semilattice, and we will therefore only explain the concrete picture for the functor and monad of interest. The fact that we take B to be a semilattice is enough to guarantee that $\mathcal{M}T(X) = B \times (\mathcal{P}_\omega X)^A$ is a semilattice to. This fulfills then the proviso above, since semilattices are precisely the algebras of the monad $\mathcal{P}_\omega$.

Given a coalgebra $f : X \to \mathcal{M}\mathcal{P}_\omega X$, and because $\mathcal{M}$ has a final coalgebra, we can extend it uniquely to $f^\sharp : \mathcal{P}_\omega X \to \mathcal{M}\mathcal{P}_\omega X$ and consider the unique coalgebra homomorphism into the final coalgebra, as summarized by the following diagram:

$$\begin{tikzcd} X \arrow[r, "\{\cdot\}"] \arrow[d, "f"'] & \mathcal{P}_\omega X \arrow[r, dashed, "[\![-]\!]"] \arrow[dl, "f^\sharp"] & B^{A^*} \arrow[d, "\langle \epsilon, (-)_a \rangle"] \\ B \times (\mathcal{P}_\omega X)^A \arrow[rr, dashed, "id_B \times [\![-]\!]^A"'] & & B \times (B^{A^*})^A \end{tikzcd} \tag{3}$$

With this construction, one can coalgebraically characterize language equivalence for Moore automata and, in particular, for non-deterministic automata. Take $T = \mathcal{P}_\omega$ and $\mathcal{F} = \mathcal{D}$, which is an instance of $\mathcal{M}$ for $B = 2$, the two-element semilattice. An $\mathcal{M}T$-coalgebra is a pair (X, f) with $f : X \to 2 \times (\mathcal{P}_\omega X)^A$, *i.e.*, an NDA. Therefore every NDA (X, f) is transformed into $(\mathcal{P}_\omega X, f^\sharp)$ which is a DA. This corresponds to the classical powerset construction for determinizing non-deterministic automata. The language recognized by a state x can be defined by precomposing the unique morphism $[\![-]\!] : \mathcal{P}_\omega X \to 2^{A^*}$ with the unit of $\mathcal{P}_\omega$, which is the function $\{-\} : X \to \mathcal{P}_\omega X$ mapping each $x \in X$ into the singleton set $\{x\} \in \mathcal{P}_\omega X$.

3. Decorated trace semantics via determinization

Our aim is to reason about decorated trace equivalences of labelled transition systems. In this section, we use the generalized powerset construction and show how one can determinize arbitrary labelled transition systems obtaining particular instances of Moore automata (with different output sets) in order to model ready, failure, (complete) trace, possible-futures, ready trace and failure trace equivalences. This paves the way to building a general framework for reasoning on decorated trace equivalences in a uniform fashion, in terms of bisimulations up-to context.

A *labeled transition system* is a pair (X, δ) where X is a set of states and $\delta : X \to (\mathcal{P}_\omega X)^A$ is a function assigning to each state $x \in X$ and to each label $a \in A$ a finite set of possible successors states. We write $x \xrightarrow{a} y$ whenever $y \in \delta(x)(a)$. We extend the notion of transition to words $w = a_1 \ldots a_n \in A^*$ as follows: $x \xrightarrow{w} y$ if and only if $x \xrightarrow{a_1} \ldots \xrightarrow{a_n} y$. For $w = \varepsilon$, we have $x \xrightarrow{\varepsilon} y$ if and only if $y = x$.

We now define in a nutshell the equivalences we will be dealing with in this paper. (See (van Glabbeek 2001) for more details on the corresponding classical definitions.) For a function $\varphi \in (\mathcal{P}_\omega X)^A$, $I(\varphi)$ denotes the set of all labels "enabled" by φ, given by $I(\varphi) = \{a \in A \mid \varphi(a) \neq \emptyset\}$, while $Fail(\varphi)$ denotes the set $\{Z \subseteq A \mid Z \cap I(\varphi) = \emptyset\}$.

Let (X, δ) be a LTS and $x \in X$ be a state. A *failure pair* of x is a pair $(w, Z) \in A^* \times \mathcal{P}_\omega A$

such that $x \xrightarrow{w} y$ and $Z \in Fail(\delta(y))$. A *ready pair* of x is a pair $(w, Z) \in A^* \times \mathcal{P}_\omega A$ such that $x \xrightarrow{w} y$ and $Z = I(\delta(y))$.

A *trace* of x is a word $w \in A^*$ such that $x \xrightarrow{w} y$ for some y. A trace w of x is *complete* if $x \xrightarrow{w} y$ and y stops, i.e., $I(\delta(y)) = \emptyset$.

A pair $\langle w, T\rangle \in A^* \times \mathcal{P}(A^*)$ is a *possible future* of $x \in X$ if there is $y \in X$ such that $x \xrightarrow{w} y$ and T is the set of all traces of y.

We call a *ready trace* of a state $x_0 \in X$ a sequence $I_0 a_1 I_1 a_2 \ldots a_n I_n \in \mathcal{P}_\omega(A) \times (A \times \mathcal{P}_\omega(A))^*$, if there are $x_1, \ldots, x_n \in X$ such that $x_0 \xrightarrow{a_1} x_1 \xrightarrow{a_2} \ldots \xrightarrow{a_n} x_n$ and $I_i = I(\delta(x_i))$, for $i = 1, \ldots, n$. Orthogonally, a sequence $F_0 a_1 F_1 a_2 \ldots a_n F_n$ is called a *failure trace* of x_0 if $F_i \in Fail(\delta(x_i))$.

We use $\mathcal{T}(x)$, $\mathcal{CT}(x)$, $\mathcal{F}(x)$, $\mathcal{R}(x)$, $\mathcal{PF}(x)$, $\mathcal{RT}(x)$, $\mathcal{FT}(x)$ to denote, respectively, the set of all traces, complete traces, failure pairs, ready pairs, possible futures, ready traces and failure traces of x.

For $\mathcal{I}$ ranging over $\mathcal{T}, \mathcal{CT}, \mathcal{F}, \mathcal{R}, \mathcal{PF}, \mathcal{RT}$ and $\mathcal{FT}$, two states x and y are $\mathcal{I}$-equivalent iff $\mathcal{I}(x) = \mathcal{I}(y)$ (van Glabbeek 2001).

Intuitively, these equivalences can be described as follows:

— *ready semantics* identifies states of LTS's according to the set Z of actions they can trigger immediately after a certain action sequence w has been "consumed"; we call a pair (w, Z) a *ready pair*,

— *failure semantics* takes into account the set Z of actions that cannot be fired immediately after the execution of sequences w; we call a pair (w, Z) a *failure pair*,

— *trace semantics* identifies system states if and only if they can execute the same sets of action sequences w,

— *complete trace semantics* identifies system states that perform the same sets of "complete" traces w; we call an action sequence w a *complete trace* of a state p if and only if $p \xrightarrow{w} q$ and q cannot execute any further action.
Note that the difference between trace and complete trace semantics consists in the fact that trace semantics does not detect stagnation, whereas the latter semantics takes into consideration deadlock states.

— *possible-futures semantics* distinguishes between states that either cannot execute the same traces or, that by triggering the same sequences of actions can reach states that are not trace equivalent,

— *ready trace semantics* identifies states that can execute the same sets of sequences $w = a_1 \ldots a_n$ and, moreover, the corresponding intermediate states reached by performing action a_i (for $i = 1, \ldots,$ n) are ready to trigger the same actions,

— *failure trace semantics* apply similarly to ready trace semantics, with the difference that the associated intermediate states determined by sequences $w \in A^*$ refuse executing the same sets of actions.

The coalgebraic characterization of ready, failure and (complete) trace was obtained in (Silva et al. 2011) in the following way. Given an arbitrary LTS $(X, \delta\colon X \to (\mathcal{P}_\omega X)^A)$, we associate a *decorated* LTS represented by a coalgebra of the functor $\mathcal{F}_{\mathcal{I}}(X) = B_{\mathcal{I}} \times (\mathcal{P}_\omega X)^A$, namely $(X, \langle \overline{o}_{\mathcal{I}}, id\rangle \circ \delta \colon X \to B_{\mathcal{I}} \times (\mathcal{P}_\omega X)^A)$, where the output operation $\overline{o}_{\mathcal{I}} \colon (\mathcal{P}_\omega X)^A \to B_{\mathcal{I}}$ provides the observations of interest corresponding to the original LTS and depending on the equivalence we want to study. (At this point, $B_{\mathcal{I}}$ represents an arbitrary semilattice with a $\vee$ operation, instantiated for each of the semantics under consideration as in (Silva et al. 2011).) Then, we determinize the decorated LTS, as depicted in Figure 2.

$$
\begin{tikzcd}
X \arrow[r, "\{-\}"] \arrow[d, "\delta"'] & \mathcal{P}_\omega X \arrow[r, dashed, "\llbracket - \rrbracket"] \arrow[ddl, "\langle o,t\rangle"] & (B_{\mathcal{I}})^{A^*} \arrow[dd, "\langle \epsilon,(-)_a\rangle"] \\
(\mathcal{P}_\omega X)^A \arrow[d, "\langle \overline{o}_{\mathcal{I}},id\rangle"'] & & \\
\mathcal{F}_{\mathcal{I}}X = B_{\mathcal{I}} \times (\mathcal{P}_\omega X)^A \arrow[rr, dashed, "id_{B_{\mathcal{I}}} \times \llbracket - \rrbracket^A"'] & & B_{\mathcal{I}} \times ((B_{\mathcal{I}})^{A^*})^A
\end{tikzcd}
$$

$$
\begin{aligned}
&o(Y) = \textstyle\bigvee_{y \in Y} \overline{o}_{\mathcal{I}}(\delta(y)) \\
&t(Y)(a) = \textstyle\bigcup_{y \in Y} \delta(y)(a) \\
&\llbracket Y \rrbracket(\varepsilon) = o(Y) \\
&\llbracket Y \rrbracket(aw) = \llbracket \bigcup_{y \in Y} \delta(y)(a) \rrbracket(w)
\end{aligned}
$$

Fig. 2. The powerset construction for decorated LTS's.

Note that both the output operation and its image are parameterized by $\mathcal{I} \in \{\mathcal{R}, \mathcal{F}, \mathcal{T}, \mathcal{CT}\}$, depending on the type of decorated trace semantics under consideration.

The coalgebraic modelling of possible-futures semantics could easily be recovered by following a similar approach. However, note that for the case of ready and failure trace semantics a "preprocessing" procedure on the initial LTS is required before the determinization. This consists in enriching the action alphabet A with additional information represented by sets of actions ready (respectively actions refused) to be triggered as a first step. Consequently, each LTS $(X, \delta \colon X \to (\mathcal{P}_\omega X)^A)$ is uniquely associated a coalgebra $(X, \overline{\delta} \colon X \to (\mathcal{P}_\omega X)^{\bar{A}})$, defined in a natural fashion, as we shall see later on. The construction in Fig. 2 is eventually applied on $(X, \overline{\delta})$.

The explicit instantiations of $\overline{o}_{\mathcal{I}}$ and $B_{\mathcal{I}}$ are provided in Section 4, where we will also show that the coalgebraic modellings in fact coincide with the original definitions of the corresponding equivalences. A fact that was not formally shown in (Silva et al. 2011), for none of the aforementioned semantics.

Our coalgebraic modelling of decorated trace semantics enables the definition of the corresponding equivalences as Moore bisimulations (Rutten 2000) (*i.e.*, bisimulations for a functor $\mathcal{M} = B_{\mathcal{I}} \times X^A$). This way, checking behavioural equivalence of x_1 and x_2 reduces to checking the equality of their unique representatives in the final coalgebra: $\llbracket \{x_1\} \rrbracket$ and $\llbracket \{x_2\} \rrbracket$.

Moreover, it is worth observing that when reasoning on behavioural equivalence it is preferable to use relations as small as possible, that are not necessarily bisimulations, but contained in a bisimulation relation. These relations are referred to as *bisimulations up-to* (Sangiorgi & Rutten 2011).

In what follows we exploit the generalized powerset construction summarized in Fig. 2 and define bisimulation up-to context in the setting of decorated LTS's determinized in terms of Moore automata.

Let $L_{dec} = (X, \langle \overline{o}_{\mathcal{I}}, id \rangle \circ \delta \colon X \to B_{\mathcal{I}} \times (\mathcal{P}_\omega X)^A)$ be a decorated (possibly "preprocessed") LTS and $(\mathcal{P}_\omega X, \langle o, t \rangle \colon \mathcal{P}_\omega X \to B_{\mathcal{I}} \times (\mathcal{P}_\omega X)^A)$ its associated Moore automaton, as in Fig. 2. A *bisimulation up-to context* for L_{dec} is a relation $R \subseteq (\mathcal{P}_\omega X) \times (\mathcal{P}_\omega X)$ such that:

$$X_1 \; R \; X_2 \Rightarrow \begin{cases} o(X_1) = o(X_2) \\ (\forall a \in A) \,.\, t(X_1)(a) \; c(R) \; t(X_2)(a) \end{cases} \tag{4}$$

where $c(R)$ is the smallest relation which is closed with respect to set union and which includes R, inductively defined by the following inference rules:

$$\frac{}{\emptyset \; c(R) \; \emptyset} \qquad \frac{X \, R \, Y}{X \, c(R) \, Y} \qquad \frac{X_1 \, c(R) \, Y_1 \quad X_2 \, c(R) \, Y_2}{X_1 \cup X_2 \; c(R) \; Y_1 \cup Y_2} \tag{5}$$

Remark 3.1. Observe that by replacing $c(R)$ with R in (4) one gets the definition of *Moore bisimulation*.

Theorem 3.1. Any bisimulation up-to context for decorated LTS's is included in a bisimulation relation.

Proof. The proof consists in showing that for any bisimulation up-to context R, $c(R)$ is a bisimulation relation (recall that $R \subseteq c(R)$). The result follows by structural induction, as shown below.

Let $L_{dec} = (X, \delta^\sharp \colon X \to B_{\mathcal{I}} \times (\mathcal{P}_\omega X)^A)$ be a decorated LTS and $(\mathcal{P}_\omega X, \langle o, t \rangle \colon \mathcal{P}_\omega X \to B_{\mathcal{I}} \times (\mathcal{P}_\omega X)^A)$ be its associated Moore automaton, derived according to the powerset construction.

Let R be a bisimulation up-to context for L_{dec}.

In what follows we want to prove that $c(R)$ is a bisimulation relation (that includes R, by (5)).

We have to show that

$$X \, c(R) \; Y \Rightarrow \begin{cases} o(X) = o(Y) \\ (\forall a \in A) \,.\, t(X)(a) \; c(R) \; t(Y)(a) \end{cases} \tag{6}$$

We proceed by structural induction.

1 Let $X \, R \, Y$. Then (6) holds by definition.

2 Let $X = X_1 \cup X_2$ and $Y = Y_1 \cup Y_2$ such that $X_1 \, c(R) \; Y_1$ and $X_2 \, c(R) \; Y_2$. By induction, we have that $o(X_1) = o(Y_1)$ and $o(X_2) = o(Y_2)$. We now need to prove that $o(X) = o(Y)$.

$$o(X) = o(X_1 \cup X_2) = o(X_1) \cup o(X_2) \overset{IH}{=} o(Y_1) \cup o(Y_2) = o(Y_1 \cup Y_2) = o(Y)$$

We also have, by induction, that

$$(\forall a \in A) \,.\, t(X_1)(a)\ c(R)\ t(Y_1)(a) \text{ and } (\forall a \in A) \,.\, t(X_2)(a)\ c(R)\ t(Y_2)(a)$$

Hence, for all $a \in A$, we can easily prove that $t(X)(a)\ c(R)\ t(Y)(a)$:

$$\begin{array}{rclcl} t(X)(a) & = & t(X_1 \cup X_2)(a) & = & t(X_1)(a) \cup t(X_2)(a) \quad (IH) \\ & & & c(R) & t(Y_1)(a) \cup t(Y_2)(a) \\ & & & = & t(Y_1 \cup Y_2)(a) = t(Y)(a) \end{array}$$

At this point it holds that $c(R) \supseteq R$ is a bisimulation relation, as (6) holds for all $(X, Y) \in c(R)$.

□

Remark 3.2. Based on (1), (2) and Theorem 3.1, verifying behavioural equivalence of two states x_1, x_2 in a decorated LTS consists in identifying a bisimulation up-to context R^c relating $\{x_1\}$ and $\{x_2\}$:

$$[\![\{x_1\}]\!] = [\![\{x_2\}]\!] \text{ iff } \{x_1\}\, R^c\, \{x_2\}. \tag{7}$$

Also note that Theorem 3.1 is not a very different, but useful generalization of Theorem 2 in (Bonchi & Pous 2013) to the context of decorated LTS's.

More insight on how to derive canonical representatives of decorated trace semantics and how to apply the bisimulation up-to context proof technique is provided in Section 4.

4. Coalgebraic modelling of decorated trace semantics

In what follows we a) provide the details on the coalgebraic modelling of ready, failure, (complete) trace, possible-futures, ready trace and failure trace semantics, b) show that the corresponding representations coincide with their original definitions in (van Glabbeek 2001) and c) show, by means of examples, how the associated coalgebraic frameworks can be used in order to reason on the aforementioned equivalences in terms of Moore bisimulations (up-to).

The approaches in the subsequent sections mainly follow the same steps. For each of the decorated trace semantics we proceed by first instantiating the ingredients of Fig. 2 in Section 3 (that summarizes the generalized powerset construction in (van Glabbeek 2001)). Note that for the case of ready trace and failure trace semantics an additional "preprocessing" step is required. This eventually generates equivalent LTS's having transition enriched with additional information used for obtaining the "right" derived Moore automata (more details on the "preprocessing" procedure are provided in Section 4.6 and Section 4.7).

For $\mathcal{I}$ ranging over $\mathcal{T}, \mathcal{CT}, \mathcal{F}, \mathcal{R}, \mathcal{PF}, \mathcal{RT}$ and $\mathcal{FT}$, showing that the corresponding coalgebraic modelling and the set-theoretic definitions in (van Glabbeek 2001) are equivalent reduces to proving that, given an arbitrary state x of an LTS, $\mathcal{I}(x)$ is in one-to-one correspondence with the behaviour $[\![\{x\}]\!]$ in the final Moore coalgebra.

For a more concrete insight, we provide in each of the subsequent sections examples of (possibly not) $\mathcal{I}$-equivalent systems, and show how the coalgebraic machinery is used for reasoning on $\mathcal{I}$-equivalence.

4.1. *Ready semantics*

In this section we show how the ingredients of Fig. 2 in Section 3 can be instantiated in order to provide a coalgebraic modelling of ready semantics, as introduced in (Silva et al. 2011). Moreover, we prove that the resulting coalgebraic characterization of this semantics is equivalent to the original definition.

Consider an LTS $(X, \delta\colon X \to (\mathcal{P}_\omega X)^A)$ and recall that, for a function $\varphi\colon A \to \mathcal{P}_\omega X$, the set of *actions enabled by* φ is given by

$$I(\varphi) = \{a \in A \mid \varphi(a) \neq \emptyset\}. \tag{8}$$

For the particular case $\varphi = \delta(x)$, $I(\delta(x))$ denotes the set of all (initial) actions ready to be fired by $x \in X$.

Recall also that a *ready pair* of x is a pair $(w, Z) \in A^* \times \mathcal{P}_\omega A$ such that $x \xrightarrow{w} y$ and $Z = I(\delta(y))$. We denote by $\mathcal{R}(x)$ the set of *all ready pairs* of x.

Intuitively, ready semantics identifies states in X based on the actions $a \in A$ they can immediately trigger after performing a certain action sequence $w \in A^*$, *i.e.*, based on their ready pairs. It was originally defined as follows:

Definition 4.1 ($\mathcal{R}$-equivalence (van Glabbeek 2001)). Let $(X, \delta\colon X \to (\mathcal{P}_\omega X)^A)$ be an LTS and $x, y \in X$ two states. States x and y are *ready equivalent* ($\mathcal{R}$-equivalent) if and only if they have the same set of ready pairs, that is $\mathcal{R}(x) = \mathcal{R}(y)$.

Next, we instantiate $\overline{o}_{\mathcal{I}}$ of Fig. 2 to ready semantics, where $\mathcal{I} = \mathcal{R}$.

First note that in the setting of ready semantics, the observations provided by the output operation, which we denote by $\overline{o}_{\mathcal{R}}$, refer to the sets of actions ready to be executed by the states of the LTS. Therefore, $\overline{o}_{\mathcal{R}}$ is defined as follows:

$$\begin{aligned}&\overline{o}_{\mathcal{R}}\colon (\mathcal{P}_\omega X)^A \to \mathcal{P}_\omega(\mathcal{P}_\omega A)\\ &\overline{o}_{\mathcal{R}}(\varphi) = \{I(\varphi)\}.\end{aligned}$$

For the case $\varphi = \delta(x)$, where $x \in X$, it holds that:

$$\overline{o}_{\mathcal{R}}(\delta(x)) = \{I(\delta(x))\} = \{\{a \in A \mid \delta(x)(a) \neq \emptyset\}\}.$$

Remark 4.1. Observe that the codomain of $\bar{o}_{\mathcal{R}}$ is $\mathcal{P}_\omega(\mathcal{P}_\omega A)$, and not $\mathcal{P}_\omega A$, as one might expect. This choice is motivated by the intention of "distinguishing" between the ready actions of states $y \in Y$ in the Moore automata derived according to the powerset construction. See Example 4.3 for a concrete case study.

Consequently, $B_{\mathcal{I}} = B_{\mathcal{R}} = \mathcal{P}_\omega(\mathcal{P}_\omega A)$ and the final Moore coalgebra

$$((\mathcal{P}_\omega(\mathcal{P}_\omega A))^{A^*}, \langle \epsilon, (-)_a \rangle)$$

associates to each state $\{x\}$ the set of action sequences $w \in A^*$ such that $x \xrightarrow{w} x'$, together with the sets of actions ready to be triggered by (all such) x', for $x, x' \in X$.

Next, we will prove the equivalence between the coalgebraic modelling of ready semantics and the original definition, presented above. More explicitly, given an arbitrary LTS $(X, \delta\colon X \to (\mathcal{P}_\omega X)^A)$ and a state $x \in X$, we want to show that $[\![\{x\}]\!]$ is equal to $\mathcal{R}(x)$.

The first remark is that the behaviour of a state $x \in X$ is a function $[\![\{x\}]\!]\colon A^* \to \mathcal{P}_\omega(\mathcal{P}_\omega A)$, whereas $\mathcal{R}(x)$ is defined as a set of pairs in $A^* \times \mathcal{P}_\omega A$. However, this is no problem since the set of functions $A^* \to \mathcal{P}_\omega(\mathcal{P}_\omega A)$ and $\mathcal{P}(A^* \times \mathcal{P}_\omega A)$ are isomorphic. The set of all ready pairs $\mathcal{R}(x)$ associated to $x \in X$ is equivalently represented by $\varphi^{\mathcal{R}}_{\{x\}}$, where, for $w \in A^*$ and $Y \subseteq X$,

$$\begin{array}{l} \varphi^{\mathcal{R}}_Y\colon A^* \to \mathcal{P}_\omega(\mathcal{P}_\omega A) \\ \varphi^{\mathcal{R}}_Y(w) = \{Z \subseteq A \mid \exists y \in t(Y)(w) \wedge Z = I(\delta(y))\} \end{array}$$

At this point, showing the equivalence between the coalgebraic and the original definition of ready semantics reduces to proving that

$$(\forall x \in X)\,.\,[\![\{x\}]\!] = \varphi^{\mathcal{R}}_{\{x\}}. \tag{9}$$

Equality (9) is a direct consequence of the following theorem:

Theorem 4.1. Let $(X, \delta\colon X \to (\mathcal{P}_\omega X)^A)$ be an LTS. Then for all $Y \subseteq X$ and $w \in A^*$, $[\![Y]\!](w) = \varphi^{\mathcal{R}}_Y(w)$.

Proof. We proceed by induction on words $w \in A^*$.

— *Base case.* $w = \varepsilon$. Consider an arbitrary set $Y \subseteq X$. We have:

$$\begin{array}{rcl} [\![Y]\!](\varepsilon) & = & o(Y) = \bigcup\limits_{y \in Y} \{I(\delta(y))\} \\ \varphi^{\mathcal{R}}_Y(\varepsilon) & = & \{Z \subseteq A \mid \exists y \in Y \wedge Z = I(\delta(y))\} \text{ (by def., } (\forall y \in Y)\,.\,y \xrightarrow{\varepsilon} y) \\ & = & \bigcup\limits_{y \in Y} \{I(\delta(y))\} \end{array}$$

Hence, $[\![Y]\!](\varepsilon) = \varphi^{\mathcal{R}}_Y(\varepsilon)$, for all $Y \subseteq X$.

— *Induction step.*
Consider $w \in A^*$ and assume $[\![Y]\!](w) = \varphi^{\mathcal{R}}_Y(w)$, for all $Y \subseteq X$. We want to prove that $[\![Y]\!](aw) = \varphi^{\mathcal{R}}_Y(aw)$, where $a \in A$.

$$\begin{array}{rcl} [\![Y]\!](aw) & = & [\![t(Y)(a)]\!](w) \\ \varphi^{\mathcal{R}}_Y(aw) & = & \{Z \mid \exists y \in t(Y)(aw) \wedge Z = I(\delta(y))\} \\ & = & \{Z \mid \exists y \in t(t(Y)(a))(w) \wedge Z = I(\delta(y))\} \\ & = & \varphi^{\mathcal{R}}_{t(Y)(a)}(w) \end{array}$$

By the induction hypothesis, it follows that $[\![Y]\!](aw) = \varphi^{\mathcal{R}}_Y(aw)$, for all $Y \subseteq X$.

We have that $[\![Y]\!](w) = \varphi^{\mathcal{R}}_Y(w)$, for all $Y \subseteq X$ and $w \in A^*$. $\square$

Example 4.1. In what follows we illustrate the equivalence between the coalgebraic and the original definitions of ready semantics by means of an example. Consider the following LTS.

$$\begin{tikzcd}
& p_0 \arrow[loop right, "a"] \arrow[d, "a"] & \\
& p_1 \arrow[dl, "b"'] \arrow[dr, "b"] & \\
p_2 \arrow[d, "c"'] & & p_3 \arrow[d, "d"] \\
p_4 & & p_5
\end{tikzcd}$$

We write a^n to represent the action sequence $aa \dots a$ of length $n \geq 1$, with $n \in \mathbb{N}$. The set of all ready pairs associated to p_0 is:

$$\begin{aligned}\mathcal{R}(p_0) \quad = \quad & \{(\varepsilon, \{a\}), (a^n, \{a\}), (a^n, \{b\}), (a^n b, \{c\}), (a^n b, \{d\}), \\ & (a^n bc, \emptyset), (a^n bd, \emptyset) \mid n \in \mathbb{N} \wedge n \geq 1\}.\end{aligned}$$

We can construct a Moore automaton, for $S = \{p_0, p_1, \dots, p_5\}$,

$$(\mathcal{P}_\omega S, \langle o, t \rangle \colon \mathcal{P}_\omega S \to \mathcal{P}_\omega(\mathcal{P}_\omega A) \times (\mathcal{P}_\omega S)^A)$$

by applying the generalized powerset construction on the LTS above. The automaton will have $2^6 = 64$ states. We depict the accessible part from state $\{p_0\}$, where the output sets are indicated by double arrows:

$$\begin{tikzcd}
& & \{p_0\} \arrow[r, Rightarrow] \arrow[d, "a"] & \{\{a\}\} & \\
& & \{p_0, p_1\} \arrow[r, Rightarrow] \arrow[loop left, "a"] \arrow[d, "b"] & \{\{a\}, \{b\}\} & \\
& & \{p_2, p_3\} \arrow[r, Rightarrow] \arrow[dl, "c"'] \arrow[dr, "d"] & \{\{c\}, \{d\}\} & \\
\{\emptyset\} & \{p_4\} \arrow[l, Rightarrow] & & \{p_5\} \arrow[r, Rightarrow] & \{\emptyset\}
\end{tikzcd}$$

Fig. 3. Ready determinization when starting from $\{p_0\}$.

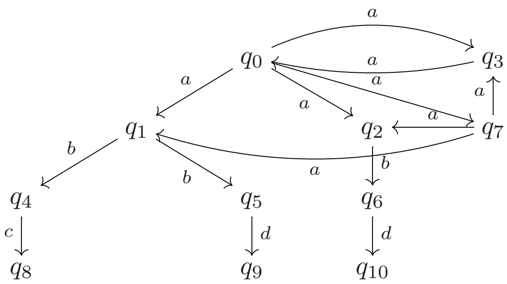
The output sets of a state Y of the Moore automaton in Fig. 3 is the set of actions associated to a certain state $y \in Y$ which can immediately be performed. For example, process p_0 in the original LTS above is ready to perform action a, whereas p_1 can immediately perform b. Therefore it holds that $o(\{p_0\}) = \{\{a\}\}$ and $o(\{p_0, p_1\}) = \{\{a\}, \{b\}\}$.

At this point, by simply looking at the automaton in Fig. 3, one can easily see that the set of action sequences $w \in A^*$ the state $\{p_0\}$ can execute, together with the corresponding possible next actions equals $\mathcal{R}(p_0)$. Therefore, the automaton generated according to the generalized powerset construction captures the set of all ready pairs of the initial LTS.

As we remarked in Section 3, ready equivalence of LTS's can be established in terms of bisimulation up-to context on Moore automata with output in $\mathcal{P}_\omega(\mathcal{P}_\omega A)$, representing the sets of actions ready to be triggered.

Next, we will explain how one can reason on ready equivalence of two LTS's, by constructing bisimulations up-to context on the associated Moore automata generated according to the powerset construction in Fig. 2.

Example 4.2. Consider the following LTS.



It is easy to check that q_0 and p_0 have the same ready pairs, that is $\mathcal{R}(q_0) = \mathcal{R}(p_0)$, where p_0 is the state in the LTS Example 4.1.

Since we have shown the coincidence between the original definition involving equality of ready pairs and the coalgebraic representation, we can now prove that q_0 and p_0 are ready equivalent by building a bisimulation up-to context relating $\{p_0\}$ and $\{q_0\}$.

First, we have to determinize the LTS above. In Fig. 4 we show the accessible part of the determinized automaton starting from state $\{q_0\}$:

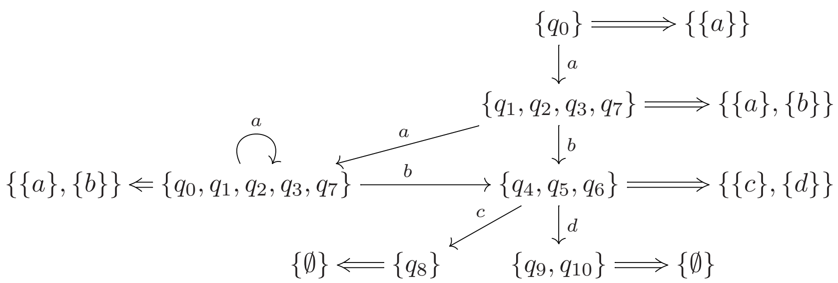


Fig. 4. Ready determinization when starting from $\{q_0\}$.

The next step is to build a bisimulation up-to context R on the sets of states of the generated Moore automata in Fig. 3 and Fig. 4, such that $(\{p_0\}, \{q_0\}) \in R$.

We start by taking $R = \{(\{p_0\}, \{q_0\})\}$ and check whether this is already a bisimulation up-to context, by considering the output values and transitions, and check whether no new states appear in $c(R)$ in the process. If new pairs of states appear, we add them to R and repeat the process.

Eventually, we end-up with a bisimulation up-to context

$$\begin{aligned} R \;=\; & \{(\{p_0\}, \{q_0\}), (\{p_0, p_1\}, \{q_1, q_2, q_3, q_7\}), \\ & (\{p_2, p_3\}, \{q_4, q_5, q_6\}), (\{p_4\}, \{q_8\}), (\{p_5\}, \{q_9, q_{10}\})\} \end{aligned}$$

By construction $(\{p_0\}, \{q_0\}) \in R$, so by (7) it follows that $[\![\{p_0\}]\!] = [\![\{q_0\}]\!]$.

Note that R is not a bisimulation relation since $\{p_0, p_1\} \xrightarrow{a} \{p_0, p_1\}$ and $\{q_1, q_2, q_3, q_7\} \xrightarrow{a} \{q_0, q_1, q_2, q_3, q_7\}$ but $(\{p_0, p_1\}, \{q_0, q_1, q_2, q_3, q_7\}) \notin R$. Nevertheless, observe that R is a bisimulation up-to context since $(\{p_0, p_1\}, \{q_0, q_1, q_2, q_3, q_7\}) \in c(R)$:

$$\begin{array}{rcll}
\{p_0, p_1\} & = & \{p_0\} \cup \{p_0, p_1\} & \\
 & c(R) & \{q_0\} \cup \{p_0, p_1\} & ((\{p_0\}, \{q_0\}) \in R) \\
 & c(R) & \{q_0\} \cup \{q_1, q_2, q_3, q_7\} & ((\{p_0, p_1\}, \{q_1, q_2, q_3, q_7\}) \in R) \\
 & = & \{q_0, q_1, q_2, q_3, q_7\} &
\end{array}$$

Also observe that the bisimulation up-to context given above is one pair smaller than the Moore bisimulation relating the automata in Fig. 3 and Fig. 4, which would also include $(\{p_0, p_1\}, \{q_0, q_1, q_2, q_3, q_7\})$.

Example 4.3. In what follows we provide an example supporting the statement in Remark 4.1. Consider the two LTS's below:

$$\begin{tikzcd}
& p_0 \arrow[dl, "a"'] \arrow[dr, "a"] & & & q_0 \arrow[d, "a"] & \\
p_1 \arrow[d, "b"'] & & p_2 \arrow[d, "c"] & & q_1 \arrow[dl, "b"'] \arrow[dr, "c"] & \\
p_3 & & p_4 & q_2 & & q_3
\end{tikzcd}$$

It is easy to see that p_0 and q_0 are not ready equivalent, as $(a, \{b, c\})$ is a ready pair of q_0 and not of p_0. Now assume the following definition of $\overline{o}_{\mathcal{R}}$ (having the codomain $\mathcal{P}_\omega A$ instead of $\mathcal{P}_\omega(\mathcal{P}_\omega A)$):

$$\begin{array}{l}
\overline{o}_{\mathcal{R}} \colon (\mathcal{P}_\omega X)^A \to \mathcal{P}_\omega A \\
\overline{o}_{\mathcal{R}}(\varphi) = I(\varphi).
\end{array}$$

The derived Moore automata starting from $\{p_0\}, \{q_0\}$ are (trivially) bisimilar:

$$\begin{tikzcd}
\{a\} & \{p_0\} \arrow[l, Rightarrow] \arrow[d, "a"'] & & & \{q_0\} \arrow[r, Rightarrow] \arrow[d, "a"] & \{a\} \\
\{b, c\} & \{p_1, p_2\} \arrow[l, Rightarrow] \arrow[dl, "b"'] \arrow[dr, "c"] & & & \{q_1\} \arrow[r, Rightarrow] \arrow[dl, "b"'] \arrow[dr, "c"] & \{b, c\} \\
\{p_3\} \arrow[d, Rightarrow] & & \{p_4\} \arrow[d, Rightarrow] & \{q_2\} \arrow[d, Rightarrow] & & \{q_3\} \arrow[d, Rightarrow] \\
\emptyset & & \emptyset & \emptyset & & \emptyset
\end{tikzcd}$$

implying that p_0 and q_0 are ready equivalent – which is a contradiction! Obviously this is a consequence of the fact that we identify states $\{p_1, p_2\}$ and $\{q_1\}$, as they both output $\{b, c\}$. One way to annihilate this drawback is to "separate" the set of actions ready to be triggered by p_1 and p_2, respectively, by considering

$$\overline{o}_{\mathcal{R}}(\varphi) = \{I(\varphi)\} \in \mathcal{P}_\omega(\mathcal{P}_\omega A).$$

The new Moore automata in Fig. 5 generated starting from $\{p_0\}$ and $\{q_0\}$ are not bisimilar, as

$$\begin{aligned} o(\{p_1, p_2\}) &= \{\{b\}, \{c\}\}, \text{ whereas} \\ o(\{q_1\}) &= \{\{b, c\}\}. \end{aligned}$$

Intuitively, the outputs above refer to the possibility of q_0 to select between executing b or c after triggering a, choice impossible for p_0.

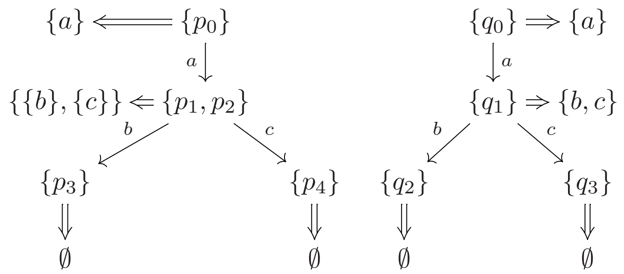


Fig. 5. Determinization from $\{p_0\}, \{q_0\}$, when $\overline{o}_{\mathcal{R}}(\varphi) = \{I(\varphi)\} \in \mathcal{P}_\omega(\mathcal{P}_\omega A)$

4.2. *Failure semantics*

In this section, we present the coalgebraic modelling of failure semantics, along the lines of the previous section. Moreover, we prove the equivalence of the coalgebraic modelling with its standard definition, and show how one can reason on failure equivalence in terms of bisimulations (up-to).

Consider an LTS $(X, \delta\colon X \to (\mathcal{P}_\omega X)^A)$ and a function $\varphi\colon A \to \mathcal{P}_\omega X$. The set of *actions φ fails to enable* is given by

$$Fail(\varphi) = \{Z \subseteq A \mid Z \cap I(\varphi) = \emptyset\}$$

where $I(\varphi)$ is defined as in (8).

Note that for the particular case $\varphi = \delta(x)$, $Fail(\delta(x))$ represents the set of subsets of all (initial) actions that cannot be triggered by $x \in X$.

A *failure pair* of x is a pair $(w, Z) \in A^* \times \mathcal{P}_\omega A$ such that $x \xrightarrow{w} y$ and $Z \in Fail(\delta(y))$.

Failure semantics identifies behaviours of states in X according to their failure pairs.

Definition 4.2 ($\mathcal{F}$-equivalence (van Glabbeek 2001)). Let $(X, \delta\colon X \to (\mathcal{P}_\omega X)^A)$ be an LTS and $x, y \in X$ two states. States x and y are *failure equivalent* ($\mathcal{F}$-equivalent) if and only if $\mathcal{F}(x) = \mathcal{F}(y)$, where

$$\mathcal{F}(x) = \{(w, Z) \in A^* \times \mathcal{P}_\omega A \mid \exists x' \in X.\, x \xrightarrow{w} x' \wedge Z \in Fail(\delta(x'))\}.$$

The coalgebraic modelling of $\mathcal{F}$-equivalence is obtained by again instantiating the ingredients of Fig. 2 as follows.

The output operation $\overline{o}_{\mathcal{I}} = \overline{o}_{\mathcal{F}}$ refers to the sets of actions the states of the LTS cannot immediately fire and is defined as follows:

$$\begin{aligned} &\overline{o}_{\mathcal{F}}\colon (\mathcal{P}_\omega X)^A \to \mathcal{P}_\omega(\mathcal{P}_\omega A) \\ &\overline{o}_{\mathcal{F}}(\varphi) = Fail(\varphi). \end{aligned}$$

If $\varphi = \delta(x)$, for $x \in X$, it holds that

$$\overline{o}_{\mathcal{F}}(\delta(x)) = Fail(\delta(x)) = \{Z \subseteq A \mid Z \cap I(\delta(x)) = \emptyset\}.$$

Consequently, the output function o of the determinized automaton generated according to the powerset construction is:

$$\begin{array}{l} o\colon (\mathcal{P}_\omega X) \to \mathcal{P}_\omega(\mathcal{P}_\omega A) \\ o(Y) = \bigcup\limits_{y \in Y} \bar{o}_{\mathcal{F}}(\delta(y)) = \bigcup\limits_{y \in Y} Fail(\delta(y)). \end{array}$$

Intuitively, o outputs the sets of actions that cannot be executed as a first step by all states $y \in Y$, for $Y \in \mathcal{P}_\omega X$. We have

$$B_{\mathcal{F}} = \mathcal{P}_\omega(\mathcal{P}_\omega A)$$

and the final Moore coalgebra is then instantiated to

$$((\mathcal{P}_\omega(\mathcal{P}_\omega A))^{A^*}, \langle \epsilon, (-)_a \rangle)$$

where the final map associates to each state $\{x\}$ the set of words $w \in A^*$ such that $x \xrightarrow{w} x'$, together with the actions (all such) x' cannot trigger, for $x, x' \in X$.

In order to show the equivalence between the two representations of failure semantics, we capture the set of all failure pairs $\mathcal{F}(x)$ associated to states $x \in X$ by means of a function $\varphi^{\mathcal{F}}_Y$ defined as follows:

$$\begin{array}{l} \varphi^{\mathcal{F}}_Y \colon A^* \to \mathcal{P}_\omega(\mathcal{P}_\omega A) \\ \varphi^{\mathcal{F}}_Y(w) = \{Z \subseteq A \mid \exists y \in t(Y)(w) \wedge Z \in Fail(\delta(y))\}. \end{array}$$

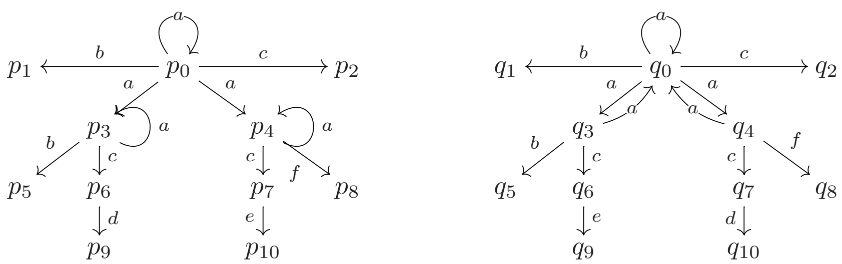
The set $\mathcal{F}(x)$ of all failure pairs of a state $x \in X$ is equivalently represented by $\varphi^{\mathcal{F}}_{\{x\}}$. Therefore, the equivalence between the two representations of failure semantics reduces to showing that

$$(\forall x \in X) \,.\, [\![\{x\}]\!] = \varphi^{\mathcal{F}}_{\{x\}}. \tag{10}$$

The statement in (10) follows directly from the following Theorem.

Theorem 4.2. Let $(X, \delta\colon X \to (\mathcal{P}_\omega X)^A)$ be an LTS. Then for all $Y \subseteq X$ and $w \in A^*$, $[\![Y]\!](w) = \varphi^{\mathcal{F}}_Y(w)$.

Example 4.4. Consider the following LTS's.



Let $A = \{a_1, a_2, \ldots, a_n\}$ be the set of actions a process fails executing as a first step. For the simplicity of notation, we write $[a_1 a_2 \ldots a_n]$ to denote the set of all non-empty subsets $Z \subseteq A$. For example, if $A = \{a_1, a_2\}$, then $[a_1 a_2]$ stands for $\{\{a_1\}, \{a_2\}, \{a_1, a_2\}\}$.

Note that p_0 and q_0 are $\mathcal{F}$-equivalent, according to Definition 4.2, as they have the same sets of failure pairs:

$$\begin{aligned}\mathcal{F}(p_0) = \mathcal{F}(q_0) \quad = \quad & \{(\varepsilon, [def]), (b, [abcdef]), (c, [abcdef])\} \cup \\ & \{(a^n, [def]), (a^n, [bde]), (a^n b, [abcdef]), (a^n c, [abcdef]), \\ & \ (a^n c, [abcef]), (a^n c, [abcdf]), (a^n f, [abcdef]), \\ & \ (a^n cd, [abcdef]), (a^n ce, [abcdef]) \mid n \in \mathbb{N}, n \geq 1\}\end{aligned}$$

The same conclusion can be reached by checking behavioural equivalence of the two Moore automata generated according to the powerset construction, starting with $\{p_0\}$ and $\{q_0\}$. The fragments of the two automata starting from the states $\{p_0\}$ and $\{q_0\}$ are depicted in Fig. 6.

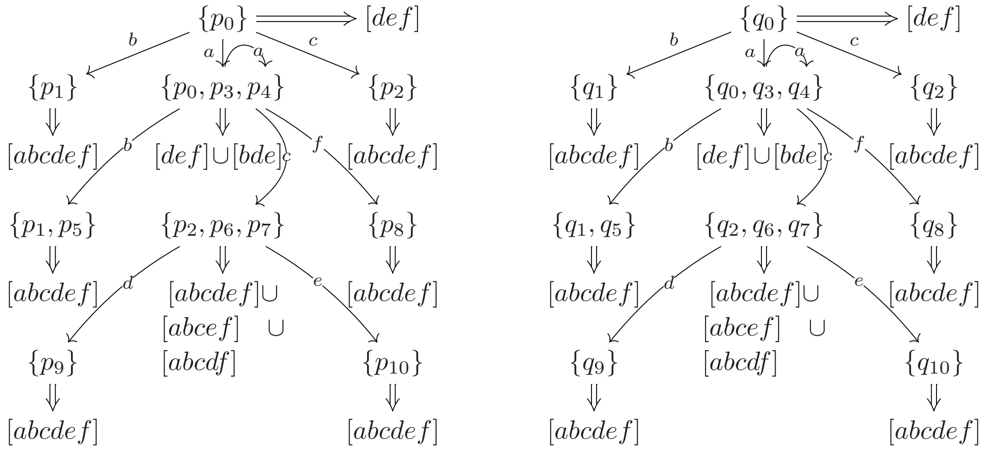


Fig. 6. Failure determinization when starting from $\{p_0\}$ and $\{q_0\}$.

Obviously $\{p_0\}$ and $\{q_0\}$ are Moore bisimilar, since the automata above have the same branching structure, the transitions have the same labels, and the states the same outputs.

4.3. *Trace semantics*

In this section we adapt the setting illustrated in Fig. 2 in Section 3 and provide a coalgebraic modelling of trace semantics. We also show that the coalgebraic characterization we get is equivalent to the original definition.

Consider an LTS $(X, \delta\colon X \to (\mathcal{P}_\omega X)^A)$. Trace semantics identifies states in X according to the set of words $w \in A^*$ they can execute.

Definition 4.3 ($\mathcal{T}$-equivalence (van Glabbeek 2001)). Let $(X, \delta\colon X \to (\mathcal{P}_\omega X)^A)$ be an LTS and $x, y \in X$ two states. States x and y are *trace equivalent* ($\mathcal{T}$-equivalent) if and only if $\mathcal{T}(x) = \mathcal{T}(y)$, where

$$\mathcal{T}(x) = \{w \in A^* \mid \exists x' \in X.\, x \xrightarrow{w} x'\}. \tag{11}$$

First note that for this type of semantics, one does not distinguish between traces and complete traces. Intuitively, all states are accepting, so they have the same observable behaviour, no matter the transitions they perform. Therefore, we define $\overline{o}_{\mathcal{T}}$ as:

$$\begin{aligned}&\overline{o}_{\mathcal{T}} \colon (\mathcal{P}_\omega X)^A \to 2\\ &\overline{o}_{\mathcal{T}}(\varphi) = 1.\end{aligned}$$

Consequently, the output function $o\colon \mathcal{P}_\omega X \to 2$ of the generated Moore automaton according to the generalized powerset construction is defined by $o(Y) = 1$.

Note that $B_{\mathcal{T}} = 2$ and the final Moore coalgebra in Fig. 2 is the set of languages 2^{A^*} over A (and the transition structure $\langle \epsilon, (-)_a\rangle$ is simply given by Brzozowski derivatives). Therefore, we can state that the map into the final coalgebra associates to each state $Y \in \mathcal{P}_\omega X$ the set of all traces corresponding to states $y \in Y$, namely, the language:

$$L = \bigcup_{y\in Y} \{w \in A^* \mid (\exists y' \in X) \,.\, y \xrightarrow{w} y'\}.$$

The set $\mathcal{P}(A^*)$ is isomorphic to the set of functions 2^{A^*} which enables us to represent the set $\mathcal{T}(x)$ in terms of a function $\varphi^{\mathcal{T}}_Y$ defined, for $w \in A^*$ and $Y \subseteq X$, as follows:

$$\begin{aligned}&\varphi^{\mathcal{T}}_Y \colon A^* \to 2\\ &\varphi^{\mathcal{T}}_Y(w) = 1 \text{ if } (\exists y \in Y, y' \in X) \,.\, y \xrightarrow{w} y'\end{aligned}$$

Equivalently, $\varphi^{\mathcal{T}}_Y(w) = 1$ if and only if $t(Y)(w) \neq \emptyset$.

The set of all traces $\mathcal{T}(x)$ corresponding to $x \in X$ is modelled by $\varphi^{\mathcal{T}}_{\{x\}}$.

Recall that the behaviour of a state $x \in X$, *i.e.*, the traces of x, is represented in the final coalgebra by $[\![\{x\}]\!]$. Therefore, proving the equivalence between the coalgebraic and the classic definition of trace semantics reduces to showing that

$$(\forall x \in X) \,.\, [\![\{x\}]\!] = \varphi^{\mathcal{T}}_{\{x\}}. \tag{12}$$

Equality (12) is a direct result of the following theorem:

Theorem 4.3. Let $(X, \delta\colon X \to (\mathcal{P}_\omega X)^A)$ be an LTS. Then for all $Y \subseteq X$ and $w \in A^*$, $[\![Y]\!](w) = \varphi^{\mathcal{T}}_Y(w)$.

Example 4.5. Consider the following LTS's.

$$\begin{tikzcd}
& r_0 \arrow[r, "a"] \arrow[d, "a"'] & r_6 \\
& r_1 \arrow[dl, "b"'] \arrow[dr, "b"] & \\
r_2 \arrow[d, "c"'] & & r_3 \arrow[d, "d"] \\
r_4 & & r_5
\end{tikzcd}
\qquad
\begin{tikzcd}
& s_0 \arrow[d, "a"'] \arrow[drrr, "a"] & & & & \\
& s_1 \arrow[d, "b"'] & & & s_2 \arrow[d, "b"] & \\
& s_3 \arrow[dl, "c"'] \arrow[dr, "d"] & & & s_4 \arrow[dl, "d"'] \arrow[dr, "d"] & \\
s_5 & & s_6 & s_7 & & s_8
\end{tikzcd}$$

It is easy to see that the set of all traces corresponding to r_0 and s_0 are equal:

$$\mathcal{T}(r_0) = \mathcal{T}(s_0) = \{\varepsilon, a, ab, abc, abd\}$$

therefore, they are trace-equivalent, according to Definition 4.3. Next we depict part of the determinizations of the LTS's above (omitting outputs, since they are all 1):

$$
\begin{tikzcd}[column sep=small]
& \{r_0\} \arrow[d, "a"'] & \\
& \{r_1, r_6\} \arrow[d, "b"'] & \\
& \{r_2, r_3\} \arrow[dl, "c"'] \arrow[dr, "d"] & \\
\{r_4\} & & \{r_5\}
\end{tikzcd}
\qquad
\begin{tikzcd}[column sep=small]
& \{s_0\} \arrow[d, "a"] & \\
& \{s_1, s_2\} \arrow[d, "b"] & \\
& \{s_3, s_4\} \arrow[dl, "c"'] \arrow[dr, "d"] & \\
\{s_5\} & & \{s_6, s_7, s_8\}
\end{tikzcd}
$$

Fig. 7. Trace determinizations when starting from $\{r_0\}$ and $\{s_0\}$.

It is easy to observe that the generated Moore automata are bisimilar, therefore by (12) it follows that r_0 and s_0 are indeed trace-equivalent.

4.4. *Complete trace semantics*

In this section we model coalgebraically complete trace semantics. Similar to the previous sections, we also show that the coalgebraic representation of this semantics is equivalent to the original definition in (van Glabbeek 2001).

Consider an LTS $(X, \delta \colon X \to (\mathcal{P}_\omega X)^A)$. Complete trace semantics identifies states $x \in X$ based on their set of complete traces. Recall that a trace $w \in A^*$ of x is complete if and only if x can perform w and reach a deadlock state y or, equivalently,

$$(\exists y \in X) \,.\, x \xrightarrow{w} y \wedge I(\delta(y)) = \emptyset.$$

The difference with the trace semantics in Section 4.3 is that now an external observer detects stagnation, or deadlock states of a system.

Formally, complete trace equivalence is defined as follows.

Definition 4.4 ($\mathcal{CT}$-equivalence (Aceto, Fokkink & Verhoef 1999)). Let $(X, \delta \colon X \to (\mathcal{P}_\omega X)^A)$ be an LTS and $x, y \in X$ two states. States x and y are *complete trace equivalent* ($\mathcal{CT}$-equivalent) if and only if $\mathcal{CT}(x) = \mathcal{CT}(y)$, where

$$\mathcal{CT}(x) = \{w \in A^* \mid \exists x' \in X.\, x \xrightarrow{w} x' \wedge I(\delta(x')) = \emptyset\}.$$

In what follows we instantiate the constituents of Fig. 2 in order to provide the coalgebraic modelling of complete trace semantics.

The distinction between deadlock states and states that can still execute actions $a \in A$ is made by the function $\overline{o}_{\mathcal{CT}}$ defined as:

$$\overline{o}_{\mathcal{CT}} \colon (\mathcal{P}_\omega X)^A \to 2$$
$$\overline{o}_{\mathcal{CT}}(\varphi) = \begin{cases} 1 & \text{if } I(\varphi) = \emptyset \\ 0 & \text{otherwise} \end{cases}$$

As in the previous section, $B_{\mathcal{CT}} = 2$ and the final coalgebra is the set of languages 2^{A^*}.

Consider, for example, the following LTS:

$$
\begin{tikzcd}
p_1 & p_0 \arrow[l, "a"'] \arrow[r, bend left, "a"] & p_2 \arrow[l, bend left, "b"]
\end{tikzcd}
$$

Note that $(ab)^*a$ is a complete trace of p_0, as

$$p_0 \xrightarrow{a} p_2 \xrightarrow{b} p_0 \xrightarrow{a} p_2 \xrightarrow{b} \dots \xrightarrow{b} p_0 \xrightarrow{a} p_1 \tag{13}$$

where p_1 cannot perform any further action.

The above behaviour, described in terms of transitions between states of the Moore automaton derived according to the generalized powerset construction, can be depicted as follows:

$$\{p_0\} \xrightarrow{a} \{p_1, p_2\} \xrightarrow{b} \{p_0\} \xrightarrow{a} \{p_1, p_2\} \xrightarrow{b} \dots \xrightarrow{b} \{p_0\} \xrightarrow{a} \{p_1, p_2\}$$

where p_1 is a deadlock state and p_2 is not.

Intuitively, we can state that $(ab)^*a$ is a complete trace of $\{p_0\}$, as the deadlock state $p_2 \in \{p_1, p_2\}$ can be reached from $\{p_0\}$ by performing $(ab)^*a$ (see (13)).

Therefore, given $Y_1, Y_2 \subseteq X$ and $w \in A^*$ such that $Y_1 \xrightarrow{w} Y_2$, we observe that w is a complete trace of Y_1 whenever there exists a deadlock state $y \in Y_2$. Otherwise, w is not a complete trace of Y_1.

In the coalgebraic modelling, the above observations regarding (non)stagnating states appear in the definition of the output function $o\colon (\mathcal{P}_\omega X)^A \to 2$:

$$o(Y) = \begin{cases} 1 & \text{if } (\exists y \in Y) \,.\, I(\delta(y)) = \emptyset \\ 0 & \text{otherwise} \end{cases}$$

According to Definition 4.4, in order for two processes $x, y \in X$ to be $\mathcal{CT}$-equivalent, $\mathcal{CT}(x) = \mathcal{CT}(y)$ must hold.

Recall that $t(Y)(w)$ stands for the set of all states y that can be reached via transitions $x \xrightarrow{w} y$, for some $x \in Y$. We further represent the set $\mathcal{CT}(x)$ associated to a state $x \in X$ in the initial LTS by means of the function $\varphi_Y^{\mathcal{CT}} \colon A^* \to 2$:

$$\varphi_Y^{\mathcal{CT}}(w) = \begin{cases} 1 & \text{if } (\exists y \in t(Y)(w)) \,.\, I(\delta(y)) = \emptyset \\ 0 & \text{otherwise} \end{cases}$$

The set of complete traces corresponding to a state $x \in X$ is modelled by $\varphi_{\{x\}}^{\mathcal{CT}}$.

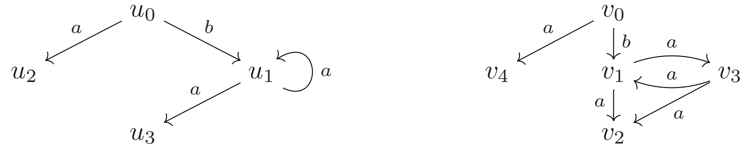
Proving that the coalgebraic modelling and the standard definition of complete trace semantics coincide consists in showing that

$$(\forall x \in X) \,.\, [\![\{x\}]\!] = \varphi_{\{x\}}^{\mathcal{CT}}. \tag{14}$$

Equality (14) follows directly from the following theorem.

Theorem 4.4. Let $(X, \delta\colon X \to (\mathcal{P}_\omega X)^A)$ be an LTS. Then for all $Y \subseteq X$ and $w \in A^*$, $[\![Y]\!](w) = \varphi_Y^{\mathcal{CT}}(w)$.

Example 4.6. Consider the following two LTS's.



The set of complete traces of states u_0 and v_0 are equal:

$$\mathcal{CT}(u_0) \quad = \quad \mathcal{CT}(v_0) \quad = \quad \{a\} \cup \{ba^n \mid n \in \mathbb{N}, n \geq 1\}.$$

Therefore, by Definition 4.4, u_0 and v_0 are $\mathcal{CT}$-equivalent.

We next show how one can prove equivalence using the coalgebraic modelling of complete trace semantics introduced in this section or, more precisely, the generalized determinization of the LTS's above for complete traces, of which we show a fragment in Fig. 8.

$$\begin{tikzcd}
& \{u_0\} \arrow[dl, "a"'] \arrow[d, "b"] \arrow[r, Rightarrow] & 0 \\
\{u_2\} \arrow[d, Rightarrow] & \{u_1\} \arrow[d, "a"] \arrow[r, Rightarrow] & 0 \\
1 & \{u_1, u_3\} \arrow[r, Rightarrow] \arrow[loop below, "a"] & 1
\end{tikzcd}
\qquad
\begin{tikzcd}
& \{v_0\} \arrow[dl, "a"'] \arrow[d, "b"] \arrow[r, Rightarrow] & 0 \\
\{v_4\} \arrow[d, Rightarrow] & \{v_1\} \arrow[d, "a"] \arrow[r, Rightarrow] & 0 \\
1 & \{v_2, v_3\} \arrow[r, Rightarrow] \arrow[d, bend left, "a"] & 1 \\
& \{v_1, v_2\} \arrow[r, Rightarrow] \arrow[u, bend left, "a"] & 1
\end{tikzcd}$$

Fig. 8. Complete trace determinization when starting from $\{u_0\}, \{v_0\}$.

In Fig. 8, states such as $\{u_2\}$ or $\{v_2, v_3\}$ output 1 as they contain deadlock states in the initial LTS's, namely u_2 and v_2. Therefore, $\{u_2\}$ and $\{v_2, v_3\}$ are at the end of paths corresponding to complete traces of the shape a and $ba^n (n \geq 1)$, respectively. It is easy to see that the top states of the systems in Fig. 8 recognize the same sets of complete traces as u_0 and v_0. Moreover, the states $\{u_0\}$ and $\{v_0\}$ of the systems above are behaviourally equivalent, as $(\{u_0\}, \{v_0\})$ is contained in the following Moore bisimulation relation:

$$\begin{aligned} R = \{ \quad & (\{u_0\}, \{v_0\}), (\{u_2\}, \{v_4\}), (\{u_1\}, \{v_1\}), \\ & (\{u_1, u_3\}, \{v_2, v_3\}), (\{u_1, u_3\}, \{v_1, v_2\}) \ \}. \end{aligned}$$

Therefore, one can prove complete trace equivalence of u_0 and v_0 by employing Moore bisimulations. Next, consider the following two LTS's

$$\begin{tikzcd}
w_1 & w_0 \arrow[l, "a"'] \arrow[loop right, "a"]
\end{tikzcd}
\qquad\qquad
\begin{tikzcd}
w_0' \arrow[loop right, "a"]
\end{tikzcd}$$

Observe that w_0 and w_0' are trace equivalent (according to Definition 4.3), as they output the same sets of traces

$$\mathcal{T}(w_0) = \mathcal{T}(w_0') = \{\varepsilon\} \cup \{a^n \mid n \in \mathbb{N}, n \geq 1\}$$

but they are not complete trace equivalent (according to Definition 4.4), as w_0' can never reach a deadlock state, whereas w_0 can reach the stagnating state w_1.

The complete trace determinization contains the sub-automata starting from states $\{w_0\}$ and $\{w_0'\}$ depicted in Fig. 9:

$$\begin{tikzcd} 0 & 1 & & 0 \\ \{w_0\} \arrow[u, Rightarrow] \arrow[r, "a"] & \{w_0, w_1\} \arrow[u, Rightarrow] \arrow[loop below, "a"] & & \{w_0'\} \arrow[u, Rightarrow] \arrow[loop right, "a"] \end{tikzcd}$$

Fig. 9. Complete trace determinization when starting from $\{w_0\}, \{w_0'\}$.

States $\{w_0\}$ and $\{w_0'\}$ are not behaviourally equivalent, since $\{w_0, w_1\}$ outputs 1, whereas $\{w_0'\}$ never reaches a state with this output. Hence, as expected, we will never be able to build a bisimulation containing states $\{w_0\}$ and $\{w_0'\}$.

4.5. *Possible-futures semantics*

In what follows we provide a coalgebraic modelling of possible-futures semantics and show that it coincides with the original definition in (van Glabbeek 2001). We also give an example on how the generalized powerset construction and Moore bisimulations (up-to) can be used in order to reason on possible-futures equivalence.

Let $(X, \delta \colon X \to (\mathcal{P}_\omega X)^A)$ be an LTS and recall that a *possible future* of $x \in X$ is a pair $\langle w, T\rangle \in A^* \times \mathcal{P}(A^*)$ such that $x \xrightarrow{w} y$ and $T = \mathcal{T}(y)$ (where $\mathcal{T}(y)$ is the set of traces of y, as in Section 4.3).

Possible-futures semantics identifies states that can trigger the same sets of traces $w \in A^*$ and moreover, by executing such w, they reach trace-equivalent states.

Definition 4.5 ($\mathcal{PF}$-equivalence (van Glabbeek 2001)). Let $(X, \delta \colon X \to (\mathcal{P}_\omega X)^A)$ be an LTS and $x, y \in X$ two states. States x and y are *possible-futures equivalent* ($\mathcal{PF}$-equivalent) if and only if $\mathcal{PF}(x) = \mathcal{PF}(y)$, where

$$\mathcal{PF}(x) = \{\langle w, T\rangle \in A^* \times \mathcal{P}(A^*) \mid \exists x' \in X.\, x \xrightarrow{w} x' \wedge T = \mathcal{T}(x')\}.$$

The ingredients of Fig. 2 are instantiated as follows.

The output function $\bar{o}_{\mathcal{I}} = \bar{o}_{\mathcal{PF}}$, which refers to the set of traces enabled by states $x \in X$ of the LTS, is defined as

$$\begin{array}{l} \bar{o}_{\mathcal{PF}} \colon (\mathcal{P}_\omega X)^A \to \mathcal{P}_\omega(\mathcal{P}A^*) \\ \bar{o}_{\mathcal{PF}}(\varphi) = \{a \boxdot \overline{\mathcal{T}}(\varphi(a)) \mid \varphi(a) \neq \emptyset\}, \text{ where} \\ \overline{\mathcal{T}}(\varphi(a)) = \bigcup_{y \in \varphi(a)} \mathcal{T}(y) \\ a \boxdot \{w_i \mid i \in I\} = \{a w_i \mid i \in I\}. \end{array}$$

For $\varphi = \delta(x)$, with $x \in X$, it holds that

$$\begin{array}{rcl} \bar{o}_{\mathcal{PF}}(\delta(x)) & = & \{\{a \boxdot \overline{\mathcal{T}}(\delta(x)(a)) \mid \delta(x)(a) \neq \emptyset\}\} \\ & = & \{\{aw \mid \delta(x)(a) \neq \emptyset \wedge \exists y \in \delta(x)(a)\,.\, w \in \mathcal{T}(y)\}\} \\ & = & \mathcal{T}(x). \end{array}$$

We consider $\mathcal{P}_\omega(\mathcal{P}A^*)$ instead of just $\mathcal{P}A^*$ for the codomain of $\bar{o}_{\mathcal{PF}}$ in order to distinguish between the traces of states $y \in Y$ "collected" within states Y of the derived Moore automata (see Remark 4.1 and Example 4.3 in Section 4.1 for more details on a similar approach for the case of ready semantics).

Consequently,

$$B_{\mathcal{I}} = B_{\mathcal{PF}} = \mathcal{P}_\omega(\mathcal{P}A^*)$$

and the behaviour of a state $x \in X$ in the final coalgebra is given in terms of a function

$$[\![\{x\}]\!] \colon A^* \to \mathcal{P}_\omega(\mathcal{P}A^*)$$

which, intuitively, for each $w \in A^*$ returns the set of traces corresponding to states $y \in X$ such that $x \xrightarrow{w} y$.

Next we want to show that for each $x \in X$, $[\![\{x\}]\!]$ and $\mathcal{PF}(x)$ coincide.

First we choose to equivalently represent $\mathcal{PF}(x) \in \mathcal{P}_\omega(A^* \times \mathcal{P}(A^*))$ – the set of all possible futures of a state $x \in X$ – in terms of $\varphi^{\mathcal{PF}}_{\{x\}}$, where

$$\begin{array}{l} \varphi^{\mathcal{PF}}_Y \colon A^* \to \mathcal{P}_\omega(\mathcal{P}A^*) \\ \varphi^{\mathcal{R}}_Y(w) = \{\mathcal{T}(y) \mid y \in t(Y)(w)\} \end{array}$$

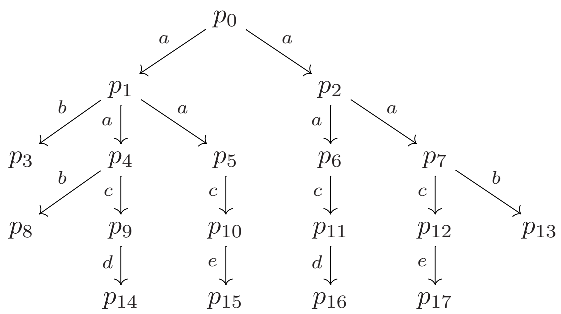
(note that $\mathcal{P}_\omega(A^* \times \mathcal{P}(A^*))$ and $(\mathcal{P}_\omega(\mathcal{P}A^*))^{A^*}$ are isomorphic structures). Therefore, showing the equivalence between the coalgebraic and the original definition of possible-futures semantics reduces to proving that

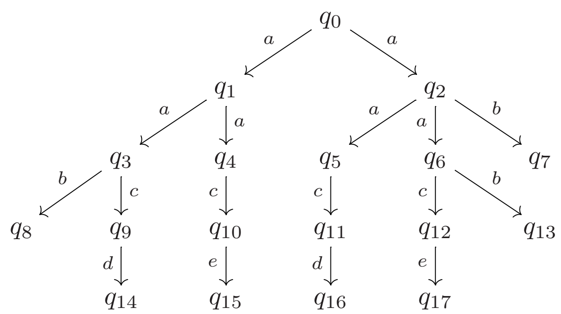
$$(\forall x \in X)\,.\, [\![\{x\}]\!] = \varphi^{\mathcal{PF}}_{\{x\}}. \tag{15}$$

Equality (15) is a direct consequence of the following theorem:

Theorem 4.5. Let $(X, \delta \colon X \to (\mathcal{P}_\omega X)^A)$ be an LTS. Then for all $Y \subseteq X$ and $w \in A^*$, $[\![Y]\!](w) = \varphi^{\mathcal{PF}}_Y(w)$.

Example 4.7. Consider the following LTS's.





$$
\begin{tikzcd}
 & & \{p_0\} \arrow[r, Rightarrow] \arrow[d, "a"'] & \{\mathcal{T}(p_0)\} \\
 & & \{p_1, p_2\} \arrow[r, Rightarrow] \arrow[dl, "a"'] \arrow[d, "b"'] & \{\mathcal{T}(p_1), \mathcal{T}(p_2)\} \\
\{\mathcal{T}(p_4), \mathcal{T}(p_5), \mathcal{T}(p_6), \mathcal{T}(p_7)\} & \{p_4, p_5, p_6, p_7\} \arrow[l, Rightarrow] \arrow[d, "b"'] \arrow[dr, "c"] & \{p_3\} \arrow[r, Rightarrow] & \{\emptyset\} \\
\{\emptyset\} & \{p_8, p_{13}\} \arrow[l, Rightarrow] & \{p_9, p_{10}, p_{11}, p_{12}\} \arrow[r, Rightarrow] \arrow[dl, "d"'] \arrow[d, "e"] & \{\mathcal{T}(p_9), \mathcal{T}(p_{10}), \mathcal{T}(p_{11}), \mathcal{T}(p_{12})\} \\
 & \{p_{14}, p_{16}\} \arrow[d, Rightarrow] & \{p_{15}, p_{17}\} \arrow[d, Rightarrow] & \\
 & \{\emptyset\} & \{\emptyset\} &
\end{tikzcd}
$$

$$
\begin{tikzcd}
 & & \{q_0\} \arrow[r, Rightarrow] \arrow[d, "a"'] & \{\mathcal{T}(q_0)\} \\
 & & \{q_1, q_2\} \arrow[r, Rightarrow] \arrow[dl, "a"'] \arrow[d, "b"'] & \{\mathcal{T}(q_1), \mathcal{T}(q_2)\} \\
\{\mathcal{T}(q_3), \mathcal{T}(q_4), \mathcal{T}(q_5), \mathcal{T}(q_6)\} & \{q_3, q_4, q_5, q_6\} \arrow[l, Rightarrow] \arrow[d, "b"'] \arrow[dr, "c"] & \{q_7\} \arrow[r, Rightarrow] & \{\emptyset\} \\
\{\emptyset\} & \{q_8, q_{13}\} \arrow[l, Rightarrow] & \{q_9, q_{10}, q_{11}, q_{12}\} \arrow[r, Rightarrow] \arrow[dl, "d"'] \arrow[d, "e"] & \{\mathcal{T}(q_9), \mathcal{T}(q_{10}), \mathcal{T}(q_{11}), \mathcal{T}(q_{12})\} \\
 & \{q_{14}, q_{16}\} \arrow[d, Rightarrow] & \{q_{15}, q_{17}\} \arrow[d, Rightarrow] & \\
 & \{\emptyset\} & \{\emptyset\} &
\end{tikzcd}
$$

Fig. 10. Possible-futures determinization when starting from $\{p_0\}, \{q_0\}$.

Note that p_0 and q_0 are possible-futures equivalent, as the traces both can follow are sequences $w \in \{a, ab, aa, aab, aac, aacd, aace\}$ and moreover, by triggering the same w they reach states with equal sets of traces. The equivalence between p_0 and q_0 can be formally captured in terms of a bisimulation relation R on the associated Moore automata (generated according to the generalized powerset construction) depicted in Fig. 10, where

$$
\begin{aligned}
R = \{ \quad & (\{p_0\}, \{q_0\}), (\{p_1, p_2\}, \{q_1, q_2\}), (\{p_3\}, \{q_7\}), \\
& (\{p_5, p_5, p_6, p_7\}, \{q_3, q_4, q_5, q_6\}), (\{p_8, p_{13}\}, \{q_8, q_{13}\}), \\
& (\{p_9, p_{10}, p_{11}, p_{12}\}, \{q_9, q_{10}, q_{11}, q_{12}\}), \\
& (\{p_{14}, p_{16}\}, \{q_{14}, q_{16}\}), (\{p_{15}, p_{17}\}, \{q_{15}, q_{17}\}) \ \}.
\end{aligned}
$$

It is easy to check that R is a bisimulation, since both automata in Fig. 10 have the same branching structure, the corresponding transitions are labelled the same, and the outputs associated to the related states are equal. (Note that equality of the outputs – which are sets of traces – can be established using the framework introduced in Section 4.3.)

4.6. *Ready trace semantics*

In this section we provide a coalgebraic modelling of ready trace semantics by employing the generalized powerset construction. Similarly to the other semantics tackled so far, we show a) that the coalgebraic representation coincides with the original definition in (van Glabbeek 2001) and b) how to reason on ready trace equivalence in terms of Moore bisimulations (up-to).

We proceed by recalling some basic concepts.

Intuitively, ready trace semantics identifies two states if and only if they can follow the same traces w, and moreover, the corresponding (pairwise-taken) states determined by such w's have equivalent one-step behaviours. Formally, the definition is as follows:

Definition 4.6 ($\mathcal{RT}$-equivalence (van Glabbeek 2001)). Let $(X, \delta\colon X \to (\mathcal{P}_\omega X)^A)$ be an LTS and $x, y \in X$ two states. States x and y are *ready trace equivalent* ($\mathcal{RT}$-equivalent) if and only if $\mathcal{RT}(x) = \mathcal{RT}(y)$, where

$$\begin{array}{rl} \mathcal{RT}(x) = \{ & I_0 a_1 I_1 a_2 \ldots a_n I_n \in \mathcal{P}_\omega(A) \times (A \times \mathcal{P}_\omega(A))^* \mid \\ & (\exists x_1, \ldots, x_n \in X) \,.\, x_0 \xrightarrow{a_1} x_1 \xrightarrow{a_2} \ldots \xrightarrow{a_n} x_n \wedge \\ & (\forall i = 1, \ldots, n) \,.\, I_i = I(\delta(x_i)) \;\}. \end{array}$$

We call an element of $\mathcal{RT}(x)$ a *ready trace* of x.

As an element of novelty, note that in the current setting (and for the case of failure trace semantics in Section 4.7 as well), the instantiation of the ingredients in Fig. 2 is performed in the context of "preprocessed" versions of the original LTS's, enriched with some additional information.

On short, the preprocessing process consists in encoding within transitions of shape $x \xrightarrow{a} y$ also the set of actions ready (respectively, failed) to be triggered by x. Namely, $x \xrightarrow{\langle a, I(\delta(x))\rangle} y$ (respectively, $x \xrightarrow{\langle a, F\rangle} y$ where $F = Fail(\delta(x))$). This will eventually enable the construction of Moore automata "collecting" states that have been reached not only via one-step transitions labelled the same, but also from processes sharing the same sets of ready (respectively, failure) actions.

Each LTS $(X, \delta\colon X \to (\mathcal{P}_\omega X)^A)$ is associated a unique coalgebra $(X, \bar{\delta}\colon X \to (\mathcal{P}_\omega X)^{\bar{A}})$, where

$$\begin{array}{rcl} \bar{A} & = & A \times \mathcal{P}_\omega(A) \\ \bar{\delta}(x)(\langle a, I_x\rangle) & = & \begin{cases} \delta(x)(a) & \text{if } I_x = I(\delta(x)) \\ \emptyset & \text{otherwise} \end{cases} \end{array}$$

(See Example 4.8 for a more concrete insight on the preprocessing procedure and its effect.)

Using the new construction $(X, \bar{\delta}\colon X \to (\mathcal{P}_\omega X)^{\bar{A}})$ as a starting point, defining the constituents of Fig. 2 follows the recipe described in Section 4.1.

The output function $\bar{o}_{\mathcal{I}} = \bar{o}_{\mathcal{RT}}$ provides information with respect to the actions ready to be triggered by a state $x \in X$ as a first step:

$$\begin{array}{l} \overline{o}_{\mathcal{RT}} \colon (\mathcal{P}_\omega X)^{\bar{A}} \to \mathcal{P}_\omega(\mathcal{P}_\omega A) \\ \overline{o}_{\mathcal{RT}}(\bar{\delta}(x)) = \{I(\delta(x))\}. \end{array}$$

Consequently, we define

$$\begin{array}{rcl} B_{\mathcal{I}} = B_{\mathcal{RT}} & = & \mathcal{P}_\omega(\mathcal{P}_\omega A) \\ o(Y) & = & \bigcup_{y \in Y} \bar{o}_{\mathcal{RT}}(\bar{\delta}(y)) \\ t(Y)(\langle a, S\rangle) & = & \bigcup_{y \in Y} \bar{\delta}(y)(\langle a, S\rangle). \end{array}$$

Proving that the coalgebraic modelling of ready trace semantics coincides with the original definition in (van Glabbeek 2001) consists in showing that for $x_0 \in X$, there is a one-to-one correspondence between $[\![\{x_0\}]\!]$ and $\mathcal{RT}(x_0)$. Intuitively, each behaviour

$$[\![\{x_0\}]\!](\bar{w}) = \bigcup_{j \in J} \{I_n^j\}, \text{ where } \bar{w} = \langle a_1, I_0\rangle \ \ldots \ \langle a_n, I_{n-1}\rangle \in (\bar{A})^*$$

corresponds to a set of ready traces of shape

$$I_0 a_1 I_1 a_2 \ldots I_{n-1} a_n I_n^j \in \mathcal{RT}(x_0)$$

such that

$$\begin{array}{l} (\exists x_1, \ldots, x_n \in X) \,.\, x_0 \xrightarrow{a_1} x_1 \xrightarrow{a_2} \ldots \xrightarrow{a_n} x_n \ \wedge \\ (\forall i = 1, \ldots, n-1) \,.\, I_i = I(\delta(x_i)) \ \wedge \\ I_n^j = I(\delta(x_n)). \end{array}$$

And similarly the other way around.

Given a state $x \in X$ we choose to represent $\mathcal{RT}(x) \in \mathcal{P}(\mathcal{P}_\omega(A) \times (A \times \mathcal{P}_\omega(A))^*) = \mathcal{P}(\mathcal{P}_\omega(A) \times (\bar{A})^*)$ in terms of a function $\varphi_{\{x\}}^{\mathcal{RT}}$ such that

$$\begin{array}{l} \varphi_Y^{\mathcal{RT}} \colon (\bar{A})^* \to \mathcal{P}_\omega(\mathcal{P}_\omega A) \\ \varphi_Y^{\mathcal{RT}}(\bar{w}) = \{Z \subseteq A \mid \exists y \in t(Y)(\bar{w}) \wedge Z = I(\delta(y))\} \end{array}$$

(note that $\mathcal{P}(\mathcal{P}_\omega(A) \times (\bar{A})^*)$ and $(\mathcal{P}_\omega(\mathcal{P}_\omega A))^{(\bar{A})^*}$ are isomorphic structures).

At this point, showing the equivalence between the coalgebraic and the original definition of ready trace semantics consists in proving that

$$(\forall x \in X) \,.\, [\![\{x\}]\!] = \varphi_{\{x\}}^{\mathcal{RT}}. \tag{16}$$

Equality (17) is a direct consequence of the following theorem:

Theorem 4.6. Let $(X, \delta \colon X \to (\mathcal{P}_\omega X)^A)$ be an LTS and $(X, \bar{\delta} \colon X \to (\mathcal{P}_\omega X)^{\bar{A}})$ the corresponding LTS generated according to the "preprocessing" procedure. Then for all $Y \subseteq X$ and $\bar{w} \in (\bar{A})^*$, $[\![Y]\!](\bar{w}) = \varphi_Y^{\mathcal{RT}}(\bar{w})$.

Proof. The proof follows by induction on words $\bar{w} \in (\bar{A})^*$ (in the same fashion with the proof of Theorem 4.1). □

Example 4.8. Consider the following two systems:

$$
\begin{tikzcd}[column sep=small]
 & & p_0 \arrow[dl, "a"'] \arrow[dr, "a"] & & \\
 & p_1 \arrow[dl, "b"'] \arrow[d, "c"'] & & p_2 \arrow[d, "c"'] \arrow[dr, "f"] & \\
p_3 & p_4 \arrow[d, "d"'] & & p_5 \arrow[d, "e"'] & p_6 \\
 & p_7 & & p_8 &
\end{tikzcd}
\qquad
\begin{tikzcd}[column sep=small]
 & & q_0 \arrow[dl, "a"'] \arrow[dr, "a"] & & \\
 & q_1 \arrow[dl, "b"'] \arrow[d, "c"'] & & q_2 \arrow[d, "c"'] \arrow[dr, "f"] & \\
q_3 & q_4 \arrow[d, "e"'] & & q_5 \arrow[d, "d"'] & q_6 \\
 & q_7 & & q_8 &
\end{tikzcd}
$$

Note that they are not ready trace equivalent as, for example, $\{a\}a\{c,f\}c\{e\}$ is a ready trace of p_0 but not of q_0.

By assuming only the generalized powerset construction (starting with $\{p_0\},\{q_0\}$) as in Section 4.1, without the "preprocessing" step, one gets the following (obviously) bisimilar Moore automata:

$$
\begin{tikzcd}
 & & \{p_0\} \arrow[r, Rightarrow] \arrow[d, "a"'] & \{\{a\}\} & \\
 & & \{p_1,p_2\} \arrow[r, Rightarrow] \arrow[dl, "b"'] \arrow[d, "c"'] \arrow[dr, "f"] & \{\{b,c\},\{c,f\}\} & \\
\{\emptyset\} & \{p_3\} \arrow[l, Rightarrow] & \{p_4,p_5\} \arrow[dl, "d"'] \arrow[d, "e"'] \arrow[dr, Rightarrow] & \{p_6\} \arrow[r, Rightarrow] & \{\emptyset\} \\
 & \{p_7\} \arrow[d, Rightarrow] & \{p_8\} \arrow[d, Rightarrow] & \{\{d\},\{e\}\} & \\
 & \{\emptyset\} & \{\emptyset\} & &
\end{tikzcd}
$$

$$
\begin{tikzcd}
 & & \{q_0\} \arrow[r, Rightarrow] \arrow[d, "a"'] & \{\{a\}\} & \\
 & & \{q_1,q_2\} \arrow[r, Rightarrow] \arrow[dl, "b"'] \arrow[d, "c"'] \arrow[dr, "f"] & \{\{b,c\},\{c,f\}\} & \\
\{\emptyset\} & \{q_3\} \arrow[l, Rightarrow] & \{q_4,q_5\} \arrow[dl, "d"'] \arrow[d, "e"'] \arrow[dr, Rightarrow] & \{q_6\} \arrow[r, Rightarrow] & \{\emptyset\} \\
 & \{q_8\} \arrow[d, Rightarrow] & \{q_7\} \arrow[d, Rightarrow] & \{\{d\},\{e\}\} & \\
 & \{\emptyset\} & \{\emptyset\} & &
\end{tikzcd}
$$

which would indicate that the initial LTS's are behavioural equivalent (which is false!).

The preprocessing of p_0, q_0 generates the automata (with actions in $\bar{A} = A \times \mathcal{P}_\omega(A)$) in Fig. 11. The determinization would therefore derive the two Moore automata in Fig. 12.

Note that the systems in Fig. 12 are not behaviourally equivalent as, for example, both states $\{p_4\}$ and $\{q_4\}$ can be reached via transitions labelled the same, but they output different sets of ready actions – namely $\{\{d\}\}$ and $\{\{e\}\}$, respectively. Therefore we conclude that p_0 and q_0 are not ready trace equivalent.

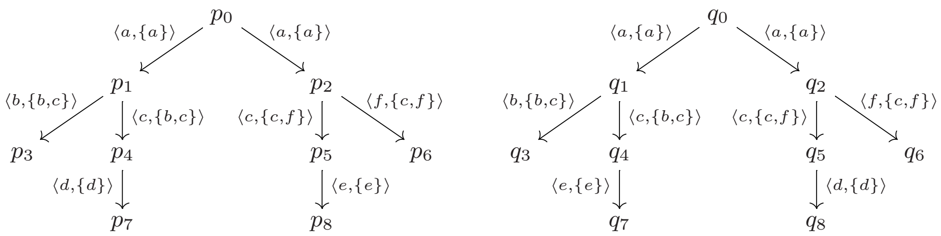
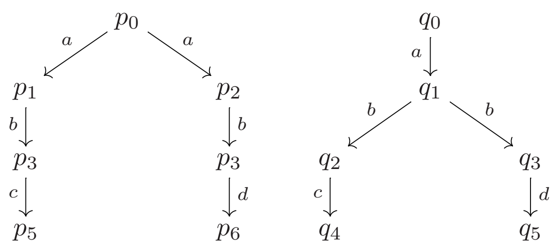


Fig. 11. Preprocessed versions of p_0, q_0.

The purpose of enriching the transition labels with sets of ready actions is to collect in a Moore state only states of the initial LTS's that have been reached from "parents" with the same one-step (initial) behaviour. Or dually, to distinguish between states that have "parents" ready to trigger different sets of actions. This way one avoids the unfortunate situation of encapsulating, for example, the states p_4, p_5, respectively q_4, q_5, fact which eventually would lead to providing a positive answer with respect to the ready trace equivalence of p_0 and q_0.

In other words, the preprocessing step is one of the ingredients needed in order to guarantee that whenever two states of an LTS are ready trace equivalent, the (pairwise-taken) states determined by the executions of a given trace have the same initial behaviour.

Example 4.9. In what follows we show an example on how our framework can be used in order to reason on ready trace equivalence. Assume the following two systems:



Observe that p_0 and q_0 are ready trace equivalent, as:

$$\begin{aligned}\mathcal{RT}(p_0) = \mathcal{RT}(q_0) = \{ \quad & \{a\}, \{a\}a\{b\}, \{a\}a\{b\}b\{c\}, \\ & \{a\}a\{b\}b\{d\}, \{a\}a\{b\}b\{c\}c\,\emptyset, \{a\}a\{b\}b\{d\}d\,\emptyset \,\}.\end{aligned}$$

It is straightforward to check that the Moore automata (starting with $\{p_0\}, \{q_0\}$) in Fig. 13, derived from the corresponding "preprocessed" versions of the initial LTS's, are bisimilar. Therefore, p_0 and q_0 are ready trace equivalent.

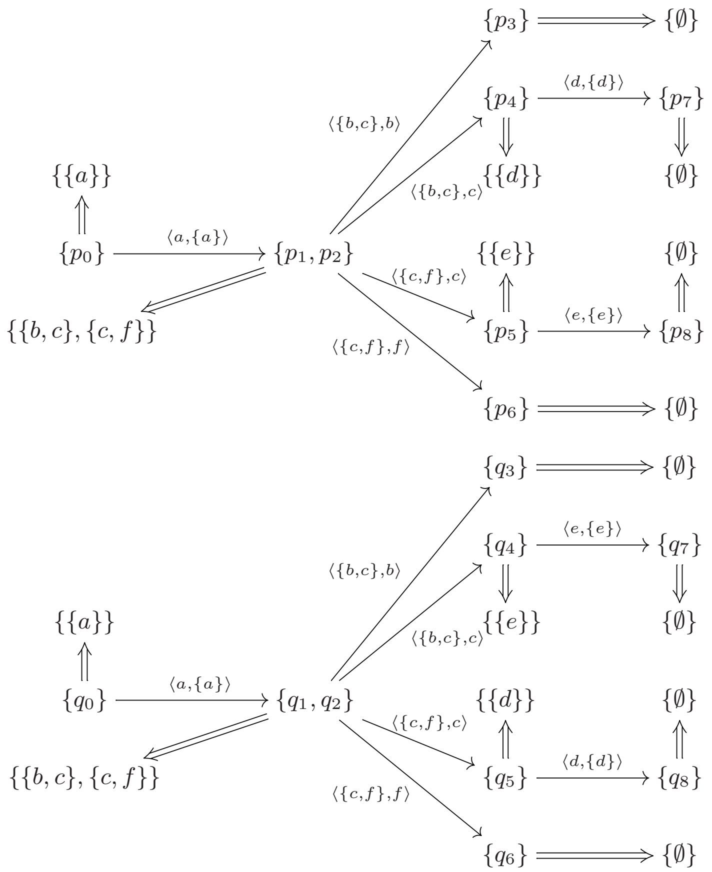


Fig. 12. Determinization of the preprocessed LTS's starting from $\{p_0\},\{q_0\}$.

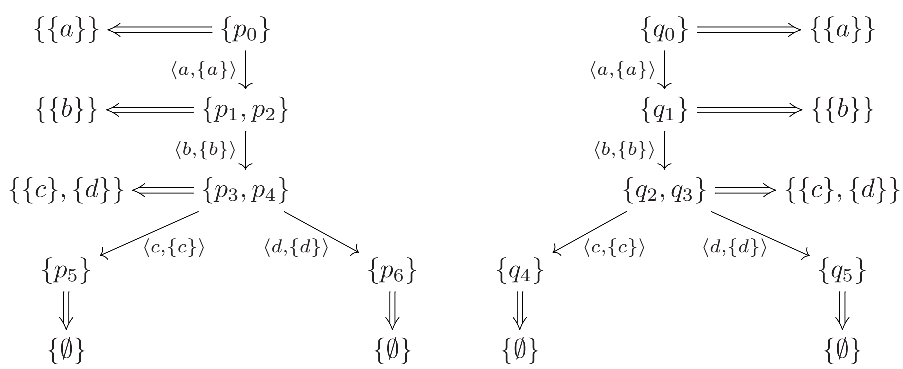


Fig. 13. Determinization of the preprocessed LTS's, starting from $\{p_0\},\{q_0\}$.

4.7. *Failure trace semantics*

In this section we provide a coalgebraic modelling for the last semantics of our suite – namely, failure trace semantics. We also show the equivalence between the coalgebraic representation and the original definition in (van Glabbeek 2001), and give examples on how to use the framework for reasoning on failure traces.

Intuitively, failure trace semantics identifies states that can trigger the same traces w, and moreover, the (pairwise-taken) intermediate states occurring during the execution of a such w fail triggering the same (sets of) actions.

Formally, this can be concluded from Corollary 5.1 in (van Glabbeek 2001), as follows:

Definition 4.7 ($\mathcal{FT}$-equivalence). Let $(X, \delta\colon X \to (\mathcal{P}_\omega X)^A)$ be an LTS and $x, y \in X$ two states. States x and y are *failure trace equivalent* ($\mathcal{FT}$-equivalent) if and only if $\mathcal{FT}(x) = \mathcal{FT}(y)$, where

$$\begin{aligned} \mathcal{FT}(x) = \{ \quad & F_0 a_1 F_1 a_2 \ldots a_n F_n \in \mathcal{P}_\omega(A) \times (A \times \mathcal{P}_\omega(A))^* \mid \\ & (\exists x_1, \ldots, x_n \in X) \,.\, x_0 \xrightarrow{a_1} x_1 \xrightarrow{a_2} \ldots \xrightarrow{a_n} x_n \wedge \\ & (\forall i = 1, \ldots, n) \,.\, F_i \in Fail(\delta(x_i)) \ \}. \end{aligned}$$

We call an element of $\mathcal{FT}(x)$ a *failure trace* of x.

Similarly to the case of ready trace semantics (see Section 4.6), each LTS $(X, \delta\colon X \to (\mathcal{P}_\omega X)^A)$ is uniquely associated a coalgebra $(X, \bar{\delta}\colon X \to (\mathcal{P}_\omega X)^{\bar{A}})$, where

$$\begin{aligned} \bar{A} \quad &= \quad A \times \mathcal{P}_\omega(\mathcal{P}_\omega(A)) \\ \bar{\delta}(x)(\langle a, F_x \rangle) \quad &= \quad \begin{cases} \delta(x)(a) & \text{if } F_x = Fail(\delta(x)) \\ \emptyset & \text{otherwise} \end{cases} \end{aligned}$$

Note that in the setting of failure trace semantics, the "preprocessing" above lifts transitions of shape $x \xrightarrow{a} y$ (in the initial LTS) to transitions $x \xrightarrow{\langle a, F_x \rangle} y$, where $F_x = Fail(\delta(x))$. This way, each ("structured") state of the Moore automata derived according to the generalized powerset construction will consist of states (of the initial LTS) that have been reached not only via one-step transitions labelled the same, but also from processes sharing the same sets of failure actions. This will eventually guarantee a sound extension of failure trace equivalence of states in X (as in Definition 4.7) to equivalence of "structured" states in $\mathcal{P}_\omega X$ of the Moore systems.

From this point onwards, the coalgebraic modelling of failure traces follows the pattern of the previous sections.

The ingredients of Fig. 2 are instantiated as follows. We start with the "preprocessed" system $(X, \bar{\delta}\colon X \to (\mathcal{P}_\omega X)^{\bar{A}})$ and define the output function $\bar{o}_{\mathcal{I}} = \bar{o}_{\mathcal{FT}}$ revealing information with respect to the actions refused to be triggered by a state $x \in X$ as a first step:

$$\begin{aligned} & \bar{o}_{\mathcal{FT}}\colon (\mathcal{P}_\omega X)^{\bar{A}} \to \mathcal{P}_\omega(\mathcal{P}_\omega A) \\ & \bar{o}_{\mathcal{FT}}(\bar{\delta}(x)) = Fail(\delta(x)) = \{Z \subseteq A \mid Z \cap I(\delta(x)) = \emptyset\}. \end{aligned}$$

We further define

$$\begin{array}{rcl} B_{\mathcal{I}} = B_{\mathcal{FT}} & = & \mathcal{P}_\omega(\mathcal{P}_\omega A) \\ o(Y) & = & \bigcup_{y \in Y} \bar{o}_{\mathcal{FT}}(\bar{\delta}(y)) \\ t(Y)(\langle a, S\rangle) & = & \bigcup_{y \in Y} \bar{\delta}(y)(\langle a, S\rangle). \end{array}$$

Next we show that the coalgebraic modelling of failure trace semantics coincides with the original definition in (van Glabbeek 2001), by exploiting the one-to-one correspondence between $[\![\{-\}]\!]$ and $\mathcal{FT}(-)$. Given a state $x \in X$ we represent $\mathcal{FT}(x) \in \mathcal{P}(\mathcal{P}_\omega(A) \times (A \times \mathcal{P}_\omega(A))^*)$ by means of $\varphi^{\mathcal{FT}}_{\{x\}}$, where

$$\begin{array}{l} \varphi^{\mathcal{FT}}_Y \colon (\bar{A})^* \to \mathcal{P}_\omega(\mathcal{P}_\omega A) \\ \varphi^{\mathcal{FT}}_Y(\bar{w}) = \{Z \subseteq A \mid \exists y \in t(Y)(\bar{w}) \wedge Z \in \mathit{Fail}(\delta(y))\} \end{array}$$

(recall that $\bar{A} = A \times \mathcal{P}_\omega(\mathcal{P}_\omega(A))$, and note that $\mathcal{P}(\mathcal{P}_\omega(A) \times (A \times \mathcal{P}_\omega(A))^*)$ and $(\mathcal{P}_\omega(\mathcal{P}_\omega A))^{(\bar{A})^*}$ are isomorphic structures). Therefore we have to show that

$$(\forall x \in X)\,.\,[\![\{x\}]\!] = \varphi^{\mathcal{RT}}_{\{x\}}. \tag{17}$$

Equality (17) is a direct consequence of the following theorem:

Theorem 4.7. Let $(X, \delta \colon X \to (\mathcal{P}_\omega X)^A)$ be an LTS and $(X, \bar{\delta} \colon X \to (\mathcal{P}_\omega X)^{\bar{A}})$ the corresponding LTS generated according to the "preprocessing" procedure. Then for all $Y \subseteq X$ and $\bar{w} \in (\bar{A})^*$, $[\![Y]\!](\bar{w}) = \varphi^{\mathcal{FT}}_Y(\bar{w})$.

Example 4.10. Consider again the automata in Example 4.8 in Section 4.6:

$$\begin{tikzcd} & & p_0 \arrow[dl, "a"'] \arrow[dr, "a"] & & \\ & p_1 \arrow[dl, "b"'] \arrow[d, "c"'] & & p_2 \arrow[d, "c"'] \arrow[dr, "f"] & \\ p_3 & p_4 \arrow[d, "d"'] & & p_5 \arrow[d, "e"'] & p_6 \\ & p_7 & & p_8 & \end{tikzcd} \qquad \begin{tikzcd} & & q_0 \arrow[dl, "a"'] \arrow[dr, "a"] & & \\ & q_1 \arrow[dl, "b"'] \arrow[d, "c"'] & & q_2 \arrow[d, "c"'] \arrow[dr, "f"] & \\ q_3 & q_4 \arrow[d, "e"'] & & q_5 \arrow[d, "d"'] & q_6 \\ & q_7 & & q_8 & \end{tikzcd}$$

Observe that they are not failure trace equivalent as, for example, $\{b, c, d, e, f\}a\{a, d, e, f\}c\{a, b, c, e, f\}d\{a, b, c, d, e, f\}$ is a failure trace of p_0 but not of q_0.

We further show that our framework provides the same answer with respect to the equivalence of p_0 and q_0.

First, recall from Section 4.2 that for the simplicity of notation, we write $[a_1 a_2 \ldots a_n]$ to denote the set of all non-empty subsets $Z \subseteq A$, where $A = \{a_1, a_2, \ldots, a_n\}$.

The "preprocessed" LTS's corresponding to p_0, q_0 are depicted in Fig. 14.

Consequently, the associated Moore automata (starting with $\{p_0\}, \{q_0\}$) derived according to the generalized powerset construction are as follows are as illustrated in Fig. 15. It is easy to see that the automata in Fig. 15 are not bisimilar as, for example, both $\{p_4\}$ and $\{q_4\}$ are reached via transitions labelled the same, but have different outputs. Therefore we conclude that p_0 and q_0 are not failure trace equivalent.

Note that the generalized powerset construction applied on the initial not "prepro-

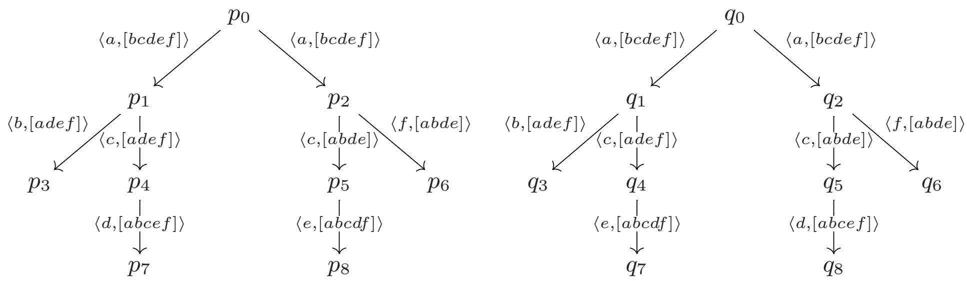


Fig. 14. Preprocessed versions of p_0, q_0.

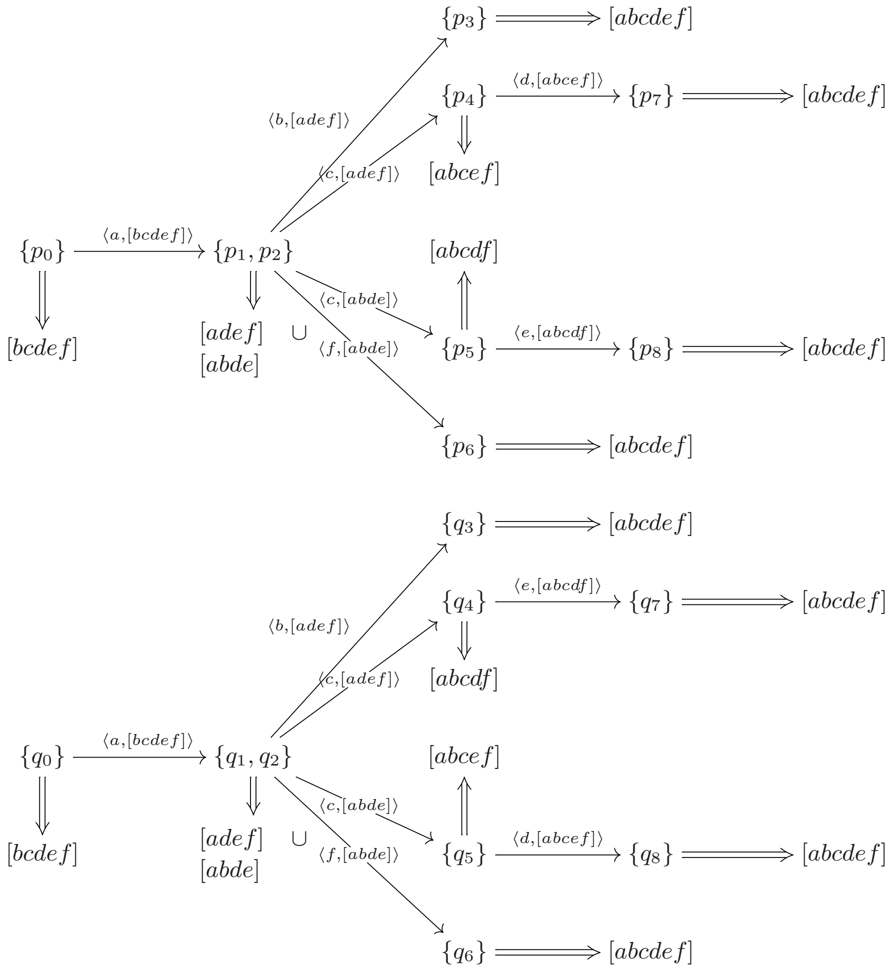


Fig. 15. Determinization of the preprocessed LTS's starting from $\{p_0\}, \{q_0\}$.

cessed" LTS's p_0, q_0 would derive two bisimilar Moore automata, similarly to the case of ready trace equivalence (in Section 4.6), with the only difference that the output function is defined as $o(Y) = \bigcup_{y \in Y} Fail(\delta(y))$ (instead of $o(Y) = \bigcup_{y \in Y} \{I(\delta(y))\}$).

In a nutshell. Next we provide a more compact overview on the coalgebraic machineries introduced in Section 4.1–Section 4.7. This also in order to emphasize on the generality and uniformity of our coalgebraic framework.

Recall that for each of the decorated trace semantics we first instantiate the constituents of Fig. 2 (summarizing the generalized powerset construction). Also, recall that for ready trace and failure trace a preprocessing of the original LTS's was required before applying the determinization procedure. All this, together with the definitions of functions $\varphi_Y^{\mathcal{I}}$ equivalently capturing the set-theoretic characterizations of all the semantics under consideration are illustrated in Fig. 16, for an arbitrary LTS $(X, \delta \colon X \to (\mathcal{P}_\omega X)^A)$.

Once the ingredients of Fig. 2 and $\varphi_Y^{\mathcal{I}}$ are defined, we formalize the equivalence between the coalgebraic modelling of $\mathcal{I}$-semantics (for $\mathcal{I}$ ranging over $\mathcal{T}, \mathcal{CT}, \mathcal{F}, \mathcal{R}, \mathcal{PF}, \mathcal{RT}$ and $\mathcal{FT}$) and its original definition in (van Glabbeek 2001) in terms of the (generic) Theorem 4.8.

Theorem 4.8. Let $(X, \delta \colon X \to (\mathcal{P}_\omega X)^A)$ be an LTS. [For ready trace and failure trace: let $(X, \bar{\delta} \colon X \to (\mathcal{P}_\omega X)^{\bar{A}})$ be the corresponding LTS generated according to the "preprocessing" procedure.] Then for all $Y \subseteq X$ and $w \in A^*$ [for ready trace and failure trace: $w \in (\bar{A})^*$], $[\![Y]\!](w) = \varphi_Y^{\mathcal{I}}(w)$.

For each of the semantics under consideration, the proof of Theorem 4.8 follows by induction on words $w \in A^*$ (respectively $w \in (\bar{A})^*$). For more details see the proof of Theorem 4.1 in Section 4.1.

Concrete examples on how to use the coalgebraic frameworks are provided for each of the decorated trace semantics. We show how to (apply the preprocessing procedure and) derive determinizations of LTS's in terms of Moore automata, which eventually are used to reason on the corresponding equivalences in terms of Moore bisimulations (up-to).

5. Canonical representatives

Given a *decorated* LTS $(X, \langle \overline{o}_{\mathcal{I}}, id \rangle \circ \delta)$, we showed in the previous section how to construct a *determinized decorated* LTS $(\mathcal{P}_\omega X, \langle o, t \rangle)$. The map $[\![-]\!] \colon \mathcal{P}_\omega X \to B_{\mathcal{I}}^{A^*}$ provides us with a *canonical representative* of the behaviour of each state in $\mathcal{P}_\omega X$. (A represents the – possibly enriched – action alphabet.) The image (C, δ') of $(\mathcal{P}_\omega X, \langle o, t \rangle)$, via the map $[\![-]\!]$, can be viewed as the minimization w.r.t. the equivalence $\mathcal{I}$.

Recall that the states of the final coalgebra $(B_{\mathcal{I}}^{A^*}, \langle \epsilon, (-)_a \rangle)$ are functions $\varphi \colon A^* \to B_{\mathcal{I}}$ and that their decorations and transitions are given by the functions $\epsilon \colon B_{\mathcal{I}}^{A^*} \to B_{\mathcal{I}}$ and $(-)_a \colon B_{\mathcal{I}}^{A^*} \to (B_{\mathcal{I}}^{A^*})^A$, defined in Section 2. The states of the canonical representative (C, δ') are also functions $\varphi \colon A^* \to B_{\mathcal{I}}$, i.e., $C \subseteq B_{\mathcal{I}}^{A^*}$. Moreover, the function $\delta' \colon C \to B_{\mathcal{I}} \times C^A$ is simply the restriction of $\langle \epsilon, (-)_a \rangle$ to C, that means $\delta'(\varphi) = \langle \varphi(\epsilon), (\varphi)_a \rangle$ for all $\varphi \in C$.

$\mathcal{I}$	preprocessing	$B_{\mathcal{I}}$	$\bar{o}_{\mathcal{I}}$	$\varphi_Y^{\mathcal{I}}$
$\mathcal{R}$	no	$\mathcal{P}_\omega(\mathcal{P}_\omega A)$	$\bar{o}_{\mathcal{R}} \colon (\mathcal{P}_\omega X)^A \to \mathcal{P}_\omega(\mathcal{P}_\omega A)$ $\bar{o}_{\mathcal{R}}(\varphi) = \{I(\varphi)\}$	$\varphi_Y^{\mathcal{R}} \colon A^* \to \mathcal{P}_\omega(\mathcal{P}_\omega A)$ $\varphi_Y^{\mathcal{R}}(w) = \{Z \subseteq A \mid \exists y \in t(Y)(w) \wedge Z = I(\delta(y))\}$
$\mathcal{F}$	no	$\mathcal{P}_\omega(\mathcal{P}_\omega A)$	$\bar{o}_{\mathcal{F}} \colon (\mathcal{P}_\omega X)^A \to \mathcal{P}_\omega(\mathcal{P}_\omega A)$ $\bar{o}_{\mathcal{F}}(\varphi) = Fail(\varphi)$	$\varphi_Y^{\mathcal{F}} \colon A^* \to \mathcal{P}_\omega(\mathcal{P}_\omega A)$ $\varphi_Y^{\mathcal{F}}(w) = \{Z \subseteq A \mid \exists y \in t(Y)(w) \wedge Z \in Fail(\delta(y))\}$
$\mathcal{T}$	no	2	$\bar{o}_{\mathcal{T}} \colon (\mathcal{P}_\omega X)^A \to 2$ $\bar{o}_{\mathcal{T}}(\varphi) = 1$	$\varphi_Y^{\mathcal{T}} \colon A^* \to 2$ $\varphi_Y^{\mathcal{T}}(w) = 1$ if $(\exists y \in Y, y' \in X) \,.\, y \xrightarrow{w} y'$
$\mathcal{CT}$	no	2	$\bar{o}_{\mathcal{CT}} \colon (\mathcal{P}_\omega X)^A \to 2$ $\bar{o}_{\mathcal{CT}}(\varphi) = \begin{cases} 1 & \text{if } I(\varphi) = \emptyset \\ 0 & \text{otherwise} \end{cases}$	$\varphi_Y^{\mathcal{CT}} \colon A^* \to 2$ $\varphi_Y^{\mathcal{CT}}(w) = \begin{cases} 1 & \text{if } (\exists y \in t(Y)(w)) \,.\, I(\delta(y)) = \emptyset \\ 0 & \text{otherwise} \end{cases}$
$\mathcal{PF}$	no	$\mathcal{P}_\omega(\mathcal{P}A^*)$	$\bar{o}_{\mathcal{PF}} \colon (\mathcal{P}_\omega X)^A \to \mathcal{P}_\omega(\mathcal{P}A^*)$ $\bar{o}_{\mathcal{PF}}(\delta(x)) = \mathcal{T}(x)$	$\varphi_Y^{\mathcal{PF}} \colon A^* \to \mathcal{P}_\omega(\mathcal{P}A^*)$ $\varphi_Y^{\mathcal{R}}(w) = \{\mathcal{T}(y) \mid y \in t(Y)(w)\}$
$\mathcal{RT}$	yes $\bar{A} = A \times \mathcal{P}_\omega(A)$ $\bar{\delta}(x)(\langle a, I_x \rangle) =$ $\begin{cases} \delta(x)(a) & \text{if } I_x = I(\delta(x)) \\ \emptyset & \text{otherwise} \end{cases}$	$\mathcal{P}_\omega(\mathcal{P}_\omega A)$	$\bar{o}_{\mathcal{RT}} \colon (\mathcal{P}_\omega X)^{\bar{A}} \to \mathcal{P}_\omega(\mathcal{P}_\omega A)$ $\bar{o}_{\mathcal{RT}}(\bar{\delta}(x)) = \{I(\delta(x))\}$	$\varphi_Y^{\mathcal{RT}} \colon (\bar{A})^* \to \mathcal{P}_\omega(\mathcal{P}_\omega A)$ $\varphi_Y^{\mathcal{RT}}(\bar{w}) = \{Z \subseteq A \mid \exists y \in t(Y)(\bar{w}) \wedge Z = I(\delta(y))\}$
$\mathcal{FT}$	yes $\bar{A} = A \times \mathcal{P}_\omega(\mathcal{P}_\omega(A))$ $\bar{\delta}(x)(\langle a, F_x \rangle) =$ $\begin{cases} \delta(x)(a) & \text{if } F_x = Fail(\delta(x)) \\ \emptyset & \text{otherwise} \end{cases}$	$\mathcal{P}_\omega(\mathcal{P}_\omega A)$	$\bar{o}_{\mathcal{FT}} \colon (\mathcal{P}_\omega X)^{\bar{A}} \to \mathcal{P}_\omega(\mathcal{P}_\omega A)$ $\bar{o}_{\mathcal{FT}}(\bar{\delta}(x)) = Fail(\delta(x))$	$\varphi_Y^{\mathcal{FT}} \colon (\bar{A})^* \to \mathcal{P}_\omega(\mathcal{P}_\omega A)$ $\varphi_Y^{\mathcal{FT}}(\bar{w}) = \{Z \subseteq A \mid \exists y \in t(Y)(\bar{w}) \wedge Z \in Fail(\delta(y))\}$

Fig. 16. The coalgebraic framework in a nutshell.

Finally, it is interesting to observe that $B_{\mathcal{I}}^{A^*}$ carries a semilattice structure (inherited by $B_{\mathcal{I}}$) and that $[\![-]\!] \colon \mathcal{P}_\omega X \to B_{\mathcal{I}}^{A^*}$ is a semilattice homomorphism. From this observation, it is immediate to conclude that also C is a semilattice, but it is not necessarily freely generated, i.e., it is not necessarily a powerset.

6. Conclusions and future work

In this paper, we have proved that the coalgebraic characterizations of ready, failure, (complete) trace, possible-futures, ready trace and failure trace semantics are equivalent with the corresponding standard definitions. More precisely, we have shown that for a state x in a labelled transition system, the coalgebraic canonical representative $[\![\{x\}]\!]$, given by determinization and finality, coincides with the classical semantics $\mathcal{I}(x)$, for $\mathcal{I}$ ranging over $\mathcal{T}, \mathcal{CT}, \mathcal{F}, \mathcal{R}, \mathcal{PF}, \mathcal{RT}$ and $\mathcal{FT}$, representing the traces, complete traces, ready pairs, failure pairs, possible futures, ready traces and respectively failure traces of x. In addition, we have illustrated how to reason about decorated trace equivalence using coinduction, by constructing suitable bisimulations up-to context. This is a very efficient sound and complete proof technique, and represents an important step towards automated reasoning, as it opens the way for the use of, for instance, coinductive theorem provers such as CIRC (Roşu & Lucanu 2009).

A similar idea of system determinization was also applied in (Cleaveland & Hennessy 1993), in a non-coalgebraic setting, for the case of testing semantics where *must testing* coincides with failure semantics in the absence of divergence. A coalgebraic characterization of the spectrum was also attempted in (Monteiro 2008), in a somewhat *ad hoc* fashion. Connections with these works are still to be explored.

There are several possible directions for future works. One option is to investigate to what extent the coalgebraic treatment of decorated trace semantics can be applied in the context of probabilistic systems.

We would also like to understand how our approach can be combined with (Boreale & Gadducci 2006) to obtain a coinductive approach to denotational (linear-time) semantics of different kinds of processes calculi.

Nevertheless, it is worth mentioning our intention of providing coalgebraic modellings for the remaining semantics of the spectrum in (van Glabbeek 2001), and maybe come up with a new representation of possible-futures semantics. The latter is motivated by the current drawback of storing for each state of the LTS's the corresponding set of traces. In this context it might be more appropriate considering the definition of possible-futures semantics given in terms of nested bisimulations (Hennessy & Milner 1985), rather than the set-theoretic one in (van Glabbeek 2001).

References

Aceto, L., Fokkink, W. & Verhoef, C. (1999), Structural operational semantics, *in* ‘Handbook of Process Algebra’, Elsevier, pp. 197–292.

Awodey, S. (2010), *Category theory*, Oxford Logic Guides, Oxford University Press.

Bonchi, F., Bonsangue, M., Caltais, G., Rutten, J. & Silva, A. (2012), Final semantics for decorated traces. To appear in ENTCS.

Bonchi, F. & Pous, D. (2013), Checking NFA equivalence with bisimulations up to congruence. To appear in POPL.

Boreale, M. & Gadducci, F. (2006), ‘Processes as formal power series: A coinductive approach to denotational semantics’, *Theor. Comput. Sci.* **360**(1-3), 440–458.

Cleaveland, R. & Hennessy, M. (1993), ‘Testing equivalence as a bisimulation equivalence’, *Formal Asp. Comput.* **5**(1), 1–20.

Hasuo, I., Jacobs, B. & Sokolova, A. (2007), ‘Generic trace semantics via coinduction’, *Logical Methods in Computer Science* **3**(4).

Hennessy, M. & Milner, R. (1985), ‘Algebraic laws for nondeterminism and concurrency’, *J. ACM* **32**(1), 137–161.
URL: *http://doi.acm.org/10.1145/2455.2460*

Milner, R. (1989), *Communication and concurrency*, Prentice-Hall international series in computer science, Prentice Hall.

Monteiro, L. (2008), A coalgebraic characterization of behaviours in the linear time - branching time spectrum, *in* A. Corradini & U. Montanari, eds, 'WADT', Vol. 5486 of *Lecture Notes in Computer Science*, Springer, pp. 251–265.

Park, D. M. R. (1981), Concurrency and automata on infinite sequences, *in* P. Deussen, ed., 'Theoretical Computer Science', Vol. 104 of *Lecture Notes in Computer Science*, Springer, pp. 167–183.

Roşu, G. & Lucanu, D. (2009), Circular Coinduction – A Proof Theoretical Foundation, *in* 'CALCO'09', LNCS.

Rutten, J. J. M. M. (2000), 'Universal coalgebra: a theory of systems', *Theor. Comput. Sci.* **249**(1), 3–80.

Sangiorgi, D. & Rutten, J. (2011), *Advanced Topics in Bisimulation and Coinduction*, Cambridge Tracts in Theoretical Computer Science, Cambridge University Press.

Silva, A., Bonchi, F., Bonsangue, M. M. & Rutten, J. J. M. M. (2010), Generalizing the powerset construction, coalgebraically, *in* K. Lodaya & M. Mahajan, eds, 'FSTTCS 2010', Vol. 8 of *LIPIcs*, pp. 272–283.
URL: *http://drops.dagstuhl.de/opus/volltexte/2010/2870*

Silva, A., Bonchi, F., Bonsangue, M. & Rutten, J. (2011), 'Generalizing determinization from automata to coalgebras'. Submitted.

van Glabbeek, R. (2001), The linear time - branching time spectrum I. The semantics of concrete, sequential processes, *in* J. Bergstra, A. Ponse & S. Smolka, eds, 'Handbook of Process Algebra', Elsevier, pp. 3–99.