Multipebble Simulations for Alternating Automata
(Extended Abstract)

Lorenzo Clemente and Richard Mayr

LFCS. School of Informatics. University of Edinburgh. UK

Abstract. We study generalized simulation relations for alternating Büchi automata (ABA), as well as alternating finite automata. Having multiple pebbles allows the Duplicator to "hedge her bets" and delay decisions in the simulation game, thus yielding a coarser simulation relation. We define (k_1, k_2)-simulations, with k_1/k_2 pebbles on the left/right, respectively. This generalizes previous work on ordinary simulation (i.e., $(1, 1)$-simulation) for nondeterministic Büchi automata (NBA) in [4] and ABA in [5], and $(1, k)$-simulation for NBA in [3].
We consider direct, delayed and fair simulations. In each case, the (k_1, k_2)-simulations induce a complete lattice of simulations where $(1, 1)$- and (n, n)-simulations are the bottom and top element (if the automaton has n states), respectively, and the order is strict. For any fixed k_1, k_2, the (k_1, k_2)-simulation implies (ω-)language inclusion and can be computed in polynomial time. Furthermore, quotienting an ABA w.r.t. $(1, n)$-delayed simulation preserves its language. Finally, multipebble simulations yield new insights into the Miyano-Hayashi construction [10] on ABA. A technical report with full proofs is available [2].

1 Introduction

We consider simulation relations on (alternating) finite- and infinite word automata: nondeterministic finite automata (NFA), alternating finite automata (AFA), nondeterministic Büchi automata (NBA) and alternating Büchi automata (ABA). Simulation preorder is a notion of semantic comparison of two states, called left state and right state, in automata, where the larger right state can match all moves of the smaller left one in a stepwise way. Simulation preorder implies language inclusion on NFA/AFA/NBA/ABA [4, 5], but not vice-versa. While checking language inclusion is PSPACE-complete for all these classes of automata [8, 11], the simulation relation can be computed in polynomial time [4, 5].

Checking simulation preorder between two states can be presented as a game with two players, Spoiler and Duplicator, where Spoiler tries to prove that the simulation relation does not hold while Duplicator has the opposite objective. In every round of the simulation game, Spoiler chooses a transition from the current left state and Duplicator must choose a transition from the current right state which has the same action label. Duplicator wins iff the game respects the accepting states in the automata, and different requirements for this yield finer or coarser simulation relations. In *direct simulation*, whenever the left state is accepting, the right state must be accepting. In *delayed simulation*, whenever the left state is accepting, the right state must be eventually accepting. In *fair simulation*, if the left state is accepting infinitely often, then the right state must be accepting infinitely often. For finite-word automata, only direct simulation is meaningful, but for Büchi automata delayed and fair simulation yield coarser relations; see [4] for an overview.

These notions have been extended in two directions. Etessami [3] defined a hierarchy of $(1, k)$ multipebble simulations on NBA. Intuitively, the k pebbles on the right side allow Duplicator to "hedge her bets" and thus to delay making decisions. This extra power of Duplicator increases with larger k and yields coarser simulation relations.

A different extension by Wilke and Fritz [5] considered simulations on ABA. In an ABA, a state is either existential or universal. The idea is that Spoiler moves from existential left states and universal right states, and dually for Duplicator.

Our contribution. We consider (k_1, k_2)-simulations on ABA, i.e., with multiple pebbles on both sides: k_1 on the left and k_2 on the right. Intuitively, Duplicator controls pebbles on universal states on the left and existential states on the right (and dually for Spoiler). This generalizes all previous results: the $(1, k)$-simulations on NBA of [3] and the $(1, 1)$-simulations on ABA of [5].

For each acceptance condition (direct, delayed, fair) this yields a lattice-structured hierarchy of (k_1, k_2)-simulations, where $(1, 1)$- and (n, n)-simulations are the bottom and top element if the automaton has n states. Furthermore, the order is strict, i.e., more pebbles make the simulation relation strictly coarser in general. For each fixed $k_1, k_2 \geq 0$, (k_1, k_2)-simulations are computable in polynomial time and they imply language inclusion (over finite or infinite words, depending on the type of simulation).

Quotienting AFA w.r.t. (k_1, k_2)-simulation preserves their language. We also provide a corresponding result for ABA by showing that quotienting ABA w.r.t. $(1, n)$-delayed simulation preserves the ω-language. This is a non-trivial result, since a naïve generalization of the definition of semielective-quotients [5] does not work. We provide the correct notion of semielective-quotients for $(1, n)$-simulations on ABA, and show its correctness. Moreover, unlike for NBA [3], quotienting ABA w.r.t. $(1, k)$ delayed simulation for $1 < k < n$ does *not* preserve their language in general.

Finally, multipebble simulations have close connections to various determinization-like constructions like the subset construction for NFA/AFA and the Miyano-Hayashi construction [10] on ABA. In particular, multipebble simulations yield new insights into the Miyano-Hayashi construction and an alternative correctness proof showing an even stronger property. For full proofs, please see the technical report [2].

2 Preliminaries and Basic Definitions

Automata. An alternating Büchi automaton (ABA) $\mathcal{Q}$ is a tuple $(Q, \Sigma, q_I, \Delta, E, U, F)$, where Q is a finite set of states, Σ is a finite alphabet, q_I is the initial state, $\{E, U\}$ is a partition of Q into *existential* and *universal* states, $\Delta \subseteq Q \times \Sigma \times Q$ is the transition relation and $F \subseteq Q$ is the set of accepting states. We say that a state q is accepting if $q \in F$. We use n to denote the cardinality of Q. A nondeterministic Büchi automaton (NBA) is an ABA with $U = \emptyset$, i.e., where all choices are existential. We say that $\mathcal{Q}$ is *complete* iff $\forall (q, a) \in Q \times \Sigma.\ \exists (q, a, q') \in \Delta$.

An ABA $\mathcal{Q}$ recognizes a language of infinite words $\mathcal{L}^\omega(\mathcal{Q})$. The acceptance condition is best described in a game-theoretic way [6]. Given an input word $w \in \Sigma^\omega$, the *acceptance game* $\mathbb{G}^\omega(\mathcal{Q}, w)$ is played by two players, Pathfinder and Automaton. Existential states are controlled by Automaton, while Pathfinder controls universal states. Automaton wins the game $\mathbb{G}^\omega(\mathcal{Q}, w)$ iff she has a winning strategy s.t., for any

Pathfinder counter-strategy, the resulting computation visits some accepting state in F infinitely often. The language $\mathcal{L}^{\omega}(\mathcal{Q})$ recognized by $\mathcal{Q}$ is defined as the set of words $w \in \Sigma^{\omega}$ s.t. Automaton wins $\mathbb{G}^{\omega}(\mathcal{Q}, w)$. See [5] for a formal definition.

If we view an ABA $\mathcal{Q}$ as an acceptor of *finite* words, then we obtain an alternating finite automaton (AFA). For $w = w_0 \dots w_m \in \Sigma^*$, the finite acceptance game $\mathbb{G}^{\mathrm{fin}}(\mathcal{Q}, w)$ is defined as above for $\mathbb{G}^{\omega}(\mathcal{Q}, w)$, except that the game stops when the last symbol w_m of w has been read, and Automaton wins if the last state is in F. $\mathcal{L}^{\mathrm{fin}}(\mathcal{Q})$ is defined in the obvious way. An alternating transition system (ATS) $\mathcal{Q}$ is an AFA where all states are accepting, and $\mathcal{T}r(\mathcal{Q}) := \mathcal{L}^{\mathrm{fin}}(\mathcal{Q})$ is its trace language. When we just say "automaton", it can be an ABA, AFA or ATS, depending on the context.

If Q is a set, with 2^Q we denote the set of subsets of Q, and, for any $k \in \mathbb{N}$, with $2^{Q,k}$ we denote the subset of 2^Q consisting of elements of cardinality at most k. When drawing pictures, we represent existential states by $\textcircled{q}$ and universal states by $\boxed{q}$.

Multipebble simulations. We define multipebble simulations in a game-theoretic way. The game is played by two players, Spoiler and Duplicator, who play in rounds. The objective of Duplicator is to show that simulation holds, while Spoiler has the complementary objective. We use the metaphor of pebbles for describing the game: We call a pebble existential if it is on an existential state, and universal otherwise; *Left* if it is on the l.h.s. of the simulation relation, and *Right* otherwise. Intuitively, Spoiler controls existential *Left* pebbles and universal *Right* pebbles, while Duplicator controls universal *Left* pebbles and existential *Right* pebbles. The presence of > 1 pebbles in each side is due to the further ability of Duplicator to split pebbles to several successors. Moreover, Duplicator always has the possibility of "taking pebbles away". Since not all available pebbles have to be on the automaton, $k+1$ pebbles are at least as good as k.

Formally, let $\mathcal{Q}$ be an alternating automaton, $\mathbf{q}_0 \in 2^{Q,k_1}$ a k_1-set and $\mathbf{s}_0 \in 2^{Q,k_2}$ a k_2-set. We define the basic (k_1, k_2)-simulation game $\mathbb{G}_{(k_1,k_2)}(\mathbf{q}_0, \mathbf{s}_0)$ as follows. Let Γ^{Sp} and Γ^{Dup} be a set of actions (or transitions) for the two players (to be specified below). In the initial configuration $\langle \mathbf{q}_0, \mathbf{s}_0 \rangle$, *Left* pebbles are on $\mathbf{q}_0$ and *Right* pebbles on $\mathbf{s}_0$. If the current configuration at round i is $\langle \mathbf{q}_i, \mathbf{s}_i \rangle$, then the next configuration $\langle \mathbf{q}_{i+1}, \mathbf{s}_{i+1} \rangle$ is determined as follows:

- Spoiler chooses a transition $(\mathbf{q}_i, \mathbf{s}_i, a_i, \mathbf{q}', \mathbf{s}') \in \Gamma^{\mathrm{Sp}}$.
- Duplicator chooses a transition $(\mathbf{q}_i, \mathbf{s}_i, a_i, \mathbf{q}', \mathbf{s}', \mathbf{q}_{i+1}, \mathbf{s}_{i+1}) \in \Gamma^{\mathrm{Dup}}$.

We now define the two transition relations. Let $\mathbf{q}^E := \mathbf{q} \cap E$ be the set of existential states in $\mathbf{q}$, and define $\mathbf{q}^U, \mathbf{s}^E, \mathbf{s}^U$ similarly. Let $P_1 := 2^{Q,k_1} \times 2^{Q,k_2}$ and $P_0 := \Sigma \times 2^{Q,k_1} \times 2^{Q,k_2}$. $\Gamma^{\mathrm{Sp}} \subseteq P_1 \times P_0$ models Spoiler's moves: $(\mathbf{q}, \mathbf{s}, a, \mathbf{q}', \mathbf{s}') \in \Gamma^{\mathrm{Sp}}$ iff Spoiler chooses a as the next input symbol, and

- $\mathbf{q}'$ is obtained from $\mathbf{q}^E$ by choosing a successor for *each pebble* in $\mathbf{q}^E$. Formally, $\mathbf{q}' = \{\, \mathrm{select}(\Delta(q, a)) \mid q \in \mathbf{q}^E \,\}$, where $\mathrm{select}(\mathbf{r})$ chooses an element in $\mathbf{r}$.
- Similarly, $\mathbf{s}'$ is obtained from $\mathbf{s}^U$ by choosing a successor for each pebble in $\mathbf{s}^U$.

Duplicator's moves are of the form $(\mathbf{q}, \mathbf{s}, a, \mathbf{q}', \mathbf{s}', \mathbf{q}'', \mathbf{s}'') \in \Gamma^{\mathrm{Dup}} \subseteq P_1 \times P_0 \times P_1$:

- $\mathbf{q}''$ is a non-empty k_1-subset of $\mathbf{q}' \cup \Delta(\mathbf{q}^U, a)$, and
- $\mathbf{s}''$ is a non-empty k_2-subset of $\mathbf{s}' \cup \Delta(\mathbf{s}^E, a)$.

Notice that Duplicator is always allowed to "take pebbles away", and to "hedge her bets" by splitting pebbles into different successors. We say that a pebble on state q is *stuck* if q has no a-successor (where a is clear from the context).

We now formally define strategies. A strategy for Spoiler is a function $\delta : P_1^* P_1 \mapsto P_0$ *compatible with* Γ^{Sp}, i.e., for any $(\pi \cdot \langle \mathbf{q}, \mathbf{s} \rangle) \in P_1^* P_1$, $\delta(\pi \cdot \langle \mathbf{q}, \mathbf{s} \rangle) = (a, \mathbf{q}', \mathbf{s}')$ implies $(\mathbf{q}, \mathbf{s}, a, \mathbf{q}', \mathbf{s}') \in \Gamma^{\mathrm{Sp}}$. Similarly, a strategy for Duplicator is a function $\sigma : P_1^* P_1 \mapsto (P_0 \mapsto P_1)$ *compatible with* Γ^{Dup}, i.e., for any $\pi \in P_1^* P_1$ and $(a, \mathbf{q}', \mathbf{s}') \in P_0$, $\sigma(\pi)(a, \mathbf{q}', \mathbf{s}') = \langle \mathbf{q}'', \mathbf{s}'' \rangle$ implies $(\mathbf{q}, \mathbf{s}, a, \mathbf{q}', \mathbf{s}', \mathbf{q}'', \mathbf{s}'') \in \Gamma^{\mathrm{Dup}}$. A play $\pi = \langle \mathbf{q}_0, \mathbf{s}_0 \rangle \langle \mathbf{q}_1, \mathbf{s}_1 \rangle \cdots \in P_1^* \cup P_1^\omega$ is a finite or infinite sequence of configurations in P_1. For a word $w = a_0 a_1 \cdots \in \Sigma^* \cup \Sigma^\omega$ s.t. $|w| = |\pi| - 1$ (with $|\pi| = \omega = \omega - 1$ if $\pi \in \Sigma^\omega$), we say that a play π is *σ-conform to* w iff, for any $i < |\pi|$, there exists some $(\mathbf{q}_i, \mathbf{s}_i, a_i, \mathbf{q}'_i, \mathbf{s}'_i) \in \Gamma^{\mathrm{Sp}}$ s.t. $\sigma(\langle \mathbf{q}_0, \mathbf{s}_0 \rangle \ldots \langle \mathbf{q}_i, \mathbf{s}_i \rangle)(a_i, \mathbf{q}'_i, \mathbf{s}'_i) = \langle \mathbf{q}_{i+1}, \mathbf{s}_{i+1} \rangle$. Intuitively, σ-conform plays are those plays which originate when Duplicator's strategy is fixed to σ; δ-conform plays, for δ a Spoiler's strategy, are defined similarly. Below, both strategies are fixed, and the resulting, unique play is conform to both.

The game can halt prematurely, for pebbles may get stuck. In this case, the winning condition is as follows: If there exists a *Left* pebble which cannot be moved, then Duplicator wins. Dually, if no *Right* pebble can be moved, then Spoiler wins.

Remark 1. Our winning condition differs from the one in [5] when pebbles get stuck. There, the losing player is always the one who got stuck. If we let Duplicator win when Spoiler is stuck on a universal *Right* pebble, we would obtain a simulation which *does not imply* language containment. (Notice that "simulation implies containment" is proved in [5] under the assumption that pebbles do not get stuck.) Furthermore, the condition in [5] is unnecessarily strong when Duplicator is stuck on a universal *Left* pebble, where letting Spoiler win is too conservative. Our definition generalizes the correct winning condition to multiple pebbles, for which we prove "simulation implies containment" without further assumptions.

In all other cases, we have that all *Left* pebbles can be moved and at least one *Right* pebble can be moved, and the two players build an infinite sequence of configurations $\pi = \langle \mathbf{q}_0, \mathbf{s}_0 \rangle \langle \mathbf{q}_1, \mathbf{s}_1 \rangle \cdots \in P_1^\omega$. The winning condition is defined in terms of a predicate $C(\pi)$ on π. Different choices of $C(\pi)$ lead to different notions of simulation.

1. *Ordinary* (k_1, k_2)*-simulation*. The acceptance condition is ignored, and Duplicator wins as long as the game doesn't halt: $C(\pi) :\Longleftrightarrow \mathsf{true}$.
2. *Existential direct* (k_1, k_2)*-simulation*. Duplicator wins if, whenever *every* $q \in \mathbf{q}_i$ is accepting, then *some* $s \in \mathbf{s}_i$ is accepting:
$$C(\pi) :\Longleftrightarrow (\forall i.\ \mathbf{q}_i \subseteq F \implies \mathbf{s}_i \cap F \neq \emptyset)\ .$$
3. *Universal direct* (k_1, k_2)*-simulation*. Duplicator wins if, whenever *some* $q \in \mathbf{q}_i$ is accepting, then *every* $s \in \mathbf{s}_i$ is accepting:
$$C(\pi) :\Longleftrightarrow (\forall i.\ \mathbf{q}_i \cap F \neq \emptyset \implies \mathbf{s}_i \subseteq F)\ .$$

As we will see, ordinary simulation is used for ATSs, while existential and universal direct simulation are used for automata over finite and infinite words, respectively.

The winning condition for delayed and fair simulation requires some technical preparation, which consists in the notion of being existentially/universally good since some previous round. Given the current round m, we say that a state $q \in \mathbf{q}_m$ *has seen* a state $\hat{q}$ since some previous round $i \leq m$, written $\mathrm{has\_seen}^i_m(q, \hat{q})$, iff either 1) $q = \hat{q}$, or $i < m$ and there exists $q' \in \mathbf{q}_{m-1}$ s.t. 2.1) $q \in \Delta(q', a_{m-1})$, and 2.2) $\mathrm{has\_seen}^i_{m-1}(q', \hat{q})$. Dually, we write $\mathrm{cant\_avoid}^i_m(q, \hat{q})$ iff either 1) $q = \hat{q}$, or $i < m$ and, for all $q' \in \mathbf{q}_{m-1}$, $q \in \Delta(q', a_{m-1})$ implies $\mathrm{cant\_avoid}^i_{m-1}(q', \hat{q})$. We overload the notation on the set of accepting states, and we write $\mathrm{has\_seen}^i_m(q, F)$ to mean that q has seen some $\hat{q} \in F$; and similarly for $\mathrm{cant\_avoid}^i_m(q, F)$. Finally, we say that $\mathbf{s}_j$ is *existentially good since round* $i \leq j$, written $\mathrm{good}^\exists(\mathbf{s}_j, i)$, if at round j every state in $\mathbf{s}_j$ has seen an accepting state since round i, and j is the least round for which this holds [3]. Similarly, we say that $\mathbf{q}_j$ is *universally good since round* $i \leq j$, written $\mathrm{good}^\forall(\mathbf{s}_j, i)$, if at round j every state in $\mathbf{q}_j$ cannot avoid an accepting state since round i, and j is the least round for which this holds. Formally,

$$\begin{aligned}
\mathrm{good}^\exists(\mathbf{s}_j, i) \iff\ & (\forall s \in \mathbf{s}_j.\ \mathrm{has_seen}^i_j(s, F)) \quad \wedge \\
& \forall j'.\ (\forall s' \in \mathbf{s}_{j'}.\ \mathrm{has_seen}^i_{j'}(s', F)) \implies j' \geq j \\
\mathrm{good}^\forall(\mathbf{s}_j, i) \iff\ & (\forall s \in \mathbf{s}_j.\ \mathrm{cant_avoid}^i_j(s, F)) \quad \wedge \\
& \forall j'.\ (\forall s' \in \mathbf{s}_{j'}.\ \mathrm{cant_avoid}^i_{j'}(s', F)) \implies j' \geq j
\end{aligned}$$

We write $\mathrm{good}^\exists(\mathbf{s}_j)$, omitting the second argument, when we just say that $\mathbf{s}_j$ is good since *some* previous round. For a path $\pi = \mathbf{s}_0\mathbf{s}_1\ldots$, we write $\mathrm{good}^\exists(\pi, \infty)$, with the second argument instantiated to $i = \infty$, to mean that $\mathrm{good}^\exists(\mathbf{s}_j)$ holds for infinitely many j's; and similarly for $\mathrm{good}^\forall(\mathbf{s}_j)$ and $\mathrm{good}^\forall(\pi, \infty)$.

We are now ready to define delayed and fair simulations.

4. *Delayed* (k_1, k_2)*-simulation*. Duplicator wins if, whenever $\mathbf{q}_i$ is universally good, then there exists $j \geq i$ s.t. $\mathbf{s}_j$ is existentially good since round i:

$$C(\pi) :\iff \forall i.\ \mathrm{good}^\forall(\mathbf{q}_i) \implies \exists j \geq i.\ \mathrm{good}^\exists(\mathbf{s}_j, i)\ .$$

5. *Fair* (k_1, k_2)*-simulation*. Duplicator wins if, whenever there are infinitely many i's s.t. $\mathbf{q}_i$ is universally good, then, for any such i, there exists $j \geq i$ s.t. $\mathbf{s}_j$ is existentially good since round i:

$$C(\pi) :\iff \mathrm{good}^\forall(\pi_0, \infty) \implies (\forall i.\ \mathrm{good}^\forall(\mathbf{q}_i) \implies \exists j \geq i.\ \mathrm{good}^\exists(\mathbf{s}_j, i))\ ,$$

where $\pi_0 = \mathbf{q}_0\mathbf{q}_1\ldots$ is the projection of π onto its first component.

We will denote the previous acceptance conditions with $x \in \{\mathrm{o}, \exists\mathrm{di}, \forall\mathrm{di}, \mathrm{de}, \mathrm{f}\}$, and the corresponding game is denoted as $\mathbb{G}^x_{(k_1,k_2)}(\mathbf{q_0}, \mathbf{s_0})$.

Remark 2. Notice that the condition for fair simulation is equivalent to the following simpler one: If $\mathbf{q}_i$ is universally good since some previous round infinitely often, then $\mathbf{s}_i$ is existentially good since some previous round infinitely often: $C'(\pi) :\iff \mathrm{good}^\forall(\pi_0, \infty) \implies \mathrm{good}^\exists(\pi_1, \infty)$, where $\pi_1 = \mathbf{s}_0\mathbf{s}_1\ldots$ is the projection of π onto its second component.

We are now ready to define the simulation relation $\sqsubseteq^x_{(k_1,k_2)}$, with x as above. We say that a k_2-set $\mathbf{s}$ x-simulates a k_1-set $\mathbf{q}$, written $\mathbf{q} \sqsubseteq^x_{(k_1,k_2)} \mathbf{s}$, if Duplicator has a winning strategy in $\mathbb{G}^x_{(k_1,k_2)}(\mathbf{q},\mathbf{s})$. We overload the simulation relation $\sqsubseteq^x_{(k_1,k_2)}$ on singletons: $q \sqsubseteq^x_{(k_1,k_2)} s \iff \{q\} \sqsubseteq^x_{(k_1,k_2)} \{s\}$. For two automata $\mathcal{A}$ and $\mathcal{B}$, we write $\mathcal{A} \sqsubseteq^x_{(k_1,k_2)} \mathcal{B}$ for $q_I^{\mathcal{A}} \sqsubseteq^x_{(k_1,k_2)} q_I^{\mathcal{B}}$, where the simulation is actually computed on the disjoint union of $\mathcal{A}$ and $\mathcal{B}$. If $\sqsubseteq^x_{(k_1,k_2)}$ is a simulation, then its transitive closure is defined as $\preceq^x_{(k_1,k_2)}$. Note that, in general, $\sqsubseteq^x_{(k_1,k_2)}$ is not itself a transitive relation.

Multipebble simulations hierarchy. In general, having more pebbles (possibly) gives more power to the Duplicator. This is similar to the $(1,k)$-simulations for NBA studied in [3], but in our context there are two independent directions of "growing power".

Theorem 1. *Let $x \in \{\mathrm{o}, \exists\mathrm{di}, \forall\mathrm{di}, \mathrm{de}, \mathrm{f}\}$ and $k_1' \geq k_1$, $k_2' \geq k_2$.*

1. *Inclusion: $\sqsubseteq^x_{(k_1,k_2)} \subseteq \sqsubseteq^x_{(k_1',k_2')}$. (In particular, $\preceq^x_{(k_1,k_2)} \subseteq \preceq^x_{(k_1',k_2')}$.)*
2. *Strictness: If $k_1' > k_1$ or $k_2' > k_2$, there exists an automaton $\mathcal{Q}'$ s.t. $\sqsubseteq^x_{(k_1,k_2)} \neq \sqsubseteq^x_{(k_1',k_2')}$.*

Proof (Sketch). Point 1) follows directly from the definitions, since Duplicator can always take pebbles away. Point 2) is illustrated in Figure 1, which holds for any kind of simulation $x \in \{\mathrm{o}, \exists\mathrm{di}, \forall\mathrm{di}, \mathrm{de}, \mathrm{f}\}$. □

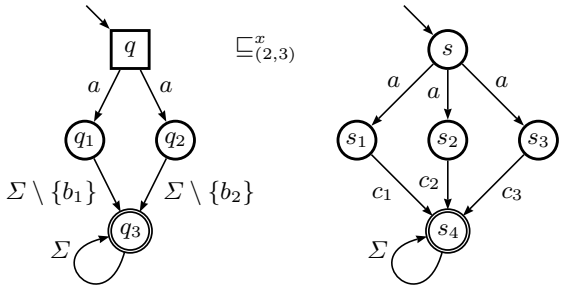


Fig. 1. Example in which $q \sqsubseteq^x_{(2,3)} s$, but $q \not\sqsubseteq^x_{(k_1,k_2)} s$ for any $k_1 \leq 2, k_2 \leq 3$, with $k_1 < 2$ or $k_2 < 3$. The alphabet is $\Sigma' = \{a\} \cup \Sigma$, with $\Sigma = \{b_1, b_2, c_1, c_2, c_3\}$. Note that both automata recognize the same language, both over finite and infinite words: $\mathcal{L}^{\mathrm{fin}}(q) = \mathcal{L}^{\mathrm{fin}}(s) = a(c_1 + c_2 + c_3)\Sigma^*$ and $\mathcal{L}^{\omega}(q) = \mathcal{L}^{\omega}(s) = a(c_1 + c_2 + c_3)\Sigma^{\omega}$.

Theorem 2. *For any $k_1, k_2 \in \mathbb{N}_{>0}$ and any automaton $\mathcal{Q}$,*

1. $\sqsubseteq^{\exists\mathrm{di}}_{(k_1,k_2)} \subseteq \sqsubseteq^{\mathrm{o}}_{(k_1,k_2)}$
2. $\sqsubseteq^{\forall\mathrm{di}}_{(k_1,k_2)} \subseteq \sqsubseteq^{\mathrm{de}}_{(k_1,k_2)} \subseteq \sqsubseteq^{\mathrm{f}}_{(k_1,k_2)} \subseteq \sqsubseteq^{\mathrm{o}}_{(k_1,k_2)}$.

Moreover, for each containment, there exists $\mathcal{Q}$ s.t. the containment is strict.

Proof. The containments follow directly from the definitions. For the strictness, consider again the example in Figure 1, with the modifications below. If no state on the right is accepting, then no simulation holds except ordinary simulation. If q is accepting, then universal direct simulation does not hold, but delayed simulation does. Finally, if the only accepting state is q, then delayed simulation does not hold, but fair simulation does. Is is easy to generalize this example for any $k_1, k_2 \in \mathbb{N}_{>0}$. □

3 Finite words

Lemma 1. *For any automaton $\mathcal{Q}$ with n states and states $q, s \in Q$:*

1. *$q \sqsubseteq^{\exists\mathrm{di}}_{(k_1,k_2)} s$ implies $\mathcal{L}^{\mathrm{fin}}(q) \subseteq \mathcal{L}^{\mathrm{fin}}(s)$, for any $k_1, k_2 \in \mathbb{N}_{>0}$.*
2. *$q \sqsubseteq^{\mathrm{o}}_{(k_1,k_2)} s$ implies $\mathcal{T}r(q) \subseteq \mathcal{T}r(s)$, for any $k_1, k_2 \in \mathbb{N}_{>0}$.*
3. *$\mathcal{L}^{\mathrm{fin}}(q) \subseteq \mathcal{L}^{\mathrm{fin}}(s)$ implies $q \sqsubseteq^{\exists\mathrm{di}}_{(n,n)} s$, provided that $\mathcal{Q}$ is complete.*
4. *$\mathcal{T}r(q) \subseteq \mathcal{T}r(s)$ implies $q \sqsubseteq^{\mathrm{o}}_{(n,n)} s$, provided that $\mathcal{Q}$ is complete.*

In particular, the last two points above show that existential-direct (resp., ordinary) simulation "reaches" language inclusion (resp., trace inclusion) at (n, n).

Subset constructions. The subset construction is a well-known procedure for determinizing NFAs [8]. It is not difficult to generalize it over *alternating* automata, where it can be used for eliminating existential states, i.e., to perform the *de-existentialization* of the automaton. The idea is the same as in the subset construction, except that, when considering a-successors of a macrostate (for a symbol $a \in \Sigma$), existential and universal states are treated differently. For existential states, we apply the same procedure as in the classic subset construction, by taking always all a-successors. For universal states, each a-successor induces a different transition in the subset automaton. This ensures that macrostates can be interpreted purely disjunctively, and the language of a macrostate equals the union over the language of the states belonging to it. Accordingly, a macrostate is accepting if it contains *some* state which is accepting.

The previous construction can be dualized for de-universalizing finite automata. For an AFA $\mathcal{Q}$, let $\mathcal{S}^{\exists}(\mathcal{Q})$ and $\mathcal{S}^{\forall}(\mathcal{Q})$ be its de-existentialization and de-universalization, respectively. (See Definitions 1 and 2 in Appendix B.1 of the technical report [2].)

The following lemma formalizes the intuition that multipebble simulations for AFA in fact correspond to $(1, 1)$-simulations over the appropriate subset-constructions.

Lemma 2. *Let $\mathcal{Q}_1, \mathcal{Q}_2$ be two AFAs over the same alphabet Σ, with $|Q_1| = n_1$ and $|Q_2| = n_2$. Then, for any $k_1 \leq n_1$ and $k_2 \leq n_2$,*

$$\mathcal{Q}_1 \sqsubseteq^{\exists\mathrm{di}}_{(k_1,n_2)} \mathcal{Q}_2 \iff \mathcal{Q}_1 \sqsubseteq^{\exists\mathrm{di}}_{(k_1,1)} \mathcal{S}^{\exists}(\mathcal{Q}_2) \tag{1}$$

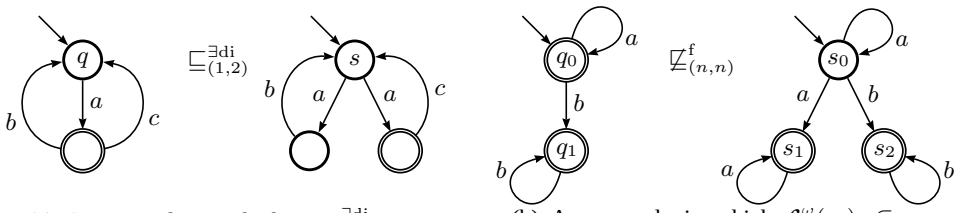
$$\mathcal{Q}_1 \sqsubseteq^{\exists\mathrm{di}}_{(n_1,k_2)} \mathcal{Q}_2 \iff \mathcal{S}^{\forall}(\mathcal{Q}_1) \sqsubseteq^{\exists\mathrm{di}}_{(1,k_2)} \mathcal{Q}_2 \tag{2}$$

$$\mathcal{Q}_1 \sqsubseteq^{\exists\mathrm{di}}_{(n_1,n_2)} \mathcal{Q}_2 \iff \mathcal{S}^{\forall}(\mathcal{Q}_1) \sqsubseteq^{\exists\mathrm{di}}_{(1,1)} \mathcal{S}^{\exists}(\mathcal{Q}_2)\,. \tag{3}$$

4 Infinite words

Multipebble existential-direct simulation is not suitable for being used for ω-automata, since it does not even imply ω-language inclusion.

Theorem 3. *For any $k_1, k_2 \in \mathbb{N}_{>0}$, not both equal to 1, there exist an automaton $\mathcal{Q}$ and states $q, s \in \mathcal{Q}$ s.t. $q \sqsubseteq^{\exists di}_{(k_1,k_2)} s$ holds, but $\mathcal{L}^{\omega}(q) \not\subseteq \mathcal{L}^{\omega}(s)$.*



(a) An example in which $q \sqsubseteq^{\exists\mathrm{di}}_{(1,2)} s$ holds, but $\mathcal{L}^\omega(q) \not\subseteq \mathcal{L}^\omega(s)$.

(b) An example in which $\mathcal{L}^\omega(q_0) \subseteq \mathcal{L}^\omega(s_0)$ holds, but $q_0 \not\sqsubseteq^{\mathrm{f}}_{(n,n)} s_0$.

Fig. 2. Two examples.

Proof. Consider the example in Figure 2(a). Clearly, $q \sqsubseteq^{\exists\mathrm{di}}_{(1,2)} s$ holds, since Duplicator can split pebbles on the successors of s, and one such pebble is accepting, as required by existential-direct simulation. But $\mathcal{L}^\omega(q) \not\subseteq \mathcal{L}^\omega(s)$: In fact, $(ab)^\omega \in \mathcal{L}^\omega(q) = (a(b+c))^\omega$, but $(ab)^\omega \notin \mathcal{L}^\omega(s) = ((ab)^*ac)^\omega$. □

This motivates the definition of *universal*-direct simulation, which *does imply* ω-*language inclusion*, like the coarser delayed and fair simulations.

Theorem 4. *For* $x \in \{\forall\mathrm{di}, \mathrm{de}, \mathrm{f}\}$*, automaton* $\mathcal{Q}$*,* $k_1, k_2 \in \mathbb{N}_{>0}$ *and states* $q, s \in \mathcal{Q}$*,*

$$q \sqsubseteq^{x}_{(k_1,k_2)} s \quad \textit{implies} \quad \mathcal{L}^\omega(q) \subseteq \mathcal{L}^\omega(s)\ .$$

Unlike in the finite word case, ω-language inclusion is not "reached" by the simulations $\{\forall\mathrm{di}, \mathrm{de}, \mathrm{f}\}$. See Figure 2(b) and Appendix C in the technical report [2].

Theorem 5. *For any* $x \in \{\forall\mathrm{di}, \mathrm{de}, \mathrm{f}\}$*, there exist an automaton* $\mathcal{Q}$ *and states* $q_0, s_0 \in \mathcal{Q}$ *s.t.* $\mathcal{L}^\omega(q_0) \subseteq \mathcal{L}^\omega(s_0)$*, but* $q_0 \not\sqsubseteq^{x}_{(n,n)} s_0$*.*

The Miyano-Hayashi construction The Miyano-Hayashi (MH) construction [10] is a subset-like construction for ABAs which removes universal non-determinism, i.e., it performs the *de-universalization* of ω-automata. The idea is similar to the analogous construction over finite words, with extra bookkeeping needed for recording visits to accepting states, which may occur not simultaneously for different runs. A set of obligations is maintained, encoding the requirement that, independently of how universal non-determinism is resolved, an accepting state has to be eventually reached. There is a tight relationship between these obligations and fair multipebble simulation. For an ABA $\mathcal{Q}$, let $\mathcal{Q}_{\mathrm{nd}}$ be the de-universalized automaton obtained by applying the MH-construction. (See also Definition 3 in Appendix C.1 of the technical report [2].)

The following lemma says that the MH-construction produces an automaton which is $(n,1)$-fair-simulation equivalent to the original one, and this result is "tight" in the sense that it does not hold for either direct, or delayed simulation.

Lemma 3. *For any ABA* $\mathcal{Q}$*, let* $\mathcal{Q}_{nd}$ *be the NBA obtained according to the Miyano-Hayashi de-universalization procedure applied to* $\mathcal{Q}$*. Then,*

a) $\mathcal{Q} \sqsubseteq^{x}_{(n,1)} \mathcal{Q}_{nd}$*, for* $x \in \{\mathrm{f}, \forall\mathrm{di}\}$*, and a')* $\exists$ *automaton* $\mathcal{Q}^1$ *s.t.* $\mathcal{Q}^1 \not\sqsubseteq^{\mathrm{de}}_{(n,1)} \mathcal{Q}^1_{nd}$*,*
b) $\mathcal{Q}_{nd} \sqsubseteq^{\mathrm{f}}_{(1,1)} \mathcal{Q}$*, and b')* $\exists$ *automaton* $\mathcal{Q}^2$ *s.t.* $\mathcal{Q}^2_{nd} \not\sqsubseteq^{x}_{(1,1)} \mathcal{Q}^2$*, for* $x \in \{\mathrm{de}, \forall\mathrm{di}\}$*.*

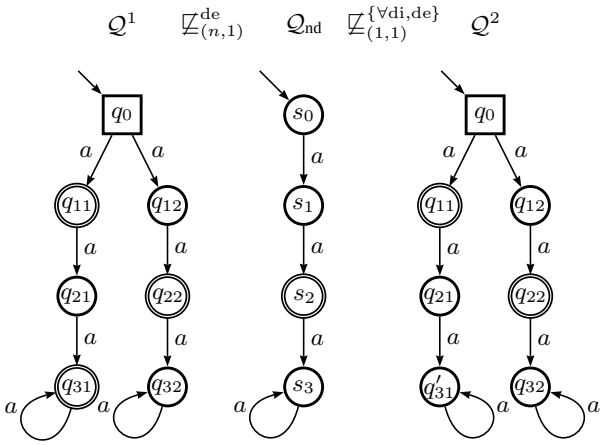


Fig. 3. An example showing automata $\mathcal{Q}^1$ and $\mathcal{Q}^2$ s.t. $\mathcal{Q}^1 \not\sqsubseteq^{\mathrm{de}}_{(n,1)} \mathcal{Q}_{\mathrm{nd}}$ ($n = 2$ suffices), and $\mathcal{Q}_{\mathrm{nd}} \not\sqsubseteq^{x}_{(1,1)} \mathcal{Q}^2$ for $x \in \{\forall\mathrm{di}, \mathrm{de}\}$. The only difference between $\mathcal{Q}^1$ and $\mathcal{Q}^2$ is the state q_{31} being accepting in the former and q'_{31} being non-accepting in the latter. Notice that $\mathcal{Q}^1_{\mathrm{nd}} = \mathcal{Q}^2_{\mathrm{nd}} = \mathcal{Q}_{\mathrm{nd}}$. The states in $\mathcal{Q}_{\mathrm{nd}}$ are: $s_0 = (\{q_0\}, \{q_0\})$, $s_1 = (\{q_{11}, q_{12}\}, \{q_{12}\})$, $s_2 = (\{q_{21}, q_{22}\}, \emptyset)$, $s_3 = (\{q_{31}, q_{32}\}, \{q_{32}\})$.

Since fair simulation implies language inclusion, $\mathcal{Q}$ and $\mathcal{Q}_{\mathrm{nd}}$ have the same language. This constitutes an alternative proof of correctness for the MH-construction.

The MH-construction "preserves" fair simulation in the following sense.

Lemma 4. *Let $\mathcal{Q}, \mathcal{S}$ be two ABAs. Then, $\mathcal{Q} \sqsubseteq^{\mathrm{f}}_{(n,1)} \mathcal{S} \iff \mathcal{Q}_{nd} \sqsubseteq^{\mathrm{f}}_{(1,1)} \mathcal{S}_{nd}$.*

Remark 3. A weaker version of the "only if" direction of Lemma 4 above, namely $\mathcal{Q} \sqsubseteq^{\mathrm{f}}_{(1,1)} \mathcal{S} \implies \mathcal{Q}_{\mathrm{nd}} \sqsubseteq^{\mathrm{f}}_{(1,1)} \mathcal{S}_{\mathrm{nd}}$ (notice the $(1,1)$ in the premise), had already appeared in [5]. The same statement for both direct and delayed simulation is false, unlike as incorrectly claimed in [5]. In fact, it can be shown (with an example similar to Figure 3) that there exist automata $\mathcal{Q}$ and $\mathcal{S}$ s.t. $\mathcal{Q} \sqsubseteq^{x}_{(1,1)} \mathcal{S}$, but $\mathcal{Q}_{\mathrm{nd}} \not\sqsubseteq^{x}_{(1,1)} \mathcal{S}_{\mathrm{nd}}$, with $x \in \{\mathrm{di}, \mathrm{de}\}$. Finally, the "if" direction of Lemma 4 can only be established in the context of multiple pebbles, and it is new.

Transitivity. While most (k_1, k_2)-simulations are not transitive, some limit cases are. By defining a notion of join for $(1,n)$- and $(n,1)$-strategies (see Appendix C.2 in the technical report [2]), we establish that $(1,n)$ and $(n,1)$ simulations are transitive.

Theorem 6. *Let $\mathcal{Q}$ be an ABA with n states, and let $x \in \{\forall\mathrm{di}, \mathrm{de}, \mathrm{f}\}$. Then, $\sqsubseteq^{x}_{(1,n)}$ and $\sqsubseteq^{x}_{(n,1)}$ are transitive.*

Remark 4 (Difficulties for (n,n) transitivity.). We did consider transitivity for (n,n)-simulations on ABA, but found two major issues there. The first issue concerns directly the definition of the join of two (n,n)-strategies, and this holds for any $x \in \{\forall\mathrm{di}, \mathrm{de}, \mathrm{f}\}$: The so-called "puppeteering technique", currently used for defining the join for $(1,n)$- and $(n,1)$-strategies, requires to maintain several games, and to pipe the output from one game to the input of one or more other games. This creates a notion of dependency

between different games. For $(1, n)$ and $(n, 1)$, there are no cyclic dependencies, and we were able to define the joint strategy. However, for (n, n)-simulations, there are cyclic dependencies, and it is not clear how the joint strategy should be defined.

The second issue arises from the fact that we further require that the join of two winning strategies is itself a winning strategy. Therefore, the joint strategy needs to carry an invariant which implies the x-winning condition, for $x \in \{\forall\mathrm{di}, \mathrm{de}, \mathrm{f}\}$. While such an invariant for $x = \forall\mathrm{di}$ is straightforward, it is not clear what the correct invariant should be for either delayed or fair simulation.

5 Quotienting

In the following we discuss how multipebble simulation preorders can be used for state-space reduction of alternating automata, i.e., we discuss under which notions of quotient the quotient automaton recognizes the same language as the original one.

Let $\mathcal{Q} = (Q, \Sigma, q_I, \Delta, E, U, F)$ be an alternating automaton, over finite or infinite words. Let $\preceq$ be any binary relation on Q, and let $\approx$ be the induced equivalence, defined as $\approx = \preceq^* \cap (\preceq^*)^{-1}$. $[\cdot] : Q \mapsto [Q]$ is the function that maps each element $q \in Q$ to the equivalence class $[q] \in [Q]$ it belongs to, i.e., $[q] := \{q' \mid q \approx q'\}$. We overload $[P]$ on sets $P \subseteq Q$ by taking the set of equivalence classes.

In all the notions of quotients that will be defined, only the transition relation varies. Thus, we gather the common part under a quotient skeleton. We define the *quotient skeleton* $\mathcal{Q}_\approx = ([Q], \Sigma, [q_I], \Delta_\approx, E', U', F')$ as follows: $E' := [E]$, $U' := [Q] \setminus E' = \{\ [q] \mid [q] \subseteq U\ \}$ and $F' = [F]$. We leave $\Delta_\approx$ unspecified at this time, as it will have different concrete instantiations later. Notice that mixed classes, i.e., classes containing both existential and universal states, are declared existential.

The following definitions are borrowed from [5]. We say that $q' \in \Delta(q, a)$ is a k-*x-minimal a-successor of* q iff there there is no strictly $\sqsubseteq^x_{(1,k)}$-smaller a-successor of q, i.e., for any $q'' \in \Delta(q, a)$, $q'' \sqsubseteq^x_{(1,k)} q'$ implies $q' \sqsubseteq^x_{(1,k)} q''$. Similarly, $q' \in \Delta(q, a)$ is a k-x-*maximal a-successor of* q iff for any $q'' \in \Delta(q, a)$, $q' \sqsubseteq^x_{(1,k)} q''$ implies $q'' \sqsubseteq^x_{(1,k)} q'$. Let $\min^{k,x}_a(q)/\max^{k,x}_a(q)$ be the set of minimal/maximal successors.

5.1 Finite words

Let $\preceq$ be *any* preorder which implies language inclusion over finite words, i.e., $q \preceq s \implies \mathcal{L}^{\mathrm{fin}}(q) \subseteq \mathcal{L}^{\mathrm{fin}}(s)$. In particular, one can take $\preceq = (\sqsubseteq^{\exists\mathrm{di}}_{(k_1,k_2)})^*$, or even $\preceq$ equal to language inclusion itself. As before, let $\approx$ be the equivalence induced by $\preceq$. It is well known that automata over finite words can be quotiented w.r.t. any preorder which implies language equivalence. Here, we show that not all transitions are needed, and that is is sufficient to consider $\preceq$-maximal successors of existential states and $\preceq$-minimal successors of universal states. We define the *minimax* [5] quotient automaton $\mathcal{Q}^{\mathrm{m}}_\approx$ by instantiating the quotient skeleton (see Section 5) with transition relation $\Delta_\approx := \Delta^{\mathrm{m}}_\approx$, where $([q], a, [q']) \in \Delta^{\mathrm{m}}_\approx$ iff either

- $[q] \in E'$ and $\exists\, \widehat{q} \in [q] \cap E, \widehat{q}' \in [q']$ s.t. $(\widehat{q}, a, \widehat{q}') \in \Delta \wedge \widehat{q}' \in \max^{\preceq}_a(\widehat{q})$, or
- $[q] \in U'$ and $\exists\, \widehat{q} \in [q], \widehat{q}' \in [q']$ s.t. $(\widehat{q}, a, \widehat{q}') \in \Delta$ and $\widehat{q}' \in \min^{\preceq}_a(\widehat{q})$.

Notice that transitions from universal states in mixed classes are ignored altogether.

Lemma 5. *Let $\mathcal{Q}$ be any alternating finite automaton, and let $\preceq$ be any preorder which implies finite-language inclusion. Then, for any $q \in Q$, $\mathcal{L}^{\mathrm{fin}}(q) = \mathcal{L}^{\mathrm{fin}}([q]_m)$.*

5.2 Infinite words

Unlike for finite words, it is well known that quotienting ω-automata w.r.t. ω-language-equivalence does not preserve the ω-language. It has even been shown that quotienting w.r.t. $(1,1)$-fair (bi)simulation does not preserve the ω-language either [7, 4]. Therefore, one has to look for finer simulations, like delayed or direct simulation. Notice that multipebble *existential*-direct simulation cannot be used for quotienting, since it does not even imply ω-language inclusion—see Theorem 3.

Theorem 7. *For any $k_1, k_2 \in \mathbb{N}^{>0}$ and $x \in \{\exists\mathrm{di},\ f\}$ there exists an ABA $\mathcal{Q}$ s.t. $\mathcal{L}^\omega(\mathcal{Q}) \neq \mathcal{L}^\omega(\mathcal{Q}_{\approx})$, with $\approx := \approx^x_{(k_1,k_2)}$. For $x = \exists\mathrm{di}$, k_1 and k_2 must not be both equal to 1. (Note that $\approx^{\exists\mathrm{di}}_{(1,1)}$-quotienting* does preserve *the ω-language.)*

Thus, in the following we concentrate on *universal*-direct and delayed simulation.

Minimax quotients for universal-direct simulation. In [5] it has been shown that minimax quotients preserve the ω-language (for direct simulation), and that one can consider just maximal/minimal successors of existential/universal states, respectively. Here, we improve this notion, by showing that, when considering multiple-pebbles, it is not needed to consider *every* maximal successor of existential states, but it is safe to discard those maximal successors which are $(1,k)$-simulated by a k-*set* of other maximal successors. This suggests the following definition: For $\widehat{q} \in E$, $a \in \Sigma$ and $k > 0$, we say that $\widehat{\mathbf{q}}'$ is a set of *k-maximal representatives for a-successors of $\widehat{q}$* iff

$$\widehat{\mathbf{q}}' \subseteq \max_a^{k,\forall\mathrm{di}}(\widehat{q}) \wedge \left(\forall q'' \in \left(\max_a^{k,\forall\mathrm{di}}(\widehat{q}) \setminus \widehat{\mathbf{q}}'\right) . \exists \widehat{\mathbf{q}}'' \in 2^{\widehat{\mathbf{q}}',k} .\ q'' \sqsubseteq^{\forall\mathrm{di}}_{(1,k)} \widehat{\mathbf{q}}''\right) \quad (4)$$

Notice that the above definition is non-deterministic, in the sense that there might be different sets of maximal representatives: In this case, one can just take any $\subseteq$-minimal set satisfying Equation 4. In the following, we assume that a set of maximal representatives $\widehat{\mathbf{q}}'$ has been selected for any $\widehat{q} \in E$ and $a \in \Sigma$.

We define the *minimax+* quotient automaton $\mathcal{Q}^{\mathrm{m}+}_{\approx}$ by instantiating the quotient skeleton (see Section 5) with transition relation $\Delta_{\approx} := \Delta^{\mathrm{m}+}_{\approx}$, which differs from $\Delta^{\mathrm{m}}_{\approx}$ just for existential and mixed classes: $([q], a, [q']) \in \Delta^{\mathrm{m}+}_{\approx}$ with $[q] \in E'$ iff

– there exist $\widehat{q} \in [q] \cap E$ and $\widehat{q}' \in [q']$ s.t. $(\widehat{q}, a, \widehat{q}') \in \Delta$ and $\widehat{q}' \in \widehat{\mathbf{q}}'$, where $\widehat{\mathbf{q}}'$ is a fixed set of k-maximal representatives for a-successors of $\widehat{q}$, as defined above.

Our definition of minimax+ quotient differs from the one in [5] also w.r.t. the treatment of mixed classes, as discussed in the following remarks.

Remark 5. While in [5] universal states in mixed classes do induce transitions (to minimal elements), in our definition we ignore these transitions altogether. In the setting of $(1,1)$-simulations these two definitions coincide, as they are shown in [5] to yield

exactly the same transitions, but this needs not be the case in our setting: In the context of multiple-pebbles, one minimal transition from an universal state q^U might be subsumed by no single transition from some existential state q^E in the same class, but it is always the case that q^E has a set of transitions which *together* subsume the one from q^U (cf. Lemma 15 in Appendix D.3 [2]). In this case, we show that one can in fact always discard the transitions from q^U. Thus, in the context of multiple-pebbles, minimax+ quotients result in less transitions than just minimax quotients from [5].

Remark 6. While minimax mixed classes are deterministic when considering $(1,1)$-simulations [5], this is not necessarily true when multiple pebbles are used.

Theorem 8. $q \approx^{\forall\mathrm{di}}_{(1,n)} [q]_{\mathrm{m}+}$, *where the quotient is taken w.r.t. the transitive closure of* $\sqsubseteq^{\forall\mathrm{di}}_{(1,k)}$, *for any* k *such that* $1 \leq k \leq n$. *In particular,* $\mathcal{L}^\omega(q) = \mathcal{L}^\omega([q]_{\mathrm{m}+})$.

Semielective quotients for delayed simulation. It has been shown in [5] that minimax quotients w.r.t $(1,1)$-delayed simulation on ABA do not preserve the ω-language. The reason is that taking just maximal successors of existential states is incorrect for delayed simulation, since a visit to an accepting state might only occur by performing a non-maximal transition. (This is not the case with direct simulation, where if a simulation-smaller state is accepting, then every bigger state is accepting too.) This motivates the definition of *semielective quotients* [5], which are like minimax quotients, with the only difference that *every* transition induced by existential states is considered, not just maximal ones. Except for that, all previous remarks still apply. In particular, in mixed classes in semielective quotients it is necessary to ignore non-minimal transitions from universal states—the quotient automaton would recognize a bigger language otherwise.

While for the $(1,1)$-simulations on ABA in [5] it is actually possible to ignore transitions from universal states in mixed classes altogether (see Remark 5), in the context of multiple-pebbles this is actually incorrect (see Figure 5 in Appendix D.3 of the technical report [2]). The reason is similar as why non-maximal transitions from existential states cannot be discarded: This might prevent accepting states from being visited. We define the *semielelective+* quotient automaton $\mathcal{Q}^{\mathrm{se}+}_{\approx}$ by instantiating the quotient skeleton (see Section 5) with $\Delta_\approx := \Delta^{\mathrm{se}+}_{\approx}$, where

$$([q], a, [q']) \in \Delta^{\mathrm{se}+}_{\approx} \iff (q, a, q') \in \Delta \text{ and either } q \in E, \text{ or } q \in U \text{ and } q' \in \min{}^{n,\mathrm{de}}_a(q)$$

Theorem 9. $q \approx^{\mathrm{de}}_{(1,n)} [q]_{\mathrm{se}+}$, *where the quotient is taken w.r.t.* $\sqsubseteq^{\mathrm{de}}_{(1,n)}$. *In particular,* $\mathcal{L}^\omega(q) = \mathcal{L}^\omega([q]_{\mathrm{se}+})$.

Remark 7. It is surprising that, unlike for NBA [3], quotienting ABA w.r.t. $(1,k)$-de simulations, for $1 < k < n$, does not preserve the language of the automaton in general. The problem is again in the mixed classes, where minimal transitions from universal states can be selected only by looking at the full $(1,n)$-simulation. See the counterexample in Figure 4, where the dashed transition is present in the $(1,k)$-quotient, despite being non-$(1,n)$-minimal.

Remark 8. Semielective multipebble quotients can achieve arbitrarily high compression ratios relative to semielective 1-pebble quotients, (multipebble-)direct minimax quotients and mediated preorder quotients [1] (see Figure 6 in Appendix D.3 [2]).

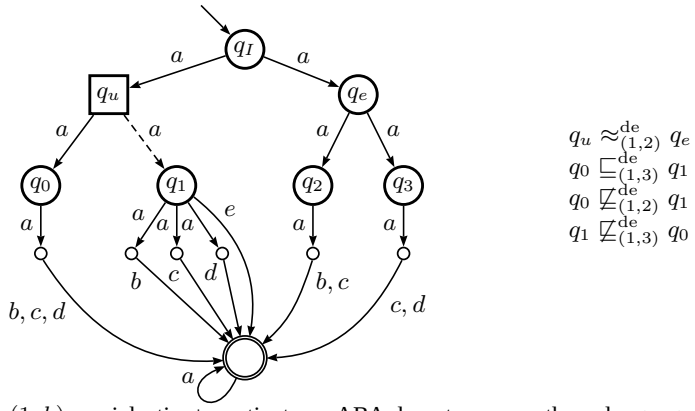


Fig. 4. $(1, k)$-semielective+ quotients on ABA do not preserve the ω-language for $1 < k < n$ in general. Let $k = 2$. The only two $(1, k)/(1, n)$-equivalent states are q_u and q_e, and in the quotient they form a mixed class. q_1 *is not* a $(1, n)$-minimal a-successor of q_u, but it is a $(1, k)$-minimal successor for $k = 2$. Thus, the only difference between the $(1, n)$- and $(1, k)$-semielective+ quotients is that the dashed transition is (correctly) not included in the former, but (incorrectly) included in the latter. Thus the $(1, k)$-semielective+ quotient automaton would incorrectly accept the word $w = aaea^\omega \notin \mathcal{L}^\omega(q_I) = aaa\{b + c + d\}a^\omega$.

6 Solving Multipebble Simulation Games

In this section we show how to solve the multipebble simulation games previously defined. We encode each simulation game into a 2-player game-graph with an ω-regular winning condition. In the game-graph, Eve will take the rôle of Duplicator, and Adam the one of Spoiler. A game-graph is a tuple $\mathcal{G} = \langle V_{\mathrm{E}}, V_{\mathrm{A}}, \rightarrow \rangle$, where nodes in V_{E} belong to Eve (mimicking Duplicator), and nodes in V_{A} belong to Adam (mimicking Spoiler). Transitions are represented by elements in $\rightarrow \subseteq (V_{\mathrm{E}} \times V_{\mathrm{A}} \cup V_{\mathrm{A}} \times V_{\mathrm{E}})$, where we write $p \rightarrow q$ for $(p, q) \in \rightarrow$. Notice that the two players strictly alternate while playing, i.e., the game graph is *bipartite*. We write V for $V_{\mathrm{E}} \cup V_{\mathrm{A}}$. We introduce the following monotone operator on $2^{V_{\mathrm{A}}}$: For any $\mathbf{x} \subseteq V_{\mathrm{A}}$, $\mathrm{cpre}(\mathbf{x}) := \{v_0 \in V_{\mathrm{A}} \mid \forall v_1 \in V_{\mathrm{E}}.\ (v_0 \rightarrow v_1 \implies \exists v_2 \in \mathbf{x}.\ v_1 \rightarrow v_2)\}$, i.e., $\mathrm{cpre}(\mathbf{x})$ is the set of nodes where Eve can force the game into $\mathbf{x}$.

We define various game-graphs for solving simulations. We express the winning region of Eve as a μ-calculus fixpoint expression over V_{A} [9], which can then be evaluated using standard fixpoint algorithms. We derive the desired complexity upper bounds using the following fact:

Lemma 6. *Let e be a fixpoint expression over a graph V, with $|V| \in n^{O(k)}$. Then, for any fixed $k \in \mathbb{N}$, evaluating e can be done in time polynomial in n.*

For solving direct and fair simulation, we refer the reader to Appendix E [2]. Here, we consider just delayed simulation, which is the most difficult (and interesting).

The natural starting point for defining $\mathcal{G}^{\mathrm{de}}$ is the definition in [3] of the game-graph for computing $(1, k)$-simulations for NBAs. Unfortunately, the game-graph in [3] is actually incorrect: According to the definition of delayed simulation (cf. Section 2), every new obligation encountered when the left side is accepting at some round should be *independently* satisfied by the right side, which has to be good since *that* round. Now, the algorithm in [3] just tries to satisfy the most recent obligation, which overrides all the previous ones. This is an issue: If the left side is continuously accepting, for example, then the right side might simply have not enough time to satisfy any obligation at all. Therefore, [3] actually computes an *under-approximation* to delayed simulation.

We overcome this difficulty by explictly bookkeeping all pending constraints. This leads to the following definitions. The game-graph for delayed simulation is $\mathcal{G}^{\mathrm{de}} = \langle V_{\mathrm{E}}^{\mathrm{de}}, V_{\mathrm{A}}^{\mathrm{de}}, \rightarrow^{\mathrm{de}} \rangle$, where nodes in $V_{\mathrm{A}}^{\mathrm{de}}$ are of the form $v_{(\mathbf{q},\mathsf{Bad},\mathbf{s},\mathsf{Good})}$, and nodes in $V_{\mathrm{E}}^{\mathrm{de}}$ of the form $v_{(\mathbf{q},\mathsf{Bad},\mathbf{s},\mathsf{Good},a,\mathbf{q}',\mathbf{s}')}$, with $\mathbf{q},\mathbf{q}',\mathbf{s},\mathbf{s}' \subseteq Q$. $\mathsf{Bad} = \langle \mathbf{b}_1 \supset \cdots \supset \mathbf{b}_{m_1} \rangle$ and $\mathsf{Good} = \langle \mathbf{g}_1 \subset \cdots \subset \mathbf{g}_{m_2} \rangle$ are two *sequences* of sets of states from Q, strictly ordered by set-inclusion, which are used to keep track of multiple obligations.

Intuitively, Bad is used to detect when new constraints should be created, i.e., to detect when every *Left* pebble is universally good since some previous round. At each round, a new set of bad pebbles $\mathbf{b} = \mathbf{q} \setminus F$ is added to Bad. When accepting states are visited by *Left* pebbles, they are discarded from every set $\mathbf{b} \in \mathsf{Bad}$. When some $\mathbf{b}$ becomes eventually empty, this means that, at the current round, all *Left* pebbles are universally good since some previous round: At this point, $\mathbf{b}$ is removed from Bad, and we say that *the red light flashes*.

The sequence Good represents a set of constraints to be eventually satisfied. Each $\mathbf{g} \in \mathsf{Good}$ is a set of good pebbles, which we require to "grow" until it becomes equal to $\mathbf{s}$. When $\mathsf{Good} = \emptyset$, there is no pending constraint. Constraints are added to Good when the red light flashes (see above): In this case, we update Good by adding the new empty constraint $\mathbf{g} = \emptyset$. When accepting states are visited by *Right* pebbles, we upgrade every constraint $\mathbf{g} \in \mathsf{Good}$ by adding accepting states. Completed constraints $\mathbf{g} = \mathbf{s}$ are then removed from Good, and we say that *the green light flashes*.

Lemma 7. $\left|V^{\mathrm{de}}\right| \leq 2 \cdot (n+1)^{2(k_1+k_2)} \cdot \left(1 + (k_1+1)^{k_1+1}\right) \cdot \left(1 + 2(k_2+1)^{k_2+1}\right) \cdot |\Sigma|$.

Transitions in $\mathcal{G}^{\mathrm{de}}$ are defined as follows. For any $(\mathbf{q},\mathbf{s},a,\mathbf{q}'',\mathbf{s}'') \in \Gamma^{\mathrm{Sp}}$, we have $v_{(\mathbf{q},\mathsf{Bad},\mathbf{s},\mathsf{Good})} \rightarrow^{\mathrm{de}} v_{(\mathbf{q},\mathsf{Bad},\mathbf{s},\mathsf{Good},a,\mathbf{q}'',\mathbf{s}'')}$, and for $(\mathbf{q},\mathbf{s},a,\mathbf{q}'',\mathbf{s}'',\mathbf{q}',\mathbf{s}') \in \Gamma^{\mathrm{Dup}}$, we have $v_{(\mathbf{q},\mathsf{Bad},\mathbf{s},\mathsf{Good},a,\mathbf{q}'',\mathbf{s}'')} \rightarrow^{\mathrm{de}} v_{(\mathbf{q}',\mathsf{Bad}',\mathbf{s}',\mathsf{Good}')}$, where $\mathsf{Bad}', \mathsf{Good}'$ are computed according to Algorithm 1 in Appendix E.3 of the technical report [2].

We have that Eve wins iff every red flash is matched by at least one green flash, and different red flashes are matched by different green ones. This can be checked by verifying that infinitely often either $\mathsf{Good} = \emptyset$ or $\mathbf{s} \in \mathsf{Good}$, i.e., it is not the case that Good contains a constraint that it is not eventually "completed" and discarded. Let $T = \{v_{(\mathbf{q},\mathsf{Bad},\mathbf{s},\mathsf{Good})} \mid \mathsf{Good} = \emptyset \vee \mathbf{s} \in \mathsf{Good}\}$, and define the initial configuration as

$$v_I = \begin{cases} v_{(\mathbf{q},\{\mathbf{q}\setminus F\},\mathbf{s},\emptyset)} & \text{if } \mathbf{q} \setminus F \neq \emptyset \\ v_{(\mathbf{q},\emptyset,\mathbf{s},\{\mathbf{s}\cap F\})} & \text{otherwise} \end{cases}$$

$\mathbf{q} \sqsubseteq^{\mathrm{de}}_{k_1,k_2} \mathbf{s}$ iff T is visited infinitely often iff $v_I \in W^{\mathrm{de}} = \nu\mathbf{x}\mu\mathbf{y}\,(\mathrm{cpre}(\mathbf{y}) \cup T \cap \mathrm{cpre}(\mathbf{x}))$.

Theorem 10. *For any fixed $k_1, k_2 \in \mathbb{N}$, $x \in \{\forall\mathrm{di}, \exists\mathrm{di}, \mathrm{de}, \mathrm{f}\}$ and sets $\mathbf{q},\mathbf{s} \subseteq Q$, deciding whether $\mathbf{q} \sqsubseteq^{x}_{(k_1,k_2)} \mathbf{s}$ can be done in polynomial time.*

7 Conclusions and Future Work

Transitivity for (n,n)-simulations. As discussed at the end of Section 4, composing (n,n) (winning) strategies is apparently much more difficult than in the $(1,n)$ and $(n,1)$ case. We conjecture that all types of (n,n)-simulations discussed in this paper are transitive, and showing this would conceivably solve the join problem as well.

Quotienting with $(n,1)$- and (n,n)-simulations. While we have dealt with $(1,n)$-quotients, we have not considered $(n,1)$- or (n,n)-quotients. For the latter, one should first solve the associated transitivity problem, and, for both, an appropriate notion of semielective-quotient has to be provided. We have shown that this is already a non-trivial task for $(1,n)$-simulations on ABA.

Future directions. Our work on delayed simulation has shown that several generalizations are possible. In particular, two issues need to be addressed. The first is the complexity of the structure of the game-graph needed for computing delayed simulation. A possible generalization of delayed simulation involving looser "synchronization requirements" between obligations and their satisfaction might result in simpler game-graphs. The second issue concerns Lemmas 3 and 4: We would like to find a weaker delayed-like simulation for which the counterexample shown there does not hold. This would give a better understanding of the MH-construction.

As in [4], it is still open to find a hierarchy of (k_1,k_2)-multipebble simulations converging to ω-language inclusion when $k_1 = k_2 = n$.

Acknowledgment. We thank K. Etessami and C. Fritz for helpful discussions, and an anonymous referee for suggesting the comparison to mediated preorder [1].

References

1. P. A. Abdulla, Y.-F. Chen, L. Holik, and T. Vojnar. Mediating for reduction (on minimizing alternating Büchi automata). In *FSTTCS 2009*, volume 4 of *LIPIcs*, pages 1–12, Dagstuhl, Germany.
2. L. Clemente and R. Mayr. Multipebble simulations for alternating automata. Technical Report EDI-INF-RR-1376, University of Edinburgh, School of Informatics, 2010. Available at http://www.inf.ed.ac.uk/publications/report/1376.html.
3. K. Etessami. A hierarchy of polynomial-time computable simulations for automata. In *CONCUR '02: Proceedings of the 13th International Conference on Concurrency Theory*, pages 131–144, London, UK, 2002. Springer-Verlag.
4. K. Etessami, T. Wilke, and R. A. Schuller. Fair simulation relations, parity games, and state space reduction for Büchi automata. *SIAM J. Comput.*, 34(5):1159–1175, 2005.
5. C. Fritz and T. Wilke. Simulation relations for alternating Büchi automata. *Theor. Comput. Sci.*, 338(1-3):275–314, 2005.
6. Y. Gurevich and L. Harrington. Trees, automata, and games. In *STOC '82: Proceedings of the fourteenth annual ACM symposium on Theory of computing*, pages 60–65, New York, NY, USA, 1982. ACM.
7. T. A. Henzinger and S. Rajamani. Fair bisimulation. In *TACAS '00: Proceedings of the 6th International Conference on Tools and Algorithms for Construction and Analysis of Systems*, pages 299–314. Springer-Verlag, 2000.
8. J. Hopcroft and J. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley, 1979.
9. D. Kozen. Results on the propositional mu-calculus. *Theor. Comput. Sci.*, 27:333–354, 1983.
10. S. Miyano and T. Hayashi. Alternating finite automata on ω-words. *Theoretical Computer Science*, 32:321–330, 1984.
11. M. Y. Vardi. Alternating automata and program verification. In *Computer Science Today*, volume 1000 of *LNCS*, pages 471–485. Springer-Verlag, 1995.