Simulations in Rank-Based Büchi Automata Complementation

Yu-Fang Chen[1], Vojtěch Havlena[2], and Ondřej Lengál[2(✉)]

[1] Academia Sinica, Taipei, Taiwan
[2] FIT, IT4I Centre of Excellence, Brno University of Technology, Brno, Czech Republic
lengal@fit.vutbr.cz

Abstract. Complementation of Büchi automata is an essential technique used in some approaches for termination analysis of programs. The long search for an optimal complementation construction climaxed with the work of Schewe, who proposed a worst-case optimal rank-based procedure that generates complements of a size matching the theoretical lower bound of $(0.76n)^n$, modulo a polynomial factor of $\mathcal{O}(n^2)$. Although worst-case optimal, the procedure in many cases produces automata that are unnecessarily large. In this paper, we propose several ways of how to use the direct and delayed simulation relations to reduce the size of the automaton obtained in the rank-based complementation procedure. Our techniques are based on either (i) ignoring macrostates that cannot be used for accepting a word in the complement or (ii) saturating macrostates with simulation-smaller states, in order to decrease their total number. We experimentally showed that our techniques can indeed considerably decrease the size of the output of the complementation.

1 Introduction

Büchi automata (BA) complementation is a fundamental problem in program analysis and formal verification, from both theoretical and practical angles. It is, for instance, a critical step in some approaches for termination analysis, which is an essential part of establishing total correctness of programs [9,14,19]. Moreover, BA complementation is used as a component of decision procedures of some logics for reasoning about programs, such as S1S capturing a decidable fragment of second-order arithmetic [6] or the temporal logics ETL and QPTL [35].

The study of the BA complementation problem can be traced back to 1962, when Büchi introduced his automaton model in the seminal paper [6] in the context of a decision procedure for the S1S fragment of second-order arithmetic. In the paper, a doubly exponential complementation algorithm based on the infinite Ramsey theorem is proposed. In 1988, Safra [32] introduced a complementation procedure with an $n^{\mathcal{O}(n)}$ upper bound and, in the same year, Michel [28] established an $n!$ lower bound. From the traditional theoretical point of view, the problem was already solved, since exponents in the two bounds matched under

A. W. Lin (Ed.): APLAS 2019, LNCS 11893, pp. 447–467, 2019.
https://doi.org/10.1007/978-3-030-34175-6_23

the $\mathcal{O}$ notation (recall that $n!$ is approximately $(n/e)^n$). From a more practical point of view, a linear factor in an exponent has a significant impact on real-world applications. It was established that the upper bound of Safra's construction is 2^{2n}, so the hunt for an optimal algorithm continued [38]. A series of research efforts participated in narrowing the gap [15,23,24,39,41]. The long journey climaxed with the result of Schewe [33], who proposed an optimal rank-based procedure that generates complements of a size matching the theoretical lower bound of $(0.76n)^n$ found by Yan [41], modulo a polynomial factor of $\mathcal{O}(n^2)$.

Although the algorithm of Schewe is worst-case optimal, it often generates unnecessarily large complements. The standard approach to alleviate this problem is to decrease the size of the input BA before the complementation starts. Since minimization of (nondeterministic) BAs is a PSpace-complete problem, more lightweight reduction methods are necessary. The most prevalent approaches are those based on various notions of *simulation-based reduction*, such as reductions based on *direct simulation* [7,36], a richer *delayed simulation* [12], or their *multi-pebble* variants [13]. These approaches first compute a simulation relation over the input BA—which can be done with the time complexity $\mathcal{O}(mn)$ [8,20,22,30,31] and $\mathcal{O}(mn^3)$ [12] for direct and delayed simulation respectively, with the number of states n and transitions m—and then construct a *quotient* BA by merging simulation-equivalent states, while preserving the language of the input BA. The other approach is a reduction based on *fair simulation* [18]. The fair simulation cannot, however, be used for quotienting, but still it can be used for merging certain states and removing transitions. The reduced BA is used as the input of the complementation, which often significantly reduces the size of the result.

In this paper, we propose several ways of how to exploit the direct and delayed simulations in BA complementation even further to obtain smaller complements and shorter running times. We focus, in particular, on the optimal *rank-based* complementation procedure of Schewe [33]. Essentially, the rank-based construction is an extension of traditional subset construction for determinizing finite automata, with some additional information kept in each macrostate (a state in the complemented BA) to track the acceptance condition of all runs of the input automaton on a given word. In particular, it stores the *rank* of each state in a macrostate, which, informally, measures the distance to the last accepting state on the corresponding run in the input BA. The main contributions of this paper are the following optimisations of rank-based complementation for BAs, for an input BA $\mathcal{A}$ and the output of the rank-based complementation algorithm $\mathcal{B}$.

1. *Purging*: We use simulation relations over $\mathcal{A}$ to remove some useless macrostates during the construction of $\mathcal{B}$. In particular, if a state p is simulated by q in $\mathcal{A}$, this puts a restriction on the relation between the ranks of runs from p and from q. As a consequence, macrostates that assign ranks violating this restriction can be purged from $\mathcal{B}$.
2. *Saturation*: We saturate macrostates with states that are simulated by the macrostate; this can reduce the total number of states of $\mathcal{B}$ because two or more macrostates can be mapped to a single saturated macrostate. This is

inspired by the technique of Glabbeek and Ploeger that uses *closures* in finite automata determinization [17].

The proposed optimizations are orthogonal to simulation-based size reduction mentioned above. Since the quotienting methods are based on taking only the symmetric fragment of the simulation, i.e., they merge states that simulate *each other*, after the quotienting, there might still be many pairs where the simulation holds in only one way, and can therefore be exploited by our techniques. Since the considered notions of simulation-based quotienting preserve the respective simulations, our techniques can be used to optimize the complementation *at no additional cost*. Our experimental evaluation of the optimizations showed that in many cases, they indeed significantly reduce the size of the complemented BA.

2 Preliminaries

We fix a finite nonempty alphabet Σ and the first infinite ordinal $\omega = \{0, 1, \dots\}$. For $n \in \omega$, by $[n]$ we denote the set $\{0, \dots, n\}$. An (infinite) word α is represented as a function $\alpha : \omega \to \Sigma$ where the i-th symbol is denoted as α_i. A finite word w of length $n+1$ is represented as a function $w : [n] \to \Sigma$. The finite word of length 0 is denoted as ϵ. We abuse notation and sometimes also represent α as an infinite sequence $\alpha = \alpha_0\alpha_1 \dots$ and w as a finite sequence $w = w_0 \dots w_{n-1}$. The suffix $\alpha_i\alpha_{i+1} \dots$ of α is denoted by $\alpha_{i:\omega}$. We use Σ^ω to denote the set of all infinite words over Σ and Σ^* to denote the set of all finite words. For $L \subseteq \Sigma^*$ we define $L^* = \{u \in \Sigma^* \mid u = w_1 \cdots w_n \wedge \forall 1 \leq i \leq n : w_i \in L\}$ and $L^\omega = \{\alpha \in \Sigma^\omega \mid \alpha = w_1 w_2 \cdots \wedge \forall i \geq 1 : w_i \in L\}$ (note that $\{\epsilon\}^\omega = \emptyset$). Given $L_1, L_2 \subseteq \Sigma^*$, we use L_1L_2 to denote the set $\{w_1w_2 \mid w_1 \in L_1, w_2 \in L_2\}$.

A (nondeterministic) *Büchi automaton* (BA) over Σ is a quadruple $\mathcal{A} = (Q, \delta, I, F)$ where Q is a finite set of *states*, δ is a *transition function* $\delta : Q \times \Sigma \to 2^Q$, and $I, F \subseteq Q$ are the sets of *initial* and *accepting* states respectively. We sometimes treat δ as a set of transitions $p \xrightarrow{a} q$, for instance, we use $p \xrightarrow{a} q \in \delta$ to denote that $q \in \delta(p, a)$. Moreover, we extend δ to sets of states $P \subseteq Q$ as $\delta(P, a) = \bigcup_{p \in P} \delta(p, a)$. A *run* of $\mathcal{A}$ from $q \in Q$ on an input word α is an infinite sequence $\rho : \omega \to Q$ that starts in q and respects δ, i.e., $\rho_0 = q$ and $\forall i \geq 0 : \rho_i \xrightarrow{\alpha_i} \rho_{i+1} \in \delta$. We say that ρ is accepting iff it contains infinitely many occurrences of some accepting state, i.e., $\exists q_f \in F : |\{i \in \omega \mid \rho_i = q_f\}| = \omega$. A word α is accepted by $\mathcal{A}$ from a state $q \in Q$ if there is an accepting run ρ of $\mathcal{A}$ from q, i.e., $\rho_0 = q$. The set $\mathcal{L}_\mathcal{A}(q) = \{\alpha \in \Sigma^\omega \mid \mathcal{A} \text{ accepts } \alpha \text{ from } q\}$ is called the *language* of q (in $\mathcal{A}$). Given a set of states $R \subseteq Q$, we define the language of R as $\mathcal{L}_\mathcal{A}(R) = \bigcup_{q \in R} \mathcal{L}_\mathcal{A}(q)$ and the language of $\mathcal{A}$ as $\mathcal{L}(\mathcal{A}) = \mathcal{L}_\mathcal{A}(I)$. For a pair of states p and q in $\mathcal{A}$, we use $p \subseteq_\mathcal{L} q$ to denote $\mathcal{L}_\mathcal{A}(p) \subseteq \mathcal{L}_\mathcal{A}(q)$.

Without loss of generality, in this paper, we assume $\mathcal{A}$ to be complete, i.e., for every state q and symbol a, it holds that $\delta(q, a) \neq \emptyset$. A *trace* over a word α is an infinite sequence $\pi = q_0 \xrightarrow{\alpha_0} q_1 \xrightarrow{\alpha_1} \cdots$ such that $\rho = q_0q_1 \dots$ is a run of $\mathcal{A}$ over α from q_0. We say π is *fair* if it contains infinitely many accepting states. Moreover, we use $p \overset{w}{\rightsquigarrow} q$ for $w \in \Sigma^*$ to denote that q is reachable from p over the word w; if a path from p to q over w contains an accepting state, we can write $p \overset{w}{\underset{F}{\rightsquigarrow}} q$. In this paper, we fix a complete BA $\mathcal{A} = (Q, \delta, I, F)$.

2.1 Simulations

We introduce simulation relations between states of a BA $\mathcal{A}$ using the game semantics in a similar manner as in the extensive study of Clemente and Mayr [26]. In particular, in a *simulation game* between two players (called Spoiler and Duplicator) in $\mathcal{A}$ from a pair of states (p_0, r_0), for any (infinite) trace over a word α that Spoiler takes starting from p_0, Duplicator tries to mimic the trace starting from r_0. On the other hand, Spoiler tries to find a trace that Duplicator cannot mimic. The game starts in the configuration (p_0, r_0) and every i-th round proceeds by, first, Spoiler choosing a transition $p_i \xrightarrow{\alpha_i} p_{i+1}$ and, second, Duplicator mimicking Spoiler by choosing a matching transition $r_i \xrightarrow{\alpha_i} r_{i+1}$ over the same symbol α_i. The next game configuration is (p_{i+1}, r_{i+1}). Suppose that $\pi_p = p_0 \xrightarrow{\alpha_0} p_1 \xrightarrow{\alpha_1} \cdots$ and $\pi_r = r_0 \xrightarrow{\alpha_0} r_1 \xrightarrow{\alpha_1} \cdots$ are the two (infinite) traces constructed during the game. Duplicator *wins* the simulation game if $\mathcal{C}^x(\pi_p, \pi_r)$ holds, where $\mathcal{C}^x(\pi_p, \pi_r)$ is a condition that depends on the particular simulation. In the current paper, we consider the following simulation relations:

- **direct** [11]: $\mathcal{C}^{di}(\pi_p, \pi_r) \overset{\text{def}}{\iff} \forall i : p_i \in F \Rightarrow r_i \in F$,
- **delayed** [12]: $\mathcal{C}^{de}(\pi_p, \pi_r) \overset{\text{def}}{\iff} \forall i : p_i \in F \Rightarrow \exists k \geq i : r_k \in F$, and
- **fair** [21]: $\mathcal{C}^{f}(\pi_p, \pi_r) \overset{\text{def}}{\iff}$ if π_p is fair, then π_r is fair.

A maximal x-simulation relation $\preceq_x \subseteq Q \times Q$, for $x \in \{di, de, f\}$, is defined such that $p \preceq_x r$ iff Duplicator has a winning strategy in the simulation game with the winning condition $\mathcal{C}^x$ starting from (p, r). Formally, we define a strategy to be a (total) mapping $\sigma : Q \times (Q \times \Sigma \times Q) \to Q$ such that $\sigma(r, p \xrightarrow{a} p') \in \delta(r, a)$, i.e., if Duplicator is in state r and Spoiler selects a transition $p \xrightarrow{a} p'$, the strategy picks a state r' such that $r \xrightarrow{a} r' \in \delta$ (and because $\mathcal{A}$ is complete, such a transition always exists). Note that Duplicator cannot look ahead at Spoiler's future moves. We use σ_x to denote any winning strategy of Duplicator in the $\mathcal{C}^x$ simulation game. Let σ_x and σ'_x be a pair of winning strategies in the $\mathcal{C}^x$ simulation game. We say that σ_x is *dominated* by σ'_x if for all states p and all transitions $q \xrightarrow{a} q'$ it holds that $\sigma_x(p, q \xrightarrow{a} q') \preceq_x \sigma'_x(p, q \xrightarrow{a} q')$, and that σ_x is *strictly dominated* by σ'_x if σ_x is dominated by σ'_x and σ_x does not dominate σ'_x. A strategy is *dominating* if it is not strictly dominated by any other strategy. Strategies are also lifted to traces as follows: let π_p be as above, then $\sigma(r_0, \pi_p) = r_0 \xrightarrow{\alpha_0} r_1 \xrightarrow{\alpha_1} \cdots$ where for all $i \leq 0$ it holds that $\sigma(r_i, p_i \xrightarrow{\alpha_i} p_{i+1}) = r_{i+1}$. The considered simulation relations form the following hierarchy: $\preceq_{di} \subseteq \preceq_{de} \subseteq \preceq_f \subseteq \subseteq_{\mathcal{L}}$. Note that every maximal simulation relation is a preorder, i.e., reflexive and transitive.

2.2 Run DAGs

In this section, we recall the terminology from [33] (which is a minor modification of the terminology from [24]). We fix the definition of the *run DAG* of $\mathcal{A}$ over a word α to be a DAG (directed acyclic graph) $\mathcal{G}_\alpha = (V, E)$ of vertices V and edges E where

- $V \subseteq Q \times \omega$ s.t. $(q, i) \in V$ iff there is a run ρ of $\mathcal{A}$ over α with $\rho_i = q$,
- $E \subseteq V \times V$ s.t. $((q, i), (q', i')) \in E$ iff $i' = i + 1$ and $q' \in \delta(q, \alpha_i)$.

Given $\mathcal{G}_\alpha$ as above, we will write $(p, i) \in \mathcal{G}_\alpha$ to denote that $(p, i) \in V$. We call (p, i) *accepting* if p is an accepting state. $\mathcal{G}_\alpha$ is *rejecting* if it contains no path with infinitely many accepting vertices. A vertex $(p, i) \in \mathcal{G}_\alpha$ is *finite* if the set of vertices reachable from (p, i) is finite, *infinite* if it is not finite, and *endangered* if (p, i) cannot reach an accepting vertex.

We assign ranks to vertices of run DAGs as follows: Let $\mathcal{G}^0_\alpha = \mathcal{G}_\alpha$ and $j = 0$. Repeat the following steps until the fixpoint or for at most $2n + 1$ steps, where n is the number of states of $\mathcal{A}$.

- Set $\mathit{rank}_\alpha(p, i) := j$ for all finite vertices (p, i) of $\mathcal{G}^j_\alpha$ and let $\mathcal{G}^{j+1}_\alpha$ be $\mathcal{G}^j_\alpha$ minus the vertices with the rank j.
- Set $\mathit{rank}_\alpha(p, i) := j + 1$ for all endangered vertices (p, i) of $\mathcal{G}^{j+1}_\alpha$ and let $\mathcal{G}^{j+2}_\alpha$ be $\mathcal{G}^{j+1}_\alpha$ minus the vertices with the rank $j + 1$.
- Set $j := j + 2$.

For all vertices v that have not been assigned a rank yet, we assign $\mathit{rank}_\alpha(v) := \omega$. (Note that since $\mathcal{A}$ is complete, then $\mathcal{G}^1_\alpha = \mathcal{G}^0_\alpha$.)

Lemma 1. *If $\alpha \notin \mathcal{L}(\mathcal{A})$, then $0 \leq \mathit{rank}_\alpha(v) \leq 2n$ for all $v \in \mathcal{G}_\alpha$. Moreover, if $\alpha \in \mathcal{L}(\mathcal{A})$, then there is a vertex $(p, 0) \in \mathcal{G}_\alpha$ s.t. $\mathit{rank}_\alpha(p, 0) = \omega$.*

Proof. Follows from Corollary 3.3 in [24]. □

3 Complementing Büchi Automata

We use as the starting point the complementation procedure of Schewe [33, Section 3.1], which we denote as $\textsc{Comp}_\text{S}$ (the ‘S’ stands for ‘Schewe’). The procedure works with the notion of level rankings. Given $n = |Q|$, a *(level) ranking* is a function $f : Q \to [2n]$ such that $\{f(q_f) \mid q_f \in F\} \subseteq \{0, 2, \ldots, 2n\}$, i.e., f assigns even ranks to accepting states of $\mathcal{A}$. [1] For a ranking f, the *rank* of f is defined as $\mathit{rank}(f) = \max\{f(q) \mid q \in Q\}$. For a set of states $S \subseteq Q$, we call f to be *S-tight* if (i) it has an odd rank r, (ii) $\{f(s) \mid s \in S\} \supseteq \{1, 3, \ldots, r\}$, and (iii) $\{f(q) \mid q \notin S\} = \{0\}$. A ranking is *tight* if it is Q-tight; we use $\mathcal{T}$ to denote

[1] Note that our basic definitions slightly differs from the ones in Sect. 2.3 of [33]. This is because of a typo in [33]; indeed, if the procedure from [33] is implemented as is, the output does not accept the complement (there might be a macrostate (S, O, f) where S contains accepting states and O is empty, and, therefore, the whole macrostate is accepting, which is wrong).

the set of all tight rankings. For a pair of rankings f and f', a set $S \subseteq Q$, and a symbol $a \in \Sigma$, we use $f' \leq_a^S f$ iff for every $q \in S$ and $q' \in \delta(q, a)$ it holds that $f'(q') \leq f(q)$.

The Comp$_S$ procedure constructs the BA $\mathcal{B}_S = (Q', \delta', I', F')$ whose components are defined as follows:

- $Q' = Q_1 \cup Q_2$ where
  - $Q_1 = 2^Q$ and
  - $Q_2 = \{(S, O, f, i) \in 2^Q \times 2^Q \times \mathcal{T} \times \{0, 2, \ldots, 2n-2\} \mid f \text{ is } S\text{-tight}, O \subseteq S \cap f^{-1}(i)\}$,
- $I' = \{I\}$,
- $\delta' = \delta_1 \cup \delta_2 \cup \delta_3$ where
  - $\delta_1 : Q_1 \times \Sigma \to 2^{Q_1}$ such that $\delta_1(S, a) = \{\delta(S, a)\}$,
  - $\delta_2 : Q_1 \times \Sigma \to 2^{Q_2}$ such that $\delta_2(S, a) = \{(S', \emptyset, f, 0) \mid S' = \delta(S, a), f \text{ is } S'\text{-tight}\}$, and
  - $\delta_3 : Q_2 \times \Sigma \to 2^{Q_2}$ such that $(S', O', f', i') \in \delta_3((S, O, f, i), a)$ iff $S' = \delta(S, a)$, $f' \leq_a^S f$, $\mathit{rank}(f) = \mathit{rank}(f')$, f' is S'-tight, and
    - $i' = (i + 2) \mod (\mathit{rank}(f') + 1)$ and $O' = f'^{-1}(i')$ if $O = \emptyset$ or
    - $i' = i$ and $O' = \delta(O, a) \cap f'^{-1}(i)$ if $O \neq \emptyset$, and
- $F' = \{\emptyset\} \cup ((2^Q \times \{\emptyset\} \times \mathcal{T} \times \omega) \cap Q_2)$.

Intuitively, Comp$_S$ is an extension of the classical subset construction for determinization of finite automata. In particular, Q_1, δ_1, and I_1 constitute the deterministic finite automaton obtained from $\mathcal{A}$ using the subset construction. The automaton can, however, nondeterministically guess a point at which it will make a transition to a *macrostate* (S, O, f, i) in the Q_2 part; this guess corresponds to a level in the run DAG of the accepted word from which the ranks of all levels form an S-tight ranking, where the S component of the macrostate is again a subset from the subset construction. In the Q_2 part, $\mathcal{B}_S$ makes sure that in order for a word to be accepted by $\mathcal{B}_S$, all runs of $\mathcal{A}$ over the word need to touch an accepting state only finitely many times. This is ensured by the f component, which, roughly speaking, maps states to ranks of corresponding vertices in the run DAG over the given word. The O component is used for a standard cut-point construction, and is used to make sure that all runs that have reached an accepting state in $\mathcal{A}$ will eventually leave it (this can happen for different runs at a different point). The S, O, and f components were already present in [24]. The i component was introduced by Schewe to improve the complexity of the construction; it is used to cycle over phases, where in each phase we focus on cut-points of a different rank. See [33] for a more elaborate exposition.

Proposition 1 (Corollary 3.3 in [33]). $\mathcal{L}(\mathcal{B}_S) = \overline{\mathcal{L}(\mathcal{A})}$.

4 Purging Macrostates with Incompatible Rankings

Our first optimisation is based on removing from $\mathcal{B}_S$ macrostates $(S, O, f, i) \in Q_2$ whose level ranking f assigns some states of S an unnecessarily high rank.

Intuitively, when S contains a state p and a state q such that p is (directly) simulated by q, i.e. $p \preceq_{di} q$, then $f(p)$ needs to be at most $f(q)$. This is because in any word α and its run DAG $\mathcal{G}_\alpha$ in $\mathcal{A}$, if p and q are at the same level i of $\mathcal{G}_\alpha$, then the ranks of their vertices v_p and v_q at the given level are either both ω (when $\alpha \in \mathcal{L}(\mathcal{A})$), or such that $rank_\alpha(v_p) \leq rank_\alpha(v_q)$ otherwise. This is because, intuitively, the DAG rooted in v_p in $\mathcal{G}_\alpha$ is isomorphic to a subgraph of the DAG rooted in v_q.

Formally, consider the following predicate on macrostates of $\mathcal{B}_S$:

$$\mathcal{P}_{di}(S, O, f, i) \quad \text{iff} \quad \exists p, q \in S : p \preceq_{di} q \wedge f(p) > f(q). \tag{1}$$

We modify Comps to purge macrostates that satisfy $\mathcal{P}_{di}$. That is, we create a new procedure Purge_{di} obtained from Comps by modifying the definition of $\mathcal{B}_S$ such that all occurrences of Q_2 are substituted by Q_2^{di} and

$$Q_2^{di} = Q_2 \setminus \{(S, O, f, i) \in Q_2 \mid \mathcal{P}_{di}(S, O, f, i)\}. \tag{2}$$

We denote the BA obtained from Purge_{di} as $\mathcal{B}_S^{di}$. The following lemma, proved in Sect. 4.1 states the correctness of this construction.

Lemma 2. $\mathcal{L}(\mathcal{B}_S^{di}) = \mathcal{L}(\mathcal{B}_S)$

The following natural question arises: Is it possible to extend the purging technique from direct simulation to other notions of simulation? For *fair* simulation, this cannot be done. The reason is that, for a pair of states p and q s.t. $p \preceq_f q$, it can happen that for a word $\beta \in \Sigma^\omega$, there can be a trace from p over β that finitely many times touches an accepting state (i.e., a vertex of p in the corresponding run DAG can have any rank between 0 and $2n$), while all traces from q over β can completely avoid touching any accepting state. From the point of view of fair simulation, these are both unfair traces, and, therefore, disregarded.

On the other hand, *delayed* simulation—which is often much richer than direct simulation—can be used, with a small change. Intuitively, the delayed simulation can be used because $p \preceq_{de} q$ guarantees that on every level of trees in $\mathcal{G}_\alpha$ rooted in v_p and in v_q respectively, the rank of the vertex v_p is at most by one larger than the rank of vertex v_q (or by any number smaller). Formally, let $\mathcal{P}_{de}$ be the following predicate on macrostates of $\mathcal{B}_S$:

$$\mathcal{P}_{de}(S, O, f, i) \quad \text{iff} \quad \exists p, q \in S : p \preceq_{de} q \wedge f(p) > [\![f(q)]\!], \tag{3}$$

where $[\![x]\!]$ for $x \in \omega$ denotes the smallest even number greater or equal to x and $[\![\omega]\!] = \omega$. Similarly as above, we create a new procedure, called Purge_{de}, which is obtained from Comps by modifying the definition of $\mathcal{B}_S$ such that all occurrences of Q_2 are substituted by Q_2^{de} and

$$Q_2^{de} = Q_2 \setminus \{(S, O, f, i) \in Q_2 \mid \mathcal{P}_{de}(S, O, f, i)\}. \tag{4}$$

We denote the BA obtained from Purge_{de} as $\mathcal{B}_S^{de}$.

Lemma 3. $\mathcal{L}(\mathcal{B}_S^{de}) = \mathcal{L}(\mathcal{B}_S)$

The use of $\llbracket f(q) \rrbracket$ in $\mathcal{P}_{de}$ results in the fact that the two purging techniques are incomparable. For instance, consider a macrostate $(\{p, q\}, \emptyset, \{p \mapsto 2, q \mapsto 1\}, 0)$ such that $p \preceq_{di} q$ and $p \preceq_{de} q$. Then the macrostate will be purged in $\textsc{Purge}_{di}$, but not in $\textsc{Purge}_{de}$.

The two techniques can, however, be easily combined into a third procedure $\textsc{Purge}_{di+de}$, when Q_2 is substituted in $\textsc{Comp}_{\mathrm{S}}$ with Q_2^{di+de} defined as

$$Q_2^{di+de} = Q_2 \setminus \{(S, O, f, i) \in Q_2 \mid \mathcal{P}_{di}(S, O, f, i) \vee \mathcal{P}_{de}(S, O, f, i)\}. \tag{5}$$

We denote the resulting BA as $\mathcal{B}_S^{di+de}$.

Lemma 4. $\mathcal{L}(\mathcal{B}_S^{di+de}) = \mathcal{L}(\mathcal{B}_S)$

4.1 Proofs of Lemmas 2, 3, and 4

We first give a lemma that an x-strategy σ_x preserves an x-simulation $\preceq_x$.

Lemma 5. *Let $\preceq_x$ be an x-simulation (for $x \in \{di, de, f\}$). Then, the following holds: $\forall p, q \in Q : p \preceq_x q \wedge p \xrightarrow{a} p' \in \delta \Rightarrow \exists q' \in Q : q \xrightarrow{a} q' \in \delta \wedge p' \preceq_x q'$.*

Proof. Let $p, q \in Q$ such that $p \preceq_x q$ and $p \xrightarrow{a} p' \in \delta$, and let π_p be a trace starting from p with the first transition $p \xrightarrow{a} p'$. From the definition of x-simulation, there is a winning Duplicator strategy σ_x; let $\pi_q = \sigma_x(q', \pi_p)$ and let $q \xrightarrow{a} q'$ be the first transition of π_q. Let $\pi_{p'}$ and $\pi_{r'}$ be traces obtained from π_p and π_r by removing their first transitions. It is easy to see that if $\mathcal{C}^x(\pi_p, \pi_r)$ then also $\mathcal{C}^x(\pi_{p'}, \pi_{r'})$ for any $x \in \{di, de, f\}$. It follows that σ_x is also a winning Duplicator strategy from (p', r'). □

Next, we focus on delayed simulation and the proof of Lemma 3. In the next lemma, we show that if there is a pair of vertices on some level of the run DAG where one vertex delay-simulates the other one, there exists a relation between their rankings. This will be used to purge some useless rankings from the complemented BA.

Lemma 6. *Let $p, q \in Q$ such that $p \preceq_{de} q$ and $\mathcal{G}_\alpha = (V, E)$ be the run DAG of $\mathcal{A}$ over α. For all $i \geq 0$, it holds that $(p, i) \in V \wedge (q, i) \in V \Rightarrow rank_\alpha(p, i) \leq \llbracket rank_\alpha(q, i) \rrbracket$.*

Proof. Consider some $(p, i) \in V$ and $(q, i) \in V$. First, suppose that $rank_\alpha(q, i) = \omega$. Since the rank can be at most ω, it will always hold that $rank_\alpha(p, i) \leq \llbracket rank_\alpha(q, i) \rrbracket$.

On the other hand, suppose that $rank_\alpha(q, i)$ is finite, i.e., $\alpha_{i:\omega}$ is not accepted by q. Then, due to Lemma 1, $0 \leq rank_\alpha(q, i) \leq 2n$. Because $p \preceq_{de} q$, it holds that $\alpha_{i:\omega}$ is also not accepted by p, and therefore also $0 \leq rank_\alpha(p, i) \leq 2n$. We now need to show that $0 \leq rank_\alpha(p, i) \leq \llbracket rank_\alpha(q, i) \rrbracket \leq 2n$.

Let $\{\mathcal{G}_\alpha^k\}_{k=0}^{2n+1}$ be the sequence of run DAGs obtained from $\mathcal{G}_\alpha$ in the ranking procedure from Sect. 2.2. In the following text we use the abbreviation $v \in \mathcal{G}_\alpha^m \setminus \mathcal{G}_\alpha^n$ for $v \in \mathcal{G}_\alpha^m \wedge v \notin \mathcal{G}_\alpha^n$. Since the rank of a node (r, j) is given as the number l s.t. $(r, j) \in \mathcal{G}_\alpha^l \setminus \mathcal{G}_\alpha^{l+1}$, we will finish the proof of this lemma by proving the following claim:

Claim. Let k and l be s.t. $(p,i) \in \mathcal{G}_\alpha^k \setminus \mathcal{G}_\alpha^{k+1}$ and $(q,i) \in \mathcal{G}_\alpha^l \setminus \mathcal{G}_\alpha^{l+1}$. Then $k \leq [\![l]\!]$.

Proof: We prove the claim by induction on l.

- Base case: $(l=0)$ Since we assume $\mathcal{A}$ is complete, no vertex in $\mathcal{G}_\alpha^0$ is finite.
  $(l=1)$ We prove that if (q,i) is endangered in $\mathcal{G}_\alpha^1$, then (p,i) is endangered in $\mathcal{G}_\alpha^1$ as well (so both would be removed in $\mathcal{G}_\alpha^2$). For the sake of contradiction, assume that (q,i) is endangered in $\mathcal{G}_\alpha^1$ and (p,i) is not. Therefore, since $\mathcal{G}_\alpha^1$ contains no finite vertices, there is an infinite path π from (p,i) s.t. π contains at least one accepting state. In the following, we abuse notation and, given a strategy σ_{de} and a state $s \in Q$, use $\sigma_{de}((s,i),\pi)$ to denote the path $(s_0,i)(s_1,i+1)(s_2,i+2)\ldots$ such that $s_0 = s$ and $\forall j \geq 0$, it holds that $s_{j+1} = \sigma_{de}(s_j, r_{i+j} \xrightarrow{\alpha_{i+j}} r_{i+j+1})$ where $\pi_x = (r_x, x)$ for every $x \geq 0$. Since $p \preceq_{de} q$, there is a corresponding infinite path $\pi' = \sigma_{de}((q,i),\pi)$ that also contains at least one accepting state. Therefore, (q,i) is not endangered, a contradiction to the assumption, so we conclude that $l=1 \Rightarrow k=1$.
- Inductive step: We assume the claim holds for all $l < 2j$ and prove the inductive step for even and odd steps independently.
  $(l=2j)$ We prove that if (q,i) is finite in $\mathcal{G}_\alpha^l$ (and therefore would be removed in $\mathcal{G}_\alpha^{l+1}$), then either $(p,i) \notin \mathcal{G}_\alpha^l$, or (p,i) is also finite in $\mathcal{G}_\alpha^l$. For the sake of contradiction, we assume that (q,i) is finite in $\mathcal{G}_\alpha^l$ and that (p,i) is in $\mathcal{G}_\alpha^l$, but is not finite there (and, therefore, $k > l$). Since (p,i) is not finite in $\mathcal{G}_\alpha^l$, there is an infinite path π from (p,i) in $\mathcal{G}_\alpha^l$. Because $p \preceq_{de} q$, it follows that there is an infinite path $\pi' = \sigma_{de}((q,i),\pi)$ in $\mathcal{G}_\alpha^0$ (π' is not in $\mathcal{G}_\alpha^l$ because (q,i) is finite there). Using Lemma 5 (possibly multiple times) and the fact that (q,i) is finite, we can find vertices (p',x) in π and (q',x) in π' s.t. $p' \preceq_{de} q'$ and (q',x) is not in $\mathcal{G}_\alpha^l$, therefore, $(q',x) \in \mathcal{G}_\alpha^e \setminus \mathcal{G}_\alpha^{e+1}$ for some $e < l$. Because $(p',x) \in \mathcal{G}_\alpha^l$ and it is not finite (π is infinite), it follows that $(p',x) \in \mathcal{G}_\alpha^f \setminus \mathcal{G}_\alpha^{f+1}$ for some $f > l$, and since $e < l < f$, we have that $f \not\leq e+1$, implying $f \not\leq [\![e]\!]$, which is in contradiction to the induction hypothesis.
  $(l=2j+1)$ We prove that if (q,i) is endangered in $\mathcal{G}_\alpha^l$ (and therefore would be removed in $\mathcal{G}_\alpha^{l+1}$), then either $(p,i) \notin \mathcal{G}_\alpha^l$, or (p,i) is removed at the latest in $\mathcal{G}_\alpha^{l+1}$. For the sake of contradiction, assume that (q,i) is endangered in $\mathcal{G}_\alpha^l$ while (p,i) is removed later than in $\mathcal{G}_\alpha^{l+1}$. Therefore, since $\mathcal{G}_\alpha^l$ contains no finite vertices (they were removed in the $(l-1)$-th step), there is an infinite path π from (p,i) s.t. π contains at least one accepting state. Because $p \preceq_{de} q$, there is a corresponding path $\pi' = \sigma_{de}((q,i),\pi)$ from (q,i) in $\mathcal{G}_\alpha^0$ that also contains at least one accepting state and moreover $\pi' \notin \mathcal{G}_\alpha^l$. Since π' has an infinite number of states (and at least one accepting), not all states from π' were removed in $\mathcal{G}_\alpha^{l-1}$, i.e., there is at least one node with rank less or equal to $l-2$. Using Lemma 5 (also possibly multiple times) we can hence find states (p',x) in π and (q',x) in π' s.t. $p' \preceq_{de} q'$ and (q',x) is not in $\mathcal{G}_\alpha^l$ and has a rank less or equal to $l-2$, therefore, $(q',x) \in \mathcal{G}_\alpha^e \setminus \mathcal{G}_\alpha^{e+1}$ for some $e < l-1$. Because $(p',x) \in \mathcal{G}_\alpha^l$, it follows that $(p',x) \in \mathcal{G}_\alpha^f \setminus \mathcal{G}_\alpha^{f+1}$ for some $f \geq l$, and, therefore, $f \not\leq e+1$, which is in contradiction to the induction hypothesis. ■

This concludes the proof. □

Lemma 7. *Let $p, q \in Q$ such that $p \preceq_{di} q$ and $\mathcal{G}_\alpha = (V, E)$ be the run DAG of $\mathcal{A}$ over α. For all $i \geq 0$, it holds that $(p, i) \in V \wedge (q, i) \in V \Rightarrow rank_\alpha(p, i) \leq rank_\alpha(q, i)$.*

Proof. Can be obtained as a simplified version of the proof of Lemma 6. □

We are now ready to prove Lemma 3.

Lemma 3. $\mathcal{L}(\mathcal{B}_S^{de}) = \mathcal{L}(\mathcal{B}_S)$

Proof. ($\subseteq$) Follows directly from the fact that $\mathcal{B}_S^{de}$ is obtained by removing states from $\mathcal{B}_S$.
($\supseteq$) Let $\alpha \in \mathcal{L}(\mathcal{B}_S)$. As shown in the proof of Lemma 3.2 in [33], there are two cases. The first case is when all vertices of $\mathcal{G}_\alpha$ are finite, which we do not need to consider, since we assume complete automata.
The other case is when $\mathcal{G}_\alpha$ contains an infinite vertex. In this case, $\mathcal{B}_S$ contains an accepting run

$$\rho = S_0 S_1 \ldots S_p (S_{p+1}, O_{p+1}, f_{p+1}, i_{p+1})(S_{p+2}, O_{p+2}, f_{p+2}, i_{p+2}) \ldots$$

with

- $S_0 = I, O_{p+1} = \emptyset$, and $i_{p+1} = 0$,
- $S_{j+1} = \delta(S_j, \alpha_j)$ for all $j \in \omega$,

and, for all $j > p$,

- $O_{j+1} = f_{j+1}^{-1}(i_{j+1})$ if $O_j = \emptyset$ or
  $O_{j+1} = \delta(O_j, \alpha_j) \cap f_{j+1}^{-1}(i_{j+1})$ if $O_j \neq \emptyset$, respectively,
- f_j is the S_j-tight level ranking that maps each $q \in S_j$ to the rank of $(q, j) \in \mathcal{G}_\alpha$,
- $i_{j+1} = i_j$ if $O_j \neq \emptyset$ or
  $i_{j+1} = (i_j + 2) \mod (rank(f) + 1)$ if $O_j = \emptyset$, respectively.

The ranks assigned by f_j to states of S_j match the ranks of the corresponding vertices in $\mathcal{G}_\alpha$.
⊛ Using Lemma 6, we conclude that ρ contains no macrostate (S, O, f, j) where $f(p) > \llbracket f(q) \rrbracket$ and $p \preceq_{de} q$ for $p, q \in S$. Therefore, ρ is also an accepting run in $\mathcal{B}_S^{de}$. (We use ⊛ to refer to this paragraph later.) □

Lemma 2. $\mathcal{L}(\mathcal{B}_S^{di}) = \mathcal{L}(\mathcal{B}_S)$

Proof. The same as for Lemma 3 with ⊛ substituted by the following:
⊛ Using Lemma 7, we conclude that ρ contains no macrostate (S, O, f, j) where $f(p) > f(q)$ and $p \preceq_{di} q$ for $p, q \in S$. So ρ is also an accepting run in $\mathcal{B}_S^{di}$. □

Lemma 4. $\mathcal{L}(\mathcal{B}_S^{di+de}) = \mathcal{L}(\mathcal{B}_S)$

Proof. The same as for Lemma 3 with ⊛ substituted by the following:
⊛ Using Lemmas 7 and 6, we conclude that ρ contains no macrostate (S, O, f, j) where either $f(p) > f(q)$ and $p \preceq_{di} q$, or $f(p) > \llbracket f(q) \rrbracket$ and $p \preceq_{de} q$ for $p, q \in S$. Therefore, ρ is also an accepting run in $\mathcal{B}_S^{di+de}$. □

5 Saturation of Macrostates

Our second optimisation is inspired by an optimisation of determinisation of classical finite automata from [17, Section 5]. Their optimisation is based on saturating every constructed macrostate in the classical subset construction with all direct-simulation-smaller states. This can reduce the total number of states of the determinized automaton because two or more macrostates can be mapped to a single saturated macrostate. (In Sect. 5.2, we show why an analogue of their *compression* cannot be used.)

We show that a similar technique can be applied to BAs. We do not restrain ourselves to direct simulation, though, and generalize the technique to delayed simulation. In particular, in our optimisation, we saturate the S components of macrostates (S, O, f, i) obtained in COMPS with all $\preceq_{de}$-smaller states. Formally, we modify COMPS by substituting the definition of the constructed transition function δ' with δ'_{Sat} defined as follows:

- $\delta'_{Sat} = \delta_1^{Sat} \cup \delta_2^{Sat} \cup \delta_3^{Sat}$ where
  - $\delta_1^{Sat} : Q_1 \times \Sigma \to 2^{Q_1}$ with $\delta_1^{Sat}(S, a) = \{cl[\delta(S, a)]\}$,
  - $\delta_2^{Sat} : Q_1 \times \Sigma \to 2^{Q_2}$ with $\delta_2^{Sat}(S, a) = \{(S', \emptyset, f, 0) \mid S' = cl[\delta(S, a)]\}$, and
  - $\delta_3^{Sat} : Q_2 \times \Sigma \to 2^{Q_2}$ with $(S', O', f', i') \in \delta_3^{Sat}((S, O, f, i), a)$ iff $S' = cl[\delta(S, a)], f' \leq_a^S f$, $rank(f) = rank(f')$, and
    - $i' = (i + 2) \mod (rank(f') + 1)$ and $O' = f'^{-1}(i')$ if $O = \emptyset$ or
    - $i' = i$ and $O' = \delta(O, a) \cap f'^{-1}(i)$ if $O \neq \emptyset$,

where $cl[S] = \{q \in Q \mid \exists s \in S : q \preceq_{de} s\}$. We denote the obtained procedure as SATURATE and the obtained BA as $\mathcal{B}_{Sat}$.

Lemma 8. $\mathcal{L}(\mathcal{B}_{Sat}) = \mathcal{L}(\mathcal{B}_S)$

Obviously, as direct simulation is stronger than delayed simulation, the previous technique can also use direct simulation only (e.g., when computing the full delayed simulation is computationally too demanding). Moreover, SATURATE is also compatible with all PURGE$_x$ algorithms for $x \in \{di, de, di + de\}$ (because they just remove macrostates with incompatible rankings from Q_2)—we call the combined versions PURGE$_x$+SATURATE and the complement BAs they output $\mathcal{B}_{Sat}^x$.

Lemma 9. $\mathcal{L}(\mathcal{B}_{Sat}^{di}) = \mathcal{L}(\mathcal{B}_{Sat}^{de}) = \mathcal{L}(\mathcal{B}_{Sat}^{di+de}) = \mathcal{L}(\mathcal{B}_S)$

5.1 Proofs of Lemmas 8 and 9

We start with a lemma, used later, that talks about languages of states related by delayed simulation when there is a path between them.

Lemma 10. *For $p, q \in Q$ such that $p \preceq_{de} q$, let $L_\top = \{w \in \Sigma^* \mid p \overset{w}{\underset{F}{\rightsquigarrow}} q\}$ and $L_\perp = \{w \in \Sigma^* \mid p \overset{w}{\rightsquigarrow} q\}$. Then $L(q) \supseteq (L_\perp^* L_\top)^\omega$.*

Proof. First we prove the following claim:

Claim. For every word $\alpha = w_0 w_1 w_2 \cdots \in \Sigma^\omega$ where $w_i \in L_\top \cup L_\bot$, we can construct a trace $\pi = p \overset{w_0}{\rightsquigarrow} q_0 \overset{w_1}{\rightsquigarrow} q_1 \overset{w_2}{\rightsquigarrow} \cdots$ over α such that $p \preceq_{de} q_0$ and $q_i \preceq_{de} q_{i+1}$ for all $i \geq 0$.

Proof: We assign $q_0 := q$ and construct the rest of π by the following inductive construction.

- Base case: ($i = 0$) From the assumption it holds that $p \overset{w_1}{\rightsquigarrow} q_0$ and $p \preceq_{de} q_0$. From Lemma 5 there is some $r \in Q$ s.t. $q_0 \overset{w_1}{\rightsquigarrow} r$ and $q_0 \preceq_{de} r$. We assign $q_1 := r$, so $q_0 \preceq_{de} q_1$.
- Inductive step: Let $\pi' = p \overset{w_0}{\rightsquigarrow} q_0 \overset{w_1}{\rightsquigarrow} \cdots \overset{w_i}{\rightsquigarrow} q_i$ be a prefix of a trace such that $q_j \preceq_{de} q_{j+1}$ for every $j < i$. From the transitivity of $\preceq_{de}$, it follows that $p \preceq_{de} q_i$. From Lemma 5 there is some $r \in Q$ s.t. $q_i \overset{w_i}{\rightsquigarrow} r$ and $q \preceq_{de} r$. We assign $q_{i+1} := r$, so $q_i \preceq_{de} q_{i+1}$. ■

Consider a word $\alpha \in (L_\bot^* L_\top)^\omega$ such that $\alpha = w_0 w_1 w_2 \ldots$ for $w_i \in L_\top \cup L_\bot$. We show that $\alpha \in \mathcal{L}(q)$. According to the previous claim, we can construct a trace $\pi = p \overset{w_0}{\rightsquigarrow} q = q_0 \overset{w_1}{\rightsquigarrow} q_1 \overset{w_2}{\rightsquigarrow} \cdots$ over α s.t. $p \preceq_{de} q_0$ and $q_i \preceq_{de} q_{i+1}$ for all $i \geq 0$. Since $p \preceq_{de} q$, from Lemma 5 it follows that we can construct a trace $\pi' = q \overset{w_0}{\rightsquigarrow} r_0 \overset{w_1}{\rightsquigarrow} r_1 \overset{w_2}{\rightsquigarrow} \cdots$ s.t. $q_i \preceq_{de} r_i$ for every $i \geq 0$. Because α contains infinitely often a subword from $L_\top$, there is some $\ell \in \omega$ such that $q_\ell \overset{w_\ell}{\rightsquigarrow} q_{\ell+1}$ and $r_\ell \overset{w_\ell}{\rightsquigarrow} r_{\ell+1}$ for $w_\ell \in L_\top$. Note that it holds that $p \preceq_{de} q_\ell \preceq_{de} r_\ell$. We can again use the claim above to construct a trace $\pi^\star = p \underset{F}{\overset{w_\ell}{\rightsquigarrow}} q = s_0 \overset{w_{\ell+1}}{\rightsquigarrow} s_1 \overset{w_{\ell+2}}{\rightsquigarrow} \cdots$ over $\alpha_\ell = w_\ell w_{\ell+1} w_{\ell+2} \ldots$ such that $p \preceq_{de} s_0$ and $s_i \preceq_{de} s_{i+1}$ for all $i \geq 0$. Since $p \preceq_{de} r_\ell$, we can simulate $\pi^\star$ from r_ℓ by a trace $\pi^{\star\prime}$, and because $p \underset{F}{\overset{w_\ell}{\rightsquigarrow}} q$, we know that $\pi^{\star\prime}$ will touch an accepting state in finitely many steps (this holds because w_ℓ is from $L_\top$, which are the words over which we can go from p to q and touch an accepting state). Consider $m \geq \ell$ such that s_m is the first state after the accepting state that is one of the $\{s_0, s_1, \ldots\}$ in $\pi^{\star\prime}$. This reasoning could be repeated for all occurrences of a subword from $L_\top$ in $\pi^\star$, therefore $\alpha \in \mathcal{L}(q)$. □

Next, we give a lemma used for establishing correctness of saturating macrostates with $\preceq_{de}$-smaller states.

Lemma 11. *Let $p, q, r \in Q$ such that $r \xrightarrow{a} q \in \delta$ and $p \preceq_{de} q$. Further, let $\mathcal{A}' = (Q, \delta', I, F)$ where $\delta' = \delta \cup \{r \xrightarrow{a} p\}$. Then $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{A}')$.*

Proof. ($\subseteq$) Clear.
($\supseteq$) Consider some $\alpha \in \mathcal{L}(\mathcal{A}')$ and an accepting trace π in $\mathcal{A}'$ over α. There are two cases:

1. (π contains only finitely many transitions $r \xrightarrow{a} p$)
   In this case, π is of the form $\pi = \pi_i \pi_\omega$ where π_i is a finite prefix $\pi_i = q_0 \overset{w_0}{\rightsquigarrow} r \xrightarrow{a} p \overset{w_1}{\rightsquigarrow} r \xrightarrow{a} p \overset{w_2}{\rightsquigarrow} \cdots \overset{w_n}{\rightsquigarrow} r \xrightarrow{a} p$, for $q_0 \in I$, and π_ω is an infinite trace from p that does not contain any occurrence of the transition $r \xrightarrow{a} p$. We construct

in $\mathcal{A}$ a trace $\pi' = q_0 \overset{w_0}{\rightsquigarrow} r \xrightarrow{a} q \overset{w_1}{\rightsquigarrow} r_1 \xrightarrow{a} q_1 \overset{w_2}{\rightsquigarrow} \cdots \overset{w_n}{\rightsquigarrow} r_n \xrightarrow{a} q_n.\pi'_\omega$ as follows. Let σ_{de} be a strategy for $\preceq_{de}$. We set $r_1 := \sigma_{de}(q, p \overset{w_1}{\rightsquigarrow} r)$, so $r \preceq_{de} r_1$. Since $r \xrightarrow{a} q \in \delta$, it follows that there is $r_1 \xrightarrow{a} q_1 \in \delta$ such that $p \preceq_{de} q_1$. For $i > 1$, we set $r_i := \sigma_{de}(q_{i-1}, p \overset{w_i}{\rightsquigarrow} r)$. By induction, it follows that $\forall 1 \leq i \leq n : p \preceq_{de} q_i$, in particular $p \preceq_{de} q_n$. We set $\pi'_\omega := \sigma_{de}(q_n, \pi_\omega)$. Since π_ω starts in p and contains infinitely many accepting states and π'_ω starts in q_n and $p \preceq_{de} q_n$, then π'_ω also contains infinitely many accepting states. It follows that π' is accepting, so $\alpha \in \mathcal{L}(\mathcal{A})$.
2. (π contains infinitely many transitions $r \xrightarrow{a} p$)
In this case, π is of the form $\pi = q_0 \overset{w_0}{\rightsquigarrow} r \xrightarrow{a} p \overset{w_1}{\rightsquigarrow} r \xrightarrow{a} p \overset{w_2}{\rightsquigarrow} \cdots \overset{w_n}{\rightsquigarrow} r \xrightarrow{a} p \overset{w_\omega}{\rightsquigarrow} \cdots$, for $q_0 \in I$ and $\alpha = w_0 a w_1 a w_2 \ldots$ Since π is accepting, for infinitely many $i \in \omega$, we have $p \underset{F}{\overset{w_i a}{\rightsquigarrow}} p$ in $\mathcal{A}'$ and hence also $p \underset{F}{\overset{w_i a}{\rightsquigarrow}} q$ in the original BA $\mathcal{A}$. Using Lemma 10 and the fact that $p \preceq_{de} q$, we have $w_1 a w_2 a \cdots \in L(q)$ and hence $\alpha = w_0 a w_1 a w_2 a \cdots \in \mathcal{L}(\mathcal{A})$. □

The following lemma guarantees that adding transitions in the way of Lemma 11 does not break the computed delayed simulation and can, therefore, be performed repeatedly, without the need to recompute the simulation.

Lemma 12. *Let $\preceq_{de}$ be the delayed simulation on $\mathcal{A}$. Further, let $p, q, r \in Q$ be such that $r \xrightarrow{a} q \in \delta$ and $p \preceq_{de} q$, and let $\mathcal{A}' = (Q, \delta', I, F)$ where $\delta' = \delta \cup \{r \xrightarrow{a} p\}$. Then $\preceq_{de}$ is included in the delayed simulation on $\mathcal{A}'$.*

Proof. Let σ_{de} be a dominating strategy compatible with $\preceq_{de}$ and σ'_{de} be a strategy defined for all $s \in Q$ such that $r \preceq_{de} s$ as $\sigma'_{de}(s, x) = \sigma_{de}(s, x)$ when $x \neq (r \xrightarrow{a} p)$ and $\sigma'_{de}(s, r \xrightarrow{a} p) = \sigma_{de}(s, r \xrightarrow{a} q)$. Note that σ'_{de} is also dominating wrt $\preceq_{de}$. This can be shown by the following proof by contradiction: Suppose σ'_{de} is not dominating; then there is a strategy ρ such that $\sigma'_{de}(s, r \xrightarrow{a} p)$ must be simulated by $\rho(s, r \xrightarrow{a} p) = t$. But then $\sigma_{de}(s, r \xrightarrow{a} q)$ must also (transitivity of simulation) be simulated by t, so σ_{de} is not dominating. Contradiction.

Further, let $t, u \in Q$ be such that $t \preceq_{de} u$. Let $\pi_t = t \overset{w_1}{\rightsquigarrow} t_f \overset{w_2}{\rightsquigarrow} r \xrightarrow{a} p.\pi'_t$ be a trace over $\alpha = w_1 w_2 a w_\omega \in \Sigma^\omega$ in $\mathcal{A}'$ such that t_f is an accepting state and $t_f \overset{w_2}{\rightsquigarrow} r$ does not contain any occurrence of $r \xrightarrow{a} p$. Further, let $\pi_u = u_0 \overset{w_1}{\rightsquigarrow} u_f \overset{w_2}{\rightsquigarrow} u_i \xrightarrow{a} u_{i+1}.\pi'_u$ be a trace corresponding to a run $u_0 u_1 u_2 \ldots$ over α in $\mathcal{A}$, where $u_0 = u$, constructed as $\pi_u = \sigma'_{de}(u, \pi_t)$.

Claim. There is a trace $\pi_v = t \overset{w_1}{\rightsquigarrow} v_f.\pi'_v$ over α such that π'_v contains an accepting state and π_v is $\preceq_{de}$-simulated by π_u at every position.

Proof: We have the following two cases:

- ($t \overset{w_1}{\rightsquigarrow} t_f$ does not contain any occurrence of $r \xrightarrow{a} p$)
Let $\pi_v = t \overset{w_1}{\rightsquigarrow} t_f \overset{w_2}{\rightsquigarrow} r \xrightarrow{a} q.\pi'_v$ be a trace in $\mathcal{A}$ over α obtained from π_t by starting with its prefix up to r, taking $r \xrightarrow{a} q$, and continuing with $\pi'_v = \sigma'_{de}(q, \pi'_t)$. Since in π_v, it holds that t_f is at the same position as t_f in π_t, the first part of the claim holds. Further, π_u clearly $\preceq_{de}$-simulates π_v on

$t \overset{w_1}{\rightsquigarrow} t_f \overset{w_2}{\rightsquigarrow} r$, and because σ'_{de} simulates $r \xrightarrow{a} p$ by a transition to a state u_{i+1} such that $q \preceq_{de} u_{i+1}$ and π'_v is constructed using σ'_{de}, then also the second part of the claim holds.

– ($t \overset{w_1}{\rightsquigarrow} t_f$ contains at least one occurrence of $r \xrightarrow{a} p$)
Suppose that π_t starts with $t \overset{w_{11}}{\rightsquigarrow} r \xrightarrow{a} p \overset{w_{12}}{\rightsquigarrow} t_f$ such that $t \overset{w_{11}}{\rightsquigarrow} r$ does not contain any $r \xrightarrow{a} p$. Then let us start building π_v such that it starts with $t \overset{w_{11}}{\rightsquigarrow} r \xrightarrow{a} q$. On this prefix, π_v is clearly $\preceq_{de}$-simulated by the corresponding prefix of π_u. We continue from q using the strategy σ'_{de}. In particular, the next time we reach $r \xrightarrow{a} p$ in π_t while we are at some state v_1 such that $r \preceq_{de} v_1$, we simulate the transition by $\sigma'_{de}(v_1, r \xrightarrow{a} p)$ and so on. We can observe that when we arrive to t_f in π_t, we also arrive to v_f in π_v such that $t_f \preceq_{de} v_f$. Therefore, π'_v contains an accepting state. Moreover, since σ'_{de} is dominating, the second part of the claim also holds. ■

From the claim above, it follows that the trace $u_f \overset{w_2}{\rightsquigarrow} u_i \xrightarrow{a} u_{i+1}.\pi'_u$ contains an accepting state, so $\mathcal{C}^{de}(\pi_t, \pi_u)$. □

Finally, we are ready to prove Lemma 8.

Lemma 8. $\mathcal{L}(\mathcal{B}_{Sat}) = \mathcal{L}(\mathcal{B}_S)$

Proof. ($\subseteq$) Let $\alpha \in \mathcal{L}(\mathcal{B}_{Sat})$ and ρ be an arbitrary accepting run over α in $\mathcal{B}_{Sat}$ such that $\rho = S_0 S_1 \ldots S_{n-1}(S_n, O_n, f_n, i_n)(S_{n+1}, O_{n+1}, f_{n+1}, i_{n+1}) \ldots$. For the sake of contradiction, assume that $\alpha \in \mathcal{L}(\mathcal{A})$, therefore, there is a run ρ' on α in $\mathcal{A}$ having infinitely many accepting states. From the fact that tight level rankings form a non-increasing sequence, we have that $f_n(\rho'(n)) \geq f_{n+1}(\rho'(n+1)) \geq \cdots$. This sequence eventually stabilizes and from the property of level rankings and the fact that ρ' is accepting, it stabilizes in some ℓ such that $f_\ell(\rho'(\ell))$ is even. This, however, means that the O component of macrostates in ρ cannot be emptied infinitely often, and, therefore, ρ is not accepting, which is a contradiction. Hence $\alpha \notin \mathcal{L}(\mathcal{A})$, so (from Proposition 1) $\alpha \in \mathcal{L}(\mathcal{B}_S)$.
($\supseteq$) Consider some $\alpha \in \mathcal{L}(\mathcal{B}_S)$. Let $\mathcal{A}'$ be a BA obtained from $\mathcal{A}$ by adding transitions according to Lemma 12. Then from Lemma 11, we have that $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{A}')$. Therefore, $\alpha \in \mathcal{L}(\mathcal{B}'_S)$ where $\mathcal{B}'_S$ is the BA obtained from $\mathcal{A}'$ using COMPS. It is easy to see that we can construct a run in $\mathcal{B}_{Sat}$ that mimics the levels of run DAG of α in $\mathcal{A}'$ (i.e., we are able to empty the O component infinitely often). Hence $\alpha \in \mathcal{L}(\mathcal{B}_{Sat})$. □

Lemma 9. $\mathcal{L}(\mathcal{B}^{di}_{Sat}) = \mathcal{L}(\mathcal{B}^{de}_{Sat}) = \mathcal{L}(\mathcal{B}^{di+de}_{Sat}) = \mathcal{L}(\mathcal{B}_S)$

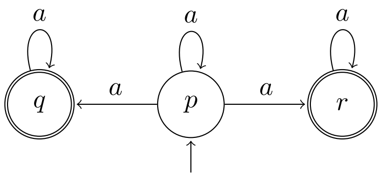
Proof. ($\subseteq$) This part is the same as in the proof of Lemma 8.
($\supseteq$) Consider some $\alpha \in \mathcal{L}(\mathcal{B}_S)$. Let $\mathcal{A}'$ be a BA obtained from $\mathcal{A}$ by adding transitions according to Lemma 12. Then from Lemma 11, we have that $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{A}')$. Therefore, $\alpha \in \mathcal{L}(\mathcal{B}'_S)$ where $\mathcal{B}'_S$ is the BA obtained from $\mathcal{A}'$ using COMPS. It is easy to see that we can construct a run in $\mathcal{B}_{Sat}$ that mimics the levels of run DAG of α in $\mathcal{A}'$ (i.e., we are able to empty the O component infinitely often). Using Lemmas 7 and 6, we can conclude that the run contains no macrostate

of the form (S, O, f, j), where $f(p) > f(q)$ and $p \preceq_{di} q$, or $f(p) > \llbracket f(q) \rrbracket$ and $p \preceq_{de} q$ for $p, q \in S$. Therefore, ρ is also an accepting run in $\mathcal{B}_{Sat}^{di+de}$. Hence $\alpha \in \mathcal{L}(\mathcal{B}_{Sat}^{di+de})$. □

5.2 Remarks on Compression of Macrostates

An analogy to saturation of macrostates is their compression [17, Section 6], based on removing simulation-smaller states from a macrostate. This is, however, not possible even for direct simulation, as we can see in the following example.

Example 1. Consider the BA over $\Sigma = \{a\}$ given below.



For this BA we have $q \preceq_{di} r$ and $r \preceq_{di} q$. If we compress the macrostates obtained in Comps, there is the following trace in the output automaton:

$$\{p\} \xrightarrow{a} (\{p,q\}, \emptyset, \{p \mapsto 3, q \mapsto 2, r \mapsto 1\}, 0) \xrightarrow{a} (\{p,r\}, \{r\}, \{p \mapsto 3, q \mapsto 1, r \mapsto 2\}, 2)$$
$$\xrightarrow{a} (\{p,q\}, \emptyset, \{p \mapsto 3, q \mapsto 2, r \mapsto 1\}, 2) \xrightarrow{a} (\{p,r\}, \{r\}, \{p \mapsto 3, q \mapsto 1, r \mapsto 2\}, 0)$$
$$\xrightarrow{a} (\{p,q\}, \emptyset, \{p \mapsto 3, q \mapsto 2, r \mapsto 1\}, 0) \xrightarrow{a} \cdots$$

This trace contains infinitely many final states (we flush the O-set infinitely often), hence we are able to accept the word a^ω, which is, however, in the language of the input BA. □

6 Use After Simulation Quotienting

In this short section, we establish that our optimizations introduced in Sects. 4 and 5 can be applied with no additional cost in the setting when BA complementation is preceded with simulation-based reduction of the input BA (which is usually helpful), i.e., when the simulation is already computed beforehand for another purpose. In particular, we show that simulation-based reduction preserves the simulation (when naturally extended to the quotient automaton). First, let us formally define the operation of quotienting.

Given an x-simulation $\preceq_x$ for $x \in \{di, de\}$, we use $\approx_x$ to denote the x-*similarity* relation (i.e., the symmetric fragment) $\approx_x = \preceq_x \cap \preceq_x^{-1}$. Note that since $\preceq_x$ is a preorder, it holds that $\approx_x$ is an equivalence. We use $[q]_x$ to denote the equivalence class of q wrt $\approx_x$. The *quotient* of a BA $\mathcal{A} = (Q, \delta, I, F)$ wrt $\approx_x$ is the automaton

$$\mathcal{A}/{\approx_x} = (Q/{\approx_x}, \delta_{\approx_x}, I_{\approx_x}, F_{\approx_x}) \quad (6)$$

with the transition function $\delta_{\approx_x}([q]_x, a) = \{[r]_x \mid r \in \delta([q]_x, a)\}$ and the set of initial and accepting states $I_{\approx_x} = \{[q]_x \in Q/{\approx_x} \mid q \in I\}$ and $F_{\approx_x} = \{[q]_x \in Q/{\approx_x} \mid q \in F\}$ respectively.

Proposition 2 ([7], [12]). *If $x \in \{di, de\}$, then $\mathcal{L}(\mathcal{A}/{\approx_x}) = \mathcal{L}(\mathcal{A})$.*

Remark 1 ([12]). $\mathcal{L}(\mathcal{A}/{\approx_f}) \neq \mathcal{L}(\mathcal{A})$

Finally, the following lemma shows that quotienting preserves direct and delayed simulations, therefore, when complementing $\mathcal{A}$, it is possible to first quotient $\mathcal{A}$ wrt a direct/delayed simulation and then use the same simulation (lifted to the states of the quotient automaton) to optimize the complementation.

Lemma 13. *Let $\preceq_x$ be the x-simulation on $\mathcal{A}$ for $x \in \{di, de\}$. Then the relation $\preceq_x^{\approx}$ defined as $[q]_x \preceq_x^{\approx} [r]_x$ iff $q \preceq_x r$ is the x-simulation on $\mathcal{A}/{\approx_x}$.*

Proof. First, we show that $\preceq_x^{\approx}$ is well defined, i.e., if $q \preceq_x r$, then for all $q' \in [q]_x$ and $r' \in [r]_x$, it holds that $q' \preceq_x r'$. Indeed, this holds because $q' \approx_x q$ and $r \approx_x r$, and therefore $q' \preceq_x q \preceq_x r \preceq_x r'$; the transitivity of simulation yields $q' \preceq_x r'$.

Next, let σ_x be a strategy that gives $\preceq_x$. Consider a trace defined as $[\pi_q]_x = [q_0]_x \xrightarrow{\alpha_0} [q_1]_x \xrightarrow{\alpha_1} \cdots$ over a word $\alpha \in \Sigma^\omega$ in $\mathcal{A}/{\approx_x}$. Then,

1. for $x = di$ there is a trace $\pi_q = q'_0 \xrightarrow{\alpha_0} q'_1 \xrightarrow{\alpha_1} \cdots$ in $\mathcal{A}$ s.t. $q'_0 \in [q_0]_x$ and $q_i \preceq_x q'_i$ for $i \geq 0$. Therefore, if $[q_i]_x$ is accepting then so is q'_i;
2. for $x = de$ there is a trace $\pi_q = q'_0 \xrightarrow{\alpha_0} q'_1 \xrightarrow{\alpha_1} \cdots$ in $\mathcal{A}$ s.t. $q'_0 \in [q_0]_x$, $q_i \preceq_x q'_i$ for $i \geq 0$ and, moreover, if $[q_i]_x$ is accepting then there is q'_k for $k \geq i$ s.t. $q'_k \in F$.

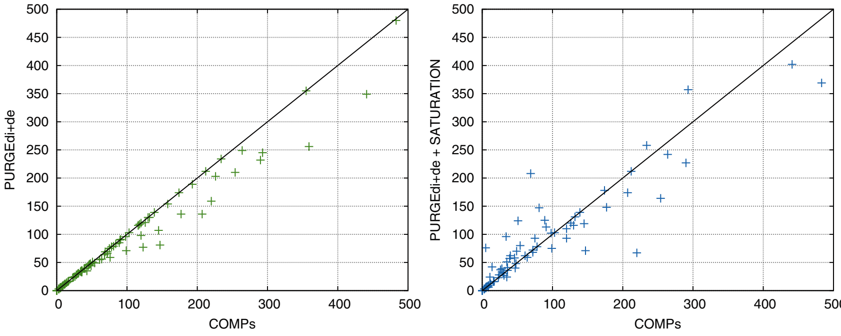
Further, let $[q_0]_x \preceq_x^{\approx} [r_0]_x$. Then there is a trace $\pi_r = \sigma_x(r, \pi_q) = (r = r_0) \xrightarrow{\alpha_0} r_1 \xrightarrow{\alpha_1} \cdots$ simulating π_q in $\mathcal{A}$ from r. Further, consider its projection $[\pi_r]_x = [r_0]_x \xrightarrow{\alpha_0} [r_1]_x \xrightarrow{\alpha_1} \cdots$ into $\mathcal{A}/{\approx_x}$. For all $i \geq 0$, we have that $q_i \preceq_x r_i$, and therefore also $[q_i]_x \preceq_x^{\approx} [r_i]_x$. Since $\mathcal{C}^x(\pi_q, \pi_r)$, then also $\mathcal{C}^x([\pi_q]_x, [\pi_r]_x)$.

Finally, we show that $\preceq_x^{\approx}$ is maximal. For the sake of contradiction, suppose that $[r]_x$ is x-simulating $[q]_x$ for some $q, r \in Q$ s.t. $q \not\preceq_x r$. Consider a word $\alpha \in \Sigma^\omega$ and a trace $\pi_q = (q = q_0) \xrightarrow{\alpha_0} q_1 \xrightarrow{\alpha_1} \cdots$ over α in $\mathcal{A}$. Then there is a trace $[\pi_q]_x = [q = q_0]_x \xrightarrow{\alpha_0} [q_1]_x \xrightarrow{\alpha_1} \cdots$ over α in $\mathcal{A}/{\approx_x}$. According to the assumption, there is also a trace $[\pi_r]_x = [r = r_0]_x \xrightarrow{\alpha_0} [r_1]_x \xrightarrow{\alpha_1} \cdots$ such that $[\pi_r]_x$ is x-simulating $[\pi_q]_x$. But then there will also exist a trace $\pi_r = (r = r_0) \xrightarrow{\alpha_0} r'_1 \xrightarrow{\alpha_1} r'_1 \xrightarrow{\alpha_2} \cdots$ such that $r_i \preceq_x r'_i$ for all $i \in \omega$ and $\mathcal{C}^x(\pi_q, \pi_r)$ (see the previous part of the proof). Therefore, since $\preceq_x$ is maximal, we have that $q \preceq_x r$, which is in contradiction with the assumption. □

7 Experimental Evaluation

We implemented our optimisations in a prototype tool [2] written in Haskell and performed preliminary experimental evaluation on a set of 124 random BAs with

[2] https://github.com/vhavlena/ba-complement .



(a) PURGE_{di+de} vs. COMP_S (b) PURGE_{di+de}+SATURATE vs. COMP_S

Fig. 1. Comparison of the number of states of complement BAs generated by COMP_S and our optimizations (lower is better)

a non-trivial language over a two-symbol alphabet generated using Tabakov and Vardi's model [37]. The parameters of input automata were set to the following bounds: number of states: 6–7, transition density: 1.2–1.3, and acceptance density: 0.35–0.5. Before complementing, the BAs were quotiented wrt the direct simulation for experiments with PURGE_{di} and the delayed simulation for experiments with PURGE_{de} and PURGE_{di+de}. The timeout was set to 300 s.

We present the results for our strongest optimizations for *outputs* of the size up to 500 states in Fig. 1. As can be seen in Fig. 1a, purging alone often significantly reduces the size of the output. The situation with saturation is, on the other hand, more complicated. In Fig. 1b, we can see that in some cases, the saturation produces even smaller BAs than only purging, on the other hand, in some cases, larger BAs are produced. This is expected, because saturating the S component of macrostates also means that more level rankings (the f component) need to be considered.

For outputs of a larger size (we had 11 of them), the results follow a similar trend, but the probability that saturation will increase the size of the result decreases. For some concrete results, for one BA, the size of the output BA decreased from 4065 (COMP_S) to 985 (PURGE_{di+de}) to 929 (PURGE_{di+de} +SATURATE), which yields a reduction to 24 %, resp. 22 %! Further, we observed that all PURGE_x methods usually give similar results, with the difference of only a few states (when PURGE_{di} and PURGE_{de} differ, PURGE_{di} usually wins over PURGE_{de}).

8 Related Work

BA complementation has a long research track. Known approaches can be roughly classified into Ramsey-based [34], determinization-based [29,32], rank-based [33], slice-based [23,39], learning-based [25], and the recently proposed subset-tuple construction [4]. Those approaches build on top of different concepts of capturing words accepted by a complement automaton. Some concepts can be translated into others, such as the slice-based approach, which can be translated to the rank-based approach [40]. Such a translation can help us get a deeper understanding of the BA complementation problem and the relationship between optimization techniques for different complementation algorithms.

Because of the high computational complexity of complementing a BA, and, consequently, also checking BA inclusion and universality (which use complementation as their component), there has been some effort to develop heuristics that help to reduce the number of explored states in practical cases. The most prominent ones are heuristics that leverage various notions of simulation relations, which often provide a good compromise between the overhead they impose and the achieved state space reduction. Direct [7,36], delayed [12], fair [12], their variants for alternating Büchi automata [16], and multi-pebble simulations [13] are the best-studied relations of this kind. Some of the relations can be used quotienting, but also for pruning transitions entering simulation-smaller states (which may cause some parts of the BA to become inaccessible). A series of results in this direction was recently developed by Clemente and Mayr [10,26,27].

Not only can the relations be used for reducing the size of the input BA, they can also be used for under-approximating inclusion of languages of states. For instance, during a BA inclusion test $\mathcal{L}(\mathcal{A}_S) \overset{?}{\subseteq} \mathcal{L}(\mathcal{A}_B)$, if every initial state of $\mathcal{A}_S$ is simulated by an initial state of $\mathcal{A}_B$, the inclusion holds and no complementation needs to be performed. But simulations can also be used to reduce the explored state space within, e.g., the inclusion check itself, for instance in the context of Ramsey-based algorithms [1,2]. Ramsey-based complementation algorithms [34] in the worst case produce $2^{\mathcal{O}(n^2)}$ states, which is a significant gap from the lower bound of Michel [28] and Yan [41]. The Ramsey-based construction was, however, later improved by Breuers et al. [5] to match the upper bound $2^{\mathcal{O}(n \log n)}$. The way simulations are applied in the Ramsey-based approach is fundamentally different from the current work, which is based on rank-based construction. Taking universality checking as an example, the algorithm checks if the language of the complement automaton is empty. They run the complementation algorithm and the emptiness check together, on the fly, and during the construction check if a macrostate with a larger language has been produced before; if yes, then they can stop the search from the language-smaller macrostate. Note that, in contrast to our approach, their algorithm does not produce the complement automaton.

9 Conclusion and Future Work

We developed two novel optimizations of the rank-based complementation algorithm for Büchi automata that are based on leveraging direct and delayed simulation relations to reduce the number of states of the complemented automaton.

The optimizations are directly usable in rank-based BA inclusion and universality checking. We conjecture that the decision problem of checking BA language inclusion might also bring another opportunities for exploiting simulation, such as in a similar manner as in [3]. Another, orthogonal, directions of future work are (i) applying simulation in other than the rank-based approach (in addition to the particular use within [1,2]), e.g., complementation based on Safra's construction [32], which, according to our experience, often produces smaller complements than the rank-based procedure, (ii) applying our ideas within determinization constructions for BAs, and (iii) generalizing our techniques for richer simulations, such as the multi-pebble simulation [13] or various look-ahead simulations [26,27]. Since the richer simulations are usually harder to compute, it would be interesting to find the sweet spot between the overhead of simulation computation and the achieved state space reduction.

Acknowledgement. We thank the anonymous reviewers for their helpful comments on how to improve the exposition in this paper. This work was supported by the Ministry of Science and Technology of Taiwan project 106-2221-E-001-009-MY3 the Czech Science Foundation project 19-24397S, the FIT BUT internal project FIT-S-17-4014, and The Ministry of Education, Youth and Sports from the National Programme of Sustainability (NPU II) project IT4Innovations excellence in science—LQ1602.

References

1. Abdulla, P.A., et al.: Simulation subsumption in Ramsey-based Büchi automata universality and inclusion testing. In: Touili, T., Cook, B., Jackson, P. (eds.) CAV 2010. LNCS, vol. 6174, pp. 132–147. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-14295-6_14
2. Abdulla, P.A., et al.: Advanced Ramsey-based Büchi automata inclusion testing. In: Katoen, J.-P., König, B. (eds.) CONCUR 2011. LNCS, vol. 6901, pp. 187–202. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-23217-6_13
3. Abdulla, P.A., Chen, Y.-F., Holík, L., Mayr, R., Vojnar, T.: When simulation meets antichains. In: Esparza, J., Majumdar, R. (eds.) TACAS 2010. LNCS, vol. 6015, pp. 158–174. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-12002-2_14
4. Allred, J.D., Ultes-Nitsche, U.: A Simple and optimal complementation algorithm for Büchi automata. In: Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science, pp. 46–55. ACM (2018)
5. Breuers, S., Löding, C., Olschewski, J.: Improved Ramsey-based Büchi complementation. In: Birkedal, L. (ed.) FoSSaCS 2012. LNCS, vol. 7213, pp. 150–164. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-28729-9_10
6. Büchi, J.R.: On a decision method in restricted second order arithmetic. In: Proceedings of International Congress on Logic, Method, and Philosophy of Science 1960. Stanford University Press, Stanford (1962)
7. Bustan, D., Grumberg, O.: Simulation-based minimization. ACM Trans. Comput. Logic **4**(2), 181–206 (2003)
8. Cécé, G.: Foundation for a series of efficient simulation algorithms. In: Proceedings of LICS 2017, pp. 1–12 (2017)

9. Chen, Y., et al.: Advanced automata-based algorithms for program termination checking. In: Proceedings of PLDI 2018, pp. 135–150. ACM (2018)
10. Clemente, L.: Büchi automata can have smaller quotients. In: Aceto, L., Henzinger, M., Sgall, J. (eds.) ICALP 2011. LNCS, vol. 6756, pp. 258–270. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-22012-8_20
11. Dill, D.L., Hu, A.J., Wong-Toi, H.: Checking for language inclusion using simulation preorders. In: Larsen, K.G., Skou, A. (eds.) CAV 1991. LNCS, vol. 575, pp. 255–265. Springer, Heidelberg (1992). https://doi.org/10.1007/3-540-55179-4_25
12. Etessami, K., Wilke, T., Schuller, R.: Fair simulation relations, parity games, and state space reduction for Büchi automata. SIAM J. Comput. **34**(5), 1159–1175 (2005)
13. Etessami, K.: A hierarchy of polynomial-time computable simulations for automata. In: Brim, L., Křetínský, M., Kučera, A., Jančar, P. (eds.) CONCUR 2002. LNCS, vol. 2421, pp. 131–144. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-45694-5_10
14. Fogarty, S., Vardi, M.Y.: Büchi complementation and size-change termination. In: Kowalewski, S., Philippou, A. (eds.) TACAS 2009. LNCS, vol. 5505, pp. 16–30. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-00768-2_2
15. Friedgut, E., Kupferman, O., Vardi, M.: Büchi complementation made tighter. Int. J. Found. Comput. Sci. **17**, 851–868 (2006)
16. Fritz, C., Wilke, T.: Simulation relations for alternating Büchi automata. Theor. Comput. Sci. **338**(1), 275–314 (2005)
17. van Glabbeek, R., Ploeger, B.: Five determinisation algorithms. In: Ibarra, O.H., Ravikumar, B. (eds.) CIAA 2008. LNCS, vol. 5148, pp. 161–170. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-70844-5_17
18. Gurumurthy, S., Bloem, R., Somenzi, F.: Fair simulation minimization. In: Brinksma, E., Larsen, K.G. (eds.) CAV 2002. LNCS, vol. 2404, pp. 610–623. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-45657-0_51
19. Heizmann, M., Hoenicke, J., Podelski, A.: Termination analysis by learning terminating programs. In: Biere, A., Bloem, R. (eds.) CAV 2014. LNCS, vol. 8559, pp. 797–813. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-08867-9_53
20. Henzinger, M.R., Henzinger, T.A., Kopke, P.W.: Computing simulations on finite and infinite graphs. In: Proceedings of FOCS 1995, pp. 453–462. IEEE Computer Society (1995)
21. Henzinger, T.A., Kupferman, O., Rajamani, S.K.: Fair simulation. Inf. Comput. **173**(1), 64–81 (2002)
22. Ilie, L., Navarro, G., Yu, S.: On NFA reductions. In: Karhumäki, J., Maurer, H., Păun, G., Rozenberg, G. (eds.) Theory Is Forever. LNCS, vol. 3113, pp. 112–124. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-27812-2_11
23. Kähler, D., Wilke, T.: Complementation, disambiguation, and determinization of Büchi automata unified. In: Aceto, L., Damgård, I., Goldberg, L.A., Halldórsson, M.M., Ingólfsdóttir, A., Walukiewicz, I. (eds.) ICALP 2008. LNCS, vol. 5125, pp. 724–735. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-70575-8_59
24. Kupferman, O., Vardi, M.Y.: Weak alternating automata are not that weak. ACM Trans. Comput. Logic **2**(3), 408–429 (2001)
25. Li, Y., Turrini, A., Zhang, L., Schewe, S.: Learning to complement Büchi automata. In: Dillig, I., Palsberg, J. (eds.) Verification, Model Checking, and Abstract Interpretation. LNCS, vol. 10747, pp. 313–335. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-73721-8_15

26. Mayr, R., Clemente, L.: Advanced automata minimization. In: Proceedings of POPL 2013, pp. 63–74 (2013)
27. Mayr, R., Clemente, L.: Efficient reduction of nondeterministic automata with application to language inclusion testing. Logical Methods Comput. Sci. **15**, 12:1–12:73 (2019)
28. Michel, M.: Complementation is more difficult with automata on infinite words. In: CNET, Paris, vol. 15 (1988)
29. Piterman, N.: From nondeterministic Büchi and Streett automata to deterministic parity automata. In: Proceedings of LICS 2006. pp. 255–264. IEEE (2006)
30. Ranzato, F., Tapparo, F.: A new efficient simulation equivalence algorithm. In: Proceedings of LICS 2007, pp. 171–180 (2007)
31. Ranzato, F., Tapparo, F.: An efficient simulation algorithm based on abstract interpretation. Inf. Comput. **208**(1), 1–22 (2010)
32. Safra, S.: On the complexity of ω-automata. In: Proceedings of FOCS 1988, pp. 319–327. IEEE (1988)
33. Schewe, S.: Büchi complementation made tight. In: Proceedings of STACS 2009, pp. 661–672. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik (2009)
34. Sistla, A.P., Vardi, M.Y., Wolper, P.: The complementation problem for Büchi automata with applications to temporal logic. Theor. Comput. Sci. **49**(2–3), 217–237 (1987)
35. Sistla, A.P., Vardi, M.Y., Wolper, P.: The complementation problem for Büchi automata with applications to temporal logic. In: Brauer, W. (ed.) ICALP 1985. LNCS, vol. 194, pp. 465–474. Springer, Heidelberg (1985). https://doi.org/10.1007/BFb0015772
36. Somenzi, F., Bloem, R.: Efficient Büchi automata from LTL formulae. In: Emerson, E.A., Sistla, A.P. (eds.) CAV 2000. LNCS, vol. 1855, pp. 248–263. Springer, Heidelberg (2000). https://doi.org/10.1007/10722167_21
37. Tabakov, D., Vardi, M.Y.: Experimental evaluation of classical automata constructions. In: Sutcliffe, G., Voronkov, A. (eds.) LPAR 2005. LNCS (LNAI), vol. 3835, pp. 396–411. Springer, Heidelberg (2005). https://doi.org/10.1007/11591191_28
38. Vardi, M.Y.: The Büchi complementation saga. In: Thomas, W., Weil, P. (eds.) STACS 2007. LNCS, vol. 4393, pp. 12–22. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-70918-3_2
39. Vardi, M.Y., Wilke, T.: Automata: from logics to algorithms. Logic Automata **2**, 629–736 (2008)
40. Vardi, M.Y., Wilke, T., Kupferman, O., Fogarty, S.J.: Unifying Büchi complementation constructions. Logical Methods Comput. Sci. **9**, 1–25 (2013)
41. Yan, Q.: Lower bounds for complementation of ω-automata via the full automata technique. In: Bugliesi, M., Preneel, B., Sassone, V., Wegener, I. (eds.) ICALP 2006. LNCS, vol. 4052, pp. 589–600. Springer, Heidelberg (2006). https://doi.org/10.1007/11787006_50