Fair Bisimulation* **

Thomas A. Henzinger*** Sriram K. Rajamani†

Abstract. Bisimulations enjoy numerous applications in the analysis of labeled transition systems. Many of these applications are based on two central observations: first, bisimilar systems satisfy the same branching-time properties; second, bisimilarity can be checked efficiently for finite-state systems. The local character of bisimulation, however, makes it difficult to address liveness concerns. Indeed, the definitions of fair bisimulation that have been proposed in the literature sacrifice locality, and with it, also efficient checkability. We put forward a new definition of fair bisimulation which does not suffer from this drawback.

The bisimilarity of two systems can be viewed in terms of a game played between a protagonist and an adversary. In each step of the infinite bisimulation game, the adversary chooses one system, makes a move, and the protagonist matches it with a move of the other system. Consistent with this game-based view, we call two fair transition systems bisimilar if in the bisimulation game, the infinite path produced in the first system is fair iff the infinite path produced in the second system is fair.

We show that this notion of fair bisimulation enjoys the following properties. First, fairly bisimilar systems satisfy the same formulas of the logics Fair-AFMC (the fair alternation-free μ-calculus) and Fair-CTL*. Therefore, fair bisimulations can serve as property-preserving abstractions for these logics and weaker ones, such as Fair-CTL and LTL. Indeed, Fair-AFMC provides an exact logical characterization of fair bisimilarity. Second, it can be checked in time polynomial in the number of states if two systems are fairly bisimilar. This is in stark contrast to all trace-based equivalences, which are traditionally used for addressing liveness but require exponential time for checking.

1 Introduction

In system analysis, a key question is when two systems should be considered equivalent. One way of answering this question is to consider a class of queries

* A preliminary version of this paper appeared in the *Proceedings of the Sixth Workshop on Tools and Algorithms for the Construction and Analysis of Systems* (TACAS 00), *Lecture Notes in Computer Science* **1785**, Springer-Verlag, 2000, pp. 299–314.

** This research was supported in part by the DARPA (NASA) grant NAG2-1214, the SRC contract 99-TJ-683.003, the MARCO grant 98-DT-660, the DARPA (MARCO) grant MDA972-99-1-0001, and the NSF CAREER award CCR-9501708.

*** University of California at Berkeley, tah@eecs.berkeley.edu

† Microsoft Research, sriram@microsoft.com

and to identify those systems which cannot be distinguished by any query from the considered class. Queries typically have the form "does a system satisfy a requirement specified in a given logic?" If one considers finite behaviors of systems, then a useful model is the labeled transition graph, whose states or transitions are labeled with observations, and the finest reasonable equivalence on labeled transition graphs is bisimilarity [Par80,Mil89]. On one hand, no μ-calculus query, no matter how complex, can distinguish bisimilar systems. On the other hand, bisimilarity is not too fine for constructing an abstract quotient system if branching-time properties are of interest. This is because simple Hennessy-Milner queries, which correspond to the quantifier-free subset of the μ-calculus, can distinguish systems that are not bisimilar.

If one wishes to consider infinite limit behaviors also, then the labeled transition graph needs to be equipped with fairness constraints. The most common fairness constraints have either Büchi form (requiring that a transition cannot be enabled forever without being taken) or Streett form (requiring that a transition cannot be enabled infinitely often without being taken). If we can observe whether a transition is enabled or taken —that is, if the query logic can refer to these events— then bisimilarity still captures the equivalence induced by branching-time queries. However, if, as is often the case in system design, the private (i.e., unobservable) part of the system state contributes both to whether a transition is enabled and to the result of the transition, then bisimilarity is too coarse for branching-time queries. For example, if we ask whether a system has an infinite fair behavior along which some observation repeats infinitely often, then the answer may be Yes and No, respectively, for two bisimilar systems, because infinite behaviors may be identical in their observations yet different in their fairness. (One should note that one solution, albeit a nonoptimal one, is simply to define bisimilarity with respect to an extended set of observations whose new elements make fairness observable. This solution is nonoptimal as the resulting "extended-bisimilarity" relation is generally too fine: there can be systems that are not extended-bisimilar, yet cannot be distinguished by queries that refer to the newly introduced observations in a restricted way, namely, only for checking if an infinite behavior is fair. An example of this is given in Section 5).

It is therefore not surprising that generalized notions of bisimilarity have been proposed which take into account fairness constraints. These notions generally have in common that they start from a query logic, such as Fair-CTL [ASB94] or Fair-CTL* [GL94] (where all path quantifiers range over fair behaviors only), and define the equivalence induced by that logic: two systems are equivalent iff no query can distinguish them. Unfortunately, the resulting equivalences are unsuitable for use in automatic finite-state tools, because checking equivalence between two systems is either not known to be polynomial (for Fair-CTL based bisimilarity) or known to be PSPACE-hard (for Fair-CTL* based bisimilarity) in the combined number of states [KV96]. This is in stark contrast to the unfair case, where bisimilarity for finite-state systems can be checked efficiently [PT87,KS90,CPS93].

Borrowing ideas from earlier work on fair simulations [HKR97], we show that a fair refinement of bisimilarity can be defined which (1) corresponds to a natural query logic and (2) can be checked efficiently. Our starting point is the game-based view of bisimilarity. The bisimilarity of two systems can be viewed in terms of a two-player game between a protagonist and an adversary. In each step of the game, the adversary chooses one of the systems together with a transition, and the protagonist must match the resulting observation by a transition of the other system. The game proceeds either until the protagonist cannot match, in which case the adversary wins, or for an infinite number of steps, in which case the protagonist wins. If the adversary has a winning strategy, then the two systems are not bisimilar; if the protagonist has a winning strategy, then the systems are bisimilar. In the presence of fairness constraints, we generalize this game as follows. If the bisimulation game is played for a finite number of steps, then the adversary wins as before. However, if the bisimulation game is played for an infinite number of steps, then the winner is determined differently. If the infinite paths traversed in the two systems are either both fair or both unfair, then the protagonist wins; otherwise the adversary wins. In other words, the objective of the protagonist is not only to match observations but also to match both the satisfaction and the violation of fairness constraints.

In Section 2, we define our notion of fair bisimilarity formally and show that it is finer than the previously proposed notions; that is, it distinguishes states that cannot be distinguished by Fair-CTL*. The main benefit of our definition is its efficient implementability in finite-state tools: it can be checked in time polynomial in the combined number of states if two systems are fairly bisimilar according to our definition. A tree-automata based algorithm is given in Section 3 together with its complexity analysis. In Section 4, we prove that two systems with Büchi or Streett constraints are fairly bisimilar, in our sense, iff they satisfy the same formulas of Fair-AFMC (the fair alternation-free μ-calculus). It follows that Fair-AFMC provides an exact logical characterization and a query language for our fair bisimilarity. Finally, in Section 5, we discuss several issues in constructing system abstractions using fair-bisimilarity quotients.

Related work. In process algebra, several preorders and equivalences on labeled transition systems have been defined to account for fairness and have been studied from axiomatic and denotational angles [BW90,HK96]. That line of research usually considers fairness in the context of divergence (infinitely many silent τ actions). By contrast, our model has no silent actions, and our notions of Büchi and Streett fairness are inspired from ω automata. Also, our focus is on efficient algorithms. In contrast, all fair preorders based on failures [BKO87] and testing [Hen87,EB95,NC95] are closely related to fair trace containment, and the problems of checking them are hard for PSPACE.

2 Defining Fair Bisimilarity, Game-theoretically

A (*Kripke*) *structure* is a 5-tuple $K = \langle \Sigma, W, \hat{w}, R, L \rangle$ with the following components:

- A finite alphabet Σ of observations. Usually, we have a finite set P of propositions and $\Sigma = 2^P$.
- A finite set W of states.
- An initial state $\hat{w} \in W$.
- A transition relation $R \subseteq W \times W$.
- A labeling function $L : W \to \Sigma$ that maps each state to an observation.

The structure K is *deterministic* if whenever $R(w, w_1)$ and $R(w, w_2)$ for $w_1 \neq w_2$, then $L(w_1) \neq L(w_2)$. For a state $w \in W$, a *w-run* of K is a finite or infinite sequence $\overline{w} = w_0 \cdot w_1 \cdot w_2 \cdots$ of states $w_i \in W$ such that $w_0 = w$ and $R(w_i, w_{i+1})$ for all $i \geq 0$. If $\overline{w} = w_0 \cdot w_1 \cdot w_2 \cdots w_n$ then $|\overline{w}|$ is n. If $\overline{w}$ is infinite, then $|\overline{w}|$ is ω. We write $\inf(\overline{w})$ for the set of states that occur infinitely often in $\overline{w}$. A *run* of K is a $\hat{w}$-run, for the initial state $\hat{w}$. Let σ be the a finite or infinite sequence.

A *fairness constraint* for K is a function that maps every infinite run of K to the binary set $\{fair, unfair\}$. We consider two kinds of fairness constraints:

- A *Büchi* constraint F is specified by a set $F_B \subseteq W$ of states. Then, for an infinite run $\overline{w}$ of K, we have $F(\overline{w}) = fair$ iff $\inf(\overline{w}) \cap F_B \neq \emptyset$. Büchi constraints can be used for specifying the weak fairness of transitions (e.g., a transition is infinitely often either taken or disabled).
- A *Streett* constraint F is specified by a set $F_S \subseteq 2^W \times 2^W$ of pairs of state sets. Then, for an infinite run $\overline{w}$ of K, we have $F(\overline{w}) = fair$ iff for every pair $\langle l, r \rangle \in F_S$, if $\inf(\overline{w}) \cap l \neq \emptyset$ then $\inf(\overline{w}) \cap r \neq \emptyset$. Streett constraints can be used for specifying the strong fairness of transitions (e.g., if a transition is infinitely often enabled, then it is infinitely often taken).

A *fair structure* $\mathcal{K} = \langle K, F \rangle$ consists of a structure K and a fairness constraint F for K. The fair structure $\mathcal{K}$ is a Büchi structure if F is a Büchi constraint, and $\mathcal{K}$ is a Streett structure if F is a Streett constraint. In particular, every Büchi structure is also a Streett structure. For a state $w \in W$, a *fair* w-run of $\mathcal{K}$ is either a finite w-run of K or an infinite w-run $\overline{w}$ of K such that $F(\overline{w}) = fair$. A fair run of $\mathcal{K}$ is a fair $\hat{w}$-run, for the initial state $\hat{w}$.

In the following, we consider two structures $K_1 = \langle \Sigma, W_1, \hat{w}_1, R_1, L_1 \rangle$ and $K_2 = \langle \Sigma, W_2, \hat{w}_2, R_2, L_2 \rangle$ over the same alphabet, and two fair structures $\mathcal{K}_1 = \langle K_1, F_1 \rangle$ and $\mathcal{K}_2 = \langle K_2, F_2 \rangle$.

Bisimulation

A binary relation $S \subseteq W_1 \times W_2$ is a *bisimulation* between K_1 and K_2 if the following three conditions hold [Par80,Mil89]:

1. If $S(w_1, w_2)$, then $L_1(w_1) = L_2(w_2)$.
2. If $S(w_1, w_2)$ and $R_1(w_1, w_1')$, then there is a state $w_2' \in W_2$ such that $R_2(w_2, w_2')$ and $S(w_1', w_2')$.
3. If $S(w_1, w_2)$ and $R_2(w_2, w_2')$, then there is a state $w_1' \in W_1$ such that $R_1(w_1, w_1')$ and $S(w_1', w_2')$.

The structures K_1 and K_2 are *bisimilar* if there is a bisimulation S between K_1 and K_2 such that $S(\hat{w}_1, \hat{w}_2)$. The problem of checking if K_1 and K_2 are bisimilar can be solved in time $O((|R_1| + |R_2|) \cdot \log(|W_1| + |W_2|))$ [PT87]. The following alternative definitions of bisimilarity are equivalent to the definition above.

The game-theoretic view. Consider a two-player game whose positions are pairs $\langle w_1, w_2 \rangle \in W_1 \times W_2$ of states. The initial position is $\langle \hat{w}_1, \hat{w}_2 \rangle$. The game is played between an adversary and a protagonist and it proceeds in a sequence of rounds. In each round, if $\langle w_1, w_2 \rangle$ is the current position, the adversary chooses a structure and makes a move that respects its transition relation. Then, the protagonist makes a matching move in the other structure. If the adversary chooses to move in K_1, and updates the first component w_1 to an R_1-successor w_1', then the protagonist must update the second component w_2 to some R_2-successor w_2' such that $L_1(w_1') = L_2(w_2')$. If no such w_2' exists, then the protagonist loses. Similarly, if the adversary chooses to move in K_2, and updates the second component w_2 to an R_2-successor w_2', then the protagonist must update the first component w_1 to some R_1-successor w_1' such that $L_1(w_1') = L_2(w_2')$. If no such w_1' exists, then the protagonist loses. If the game proceeds ad infinitum, for ω rounds, then the adversary loses. It is easy to see that K_1 and K_2 are bisimilar iff the protagonist has a winning strategy.

The temporal-logic view. Bisimilarity provides a fully abstract semantics for the branching-time logics CTL, CTL*, AFMC (the alternation-free fragment of the μ-calculus), and MC (the μ-calculus) [BCG88]. Formally, two structures K_1 and K_2 are bisimilar iff for every formula ψ of CTL (or CTL* or AFMC or MC), K_1 satisfies ψ iff K_2 satisfies ψ.

Previous definitions of fair bisimulation

In the literature, we find two extensions of bisimilarity that account for fairness constraints. The two extensions are motivated by the branching-time logics Fair-CTL and Fair-CTL*, which are interpreted over fair structures with the path quantifiers being restricted to the infinite runs that are fair [CES86].

CTL-**bisimulation** [ASB94]. A binary relation $S \subseteq W_1 \times W_2$ is a CTL-*bisimulation* between $\mathcal{K}_1$ and $\mathcal{K}_2$ if the following three conditions hold:

1. S is a bisimulation between K_1 and K_2.
2. If $S(w_1, w_2)$, then for every periodic fair w_1-run $\overline{w} = u_0 \cdot u_1 \cdot u_2 \cdots u_n \cdot (u_{n+1} \cdot u_{n+2} \cdots u_{n+k})^\omega$ of $\mathcal{K}_1$, there is a fair w_2-run $\overline{w}' = u_0' \cdot u_1' \cdot u_2' \cdots$ of $\mathcal{K}_2$ such that for $1 \leq i \leq n$ we have $S(u_i, u_i')$, and for $i > n$ there exists $u \in \inf(\overline{w})$ such that $S(u, u_i')$.
3. If $S(w_1, w_2)$, then for every periodic fair w_2-run $\overline{w}' = u_0' \cdot u_1' \cdot u_2' \cdots u_n' \cdot (u_{n+1}' \cdot u_{n+2}' \cdots u_{n+k}')^\omega$ of $\mathcal{K}_2$, there is a fair w_1-run $\overline{w} = u_0 \cdot u_1 \cdot u_2 \cdots$ of $\mathcal{K}_1$ such that for $1 \leq i \leq n$ we have $S(u_i, u_i')$, and for $i > n$ there exists $u' \in \inf(\overline{w}')$ such that $S(u_i, u')$.

The fair structures $\mathcal{K}_1$ and $\mathcal{K}_2$ are CTL-*bisimilar* if there is a CTL-bisimulation S between $\mathcal{K}_1$ and $\mathcal{K}_2$ such that $S(\hat{w}_1, \hat{w}_2)$. For Büchi or Streett constraints F_1 and F_2, the problem of checking if there is a CTL-bisimulation between $\mathcal{K}_1$ and $\mathcal{K}_2$ is known to be in PSPACE. No matching lower bound is known, but the best known algorithm has a time complexity exponential in the number of states. Two fair structures $\mathcal{K}_1$ and $\mathcal{K}_2$ are CTL-bisimilar iff for every formula ψ of Fair-CTL, $\mathcal{K}_1$ satisfies ψ iff $\mathcal{K}_2$ satisfies ψ [ASB94].

CTL*-**bisimulation** [ASB94,GL94]. A binary relation $S \subseteq W_1 \times W_2$ is a CTL*-*bisimulation* between $\mathcal{K}_1$ and $\mathcal{K}_2$ if the following three conditions hold:

1. If $S(w_1, w_2)$, then $L_1(w_1) = L_2(w_2)$.
2. If $S(w_1, w_2)$, then for every fair w_1-run $\overline{w} = u_0 \cdot u_1 \cdot u_2 \cdots$ of $\mathcal{K}_1$, there is a fair w_2-run $\overline{w}' = u'_0 \cdot u'_1 \cdot u'_2 \cdots$ of $\mathcal{K}_2$ such that $\overline{w}'$ *S-matches* $\overline{w}$; that is, $|\overline{w}'| = |\overline{w}|$ and $S(u_i, u'_i)$ for all $0 \leq i \leq |\overline{w}|$.
3. If $S(w_1, w_2)$, then for every fair w_2-run $\overline{w}' = u'_0 \cdot u'_1 \cdot u'_2 \cdots$ of $\mathcal{K}_2$, there is a fair w_1-run $\overline{w} = u_0 \cdot u_1 \cdot u_2 \cdots$ of $\mathcal{K}_1$ such that $\overline{w}'$ *S-matches* $\overline{w}$.

Every CTL*-bisimulation between $\mathcal{K}_1$ and $\mathcal{K}_2$ is a bisimulation between K_1 and K_2. The fair structures $\mathcal{K}_1$ and $\mathcal{K}_2$ are CTL*-*bisimilar* if there is a CTL*-bisimulation S between $\mathcal{K}_1$ and $\mathcal{K}_2$ such that $S(\hat{w}_1, \hat{w}_2)$. For Büchi or Streett constraints F_1 and F_2, the problem of checking if there is a CTL*-bisimulation between $\mathcal{K}_1$ and $\mathcal{K}_2$ is complete for PSPACE. In particular, the problem is PSPACE-hard in the combined number $|W_1| + |W_2|$ of states [KV96]. Two fair structures $\mathcal{K}_1$ and $\mathcal{K}_2$ are CTL*-bisimilar iff for every formula ψ of Fair-CTL*, $\mathcal{K}_1$ satisfies ψ iff $\mathcal{K}_2$ satisfies ψ [ASB94,GL94].

CTL*-bisimilarity is strictly stronger than CTL-bisimilarity [ASB94]. Formally, for all fair structures $\mathcal{K}_1$ and $\mathcal{K}_2$, if $\mathcal{K}_1$ and $\mathcal{K}_2$ are CTL*-bisimilar, then $\mathcal{K}_1$ and $\mathcal{K}_2$ are CTL-bisimilar. Moreover, there are two Büchi structures $\mathcal{K}_1$ and $\mathcal{K}_2$ such that $\mathcal{K}_1$ and $\mathcal{K}_2$ are CTL-bisimilar, but $\mathcal{K}_1$ and $\mathcal{K}_2$ are not CTL*-bisimilar. This is in contrast to the unfair case, where CTL and CTL* have the same distinguishing power on Kripke structures.

Our definition of fair bisimulation

Let $\mathcal{K}_1$ and $\mathcal{K}_2$ be fair structures. Recall the bisimulation game played between the adversary and the protagonist. A *strategy* τ is a pair of functions, $\tau = \langle \tau_1, \tau_2 \rangle$, where τ_1 is a partial function from $(W_1 \times W_2)^* \times W_1$ to W_2, and τ_2 is a partial function from $(W_1 \times W_2)^* \times W_2$ to W_1. The strategy is used by the protagonist to play a game against the adversary. The game proceeds as follows. The game starts at some position in $W_1 \times W_2$. If the game so far has produced the sequence $\pi \in (W_1 \times W_2)^*$ of positions, and $\langle u, u' \rangle$ is the last position in π, the adversary has two sets of choices. It can move either in K_1 or in K_2. If the adversary moves to w in K_1, such that $R_1(u, w)$, then the first component τ_1 of the strategy instructs the protagonist to move to $w' = \tau_1(\pi, w)$, where $R_2(u', w')$, thus resulting in the new position $\langle w, w' \rangle$. If the adversary moves to w' in K_2, such that $R_2(u', w')$ then the second component τ_2 of the strategy instructs

the protagonist to move to $w = \tau_2(\pi, w')$, where $R_1(u, w)$, thus resulting in the new position $\langle w, w' \rangle$. A finite or infinite sequence $\overline{w}$ is an *outcome* of the strategy τ if $\overline{w}$ results from letting the adversary make an arbitrary move at each step, and making the protagonist respond using τ in each step. Formally, $\overline{w} = \langle w_0, w'_0 \rangle \cdot \langle w_1, w'_1 \rangle \cdots \in (W_1 \times W_2)^* \cup (W_1 \times W_2)^\omega$ is an *outcome* of the strategy τ if for all $0 \leq i < |\overline{w}|$, either (1) $w'_{i+1} = \tau_1(\langle w_0, w'_0 \rangle \cdots \langle w_i, w'_i \rangle \cdot w_{i+1})$, or (2) $w_{i+1} = \tau_2(\langle w_0, w'_0 \rangle \cdots \langle w_i, w'_i \rangle \cdot w'_{i+1})$.

A binary relation $S \subseteq W_1 \times W_2$ is a *fair bisimulation* between $\mathcal{K}_1$ and $\mathcal{K}_2$ if the following two conditions hold:

1. If $S(w, w')$, then $L_1(w) = L_2(w')$.
2. There exists a strategy τ such that, if $S(u, u')$, then every outcome $\overline{w} = \langle w_0, w'_0 \rangle \cdot \langle w_1, w'_1 \rangle \cdots$ of τ with $w_0 = u$ and $w'_0 = u'$ has the following two properties: (1) for all $0 \leq i \leq |\overline{w}|$, we have $S(w_i, w'_i)$, and (2) the projection $w_0 \cdot w_1 \cdots$ of $\overline{w}$ to W_1 is a fair w_0-run of $\mathcal{K}_1$ iff the projection $w'_0 \cdot w'_1 \cdots$ of $\overline{w}$ to W_2 is a fair w'_0-run of $\mathcal{K}_2$.

Every fair bisimulation between $\mathcal{K}_1$ and $\mathcal{K}_2$ is a bisimulation between K_1 and K_2. The fair structures $\mathcal{K}_1$ and $\mathcal{K}_2$ are *fairly bisimilar* if there is a fair bisimulation S between $\mathcal{K}_1$ and $\mathcal{K}_2$ such that $S(\hat{w}_1, \hat{w}_2)$. In Section 3 we give an efficient (polynomial in the combined number of states) algorithm to check if two fair structures are fairly bisimilar. For two fair structures $\mathcal{K}_1$ and $\mathcal{K}_2$, we show in Section 4 that $\mathcal{K}_1$ and $\mathcal{K}_2$ are fairly bisimilar iff for every formula ψ of Fair-AFMC, $\mathcal{K}_1$ satisfies ψ iff $\mathcal{K}_2$ satisfies ψ. The following propositions state that fair bisimilarity is stronger than CTL*-bisimilarity.

Proposition 1. *For all fair structures $\mathcal{K}_1$ and $\mathcal{K}_2$, if $\mathcal{K}_1$ and $\mathcal{K}_2$ are fairly bisimilar, then $\mathcal{K}_1$ and $\mathcal{K}_2$ are* CTL**-bisimilar.*

Proposition 2. *There are two Büchi structures $\mathcal{K}_1$ and $\mathcal{K}_2$ such that $\mathcal{K}_1$ and $\mathcal{K}_2$ are* CTL**-bisimilar, but $\mathcal{K}_1$ and $\mathcal{K}_2$ are not fairly bisimilar.*

Proof. Consider the Büchi structures $\mathcal{K}_1$ and $\mathcal{K}_2$ shown in Figure 1 (the Büchi states are marked). Consider the relation $S \subseteq W_1 \times W_2$, where $S = \{(w, w') \mid w \in W_1, w' \in W_2, \text{ and } L_1(w) = L_2(w')\}$. It can be checked that S is a CTL*-bisimulation between $\mathcal{K}_1$ and $\mathcal{K}_2$. Consider the bisimulation game starting at position $\langle s_1, t_1 \rangle$. The adversary first chooses to move in $\mathcal{K}_2$ and moves to t''_2. The protagonist can respond by moving to either s_2 or s'_2. If the protagonist moves to s_2, then the adversary switches to $\mathcal{K}_1$ and moves to s_3, forcing the protagonist to move in $\mathcal{K}_2$ to t''_3. If the protagonist moves to s'_2, then the adversary switches to $\mathcal{K}_1$ and moves to s'_4, forcing the protagonist to move in $\mathcal{K}_2$ to t''_4. In both cases, the game goes back to the initial state $\langle s_1, t_1 \rangle$ in the next round. By repeating this sequence ad infinitum, the adversary ensures that the run produced in $\mathcal{K}_1$ is fair, while the run produced in $\mathcal{K}_2$ is not. Thus $\mathcal{K}_1$ and $\mathcal{K}_2$ are not fairly bisimilar. □

Our game-theoretic definition of fair bisimulation is inspired by the notion of fair simulation from [HKR97]. It should be noted that, as in the unfair case,

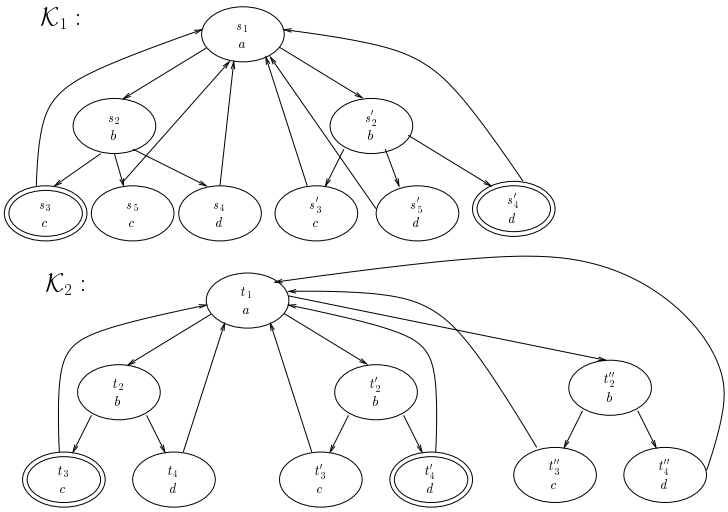


Fig. 1. Fair bisimilarity is stronger than CTL*-bisimilarity

fair bisimulation is stronger than mutual fair simulation. Consider again the two structures in Figure 1. Then $\mathcal{K}_1$ fairly simulates $\mathcal{K}_2$ and $\mathcal{K}_2$ fairly simulates $\mathcal{K}_1$, despite the fact that $\mathcal{K}_1$ and $\mathcal{K}_2$ are not fairly bisimilar. It should also be noted that, in the example of Figure 1, the adversary needs to switch between $\mathcal{K}_1$ and $\mathcal{K}_2$ infinitely often to win the fair-bisimulation game.

3 Checking Fair Bisimilarity, Efficiently

We present an algorithm for checking if two fair structures are fairly bisimilar. The time complexity of our algorithm is polynomial in the combined number of states. The algorithm exploits properties of a weak version of fair bisimulation, where the game is required to start at the initial states.

Init-fair bisimulation

A binary relation $S \subseteq W_1 \times W_2$ is an *init-fair bisimulation* between $\mathcal{K}_1$ and $\mathcal{K}_2$ if the following three conditions hold:

1. $S(\hat{w}_1, \hat{w}_2)$.
2. If $S(s,t)$, then $L_1(s) = L_2(t)$.
3. There exists a strategy τ such that every outcome $\overline{w} = \langle w_0, w'_0 \rangle \cdot \langle w_1, w'_1 \rangle \cdots$ of τ with $w_0 = \hat{w}_1$ and $w'_0 = \hat{w}_2$ has the following two properties: (1) for all $0 \leq i \leq |\overline{w}|$, we have $S(w_i, w'_i)$, and (2) the projection $w_0 \cdot w_1 \cdots$ of $\overline{w}$ to W_1 is a fair run of $\mathcal{K}_1$ iff the projection $w'_0 \cdot w'_1 \cdots$ of $\overline{w}$ to W_2 is a fair run of $\mathcal{K}_2$.

The fair structures $\mathcal{K}_1$ and $\mathcal{K}_2$ are *init-fairly bisimilar* if there is an init-fair bisimulation S between $\mathcal{K}_1$ and $\mathcal{K}_2$. Every fair bisimulation S between $\mathcal{K}_1$ and $\mathcal{K}_2$ with $S(\hat{w}_1, \hat{w}_2)$ is also an init-fair bisimulation between $\mathcal{K}_1$ and $\mathcal{K}_2$, but not every init-fair bisimulation is necessarily a fair bisimulation. Init-fair bisimulations are useful to us because of the following monotonicity property.

Proposition 3. *For all fair structures $\mathcal{K}_1 = \langle K_1, F_1 \rangle$ and $\mathcal{K}_2 = \langle K_2, F_2 \rangle$, if S is an init-fair bisimulation between $\mathcal{K}_1$ and $\mathcal{K}_2$, and $S' \supseteq S$ is a bisimulation between K_1 and K_2, then S' is also an init-fair bisimulation between $\mathcal{K}_1$ and $\mathcal{K}_2$.*

Moreover, checking for the existence of a fair bisimulation can be reduced to checking for the existence of an init-fair bisimulation.

Proposition 4. *For all fair structures $\mathcal{K}_1 = \langle K_1, F_1 \rangle$ and $\mathcal{K}_2 = \langle K_2, F_2 \rangle$, $\mathcal{K}_1$ and $\mathcal{K}_2$ are init-fairly bisimilar iff $\mathcal{K}_1$ and $\mathcal{K}_2$ are fairly bisimilar.*

The proofs of both propositions are similar to the simulation case [HKR97].

The algorithm

Given two structures $K_1 = \langle \Sigma, W_1, \hat{w}_1, R_1, L_1 \rangle$ and $K_2 = \langle \Sigma, W_2, \hat{w}_2, R_2, L_2 \rangle$, and two fair structures $\mathcal{K}_1 = \langle K_1, F_1 \rangle$ and $\mathcal{K}_2 = \langle K_2, F_2 \rangle$, we present an automata-based algorithm that checks, in time polynomial in K_1 and K_2, whether there is a fair bisimulation between $\mathcal{K}_1$ and $\mathcal{K}_2$.

A *coarsest bisimulation* between K_1 and K_2 is a binary relation $\hat{S} \subseteq W_1 \times W_2$ such that (1) $\hat{S}$ is a bisimulation between K_1 and K_2, and (2) for every bisimulation S between K_1 and K_2, we have $S \subseteq \hat{S}$. The following proposition, which follows from Propositions 3 and 4, reduces the problem of checking if there is a fair bisimulation between $\mathcal{K}_1$ and $\mathcal{K}_2$ to checking if the (unique) coarsest bisimulation between K_1 and K_2 is an init-fair bisimulation between $\mathcal{K}_1$ and $\mathcal{K}_2$.

Proposition 5. *For all fair structures $\mathcal{K}_1 = \langle K_1, F_1 \rangle$ and $\mathcal{K}_2 = \langle K_2, F_2 \rangle$, if $\hat{S}$ is the coarsest bisimulation between K_1 and K_2, then $\mathcal{K}_1$ and $\mathcal{K}_2$ are fairly bisimilar iff $\hat{S}$ is an init-fair bisimulation between $\mathcal{K}_1$ and $\mathcal{K}_2$.*

The coarsest bisimulation between K_1 and K_2 can be constructed in time $O((|R_1| + |R_2|) \cdot \log(|W_1| + |W_2|))$ using the Paige-Tarjan algorithm [PT87]. Hence, we are left to find an algorithm that efficiently checks, given a relation $S \subseteq W_1 \times W_2$, if S is an init-fair bisimulation between $\mathcal{K}_1$ and $\mathcal{K}_2$. For this purpose, consider the structure $K_S = \langle \Sigma_S, W, \hat{w}, R, L \rangle$ with the following components:

- $\Sigma_S = W_1 \cup W_2$. Thus, each state of K_S is labeled by a state of K_1 or K_2.
- $W = (S \times \{\mathbf{a}\}) \cup (W_1 \times W_2 \times \{1, 2\} \times \{\mathbf{p}\})$. Thus, there are two types of states: adversary-states, in which the W_1-component is related by S to the W_2-component, and protagonist-states, which are not restricted. We regard the states of K_S as positions in a game, with the adversary moving in adversary-states and the protagonist moving in protagonist-states. Since the adversary can choose to move either in $\mathcal{K}_1$ or in $\mathcal{K}_2$, we record this choice in the protagonist states. If the third component of a protagonist state is 1 (2), then the protagonist needs to make a move in $\mathcal{K}_2$ ($\mathcal{K}_1$).

- $\hat{w} = \langle \hat{w}_1, \hat{w}_2, \mathbf{a} \rangle$. This is the initial game position.
- $R = \{(\langle w_1, w_2, \mathbf{a} \rangle, \langle w_1', w_2, 1, \mathbf{p} \rangle) \mid R_1(w_1, w_1')\} \cup \{(\langle w_1, w_2, \mathbf{a} \rangle, \langle w_1, w_2', 2, \mathbf{p} \rangle) \mid R_2(w_2, w_2')\} \cup \{(\langle w_1, w_2, 2, \mathbf{p} \rangle, \langle w_1', w_2, \mathbf{a} \rangle) \mid R_1(w_1, w_1')\} \cup \{(\langle w_1, w_2, 1, \mathbf{p} \rangle, \langle w_1, w_2', \mathbf{a} \rangle) \mid R_2(w_2, w_2')\}$. Thus, the adversary and the protagonist alternate moves. The adversary moves along transitions that correspond to transitions of either K_1 or K_2. If the adversary makes a move along a transition of K_1 (K_2), the protagonist must reply with a move along a transition of K_2 (K_1). Since adversary-states consist only of pairs in S, the protagonist must reply to each move of the adversary with a move to a state $\langle w_1, w_2, \mathbf{a} \rangle$ for which $S(w_1, w_2)$.
- We label an adversary-state by its W_1-component and we label a protagonist-state by its W_2-component: $L(\langle w_1, w_2, \mathbf{a} \rangle) = \{w_1\}$, and $L(\langle w_1, w_2, \cdot, \mathbf{p} \rangle) = \{w_2\}$.

We say that a run $\overline{w}$ of K_S satisfies a fairness constraint F if $F(L(\overline{w})) = \mathit{fair}$. The protagonist wins the game on K_S if (1) whenever the game position is a protagonist-state, the protagonist can proceed with a move, and (2) whenever the game produces an infinite run of K_S, the run satisfies F_1 iff it satisfies F_2. Then, the protagonist has a winning strategy in this game iff S is an init-fair bisimulation between $\mathcal{K}_1$ and $\mathcal{K}_2$.

The problem of checking the existence of a winning strategy (and the synthesis of such a strategy) can be reduced to the nonemptiness problem for tree automata. We construct two tree automata:

1. The tree automaton $\mathcal{A}_S$ accepts all infinite $(W_1 \cup W_2)$-labeled trees that can be obtained by unrolling K_S and pruning it such that every adversary-state retains all its successors, and every protagonist-state retains exactly one of its successors. The intuition is that each tree accepted by $\mathcal{A}_S$ corresponds to a strategy of the protagonist. The automaton $\mathcal{A}_S$ has $O(|W_1| \cdot |W_2|)$ states, and it has a vacuous acceptance condition.
2. The tree automaton $\mathcal{A}_F$ accepts all $(W_1 \cup W_2)$-labeled trees in which all paths have the following property: F_1 is satisfied iff F_2 is satisfied. When $\mathcal{K}_1$ and $\mathcal{K}_2$ are Büchi structures, the automaton $\mathcal{A}_F$ can be defined as a Streett automaton with two states and two pairs in the Streett constraint. When $\mathcal{K}_1$ and $\mathcal{K}_2$ are Streett structures, the automaton $\mathcal{A}_F$ can be defined as a Streett automaton with $3^{(|F_1|+|F_2|)} \cdot |F_1| \cdot |F_2|$ states, and $3 \cdot (|F_1| + |F_2|)$ pairs in the Streett constraint.

The protagonist has a winning strategy iff the intersection of the Streett automata $\mathcal{A}_S$ and $\mathcal{A}_F$ is nonempty. To check this, we define and check the nonemptiness of the product automaton $\mathcal{A}_S \times \mathcal{A}_F$. Since $\mathcal{A}_S$ has a vacuous acceptance condition, the product automaton is a Streett automaton with the same number of pairs as $\mathcal{A}_F$. Finally, since checking the nonemptiness of a Streett tree automaton with n states and f pairs requires time $O(n^{(2f+1)} \cdot f!)$ [KV98], the theorem below follows.

Theorem 1. *Given two fair structures $\mathcal{K}_1$ and $\mathcal{K}_2$ with state sets W_1 and W_2, transition relations R_1 and R_2, and fairness constraints F_1 and F_2, we can check whether $\mathcal{K}_1$ and $\mathcal{K}_2$ are fairly bisimilar in time:*

- $O((|W_1| \cdot |W_2|)^5)$, *for Büchi structures.*
- $O(n^{(2f+1)} \cdot 3^{(2f^2+f)/3} \cdot f!)$, *where* $n = |W_1| \cdot |W_2| \cdot |F_1| \cdot |F_2|$ *and* $f = 3 \cdot (|F_1| + |F_2|)$, *for Streett structures.*

4 Characterizing Fair Bisimilarity, Logically

We show that fair bisimilarity characterizes precisely the distinguishing power of the fair alternation-free μ-calculus (Fair-AFMC). A formula of the μ-calculus (MC) is one of the following:

- **true**, **false**, p, or $\neg p$, for a proposition $p \in P$.
- y, for a propositional variable $y \in V$.
- $\varphi_1 \vee \varphi_2$ or $\varphi_1 \wedge \varphi_2$, where φ_1 and φ_2 are MC formulas.
- $\exists\bigcirc\, \varphi$ or $\forall\bigcirc\, \varphi$, where φ is a MC formula.
- $\mu y.f(y)$ or $\nu y.f(y)$, where $f(y)$ is a MC formula. All free occurrences of the variable y in $\mu y.f(y)$ and $\nu y.f(y)$ are bound by the initial fixpoint quantifier.

A MC formula is *alternation-free* if for all variables $y \in V$, there are respectively no occurrences of ν (μ) in any syntactic path from a binding occurrence μy (νy) to a corresponding bound occurrence of y. For example, the formula $\mu x.(p \vee \mu y.(x \vee \exists\bigcirc\, y))$ is alternation-free; the formula $\mu x.(p \vee \nu y.(x \wedge \exists\bigcirc\, y))$ is not. The AFMC formulas are the MC formulas that are alternation-free.

The semantics of AFMC is defined for formulas without free occurrences of variables. We interpret the closed AFMC formulas over fair structures, thus obtaining the logic Fair-AFMC. Unlike in Fair-CTL and Fair-CTL*, where the path quantifiers are restricted to fair runs, the μ-calculus does not explicitly refer to paths, and the definition of the satisfaction relation for Fair-AFMC is more involved. An AFMC formula can be thought of being evaluated by "unrolling" the fixpoint quantifiers; for example, $\nu y.f(y)$ is unrolled to $f(\nu y.f(y))$. Least-fixpoint (μ) quantifiers are unrolled a finite number of times, but greatest-fixpoint (ν) quantifiers are unrolled ad infinitum. In Fair-AFMC, we need to ensure that all ν-unrollings are fair. This is done formally using the notion of sat-trees.

The *closure* $cl(\psi)$ of a Fair-AFMC formula ψ is the least set of formulas that satisfies the following conditions:

- **true** $\in cl(\psi)$ and **false** $\in cl(\psi)$.
- $\psi \in cl(\psi)$.
- If $\varphi_1 \wedge \varphi_2$ or $\varphi_1 \vee \varphi_2$ is in $cl(\psi)$, then $\varphi_1 \in cl(\psi)$ and $\varphi_2 \in cl(\psi)$.
- If $\exists\bigcirc\, \varphi$ or $\forall\bigcirc\, \varphi$ is in $cl(\psi)$, then $\varphi \in cl(\psi)$.
- If $\mu y.f(y) \in cl(\psi)$, then $f(\mu y.f(y)) \in cl(\psi)$.
- If $\nu y.f(y) \in cl(\psi)$, then $f(\nu y.f(y)) \in cl(\psi)$.

Each Fair-AFMC formula ψ specifies a set of "obligations" —a subset of formulas in $cl(\psi)$— that need to be satisfied. The witness to the satisfaction of a formula is a tree called a sat-tree.

We first define labeled trees formally. A (finite or infinite) *tree* is a set $t \subseteq \mathbb{N}^*$ such that if $xn \in t$, for $x \in \mathbb{N}^*$ and $n \in \mathbb{N}$, then $x \in t$ and $xm \in t$ for all $0 \leq m < n$. The elements of t represent nodes: the empty word ϵ is the root of t, and for each node x, the nodes of the form xn, for $n \in \mathbb{N}$, are the children of x. The number of children of the node x is denoted by $deg(x)$. A *path* ρ of t is a finite or infinite set $\rho \subseteq t$ of nodes that satisfies the following three conditions: (1) $\epsilon \in \rho$, (2) for each node $x \in \rho$, there exists at most one $n \in \mathbb{N}$ with $xn \in \rho$, and (3) if $xn \in \rho$, then $x \in \rho$. Given a set A, an *A-labeled tree* is a pair $\langle t, \lambda \rangle$, where t is a tree and $\lambda : t \to A$ is a labeling function that maps each node of t to an element in A. Then, every path $\rho = \{\epsilon, n_0, n_0 n_1, n_0 n_1 n_2, \ldots\}$ of t generates a sequence $\lambda(\rho) = \lambda(\epsilon) \cdot \lambda(n_0) \cdot \lambda(n_0 n_1) \cdots$ of elements in A.

Given a fair structure $\mathcal{K} = \langle K, F \rangle$ with $K = \langle \Sigma, W, w, R, L \rangle$, and a Fair-AFMC formula ψ, a *sat-tree* $\langle t, \lambda \rangle$ of $\mathcal{K}$ for ψ is a $(W \times cl(\psi))$-labeled tree $\langle t, \lambda \rangle$ that satisfies the following conditions:

- $\lambda(\epsilon) = \langle \hat{w}, \psi \rangle$. Thus, the root of the tree, which corresponds to the initial obligation, is labeled by the initial state of K and ψ itself.
- If $\lambda(x) = \langle w, \mathbf{false} \rangle$ or $\lambda(x) = \langle w, \mathbf{true} \rangle$, then $deg(x) = 0$.
- If $\lambda(x) = \langle w, p \rangle$, where $p \in P$, then $deg(x) = 1$. If $p \in L(w)$, then $\lambda(x0) = \langle w, \mathbf{true} \rangle$; otherwise $\lambda(x0) = \langle w, \mathbf{false} \rangle$.
- If $\lambda(x) = \langle w, \neg p \rangle$, where $p \in P$, then $deg(x) = 1$. If $p \in L(w)$, then $\lambda(x0) = \langle w, \mathbf{false} \rangle$; otherwise $\lambda(x0) = \langle w, \mathbf{true} \rangle$.
- If $\lambda(x) = \langle w, \varphi_1 \vee \varphi_2 \rangle$, then $deg(x) = 1$ and $\lambda(x0) \in \{\langle w, \varphi_1 \rangle, \langle w, \varphi_2 \rangle\}$.
- If $\lambda(x) = \langle w, \varphi_1 \wedge \varphi_2 \rangle$, then $deg(x) = 2$, $\lambda(x0) = \langle w, \varphi_1 \rangle$, and $\lambda(x1) = \langle w, \varphi_2 \rangle$.
- If $\lambda(x) = \langle w, \exists \bigcirc \varphi \rangle$, then $deg(x) = 1$ and $\lambda(x0) \in \{\langle w', \varphi \rangle \mid R(w, w')\}$.
- If $\lambda(x) = \langle w, \forall \bigcirc \varphi \rangle$, and $\{w_0, w_1, \ldots, w_n\}$ are the successors of w in K, in some arbitrary (but fixed) order, then $deg(x) = n + 1$, and for $0 \leq i \leq n$, we have $\lambda(xi) = \langle w_i, \varphi \rangle$.
- If $\lambda(x) = \langle w, \nu y.f(y) \rangle$, then $deg(x) = 1$ and $\lambda(x0) = \langle w, f(\nu y.f(y)) \rangle$.
- If $\lambda(x) = \langle w, \mu y.f(y) \rangle$, then $deg(x) = 1$ and $\lambda(x0) = \langle w, f(\mu y.f(y)) \rangle$.

Consider a sat-tree $\langle t, \lambda \rangle$ of $\mathcal{K}$ for ψ. If $\langle t, \lambda \rangle$ contains no node labeled $\langle w, \mathbf{false} \rangle$, then it provides a witness to the satisfaction of all local obligations induced by ψ. In addition, we have to make sure that least-fixpoint obligations are not propagated forever, and that greatest-fixpoint obligations are satisfied along fair runs of $\mathcal{K}$. Formally, the sat-tree $\langle t, \lambda \rangle$ of $\mathcal{K}$ for ψ is *convincing* if the following three conditions hold:

1. The sat-tree $\langle t, \lambda \rangle$ contains no node labeled $\langle w, \mathbf{false} \rangle$. Thus, all local obligations induced by ψ are satisfied.
2. For all infinite paths ρ of $\langle t, \lambda \rangle$, the projection of $\lambda(\rho)$ on the $cl(\psi)$-component contains only finitely many occurrences of formulas of the form $\mu y.f(y)$. Thus, no least-fixpoint obligation is propagated forever.

3. For all infinite paths ρ of $\langle t, \lambda \rangle$, the projection of $\lambda(\rho)$ on the W-component satisfies the fairness constraint F of $\mathcal{K}$.

The fair structure $\mathcal{K}$ *satisfies* the Fair-AFMC formula ψ if there is a convincing sat-tree $\langle t, \lambda \rangle$ of $\mathcal{K}$ for ψ.

If $\mathcal{K}_1$ and $\mathcal{K}_2$ are not fairly bisimilar, we can construct a Fair-AFMC formula ψ such that $\mathcal{K}_1$ satisfies ψ and $\mathcal{K}_2$ does not satisfy ψ. Consider the structures from Figure 1. The formula $\nu z.\forall\bigcirc\ (\exists\bigcirc\ (c \wedge \exists\bigcirc\ z) \vee \exists\bigcirc\ (d \wedge \exists\bigcirc\ z))$ is satisfied in $\mathcal{K}_1$ and not satisfied in $\mathcal{K}_2$. Conversely, if $\mathcal{K}_1$ and $\mathcal{K}_2$ are bisimilar, and $\mathcal{K}_1$ satisifes a Fair-AFMC formula ψ, we can use the convincing sat-tree of $\mathcal{K}_1$ for ψ and the winning strategy of the bisimulation game, to construct a convincing sat-tree of $\mathcal{K}_2$ for ψ.

Theorem 2. *For all fair structures $\mathcal{K}_1$ and $\mathcal{K}_2$, the following two statements are equivalent:*

1. *$\mathcal{K}_1$ and $\mathcal{K}_2$ are fairly bisimilar.*
2. *For every formula ψ of Fair-AFMC, $\mathcal{K}_1$ satisfies ψ iff $\mathcal{K}_2$ satisfies ψ.*

It is an open problem if the full μ-calculus over fair structures (Fair-MC) can be defined in a meaningful way, and to characterize its distinguishing power. In particular, condition 2 in the definition of convincing sat-trees for Fair-AFMC is no longer appropriate in the presence of alternating fixpoint quantifiers.

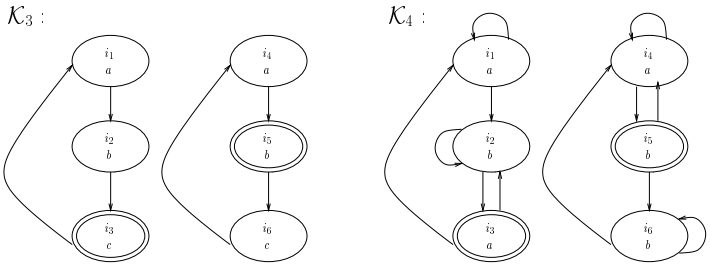
5 Discussion

An important topic that we have not addressed in this paper is the construction of fair abstractions. Here, we discuss some issues and difficulties in doing this. Let $K = \langle \Sigma, W, \hat{w}, R, L \rangle$ be a structure. Let $E \subseteq W \times W$ be an equivalence relation that is *observation-preserving*, i.e., if $E(s,t)$, then $L(s) = L(t)$. We define the *quotient* of K with respect of E, denoted $K/_E = \langle \Sigma, W', \hat{w}', R', L' \rangle$, as follows:

- The state set is $W' = W/_E$, the set of equivalence classes of W with respect to E. We denote the equivalence class of state $w \in W$ by $[w]_E$.
- The initial state is $\hat{w}' = [\hat{w}]_E$.
- The transition relation is $R' = \{([w]_E, [w']_E) \mid R(w, w')\}$.
- The labeling function L' is given by $L'([w]_E) = L(w)$. Note that L' is well-defined, because E is observation-preserving.

If S is the coarsest bisimulation between K and K, then $K/_S$ is called the *bisimilarity quotient* of K. It is not difficult to check that K and $K/_S$ are bisimilar, and that $K/_S$ is the smallest structure that is bisimilar to K. Since the construction of $K/_S$ is efficient, it may be a useful preprocessing step for model checking CTL, CTL*, and the μ-calculus.

Let $\mathcal{K} = \langle K, F \rangle$ be a fair structure. We are interested in finding a fair structure $\mathcal{K}'$ which (1) is fairly bisimilar to $\mathcal{K}$, and (2) has fewer states than $\mathcal{K}$. Such a $\mathcal{K}'$ is an abstraction of $\mathcal{K}$ which preserves all Fair-AFMC properties, and by



(a) $abs(\mathcal{K}_3)$ is not minimal

(b) $fabs(\mathcal{K}_4)$ is not fairly bisimilar to $\mathcal{K}$

Fig. 2. Constructing minimal fairly bisimilar abstractions

Proposition 1, also all Fair-CTL* properties. If the construction of $\mathcal{K}'$ is efficient, then it may be a useful preprocessing step for Fair-AFMC and Fair-CTL* model checking. We present two attempts at defining $\mathcal{K}'$ and point out why neither attempt is satisfactory.

The first attempt makes the fair states observable before constructing a minimal quotient. This attempt produces a fairly bisimilar abstraction, but not necessarily a minimal one. Define the binary relation $H \subseteq W \times W$ such that $H(w, w')$ iff (1) $L(w) = L(w')$, and (2) the fairness constraint F treats w and w' identically (i.e, if F is a Büchi constraint, then $w \in F$ iff $w' \in F$; if F is a Streett constraint, then for every Streett pair $\langle l, r\rangle$, we have $w \in l$ iff $w' \in l$, and $w \in r$ iff $w' \in r$). Clearly, H is an equivalence relation. Let $\hat{H} \subseteq W \times W$ be the coarsest bisimulation between K and K that refines H. Let $abs(K) = \langle K/_{\hat{H}}, F'\rangle$, where F' is obtained by lifting the fairness constraint F to $K/_{\hat{H}}$. Formally, given a set $A \subseteq W$ of states, define $\alpha(A) = \{[w]_{\hat{H}} \mid [w]_{\hat{H}} \cap A \neq \emptyset\}$. If F is a Büchi constraint, let $F' = \alpha(F)$; if F is a Streett constraint, let $F' = \{\langle \alpha(l), \alpha(r)\rangle \mid \langle l, r\rangle \in F\}$. It can be checked that $\mathcal{K}$ and $abs(K)$ are fairly bisimilar. However, $abs(K)$ is, in general, not the minimal fair structure which is fairly bisimilar to $\mathcal{K}$. For example, consider the Büchi structure $\mathcal{K}_3$ of Figure 2(a). In this example, $abs(K_3)$ is isomorphic to $\mathcal{K}_3$. But we can merge the states i_1 and i_4 to produce a fairly bisimilar abstraction which has only 5 states, and thus is smaller.

The second attempt constructs a minimal fair quotient, which is then equipped with a fairness constraint. However, there are cases where the straight-forward way of equipping the fair quotient with a fairness constraint does not result in a fairly bisimilar system. Let $S \subseteq W \times W$ be the coarsest fair bisimulation between $\mathcal{K}$ and $\mathcal{K}$. Define the relation $J \subseteq W \times W$ such that $J(w, w')$ iff (1) $S(w, w')$, and (2) the fairness constraint F treats w and w' identically. Clearly, J is an equivalence relation. Let $fabs(\mathcal{K}) = \langle K/_J, F'\rangle$, where F' is obtained by lifting the fairness constraint F to $K/_J$. Returning to the structure $\mathcal{K}_3$ of Figure 2(a), we find that $fabs(\mathcal{K}_3)$ indeed merges i_1 and i_4 and produces a fairly bisimilar

abstraction with 5 states. However, for the Büchi structure $\mathcal{K}_4$ of Figure 2(b), $fabs(\mathcal{K}_4)$ and $\mathcal{K}_4$ are not fairly bisimilar.

It therefore remains an open problem to construct, in general, a minimal structure which is fairly bisimilar to $\mathcal{K}$ (where minimality is measured in the number of states).

References

[ASB94] A. Aziz, V. Singhal, F. Balarin, R.K. Brayton, and A.L. Sangiovanni-Vincentelli. Equivalences for fair kripke structures. In S. Abiteboul and E. Shamir, editors, *ICALP 94: International Colloquium on Automata, Languages, and Programming*, Lecture Notes in Computer Science 820, pages 364–375. Springer-Verlag, 1994.

[BCG88] M.C. Browne, E.M. Clarke, and O. Grumberg. Characterizing finite Kripke structures in propositional temporal logic. *Theoretical Computer Science*, 59:115–131, 1988.

[BKO87] J.A. Bergstra, J.W. Klop, and E.R. Olderog. Failures without chaos: a new process semantics for fair abstraction. In *Formal Description Techniques III*, pages 77–103. Elsevier, 1987.

[BW90] J.C.M. Baeten and W.P. Weijland. *Process Algebra.* Cambridge University Press, 1990.

[CES86] E.M. Clarke, E.A. Emerson, and A.P. Sistla. Automatic verification of finite-state concurrent systems using temporal-logic specifications. *ACM Transactions on Programming Languages and Systems*, 8(2):244–263, 1986.

[CPS93] R.J. Cleaveland, J. Parrow, and B. Steffen. The Concurrency Workbench: a semantics-based tool for the verification of finite-state systems. *ACM Transactions on Programming Languages and Systems*, 15(1):36–72, 1993.

[EB95] W. Vogler E. Brinksma, A. Rensink. Fair testing. In I. Lee and S. Smolka, editors, *CONCUR 95: Theories of Concurrency*, Lecture Notes in Computer Science 962, pages 313–327. Springer-Verlag, July 1995.

[GL94] O. Grumberg and D.E. Long. Model checking and modular verification. *ACM Transactions on Programming Languages and Systems*, 16(3):843–871, 1994.

[Hen87] M.C.B. Hennessy. An algebraic theory of fair asynchronous communicating processes. *Theretical Computer Science*, 49:121–143, 1987.

[HK96] M. Huth and M. Kwiatkowska. The semantics for fair recursion with divergence. Technical Report CSR-96-4, University of Birmingham, 1996.

[HKR97] T.A. Henzinger, O. Kupferman, and S. K. Rajamani. Fair simulation. In *CONCUR 97: Theories of Concurrency*, Lecture Notes in Computer Science 1243, pages 273–287. Springer-Verlag, July 1997.

[KS90] P.C. Kanellakis and S.A. Smolka. CCS expressions, finite-state processes, and three problems of equivalence. *Information and Computation*, 86:43–68, 1990.

[KV96] O. Kupferman and M.Y. Vardi. Verification of fair transition systems. In R. Alur and T.A. Henzinger, editors, *CAV 96: Computer Aided Verification*, Lecture Notes in Computer Science 1102, pages 372–381. Springer-Verlag, 1996.

[KV98] O. Kupferman and M.Y. Vardi. Weak alternating automata and tree automata emptiness. In *Proceedings of the 30th ACM Symposium on Theory of Computing*, pages 224–233. ACM Press, 1998.

[Mil89] R. Milner. *Communication and Concurrency.* Prentice-Hall, 1989.

[NC95] V. Natarajan and R. Cleaveland. Divergence and fair testing. In *ICALP '95: International Colloquium on Automata, Languages, and Programming*, Lecture Notes in Computer Science 944, pages 648–659. Springer-Verlag, 1995.

[Par80] D.M.R. Park. *Concurrency and Automata on Infinite Sequences.* Lecture Notes in Computer Science 104. Springer-Verlag, 1980.

[PT87] R. Paige and R.E. Tarjan. Three partition-refinement algorithms. *SIAM Journal of Computing*, 16(6):973–989, 1987.