Journal of Universal Computer Science, vol. 12, no. 11 (2006), 1521-1550
submitted: 1/5/06, accepted: 15/10/06, appeared: 28/11/06 © J.UCS

Process Equivalences as Global Bisimulations[1]

David de Frutos Escrig
Department of Sistemas Informáticos y Programación
Universidad Complutense de Madrid, Spain
defrutos@sip.ucm.es

Carlos Gregorio Rodríguez
Department of Sistemas Informáticos y Programación
Universidad Complutense de Madrid, Spain
cgr@sip.ucm.es

Abstract: Bisimulation can be defined in a simple way using coinductive methods, and has rather pleasant properties. Ready similarity was proposed by Meyer et al. as a way to weakening the bisimulation equivalence thus getting a semantics defined in a similar way, but supported for more reasonable (weaker) observational properties.

Global bisimulations were introduced by Frutos et al. in order to study different variants of non-determinism getting, in particular, a semantics under which the internal choice operator becomes associative. Global bisimulations are defined as plain bisimulations but allowing the use of new moves, called global transitions, that can change the processes not only locally in its head, but anywhere.

Now we are continuing the study of global bisimulation but focusing on the way different semantics can be characterised as global bisimulation semantics. In particular, we have studied ready similarity, on the one hand because it was proposed as the strongest reasonable semantics weaker than bisimulation; on the other hand, because ready similarity was not directly defined as an equivalence relation but as the nucleus of an order relation, and this open the question whether it is also possible to define it as a symmetric bisimulation-like semantics.

We have got a simple and elegant characterisation of ready similarity as a global bisimulation semantics, that provides a direct symmetric characterisation of it as an equivalence relation, without using any order as intermediate concept. Besides, we have found that it is not necessary to start from a simulation based semantics to get an equivalent global bisimulation. What has proved to be very useful is the axiomatic characterisation of the semantics. Following these ideas we have got also global bisimulation for several semantics, including refusals and traces. That provides a general framework that allows to relate both intensional and extensional semantics.

Key Words: bisimulation, ready simulation, refusal, traces, concurrent process equivalences and preorders, comparative semantics.
Category: F.1.2, F.3.2, D.3.1

1 Introduction

A cornerstone for the *Theory of Concurrent Processes* is to determine whether two processes are equivalent, that is, whether the *behaviour* of two given processes

[1] Partially supported by the projects TERMAS TIC2003-07848-C02-01, MIDAS TIC2003-01000, PAC-03-001 and MRTN-CT-2003-505121/TAROT

is essentially the same.

There exist in the literature many different proposals defining process equivalences. Park and Milner's bisimulation equivalence [Park, 1981, Milner, 1989] has been recognised as a fundamental notion in the theory of concurrency and has set up by itself a large and fertile field of study. Bisimulation is a mathematically elegant concept, it is recursively defined over the intensional definition of processes, and captures some (relatively) natural logical properties of the processes.

However, several objections have been made to this equivalence relation on the grounds of its excessive discriminatory power. Following similar formulations to that of bisimulation other weaker equivalences have been proposed, amongst them 2-nested simulation equivalence [Groote and Vaandrager, 1992] and ready simulation equivalence [Bloom et al., 1995], also called ready similarity, that was independently presented in a different context, with the name of $\frac{2}{3}$-bisimulation in [Larsen and Skou, 1991]

Ready simulation equivalence, enjoys many interesting properties: it is the finest equivalence that is a congruence with respect to all the languages defined in GSOS format; has a characterisation in terms of a modal logic; and, as there have been largely argued, it has a characterisation, in terms of testing or button-pushing scenario, that is more reasonable than that necessary to characterise the bisimulation equivalence.

These relations provide further identification of processes, but they are still quite far from the identification power of the usual equivalence relations based on extensional models. Extensional, or denotational, models provide semantic equivalences in a quite different way to bisimulation. Processes are represented in terms of mathematical objects that are defined in terms of the observations made on the behaviour of each process. Trace equivalence [Hoare, 1985] and failures equivalence [Brookes et al., 1984] are probably the most well known extensional semantics.

Also extensional are the semantics defined by using the testing methodology [Hennessy, 1988] in which two processes are said to be equivalent if they pass the same tests. This formulation provides an strong metaphor[2] and is quite appealing and intuitive. It is easy to find into this framework equivalences which raise the distinction power from traces and failures, increasing the range of possible equivalences: readiness, refusal, ready trace... [Pnueli, 1985, Olderog and Hoare, 1986, Phillips, 1987, Gregorio-Rodríguez and Nuñez, 1999].

There have been several works trying to connect somehow intensional and extensional approaches to define equivalences on processes. In [Abramsky, 1987], Abramsky exposed clearly one of the links: bisimulation equivalence can be view

[2] This testing scenario was also used by Milner to motivate his equivalence, but the definition of bisimulation equivalence definition is clearly intensional.

has an special kind of testing equivalence. Classical equivalences, as that induced by traces and failures, could also be described in the same framework.

In [Cleaveland and Hennessy, 1992] it is shown how to decide some testing equivalences for finite-state processes using bisimulation equivalence checkers. To achieve that, the intensional definition of concurrent processes, that is, the label transition system, is transformed in such a way that deciding testing equivalence is reduced to deciding a slightly generalisation of bisimulation equivalence.

A fundamental work on the comparison between semantics of processes is the paper by Van Glabbeek [Glabbeek, 2001]. There, all the inclusion relations between twelve different semantics are proved, see Figure 1. Besides, for each equivalence a motivating testing scenario is presented and a complete axiomatization is provided.

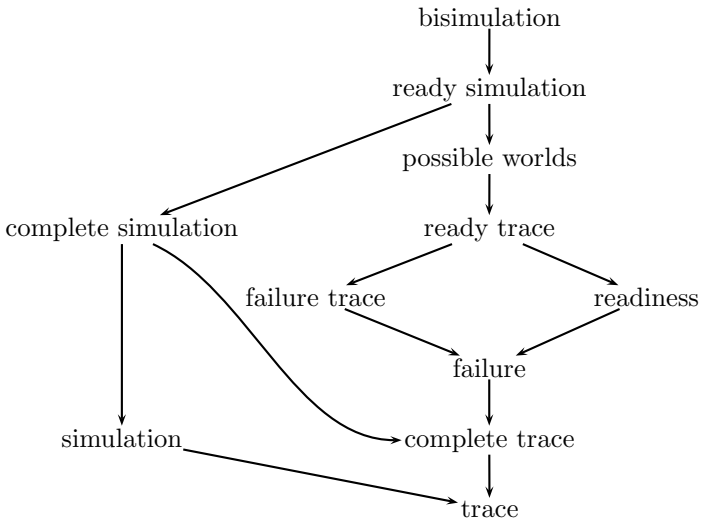


Figure 1: Semantics with an Axiomatic Characterisation in the Linear Time-Branching Time Spectrum I

One of the goals of our work is to define a general framework based on bisimulation semantics where weaker equivalences (both intensionally and extensionally defined [Cleaveland and Smolka, 1996]) can be defined, preserving the coinductive flavour of bisimulation, in order to be able to use, as a tool for their study, coalgebraic methods that have proved to be mathematically powerful and elegant.

For that, we propose a generalisation of the idea of bisimulation, that we

call *global bisimulation*. Global bisimulation is not a single notion but a general scenario in which we can define several concrete instances, that we have proved to be equivalent to many of the most useful and well known classical equivalences between processes, such as:

- Strong bisimulation equivalence [Milner, 1989]
- Ready similarity equivalence [Bloom et al., 1995, Larsen and Skou, 1991]
- Failures equivalence [Brookes et al., 1984, Hoare, 1985, Hennessy, 1988]
- Trace equivalence [Hoare, 1985]

The power of global bisimulation to express these four equivalences demonstrates its versatility (originally, strong bisimulation and ready similarity are intensionally defined while failures and traces are extensional) but, what is more important, it provides us with the framework to study under a new light the similarities and differences between several semantics.

The key idea underlying global bisimulation is the combination of two kind of transitions when describing the evolution of a process. We will consider *dynamic* transitions, which correspond to the execution of visible actions, and *static* transitions, which have to do with the partial resolution or delaying of non-determinism. Changing the allowed static transitions we will define different notions of equivalence.

The ideas in which global bisimulation are based were introduced, in a slightly different context, in [de Frutos-Escrig et al., 1999]. In that paper an equivalence coarser than the weak timed bisimulation is proposed, under which the internal choice operator becomes associative.

There have been in the literature other generalisations of the notion of bisimulation. Let us just cite here [Gardiner, 2003] where power simulation is elegantly described in terms of predicate transformers providing an alternative characterisation to trace and failure orders and equivalences; however, it is strange that bisimulation equivalence itself cannot be achieved as a power simulation.

The paper is structured as follows. In Section 2 we recall the definition of bisimulation equivalence and introduce the main ideas of global bisimulation in the framework of our research on comparative semantics. In Section 3 we face the problem of defining a global bisimulation equivalence relation with the same discriminatory power as ready simulation equivalence. We discuss the difficulties encountered and the possibilities global bisimulation offers to overcome them. In Section 4 we formally define a global bisimulation that induces an equivalence relation equivalent to the ready similarity, proving this result. In Section 5 we shift the focus from intensional semantics to extensional ones. We prove that it is also possible to define global bisimulations to characterise failure equivalence and trace equivalence. In Section 6 we extend the results of previous sections to

infinite finitary tree processes. To achieve this we use continuity arguments: we define the notion of level continuity and we prove how the global bisimulations are level continuous. Finally, in Section 7 we present some conclusions and directions for further work.

2 Preliminaries and Goals

The behaviour of processes is usually described using the well established formalism of *labelled transition systems* [Plotkin, 1981], or lts for short.

Definition 1. A labelled transition system is a structure $(\mathcal{P}, Act, \rightarrow)$ where

- $\mathcal{P}$ is a set of process, agents or states,
- Act is a set of actions and
- $\rightarrow \subseteq \mathcal{P} \times Act \times \mathcal{P}$ is a transition relation.

Act is the set of actions that processes can perform and the transition relation $\rightarrow$ describes the process transitions after the execution of actions. The triple $\langle p, a, q\rangle \in \rightarrow$ is represented by $p \xrightarrow{a} q$, indicating that process p performs action a evolving to process q.

Some usual notation on lts is used. We write $p \xrightarrow{a}$ if there exists a process q such that $p \xrightarrow{a} q$ and, on the contrary, we write $p \not\xrightarrow{a}$ if there exists no process q such that $p \xrightarrow{a} q$. For a string of actions $\sigma = a_1 a_2 \cdots a_n$, $a_i \in Act$, $p \xrightarrow{\sigma} q$ means that there exist processes $q_1 \ldots q_{n-1}$, such that $p \xrightarrow{a_1} q_1 \xrightarrow{a_2} q_2 \xrightarrow{a_3} \cdots q_{n-1} \xrightarrow{a_n} q$. The function I calculates the set of initial actions of a process, $I(p) = \{a \mid a \in Act \text{ and } p \xrightarrow{a}\}$. A process ap' is a summand of the process p if and only if $p \xrightarrow{a} p'$.

Labelled transition systems define the behaviour of processes in terms of the actions they can perform, so it is natural to define a process equivalence in terms of these action transitions. That is precisely what bisimulation does: to inductively explore the intensionally defined behaviour of processes. Bisimulation has became one of the fundamental notions in the theory of concurrent processes. It can be defined as follows:

Definition 2 [Milner, 1989]. A binary relation $\mathcal{R}$ is called a (strong) *bisimulation* if for all p, q processes such as $p\ \mathcal{R}\ q$, and for all $a \in Act$ the following properties are satisfied:

- Whenever $p \xrightarrow{a} p'$ there exists some q' such that $q \xrightarrow{a} q'$ and $p'\ \mathcal{R}\ q'$.
- Whenever $q \xrightarrow{a} q'$ there exists some p' such that $p \xrightarrow{a} p'$ and $p'\ \mathcal{R}\ q'$.

Two processes p and q are *bisimilar*, which we denote by $p =_B q$, if there exists a bisimulation containing the pair $\langle p, q \rangle$.

Strong bisimulation states that two processes, p and q, are equivalent if q can *simulate* p and conversely (thus the prefix bi) p can simulate q, both at the same time. Process q can simulate p if whenever p performs an action a and evolves to p' then q is also able to perform action a and reach some q' which is equivalent to p'. Let us note that the definition imposes *simultaneous* simulations by means of a single symmetrical definition of bisimulations. If instead, *separate* simulations are considered, the induced simulation equivalence is weaker than bisimulation equivalence (see for instance [Glabbeek, 2001] for more details).

This simple and elegant recursive definition of bisimulations induces an equivalence between processes which has several nice properties: it can be characterised by a simple and natural logic [Hennessy and Milner, 1985]; there exist efficient algorithms that allow to decide bisimulation [Paige and Tarjan, 1987, Kanellakis and Smolka, 1990]; and therefore, there are several tools that can effectively check process bisimilarity [Cleaveland et al., 1993].

Bisimulation equivalence can also be defined in terms of a game (see for instance [Stirling, 1998]). Trying to prove or disprove the bisimilarity of two processes we consider two players: the one who starts is the attacker and tries to prove non-bisimilarity; in front of him we have the defender, trying to prove bisimilarity. The board of the game is formed by the processes transition diagrams, where the players have to move by performing actions of them. The attacker starts, chooses one of the processes and performs an action. Then, the defender has to match the same action on the other process, trying to arrive to a state which is bisimilar to that reached by the attacker. An important fact related with the symmetry of the definition is that the attacker can make his moves on either process transition diagram, while the defender has to find the right movement on the other process. If the attacker is able to arrive to a state in which he can make a move, that is, to execute an action that the defender cannot execute in its current state, then the compared processes are not bisimilar. On the other hand, if the defender can always simulate the attacker movements, then the two compared processes are bisimilar. This presentation of bisimulation as a game will be very useful to explain the main ideas of our global bisimulation.

One of the main goals of our work is to provide a framework in which to define, study and compare different equivalences on processes. The idea of starting from the definition of bisimulation is quite appellant due to its good properties but, as we mentioned in the introduction, bisimulation defines a quite narrow[3] equivalence. Therefore, in order to get a variety of equivalences from the definition of bisimulation it is necessary to relax in some way the strong restrictions imposed in that definition.

[3] For instance, none of the pairs of different processes in Figure 2 is bisimilar.

From the game-semantics point of view one possibility is to give to the defender more chances to simulate any movement of the opponent. But this is not an easy task if we want to maintain a symmetric definition, that is, the same rules for both players: whenever the board-game is enlarged to allow more freedom to the defender we also give the attacker new possibilities to use in his own profit!

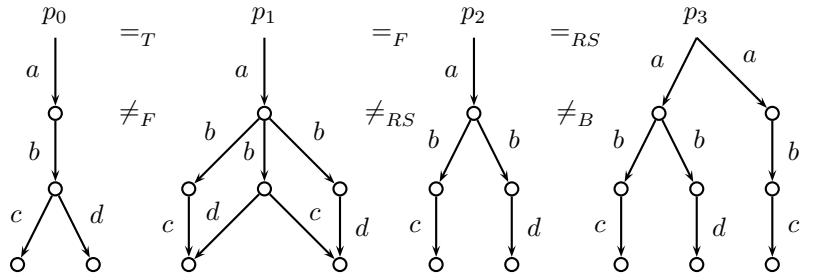


Figure 2: Examples/Counterexamples of Equivalence Relations over Processes.

Bisimulation is based on the execution of (local) action transitions defining the process behaviour. However, we think that, besides action transitions, there is another fundamental concept into the definition of the behaviour of processes which plays an essential role: non-determinism. The key idea of our work is to study the role of non-determinism in the definition of bisimulation. Our generalisation of bisimulations, global bisimulations will transform the transition relation of processes in order to get essentially the same action transition but adding new transitions that introduce additional control over the non-determinism of processes.

More technically, given a labelled transition system $(\mathcal{P}, Act, \rightarrow)$ we will introduce a new relation $\rightsquigarrow$ which captures the desired way to cope with non-determinism in processes. Then, an extended set of states, $\widetilde{\mathcal{P}}$, and an extended transition relation, $\approx\!\!>\!\!=\!\!\rightsquigarrow^*\!\!\rightarrow$, will be considered, so that global bisimulation will be just plain bisimulation over the extended system $(\widetilde{\mathcal{P}}, Act, \approx\!\!>)$.

A similar strategy has been followed in [Cleaveland and Hennessy, 1992] where the transformation of the original labelled transition system that models the given processes is defined in such a way that the testing relations on the original systems can be checked using bisimulations on the transformed system. However, this transformation is defined in a less operational way than ours, and it is difficult to establish a simple relation between the added transitions and the old ones. Even more, it is not possible to distinguish these new transitions just by looking at the transformed system. We could say that these transformed systems are in fact the values of a denotational semantics[4] which defines a normal form for processes.

In contrast, in our approach we aim to preserve as much as possible the operational semantics of the processes, just adding the new transitions that are needed to capture the semantics in which we are interested, and keeping them separate of the original transitions.

In this way, global bisimulation will establish a common framework in which the differences between several semantics can be expressed in terms of the different treatment of non-determinism. In particular, when $\rightsquigarrow$ is empty, global bisimulation is just plain bisimulation and therefore our approach is *conservative*, in the sense that the original notion of bisimulation is a particular case of global bisimulation.

3 Ready Similarity and Global Bisimulation (A Case Study)

In this section we consider the problem of finding a global bisimulation being equivalent to the ready similar equivalence relation. As explained before, given a lts $\mathcal{L} = (\mathcal{P}, Act, \rightarrow)$, and the ready similarity relation $=_{RS}$, we want to find the proper definition of $\rightsquigarrow$ so that the extended transition system $\widetilde{\mathcal{L}} = (\widetilde{\mathcal{P}}, Act, \approx\!\!>)$ is such that $p =_{RS} q$ in $\mathcal{L}$ if and only if $p =_B q$ in $\widetilde{\mathcal{L}}$. First, in Section 3.1 we recall the definition of the ready simulation equivalence and we highlight the differences between simulations and bisimulations. In Section 3.2 we discuss some representative examples trying to synthesise the rules that generate the global transitions needed to characterise the ready similarity as a global bisimulation.

3.1 Ready Simulation Equivalence

Bloom, Istrail and Meyer were amongst the first authors that discussed the adequacy of bisimulation as a way to define the semantics of concurrent processes. They gave several arguments supporting their opinion along an interesting collection of papers where they thoroughly studied the discriminatory power of bisimulation.

[4] Denotational semantics are defined in a compositional way, that is, following a procedure which in general makes difficult to relate any specific part of the denotational values with a simple point in the behaviour of the corresponding process.

In [Bloom et al., 1988] a quite general format of structured rules for transitions among terms, the GSOS format, is defined, proving that bisimulation equivalence is not a congruence with respect to the languages defined in such a format. The finest equivalence that is a congruence with respect to all of these languages is called GSOS trace congruence, and a characterisation in terms of a modal logic for this equivalence was given, comparing it with the Hennessy-Milner logic characterising bisimulation equivalence. In [Bloom and Meyer, 1992] it is argued that in a context of testing or button-pushing scenario, the power of bisimulation equivalence goes beyond what an external observer can really observe. Besides, the GSOS trace congruence equivalence was proved to be the same as $\frac{2}{3}$-bisimulation, defined in a probabilistic context in [Larsen and Skou, 1991]. In this way, GSOS trace congruence could be defined in a bisimulation-like manner, but with a coarser distinction power, because the imposed conditions were not so restrictive as in the definition of bisimulation. This equivalence was called ready similarity or ready simulation equivalence.

Definition 3 [Bloom et al., 1995, Larsen and Skou, 1991]. A binary relation $\mathcal{R}$ on processes is called a *ready simulation* if for all p, q such as $p\ \mathcal{R}\ q$, and for all $a \in Act$, the following properties are satisfied:

- Whenever $p \xrightarrow{a} p'$ there exists some q' such that $q \xrightarrow{a} q'$ and $p'\ \mathcal{R}\ q'$.
- Whenever $q \xrightarrow{a}$ then $p \xrightarrow{a}$.

Two processes are ready similar, what we denote by $p =_{RS} q$, if there exists a ready simulation $\mathcal{R}$ with $p\ \mathcal{R}\ q$ ($p \sqsubseteq_{RS} q$) and also a ready simulation $\mathcal{S}$ such that $q\ \mathcal{S}\ p$ ($q \sqsubseteq_{RS} p$).

Let us note that this intensional definition is mainly that of an asymmetric simulation. The corresponding equivalence relation is obtained by mutual ready simulation. So that it is defined in a similar way to simulation semantics, but not directly in a symmetric way, as in the case of bisimulation. In terms of the game-approach we proposed on Section 2, if we wish to decide $p \sqsubseteq_{RS} q$, the rules of the game change, mainly because the attacker has to play always on p and the defender on q. This introduces an asymmetry on the game which generates an ordering relation between processes instead of an equivalence relation as in the case of the bisimulation game. Therefore, if we want to check $p =_{RS} q$, it is necessary to play the game twice, one for each inequality, $p \sqsubseteq_{RS} q$ and $q \sqsubseteq_{RS} p$. However, we provide a direct characterisation of the ready similarity equivalence in terms of a global bisimulation which now deserves to be called *global ready bisimulation*. In terms of the corresponding game-semantics, this means that we need just to play once to directly capture the ready simulation equivalence.

3.2 Getting a Global Bisimulation

As we introduced in the previous sections, global bisimulations are based not only on the original transitions of the processes, that we call *dynamic* since they correspond to the evolution of the systems by executing actions, but also on the *static* transitions that we introduce to control the resolution of non-determinism in the behaviour of processes.

In this section we search for the static rules that define the static transitions needed to extend the given lts in such a way that the ready simulation equivalence becomes plain bisimulation equivalence in the extended system. Next we will discuss the ideas and problems found when looking for the desired definition of a global bisimulation characterising the ready similarity. The formalisation of the corresponding definitions will be made in Section 4.

First we note that, without any loss of generality, we can consider that the labelled transition systems are forests, or just single trees, if we fix an initial state. Any lts can be expanded into a forest just by unlimited unfolding of the subgraph of reachable states from each state of the original system. The so obtained forest is equivalent to the unfolded system up to the identity of states, in which we are not interested when defining the semantics of systems.

In order to make easier the presentation of our global bisimulation, both at the intuitive and at the formal level, and the proof of the characterisations we are looking for, we will restrict ourself to finite tree systems until Section 6, where we will see how the definitions and results can be extended to infinite tree systems and therefore, to arbitrary systems.

To be able to compare processes, or states, from different tree systems we represent them by pairs $(\mathcal{L}, p)$. An action transition between the states p and q of the system $\mathcal{L}$, $(\mathcal{L}, p) \xrightarrow{a} (\mathcal{L}, q)$ is graphically represented as in Figure 3(a).

Instead, the static transitions do not modify the states, but will change the behaviour of the process by adding or removing some new transitions that modify the whole system, as it is illustrated in Figure 3(b), where P' is the system obtained after the adequate modification of P.

As shown in Figure 4, if we consider together both types of transitions, dynamic and static, there are two dimensions on which the processes can evolve: They can perform (vertical) action transitions or can resolve some of its non-determinism by means of the (horizontal) static transitions. Now, when playing the bisimulation game in this new board, both the attacker and the defender will be able to move by combining both kinds of transitions.

As usual, for the sake of simplicity, we often use the sum operator $(+)$ on trees, that is associative and commutative. For instance, by using the sum operator we can define a tree as a summation of subtrees $p = \sum_{i=1}^{k} a_i p_i$ where each a_i is an action and each p_i is a tree.

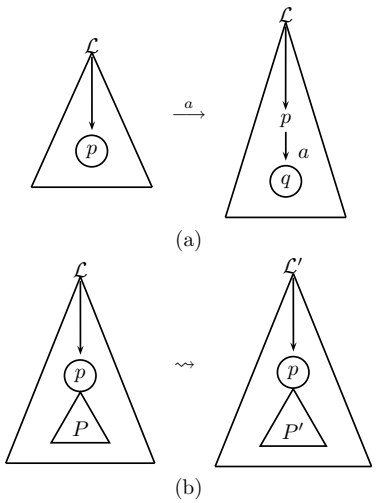


Figure 3: Dynamic and Static Transitions

In Example 1 we present two processes that are ready similar but not bisimilar.

Example 1. Let us consider the processes $p_2 = a(bc + bd)$ and $p_3 = a(bc + bd) + abc$ in Figure 2. They are ready similar, $p_2 =_{RS} p_3$, but not bisimilar, $p_2 \neq_B p_3$; we prove both facts, playing the games characterising ready simulation and bisimulation equivalences, respectively:

1. $a(bc + bd) =_{RS} a(bc + bd) + abc$.
   (a) $a(bc + bd) \sqsubseteq_{RS} a(bc + bd) + abc$.
   Obviously the second clause of the definition of ready simulations (Definition 3) is satisfied: $p_2 \xrightarrow{a} \iff p_3 \xrightarrow{a}$. For the first clause, when the attacker plays on p_2, $a(bc + bd) \xrightarrow{a} bc + bd$, then the defender does $a(bc+bd)+abc \xrightarrow{a} bc+bd$ and wins. Formally, the game should continue, but when the processes are equal the defender wins just mirroring the moves of the attacker.

   (b) $a(bc + bd) + abc \sqsubseteq_{RS} a(bc + bd)$.
   As in the case above, we have $p_3 \xrightarrow{a} \iff p_2 \xrightarrow{a}$. For the first clause now the attacker has two possible moves
      i. if he plays $a(bc + bd) + abc \xrightarrow{a} bc + bd$, then the defender moves $a(bc + bd) \xrightarrow{a} bc + bd$ and wins;

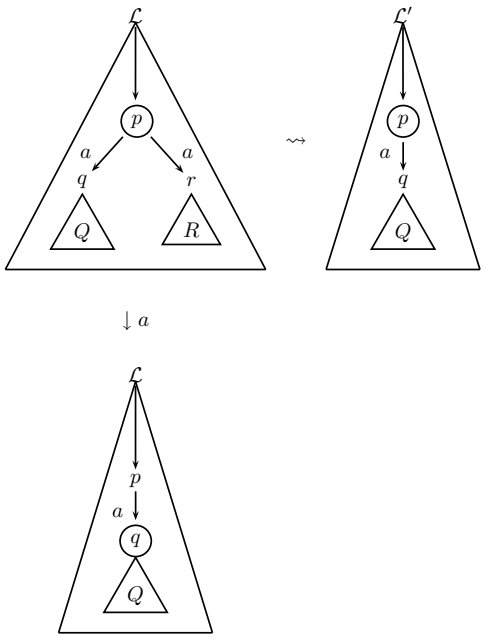


Figure 4: Combining Dynamic and Static Transitions

ii. if he chooses $a(bc + bd) + abc \xrightarrow{a} bc$, then the defender executes $a(bc + bd) \xrightarrow{a} bc + bd$ and wins, because clearly $bc \sqsubseteq_{RS} bc + bd$.

2. $a(bc + bd) \neq_B a(bc + bd) + abc$.
If the attacker starts playing at p_3 by performing the transition $a(bc + bd) + abc \xrightarrow{a} bc$, then the defender has to move on p_2, and has only one way to simulate the a transition: $a(bc + bd) \xrightarrow{a} bc + bd$. But then, the attacker can change to the other side of the board playing $bc + bd \xrightarrow{b} d$ and the defender can only execute $bc \xrightarrow{b} c$, losing the game, since, obviously, $c \neq_B d$.

Now, let us look for a modification of the rules of the bisimulation game that would allow the defender to win in Example 1. The defender lost when he was obliged to reach the state $bc + bd$ after executing a. He is reaching a non-deterministic state whose too large offering is then used by the attacker to win the game. Therefore, the defender would win if he were allowed to remove the offering of bd in advance, that is, to move from $a(bc + bd)$ to bc, when performing the action a. This is possible if he can perform an *static* move in which non-determinism is partially solved.

So, the general form of the global rule G_{nd}, G after global and nd after

non-determinism, is

$$ap + aq + r \rightsquigarrow aq + r \qquad (G_{nd})$$

graphically represented in Figure 5.

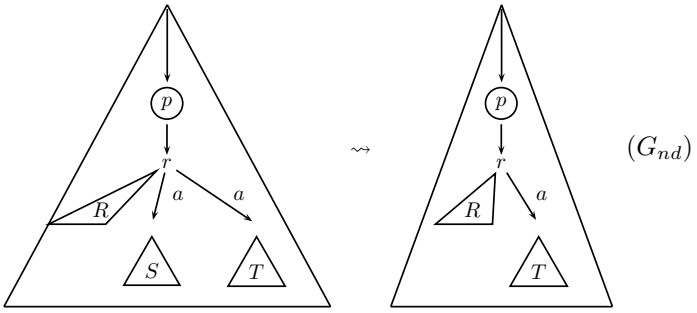


Figure 5: Global Rule G_{nd}, Removing of Non-determinism.

Sometimes we will need to apply the rule G_{nd} really *globally*, that is, not only in the head of processes but anywhere inside the processes, what is expressed in the graphical representation by the separation of the actual state p and the state r below, where the reduction of non-determinism is done.

Let us consider again the processes p_2 and p_3 in Figure 2, but where action a is refined into a_1a_2, getting the processes $p_2' = a_1a_2(bc + bd)$ and $p_3' = a_1a_2(bc + bd) + a_1a_2bc$. Rephrasing the explanations in Example 1, if the attacker starts on p_3 by executing a_1, now the defender playing in p_2 has the possibility of resolving in advance its non-determinism.

Now, if together with the transition relation $\rightarrow$, we take as the relation $\rightsquigarrow$ the one defined by rule G_{nd}, then we will denote by $=_{G_{nd}}$ the equivalence relation induced by the bisimulation equivalence over the lts including both $\rightarrow$ and $\rightsquigarrow$.

But a new example points out that this rule alone is not enough to achieve the discriminatory power of ready similarity.

Example 2. Let us consider the processes $p = d(abc + abd) + da(bc + bd)$ and $q = da(bc + bd)$ in Figure 6.

Ready similarity of p and q, $p =_{RS} q$, can easily be proved by using the axiomatization for the ready simulation equivalence in [Glabbeek, 2001]): Applying

the axiom $a(x+y) =_{RS} a(x+y)+ay$, then it follows

$$\begin{aligned} a(bc+bd) &=_{RS} a(bc+bd)+abc \\ &=_{RS} a(bc+bd)+abc+abd \end{aligned}$$

Therefore, for the process p, and the application of the previous equivalence to the underlined subprocesses we get

$$\begin{aligned} &p = d(abc+abd)+d\underline{a(bc+bd)} =_{RS} \\ &d(abc+abd)+d(\underline{a(bc+bd)+abc+abd}) \end{aligned}$$

If we continue applying the axiom in the opposite direction to the underbraced named subterms, we get

$$\begin{aligned} &d(\underbrace{abc+abd}_{y})+d(\underbrace{a(bc+bd)}_{x})+\underbrace{abc+abd}_{y}) =_{RS} \\ &d(\underbrace{a(bc+bd)}_{x})+\underbrace{abc+abd}_{y}) \end{aligned}$$

And finally, the derived axiom $a(bc+bd) =_{RS} a(bc+bd)+abc+abd$ applied to the underlined subprocess produces the desired result

$$\begin{aligned} d(\underline{a(bc+bd)+abc+abd}) &=_{RS} d(\underline{a(bc+bd)}) \\ &=_{RS} q. \end{aligned}$$

However $p \neq_{G_{nd}} q$. Taking $q' = a(bc+bd)$ and $p' = abc+abd$, if we play the global bisimulation game for $\rightsquigarrow$ induced by the rule G_{nd}, then the attacker can start playing on p, performing the move $p \xrightarrow{d} p'$, and then the defender has only three possible moves:

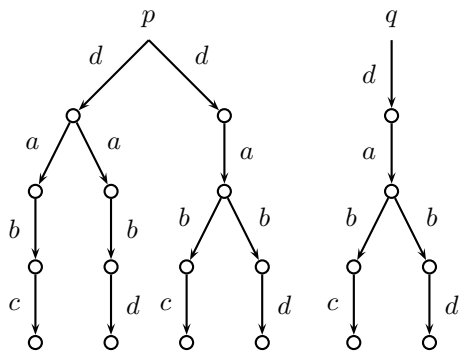
1. Either he plays action transition $q \xrightarrow{d} q'$, and then he loses, since the attacker can play on q', $q' \xrightarrow{a} bc+bd$, and then the defender can only choose between $p' \xrightarrow{a} bc$ or $p' \xrightarrow{a} bd$, and obviously no one of these processes is global bisimilar to $bc+bd$;

2. Or he plays a combination of an action transition and a static transition $q \xrightarrow{d} q' \rightsquigarrow abc$, getting a process that is not global bisimilar to p';

3. Or, symmetrically, plays the action transition and the other possible static transition $q \xrightarrow{d} q' \rightsquigarrow abd$, getting also a process not global bisimilar to p'.

This means that the defender loses and therefore the global bisimilarity between p and q is disproved.

Let us study why processes p and q in Example 2 are ready similar, $p =_{RS} q$, but not $p =_{G_{nd}} q$. The problem arises when q tries to simulate p. In fact, the



$d(abc + abd) + da(bc + bd) \quad da(bc + bd)$

Figure 6: $p =_{RS} q$ but $p \neq_{G_{nd}} q$.

key point is how to simulate the move $p \xrightarrow{d} p'$, as we have that $p \sqsubseteq_{RS} q$, since $p' \sqsubseteq_{RS} q'$. This can be proved using the axiom $ax \sqsubseteq_{RS} ax + ay$ in the following way: From $abc \sqsubseteq_{RS} a(bc + bd)$ and $abd \sqsubseteq_{RS} a(bc + bd)$ we obtain $p' = abc + abd \sqsubseteq_{RS} q' + q'$ and because $q' + q' =_B q'$ we can conclude that $p' \sqsubseteq_{RS} q' + q' \sqsubseteq_{RS} q'$.

It is a bit surprising to find that the use of the equivalence axiom $q + q = q$, which is even satisfied by strong bisimulation equivalence, and therefore by any equivalence relation coarser than it, is crucial in that proof.

We conclude that adding a new rule we could solve situations like that in the example above. The reader can easily check that applying G_{dp} once, and then G_{nd} two times, the defender can turn q' into p', thus winning the game. This new rule G_{dp}, G after global and dp after duplication, is given by

$$p \rightsquigarrow p + p \qquad (G_{dp})$$

which is represented in our tree graphic style in Figure 7.

As we will prove in the following section, taking as global (static) transitions those defined by G_{nd} and G_{dp}, the corresponding global bisimulation equivalence coincides with ready similarity, thus getting the desired characterisation.

Therefore, we can conclude that to get a global bisimulation equivalence with the same discriminatory power as ready simulation equivalence we need mainly to be able to remove non-determinism in an adequate way: Either in a direct way, using the rule G_{nd}, or in a, let us say, partial way, by applying G_{nd} to

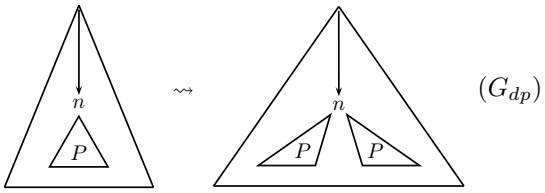


Figure 7: Global Rule G_{dp}, Duplication.

remove non-determinism in different ways from several copies of a process, first duplicated by using the rule G_{dp}.

For the sake of clarity, we have presented our ideas in a rather intuitive way. But the paths to explore are full of pitfalls. We conclude this section with an example that shows that the introduction of global transitions could be quite harmful.

Example 3. The global equivalence defined using only the G_{nd} rule, $=_{G_{nd}}$, is not weaker than bisimulation equivalence. Let us consider $q' = a(bc + bd)$, as in Example 2, and processes dq' and $d(q' + q')$.

Given that $p + p = p$ is the axiom that defines bisimulation equivalence dq' is bisimilar to $d(q' + q')$. However, $dq' \neq_{G_{nd}} d(q' + q')$: If the attacker begins with $d(q' + q') \xrightarrow{d} q' + q' \rightsquigarrow abc + abd$, then the defender can only perform d arriving to $a(bc + bd)$. Then, as we have already seen in detail in Example 2, we have $abc + abd \neq_{G_{nd}} a(bc + bd)$.

This example shows how the introduction of global rules is not an easy task. A symmetric definition of global bisimulation sets the bases for a game with the same rules for both players, attacker and defender, and therefore, when we add global rules trying to give more power to the defender, they can also be used by the attacker.

4 Ready Similarity as a Global Bisimulation

In order to formally define the global ready bisimulation generating an equivalence relation $\approx_{RS}$ which coincides with the ready simulation equivalence, $=_{RS}$, we recall Figure 4.

Both types of transitions can be put together by considering an extended

lts whose states are the pairs $(\mathcal{L},p)$ and its transition relation $\approx\!\!>$ is defined[5] as $\approx\!\!>=\rightsquigarrow^*\!\!\rightarrow$, where $\rightsquigarrow^*$ is the reflexive and transitive closure of $\rightsquigarrow$. That is, $(\mathcal{L},p) \overset{a}{\approx\!\!>} (\mathcal{L}',p')$ iff $(\mathcal{L},p) \rightsquigarrow^* (\mathcal{L}',p) \xrightarrow{a} (\mathcal{L}',p')$.

Definition 4. Given a lts $\mathcal{L}=(\mathcal{P},Act,\rightarrow)$, and G, a set of global rules defining a global transition relation $\rightsquigarrow$, we define the *global labelled transition system* $\widetilde{\mathcal{L}} = (\widetilde{\mathcal{P}},Act,\approx\!\!>)$, where $\approx\!\!>=\rightsquigarrow^*\!\!\rightarrow$ and $\widetilde{\mathcal{P}}$ is the set of states reachable via $\approx\!\!>$ starting from the initial system $\mathcal{L}$. By putting together the systems $\widetilde{\mathcal{L}}$'s corresponding to the trees $\mathcal{L}$'s in a forest-like system $\mathcal{F}$ we would obtain the corresponding global lts $\widetilde{\mathcal{F}}$.

Let us note that every state in $\mathcal{F}$, $(\mathcal{L},p)$, is reachable, and therefore it is included in $\widetilde{\mathcal{F}}$, so that the following definition makes sense.

Definition 5. Given a lts, $\mathcal{F}=(\mathcal{P},Act,\rightarrow)$ and the corresponding global lts $\widetilde{\mathcal{F}}=(\widetilde{\mathcal{P}},Act,\approx\!\!>)$, we define the *global equivalence relation* $\approx_{GB}$ as follows: For all $(\mathcal{L},p),(\mathcal{L}',p') \in \mathcal{P}$, $(\mathcal{L},p) \approx_{GB} (\mathcal{L}',p')$ (in $\mathcal{F}$) iff the corresponding states $(\mathcal{L},p)$ and $(\mathcal{L}',p') \in \widetilde{\mathcal{P}}$ are strongly bisimilar (in $\widetilde{\mathcal{F}}$), $(\mathcal{L},p) =_B (\mathcal{L}',p')$.

Definition 6. If we consider the set of global rules $G_{RS} = \{G_{nd},G_{dp}\}$, then the corresponding global equivalence defined by Definition 5 is called the *global ready equivalence relation*, denoted by $\approx_{RS}$.

In order to simplify the proof of Theorem 10 below, we introduce the following lemmata.

Lemma 7. *Let us consider the set of global rules $G_{RS} = \{G_{nd},G_{dp}\}$. If $p \rightsquigarrow^* q$ then we have $q \sqsubseteq_{RS} p$.*

Proof. It is enough to see that if $p \rightsquigarrow q$ then $q \sqsubseteq_{RS} p$. We have two cases:

- Either we use the static rule G_{nd}, $ap+aq+r \rightsquigarrow aq+r$, but then $aq+r \sqsubseteq_{RS} ap+aq+r$ (see, for instance, [Glabbeek, 2001] for an axiomatization of the preorder $\sqsubseteq_{RS}$);
- Or we use the static rule G_{dp}, $p \rightsquigarrow p+p$, but then p and $p+p$ are bisimilar, $p+p =_B p$, and therefore $p+p \sqsubseteq_{RS} p$.

Definition 8. The restriction to action a on process p, $p|_a$, defines the (sub)-process we get by adding all the a-summands of p. That is, if $p=\sum_I a_i p_i$ then $p|_a = \sum_J a_j p_j$ where $J=\{i \in I \mid a_i = a\}$.

Lemma 9. *If $p \approx_{RS} q$ then for all $a \in I(p)$ we have that $p|_a \approx_{RS} q|_a$.*

[5] It is easy to see that for the ready simulation equivalence $\rightsquigarrow$ and $\rightarrow$ commute, so that it would be equivalent to define $\approx\!\!>$ as $\rightarrow\!\rightsquigarrow^*$ or even as $\rightsquigarrow^*\!\!\rightarrow\!\rightsquigarrow^*$.

Proof. Let us assume that $p|_a \not\approx_{RS} q|_a$, then there exists p' such that $p|_a \overset{a}{\Rightarrow} p'$ and p' is not global ready equivalent to any q' such that $q|_a \overset{a}{\Rightarrow} q'$. Given that the rules in G_{RS} do not introduce any new action it is not possible to q to use $\rightsquigarrow^*$ to arrive to a state that can perform action a and that is not in $q|_a$, so that we also have $p \not\approx_{RS} q$.

Theorem 10. *Given a lts $\mathcal{L} = (\mathcal{P}, Act, \rightarrow)$, we have*

$$p =_{RS} q \text{ iff } p \approx_{RS} q.$$

Proof. We only need to prove the following three statements:

$$\begin{aligned} p =_{RS} q &\overset{(1)}{\Longrightarrow} (p \rightsquigarrow^* q \text{ and } q \rightsquigarrow^* p) \\ &\overset{(2)}{\Longrightarrow} p \approx_{RS} q \\ &\overset{(3)}{\Longrightarrow} p =_{RS} q \end{aligned}$$

To make it more readable, along the proof we will occasionally index the global transitions by the name of the rule used to generate them.

(1) We will use the axioms that characterise the ready similarity, see for instance [Glabbeek, 2001]. The axioms for lts that define the equivalence $=_{RS}$ are the following:

- $p = p + p$
- $I(x) \subseteq I(y) \Rightarrow a(x + y) = a(x + y) + ay$,

If $p =_{RS} q$ then there exists a finite sequence of applications of the axioms above such that $p =_{RS} p_1 =_{RS} p_2 \cdots =_{RS} q$. The proof that $p =_{RS} q \Rightarrow (p \rightsquigarrow^* q$ and $q \rightsquigarrow^* p)$ can be done by induction on the axiomatic derivation of $p =_{RS} q$, proving that we also have $p \rightsquigarrow^* q$ and $q \rightsquigarrow^* p$. We analyse the use of each of the axioms in the derivation and show how $\rightsquigarrow^*$ is powerful enough to simulate any of them:

- A substitution of p by $p + p$ can be simulated by $p \rightsquigarrow_{dp} p + p$.
- A substitution of $p + p$ by p can be simulated by $p + p \rightsquigarrow_{nd} p$.
- A substitution of $a(x + y) + ay$ by $a(x + y)$ can be simulated by $a(x + y) + ay \rightsquigarrow_{nd} a(x + y)$.
- A substitution of $a(x + y)$ by $a(x + y) + ay$ can be simulated by $a(x + y) \rightsquigarrow_{dp} a(x + y) + a(x + y) \rightsquigarrow_{nd} a(x + y) + ay$, since $I(x) \subseteq I(y)$.

(2) To prove that $(p \rightsquigarrow^* q$ and $q \rightsquigarrow^* p) \Rightarrow p \approx_{RS} q$ is immediate: If $p \overset{a}{\Rightarrow} p_a$ then, by definition, $p \rightsquigarrow^* p' \xrightarrow{a} p_a$ and therefore $q \rightsquigarrow^* p \rightsquigarrow^* p' \xrightarrow{a} p_a$, thus getting $q \overset{a}{\Rightarrow} p_a$.

(3) Finally, we have to prove that $p \approx_{RS} q \Rightarrow p =_{RS} q$. It is enough to show that $p|_a \approx_{RS} q|_a \Rightarrow p|_a =_{RS} q|_a$, because $=_{RS}$ is a congruence wrt the sum operators on trees. We proceed by induction on the depth of trees. For processes of depth 0 it is trivial. Otherwise, by Lemma 9 we have that if $p \approx_{RS} q$ $p|_a \approx_{RS} q|_a$, then whenever $p|_a \xrightarrow{a} p'$ we have that $q|_a \rightsquigarrow^* q' \xrightarrow{a} q'_a$. By applying the induction hypothesis we have $p' =_{RS} q'_a$ and therefore $ap' =_{RS} aq'_a$. On the other hand, by Lemma 7 we know that $q' \sqsubseteq_{RS} q|_a$ and, given that $\rightsquigarrow^*$ does not introduce any new action, $I(q') = I(q|_a) = \{a\}$. That is, $q' = aq'_a + r$ for some r with $I(r) = \{a\}$ and therefore we have that $aq'_a \sqsubseteq_{RS} q'$. All together, $ap' \sqsubseteq_{RS} aq'_a \sqsubseteq_{RS} q' \sqsubseteq_{RS} q|_a$ and therefore $p|_a \sqsubseteq_{RS} q|_a$, as we wanted to prove.

5 Extensional Semantics as Global Bisimulations

Through sections 3 and 4 we have seen how to characterise ready simulation equivalence as a global bisimulation. However, the suspicious reader may think that ready simulation and bisimulation have a rather similar coinductive flavour. Also, looking at Figure 1, he may think that our results were possible because both equivalences are rather close in the linear time-branching time spectrum.

In this section we will show how global bisimulations can also characterise other semantics. To show the strength of the formalism, we have chosen two well known semantics, traces and failures, that are defined in an extensional way and that are quite far from bisimulation in the linear time-branching time spectrum.

5.1 Failure Equivalence

For the kind of processes we are considering, a suitable definition of failure equivalence [Brookes et al., 1984, Hoare, 1985, Hennessy, 1988] is the following:

Definition 11. A pair $\langle \sigma, X \rangle \in Act^* \times \mathcal{P}(Act)$ is a failure pair of process p if there exists some q such that $p \xrightarrow{\sigma} q$ and for all $a \in X$ we have $q \not\xrightarrow{a}$. Let $F(p)$ be the set of failures of p. Two processes, p, q, are *failure equivalent* if they have the same set of failures, that is, $F(p) = F(q)$.

Failure semantics was first introduced in [Brookes et al., 1984] and it was then widely spread when Hoare used it in the definition of the description language CSP [Hoare, 1985]. Failure semantics has also aroused in other different contexts; for instance, the testing equivalence defined by De Nicola and Hennessy [Hennessy, 1988] coincides with failure equivalence. Failure semantics is finer than the well known trace semantics that we are going to study in Section 5.2.

As we already done in Section 4 for the ready similarity, our goal is to find the set of global rules that allow to define a global bisimulation with the same distinction power as the failure equivalence.

In order to motivate the rules we are going to introduce we start with an illustrative example.

Example 4. Let us consider the processes $p_1 = a(bc + bd + b(d + c))$ and $p_2 = a(bc + bd)$ that appear in Figure 2. They are failure equivalent, $p_1 =_F p_2$, but they are not bisimilar. For instance, if the attacker starts performing in p_1 the transition, $p_1 \xrightarrow{a} p_1' = bc + bd + b(d + c)$ then the defender can only replicate this movement by performing in p_2 the transition $p_2 \xrightarrow{a} p_2' = bc + bd$. But now the attacker has the possibility to perform b on p_1' by means of $p_1' \xrightarrow{b} d + c$. Then, the defender can still perform action b, but reaching either an state where action d is possible or an state where action c is possible, however, a state in which both actions c and d are possible is not reachable from p_2'.

There are a number of different runs that show that $p1 \neq_B p2$, but all of them explode the fact that p_1 has an state in which it can choose to perform c or d; process p_2 can also perform these actions but *separately*. If we want p_2 to be able to bisimulate p_1 we should allow a global transition to *delay* the non-deterministic choice on action b. The new rule that we introduce is G_{dy}, *dy* after delaying.

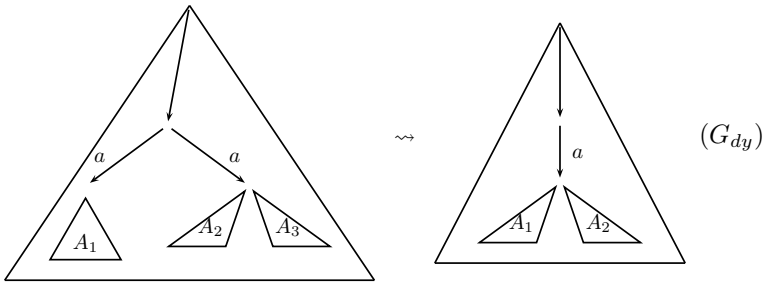


Figure 8: Global Rule G_{dy}, Delaying.

Definition 12. If we consider the set of global rules $G_F = \{G_{nd}, G_{dp}, G_{dy}\}$, then the corresponding global equivalence defined by Definition 5 is called the *global failure equivalence relation*, denoted by $\approx_F$.

Once we have defined $\approx_F$, we want to establish the equivalence between our global bisimulation and the failure semantics.

As we did in Section 4, in the proof we will use Definition 8 and the following lemmata:

Lemma 13. *Let us consider the set of global rules $G_F = \{G_{nd}, G_{dp}, G_{dy}\}$. If $p \rightsquigarrow^* q$ then we have $q \sqsubseteq_F p$.*

Proof. It is enough to prove that if $p \rightsquigarrow q$ then $q \sqsubseteq_F p$. We can distinguish the following cases:

- The use of the static rules G_{nd} and G_{dp}; then, as we already saw in Lemma 7 we have $p \rightsquigarrow q \Rightarrow q \sqsubseteq_{RS} p \Rightarrow q \sqsubseteq_F p$.
- Or the use of the static rule G_{dy}, $ap+a(q+r) \rightsquigarrow a(p+q)$, but then $a(p+q) \sqsubseteq_F ap+a(q+r)$, see for instance [Glabbeek, 2001] for an axiomatization for the preorder $\sqsubseteq_F$.

Lemma 14. *If $p \approx_F q$ then for all $a \in I(p)$ we have $p|_a \approx_F q|_a$.*

Proof. Let us assume that $p|_a \not\approx_F q|_a$, then there exists p' such that $p|_a \overset{a}{\approx\!\!>} p'$ and p' is not global ready equivalent to any q' such that $q|_a \overset{a}{\approx\!\!>} q'$. Given that the rules in G_F do not introduce any new action, it is not possible to q to use $\rightsquigarrow^*$ to arrive to a state where it can perform action a and that is not in $q|_a$ so that we also have $p \not\approx_F q$.

Theorem 15. *Given a lts $\mathcal{L} = (\mathcal{P}, Act, \rightarrow)$, we have*

$$p =_F q \text{ iff } p \approx_F q.$$

Proof. As we did in the proof of Theorem 10, we only need to prove the following three statements:

$$\begin{aligned} p =_F q &\overset{(1)}{\Longrightarrow} (p \rightsquigarrow^* q \text{ and } q \rightsquigarrow^* p) \\ &\overset{(2)}{\Longrightarrow} p \approx_F q \\ &\overset{(3)}{\Longrightarrow} p =_F q \end{aligned}$$

The proof of (2) is the same to that in Theorem 10, but the proofs of statements (1) and (3) have to be rewritten for this case.

(1) We will use the axiomatization for the failure equivalence, see for instance the one in [Glabbeek, 2001]. The axioms that characterise the equivalence $=_F$ for lts are the following:

- $ax + a(y + z) = ax + a(x + y) + a(y + z)$
- $a(bx + u) + a(by + v) = a(bx + by + u) + a(by + v)$

If $p =_F q$ then there exists a finite sequence of applications of the axioms above such that $p =_F p_1 =_F p_2 \cdots =_F q$. The proof that $p =_F q \Rightarrow (p \rightsquigarrow^* q$ and $q \rightsquigarrow^* p)$ can be done by induction on the axiomatic derivation of $p =_F q$ proving that we also have $p \rightsquigarrow^* q$ and $q \rightsquigarrow^* p$. We analyse the use of each of the axioms in the derivation and show how $\rightsquigarrow^*$ is powerful enough to simulate any of them.

- A substitution of $ax+a(y+z)$ by $ax+a(x+y)+a(y+z)$ can be simulated by $ax+a(y+z) \rightsquigarrow^*_{dp} ax+ax+a(y+z)+a(y+z) \rightsquigarrow_{dy} ax+a(x+y)+a(y+z)$.
- A substitution of $ax+a(x+y)+a(y+z)$ by $ax+a(y+z)$ can be simulated by $ax + a(x + y) + a(y + z) \rightsquigarrow_{nd} ax + a(y + z)$.
- A substitution of $a(bx+u)+a(by+v)$ by $a(bx+by+u)+a(by+v)$ can be simulated by $a(bx+u)+a(by+v) \rightsquigarrow_{dp} a(bx+u)+a(by+v)+a(by+v) \rightsquigarrow_{dy} a(bx + by + u) + a(by + v)$.
- A substitution of $a(bx + by + u) + a(by + v)$ by $a(bx + u) + a(by + v)$ can be simulated by $a(bx + by + u) + a(by + v) \rightsquigarrow_{nd} a(bx + u) + a(by + v)$.

(3) Finally, we have to prove that $p \approx_F q \Rightarrow p =_F q$. The proof is similar to that for the ready simulation. It is enough to show that $p|_a \approx_F q|_a \Rightarrow p|_a =_F q|_a$ because $=_F$ is a congruence wrt the sum operators on trees. We proceed by induction on the depth of trees. For processes of depth 0 it is trivial. Otherwise, by Lemma 14 we have that if $p \approx_F q$ we also have $p|_a \approx_F q|_a$. Then whenever $p|_a \xrightarrow{a} p'$ we have that $q|_a \rightsquigarrow^* q' \xrightarrow{a} q'_a$. By applying the induction hypothesis we have $p' =_F q'_a$ and therefore $ap' =_F aq'_a$. On the other hand, by applying Lemma 13 we know that $q' \sqsubseteq_F q|_a$ and, given that $\rightsquigarrow^*$ does not introduce any new action, $I(q') = I(q|_a) = \{a\}$. That is, $q' = aq'_a + r$ for some r with $I(r) = \{a\}$ and therefore we have $aq'_a \sqsubseteq_{RS} q'$. All together, $ap' \sqsubseteq_F aq'_a \sqsubseteq_{RS} q' \sqsubseteq_F q|_a$ and, since $\sqsubseteq_{RS} \Rightarrow \sqsubseteq_F$, we can conclude $p|_a \sqsubseteq_F q|_a$, as we wanted to prove.

5.2 Trace Equivalence

A definition of trace equivalence [Hoare, 1985] for the processes we are considering is the following:

Definition 16. We say that $\sigma \in Act^*$ is a trace of process p if there exists a process q such that $p \xrightarrow{\sigma} q$. Let $T(p)$ be the set of all traces of p. Two processes, p, q, are *trace equivalent* if they have the same set of traces, that is $T(p) = T(q)$.

Trace equivalence is based on the idea that two processes should be identified if they are able to perform the same sequences of actions. This equivalence is the coarsest one in Figure 1 and, therefore, is quite far from bisimulation

equivalence, that is the finest one. Nevertheless it is still possible to define a global bisimulation than characterises the trace equivalence. Of course we need a new global rule that has not appeared yet: G_{rt}, rt after removal of traces (see Figure 9).

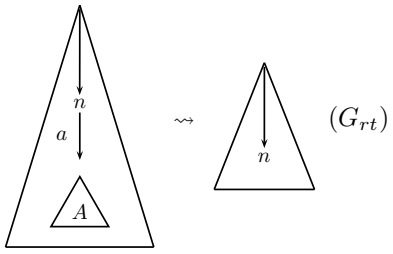


Figure 9: Global Rule G_{rt}, Removal.

Definition 17. If we consider the set of global rules $G_T = \{G_{dp}, G_{dy}G_{rt}\}$, then the corresponding global equivalence defined by Definition 5 is called the *global trace equivalence relation*, denoted by $\approx_T$.

Lemma 18. *Let us consider the set of global rules $G_T = \{G_{dp}, G_{dy}G_{rt}\}$. If $p \rightsquigarrow^* q$ then we have $q \sqsubseteq_T p$.*

Proof. It is enough to prove that if $p \rightsquigarrow q$ then $q \sqsubseteq_T p$. Since we proved in Lemma 13 that using G_{dp} and G_{dy} we preserve failure preorder and $\sqsubseteq_F \Rightarrow \sqsubseteq_T$ we get that G_{dp} and G_{dy} preserve trace preorder.

As for the new static rule G_{tr}, $x + y \rightsquigarrow x$, but then $x \sqsubseteq_T x + y$, see for instance [Glabbeek, 2001] for an axiomatization for the preorder $\sqsubseteq_T$.

Lemma 19. *If $p \approx_T q$ then for all $a \in I(p)$ we have that $p|_a \approx_T q|_a$.*

Proof. The proof follows the same lines as those in proof of Lemma 14.

Theorem 20. *Given a lts $\mathcal{L} = (\mathcal{P}, Act, \rightarrow)$, we have*

$$p =_T q \text{ iff } p \approx_T q.$$

Proof. As we did in the proof of theorems 10 and 15 we prove the following three statements:

$$p =_T q \overset{(1)}{\Longrightarrow} (p \rightsquigarrow^* q \text{ and } q \rightsquigarrow^* p) \overset{(2)}{\Longrightarrow} p \approx_T q \overset{(3)}{\Longrightarrow} p =_T q$$

The proof of (3) follows the same lines as the corresponding in Theorem 15; the proof of (2) is the same to that in Theorem 10. The proof for (1) follows similar reasonings to that of theorems 10 and 15 but needs to be tailored for each set of global rules, in this case for the set G_T.

We will use the axiomatization for the trace equivalence, see [Glabbeek, 2001]. The axioms that characterise the equivalence $=_T$ for lts are the following:

- $x \sqsubseteq x + y$
- $ax + ay = a(x + y)$

If $p =_T q$ then there exists a finite sequence of applications of the axioms above such that $p =_T p_1 =_T p_2 \cdots =_T q$. The proof that $p =_T q \Rightarrow (p \rightsquigarrow^* q$ and $q \rightsquigarrow^* p)$ can be done by induction on the axiomatic derivation of $p =_T q$ proving that we also have $p \rightsquigarrow^* q$ and $q \rightsquigarrow^* p$. We analyse the use of each of the axioms in the derivation and show how $\rightsquigarrow^*$ is powerful enough to simulate any of them.

- A substitution of $x + y$ for x can be simulated by $x + y \rightsquigarrow_{rt} x$.
- A substitution of $ax+ay$ for $a(x+y)$ can be simulated by $ax+ay \rightsquigarrow_{dy} a(x+y)$.
- A substitution of $a(x + y)$ for $ax + ay$ can be simulated by $a(x + y) \rightsquigarrow_{dp} a(x + y) + a(x + y) \rightsquigarrow^*_{rt} ax + ay$.

6 Infinite Processes

Let us now conclude commenting on the results for infinite tree systems. The essence of global bisimulations is captured by the definitions and results for the finite case. They can be extended to the infinite case without substantial changes, but some care is needed at the technical level.

Let us motivate the changes needed in the definition of global bisimulation by means of a simple example. Consider the processes in Figure 10. It is easy to check that $p =_{RS} q$, therefore we would like that also $p \approx_{RS} q$. But if the first player executes the move $p \xrightarrow{a} p'$, then the defender should remove all the offerings of bd in the infinite copies of the state q' in its completely unfolded presentation. Therefore, the defender needs to apply G_{nd} infinitely many times, and not just a finite number of times as in the finite case. So that we get as $\approx\!\!>$ the relation $\approx\!\!> = \rightsquigarrow^\infty \rightarrow$, where $\rightsquigarrow^\infty = \rightsquigarrow^* \cup \rightsquigarrow^\omega$.

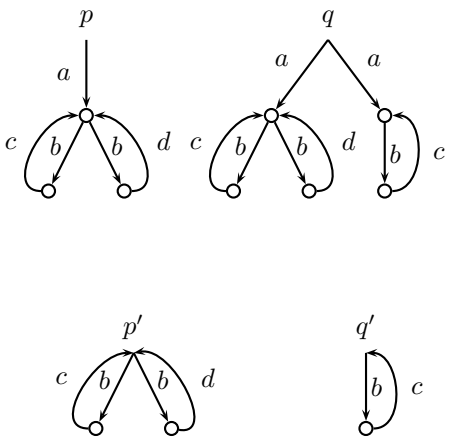


Figure 10: Infinite Processes

Fortunately the formal definition of $\rightsquigarrow^\omega$, the adequate denumerable iteration of $\rightsquigarrow$, is simpler than could be expected, whenever we restrict ourselves to finitely branching trees. In this case, any tree $\mathcal{L}$ can be finitely approximated by its finite cuts $\mathcal{L}^k$, that are obtained just cutting the tree at depth k. Then we can formally define $\rightsquigarrow^\omega$ by $\mathcal{L} \rightsquigarrow^\omega \mathcal{L}'$ if and only if $\mathcal{L}'$ is the infinite tree such that $\mathcal{L}'^k$ is defined by $\mathcal{L}^k \rightsquigarrow_1^* \mathcal{L}_1^k \rightsquigarrow_2^* \mathcal{L}_2^k \cdots \mathcal{L}_k^k = \mathcal{L}'^k$ where all the global transitions in $\rightsquigarrow_l^*$ are applied only to nodes at depth l, and for all $k > l$ the sequence $\rightsquigarrow_l^*$ in the corresponding global computation is the same. Intuitively, this means that each finite approximation $\mathcal{L}'^k$ of $\mathcal{L}'$ is obtained after a finite number of applications of global transitions, so that once fixed, the following applications will continue the construction of the following levels of $\mathcal{L}'$ without changing its finite cut $\mathcal{L}'^k$.

The definition of global bisimulation can also be extended to infinitary trees, but in this case we would need a generalised copy rule making an arbitrary number of copies of a system, and it is possible that we will have to apply rule G_{nd} infinitely many times at the same depth. This, together with the fact that finite approximations of infinitary trees should grow both in depth and width, makes the proof of continuity rather involved.

In order to prove that our global bisimulaton relation $\approx_{RS}$ also characterise ready similarity for infinite finitary trees we need to apply a continuity argument. We start by defining the adequate continuity concept.

Definition 21. A behaviour preorder is level continuous if $p \sqsubseteq q$ if and only if

for all n $p\downarrow_n \sqsubseteq q\downarrow_n$, where $p\downarrow_n$ is the result of pruning process p below level n, that is:

- $p\downarrow_0 = \mathbf{0}$
- $(\sum ap_a)\downarrow_{n+1} = \sum a(p_a)\downarrow_n$

Note that $p\downarrow_n$ is always a finite process having depth at most n. The proof of level continuity of $\approx_{RS}$ is much more complicated that the corresponding one in [de Frutos-Escrig and Gregorio-Rodríguez, 2005], since for any finite depth we have now infinitely many different trees (although most of them are bisimilar to each other, of course!). Thus we have to prove that whenever using global transitions to simulate plain transitions we can limit the use of the duplication rule at any depth.

The expansion of q wrt p is a process that is obtained by reiterated replication of some subterms of q, taking into account the form of p.

Definition 22. Let p and q be finite processes, with $p = \sum_{a\in Act}\sum_{i\in I_a} ap_a^i$ and $q = \sum_{a\in Act}\sum_{j\in J_a} aq_a^j$. We define the *expansion of q with respect to p*, $\mathrm{Xp}(p,q)$, as follows:

$\mathrm{Xp}(\mathbf{0},p) = \mathbf{0}$
$\mathrm{Xp}(\sum_{a\in Act}\sum_{j\in J_a} aq_a^j\ ,\ \sum_{a\in Act}\sum_{i\in I_a} ap_a^i) = \sum_{a\in Act}\sum_{j\in J_a, i\in I_a^+} a\ \mathrm{Xp}(q_a^j, p_a^i)$

where I_a^+ is defined by $I_a^+ = I_a$ if $I_a \neq \emptyset$, otherwise $I_a^+ = \{*\}$ and $p_a^* = \mathbf{0}$.

Proposition 23. *If $k \sqsubseteq l$ then $Xp(p\downarrow_l, q\downarrow_l)\downarrow_k = Xp(p\downarrow_k, p\downarrow_k)$*

Using the proposition above we can extend Definition 22 to finitely branching tree processes as follows:

Definition 24. Let p and q be infinite finitely branching tree processes. We define $\mathrm{Xp}(q,p)$ as the process whose finite cuts are given by

$$\mathrm{Xp}(q,p)\downarrow_k = \mathrm{Xp}(q\downarrow_k, p\downarrow_k)$$

Using the relative expanded form of processes we can limit the use of the duplication rule G_{nd}, when proving $=_{RS}$ by means of global bisimulations. This is done using the following proposition.

Proposition 25. *Let p and q be finite processes, then we have $p \sqsubseteq_{RS} q$ iff $Xp(q,p) \rightsquigarrow_{nd}^* p$*

Proof. For the right to left side, we have $q =_B \mathrm{Xp}(q,p)$ and, since $\rightsquigarrow^*$ is correct with respect to $\sqsubseteq_{RS}$, we can conclude that $p \sqsubseteq_{RS} q$.

For the left to right side we will proceed by induction on the depth of processes.

– If $p = \mathbf{0}$ then $q = \mathbf{0}$ and therefore $\mathrm{Xp}(q,p) = \mathbf{0}$.

– If $p = \sum\sum ap_a^i$, we know that for all $i \in I_a$ there exists $j_i \in J_a$ such that $p_a^i \sqsubseteq_{RS} q_a^{j_i}$. Then, by induction hypothesis $\mathrm{Xp}(q_a^{j_i}, p_a^i) \rightsquigarrow_{nd}^* p_a^i$, and therefore,

$$\begin{array}{rl} \mathrm{Xp}(q,p) = & \sum_{a\in Act}\sum_{j\in J_a, i\in I_a^+} a\mathrm{Xp}(q_a^j, p_a^i) \quad \rightsquigarrow^* \\ & \sum_{a\in Act}\sum_{j\in\{j_i \mid i\in I_a^+\}, i\in I_a^+} a\mathrm{Xp}(q_a^{j_i}, p_a^i) \rightsquigarrow^* \\ & \sum_{a\in Act}\sum_{i\in I_a^+} ap_a^i \quad = \ p \end{array}$$

Corollary 26. *Let p and q be finite processes. If $p \sqsubseteq_{RS} q$ and $p \rightsquigarrow^* p' \xrightarrow{a} p''$, then we have $q \rightsquigarrow^* Xp(q,p) \rightsquigarrow^* p \rightsquigarrow^* p' \xrightarrow{a} p''$.*

Corollary 27. *Let p and q be infinite finitely branching tree processes, then we have*

$$p \sqsubseteq_{RS} q \text{ if and only if } Xp(q,p) \rightsquigarrow_{nd}^{\infty} p$$

Proof. Since $\sqsubseteq_{RS}$ is level continuous we have $p \sqsubseteq_{RS} q$ iff for all k $p\downarrow_n \sqsubseteq_{RS} q\downarrow_n$ iff $\mathrm{Xp}(q\downarrow_k, p\downarrow_k) \rightsquigarrow_{nd}^* p\downarrow_k$. We could have several ways to make this reduction, but we are interested in the one that will be consistent with the reductions needed to get $\mathrm{Xp}(q\downarrow_l, p\downarrow_l) \rightsquigarrow_{nd}^* p\downarrow_l$ for any $l > k$.

Such a consistent way exists indeed since if we just consider the applications of $\rightsquigarrow_{nd}$ in nodes that are at depth less or equal than k in any such a reduction and cut the involved trees at that depth, we get another reduction $\mathrm{Xp}(q\downarrow_l, p\downarrow_l)\downarrow_k = \mathrm{Xp}(q\downarrow_k, p\downarrow_k) \rightsquigarrow_{nd}^* p\downarrow_k$. Since we have such a reduction for any $l \geq k$, $q\downarrow_k$ is a finite tree, and the application of $\rightsquigarrow_{nd}$ reduces the size of the tree, we conclude by a compactness argument, that there exists such a consistent reduction. Clearly, by projecting any such consistent reduction over a lower level we get another consistent reduction, and since for any level there are only a finite number of possible reductions, by applying Köning's lemma we obtain an infinite sequence of nested reductions, whose limit is the desired reduction $\mathrm{Xp}(q,p) \rightsquigarrow_{nd}^{\infty} p$.

Theorem 28. *The equivalence relation $\approx_{RS}$ is level continuous, that is $p \approx_{RS} q$ iff $\forall k \quad p\downarrow_k \approx_{RS} q\downarrow_k$.*

Proof. Left to right implication. It is clear that if S is a bisimulation in $\widetilde{\mathcal{F}}$ then $S_f = \{(p\downarrow_k, q\downarrow_k) \mid pSq\}$ is also a bisimulation in $\widetilde{\mathcal{F}}$.

Right to left implication. Let p and q be processes such that for all k we have $p\downarrow_k \approx_{RS} q\downarrow_k$. Applying Proposition 25 we have that $\mathrm{Xp}(q\downarrow_k, p\downarrow_k) \rightsquigarrow_{nd}^* p\downarrow_k$, and then we can repeat the reasonings in the proof of Corollary 27 to conclude that $\mathrm{Xp}(q,p) \rightsquigarrow_{nd}^{\infty} p$ and therefore $q \rightsquigarrow^{\infty} p$. Symmetrically, we can obtain $p \rightsquigarrow^{\infty} q$ and then, by definition of $\widetilde{\mathcal{F}}$, we immediately conclude that $p \approx_{RS} q$.

After these previous results we can now conclude that global ready bisimulation characterises ready simulation also for infinite finitary trees.

Theorem 29. *Both for finite and infinite finitely branching tree processes p and q we have $p =_{RS} q$ iff $p \approx_{RS} q$.*

Proof. It is an immediate consequence of the fact that $=_{RS}$ is level continuous, Theorem 28, together with the results in Theorem 10.

7 Conclusions and Future Work

In this paper we have presented the notion of global bisimulation, that are just plain bisimulations that take into account new global transitions together with the original action transitions.

We have found the set of global transitions needed to define a global bisimulation that generates an equivalence relation with the same discriminatory power as the ready similarity. This global bisimulation is a direct symmetric characterisation of the ready similarity, which was originally defined as the kernel of a ready simulation ordering. As a consequence, we have now the possibility of proving ready similarity of two processes in a direct way, using coinductive methods.

Ready similarity was a good candidate when looking for a semantics different to bisimulation to be characterised using global bisimulation, because it is quite close to it both in the way it is defined, and in the discriminatory power. However, studying this example we have found that it is not necessary to start from a simulation based semantics to get an equivalent global bisimulation. The reason for this is that in order to define the adequate global bisimulation what has proved to be very useful is the axiomatic characterisation of the original semantics. We have used it both to define the global transitions and to prove the equivalence between the original semantics and the defined global bisimulation semantics. Following these ideas we have also obtained global bisimulations characterising other semantics, such as refusals and traces, that are quite different, both in formulation and in discriminatory power, from bisimulation.

Therefore, we have proved that global bisimulations provide a general framework to relate both intensional and extensional semantics. We have seen that the differences between the semantics are expressed in terms of the corresponding way to treat non-determinism, expressed by the global rules we need to characterise them.

Recently, following a slightly different approach, we have found that indeed is possible to systematically generate coalgebraic definitions of the different semantics of concurrent processes by means of what we call bisimulations upto [de Frutos-Escrig and Gregorio-Rodríguez, 2005].

The main differences between both approaches is that global bisimulations have a more operational flavour, since global transitions are defined by simple transformations of the original transition system, such as duplication of transitions, removal or simple combinations, as delaying. By means of them we express

how non-determinism is treated in each case. Instead, bisimulations up-to incorporate an arbitrary preorder which can be used to transform the simulating process q before the emulation of the corresponding transition. In this way, we use mainly algebraic arguments instead of the operational ones in the global bisimulation framework, and we do not stress the symmetric definition as plain bisimulation in a modified transition system, although it is indeed possible, to get such a characterisation, but not needed for the formal developments.

Non-determinism also appears when internal moves are allowed. Some relations between global bisimulations and weak bisimulations were already studied in [de Frutos-Escrig et al., 1999] and now we plan also to continue this research in a more systematic way taking advantage of the new ideas that we have exposed in this paper.

References

[Abramsky, 1987] Abramsky, S. (1987). Observational equivalence as a testing equivalence. *Theoretical Computer Science*, 53(3):225–241.

[Bloom et al., 1988] Bloom, B., Istrail, S., and Meyer, A. R. (1988). Bisimulation can't be traced. In *Proceedings of the 15th ACM SIGPLAN-SIGACT symposium on Principles of programming languages*, pages 229–239. ACM Press.

[Bloom et al., 1995] Bloom, B., Istrail, S., and Meyer, A. R. (1995). Bisimulation can't be traced. *Journal of the ACM*, 42(1):232–268.

[Bloom and Meyer, 1992] Bloom, B. and Meyer, A. R. (1992). Experimenting with process equivalence. *Theoretical Computer Science*, 101(2):223–237.

[Brookes et al., 1984] Brookes, S. D., Hoare, C., and Roscoe, A. W. (1984). A theory of communicating sequential processes. *J. ACM*, 31(3):560–599.

[Cleaveland and Hennessy, 1992] Cleaveland, R. and Hennessy, M. (1992). Testing equivalence as a bisimulation equivalence. *Formal Aspects of Computing*, 3:1–21.

[Cleaveland et al., 1993] Cleaveland, R., Parrow, J., and Steffen, B. (1993). The concurrency workbench: a semantics-based tool for the verification of concurrent systems. *ACM Trans. Program. Lang. Syst.*, 15(1):36–72.

[Cleaveland and Smolka, 1996] Cleaveland, R. and Smolka, S. A. (1996). Strategic directions in concurrency research. *ACM Computing Surveys.*, 28(4):607–625.

[de Frutos-Escrig and Gregorio-Rodríguez, 2005] de Frutos-Escrig, D. and Gregorio-Rodríguez, C. (2005). Bisimulations up-to for the linear time-branching time spectrum. In *CONCUR 2005 - Concurrency Theory, 16th International Conference*, volume 3653 of *Lecture Notes in Computer Science*, pages 278–292. Springer.

[de Frutos-Escrig et al., 1999] de Frutos-Escrig, D., López, N., and Núñez, M. (1999). Global timed bisimulation: An introduction. In *Formal Methods for Protocol Engineering and Distributed Systems, FORTE XII / PSTV XIX*, pages 401–416. Kluwer Academic Publishers.

[Gardiner, 2003] Gardiner, P. (2003). Power simulation and its relation to traces and failures refinement. *Theoretical Computer Science*, 309:157–176.

[Glabbeek, 2001] Glabbeek, R. J. v. (2001). *Handbook of Process Algebra*, chapter The Linear Time – Branching Time Spectrum I: The Semantics of Concrete, Sequential Processes, pages 3–99. Elsevier.

[Gregorio-Rodríguez and Nuñez, 1999] Gregorio-Rodríguez, C. and Nuñez, M. (1999). Denotational semantics for probabilistic refusal testing. In *PROBMIV'98 First International Workshop on Probabilistic Methods in Verification*, volume 22 of *Electronics Notes in Theoretical Computer Science*, page 27. Elsevier.

[Groote and Vaandrager, 1992] Groote, J. F. and Vaandrager, F. W. (1992). Structured operational semantics and bisimulations as a congruence. *Information and Computation*, 100(2):202–260.

[Hennessy, 1988] Hennessy, M. (1988). *Algebraic Theory of Processes*. MIT Press.

[Hennessy and Milner, 1985] Hennessy, M. and Milner, R. (1985). Algebraic laws for nondeterminism and concurrency. *J.ACM*, 32:137–161.

[Hoare, 1985] Hoare, C. (1985). *Communicating Sequential Processes*. Prentice Hall.

[Kanellakis and Smolka, 1990] Kanellakis, P. C. and Smolka, S. A. (1990). CCS expressions, finite state processes, and three problems of equivalence. *Information and Computation*, 86(1):43–68.

[Larsen and Skou, 1991] Larsen, K. G. and Skou, A. (1991). Bisimulation through probabilistic testing. *Information and Computation*, 94(1):1–28.

[Milner, 1989] Milner, R. (1989). *Communication and Concurrency*. Prentice Hall.

[Olderog and Hoare, 1986] Olderog, E. R. and Hoare, C. (1986). Specification-oriented semantics for communicating processes. *Acta Inf.*, 23(1):9–66.

[Paige and Tarjan, 1987] Paige, R. and Tarjan, R. E. (1987). Three partition refinement algorithms. *SIAM Journal of Computing*, 16(6):973–989.

[Park, 1981] Park, D. M. (1981). Concurrency and automata on infinite sequences. In *Theoretical Computer Science, 5th GI-Conference*, volume 104 of *Lecture Notes in Computer Science*, pages 167–183. Springer.

[Phillips, 1987] Phillips, I. (1987). Refusal testing. *Theoretical Computer Science*, 50(3):241–284.

[Plotkin, 1981] Plotkin, G. D. (1981). A structural approach to operational semantics. Technical Report DAIMI FN-19, Computer Science Department, Aarhus University.

[Pnueli, 1985] Pnueli, A. (1985). Linear and branching structures in the semantics and logics of reactive systems. In *12th Int. Colloq. on Automata, Languages and Programming ICALP'85*, volume 194 of *Lecture Notes in Computer Science*, pages 15–32. Springer.

[Stirling, 1998] Stirling, C. (1998). The joys of bisimulation. In *MFCS'98*, volume 1450 of *Lecture Notes in Computer Science*, pages 142–151. Springer-Verlag.