State Space Reductions for Alternating Büchi Automata

Quotienting by Simulation Equivalences

Carsten Fritz and Thomas Wilke

Institut für Informatik und Praktische Mathematik
Christian-Albrechts-Universität, 24098 Kiel, Germany
{fritz,wilke}@ti.informatik.uni-kiel.de

Abstract. Quotienting by simulation equivalences is a well-established technique for reducing the size of nondeterministic Büchi automata. We adapt this technique to *alternating* Büchi automata. To this end we suggest two new quotients, namely minimax and semi-elective quotients, prove that they preserve the recognized languages, and show that computing them is not more difficult than computing quotients for nondeterministic Büchi automata. We explain the merits of of our quotienting procedures with respect to converting alternating Büchi automata into nondeterministic ones.

1 Introduction

Minimizing ω-automata is computationally difficult, because testing universality for nondeterministic finite automata on strings is already PSPACE-hard [GJ79][1]. Nevertheless, state-space reduction for ω-automata is an important issue in verification. Therefore, various state-space reduction heuristics have been developed, most of them for nondeterministic Büchi automata. Many of these heuristics are based on quotienting by bisimulation or simulation equivalences, that is, a given automaton is reduced in size by first computing an equivalence relation that identifies structurally "similar" states and then building a quotient with respect to this equivalence relation. In this paper, we show how this technique can be used to reduce the size of alternating Büchi automata.

Motivation. In the automata-theoretic approach to trace-based model checking [VW86] [VW94], a linear-time temporal formula (spec) is checked against a given transition system in three steps. First, the negation of the spec is translated into an equivalent nondeterministic Büchi automaton. Second, a product of the automaton and the given system is computed. Third, an emptiness test is performed for the product automaton. This solves the problem since the product automaton will accept exactly the error traces of the system with respect to the given spec. Recent work [GO01] suggests to split up the first step: the spec is translated into a (weak) alternating Büchi automaton, then, this automaton is simplified, and finally, the simplified automaton is translated into a nondeterministic Büchi automaton. Clearly, the size of the resulting nondeterministic Büchi

[1] This also implies that approximation of a minimum-size ω-automaton within a constant factor is impossible in polynomial time unless P=PSPACE.

automaton can be reduced by appropriate quotienting. However, an early quotienting of the alternating automaton may speed up the whole process.

Model-checking techniques that do not follow an automata-theoretic paradigm often interpret a given spec as a fixed-point formula and evaluate this formula on the transition system in question. This applies to various specification formalisms, for instance, CTL [McM93], and when symbolic techniques are used. Sometimes, a fixed-point logic can even be used directly for specification [LN01]. Since almost all specification formalisms and especially fixed-point logics can easily be translated into alternating automata and, conversely, alternating automata can be viewed as fixed-point formulas, see, e. g., [EJ88,MSS88,EJ91,Var94,KVW00], the following procedure suggests itself: translate the formula into an alternating automaton, reduce the automaton, interpret the reduced automaton as a fixed-point formula, and evaluate this formula on the transition system. That is, reducing automata is used for minimizing formulas. In the second step of the above process, quotienting by simulation equivalences might prove to be useful. As a start, one may want to study quotienting for alternating Büchi automata.

New Results. We first adapt direct, delayed, fair and ordinary simulation to alternating Büchi automata; this combines the definitions of [ESW01] for nondeterministic Büchi automata with the definition of [AHKV98] for alternating transition systems (Sect. 3). We then observe that because of alternation there is no simple way to define appropriate quotients with respect to any of the simulation equivalences. We suggest two new quotients, minimax and semi-elective quotients (Subsections 4.1 and 4.2, respectively), and show that these quotients preserve the languages recognized and can thus be used for state-space reduction with alternating Büchi automata. The proofs are quite complicated and cannot be given in this extended abstract; our main technical tool is a notion of composition of simulation strategies (see [FW02, Sect. 4]). We discuss how quotienting of alternating Büchi automata can be used to speed up the conversion of alternating Büchi automata to nondeterministic ones (Sect. 5). Finally, we explain why the efficient algorithms from [ESW01] can be used to compute our simulation relations and quotients for alternating Büchi automata and that computing these relations and quotients is especially easy for weak alternating Büchi automata (Sect. 6).

This extended abstract gives a semi-formal treatment of the material; for the exact details, the reader is referred to the full version, which is on-line, see [FW02].

Related Work. Henzinger, Kupferman, and Rajamani [HKR97,HR00] introduce fair bisimulation and simulation relations and describe how they can be computed efficiently. Somenzi and Bloem [SB00] and Etessami and Holzmann [EH00] use direct simulation to reduce the size of nondeterministic Büchi automata in the context of checking linear-time temporal properties and present efficient algorithms for computing direct simulation. Etessami, Schuller, and Wilke [ESW01] improve on [HKR97,HR00] and introduce delayed simulation to obtain better reductions; they make use of Jurdziński's algorithm [Jur00], which solves parity games. Gurumurthy, Bloem, and Somenzi [GBS02] build on this; Etessami [Ete02] follows different directions. Fast algorithms for computing fair simulation using games were also presented by Bustan and Grumberg [BG00]. Alur, Henzinger, Kupferman, and Vardi [AHKV98] study ordinary simulation for alternating transition systems. Gastin and Oddoux [GO01] use very weak alternating Büchi automata for translating linear-time temporal formulas into Büchi automata; they suggest some simplification rules for alternating Büchi automata, see above. Generating Büchi automata from linear-time temporal formulas is also dealt with in [GPVW95,DGV99].

There is more work on simulation in general, see, e. g., [Mil71,HHK95], and on using simulation for testing language inclusion in verification, see, e. g., [DHW91].

2 Notation and Basic Definitions

The set of natural numbers is denoted ω. Words over some alphabet Σ are viewed as functions from an initial segment of ω or ω itself to Σ, so when w is a word, then $w(i)$ denotes the letter at its ith position, where the first letter is in position 0.

For the purpose of this paper, an alternating Büchi automaton is a tuple

$$A = (Q, \Sigma, q_I, \Delta, E, U, F) \ , \tag{1}$$

where Q is a finite *set of states,* Σ a finite *alphabet,* $q_I \in Q$ an *initial state,* $\Delta \subseteq Q \times \Sigma \times Q$ a *transition relation,* $\{E, U\}$ a partition of Q in *existential* and *universal states,* where $E = \emptyset$ and $U = \emptyset$ are allowed, and $F \subseteq Q$ a set of *accepting states.* For notational simplicity, we will often write $\Delta(q, a)$ for $\{q' \mid (q, a, q') \in \Delta\}$.

Acceptance of alternating Büchi automata is best defined via games. For an alternating Büchi automaton A as above and an ω-word $w \in \Sigma^\omega$, the *word game* $G(A, w)$ is the Büchi game

$$G = (P, P_0, P_1, p_I, Z, F') \tag{2}$$

where $P = Q \times \omega$ is the set of positions, $P_0 = U \times \omega$ is the set of positions of Player 0, $P_1 = E \times \omega$ is the set of positions of Player 1, $p_I = (q_I, 0)$ is the initial position, $Z = \{((s, i), (s', i+1)) \mid (s, w(i), s') \in \Delta\}$ is the set of moves, and $F' = F \times \omega$ is the set of accepting positions. A *play* of the game is a finite or infinite sequence of positions $\pi = (p_i)_{i<n}$ (with $n \in \omega \cup \{\omega\}$) such that $p_0 = p_I$, $(p_i, p_{i+1}) \in Z$ for every $i+1 < n$, and $n < \omega$ only if there is no $p \in P$ such that $(p_{n-1}, p) \in Z$. A play π is a win for Player 1 if either infinitely many positions of π belong to F', or $n < \omega$ and $p_{n-1} \in P_0$; else π is a win for Player 0. Notice that the winning condition is phrased for Player 1, as opposed to conventions used in other papers.

Following [GH82], Player 0 will be called *Pathfinder* while Player 1 will be called *Automaton.* Acceptance is now defined as follows. The word w is *accepted* by the automaton A if Automaton wins the game $G(A, w)$, i. e., if Automaton has a winning strategy for $G(A, w)$. The language *recognized* by A is

$$L(A) = \{w \in \Sigma^\omega \mid \text{Automaton wins the game } G(A, w)\} \ . \tag{3}$$

For $q \in Q$, we will write $A(q)$ for the *translation* of A to q, which is defined to be the same automaton but with initial state q, i. e., $A(q) = (Q, \Sigma, q, \Delta, E, U, F)$.

In figures, existential states are shown as diamonds and universal states as squares; accepting states have double lines, see, e. g., Fig. 1.

3 Simulation Relations

In this section, we define three types of simulation relations for alternating Büchi automata, namely direct, delayed, and fair simulation, which are all based on the same simple game, only the winning condition varies; we follow the approach of [ESW01]. We also state basic properties of the three types of simulation relations, in particular, that simulation implies language containment.

Let $A^0 = (Q^0, \Sigma, p_I, \Delta^0, E^0, U^0, F^0)$ and $A^1 = (Q^1, \Sigma, q_I, \Delta^1, E^1, U^1, F^1)$ be alternating Büchi automata. The *basic simulation game* $G(A^0, A^1)$ is played by two players, *Spoiler* and *Duplicator,* who play the game in rounds. At the beginning of each round, a pair (p, q) of states $p \in Q^0$ and $q \in Q^1$ is given, and the players play as follows.

1. Spoiler chooses a letter $a \in \Sigma$.

2. Spoiler and Duplicator play as follows, depending on the modes of p and q.

— If $(p, q) \in E^0 \times E^1$, then Spoiler chooses a transition $(p, a, p') \in \Delta^0$ and after that Duplicator chooses a transition $(q, a, q') \in \Delta^1$.

— If $(p, q) \in U^0 \times U^1$, then Spoiler chooses a transition $(q, a, q') \in \Delta^1$ and after that Duplicator chooses a transition $(p, a, p') \in \Delta^0$.

— If $(p, q) \in E^0 \times U^1$, then Spoiler chooses transitions $(p, a, p') \in \Delta^0$ and $(q, a, q') \in \Delta^1$.

— If $(p, q) \in U^0 \times E^1$, then Duplicator chooses transitions $(p, a, p') \in \Delta^0$ and $(q, a, q') \in \Delta^1$.

3. The starting pair for the next round is (p', q').

The first round begins with the pair (p_I, q_I). If, at any point during the course of the game, a player cannot proceed any more, he or she looses (early). When the players proceed as above and no player looses early, they construct an infinite sequence $(p_0, q_0), (p_1, q_1), \ldots$ of pairs of states (with $p_0 = p_I$ and $q_0 = q_I$), and this sequence determines the winner, depending on the type of simulation relation we are interested in:

Direct simulation (di): Duplicator wins if for every i with $p_i \in F^0$ we have $q_i \in F^1$.

Delayed simulation (de): Duplicator wins if for every i with $p_i \in F^0$ there exists $j \geq i$ such that $q_j \in F^1$.

Fair simulation (f): Duplicator wins if there are only finitely many i with $p_i \in F^0$, or infinitely many j with $q_j \in F^1$.

In all other cases, Spoiler wins. This completes the description of the games.

When the basic game $G(A^0, A^1)$ is provided with one of the above winning conditions we write $\Game A^0 A^1 di$, $\Game A^0 A^1 de$, or $\Game A^0 A^1 f$ for this game. For $x \in \{di, de, f\}$, we define a relation $\leq_x$ on alternating Büchi automata. We write $A \leq_x B$ when Duplicator has a winning strategy in $\Game ABx$ and say that B *x-simulates* A. For states p of A, q of B, we write $p \leq_x q$ to indicate that $B(q)$ x-simulates $A(p)$. We write $\Game pqx$ instead of $\Game A(p)B(q)x$ when A and B are obvious from the context.

As an example, consider the automaton A over the alphabet $\Sigma = \{a, b\}$ given in Fig. 1. We argue that Duplicator wins $\Game 01x$ for $x \in \{de, di, f\}$. If Spoiler does not want to loose early, he has to choose letter b and transition $(1, b, 2)$ in the first round. Duplicator can reply by choosing transition $(0, b, 2)$, and the second round starts in position $(2, 2)$. Regardless of what Spoiler chooses in the second round, a and $(2, a, 2)$ or, alternatively, b and $(2, b, 2)$, Duplicator can reply such that after the second round, the game is in position $(2,2)$ again. The same holds true for any further round. Consequently, $0 \leq_x 1$ holds. Observe that $1 \leq_x 0$ is true for $x \in \{de, f\}$, but not for $x = di$.

For simplicity, we henceforth assume without loss of generality that all alternating Büchi automata are complete, which means that for every $q \in Q$, $a \in \Sigma$, there is a $q' \in Q$ such that $(q, a, q') \in \Delta$. Only in examples, we still allow incomplete automata in order to keep the examples small.

It is easy to see that the three simulation relations are increasingly coarser:

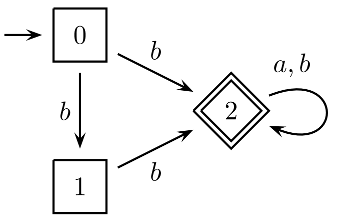

Fig. 1. Alternating Büchi automaton

Lemma 1. *The following relations hold between the three types of simulation relations:* $\leq_{di} \subsetneq \leq_{de} \subsetneq \leq_f$.

Unlike in the case of nondeterministic automata, it is not trivial to see that the three types of simulation relations are preorders, which the notation suggests. Using the strategy-composition method described in [FW02, Sect. 4], we are able to show this is, in fact, true:

Proposition 2 (simulations are preorders). *The simulation relations* $\leq_{di}$, $\leq_{de}$, *and* $\leq_f$ *are preorders, i. e., they are reflexive and transitive.*

The corresponding *simulation equivalence relations* are denoted by $\equiv_{de}$, $\equiv_{di}$, and $\equiv_f$, respectively, that is, $A \equiv_x B$ if $A \leq_x B$ and $B \leq_x A$. An analogue notation is used for states, that is, $p \equiv_x q$ if $A(p) \leq_x B(q)$ and $B(q) \leq_x A(p)$. The respective equivalence classes are denoted $[q]_x$.

We prove that all three types of simulations imply language containment and can thus be used for checking language containment:

Theorem 3 (simulation implies language containment). *Let* $x \in \{di, de, f\}$ *and let* A^0 *and* A^1 *be alternating Büchi automata.*

If $A^0 \leq_x A^1$, *then* $L(A^0) \subseteq L(A^1)$.

4 Simulation Quotients

In this section, we study how quotienting should be performed with respect to the three types of simulation relations. On the one hand, we will explain why the naive quotient construction—the construction studied in [ESW01] for nondeterministic Büchi automata and proved to work with such automata—does not work with alternating Büchi automata. On the other hand, we present two new quotient constructions, namely minimax and semi-elective quotients, which do work.

In general, when $\equiv$ is an equivalence relation on the state space of an alternating Büchi automaton A, we call an alternating Büchi automaton a *quotient of* A with respect to $\equiv$ if it is of the form

$$(Q/{\equiv}, \Sigma, [q_I]_{\equiv}, \Delta', E', U', F/{\equiv}) \tag{4}$$

and satisfies the following natural constraints for all $q, q' \in Q$, $a \in \Sigma$:

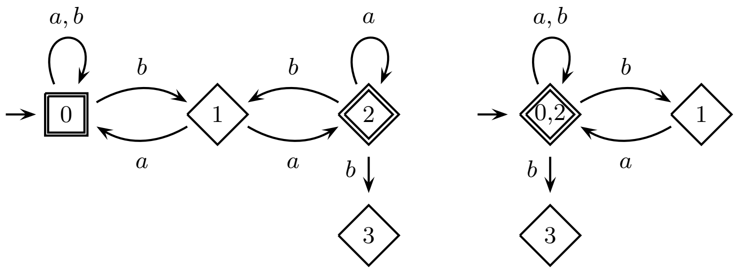

Fig. 2. Naive quotients don't work

(1) if $([q]_\equiv, a, [q']_\equiv) \in \Delta'$, then there exist states $\hat{q}$ and $\hat{q}'$ such that $\hat{q} \equiv q$, $\hat{q}' \equiv q'$ and $(\hat{q}, a, \hat{q}') \in \Delta$,

(2) if $[q]_\equiv \subseteq E$, then $[q]_\equiv \in E'$, and

(3) if $[q]_\equiv \subseteq U$, then $[q]_\equiv \in U'$.

Note that these conditions are minimal requirements so that a quotient really reflects the structure of A and is not just any automaton on the equivalence classes of $\equiv$. A quotient is a *naive quotient* if the converse holds true in (1), that is, if transitions are representative-wise.

In [ESW01], naive quotients of nondeterministic Büchi automata were proved to preserve the recognized language. This is no longer true for alternating Büchi automata, as problems arise for mixed simulation equivalence classes, i. e., classes containing both existential and universal states. In the naive quotienting, these states can be declared either existential or universal, but regardless of how they are declared the resulting naive quotients may *not* be equivalent to the original automaton. Consider, for instance, Fig. 2, showing an alternating Büchi automaton A over $\Sigma = \{a, b\}$ on the left, where $0 \equiv_x 2$ for $x \in \{de, di, f\}$, and one of the two possible naive x-quotients on the right; the other is obtained by declaring the state $\{0, 2\}$ universal. Now observe that the automaton on the left-hand side does not accept b^ω but $(ba)^\omega$, while the quotient on the right-hand side accepts all words and the other quotient does not accept $(ba)^\omega$. So neither one of the quotients is equivalent to the original automaton.

4.1 Direct Simulation and Minimax Quotients

We overcome the problems with direct simulation quotienting by using a more sophisticated transition relation for the quotient automaton which exploits the simple structure of direct simulation games.

Let q be a state of an alternating Büchi automaton A and $a \in \Sigma$. A state $q' \in \Delta(q, a)$ is an *x-maximal a-successor of* q if $q'' \leq_x q'$ holds for every $q'' \in \Delta(q, a)$ with $q' \leq_x q''$. We define

$$\max_a^x(q) = \{q' \in \Delta(q, a) \mid q' \text{ is an } x\text{-maximal } a\text{-successor of } q\} \ . \tag{5}$$

Symmetrically, *x-minimal a-successors of* q and $\min_a^x(q)$ are defined.

An *x-minimax quotient* of an alternating Büchi automaton A as in (1) is a quotient where the transition relation is given by

$$\begin{aligned}\Delta_x^m = \{([p]_x, a, [q]_x) \mid a \in \Sigma, p \in E, q \in \max_a^x(p)\}\\ \cup \{([p]_x, a, [q]_x) \mid a \in \Sigma, p \in U, q \in \min_a^x(p)\} \ . \end{aligned} \tag{6}$$

In a minimax quotient it does not matter how the mixed states are declared. This is not surprising, as we prove that a mixed class is deterministic in any minimax quotient: for every letter there is at most one outgoing edge labelled with that letter.

The main result about minimax quotients we prove is:

Theorem 4 (minimax quotients). *Let A be an alternating Büchi automaton as in (1) and B^m any di-minimax quotient of A.*

1) For all $p, q \in Q$ such that $p \leq_{di} q$, $A(q)$ di-simulates $B^m([p]_{di})$ and $B^m([q]_{di})$ di-simulates $A(p)$, that is, $[p] \leq_{di} q$ and $p \leq_{di} [q]_{di}$.

2) A and B^m di-simulate each other, that is, $A \equiv_{di} B^m$.

3) A and B^m are equivalent, that is, $L(A) = L(B^m)$.

4.2 Delayed Simulation and Semi-elective Quotients

For delayed simulation, neither naive quotients nor minimax quotients work. That naive quotients do not work follows from our previous example, see Fig. 2. To see that minimax quotients do not work, consider the automaton in Fig. 3. It is easy to see that $0 \geq_{de} 1$ but not $0 \equiv_{de} 1$, i. e., $\max_a^{de}(0) = \{0\}$. Therefore, the de-minimax quotient of the automaton has no transition from $[0]_{de}$ to $[1]_{de}$ and thus recognizes the empty language, which is not true for the automaton itself.

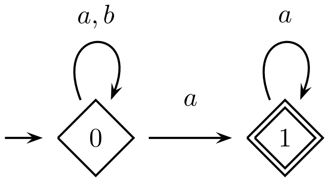

Fig. 3. De-minimax quotients don't work

To overcome the problem, we define so-called semi-elective quotients. A *semi-elective quotient* of an alternating Büchi automaton A is the quotient where the transition relation is given by

$$\begin{aligned}\Delta_x^s = \{([p]_x, a, [q]_x) \mid (p, a, q) \in \Delta, p \in E\}\\ \cup \{([p]_x, a, [q]_x) \mid a \in \Sigma, [p]_x \subseteq U, q \in \min_a^x(p)\}, \end{aligned} \tag{7}$$

and every mixed class is existential. That is, purely universal classes are treated as with minimax quotienting while purely existential and mixed classes are existential states and have representative-wise transitions. Given an alternating Büchi automaton A and $x \in \{de, di, f\}$, we write A_x^s for the respective semi-elective quotient.

Some possible optimizations of this construction are discussed in [FW02, Subsect. 7.4].

The main result about semi-elective quotients we prove is:

Theorem 5 (semi-elective quotients). *Let A be an alternating Büchi automaton as in (1) and $x \in \{de, di\}$.*

1) For all $p, q \in Q$ such that $p \leq_x q$, $A(q)$ x-simulates $A_x^s([p]_x)$ and $A_x^s([q]_x)$ x-simulates $A(p)$, that is, $[p] \leq_x q$ and $p \leq_x [q]_x$.

2) A and A_x^s x-simulate each other, that is, $A \equiv_x A_x^s$.

3) A and A_x^s are equivalent, that is, $L(A) = L(A_x^s)$.

The reader may want to verify that the automaton depicted in Fig. 1 shows that for universal classes in de-semi-elective quotients it is necessary that only de-min-successors are taken.

As an example for the construction of a semi-elective quotient automaton modulo delayed simulation, consider Fig. 4. For the automaton A on the left, we have $2 <_{de} 1 \equiv_{de} 5 <_{de} 0 \equiv_{de} 3 <_{de} 4$. Thus there are four states in the quotient automaton A_{de}^s on the right. Since $\min_b^{de}(1) = \{2\}$, the edge $([1]_{de}, b, [1]_{de})$ is not in Δ_{de}^s; since $\min_a^{de}(0) = \min_b^{de}(0) = \{1\}$, there is no edge $([0]_{de}, c, [3]_{de})$ in Δ_{de}^s with $c \in \{a, b\}$. And since $\min_a^{de}(3) = \min_b^{de}(3) = \{1\}$, there is no edge $([3]_{de}, c, [4]_{de})$ in Δ_{de}^s with $c \in \{a, b\}$. Consequently, the state $[4]_{de}$ is not reachable in A_{de}^s and should be removed in a successive optimization step of the quotient automaton.

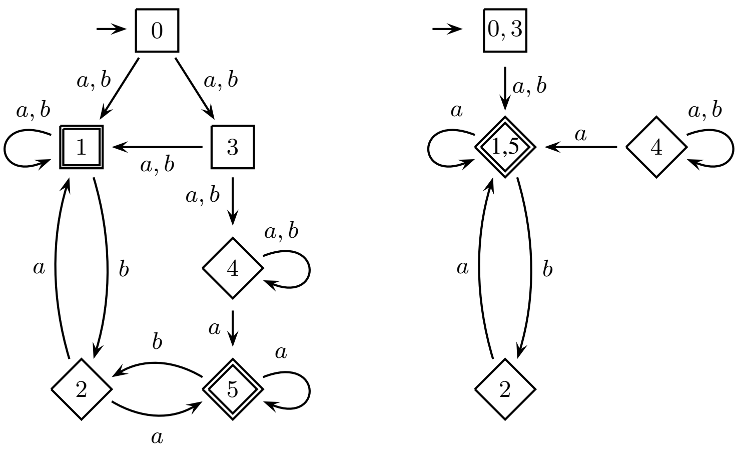

Fig. 4. Automaton and de-semi-elective quotient

5 Converting Alternating Büchi Automata to Nondeterministic Büchi Automata

Given an alternating Büchi automaton A, the standard approach for constructing an equivalent nondeterministic (i. e., non-alternating) Büchi automaton is the construction of Miyano and Hayashi [MH84]. Their construction is a modified power set construction where the states are *pairs* of subsets of the state set of A. We will call the automaton resulting from the Miyano–Hayashi construction the MH-automaton and denote it by A_{nd}. Note that A_{nd} is an exponential size automaton (and that this is necessarily so in the worst case).

In order to construct a small nondeterministic Büchi automaton from an alternating Büchi automaton, our simulation quotienting can be applied to the alternating automaton prior to the Miyano–Hayashi construction and a subsequent simulation quotienting. (The subsequent simulation quotienting should still be done.) Traditionally, simulation quotienting is only applied to the MH-automaton.

Figure 5 shows these two possible ways.

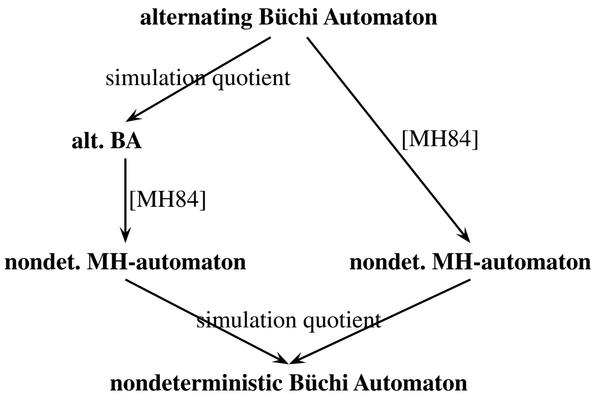

Fig. 5. Two ways from alternating BA to nondet. BA

First applying simulation quotienting to the alternating automaton is relatively cheap as compared to simulation quotienting the MH-automaton (cf. Sect. 6), since the MH-construction incurs an exponential growth (see above). For this reason, a state space reduction of the alternating automaton often results in a substantial reduction of the size of the MH-automaton. Aside from these savings in the state space, a smaller intermediate MH-automaton speeds up the subsequent simulation quotienting.

The following lemma states that the Miyano-Hayashi-construction preserves the simulation preorder.

Lemma 6. *Let A^0 and A^1 be alternating Büchi automata and $x \in \{di, de, f\}$. If $A^0 \leq_x A^1$, then $A^0_{nd} \leq_x A^1_{nd}$.*

Lemma 6 obviously implies the following corollary.

Corollary 7. *For every ABA A and $x \in \{di, de\}$, $((A^x)_{nd})^x \equiv_x (A_{nd})^x$ holds.*

Proof. We have $A^x \equiv_x A$, hence, by Lemma 6, $(A^x)_{nd} \equiv_x A_{nd}$, and $((A^x)_{nd})^x \equiv_x (A_{nd})^x$ follows immediately. □

That is, the original alternating automaton, the intermediate automata of Fig. 5 and the resulting nondeterministic Büchi automaton are all simulation equivalent.

But note that the simulation quotients of simulation equivalent (alternating or nondeterministic) automata need not be isomorphic: Some additional optimizations of the quotient construction (for example, using pseudo-accepting states as described in [FW02, Subsect. 7.4]) work to a lesser extent for A_{nd} than for A. So, when using these optimizations, the quotient resulting from taking the left-hand way in Fig. 5 is, for certain instances, smaller than the result of the right-hand way. (Without additional optimizations, the left-hand quotient will not be smaller than the right-hand quotient.)

Anyway, taking the left-hand way always results in an automaton which is no larger than the automaton arrived at by taking the right-hand way:

Proposition 8. *For every ABA A and $x \in \{di, de\}$, $((A^x)_{nd})^x$ has at most as many states as $(A_{nd})^x$.*

As an example for the possible state space reduction by quotienting of alternating automata, consider Fig. 6 showing A_3, an alternating automaton with three sub-automata B^1, B^2, B^3 (with initial states q_0^1, q_0^2, q_0^3) of sizes $2, 3, 4$, respectively. The automaton A_3 can be generalized to an automaton A_n with n sub-automata of sizes $2, \ldots, n+1$ in an obvious manner.

Then, A_n has $\theta(n^2)$ states while $(A_n)_{nd}$ has $2^{\theta(n \log n)}$ states, i. e., the size of $(A_n)_{nd}$ is in $2^{O(\sqrt{m} \log m)}$ where m is the size of A_n. But A_n^{de} only has three states since $q_0^1 \leq_{de} q_0^k$ for $k > 1$, i. e., the transitions from q_I^n to $q_0^2, \ldots, q_0^n$ are removed by de-semi-elective quotienting (and $(A_n^{de})_{nd}$ also has only three states).

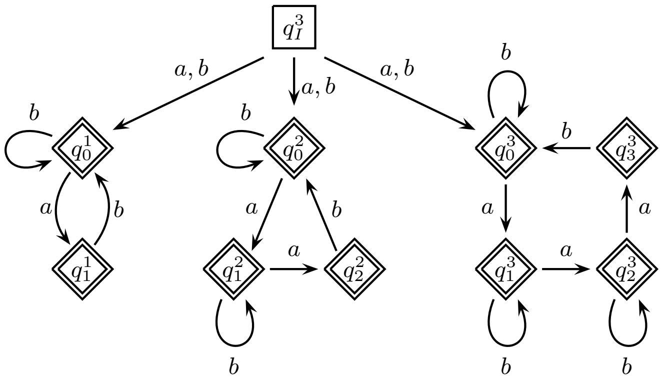

Fig. 6. A_3

6 Efficient Algorithms

Efficient algorithms for computing simulation relations and simulation quotients of nondeterministic Büchi automata are given in [ESW01], the main idea being a reduction to parity games. Using similar reductions, we obtain the same complexity bounds:

Theorem 9 (computing simulation relations). *In the following, n stands for the number of states and m for the number of transitions of an alternating Büchi automaton.*

Direct simulation relations can be computed in time $O(nm)$. Delayed and fair simulation relations can be computed in time $O(n^3m)$ and space $O(nm)$. For weak alternating automata, direct as well as delayed and fair simulation can be computed in time $O(nm)$.

We mention that computing the respective quotients, regardless of which type they are, can be done within the same complexity bounds.

7 Conclusion

We have presented new quotients for alternating Büchi automata which preserve the recognized languages, can be computed in polynomial time, and yield considerable reductions in size (just as for nondeterministic Büchi automata). They can be used to speed up explicit-state as well as symbolic model-checking. For practical applications, these reductions should be combined with the other known state-space reduction techniques.

References

[AHKV98] Rajeev Alur, Thomas A. Henzinger, Orna Kupferman, and Moshe Y. Vardi. Alternating refinement relations. In D. Sangiorgi and R. de Simone, editors, *CONCUR 1998*, vol. 1466 of *LNCS*, pp. 163–178, 1998.

[BG00] Doran Bustan and Orna Grumberg. Checking for fair simulation in models with Büchi fairness constraints, Dec. 2000. Tech. Rep. TR-CS-2000-13, Technion.

[DGV99] Marco Daniele, Fausto Giunchiglia, and Moshe Y. Vardi. Improved automata generation for linear time temporal logic. In N. Halbwachs and D. Peled, editors, *CAV 1999*, vol. 1633 of *LNCS*, pp. 249–260, 1999.

[DHW91] David L. Dill, Alan J. Hu, and Howard Wong-Toi. Checking for language inclusion using simulation preorders. In Kim G. Larsen and Arne Skou, editors, *CAV 1991*, vol. 575 of *LNCS*, pp. 255–265, 1991.

[EH00] Kousha Etessami and Gerard Holzmann. Optimizing Büchi automata. In Catuscia Palamidessi, editor, *CONCUR 2000*, vol. 1877 of *LNCS*, pp. 153–167, 2000.

[EJ88] E. Allen Emerson and Charanjit S. Jutla. The complexity of tree automata and logics of programs (extended abstract). In *FoCS 1988*, pp. 328–337. 1988. IEEE.

[EJ91] E. Allen Emerson and Charanjit S. Jutla. Tree automata, mu-calculus and determinacy. In *FoCS 1991*, pp. 368–377, San Juan, Puerto Rico, Oct. 1991. IEEE.

[ESW01] Kousha Etessami, Rebecca Schuller, and Thomas Wilke. Fair simulation relations, parity games, and state space reduction for Büchi automata. In F. Orejas, P.G. Spirakis, and J. van Leeuwen, editors, *ICALP 2001*, vol. 2076 of *LNCS*, pp. 694–707, 2001.

[Ete02] Kousha Etessami. A Hierarchy of Polynomial-Time Computable Simulations for Automata. CONCUR 2002, to appear.

[FW02] Carsten Fritz and Thomas Wilke. Simulation relations for alternating Büchi automata. Extended version of tech. rep. 2019, Institut für Informatik, CAU Kiel, July 2002. URL: `http://www.informatik.uni-kiel.de/~fritz/TechRep2019_ext.ps`.

[GH82] Yuri Gurevich and Leo Harrington. Trees, automata, and games. In *14th ACM Symp. on the Theory of Computing*, pp. 60–65, 1982.

[GJ79] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W.H. Freeman and Co., San Francisco, 1979.

[GO01] Paul Gastin and Denis Oddoux. Fast LTL to Büchi automata translation. In G. Berry, H. Comon, and A. Finkel, editors, *CAV 2001*, vol. 2102 of *LNCS*, pp. 53–65, 2001.

[GPVW95] Rob Gerth, Doron Peled, Moshe Y. Vardi, and Pierre Wolper. Simple on-the-fly automatic verification of linear temporal logic. In *PSTV 1995*, pp. 3–18, Warsaw, Poland, June 1995. Chapman Hall.

[GBS02] Sankar Gurumurthy, Roderick Bloem, and Fabio Somenzi. Fair simulation minimization. CAV 2002, to appear.

[HHK95] Monika Henzinger Rauch, Thomas A. Henzinger, and Peter W. Kopke. Computing simulations on finite and infinite graphs. In *FoCS 1995*, pp. 453–462, 1995.

[HKR97] Thomas A. Henzinger, Orna Kupferman, and Sriram K. Rajamani. Fair simulation. In *CONCUR 1997*, vol. 1243 of *LNCS*, pp. 273–287, 1997.

[HR00] Thomas A. Henzinger and Sriram K. Rajamani. Fair bisimulation. In S. Graf and M. Schwartzbach, editors, *TACAS 2000*, vol. 1785 of *LNCS*, pp. 299–314, 2000.

[Jur00] Marcin Jurdziński. Small progress measures for solving parity games. In H. Reichel and S. Tison, editors, *STACS 2000*, vol. 1770 of *LNCS*, pp. 290–301, 2000.

[KVW00] Orna Kupferman, Moshe Y. Vardi, and Pierre Wolper. An automata-theoretic approach to branching-time model checking. *Journal of the ACM*, 47(2):312–360, 2000.

[LN01] Martin Leucker and Thomas Noll. Truth/SLC - A parallel verification platform for concurrent systems. In G. Berry, H. Comon, and A. Finkel, editors, *CAV 2001*, vol. 2102 of *LNCS*, pp. 255–259, 2001.

[McM93] Kenneth L. McMillan. *Symbolic Model Checking*. Kluwer, Boston, 1993.

[MH84] S. Miyano and T. Hayashi. Alternating finite automata on ω-words. *Theoretical Computer Science*, 32:321–330, 1984.

[Mil71] Robin Milner. An algebraic definition of simulation between programs. In D. C. Cooper, editor, *Proc. of the 2nd Int. Joint Conf. on Artificial Intelligence*, pp. 481–489, London, UK, September 1971. William Kaufmann.

[MSS88] David E. Muller, Ahmed Saoudi, and Paul E. Schupp. Weak alternating automata give a simple explanation of why most temporal and dynamic logics are decidable in exponential time. In *LICS 1988*, pp. 422–427, IEEE Computer Society, 1988.

[SB00] Fabio Somenzi and Roderick Bloem. Efficient Büchi automata from LTL formulae. In E. Allen Emerson and A. Prasad Sistla, editors, *CAV 2000*, vol. 1855 of *LNCS*, pp. 248–263, 2000.

[Var94] Moshe Y. Vardi. Nontraditional applications of automata theory. In *Theoretical Aspects of Computer Software*, vol. 789 of *LNCS*, pp. 575–597, 1994.

[VW86] Moshe Y. Vardi and Pierre Wolper. An automata-theoretic approach to automatic program verification. In Dexter Kozen, editor, *LICS 1986*, pp. 332–344, Cambridge, Mass., 16–18 June 1986.

[VW94] Moshe Y. Vardi and Pierre Wolper. Reasoning about infinite computations. *Information and Computation*, 115(1):1–37, 1994.