Applicability of Fair Simulation

Doron Bustan and Orna Grumberg
Computer Science Department
Technion, Haifa 32000, Israel
email: {orna,doron2}@cs.technion.ac.il

Abstract

In this paper we compare four notions of fair simulation: direct [6], delay [8], game [12], and exists [11]. Our comparison refers to three main aspects: The time complexity of constructing the fair simulation, the ability to use it for minimization, and the relationship between the fair simulations and universal branching-time logics.

We developed two practical applications that are based on this comparison. The first is an efficient approximated minimization algorithm for the delay,game,exists simulations. The second is a new implementation for the assume-guarantee modular framework presented in [11]. The new implementation significantly improves the complexity of the framework.

1 Introduction

Temporal logic model checking is a method for verifying finite-state systems with respect to propositional temporal logic specifications. The method is fully automatic and quite efficient in time, but is limited by its high space requirements. Many approaches for overcoming the *state explosion problem* of model checking have been suggested, including abstraction, partial order reduction, modular methods, and symmetry [4]. These approaches are often based on the idea that the model of the verified system can be replaced by a more abstract model, smaller in size. The abstract and concrete models are sufficiently similar so that properties that are verified on the abstract model can be considered true for the concrete one. This idea is often formalized by relating models with the *simulation preorder* [17], in which the greater, more abstract model has "more behaviors," and the verified properties are written in a universal branching time logic such as ACTL or ACTL* [11].

It often happens that during the construction of a reduced abstract model some unrealistic infinite behaviors are added. A common way to avoid these behaviors is to add fairness constraints to distinguish between wanted (fair) and unwanted (unfair) behaviors and to exclude unfair behaviors from consideration.

The simulation preorder does not distinguish between fair and unfair behaviors. It is therefore desirable to find an alternative definition that relates only fair behaviors of the two models. This task, however, is not uniquely defined. Indeed, several distinct notions of *fair simulation* have been suggested in the literature [6, 8, 12, 11].

Researchers have addressed the question of which notion of fair simulation is preferable. In [12], some of these notions are compared with respect to the complexity of checking for fair simulation. In [8], a different set of notions is compared with respect to two criteria: The complexity of constructing the preorder, and the ability to minimize a fair model by constructing a quotient model that is language equivalent to the original one.

In this paper we make a broader comparison of four notions of fair simulation: direct [6], delay [8], game [12], and exists [11]. We refer to several criteria that emphasize the advantages of each of the notions. The results of the comparison are summarized in a table in Figure 1.

We developed two practical applications that are based on the comparison. The first is an efficient approximated minimization algorithm for the delay, game and exists simulations. For these preorders, a unique equivalent smallest model does not exist. Therefore, an approximation is appropriate.

In addition, we suggest a new implementation for the *assume-guarantee* [10, 13, 18, 19] modular framework presented in [11]. The new implementation, based on the game simulation rather than the exists simulation, significantly improves the complexity of the framework.

Our comparison refers to three main aspects of fair simulation. The first is the time complexity of constructing the preorder. There, we mainly summarize results of other works (see Figure 1). We see that constructing the direct, delay, and game simulations is polynomial in the number of states n and the number of transitions m [8]. In contrast, constructing the exists simulation is PSPACE-complete [15] , which is a great disadvantage.

The second aspect that we consider is the ability to use the preorder for minimization. We say that two models are *equivalent* with respect to a preorder if each is smaller by the preorder than the other. The goal of minimization is to find the smallest in size model that is equivalent with respect to the preorder to the original one[1].

In [3] it has been shown that for every model with no fairness constraints there exists a unique smallest in size model which is simulation equivalent to it. The minimization algorithm that constructs this smallest in size model [3] identifies and eliminates two types of redundancies in the given model. One is the existence of equivalent states. This redundancy is eliminated by constructing a *quotient model*. The other is the existence of a successor of a state whose behavior is contained in the behavior of another successor of the same state. Such a state is called *a little brother*. This redundancy is eliminated by *disconnecting little brothers*.

We thus examine, for each of the fair simulation preorders, the following three questions. Given a model M:

- 1. Is there a unique smallest in size model that is simulation equivalent to M?
- 2. Is the quotient model of M simulation equivalent to M?
- 3. Is the result of disconnecting little brothers in M simulation equivalent to M?

Our examination (see Figure 1) leads to a new minimization algorithm that uses the direct and delay simulations as approximations for the game

[1]Note that this is a stronger criterion than the one used in [8], where only language equivalence is required.

and exists simulations. The new algorithm obtains a better reduction than the algorithm suggested in [8].

The third aspect that we investigate is the relationship between the simulation preorders and universal branching-time logics. A basic requirement of using a preorder in verification is that it preserves the specification logic, i.e., if $M_1 \leq M_2$ then, for every formula ϕ in the logic, $M_2 \models \phi$ implies $M_1 \models \phi$. Indeed, all four notions of fair simulation satisfy this requirement. A stronger requirement is that the preorder have a *logical characterization* by some logic. This means that $M_1 \leq M_2$ **if and only if** for every formula ϕ in the logic, $M_2 \models \phi$ implies $M_1 \models \phi$.

Logical characterization is useful in determining if model M_2 can be used as an abstraction for model M_1, when the logic $\mathcal{L}$ should be preserved. If the preorder $\leq$ is logically characterized by $\mathcal{L}$ then checking $M_1 \leq M_2$ is a necessary and sufficient condition and will never give a false negative result.

Another important relationship between a logic and a preorder is the existence of a *maximal model* $\mathcal{T}_\phi$ for a formula ϕ with respect to the preorder. The maximal model $\mathcal{T}_\phi$ for a formula ϕ is such that for every model M', $M' \leq \mathcal{T}_\phi$ if and only if $M' \models \phi$. Maximal models are used as tableaux in the framework described in [11] for the *assume-guarantee paradigm*. The assume-guarantee is an inductive modular verification paradigm in which the environment of the verified part can be represented by a formula. The result method is a proof schema which is based on the modular structure of the system.

In [11], a semi-automatic framework for the assume-guarantee paradigm is presented. The framework uses the exists preorder and is defined with respect to the logic ACTL. It uses a tableau to represent an ACTL formula. This tableau is the maximal model for the formula with respect to the exists preorder.

In this work we show that there is also a maximal model for ACTL formulas with respect to the game simulation. In addition, we show that other conditions required for a sound implementation of the assume-guarantee paradigm hold for the game simulation. Once the game simulation replaces the exists simulation, the complexity of the implementation is dramatically reduced.

The results of our comparison are presented in the table in Figure 1. The proofs of the claims for which no citation is given appear in the next sections. The rest of the paper is organized as follows: In Section 2 we

[3] In [8] it is shown that the quotient model is language equivalent to the original model.

		minimization			relation to logic	
notion	time complexity of constructing the preorder	unique smallest model	quotient model	little brothers	has logical characterization	max model
Direct	$O(m \cdot n)$ [8]	*true*	true	*true*	*false*	*false*
Delay	$O(m \cdot n^3)$ [8]	*false*	true [3]	*false*	*false*	*false*
Game	$O(m \cdot n^3)$ [8]	*false*	false [8]	*false*	$\forall AFMC$ [12]	*true*
Exists	PSPACE complete [15]	*false*	*false*	*false*	$ACTL^*$	true [11]

Figure 1: The properties of the different notions of fair simulation

define the simulation preorder and the different notions of fair simulation. Section 3 investigates simulation minimization. For each of the fair simulations we check whether there exists a unique minimal structure, and whether constructing a quotient structure or disconnecting little brothers results in an equivalent structure. We then present a new minimization algorithm for the game and exists simulations. Section 4 investigates the relationships between fair simulation and logic. Each notion is checked for logical characterization and for the existence of a maximal structure. In Section 5 we prove that the game simulation can replace the exists simulation in the implementation of the assume-guarantee paradigm. Finally, in Section 6 we discuss some conclusions.

2 Preliminaries

Let AP be a set of atomic propositions. We model systems by a *fair Kripke structure* M over AP, $M = (S, R, S_0, L, F)$, where S is a finite set of states, $S_0 \subseteq S$ is a set of initial states, and $R \subseteq S \times S$ is the transition relation, which must be *total*. This means that for every state $s \in S$ there is a state $s' \in S$ such that $(s, s') \in R$ (states which do not satisfy this condition are deleted). $L : S \to 2^{AP}$ is a function that labels each state with the set of atomic propositions true in that state, and $F \subseteq S$ is a set of fair states.

Let s be a state in a Kripke structure M. A *trace* in M starting from s is an infinite sequence of states $\rho = s_0 s_1 s_2 \ldots$ such that $s_0 = s$, and for every $i \geq 0$, $(s_i, s_{i+1}) \in R$. The i-th state of trace ρ is denoted ρ^i. In order to capture the infinite behavior of ρ, we define

[3] Here, we show that they are delay equivalent.

$$\inf(\rho) = \{\, s \mid s = \rho^i \text{ for infinitely many } i \,\}.$$

We say that a trace ρ is *fair* according to the fair set F iff $\inf(\rho) \cap F \neq \emptyset$.

In this work we refer to two branching-time logics, ACTL* and ACTL [11].

First, we define CTL* formulas in negation normal form, namely, negation is applied only to atomic propositions. CTL* contains trace formulas and state formulas and is defined inductively:

- Let p be an atomic proposition, then p and $\neg p$ are both state formulas and trace formulas.
- Let φ and ψ be trace formulas, then
  - $(\varphi \wedge \psi)$ and $(\varphi \wedge \psi)$ are trace formulas.
  - $\mathbf{X}\varphi$, $(\varphi \mathbf{U} \psi)$ and $(\varphi \mathbf{R} \psi)$ are trace formulas.
  - $A\varphi$ and $E\varphi$ are state formulas.
- Let φ and ψ be state formulas, then
  - $(\varphi \vee \psi)$ and $(\varphi \wedge \psi)$ are state formulas.
  - $\mathbf{X}\varphi$, $(\varphi \mathbf{U} \psi)$ and $(\varphi \mathbf{R} \psi)$ are trace formulas.

Next we define the semantics of CTL* with respect to fair Kripke structures. A state formula ϕ is satisfied by a structure M at state s, denoted $M, s \models \phi$, if the following holds (M is omitted if clear from the context):

- For $p \in AP$, $s \models p$ iff $p \in L(s)$; $s \models \neg p$ iff $p \notin L(s)$.
- $s \models \phi \wedge \psi$ iff $s \models \phi$ and $s \models \psi$; $s \models \phi \vee \psi$ iff $s \models \phi$ or $s \models \psi$.
- $s \models \mathbf{A}\varphi$ iff for every fair trace ρ from s, $\rho \models \varphi$.
- $s \models \mathbf{E}\varphi$ iff there exists a fair trace ρ from s, such that $\rho \models \varphi$.

A trace formula φ is satisfied by a trace ρ, denoted $\rho \models \varphi$, if the following holds

- $\rho \models \mathbf{X}\,\phi$ iff $\rho^1 \models \phi$.
- $\rho \models \mathbf{A}[\varphi\, \mathbf{U}\, \psi]$ iff for some $i \geq 0$, $\rho^i \models \psi$ and for all $j < i$, $\rho^j \models \phi$.
- $\rho \models \mathbf{A}[\phi\, \mathbf{R}\, \psi]$ iff for all $i \geq 0$, if for every $j < i$, $\rho^j \not\models \phi$ then $\rho^i \models \psi$.

ACTL* is the universal fragment of CTL*where the only trace quantifier allowed is **A**. ACTL is a subset of ACTL* where every temporal operator is immediately proceeded by the **A** quantifier.
We say that $M \models \phi$ iff for every initial state $s_0 \in S_0$, $M, s_0 \models \phi$.

2.1 Simulation and fair simulation

We start by defining the simulation relation over Kripke structures with $F = S$ (Kripke structures with trivial fairness constraints).

Definition 2.1 *Given two structures M_1 and M_2 over AP, a relation $H \subseteq S_1 \times S_2$ is a simulation relation [17] over $M_1 \times M_2$ iff the following conditions hold:*

1. *For every $s_{01} \in S_{01}$ there exists $s_{02} \in S_{02}$ such that $(s_{01}, s_{02}) \in H$.*
2. *For all $(s_1, s_2) \in H$,*
   (a) $L_1(s_1) = L_2(s_2)$ and
   (b) $\forall s_1'[(s_1, s_1') \in R_1 \rightarrow \exists s_2'[(s_2, s_2') \in R_2 \wedge (s_1', s_2') \in H]]$.

M_2 simulates M_1 (denoted by $M_1 \leq M_2$) if there exists a simulation relation H over $M_1 \times M_2$. We say that M_1 and M_2 are *simulation equivalent if* $M_1 \leq M_2$ and $M_2 \leq M_1$. Similarly, $(s_1, s_2) \in H$, is denoted $s_1 \leq s_2$ and s_1 and s_2 are equivalent if $s_1 \leq s_2$ and $s_2 \leq s_1$. This equivalence is denoted $s_1 \equiv s_2$.

The relation $\leq$ is a preorder on the set of structures. This means that it is reflexive and transitive. In [11, 2] it is shown that $M_1 \leq M_2$ iff, for every ACTL* formula ψ (with atomic propositions in AP), $M_2 \models \psi$ implies $M_1 \models \psi$. Thus, the simulation relation has logical characterization over structures with trivial fairness constraints.

Next, we define the different notions of fair simulation. The first notion is the direct simulation, which is the most straightforward extension of the ordinary simulation.

Definition 2.2 *$H \subseteq S_1 \times S_2$ is a* direct simulation relation *[6] ($\leq_{di}$) over $M_1 \times M_2$ iff it satisfies the conditions of Definition 2.1, except that here 2a is replaced by:*
$2(a')$ $L_1(s_1) = L_2(s_2)$ and $s_1 \in F_1$ implies $s_2 \in F_2$.

We now define the exists simulation:

Definition 2.3 *[11] $H \subseteq S_1 \times S_2$ is an* exists simulation *($\leq_\exists$) over $M_1 \times M_2$ iff it satisfies the conditions of Definition 2.1, except that here 2b is replaced by:*
$2(b')$ for every fair trace ρ_1 from s_1 in M_1 there exists a fair trace ρ_2 from s_2 in M_2 such that for all $i \in IN$, $(\rho_1^i, \rho_2^i) \in H$.[4]

The next definitions are based on games over Kripke structures. We start with a game that characterizes the simulation over structures with trivial fairness constraints. Given two Kripke structures M_1, M_2, we define a game of two players over M_1, M_2. The players are called the adversary and the protagonist, where the adversary plays on M_1 and the protagonist plays on M_2.

Definition 2.4 *Given two Kripke structures, M_1 and M_2, a* simulation game *consists of a finite or infinite number of rounds. At the beginning, the adversary selects an initial state s_{01} in M_1, and the protagonist responds by selecting an initial state s_{02} in M_2 such that $L_1(s_{01}) = L_2(s_{02})$. In each round, assume that the adversary is at s_1 and the protagonist is at s_2. The adversary then moves to a successor s_1' of s_1 on M_1, after which the protagonist moves to a successor s_2' of s_2 on M_2 such that $L_1(s_1') = L_2(s_2')$.*

If the protagonist does not have a matching state, the game terminates and the protagonist fails. Otherwise, if the protagonist always has a matching successor to move to, the game proceeds ad infinitum for ω rounds and the protagonist wins. The adversary wins iff the protagonist fails.

Definition 2.5 *Given two Kripke structures M_1 and M_2, a* strategy *π of the protagonist is a function $\pi : (S_1 \times S_2 \to S_2) \cup (S_{01} \times \{\perp\} \to S_{02})$. The function π should satisfy the following: If $s_2' = \pi(s_1', s_2)$ then $(s_2, s_2') \in R_2$.*

The protagonist plays according to a strategy π if when the adversary initially selects $s_{0_1} \in S_{0_1}$, the protagonist selects $s_{0_2} = \pi(s_{0_1}, \perp)$ and, for every round i, when the adversary moves to s_1' and the protagonist is in s_2, the protagonist moves to $s_2' = \pi(s_1', s_2)$. π is a winning strategy for the protagonist if the protagonist wins whenever it plays according to π.

We can now present an alternative definition to the simulation preorder. This definition is equivalent to Definition 2.1 [12].

Definition 2.6 *Given two Kripke structures, M_1 and M_2, M_2 simulates M_1 ($M_1 \leq M_2$) iff the protagonist has a winning strategy in a simulation game over M_1, M_2.*

[4] In such a case we use the notation $(\rho_1, \rho_2) \in H$.

In order to extend the simulation game to fair simulation, we add a winning condition which refers to the infinite properties of the game. We then give two additional definitions of fair simulation, the delay ($\leq_{de}$) and the game ($\leq_g$) simulations.

Definition 2.7 *[8] The protagonist* delay wins *a game over two fair Kripke structures M_1 and M_2 iff the game is played for infinitely many rounds. Moreover, whenever the adversary reaches a fair state then the protagonist reaches a fair state within a finite number of rounds.*

Definition 2.8 *[12] The protagonist* game wins *a game over two fair Kripke structures M_1 and M_2 iff the game is played for infinitely many rounds. Moreover, if the adversary moves along a fair trace, then the protagonist moves along a fair trace as well.*

We say that π is a delay/game winning strategy for the protagonist if the protagonist delay/game wins whenever it plays according to π.

Definition 2.9 *[12, 8] Given two fair Kripke structures, M_1 and M_2, M_2 delay/game simulates M_1 iff the protagonist has a delay/game winning strategy over M_1, M_2.*

Definitions 2.2,2.3 and 2.9 are extensions of Definition 2.1 and its equivalent Definition 2.6. Consequently, on structures with trivial fairness constraints ($F = S$), all four definitions are equivalent. In [12, 8] the following relationships over the fair simulation preorders are shown:

$$M_1 \leq_{di} M_2 \;\Rightarrow\; M_1 \leq_{de} M_2 \;\Rightarrow\; M_1 \leq_g M_2 \;\Rightarrow\; M_1 \leq_\exists M_2.$$

Note that the definitions of game/exists simulation are not limited to specific types of fairness constraints. They hold even if M_1 and M_2 have different types of fairness constraints. Finally, we extend the delay/game simulations for states.

Definition 2.10 *For all states s_1 and s_2 in a structure M, $s_1 \leq_{de/g} s_2$ if the protagonist has a winning delay/game strategy in a game over $M \times M$ where the adversary starts at s_1 and the protagonist starts at s_2.*

3 Simulation minimization

For structures with trivial fairness constraints ($F = S$), two forms of redundancy are considered [3]. These redundancies are handled in [3], by first constructing a quotient structure that results in a structure without equivalent states and then disconnecting *little brothers* to eliminate the other redundancy. For structures with trivial fairness constraints, eliminating these redundancies results in a unique, smallest in size structure that is simulation equivalent to the original structure [3].

The following lemma is a direct consequence of the result in [3] if we refer to states in F as having additional labeling.

Lemma 3.1 *For every structure, there exists a unique, smallest in size structure that is direct simulation equivalent to it.*

The proof of Lemma 3.1 and the construction of the smallest structure can be obtained as in [3]. Unfortunately, performing the same operations for the other notions of fair simulations might result in an inequivalent structure. In this section we investigate minimization with respect to each notion of fair simulation. We start by checking whether the quotient structure is equivalent to the original one. Next we check whether it is safe to disconnect little brothers. We then determine whether there exists a unique smallest in size equivalent structure. Finally, we use the results of this section to suggest a new and better minimizing algorithm.

In this section we use language equivalence and language containment.The definitions are given below.

Definition 3.2

- *The language of* s_1 *is contained in the language of* s_2 ($s_1 \subseteq s_2$) *if for every fair trace* ρ_1 *from* s_1 *there is a fair trace* ρ_2 *from* s_2 *such that* $\forall i \geq 0,\ L(\rho_1^i) = L(\rho_2^i)$.
- $M_1 \subseteq M_2$ *if for every fair trace starting at an initial state* $s_{01} \in S_{01}$ *there is a fair trace starting at an initial state* $s_{02} \in S_{02}$ *such that* $\forall i \geq 0,\ L_1(\rho_1^i) = L_2(\rho_2^i)$.
- M_1 *is language equivalent to* M_2 *if* $M_1 \subseteq M_2$ *and* $M_2 \subseteq M_1$.

Clearly, all notions of fair simulation imply language containment.

3.1 Quotient structure

The quotient structure is the result of unifying all equivalent states into equivalence classes. Recall that states s_1 and s_2 are equivalent if $s_1 \leq s_2$ and $s_2 \leq s_1$. The equivalence classes are the states of the quotient structure. There is a transition from one equivalence class to another iff there exists a transition from a state in the former to a state in the latter. An equivalence class is initial if it contains an initial state and is fair if it contains a fair state. For the delay simulation, we present the following lemma.

Lemma 3.3 *Let M^Q be the quotient structure of a structure M. Then $M \equiv_{de} M^Q$.*

The proof of Lemma 3.3 appears in Appendix A and is similar to the proof in [9].

In [8] it is shown that the quotient structure with respect to game simulation is not equivalent to the original one. We show that for every preorder $\leq_{\clubsuit}$ that lies between game simulation and language containment, the quotient structure with respect to this preorder might not be equivalent to the original structure.

Lemma 3.4 *Let $\leq_{\clubsuit}$ be any preorder such that for every M_1, M_2,*

$$M_1 \leq_g M_2 \Rightarrow M_1 \leq_{\clubsuit} M_2 \Rightarrow M_1 \subseteq M_2.$$

Then there exists a structure M whose quotient structure with respect to $\leq_{\clubsuit}$ is not equivalent to M with respect to $\leq_{\clubsuit}$.

Proof Consider the structure M_1 in Figure 2. States s_0 and s_2 are equivalent with respect to game simulation. This can be seen by considering a strategy that instructs the protagonist to move to the same state the adversary moves to. This strategy proves both directions of the game equivalence. Since $M_1 \leq_g M_2 \Rightarrow M_1 \leq_{\clubsuit} M_2$, s_0 and s_2 are also equivalent with respect to $\leq_{\clubsuit}$.

However, the quotient structure that is the result of unifying states s_0 and s_2 is not equivalent to M_1 with respect to $\leq_{\clubsuit}$. Since $M_1 \leq_{\clubsuit} M_2 \Rightarrow M_1 \subseteq M_2$, it is sufficient to prove that the quotient structure is not language equivalent to M_1: the language of M_1 contains all words in which both a and b occur infinitely often, but the language of the quotient structure contains the word a^ω.

Furthermore, there is no other definition of a quotient structure of M_1 that is language equivalent to M_1. Such a quotient structure contains two

states, one in which a is true and another in which b is true and at least one of the states is fair. Assume that the state where a is true is fair. We distinguish between two cases: If there exists a transition from this state to itself, then the language of the quotient structure includes a word where b occurs only finitely many times, a contradiction. Otherwise, the word $(aab)^\omega$ is not in the language of the quotient structure, a contradiction. Assuming that the state where b is true is fair, will lead to a contradiction in a similar way. □

Corollary 3.5 *For exists/game simulation, the quotient structure is not necessarily equivalent to the original structure.*

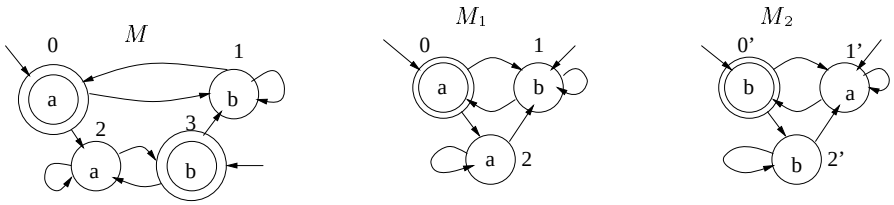

Figure 2: The structures M_1 and M_2 are equivalent to M with respect to game/exists simulation, and they are both minimal. Note that states 0 and 2 ($0'$ and $2'$) are equivalent but cannot be unified. (Double circles denote fair states.)

3.2 Disconnecting little brothers

A state s_2 is a *little brother* of another state s_3 if both states are successors of the same state s_1, $s_2 \leq s_3$, and $s_3 \not\leq s_2$. Little brothers s_2 is disconnected by removing the transition (s_1, s_2) from R.

Lemma 3.6 *Let $\leq_\spadesuit$ be a preorder such that*

$$M_1 \leq_{de} M_2 \Rightarrow M_1 \leq_\spadesuit M_2 \Rightarrow M_1 \subseteq M_2.$$

Assume that structure M' is the result of disconnecting little brothers in structure M with respect to $\leq_\spadesuit$. M' might not be equivalent to M with respect to $\leq_\spadesuit$.

Proof Consider the structure M_1 in Figure 3. State s_2 is a little brother of state s_1 with respect to $\leq_{de}$. This can be seen by considering the strategy that instructs the protagonist to move from state s_1 to state s_0 in the first round and to move to the same state the adversary moves to in the other rounds. This strategy shows that $s_2 \leq_{de} s_1$, because
$M_1 \leq_{de} M_2 \Rightarrow M_1 \leq_{\spadesuit} M_2$, $s_2 \leq_{\spadesuit} s_1$. Next, note that $s_1 \not\subseteq s_2$, since s_1 has a successor labeled c and s_2 does not. Thus $s_1 \not\leq_{\spadesuit} s_2$, and s_2 is a little brother of s_1 with respect to $\leq_{\spadesuit}$.

Next we show that the result of disconnecting s_2 from s_0 is not equivalent to M_1 with respect to $\leq_{\spadesuit}$. Since $M_1 \leq_{\spadesuit} M_2 \Rightarrow M_1 \subseteq M_2$, it is sufficient to show that the result of disconnecting s_2 from s_0 is not language equivalent to M_1. But this is true since disconnecting s_2 results in a structure with no fair traces from s_1. □

Corollary 3.7 *The structure that results when little brothers are disconnected with respect to delay/game/exists simulation might not be equivalent to the original structure with respect to delay/game/exists simulation.*

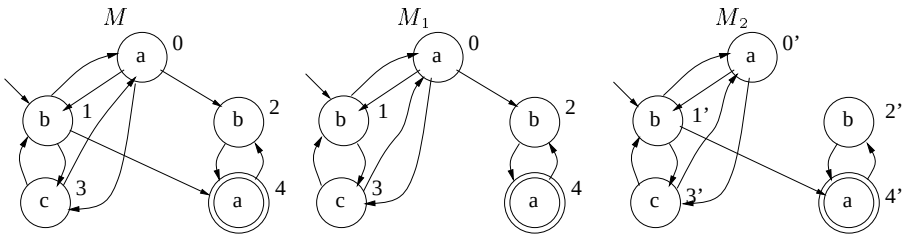

Figure 3: The structures M_1 and M_2 are equivalent with respect to delay/game/exists simulation to M, and they are both minimal. Note that state 2 ($4'$) is a little brother of 1 ($0'$) but cannot be disconnected.

3.3 Unique smallest in size structure

Lemma 3.8 *Let $\leq_{\spadesuit}$ be a preorder such that*

$$M_1 \leq_{de} M_2 \Rightarrow M_1 \leq_{\spadesuit} M_2 \Rightarrow M_1 \subseteq M_2.$$

Then there exists a structure M that has no unique smallest in size structure with respect to $\leq_{\spadesuit}$.

Proof Consider the structures in Figure 3. Structures M_1 and M_2 are delay equivalent but are not isomorphic. In order to see that $M_2 \leq_{de} M_1$, consider the strategy in which in every round the protagonist moves to the same state as the adversary, except for the transition from $1'$ to $4'$, when the protagonist moves to state 0. Similarly, we can show that $M_1 \leq_{de} M_2$. Since $M_1 \leq_{de} M_2 \Rightarrow M_1 \leq_{\spadesuit} M_2$, M_1 and M_2 are equivalent with respect to $\leq_{\spadesuit}$.

Next, we show that there is no smaller structure that is equivalent to M_1 and M_2 with respect to $\leq_{\spadesuit}$. Since $M_1 \leq_{\spadesuit} M_2 \Rightarrow M_1 \subseteq M_2$, it is sufficient to show that there is no smaller structure that is language equivalent to M_1 and M_2. Note that every equivalent structure must contain a strongly connected component with three states labeled $\{a\}$, $\{b\}$ and $\{c\}$. However, these states cannot be fair because there are no fair traces in M_1 and M_2 which have infinitely many states labeled $\{c\}$. Thus, there should be two other states labeled $\{a\}$ and $\{b\}$ on a fair, strongly connected component. Consequently, there have to be at least five states in any structure that is language equivalent to M_1 and M_2. □

Corollary 3.9 *There is no unique smallest in size structure with respect to delay/game/exists simulation.*

An interesting observation is that the minimization operations are not *independent*[5] [14]. For example, in structure M in Figure 2, states s_0 and s_1 are game/exists equivalent to states s_2 and s_3 respectively. Unifying states s_0 and s_2 results in structure M_2. Unifying states s_1 and s_3 results in structure M_1. Both structures are equivalent to M and neither can be further minimized. A similar phenomenon occurs in structure M of Figure 3: for delay/exists/game simulation, states s_4 and s_2 are little brothers of states s_0 and s_1 respectively. Disconnecting state s_4 from state s_1 results in M_1, and disconnecting state s_2 from state s_0 results in M_2. Again, both structures are equivalent to M, and neither structure can be further minimized.

3.4 An approximate minimization algorithm for delay/game/exists simulation

In [3], two efficient procedures for minimizing with respect to ordinary simulation are presented. In the previous sections we have shown that these

[5]Operations are not independent if one operation disables another.

procedures cannot be used for delay/game/exists simulation. Furthermore, we have shown that there is no equivalent unique smallest in size structure with respect to these simulations. As a result, we suggest an algorithm that performs some minimization but does not necessarily construct a minimal structure. Our algorithm uses the direct/delay simulations as an approximation of the game/exists simulation. The algorithm is presented in Figure 4. The first step results in $M' \equiv_{de} M$. The second step results in $M'' \equiv_{di} M'$.

Given a structure M,

1. Construct a quotient structure M' with respect to delay simulation.
2. Construct M'' by disconnecting little brothers in M' with respect to direct simulation.

Figure 4: Minimization algorithm for the delay/game/exists simulations.

Since direct simulation implies delay simulation, $M'' \equiv_{de} M$. M'' is also equivalent to M with respect to game/exists simulation. Thus, the algorithm combines the advantages of the direct and the delay simulations in order to produce a reduced structure that is equivalent with respect to delay/game/exists simulations to the original one. The complexity of the first step is $O(m \cdot n^3)$ [8], and of the second step $O(m \cdot n)$ [3]. Thus the total complexity of the algorithm is $O(m \cdot n^3)$.

4 Relating the simulation notions to logics

In this section we investigate the relationship between the different notions of fair simulation and the logics ACTL and ACTL*. First we check for each notion whether it has a logical characterization. Next we check whether there exists a maximal structure for ACTL with respect to each notion.

4.1 Logical characterization

Definition 4.1 *Logic $\mathcal{L}$ characterizes a preorder $\leq$ if for all structures M_1 and M_2, $M_1 \leq M_2$ if and only if for every formula ϕ in $\mathcal{L}$, $M_2 \models \phi$ implies $M_1 \models \phi$.*

In [11], it is shown that if $M_1 \leq_\exists M_2$, then the following property holds: $\forall \phi \in$ ACTL*, $M_2 \models \phi$ implies $M_1 \models \phi$. Since all other simulation notions

imply the exists simulation, this property holds for all of these notions.

We now investigate which of the fair simulations satisfy the other direction of logical characterization. We show that ACTL* characterizes the exists simulation but not the game/delay/direct simulation. On the other hand, ACTL does not characterize any of these notions.

First we prove that the exists simulation is characterized by the ACTL* logic. We prove that if $M \not\leq_\exists M'$, then there exists an ECTL* formula ϕ and an initial state s_0 of M such that $M, s_0 \models \phi$. Furthermore, for all initial states s'_0 of M', $M', s'_0 \not\models \phi$. This implies that there exists an ACTL* formula ψ which is equivalent to $\neg\phi$ such that $M' \models \psi$ but $M \not\models \psi$.

Our proof is similar to the proof in [1] for fair bisimulation. It is based on a different definition of fair simulation. This definition called rational simulation, is presented below.

Definition 4.2 *Let ρ be a trace through a Kripke structure M. ρ is a rational trace if $\exists N, K$ such that $\forall i (i > N \rightarrow \rho^i = \rho^{((i-N) mod\ K)+N})$.*

Thus, a rational trace is a trace with a prefix of length N followed by a cycle of length K.

Definition 4.3 *A state s is smaller by rational simulation than a state t ($s \leq_{rat} t$) if they lie in the coarsest preorder H that satisfies*

- *$L(s) = L(t)$.*
- *for every fair rational trace ρ_s starting at s there exists a fair rational trace ρ_t starting at t such that $(\rho_s, \rho_t) \in H$.*

$M \leq_{rat} M'$ if for every $s_0 \in S_0$ there exists $s'_0 \in S'_0$ such that $s_0 \leq_{rat} s'_0$.

Lemma 4.4 *Let s and t be states in structure M. If there exists a fair trace ρ_s from s such that for all fair traces ρ_t from t, $(\rho_s, \rho_t) \notin H$, then there exists a fair rational trace ρ_{sr} from s such that for all fair traces ρ_t from t, $(\rho_{sr}, \rho_t) \notin H$.*

The proof of Lemma 4.4 appears in Appendix B. Corollary 4.5 is straightforward from Lemma 4.4.

Corollary 4.5 *If $M \not\leq_\exists M'$ then $M \not\leq_{rat} M'$.*

In the proof we refer to one structure instead of two. This can be done when we refer to M'' which is the union of M and M' where, $S'' = S \cup S'$, (assume $S \cap S' = \emptyset$), $R'' = R \cup R'$, and $F'' = F \cup F'$. having deduced Corollary 4.5, it is now sufficient to prove the following:

Lemma 4.6 *For every structure M and states s and t, If for all ECTL* formulas ϕ, $M, s \models \phi$ implies $M, t \models \phi$, then $s \leq_{rat} t$.*

Proof We prove that $s \not\leq_{rat} t$ implies that there exists an ECTL* formula ϕ such that $M, s \models \phi$ but $M, t \not\models \phi$.

We first inductively define a sequence of preorders over $S \times S$.

Definition 4.7

- $(s,t) \in H_0$ *iff* $L(s) = L(t)$.
- $(s,t) \in H_{i+1}$ *iff for every fair rational trace ρ_s starting at s there exists a fair rational trace ρ_t starting at t such that $(\rho_s, \rho_t) \in H_i$.*

Note that for every $i \geq 0$, $H_{i+1} \subseteq H_i$. Thus, after at most $|S|^2$ preorders, we reach a fixpoint. We use H_∞ to denote the preorder at the fixpoint. It is easy to see that H_∞ is exactly the fair rational simulation.

For every state s, we define the following ECTL* formulas. For every t such that $(s,t) \notin H_i$, we define $D_i(s,t)$ such that for every $(s,v) \in H_i$, $v \models D_i(s,t)$ and $t \not\models D_i(s,t)$. We also define formulas $C_i(s)$ such that for all states $v \in S$, $v \models C_i(s)$ iff $(s,v) \in H_i$.

We define $D_i(s,t)$ and $C_i(s)$ inductively.

- Let P be the set of atomic propositions true in s. Then for all $t \in S$, such that $(s,t) \notin H_0$ $D_0(s,t) = C_0(s) = \wedge_{(p \in P)}\, p \wedge_{(p \in AP \setminus P)} \neg p$.
- Let s and t be states such that $(s,t) \notin H_{i+1}$. Then there exists a fair rational path ρ from s for which there is no H_i-corresponding trace from t.

  Assume that $\rho = s_1, s_2, \ldots, s_N, (s_{N+1}, \ldots s_{N+K})^\omega$. We first define for $1 \leq j \leq K$ a formula that describes the cycle from place $j+N$, namely the trace
  $s_{N+j}, s_{N+j+1}, \ldots s_{N+K}, s_{N+1}, \ldots, s_{N+j-1}$.
  $cycle^j_{i+1}(s,t) = C_i(s_{(N+1+((j-1) mod\ K))}) \wedge \mathbf{X}(C_i(s_{(N+1+(j\ mod\ K))}) \wedge$
  $\mathbf{X}(C_i(s_{(N+1+((j+1)\ mod\ K))}) \wedge \ldots \mathbf{X}(C_i(s_{(N+1+(j+K-2)\ mod\ K)})) \ldots)$.
  Let $cycle_{i+1}(s,t) = \vee_{j=1}^{K} cycle^j_{i+1}(s,t)$.
  Let $trace_{i+1}(s,t) = C_i(s_1) \wedge \mathbf{X}(C_i(s_2) \ldots \wedge \mathbf{X}(C_i(s_{N+K}) \wedge \mathbf{X}\,\mathbf{G}(cycle_{i+1}(s,t)) \ldots)$.
  Let $D_{i+1}(s,t) = \mathbf{E}\, trace_{i+1}(s,t)$.
  Let $C_{i+1}(s) = \wedge_{(s,t) \notin H_i} D_{i+1}(s,t)$.

Note that $\rho^{N+1} \models cycle_{i+1}(s,t)$. Furthermore,
$\rho \models C_i(s_1) \wedge \mathbf{X}(C_i(s_2), \ldots, \wedge \mathbf{X}(C_i(s_N)) \ldots)$, thus $s \models D_{i+1}(s,t)$.

Given a state v, if $v \models D_{i+1}(s,t)$, then there is a fair trace ρ' starting at v such that $\rho' \models trace_{i+1}(s,t)$. We prove that $(\rho, \rho') \in H_i$. First, for each $1 \leq j \leq N+K$, $\rho'^j \models C_i(s_j)$. Further, it is true that for $j \geq N+1$, $\rho'^j \models cycle_{i+1}(s,t)$. Using these facts, one can show by induction that for $j \geq 1$, $\rho'^{N+j} \models cycle_{i+1}^{((j-1) mod\ K)+1}(s,t)$. This implies that for each $j \geq 1$, $\rho^{N+j} \models C_i(s_{N+1+(j-1) mod\ K})$.

Once we know that for every state v such that $(s,v) \in H_{i+1}$, $v \models D_{i+1}(s,t)$, it is easy to see that for every state v, $v \models C_{i+1}(s)$ iff $(s,v) \in H_{i+1}$.

Let $C_\infty(s)$ be the formula such that for all $v \in S$, $v \models C_\infty(s) \Leftrightarrow (s,v) \in H_\infty$. Then for all $t \in S$, $t \not\models C_\infty(s) \Leftrightarrow (s,t) \notin H_\infty$. Since $s \models C_\infty(s)$, for all $t \in S$ such that $(s,t) \notin H$ there exists $\psi \in$ECTL* that differentiates between s and t.$\Box$

Assume that $M \not\leq_{rat} M'$. Then there exists an initial state $s_0 \in S_0$ such that for all initial states $s'_0 \in S'_0$, $s_0 \not\leq_{rat} s'_0$. Thus, $M, s_0 \models C_\infty(s_0)$, and for all $s'_0 \in S'_0$, $M', s'_0 \not\models C_\infty(s_0)$. Let $\psi \in$ACTL* be the formula equivalent to $\neg C_\infty$. Then, since $M, s_0 \not\models \psi$, $M \not\models \psi$ and since for all $s'_0 \in S'_0$, $M', s'_0 \models \psi$, $M' \models \psi$.

From Corollary 4.5 and Lemma 4.6 we deduce Corollary 4.8.

Corollary 4.8 *If for all ACTL* formulas ψ, $M' \models \psi$ implies $M \models \psi$, then $M \leq_\exists M'$.*

Unlike ACTL*, ACTL does not characterize the exists simulation. In [1] two structures, M_1 and M_2, are given. It is shown in [1] that for every ϕ in ACTL, $M_2 \models \phi$ implies $M_1 \models \phi$. However, there exists an ACTL* formula φ such that $M_2 \models \varphi$ but $M_1 \not\models \varphi$. Since ACTL* characterizes the exists simulation, $M_1 \not\leq_\exists M_2$.

Unfortunately, the game, direct, and delay simulations cannot be characterized by either ACTL* or ACTL. In [12] two structures, M_1 and M_2, are given such that $M_1 \leq_\exists M_2$ but $M_1 \not\leq_g M_2$. Since ACTL* characterizes the exists simulation, for every ϕ in ACTL* (and therefore ACTL), $M_2 \models \phi$ implies $M_1 \models \phi$. Therefore, ACTL* (ACTL) does not characterizes the game simulation. Since the direct/delay simulation implies the game simulation, ACTL* (ACTL) does not characterize them either.

We have shown that ACTL* characterizes the exists simulation but not the game/delay/direct simulation. Furthermore, ACTL does not characterize any of these notions. The question arises whether the direct/delay/game

simulation can be characterized by any other logic. [12] shows that the game simulation can be characterized by the Universal Alternating Free μ-Calculus ($\forall$**AFMC**) logic when interpreted over fair structures.

We show that no reasonable logic that describes the fair branching behavior of a structure can characterize the direct/delay simulation. Consider structures M_1 and M_2 in Figure 5. M_1 and M_2 cannot be distinguished by a temporal logic formula. This is because they have computation trees, with exactly the same fair traces. However, $M_1 \not\leq_{de} M_2$ and therefore, $M_1 \not\leq_{di} M_2$. To see that $M_1 \not\leq_{de} M_2$ note that if the adversary chooses the path 123^∞ the protagonist must choose the path $1'2'3'^\infty$. However 2 is a fair state while $2'$ and $3'$ are not. Thus neither simulation can be characterized by any such logic.

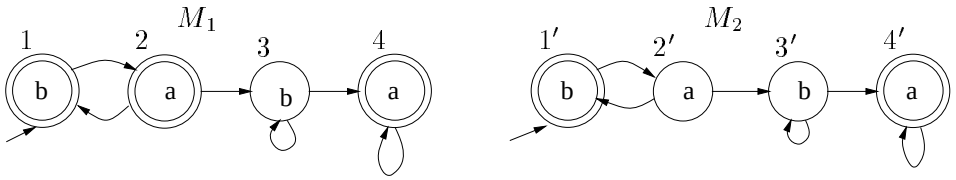

Figure 5: The direct/delay simulations cannot be characterized by temporal logics.

4.2 Maximal structure

Next we check for the existence of a maximal structure for a formula with respect to a preorder.

Definition 4.9 *A structure M_ϕ is maximal for formula ϕ with respect to preorder $\leq$ if for every structure M, $M \models \phi \Leftrightarrow M \leq M_\phi$.*

In [11] a construction of a maximal structure for ACTL formulas with respect to the exists simulation is presented. The maximal structure is used as a tableau for the formula. In this section we check whether the direct/delay/game simulations have a maximal structure. We prove that the maximal structure constructed in [11] is maximal with respect to the game simulation as well. On the other hand, we show that the formula $\mathbf{A}[a\,\mathbf{U}\,b]$ has no maximal structure with respect to the direct and delay simulations. This formula is contained in both ACTL and ACTL*.

4.3 A maximal structure for ACTL with respect to game simulation

We prove that for every ACTL formula, the tableau of the formula as defined in [11] is the maximal structure for the formula with respect to the game simulation. First, we describe the construction of the tableau as shown in [11]. In [11], a different type of fairness constraint, the generalized Büchi acceptance condition, is used. A *generalized Büchi acceptance* condition is a set $F = \{f_1, f_2, \ldots f_n\}$ of subsets of S. A trace ρ is *fair* according to F iff for every $1 \leq i \leq n$, $\inf(\rho) \cap f_i \neq \emptyset$. Since the game simulation is not limited to a certain type of fairness constraint, we do not have to change anything in its definition.

For the remainder of this section, fix an ACTL formula ψ. Let AP_ψ be the set of atomic propositions in ψ. The tableau associated with ψ is a structure $\mathcal{T}_\psi = (S_T, R_T, S_{0T}, L_T, F_T)$. The set of *elementary formulas* of ψ, $el(\psi)$, is defined as follows:

1. $el(p) = el(\neg p) = \{p\}$ if $p \in AP_\psi$.
2. $el(\phi_1 \vee \phi_2) = el(\phi_1 \wedge \phi_2) = el(\phi_1) \cup el(\phi_2)$.
3. $el(\mathbf{AX}\,\phi) = \{\mathbf{AX}\,\phi\} \cup el(\phi)$.
4. $el(\mathbf{A}[\phi_1\ \mathbf{U}\ \phi_2]) = \{\mathbf{AX}\,False, \mathbf{AX}(\mathbf{A}[\phi_1\ \mathbf{U}\ \phi_2])\} \cup el(\phi_1) \cup el(\phi_2)$.
5. $el(\mathbf{A}[\phi_1\ \mathbf{R}\ \phi_2]) = \{\mathbf{AX}\,False, \mathbf{AX}(\mathbf{A}[\phi_1\ \mathbf{R}\ \phi_2])\} \cup el(\phi_1) \cup el(\phi_2)$.

The set of tableau states is $S_T = \mathcal{P}(el(\psi))$[6]. The labeling function is $L_T(s_t) = s_t \cap AP_\psi$. In order to specify the set S_{0T} of initial states and the transition relation R_T, we need an additional function *sat* that associates with each sub-formula ϕ of ψ a set of states in S_T. Intuitively, $sat(\phi)$ will be the set of states that satisfy ϕ.

1. $sat(\phi) = \{s \mid \phi \in s\}$ where $\phi \in el(\psi)$.
2. $sat(\neg\phi) = \{s \mid \phi \notin s\}$ where ϕ is an atomic proposition. Recall that only atomic propositions can be negated in ACTL.
3. $sat(\phi \vee \varphi) = sat(\phi) \cup sat(\varphi)$.
4. $sat(\phi \wedge \varphi) = sat(\phi) \cap sat(\varphi)$.

[6]Some of the states are deleted in order to keep R_T total.

5. $sat(\mathbf{A}[\phi \mathbf{U} \varphi]) = (sat(\varphi) \cup (sat(\phi) \cap sat(\mathbf{AX}(\mathbf{A}[\phi \mathbf{U} \varphi])))) \cup sat(\mathbf{AX}\, False)$.

6. $sat(\mathbf{A}[\phi \mathbf{R} \varphi]) = (sat(\varphi) \cap (sat(\phi) \cup sat(\mathbf{AX}(\mathbf{A}[\phi \mathbf{R} \varphi])))) \cup sat(\mathbf{AX}\, False)$.

The set of initial states of the tableau is $S_{0T} = sat(\psi)$. The transition relation is defined so that if $\mathbf{AX}\,\phi$ is included in some state then all its successors should satisfy ϕ.

$$R_T(s_1, s_2) = \bigwedge_{AX\phi \in el(\psi)} (\mathbf{AX}\,\phi) \in s_1 \Rightarrow s_2 \in sat(\phi).$$

The fairness constraint guarantees that *eventuality* properties are fulfilled. This is done by requiring that for every fair trace ρ, for every elementary formula $\mathbf{AX}\,\mathbf{A}[\phi\,\mathbf{U}\,\varphi]$ of ψ, and for every state s on ρ, if $s \in sat(\mathbf{AX}\,\mathbf{A}[\phi\,\mathbf{U}\,\varphi])$, then there is a later state t on ρ such that $t \in sat(\varphi)$. Thus, we obtain the following fairness constraints:

$$F_T = \{\, ((S_T - sat(\mathbf{AX}\,\mathbf{A}[\phi\,\mathbf{U}\,\varphi])) \cup sat(\varphi)) \mid \mathbf{AX}\,\mathbf{A}[\phi\,\mathbf{U}\,\varphi] \in el(\psi) \,\}.$$

4.4 The tableau is the maximal structure for game simulation

In this section we prove that for every Kripke structure M, $M \models \psi$ iff $M \leq_g \mathcal{T}_\psi$. Most lemmas were proved in [11] for the exists simulation. We give proofs only for the lemmas that are different due to the change of the simulation preorder.

Lemma 4.10 *[11] For all subformulas ϕ of ψ, if $t \in sat(\phi)$, then $t \models \phi$.*

The main result of Lemma 4.10 is that the tableau for ψ satisfies ψ. This is because any initial state of $\mathcal{T}_\psi$ is in $sat(\psi)$, and therefore every initial state of $\mathcal{T}_\psi$ satisfies ψ. Consequently, since ACTL is preserved by the $\leq_g$ preorder, for every Kripke structure M, if $M \leq_g \mathcal{T}_\psi$, then $M \models \psi$.

Our next step is to prove that $M \models \psi$ implies $M \leq_g \mathcal{T}_\psi$. We show that if $M \models \psi$ then the protagonist has a winning strategy function in a game over $M \times \mathcal{T}_\psi$. We define the strategy function π as follows: $\pi(s_0, \perp) = \{\, \phi \mid \phi \in el(\psi), s_0 \models \phi \,\}$ and $\pi(s', t) = \{\, \phi \mid \phi \in el(\psi), s' \models \phi \,\}$. Thus, whenever the adversary moves to a state s', the protagonist moves to $t' = \pi(s', t)$, such that both s', t' satisfy exactly the same set of elementary formulas of ψ. The following lemma extends this result for all subformulas of ψ.

Lemma 4.11 *[11] If $t' = \pi(s', t)$, then for every subformula or elementary formula ϕ of ψ, $s' \models \phi$ implies $t' \in sat(\phi)$.*

Lemma 4.12 *π is a winning strategy.*

Proof

1. Any given state s' satisfies a unique subset of $el(\psi)$. Thus, for every s', t' is unique and π is a function.

2. For every $s_0 \in S_0$, by Lemma 4.11 $M, s_0 \models \psi$ implies $t_0 = \pi(s_0, \bot) \in sat(\psi)$. By the definition of S_{0T}, this implies $t_0 \in S_{0T}$.

3. Assume that $t' = \pi(s', t)$. Then for every $p \in AP_\psi$, $p \in L(s') \Leftrightarrow s' \models p \Leftrightarrow p \in L_T(t')$.

4. Assume that $t' = \pi(s', t)$. Let (s, t) be the position of the game in the previous round. Let $\mathbf{AX}\,\phi_1$, $\mathbf{AX}\,\phi_2, \ldots, \mathbf{AX}\,\phi_n$ be all the formulas of the form $\mathbf{AX}\,\phi$ in $el(\psi)$ which s satisfies. Then we have $s' \models \phi_1$, $s' \models \phi_2$, $\ldots, s' \models \phi_n$. By Lemma 4.11, $t' \in sat(\phi_1)$, $t' \in sat(\phi_2)$, $\ldots, t' \in sat(\phi_n)$. Now by the definition of π, the formulas of the form $\mathbf{AX}\,\phi$ in t must be exactly $\mathbf{AX}\,\phi_1$, $\mathbf{AX}\,\phi_2$, $\ldots$, $\mathbf{AX}\,\phi_n$. Then by the definition of R_T, we see that $(t, t') \in R_T$.

5. We prove that if ρ is a fair run, then $\pi(\rho)$ is also a fair run. Assume that $\pi(\rho)$ is not fair. By the definition of F_T, there must be some elementary subformula $\mathbf{AX}\,\mathbf{A}[\phi_a\ \mathbf{U}\ \phi_b]$ such that

$$inf(\pi(\rho)) \cap ((S_T - sat(\mathbf{AX}\,\mathbf{A}[\phi_a\ \mathbf{U}\ \phi_b])) \cup sat(\phi_b)) = \emptyset.$$

   This means that there is an $i \geq 0$ such that for all $j \geq i$, $\pi(s_j, t_{j-1}) \in sat(\mathbf{AX}\,\mathbf{A}[\phi_a\ \mathbf{U}\ \phi_b])$ but $\pi(s_j, t_{j-1}) \notin sat(\phi_b)$.

   Consider the state $t_i = \pi(s_i, t_{i-1})$. $t_i \in sat(\mathbf{AX}\,\mathbf{A}[\phi_a\ \mathbf{U}\ \phi_b])$ iff $\mathbf{AX}\,\mathbf{A}[\phi_a\ \mathbf{U}\ \phi_b] \in t_i$. The definition of π then implies that $s_i \models \mathbf{AX}\,\mathbf{A}[\phi_a\ \mathbf{U}\ \phi_b]$. In addition, Lemma 4.11 implies that if $t_i \notin sat(\phi_b)$, then $s_i \not\models \phi_b$. Since $\pi(s_i, t_{i-1}) \in sat(\mathbf{AX}\,\mathbf{A}[\phi_a\,\mathbf{U}\,\phi_b])$ and for all $j \geq i$, $\pi(s_j, t_{j-1}) \notin sat(\phi_b)$, then $s_i, s_{i+1}, \ldots$ is a fair trace in M starting at s_i, and every state on this trace satisfies $\neg\phi_b$. But $s_i \models \mathbf{AX}\,\mathbf{A}[\phi_a\ \mathbf{U}\ \phi_b]$, a contradiction. Hence $\pi(\rho)$ is in fact a fair trace in $\mathcal{T}_\psi$. $\Box$

Corollary 4.13 *For any structure M, $M \models \psi$ iff $M \leq_g \mathcal{T}_\psi$. Thus, $\mathcal{T}_\psi$ is the maximal structure for ψ with respect to game simulation.*

4.5 A maximal structure for direct/delay simulation

We now show that it is impossible to construct a maximal structure for the formula $\phi = \mathbf{A}[a\ \mathbf{U}\ b]$ with respect to the direct/delay simulations. Thus, any logic that contains this formula or an equivalent formula, in particular ACTL and ACTL*, does not have a maximal structure with respect to these simulations. More specifically, we show that there is no finite structure $\mathcal{T}_\phi$ such that $\mathcal{T}_\phi \models \phi$ and $\mathcal{T}_\phi$ is greater by the direct/delay simulation than any structure that satisfies ϕ. Since the direct simulation implies the delay simulation, it is sufficient to prove this result for the delay simulation. In

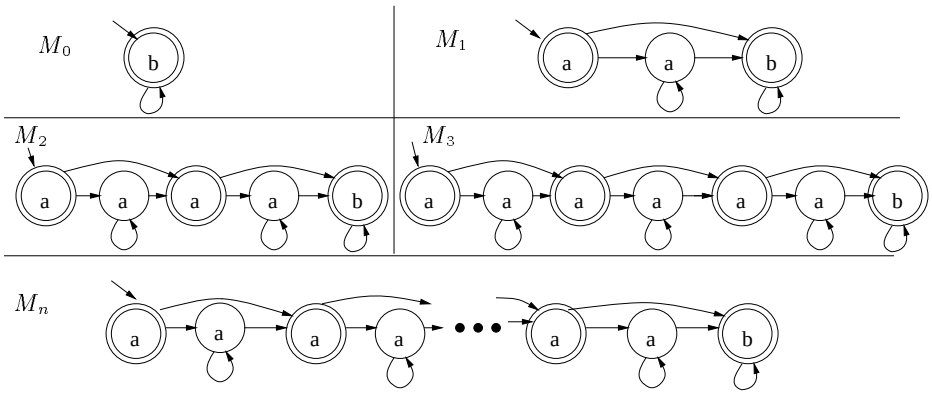

Figure 6: There is no finite structure M' such that for every n in IN, M' is greater by direct/delay simulation than M_n, and $M' \models \mathbf{A}[a\ \mathbf{U}\ b]$.

Figure 6 we present a sequence of structures $M_0, M_1, \ldots$ such that for every n in IN, $M_n \models A(aUb)$. We prove that for every n and every structure M', if $M_n \leq_{de} M'$ and $M' \models \mathbf{A}[a\ \mathbf{U}\ b]$ then $|M'| \geq n$. Thus, any structure that satisfies $\mathbf{A}[a\ \mathbf{U}\ b]$ and is greater by the delay simulation than all the structures in the sequence has to be infinite.

Lemma 4.14 *For every $n > 0$ and every structure M', if $M_n \leq_{de} M'$ and $M' \models \mathbf{A}[a\ \mathbf{U}\ b]$, then $|M'| \geq n$.*

Proof Let $n \in IN$ be a natural number and M' be a structure such that $M' \models \mathbf{A}[a\ \mathbf{U}\ b]$ and $M_n \leq_{de} M'$. In a game over $M_n \times M'$ the protagonist has a winning strategy and thus it wins in every game no matter how the adversary plays. Consider the following strategy of the adversary. It starts from the initial state. As long as the protagonist moves to a fair state the

adversary moves to the next fair state (until it reaches the last one). If the protagonist moves to a state that is not fair, then the adversary moves to the successor which is not fair in M_n and stays there until the protagonist moves to a fair state in M'. We distinguish between two cases:

1. The suffix of the game is an infinite sequence of unfair states in both structures. In this case the adversary is the last player who was in a fair state. Thus it wins the game. This means that M' is not greater than M_n by the delay simulation, a contradiction.

2. Otherwise, the adversary moves through n fair states in M_n that are labeled a to the state labeled b. Since the adversary moves to a fair state only when the protagonist is in a fair state, the protagonist has been in n fair states that are labeled a. Since $M' \models \mathbf{A}[a\ \mathbf{U}\ b]$, these states must be different (otherwise there would be an infinite fair trace which is labeled a). Thus the size of M' is at least n. $\Box$

We proved that there is no maximal structure for $\mathbf{A}[a\ \mathbf{U}\ b]$ with respect to the direct/delay simulations.

5 A new implementation for the assume-guarantee framework

This section shows that the game simulation can replace the exists simulation in the implementation of the assume-guarantee paradigm [10, 13, 18, 19], as suggested in [11].

In the assume-guarantee paradigm, properties of different parts of the systems are verified separately. The environment of the verified part is represented by a formula that describes its properties. The formula either has been verified or is given by the user. The method proves assertions of the form $\psi M \phi$, meaning that if the environment satisfies ψ then the composition of M with the environment satisfies ϕ. The method enables the creation of a proof schema which is based on the structure of the system. [11] suggests a framework that uses the assume-guarantee paradigm for semi-automatic verification. It presents a general method that uses models as assumptions; the models are either generated from a formula as a *tableau* or are abstract models given by the user. The proof of $\psi M \phi$ is done automatically by verifying that the composition of the tableau for ψ with M satisfies ϕ. The method requires a preorder $\leq$, a composition operator $||$, and a specification language $\mathcal{L}$ which satisfy the following properties:

1. For every two structures M_1, M_2, if $M_1 \leq M_2$, then for every formula ψ in $\mathcal{L}$, $M_2 \models \psi$ implies $M_1 \models \psi$.

2. For every two structures M_1, M_2, $M_1 \| M_2 \leq M_1$.

3. For every three structures M_1, M_2, M_3, $M_1 \leq M_2$ implies $M_1 \| M_3 \leq M_2 \| M_3$.

4. Let ψ be a formula in $\mathcal{L}$ and $\mathcal{T}_\psi$ be a tableau for ψ. Then $\mathcal{T}_\psi$ is the maximal structure with respect to the preorder $\leq$.

5. For every structure M, $M \leq M \| M$.

An implementation for this framework was presented in [11]. The implementation uses the *ACTL* logic as the specification language, the exists simulation preorder, and a composition operator which satisfy the properties above. In this section we suggest a new implementation which is similar to that of [11], except that the game simulation is used as the preorder. We show that the game simulation can replace the exists simulation. As we have stated, the game simulation preserves the ACTL logic, and thus property one is satisfied. In Section 4 we proved that the game simulation satisfies property four. Thus, it is left to show that the game simulation preorder and the composition operator as defined in [11] satisfy properties two, three and five. Again we use generalized Büchi constraints. In order to prove these properties we need to define the composition operator $\|$.

Definition 5.1 *Let M_1, M_2 be Kripke structures. The parallel composition of M_1 and M_2, denoted $M_1 \| M_2$, is the structure M defined as follows.*

- $AP = AP_1 \cup AP_2$.
- $S = \{(s_1, s_2) | L_1(s_1) \cap AP_2 = L_2(s_2) \cap AP_1\}$[7].
- $R = \{((s_1, s_2), (t_1, t_2)) | (s_1, t_1) \in R_1 \wedge (s_2, t_2) \in R_2\}$.
- $S_0 = (S_{0_1} \times S_{0_2}) \cap S$.
- $L((s_1, s_2)) = L_1(s_1) \cup L_2(s_2)$.
- $F = \{(f_i \times S_2) \cap S | f_i \in F_1\} \cup \{(S_1 \times f_i) \cap S | f_i \in F_2\}$.

[7] Some of the states might have to be deleted in order to keep R total.

Remark: In all notions of simulation, there is a requirement that if $s_1 \leq s_2$, then $L_1(s_1) = L_2(s_2)$. When M_1 and M_2 are defined over different **AP** we replace this requirement with $L_1(s_1) \cap AP_2 = L_2(s_2) \cap AP_1$.

Lemma 5.2 *(* **property 2.***) For every pair of Kripke structures M_1, M_2, $M_1 \| M_2 \leq_g M_1$.*

Proof We define a strategy π as follows: $\pi((s_{01}, s_{02}), \perp) = s_{01}$ and $\pi((s'_1, s'_2), s_1) = s'_1$, i .e., the protagonist moves on the projection of the adversary's trace on M_1. It is easy to see that π is a function. Let $((s_1, s_2), s_1)$ be the previous position in the game and assume that the adversary moves to (s'_1, s'_2). Then $s'_1 = \pi((s'_1, s'_2), s_1)$. Clearly, $L_{12}(s'_1, s'_2) \cap AP_1 = L_1(s'_1)$. The definition of composition implies that if $((s_1, s_2), (s'_1, s'_2))$ is a transition in $M_1 \| M_2$ then (s_1, s'_1) is a transition in M_1. Furthermore, if the adversary's trace is fair then the protagonist's trace is fair as well. □

Lemma 5.3 *(* **property 3.***) Let M_1, M_2, M_3 be Kripke structures. Then $M_1 \leq_g M_2$ implies $M_1 \| M_3 \leq_g M_2 \| M_3$.*

Proof Let π be a strategy in a game over $M_1 \times M_2$. We define a strategy π' as follow: $\pi'((s_{01}, s_{03}), \perp) = (\pi(s_{01}, \perp), s_{03})$ and $\pi'((s'_1, s'_3), (s_2, s_3)) = (\pi(s'_1, s_2), s'_3)$, i.e., whenever the adversary moves to s'_1 in M_1 and s'_3 in M_3, the protagonist moves to the same state in M_3 and to $s'_2 = \pi(s'_1, s_2)$ in M_2.

It is easy to see that π' is a function. Let $((s_1, s_3), (s_2, s_3))$ be the previous position in the game and assume that the adversary moves to (s'_1, s'_3). Then
$\pi'((s'_1, s'_3), (s_2, s_3)) = (\pi(s'_1, s_2), s'_3)$. Let $s'_2 = \pi(s'_1, s_2)$. Since π is a winning strategy, $L_1(s'_1) = L_2(s'_2)$ and (s_2, s'_2) is a transition in M_2. Thus, $L_{13}(s'_1, s'_3) = L_{23}(s'_2, s'_3)$. Furthermore, the definition of composition implies that if $((s_1, s_3), (s'_1, s'_3))$ is a transition in $M_1 \| M_3$ then $((s_2, s_3), (s'_2, s'_3))$ is a transition in $M_2 \| M_3$.

Whenever the adversary moves on a fair trace in $M_1 \| M_3$, the traces projected on M_1 and M_3 are both fair. The protagonist moves on the same trace on M_3. Thus this trace is fair. Let ρ_1 be the trace on M_1 along which the adversary moves. Since ρ_1 is fair and π is a strategy, the trace $\pi(\rho_1)$ along which the protagonist moves on M_2 is fair as well. The definition of $\|$ implies that the protagonist moves on a fair trace in $M_2 || M_3$. □

Lemma 5.4 property 5. *For every structure M, $M \leq_g M \| M$.*

Proof Consider the strategy $\pi(s_0, \perp) = (s_0, s_0)$ and $\pi(s', (s, s)) = (s', s')$. Clearly π is a winning strategy. □

We proved that the game simulation preorder and the composition operator satisfy the properties required in [11]. Therefore, game simulation can replace the exists simulation in the assume-guarantee framework presented in [11].

5.1 Complexity

Verifying a formula of the form $\psi M \varphi$ is PSPACE-complete in the size of ψ [16]. However, the real bottleneck of this framework is checking for fair simulation between models, which for the exists simulation is PSPACE complete in the size of the models. (Typically, models are much larger than formulas). Thus, replacing the exists simulation with the game simulation reduces this complexity to polynomial and eliminates the bottleneck of the framework. However, the algorithm for game simulation presented in [8] refers to Kripke structures with regular Büchi constraints, and the implementation presented in [11] refers to Kripke structures with generalized Büchi constraints. In order to apply the algorithm suggested in [8] within the assume-guarantee framework, we need a translation between these types of fairness constraints.

[5] defines a transformation of a Büchi automaton with generalized fairness constraints into a Büchi automaton with regular fairness constraints. Here we show that applying this transformation to a Kripke structure with generalized Büchi constraints results in a Kripke structure with regular Büchi constraints that is game simulation equivalent to the original one. The translation affects the size of the structure and thus the complexity of the construction of the preorder. The sizes of S and R are multiplied by $|F|$, where $|F|$ is the number of sets in F. Thus the complexity of constructing the preorder is $|F| \cdot |R| \cdot (|S| \cdot |F|)^3 = |R| \cdot |S|^3 \cdot |F|^4$. Note that in the tableau for a formula, $|F|$ is bounded by the size of the formula and the size of the tableau is exponential in the size of the formula; thus, the transformation of the tableau to regular fairness constraints result in a strucuture that is logarithmic bigger than the original one.

Definition 5.5 *[5] Let* $M =< S, R, S_0, L, \{f_1, f_2, \ldots f_n\} >$ *be a Kripke structure with generalized Büchi constraints. We define the Kripke structure* $M_r =< AP, S_r, R_r, L_r, F_r >$ *with a regular Büchi constraint, as follows:*

- $S_r = S \times \{1, 2, \ldots n\}$.
- $R_r = \cup_{i=1}^{n} \{((s_1, i), (s_2, i)) | (s_1, s_2) \in R \wedge s_1 \notin f_i\} \cup$

$$\cup_{i=1}^{n-1}\{((s_1,i),(s_2,i+1))|(s_1,s_2)\in R\wedge s_1\in f_i\}\cup$$
$$\{((s_1,n),(s_2,1))|(s_1,s_2)\in R\wedge s_1\in f_n\}.$$

- $S_{r_0}=S_0\times\{1\}$.
- $L_r(s,i)=L(s)$.
- $F_r=\{(s,n)|s\in f_n\}$.

In the proof below M denotes a Kripke structure with generalized Büchi constraints. M_r denotes the transformation of M to a Kripke structure with regular Büchi constraints. We show that $M_r\leq_g M$ and $M\leq_g M_r$.

Lemma 5.6 $M\leq_g M_r$.

Proof : First we define a strategy π for the protagonist: $\pi(s_0,\perp)=(s_0,1)$ and

$$\pi(s',(s,i))=\begin{cases}(s',i) & s\notin f_i\\ (s',i+1) & i<n\wedge s\in f_i\\ (s',1) & i=n\wedge s\in f_n.\end{cases}$$

Next, we prove that π is a winning strategy. It is easy to see that π is a function. The definition of the transformation implies that if $(s,i)=\pi(s,(t,j))$ then $L_r((s,i))=L(s)$ and that $((t,j),(s,i))$ is a transition in M_r.

It is left to prove that if the adversary moves on a fair trace ρ in M then the protagonist moves on $\pi(\rho)$, which is a fair trace in M_r.

First, we prove that for every $i\in 1,2,\ldots n$

$(*)$ there are infinitely many states of the form (s,i) in $\pi(\rho)$.

Assume to the contrary that there is an index $i\in\{1,2,\ldots n\}$ which does not satisfy (*). Let j be the minimal index which does not satisfy (*) and let k be the index before j $(k=((j-2)mod\ n)+1)$. Then there exists a suffix of $\pi(\rho)$ in which all the states are of the form (s,k). This implies that there exists a suffix of ρ without states in f_k. Thus, ρ is not fair, a contradiction.

Next we prove that $\pi(\rho)$ is fair. Since $\pi(\rho)$ contains infinitely many states of the form (s,n) and infinitely many states of the form $(s,1)$, then there exist infinitely many states in F_r. □

Lemma 5.7 $M_r\leq_g M$.

Proof We define the strategy π for the protagonist: $\pi((s_0,1),\perp) = s_0$ and $\pi((s_2,i),s_1) = s_2$. It is easy to see that π is a function and that $s_2 = \pi((s_2,j),s_1)$ implies that $L_r((s_2,j)) = L(s_2)$. $((s_1,i),(s_2,j)) \in R_r$ also implies $(s_1,s_2) \in R$. It is left to prove that if the adversary moves on a fair trace in M_r then the protagonist moves on a fair trace in M. Let $\rho = (s_0,i_0),(s_1,i_1),(s_2,i_2),\ldots$ be a fair trace in M_r. We prove that

$$\pi(\rho) = s_0, \pi((s_1,i_1),s_0), \pi((s_2,i_2),s_1), \pi((s_3,i_3),s_2),\ldots = s_0,s_1,s_2,\ldots$$

is a fair run in M. Assume to the contrary that $\pi(\rho)$ is not fair. Then there exists an index $i \in \{1\ldots n\}$ such that $\pi(\rho)$ contains only finitely many states in f_i. Thus, there is a suffix of $\pi(\rho)$ without any state in f_i. This implies that

(**) there exists a suffix of ρ, without any states of the form (s,i), where s is an element in f_i.

Let j be the minimal index that satisfies (**). Then there exists a suffix of ρ in which all the states are of the form (s,j). This implies that this suffix does not contain any states in $\{(s,n)|s \in f_n\}$. Thus ρ is not fair, a contradiction. □

6 Conclusion

This work shows that there is no notion of fair simulation which has all the desired advantages. However, it is clear that their relationship with the logics gives the exists and game simulations several advantages over the delay and direct simulations. On the other hand, the delay and direct simulations are better for minimization. Since this research is motivated by usefulness to model checking, relationships with a logic are important. Thus, it is advantageous to refer to the delay and direct simulations as approximations of the game/exists simulations. These approximations enable some minimization with respect to the exists and game simulations. Out of the four notions, we consider the game simulation to be the best. This is due to its complexity and its applicability in modular verification.

References

[1] A. Aziz, V. Singhal, T.R. Shiple, A.L. Sangiovanni-Vincentelli, F. Balarin, and R.K. Brayton. Equivalences for fair kripke structures.

In *ICALP*, LNCS 840, pages 364–375, 1994.

[2] S. Bensalem, A. Bouajjani, C. Loiseaux, and J. Sifakis. Property preserving simulation. In *Computer-Aided Verification*, volume 663 LNCS, pages 260–273, 1981.

[3] D. Bustan and O. Grumberg. Simulation based minimization. In *Conference on Automated Deduction*, volume 17, pages 255–270, 2000.

[4] E.M. Clarke, O. Grumberg, and D.A. Peled. *Model Checking.* MIT Press, 1999.

[5] C. Courcoubetis, M. Vardi, P. Wolper, and M. Yannakakis. Memory efficient algorithms for the verification of temporal properties. In *Proceedings of Computer-Aided Verification*, volume 531 of *LNCS*, pages 233– 242, 1991.

[6] D.L. Dill, A.J. Hu, and H. Wong-Toi. Checking for language inclusion using simulation relation. In *Computer-Aided Verification*, LNCS 575, pages 255–265, 1991.

[7] E. A. Emerson and C. S. Jutla. Tree automata, mu-calculus and determinacy. In *32nd Annual Symposium on Foundations of Computer Science*, pages 368–377, San Juan, Puerto Rico, 1–4 October 1991. IEEE.

[8] K. Etessami, Th. Wilke, and R. Schuller. Fair simulation relations, parity games, and state space reduction for Büchi automata. In *Automata, Languages and Programming, 28th International colloquium*, LNCS 2076, pages 694–707, 2001.

[9] K. Etessami, Th. Wilke, and R. Schuller. Faster algorithms for computing fair simulation relation, and how to use them for state space reduction. Technical Report ITD-01-40643, Bell Labs, 2001.

[10] N. Francez. *The Analysis of Cyclic Programs.* PhD thesis, Weizmann Institute of Science, 1976.

[11] O. Grumberg and D.E. Long. Model checking and modular verification. *ACM Trans. on Programming Languages and Systems (TOPLAS)*, 16(3):843–871, 1994.

[12] T.A. Henzinger, O. Kupferman, and S. Rajamani. Fair simulation. In *Proc. 8th Conference on Concurrency Theory*, LNCS 1234, 1997.

[13] C. B. Jones. Specification and design of (parallel) programs. In *International Federation for Information Processing (IFIP)*, pages 321–332, 1983.

[14] Shmuel Katz and Doron Peled. Defining conditional independence using collapses. *Theoretical Computer Science*, 101(2):337–359, 1992.

[15] O. Kupferman and M.Y. Vardi. Verification of fair transition systems. In *Computer Aided Verification (CAV'96)*, LNCS 1102, pages 372–382, 1996.

[16] O. Kupferman and M.Y. Vardi. Modular model checking. In *Proc. Compositionality Workshop*, LNCS 1536. Springer-Verlag, 1998.

[17] R. Milner. An algebraic definition of simulation between programs. In *Proc. of the 2nd International Joint Conferences on Artificial Intelligence (IJCAI)*, pages 481–489, London, UK, 1971.

[18] J. Misra and K.M. Chandy. Proofs of networks of processes. *IEEE Transactions on Software Engineering*, 7:417–426, 1981.

[19] A. Pnueli. In transition from global to modular temporal reasoning about programs. In K. R. Apt, editor, *Logics and Models of Concurrent Systems*, volume 13 of *NATO ASI series F*. sv, 1984.

A A quotient structure for the delay simulation

In this section we prove Lemma 3.3. For every structure M, let M^Q be its quotient structure with respect to the delay simulation. Then M and M^Q are equivalent with respect to the delay simulation.

The proof that $M \leq_{de} M^Q$ is straightforward. Consider the strategy $\pi(s_0, \bot) = [s_0]$ and $\pi(s', [s]) = [s']$. It easy to see that π is a winning strategy.

Before we prove the other direction, we need some new definitions. First, we extend the definition of delay simulation to a relation over the states of a structure.

In [8] it is shown that there exists a strategy π^* such that π^* is a winning strategy for every simulation game over $M \times M$, where the adversary and the

protagonist start at states s_1 and s_2 such that $s_1 \leq_{de} s_2$. Another property of π^* is presented in Proposition A.1.

Proposition A.1 *[8] Let s_1 and s_2 be states in M such that $s_1 \leq_{de} s_2$. Let s_1' be a successor of s_1 and $s_2' = \pi^*(s_1', s_2)$. Then $s_1' \leq_{de} s_2'$.*

Since the delay simulation is transitive, the following proposition is straightforward.

Proposition A.2 *Let M be a structure and let M^Q be its quotient structure. Let s_1 and s_2 be states in M such that $s_1 \leq_{de} s_2$. Then every state s_3 that is in the same equivalence class as s_1 satisfies $s_3 \leq_{de} s_2$.*

We denote by $[s]$ the equivalence class of s. Lemma A.3 and Lemma A.4 imply that $M^Q \leq_{de} M$.

Lemma A.3 *Let s_1 and s_2 be states in M such that $s_1 \leq_{de} s_2$. Then the protagonist has a strategy in a game over $M^Q \times M$ in which the adversary starts at $[s_1]$ and the protagonist starts at s_2. In each round assume that the adversary is at $[s_3]$ and the protagonist is at s_4. Then $s_3 \leq_{de} s_4$.*

Proof Let π^* be the winning strategy over $M \times M$. We define the strategy π' as follows: At the beginning, $\pi'([s_1], \perp) = s_2$. Assume that the previous position of the game was $([s_3], s_4)$ such that $s_3 \leq_{de} s_4$ and that the adversary moves to $[s_3']$. The definition of M^Q implies that there exists a transition (t_3, t_3') in M such that s_3 and t_3 are in the same class, as are s_3' and t_3'. Proposition A.2 implies that $t_3 \leq_{de} s_4$. We define $\pi'([s_3]', s_4) = \pi^*(t_3', s_4)$. By the definition of π^*, π' is well-defined. Moreover, since s_3' and t_3' are in the same equivalence class, $s_3' \leq_{de} t_3'$. Furthermore, by Proposition A.1, since $s_4' = \pi^*(t_3', s_4)$, $t_3' \leq_{de} s_4'$, and therefore $s_3' \leq_{de} s_4'$.□

Note that this strategy ensures that in every round $L^Q([s_3]) = L(s_4)$. However, it does not ensure that whenever the adversary moves to a fair state, the protagonist moves to a fair state after finitely many rounds.

Lemma A.4 *Let M be a structure and let M^Q be its quotient structure. Then $M^Q \leq_{de} M$.*

Proof We describe a strategy π'' which uses memory. In [7, 8] it is shown that if there exists a strategy with memory then there exists a memoryless strategy. The strategy π'' "remembers" two arguments: the first argument is called the *status*, which can be either *fulfilled* or *unfulfilled*. The status is unfulfilled if the protagonist has not visited a fair trace since the last time

the adversary did. Otherwise, the status is fulfilled. The second argument called the *middle*, and it "remembers" a state in M.

Let π^* be a winning strategy over $M \times M$ and π' a strategy over $M^Q \times M$ as defined in Lemma A.3. We define π'' as follows: $\pi''([s_0], \perp) = s_0$, If the status is fulfilled, then $\pi''([s_3'], s_4) = \pi'([s_3'], s_4)$. Thus the middle argument is ignored. In a round where the status becomes unfulfilled, meaning that $[s_3]$ is fair and s_4 is not, we assign middle to be a fair state in the class of s_3 (there is at least one).

If the status is not fulfilled, assume that the adversary moves to $[s_3']$. Then we assign $middle' = \pi'([s_3'], middle)$ and $\pi''([s_3'], s_4) = \pi^*(middle', s_4)$.

In order to see that π'' is a winning strategy, first consider the round where the status becomes unfulfilled. In this round, s_3 and *middle* are in the same class. Thus, if the position is $([s_3], s_4)$, then $s_3 \leq_{de} middle$. Furthermore, as long as the status does not become fulfilled, *middle* moves along a trace in M such that whenever the adversary moves to $[s_3]$, $s_3 \leq_{de} middle$. Since *middle* starts at a fair state and moves on a trace in M, by the definition of π^*, after a finite number of rounds, the protagonist moves to a fair state as well. □

B Proving Lemma 4.4

Lemma 4.4 cliams the following:
Let s and t be states in structure M. If there exists a fair trace ρ_s from s such that for all fair traces ρ_t from t, $\rho_s \not\leq_{rat} \rho_t$, then there exists a fair rational trace ρ_{sr} from s such that for all fair traces ρ_t from t, $\rho_{sr} \not\leq_{rat} \rho_t$.

We define an equivalence relation with respect to $\leq_{rat}$, such that states s and t are equivalent with respect to $\leq_{rat}$ if $s \leq_{rat} t$ and $t \leq_{rat} s$. We denote by $[s]$ the equivalence class of s. We say that $[s_1] \leq_{rat} s_2$ iff $s_1 \leq_{rat} s_2$.

Definition B.1 *Let M be a structure. We define the preorder structure M^P as follows:*

- $AP = \{C_1, C_2, \ldots C_n\}$ *where* $\{C_1, C_2, \ldots C_n\}$ *are the equivalence classes with respect to* $\leq_{rat}$.
- $S^P = \{(s, C_i) | s \in S$ *and there exists* $s' \in C_i$ *such that* $(s', s) \in H\}$.
- $((s, C_i), (t, C_j)) \in R^P \Leftrightarrow (s, t) \in R$.
- $S_0^P = \{(s_0, C_i) | s_0 \in S_0\}$.

- $L^P((s, C_i)) = C_i$.
- $(s, C_i) \in F^P \Leftrightarrow s \in F$.

Given a state s^P in M^P, we denote by $head(s^P)$ the first element of s^P and by $tail(s^P)$ the second element of s^P.

Lemma B.2 *Given a fair trace ρ_s from a state s and a state t in a structure M, the following conditions are equivalent:*

1. *There exists a fair trace ρ_t from t such that $\rho_s \leq_{rat} \rho_t$.*
2. *There exists a fair trace ρ_{tp} from $(t, [s])$ such that for all $i \geq 0$, $L^P(\rho^i_{tp}) = [\rho^i_s]$.*

Proof For the first direction, assume that there exists a fair trace ρ_t from t such that $\rho_s \leq_{rat} \rho_t$. Consider the trace ρ_{tp} such that for all $i \geq 0$, $head(\rho^i_{tp}) = \rho^i_t$ and $tail(\rho^i_{tp}) = [\rho^i_s]$. By the definition of M^P, ρ_{tp} is a trace in M^P. Since ρ_t is fair, ρ_{tp} is fair as well.

For the second direction, assume that there exists a fair trace ρ_{tp} from $(t, [s])$ such that for all $i \geq 0$, $L^P(\rho^i_{tp}) = [\rho^i_s]$. Consider the trace ρ_t that satisfies $\rho^i_t = head(\rho^i_{tp})$. By the definition of M^P, ρ^i_t is a trace in M. Furthermore, $\rho_s \leq_{rat} \rho_t$. Since ρ_{tp} is a fair trace, ρ_t is fair as well. □

Lemma B.3 *Let ρ_{sp} be a fair trace from $(s_1, [s_2])$ in M^P such that $L^P(\rho_{sp})$ is an ω-regular word. Then there exists a rational trace ρ_{s_1} from s_1 such that for all $i \geq 0$, $tail(\rho^i_{sp}) \leq_{rat} \rho^i_{s_1}$.*

Proof Since $L^P(\rho_{sp})$ is an ω-regular word, we can write it as $w_1 w_2^\omega$. Let $N = |w_1|$ and $K = |w_2|$. Consider the trace ρ_{s_1} that satisfies, $\rho^i_s = head(\rho^i_{sp})$. Then, for all $i \geq 0$, $tail(\rho^i_{sp}) \leq_{rat} \rho^i_{s_1}$. Let $[s_2] = tail(\rho^N_{sp})$. Then for all $i \geq 0$, $[s_2] \leq_{rat} \rho^{N+K\cdot i}_{s_1}$. Since M is a finite structure there exists a state $s_\spadesuit$ such that for infinitely many numbers i, $\rho^{N+K\cdot i}_{s_1} = s_\spadesuit$. Since ρ_{s_1} is fair, there are $i < j$ such that $\rho^{N+K\cdot i}_{s_1} = \rho^{N+K\cdot j}_{s_1} = s_\spadesuit$ and an index $N+K\cdot i \leq k \leq N+K\cdot j$ such that $\rho^k_{s_1}$ is a fair state.

Let $\rho_{s'}$ be the following trace: For all $0 \leq l \leq N + K \cdot i$, $\rho^l_{s'} = \rho^l_{s_1}$ and for all $l > N + K \cdot i$, $\rho^l_{s'} = \rho^{((l-N-K\cdot i) mod\ ((j-i)\cdot K))+N+K\cdot i}_{s_1}$. It is easy to see that $\rho_{s'}$ is a fair rational trace. Furthermore, the construction of $\rho_{s'}$ implies that for all $l \geq 0$, $tail(\rho^l_{sp}) \leq_{rat} \rho^l_{s'}$. □

Finally we prove Lemma 4.4: Assume that there exists a fair trace from s such that for every fair trace ρ_t from t, $\rho_s \not\leq_{rat} \rho_t$. By Lemma B.2, there

is no fair trace ρ_{tp} such that for all $i \geq 0$, $L^P(\rho^i_{tp}) = [\rho^i_s]$. We refer to $(s,[s])$ and $(t,[t])$ as two copies M^P_s and M^P_t of M^P where the former has $(s,[s])$ as a single initial state and the latter has $(t,[t])$ as a single initial state. Then the language of $M^P_s \setminus M^P_t$ is not empty. This implies that the language of $M^P_s \setminus M^P_t$ contains an ω-regular word. Thus, there exists an ω-regular word w_s in the language of $(s,[s])$ that is not in the language of $(t,[t])$. This implies that their exists a fair trace ρ'_{sP} that starts at $(s,[s])$ and $w_s = L^P(\rho'_{sP})$.

By Lemma B.3 there exists a rational fair trace ρ_s that starts at s, such that for all $i \geq 0, tail(\rho'^i_{sP}) \leq_{rat} \rho^i_s$. Assume to the contrary that there exists a fair trace ρ_t from t such that $\rho_s \leq_{rat} \rho_t$. Consider the trace ρ_{tP} such that for all $i \geq 0$, $head(\rho^i_{tP}) = \rho^i_t$ and $tail(\rho^i_{tP}) = tail(\rho'^i_{sP})$. Clearly, ρ_{tP} is a fair trace from $(t,[s])$. Furthermore, $L^P(\rho'_{sP}) = L^P(\rho_{tp})$, thus $L^P(\rho_{tP}) = w_s$. This implies that w_s is in the language of M^P_t, a contradiction. $\Box$