Büchi Automata Can Have Smaller Quotients

Lorenzo Clemente

LFCS, School of Informatics, University of Edinburgh, UK

Abstract. We study novel simulation-like preorders for quotienting nondeterministic Büchi automata. We define *fixed-word delayed simulation*, a new preorder coarser than delayed simulation. We argue that fixed-word simulation is the coarsest forward simulation-like preorder which can be used for quotienting Büchi automata, thus improving our understanding of the limits of quotienting. Also, we show that computing fixed-word simulation is PSPACE-complete.

On the practical side, we introduce *proxy simulations*, which are novel polynomial-time computable preorders sound for quotienting. In particular, *delayed proxy simulation* induce quotients that can be smaller by an arbitrarily large factor than direct backward simulation. We derive proxy simulations as the product of a theory of refinement transformers: A *refinement transformer* maps preorders nondecreasingly, preserving certain properties. We study under which general conditions refinement transformers are sound for quotienting.

1 Introduction

Büchi automata minimization is an important topic in automata theory, both for the theoretical understanding of automata over infinite words and for practical applications. Minimizing an automaton means reducing the number of its states as much as possible, while preserving the recognized language. Minimal automata need not be unique, and their structure does not necessarily bear any resemblance to the original model; in the realm of infinite words, this holds even for deterministic models. This hints at why exact minimization has high complexity: Indeed, minimality checking is PSPACE-hard for nondeterministic models (already over finite words [13]), and NP-hard for deterministic Büchi automata [20]. Moreover, even approximating the minimal model is hard [9].

By posing suitable restrictions on the minimization procedure, it is nonetheless possible to trade exact minimality for efficiency. In the approach of *quotienting*, smaller automata are obtained by merging together equivalent states, under appropriately defined equivalences. In particular, quotienting by simulation equivalence has proven to be an effective heuristics for reducing the size of automata in cases of practical relevance.

The notion of *simulation preorder* and *equivalence* [18] is a crucial tool for comparing the behaviour of systems. It is best described via a game between two players, Duplicator and Spoiler, where the former tries to stepwise match the moves of the latter. But not every simulation preorder can be used for quotienting: We call a preorder *good for quotienting* (GFQ) if the quotient automaton (w.r.t. the induced equivalence) recognizes the same language as the original automaton. In particular, a necessary condition for a simulation to be GFQ is to take into account the acceptance condition: For example, in *direct* simulation [5], Duplicator has the additional requirement to visit an accepting state whenever Spoiler does so, while in the coarser *fair* simulation [11], Duplicator has to visit infinitely many accepting states if Spoiler does so. But, while direct simulation is GFQ [2], fair simulation is not [12].[1] This prompted the development of *delayed* simulation [7], a GFQ preorder intermediate between direct and fair simulation.

We study the border of GFQ preorders. In our first attempt we generalize delayed simulation to *delayed containment*. While in simulation the two players take turns in selecting transitions, in containment the game ends in one round: First Spoiler picks an infinite path, and then Duplicator has to match it with another infinite path. The winning condition is delayed-like: Every accepting state of Spoiler has to be matched by an accepting state of Duplicator, possibly occurring later. Therefore, in delayed containment Duplicator is much stronger than in simulation; in other words, containment is coarser than simulation. In fact, it is *too coarse*: We give a counterexample where delayed containment is not GFQ. We henceforth turn our attention to finer preorders.

In our second attempt, we remedy to the deficiency above by introducing *fixed-word delayed simulation*, an intermediate notion between simulation and containment. In fixed-word simulation, Spoiler does not reveal the whole path in advance like in containment; instead, she only declares the input word beforehand. Then, the simulation game starts, but now transitions can be taken only if they match the word fixed earlier by Spoiler. Unlike containment, fixed-word delayed simulation is GFQ, as we show.

We proceed by looking at even coarser GFQ preorders. We enrich fixed-word simulation by allowing Duplicator to use *multiple pebbles*, in the style of [6]. The question arises as whether Duplicator gains more power by "hedging her bets" when she already knows the input word in advance. By using an ordinal ranking argument (reminiscent of [16]), we establish that this is not the case, and that the multipebble hierarchy collapses to the 1-pebble case, i.e., to fixed-word delayed simulation itself. Incidentally, this also shows that the whole delayed multipebble hierarchy from [6] is entirely contained in fixed-word delayed simulation—the containment being strict.

For what concerns the complexity of computing fixed-word simulation, we establish that it is PSPACE-complete, by a mutual reduction from Büchi automata universality.

With the aim of getting tractable preorders, we then look at a different way of obtaining GFQ relations, by introducing a theory of refinement transformers: A *refinement transformer* maps a preorder $\preceq$ to a coarser preorder $\preceq'$, s.t., once $\preceq$ is known, $\preceq'$ can be computed with only a polynomial time overhead. The idea is to play a simulation-like game, where we allow Duplicator to "jump" to $\preceq$-bigger states, called *proxies*, after Spoiler has selected her transition. Duplicator can then reply with a transition from the proxy instead of the original state. We say that proxy states are *dynamic* in the sense that they depend on the transition selected by Spoiler.[2] Under certain conditions, we show that refinement transformers induce GFQ preorders.

Finally, we introduce *proxy simulations*, which are novel polynomial time GFQ preorders obtained by applying refinement transformers to a concrete preorder $\preceq$, namely, to *backward* direct simulation (called reverse simulation in [21]). We define two versions of proxy simulation, direct and delayed, the latter being coarser than the former,

[1] In fact, for Büchi automata it is well-known that also language equivalence is not GFQ.

[2] Proxies are strongly related to mediators [1]. We compare them in depth in Section 6.

and both coarser than direct backward simulation. Moreover, we show that the delayed variant can achieve quotients smaller than direct proxy simulation by an arbitrarily large factor. Full proofs can be found in the technical report [3].

Related work. Delayed simulation [7] has been extended to generalized automata [14], to multiple pebbles [6], to alternating automata [8] and to the combination of the last two [4]. Fair simulation has been used for state space reduction in [10]. The abstract idea of mixing forward and backward modes in quotienting can be traced back at least to [19]; in the context of alternating automata, it has been studied in [1].

2 Preliminaries

Games. For a finite sequence $\pi = e_0 e_1 \cdots e_{k-1}$, let $|\pi| = k$ be its length, and let $\mathrm{last}(\pi) = e_{k-1}$ be its last element. If π is infinite, then take $|\pi| = \omega$.

A *game* is a tuple $G = (P, P_0, P_1, p_I, \Gamma, \Gamma_0, \Gamma_1, W)$, where P is the set of positions, partitioned into disjoint sets P_0 and P_1, $p_I \in P_0$ is the initial position, $\Gamma = \Gamma_0 \cup \Gamma_1$ is the set of moves, where $\Gamma_0 \subseteq P_0 \times P_1$ and $\Gamma_1 \subseteq P_1 \times P_0$ are the set of moves of Player 0 and Player 1, respectively, and $W \subseteq P_0^\omega$ is the winning condition. A *path* is a finite or infinite sequence of states $\pi = p_0^0 p_0^1 p_1^0 p_1^1 \cdots$ starting in p_I, such that, for all $i < |\pi|$, $(p_i^0, p_i^1) \in \Gamma_0$ and $(p_i^1, p_{i+1}^0) \in \Gamma_1$. *Partial plays* and *plays* are finite and infinite paths, respectively. We assume that there are no dead ends in the game. A play is *winning* for Player 1 iff $p_0^0 p_1^0 p_2^0 \cdots \in W$; otherwise, is it winning for Player 0.

A *strategy* for Player 0 is a partial function $\sigma_0 : (P_0 P_1)^* P_0 \mapsto P_1$ s.t., for any partial play $\pi \in (P_0 P_1)^* P_0$, if σ_0 is defined on π, then $\pi \cdot \sigma_0(\pi)$ is again a partial play. A play π is σ_0*-conform* iff, for every $i \geq 0$, $p_i^1 = \sigma_0(p_0^0 p_0^1 \cdots p_i^0)$. Similarly, a strategy for Player 1 is a partial function $\sigma_1 : (P_0 P_1)^+ \mapsto P_0$ s.t., for any partial play $\pi \in (P_0 P_1)^+$, if σ_1 is defined on π, then $\pi \cdot \sigma_1(\pi)$ is again a partial play. A play π is σ_1-conform iff, for every $i \geq 0$, $p_{i+1}^0 = \sigma_0(p_0^0 p_0^1 \cdots p_i^0 p_i^1)$. While we do not require strategies to be total functions, we do require that a strategy σ is defined on all σ-conform partial plays.

A strategy σ_i is a *winning strategy* for Player i iff all σ_i-conform plays are winning for Player i. We say that Player i wins the game G if she has a winning strategy.

Automata. A *nondeterministic Büchi automaton* (NBA) is a tuple $\mathcal{Q} = (Q, \Sigma, I, \Delta, F)$, where Q is a finite set of states, Σ is a finite alphabet, $I \subseteq Q$ is the set of initial states, $F \subseteq Q$ is the set of final states and $\Delta \subseteq Q \times \Sigma \times Q$ is the transition relation. We also write $q \xrightarrow{a} q'$ instead of $(q, a, q') \in \Delta$, and just $q \longrightarrow q'$ when $\exists a \in \Sigma \cdot q \xrightarrow{a} q'$. For two sets of states $\mathbf{q}, \mathbf{q}' \subseteq Q$, we write $\mathbf{q} \overset{a}{\Longrightarrow} \mathbf{q}'$ iff $\forall q' \in \mathbf{q}' \cdot \exists q \in \mathbf{q} \cdot q \xrightarrow{a} q'$.[3] For a state $q \in Q$, let $[q \in F] = 1$ if q is accepting, and 0 otherwise. We assume that every state is reachable from some initial state, and that the transition relation is total.

For a finite or infinite sequence of states $\rho = q_0 q_1 \cdots$ and an index $i \leq |\rho|$, let $\mathsf{cnt\text{-}final}(\rho, i)$ be the number of final states occuring in ρ up to (and including) the i-th element. Formally, $\mathsf{cnt\text{-}final}(\rho, i) = \sum_{0 \leq k < i} [q_k \in F]$, with $\mathsf{cnt\text{-}final}(\rho, 0) = 0$. Let $\mathsf{cnt\text{-}final}(\rho) = \mathsf{cnt\text{-}final}(\rho, |\rho|)$. If ρ is infinite, then $\mathsf{cnt\text{-}final}(\rho) = \omega$ iff ρ contains infinitely many accepting states.

[3] This kind of backward-compatible transition had already appeared in [17].

Fix a finite or infinite word $w = a_0 a_1 \cdots$. A *path π over w* is a sequence $q_0 \xrightarrow{a_0} q_1 \xrightarrow{a_1} q_2 \cdots$ of length $|w|+1$. A path is *initial* if it starts in an initial state $q_0 \in I$, it is a *run* if it is initial and infinite, and it is *fair* if $\mathsf{cnt\text{-}final}(\pi) = \omega$. An *accepting run* is a run which is fair. The *language $\mathcal{L}^\omega(\mathcal{Q})$* of a NBA $\mathcal{Q}$ is the set of infinite words which admit an accepting run, i.e., $\mathcal{L}^\omega(\mathcal{Q}) = \{w \in \Sigma^\omega \mid \text{ there exists an accepting run } \pi \text{ over } w\}$.

Quotients. Let $\mathcal{Q} = (Q, \Sigma, I, \Delta, F)$ be a NBA and let R be any binary relation on Q. We say that $\approx_R$ is the *equivalence induced by R* if $\approx_R$ is the largest equivalence contained in the transitive and reflexive closure of R. I.e., $\approx_R = R^* \cap (R^*)^{-1}$. Let the function $[\cdot]_R : Q \mapsto 2^Q$ map each element $q \in Q$ to the equivalence class $[q]_R \subseteq Q$ it belongs to, i.e., $[q]_R := \{q' \in Q \mid q \approx_R q'\}$. We overload $[P]_R$ on sets $P \subseteq Q$ by taking the set of equivalence classes. When clear from the context, we avoid noting the dependence of $\approx$ and $[\cdot]$ on R.

An equivalence $\approx$ on $\mathcal{Q}$ induces the *quotient automaton* $\mathcal{Q}_\approx = ([Q], \Sigma, [I], \Delta_\approx, [F])$, where, for any $q, q' \in Q$ and $a \in \Sigma$, $([q], a, [q']) \in \Delta_\approx$ iff $(q, a, q') \in \Delta$. This is called a *naïve* quotient since both initial/final states and transitions are induced representative-wise. When we quotient w.r.t. a relation R which is not itself an equivalence, we actually mean quotenting w.r.t. the induced equivalence $\approx$. We say that R is *good for quotienting* (GFQ) if quotienting $\mathcal{Q}$ w.r.t. R preserves the language, that is, $\mathcal{L}^\omega(\mathcal{Q}) = \mathcal{L}^\omega(\mathcal{Q}_\approx)$.

Lemma 1. *For two equivalences $\approx_0, \approx_1$, if $\approx_0 \subseteq \approx_1$, then $\mathcal{L}^\omega(\mathcal{Q}_{\approx_0}) \subseteq \mathcal{L}^\omega(\mathcal{Q}_{\approx_1})$. In particular, by letting $\approx_0$ be the identity, $\mathcal{L}^\omega(\mathcal{Q}) \subseteq \mathcal{L}^\omega(\mathcal{Q}_{\approx_1})$.*

3 Quotienting with Forward Simulations

In this section we study several generalizations of delayed simulation, in order to investigate the border of good for quotienting (GFQ) forward-like preorders. In our first attempt we introduce *delayed containment*, which is obtained as a modification of the usual simulation interaction between players: In the delayed containment game between q and s there are only two rounds. Spoiler moves first and selects both an infinite word $w = a_0 a_1 \cdots$ and an infinite path $q_0 \xrightarrow{a_0} q_1 \xrightarrow{a_1} \cdots$ over w starting in $q = q_0$; then, Duplicator replies with an infinite path $s_0 \xrightarrow{a_0} s_1 \xrightarrow{a_1} \cdots$ over w starting in $s = s_0$. The winning condition is delayed-like: $\forall i \cdot q_i \in F \implies \exists j \geq i \cdot s_j \in F$. If Duplicator wins the delayed containment game between q and s, we write $q \subseteq^{\text{de}} s$. Clearly, $\subseteq^{\text{de}}$ is a preorder implying language containment. One might wonder whether delayed-containment is GFQ. Unfortunately, this is not the case (see Figure 5 in the tech. rep. [3]). Therefore, $\subseteq^{\text{de}}$ is too coarse for quotienting, and we shall look at finer relations.

Lemma 2. *$\subseteq^{de}$ is not a GFQ preorder.*

3.1 Fixed-Word Delayed Simulation

Our second attempt at generalizing delayed simulation still retains the flavour of containment. While in containment $\subseteq^{\text{de}}$ Spoiler reveals both the input word w and a path over w, in *fixed-word* simulation $\sqsubseteq^{\text{de}}_{\text{fx}}$ Spoiler reveals w only. Then, after w has been fixed, the game proceeds like in delayed simulation, with the proviso that transitions match symbols in w.[4] Formally, let $w = a_0 a_1 \cdots \in \Sigma^\omega$. In the w-simulation game

[4] The related notion of fixed-word *fair* simulation clearly coincides with ω-language inclusion.

$G^{\mathrm{de}}_w(q,s)$ the set of positions of Spoiler is $P_0 = Q \times Q \times \mathbb{N}$, the set of positions of Duplicator is $P_1 = Q \times Q \times Q \times \mathbb{N}$ and $\langle q, s, 0\rangle$ is the initial position. Transitions are determined as follows: Spoiler can select a move of the form $(\langle q,s,i\rangle, \langle q,s,q',i\rangle) \in \Gamma_0^{w\text{-de}}$ if $q \xrightarrow{a_i} q'$, and Duplicator can select a move of the form $(\langle q,s,q',i\rangle, \langle q',s',i+1\rangle) \in \Gamma_1^{w\text{-de}}$ if $s \xrightarrow{a_i} s'$. Notice that the input symbol a_i is fixed, and it has to match the corresponding symbol in w. The winning condition is $W = \{\langle q_0, s_0, 0\rangle\langle q_1, s_1, 1\rangle \cdots \mid \forall i \cdot q_i \in F \implies \exists j \geq i \cdot s_j \in F\}$. Let $q \sqsubseteq^{\mathrm{de}}_w s$ iff Duplicator wins the w-simulation game $G^{\mathrm{de}}_w(q,s)$, and $q \sqsubseteq^{\mathrm{de}}_{\mathrm{fx}} s$ iff $q \sqsubseteq^{\mathrm{de}}_w s$ for all $w \in \Sigma^\omega$. Clearly, fixed-word simulation is a preorder implying containment.

Fact 1. *$\sqsubseteq^{de}_{fx}$ is a reflexive and transitive relation, and $\forall q, s \in Q \cdot q \sqsubseteq^{de}_{fx} s \implies q \subseteq^{de} s$.*

Unlike delayed containment, fixed-word delayed simulation is GFQ. Moreover, fixed-word delayed simulation quotients can be more succint than (multipebble) delayed simulation quotients by an arbitrarily large factor. See Figure 6 in [3].

Theorem 1. *$\sqsubseteq^{de}_{fx}$ is good for quotienting.*

Complexity of delayed fixed word simulation. Let q, s be two states in $\mathcal{Q}$. We reduce the problem of checking $q \sqsubseteq^{\mathrm{de}}_{\mathrm{fx}} s$ to the universality problem of a suitable alternating Büchi product automaton (ABA) $\mathcal{A}$. We design $\mathcal{A}$ to accept exactly those words w s.t. Duplicator wins $G^{\mathrm{de}}_w(q,s)$. Then, by the definition of $\sqsubseteq^{\mathrm{de}}_{\mathrm{fx}}$, it is enough to check whether $\mathcal{A}$ has universal language. See [22] (or Appendix A.1 [3]) for background on ABAs.

The idea is to enrich configurations in the fixed-word simulation game by adding an obligation bit recording whether Duplicator has any pending constraint to visit an accepting state. Initially the bit is 0, and it is set to 1 whenever Spoiler is accepting; a reset to 0 can occur afterwards, if and when Duplicator visits an accepting state.

Let $\mathcal{Q} = (Q, \Sigma, I, \Delta, F)$ be a NBA. We define a product ABA $\mathcal{A} = (A, \Sigma, \delta, \alpha)$ as follows: The set of states is $A = Q \times Q \times \{0,1\}$, final states are of the form $\alpha = Q \times Q \times \{0\}$ and, for any $\langle q,s,b\rangle \in A$ and $a \in \Sigma$,

$$\delta(\langle q,s,b\rangle, a) = \bigwedge_{q \xrightarrow{a} q'} \bigvee_{s \xrightarrow{a} s'} \langle q', s', b'\rangle, \qquad \text{where } b' = \begin{cases} 0 & \text{if } s \in F \\ 1 & \text{if } q \in F \wedge s \notin F \\ b & \text{otherwise} \end{cases}$$

It follows directly from the definitions that $q \sqsubseteq^{\mathrm{de}}_{\mathrm{fx}} s$ iff $\mathcal{L}^\omega(\langle q,s,0\rangle) = \Sigma^\omega$. A reduction in the other direction is immediate already for NBAs: In fact, an NBA $\mathcal{Q}$ is universal iff $\mathcal{U} \sqsubseteq^{\mathrm{de}}_{\mathrm{fx}} \mathcal{Q}$, where $\mathcal{U}$ is the trivial, universal one-state automaton with an accepting Σ-loop. It is well-known that universality is PSPACE-complete for ABAs/NBAs [15].

Theorem 2. *Computing fixed-word delayed simulation is PSPACE-complete.*

3.2 Multipebble Fixed-Word Delayed Simulation

Having established that fixed-word simulation is GFQ, the next question is whether we can find other natural GFQ preorders between fixed-word and delayed containment. A natural attempt is to add a multipebble facility on top of $\sqsubseteq^{\mathrm{de}}_{\mathrm{fx}}$. Intuitively, when Duplicator uses multiple pebbles she can “hedge her bets” by moving pebbles to several successors. This allows Duplicator to delay committing to any particular choice by arbitrarily many steps: In particular, she can always gain knowledge on any *finite* number of moves by Spoiler. Perhaps surprisingly, we show that *Duplicator does not gain more power by using pebbles*. This is stated in Theorem 3, and it is the major technical result of this section. It follows that, once Duplicator knows the input word in advance, there is no difference between knowing only the next step by Spoiler, or the next l steps, for any finite $l > 1$. Yet, if we allow $l = \omega$ lookahead, then we recover delayed containment $\subseteq^{\text{de}}$, which is not GFQ by Lemma 2. Therefore, w.r.t. to the degree of lookahead, $\sqsubseteq^{\text{de}}_{\text{fx}}$ is the coarsest GFQ relation included in $\subseteq^{\text{de}}$.

We now define the multipebble fixed-word delayed simulation. Let $k \geq 1$ and $w = a_0 a_1 \cdots \in \Sigma^\omega$. In the k-multipebble w-delayed simulation game $G^{k\text{-de}}_w(q, s)$ the set of positions of Spoiler is $Q \times 2^Q \times \mathbb{N}$, the set of positions of Duplicator is $Q \times 2^Q \times Q \times \mathbb{N}$, the initial position is $\langle q, \{s\}, 0\rangle$, and transitions are: $(\langle q, \mathrm{s}, i\rangle, \langle q, \mathrm{s}, q', i\rangle) \in \Gamma_0$ iff $q \xrightarrow{a_i} q'$, and $(\langle q, \mathrm{s}, q', i\rangle, \langle q', \mathrm{s}', i+1\rangle) \in \Gamma_1$ iff $\mathrm{s} \overset{a_i}{\Longrightarrow} \mathrm{s}'$ and $|\mathrm{s}'| \leq k$.

Before defining the winning set we need some preparation. Given an infinite sequence $\pi = \langle q_0, \mathrm{s}_0, 0\rangle\langle q_1, \mathrm{s}_1, 1\rangle \cdots$ over $w = a_0 a_1 \cdots$ and a round $j \geq 0$, we say that a state $s \in \mathrm{s}_j$ *has been accepting* since some previous round $i \leq j$, written $\mathsf{accepting}^i_j(s, \pi)$, iff either $s \in F$, or $i < j$ and there exists $\hat{s} \in \mathrm{s}_{j-1}$ s.t. $\hat{s} \xrightarrow{a_{j-1}} s$ and $\mathsf{accepting}^i_{j-1}(\hat{s}, \pi)$. We say that s_j is *good since round* $i \leq j$, written $\mathsf{good}^i_j(\mathrm{s}_j, \pi)$, iff at round j every state $s \in \mathrm{s}_j$ has been accepting since round i, and j is the least round for which this holds [6]. Duplicator wins a play if, whenever $q_i \in F$ there exists $j \geq i$ s.t. $\mathsf{good}^i_j(\mathrm{s}_j, \pi)$. We write $q \sqsubseteq^{k\text{-de}}_w s$ iff Duplicator wins $G^{k\text{-de}}_w(q, s)$, and we write $q \sqsubseteq^{k\text{-de}}_{\text{fx}} s$ iff $\forall w \in \Sigma^\omega \cdot q \sqsubseteq^{k\text{-de}}_w s$.

Clearly, pebble simulations induce a non-decreasing hierarcy: $\sqsubseteq^{1\text{-de}}_{\text{fx}} \subseteq \sqsubseteq^{2\text{-de}}_{\text{fx}} \subseteq \cdots$. We establish that the hierarchy actually collapses to the $k = 1$ level. This result is non-trivial, since the delayed winning condition requires reasoning not only about the *possibility* of Duplicator to visit accepting states in the future, but also about exactly *when* such a visit occurs. Technically, our argument uses a ranking argument similar to [16] (see Appendix A.2 [3]), with the notable difference that our ranks are *ordinals* ($\leq \omega^2$), instead of natural numbers. We need ordinals to represent how long a player can delay visiting accepting states, and how this events nest with each other. Finally, notice that the result above implies that the multipebble delayed simulation hierarchy of [6] is entirely contained in $\sqsubseteq^{\text{de}}_{\text{fx}}$, and the containment is strict (Fig. 6 [3]).

Theorem 3. *For any NBA $\mathcal{Q}$, $k \geq 1$ and states $q, s \in Q$, $q \sqsubseteq^{k\text{-de}}_{fx} s$ iff $q \sqsubseteq^{de}_{fx} s$.*

4 Jumping-Safe Relations

In this section we present the general technique which is used throughout the paper to establish that preorders are GFQ. We introduce *jumping-safe relations*, which are shown to be GFQ (Theorem 4). In Section 5 we use jumping-safety as an invariant when applying refinement transformers. We start off with an analysis of acceping runs.

Coherent sequences of paths. Fix an infinite word $w \in \Sigma^\omega$. Let $\Pi := \pi_0, \pi_1, \ldots$ be an infinite sequence of longer and longer *finite* initial paths in $\mathcal{Q}$ over (prefixes of) w. We are interested in finding a sufficient condition for the existence of an accepting run over w. A necessary condition is that the number of final states in π_i grows unboundedly as

i goes to ω. In the case of deterministic automata this condition is also sufficient: Indeed, in a deterministic automaton there exists a unique run over w, which is accepting exactly when the number of accepting stated visited by its prefixes goes to infinity. In this case, we say that the π_i's are *strongly coherent* since they next path extends the previous one. Unfortunately, in the general case of nondeterministic automata it is quite possible to have paths that visit arbitrarily many final states but no accepting run exists. This occurs because final states can appear arbitrarily late. Indeed, consider Figure 1. Take $w = aba^2ba^3b\cdots$: For every prefx $w_i = aba^2b\cdots a^i$ there exists a path $\pi_i = qq\cdots q\cdot s^i$ over w_i visiting a final state i times. Still, $w \notin \mathcal{L}^\omega(\mathcal{Q})$.

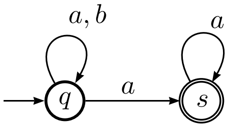


Fig. 1. Automaton $\mathcal{Q}$

Therefore, we forbid accepting states to "clump away" in the tail of the path. We ensure this by imposing the existence of an infinite sequence of indices $j_0, j_1, \cdots$ s.t., for all i, and for all k_i big enough, the number of final states in π_{k_i} up to the j_i-th state is at least i. In this way, we are guaranteed that at least i final states are present within j_i steps in all but finitely many paths.

Definition 1. *Let $\Pi := \pi_0, \pi_1, \ldots$ be an infinite sequence of finite paths. We say that Π is a* coherent sequence of paths *if the following property holds:*

$$\forall i \cdot \exists j \cdot \exists h \cdot \forall k \geq h \cdot j < |\pi_k| \wedge \mathsf{cnt\text{-}final}(\pi_k, j) \geq i\ . \tag{1}$$

Lemma 3. *If Π is coherent, then any infinite subsequence Π' thereof is coherent.*

We sketch below the proof that coherent sequences induce fair paths. Let $\Pi = \pi_0, \pi_1, \ldots$ be a coherent sequence of paths in $\mathcal{Q}$. Let $i = 1$, and let j_1 be the index witnessing Π is coherent. Since the π_k's are branches in a finitely branching tree, there are only a finite number of different prefixes of length j_1. Therefore, there exists a prefix ρ_1 which is common to infinitely many paths. Let $\Pi' = \pi'_0, \pi'_1, \ldots$ be the infinite subsequence of Π containing only suffixes of ρ_1. Clearly ρ_1 contains at least 1 final state, and each π' in Π' extends ρ_1. By Lemma 3, Π' is coherent. For $i = 2$, we can apply the reasoning again to Π', and we obtain a longer prefix ρ_2 extending ρ_1, and containing at least 2 final states. Let Π'' be the coherent subsequence of Π' containing only suffixes of ρ_2. In this fashion, we obtain an infinite sequence of *strongly* coherent (finite) paths $\rho_1, \rho_2, \cdots$ s.t. ρ_i extends ρ_{i-1} and contains at least i final states. The infinite path to which the sequence converges is the fair path we are after.

Lemma 4. *Let $w \in \Sigma^\omega$ and $\pi_0, \pi_1, \ldots$ as above. If $\pi_0, \pi_1, \ldots$ is coherent, then there exists a fair path ρ over w. Moreover, if all π_i's are initial, then ρ is initial.*

Jumping-safe relations. We established that coherent sequences induce accepting paths. Next, we introduce *jumping-safe* relations, which are designed to induce coherent sequences (and thus accepting paths) when used in quotienting. The idea is to view a path in the quotient automaton as a jumping path in the original automaton, where a "jumping path" is one that can take arbitrary jumps to equivalent states. Jumping-safe relations allows us to transform the sequence of prefixes of an accepting jumping path into a coherent sequence of non-jumping paths; by Lemma 4, this induces a (nonjumping) accepting path.

Fix a word $w = a_0 a_1 \cdots \in \Sigma^\omega$, and let R be a binary relation over Q. An *R-jumping path* is an infinite sequence

$$\pi = q_0 \; R \; q_0^F \; R \; \hat{q}_0 \xrightarrow{a_0} q_1 \; R \; q_1^F \; R \; \hat{q}_1 \xrightarrow{a_1} q_2 \cdots , \qquad (2)$$

and we say that π is *initial* if $q_0 \in I$, and *fair* if $q_i^F \in F$ for infinitely many i's.

Definition 2. *A binary relation R is* jumping-safe *iff for any initial R-jumping path π there exists an infinite sequence of initial finite paths $\pi_0, \pi_1, \ldots$ over suitable prefixes of w s.t. $\operatorname{last}(\pi_i) \; R \; q_i$ and, if π is fair, then $\pi_0, \pi_1, \ldots$ is coherent.*

Theorem 4. *Jumping-safe preorders are good for quotienting.*

In Section 5 we introduce refinement transformers, which are designed to preserve jumping-safety. Then, in Section 6 we specialize the approach to *backward direct simulation* $\sqsubseteq_{\mathrm{bw}}^{\mathrm{di}}$ [21], which provides an initial jumping-safe preorder, and which we introduce next: $\sqsubseteq_{\mathrm{bw}}^{\mathrm{di}}$ is the coarsest preorder s.t. $q \sqsubseteq_{\mathrm{bw}}^{\mathrm{di}} s$ implies 1) $\forall (q' \xrightarrow{a} q) \cdot \exists (s' \xrightarrow{a} s) \cdot q' \sqsubseteq_{\mathrm{bw}}^{\mathrm{di}} s'$, 2) $q \in F \implies s \in F$, and 3) $q \in I \implies s \in I$.

Fact 2. *$\sqsubseteq_{bw}^{di}$ is jumping-safe and computable in polynomial time.*

5 Refinement Transformers

We study how to obtain GFQ preorders coarser than forward/backward simulation. As a preliminary example, notice that it is not possible to generalize simultaneously both forward and backward simulations. See the counterexample in Fig. 2, where any relation coarser than both forward and backward simulation is not GFQ. Let $\approx_{\mathrm{bw}}^{\mathrm{di}}$ and $\approx_{\mathrm{fw}}^{\mathrm{di}}$ be backward and forward direct simulation equivalence, respectively. We have $q_1 \approx_{\mathrm{bw}}^{\mathrm{di}} q_2 \approx_{\mathrm{fw}}^{\mathrm{di}} q_3$, but "glueing together" q_1, q_2, q_3 would introduce the extraneous word ba^ω. Therefore, one needs to choose whether to extend either forward or backward simulation. The former approach has been pursued in the *mediated preorders* of [1] (in the more general context of alternating automata). Here, we extend backward refinements.

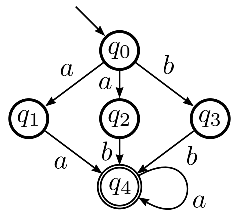


Fig. 2.

We define a *refinement transformer* τ_0 mapping a relation R to a new, coarser relation $\tau_0(R)$. We present τ_0 via a forward direct simulation-like game where Duplicator is allowed to "jump" to R-bigger states—called *proxies*. Formally, in the $\tau_0(R)$ simulation game Spoiler's positions are in $Q \times Q$, Duplicator's position are in $Q \times Q \times \Sigma \times Q$ and transitions are as follows: Spoiler picks a transition $(\langle s, q\rangle, \langle s, q, a, q'\rangle) \in \Gamma_0$ simply when $q \xrightarrow{a} q'$, and Duplicator picks a transition $(\langle s, q, a, q'\rangle, \langle s', q'\rangle) \in \Gamma_1$ iff there exists a proxy $\hat{s}$ s.t. $s \; R \; \hat{s}$ and $\hat{s} \xrightarrow{a} s'$. The winning condition is: $\forall i \geq 0 \cdot q_i \in F \implies \hat{s}_i \in F$. If Duplicator wins starting from the initial position $\langle s, q\rangle$, we write $s \; \tau_0(R) \; q$. (Notice that we swapped the usual order between q and s here.)

Lemma 5. *For a preorder R, $R \subseteq R \circ \tau_0(R) \subseteq \tau_0(R)$.*

Unfortunately, $\tau_0(R)$ is not necessarily a transitive relation. Therefore, it is not immediately clear how to define a suitable equivalence for quotienting. Figure 2 shows that taking the transitive closure of $\tau_0(R)$ is incorrect—already when R is direct backward simulation $\sqsubseteq^{\mathrm{di}}_{\mathrm{bw}}$: Let $\preceq = \tau_0(\sqsubseteq^{\mathrm{di}}_{\mathrm{bw}})$ and let $\approx = \preceq \cap \preceq^{-1}$. We have $q_3 \approx q_2 \approx q_1 \preceq q_3$, but $q_3 \not\preceq q_1$, and forcing $q_1 \approx q_3$ is incorrect, as noted earlier.

Thus, $\tau_0(R)$ is not GFQ and we need to look at its transitive fragments. Let $T \subseteq \tau_0(R)$. We say that R is *F-respecting* if $q \; R \; s \wedge q \in F \implies s \in F$, that T is *self-respecting* if Duplicator wins by never leaving T, that T is *appealing* if transitive and self-respecting, and that T *improves on* R if $R \subseteq T$.

Theorem 5. *Let R a F-respecting preorder, and let $T \subseteq \tau_0(R)$ be an appealing, improving fragment of $\tau_0(R)$. If R is jumping-safe, then T is jumping-safe.*

In particular, by Theorem 4, T is GFQ. Notice that requiring that R is GFQ is not sufficient here, and we need the stronger invariant given by jumping-safety.

Given an appealing fragment $T \subseteq \tau_0(R)$, a natural question is whether $\tau_0(T)$ improves on $\tau_0(R)$, so that τ_0 can be applied repeatedly to get bigger and bigger preorders. We see in the next lemma that this is not the case.

Lemma 6. *For any reflexive R, let $T \subseteq \tau_0(R)$ be any appealing fragment of $\tau_0(R)$. Then, $\tau_0(T) \subseteq \tau_0(R)$.*

Efficient appealing fragments. By Theorems 4 and 5, appealing fragments of τ_0 are GFQ. Yet, we have not specified any method for obtaining these. Ideally, one looks for fragments having maximal cardinality (which yelds maximal reduction under quotienting), but finding them is computationally expensive. Instead, we define a new transformer τ_1 which is guaranteed to produce only appealing fragments,[5] which, while not maximal in general, are maximal amongst all *improving* fragments (Lemma 7).

The reason why $\tau_0(R)$ is not transitive is that only Duplicator is allowed to make "R-jumps". This asymmetry is an obstacle to compose simulation games. We recover transitivity by allowing Spoiler to jump as well, thus restoring the symmetry. Formally, the $\tau_1(R)$ simulation game is identical to the one for $\tau_0(R)$, the only difference being that also Spoiler is now allowed to "jump", i.e., she can pick a transition $(\langle s, q \rangle, \langle s, q, a, q' \rangle) \in \Gamma_0$ iff there exists $\hat{q}$ s.t. $q \; R \; \hat{q}$ and $\hat{q} \xrightarrow{a} q'$. The winning condition is: $\forall i \geq 0 \cdot \hat{q}_i \in F \implies \hat{s}_i \in F$. Let $s \; \tau_1(R) \; q$ if Duplicator wins from position $\langle s, q \rangle$. It is immediate to see that $\tau_1(R)$ is an appealing fragment of $\tau_0(R)$, and that τ_1 is improving on transitive relations R's. Thus, for a preorder R, $R \subseteq \tau_1(R) \subseteq \tau_0(R)$. By Theorems 4 and 5, $\tau_1(R)$ is GFQ (if R is F-respecting).

It turns out that $\tau_1(R)$ is actually the *maximal* appealing, improving fragment of $\tau_0(R)$. This is non-obvious, since the class of appealing T's is not closed under union—still, it admits a maximal element. Therefore, τ_1 is an optimal solution to the problem of finding appealing, improving fragments of $\tau_0(R)$.

[5] τ_1 needs not be the only solution to this problem: Other ways of obtaining appealing fragments of τ_0 might exist. For this reason, we have given a separate treatment of τ_0 in its generality, together with the general correctness statement (Theorem 5).

Lemma 7. *For any R, let $T \subseteq \tau_0(R)$ be any appealing fragment of $\tau_0(R)$. If $R \subseteq T$ (i.e., R is improving), then $T \subseteq \tau_1(R)$.*

5.1 Delayed-Like Refinement Transformers

We show that the refinement transformer approach can yield relations even coarser than τ_1. Our first attempt is to generalize the direct-like winning condition of τ_0 to a delayed one. Let τ_0^{de} be the same as τ_0 except for the different winning condition, which now is: $\forall i \geq 0 \cdot q_i \in F \implies \exists j \geq i \cdot \hat{s}_j \in F$. Clearly, τ_0^{de} inherits the same transitivity issues of τ_0. Unfortunately, the approach of taking appealing fragments is not sound here, due to the weaker winning condition. See Figure 7 in [3] for a counterexample.

We overcome these issues by dropping τ_0^{de} altogether, and directly generalize τ_1 (instead of τ_0) to a delayed-like notion. The *delayed refinement transformer* τ_1^{de} is like τ_1, except for the new winning condition: $\forall i \geq 0 \cdot \hat{q}_i \in F \implies \exists j \geq i \cdot s_j \in F$. Notice that $\tau_1^{\mathrm{de}}(R)$ is at least as coarse as $\tau_1(R)$, and incomparable with $\tau_0(R)$. Once R is given, $\tau_1^{\mathrm{de}}(R)$ can be computed in polynomial time. See Appendix D in the tech. rep. [3].

Lemma 8. *For any R, $\tau_1^{de}(R)$ is transitive.*

Theorem 6. *If R is a jumping-safe F-respecting preorder, then $\tau_1^{de}(R)$ is jumping-safe.*

6 Proxy Simulations

We apply the theory of transformers from Section 5 to a specific F-respecting preorder, namely backward direct simulation, obtaining *proxy simulations*. Notice that proxy simulation-equivalent states need not have the same language; yet, proxy simulations are GFQ (and computable in polynomial time).

6.1 Direct Proxy Simulation

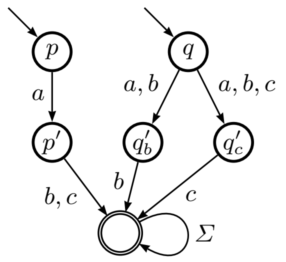
Let *direct proxy simulation*, written $\sqsubseteq_{\mathrm{xy}}^{\mathrm{di}}$, be defined as $\sqsubseteq_{\mathrm{xy}}^{\mathrm{di}} := [\tau_1(\sqsubseteq_{\mathrm{bw}}^{\mathrm{di}})]^{-1}$.

Theorem 7. *$\sqsubseteq_{xy}^{di}$ is a polynomial time GFQ preorder at least as coarse as $(\sqsubseteq_{bw}^{di})^{-1}$.*

Proxies vs mediators. Direct proxy simulation and mediated preorder [1] are in general incomparable. While proxy simulation is at least as coarse as backward direct simulation, mediated preorder is at least as coarse as *forward* direct simulation. (We have seen in Section 5 that this is somehow unavoidable, since one cannot hope to generalize simultaneously both forward and backward simulation.)

One notable difference between the two notions is that proxies are “dynamic”, while mediators are “static”: While Dupicator chooses the proxy only *after* Spoiler has selected her move, mediators are chosen uniformly w.r.t. Spoiler’s move.

In Figure 3(a) we show a simple example where $\sqsubseteq_{\mathrm{xy}}^{\mathrm{di}}$ achieves greater reduction. Recall that mediated preorder M is always a subset of $\sqsubseteq_{\mathrm{fw}}^{\mathrm{di}} \circ (\sqsubseteq_{\mathrm{bw}}^{\mathrm{di}})^{-1}$ [1]. In the example, static mediators are just the trivial ones already present in forward simulation. Thus, $\sqsubseteq_{\mathrm{fw}}^{\mathrm{di}} \circ (\sqsubseteq_{\mathrm{bw}}^{\mathrm{di}})^{-1} = \sqsubseteq_{\mathrm{fw}}^{\mathrm{di}}$ and mediated preorder M collapses to forward simulation. On the other side, $p \approx_{\mathrm{xy}}^{\mathrm{di}} q$ and $p' \approx_{\mathrm{xy}}^{\mathrm{di}} q_b'$. Letting $s = [p, q]$ and $s' = [p', q_b']$, we obtain the quotient in Figure 3(b).

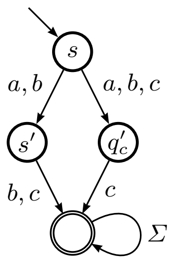


$p \approx^{\mathrm{di}}_{\mathrm{bw}} q$

$p' \sqsubseteq^{\mathrm{di}}_{\mathrm{bw}} q'_b \sqsubseteq^{\mathrm{di}}_{\mathrm{bw}} q'_c$

$\{q'_b, q'_c\} \sqsubseteq^{\mathrm{di}}_{\mathrm{fw}} p'$

(a) Original automaton



(b) Quotient automaton

Fig. 3. Direct proxy simulation quotients

6.2 Delayed Proxy Simulation

Another difference between the mediated preorder approach [1] and the approach through proxies is that proxies directly enable a delayed simulation-like generalization (see Section 5.1). Again, we fix backward delayed simulation $\sqsubseteq^{\mathrm{di}}_{\mathrm{bw}}$ as a starting refinement, and we define *delayed proxy simulation* as $\sqsubseteq^{\mathrm{de}}_{\mathrm{xy}} := [\tau^{\mathrm{de}}_1(\sqsubseteq^{\mathrm{di}}_{\mathrm{bw}})]^{-1}$.

Theorem 8. *$\sqsubseteq^{de}_{xy}$ is a polynomial time GFQ preorder.*

Notice that delayed proxy simulation is at least as coarse as direct proxy simulation. Moreover, quotients w.r.t. $\sqsubseteq^{\mathrm{de}}_{\mathrm{xy}}$ can be smaller than direct forward/backward/proxy and delayed simulation quotients by an arbitrary large factor. See Figure 4: Forward delayed simulation is just the identity, and no two states are direct backward or proxy simulation equivalent. But $q_i \sqsubseteq^{\mathrm{di}}_{\mathrm{bw}} s$ for any $0 < i \leq k - 1$. This causes any two outer states q_i, q_j to be $\sqsubseteq^{\mathrm{de}}_{\mathrm{xy}}$-equivalent. Therefore, the $\sqsubseteq^{\mathrm{de}}_{\mathrm{xy}}$-quotient automaton has only 2 states.

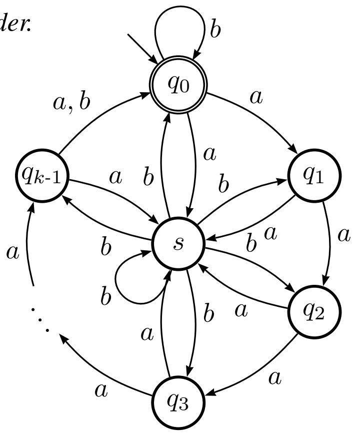


Fig. 4.

7 Conclusions and Future Work

We have proposed novel refinements for quotienting Büchi automata: fixed-word delayed simulation and direct/delayed proxy simulation. Each one has been shown to induce quotients smaller than previously known notions.

We outline a few directions for future work. First, we would like to study practical algorithms for computing fixed-word delayed simulation, and to devise efficient fragments thereof—one promising direction is to look at self-respecting fragments, which usually have lower complexity. Second, we would like to exploit the general correctness argument developed in Section 4 in order to get efficient purely backward refinements (coarser than backward direct simulation). Finally, experiments on cases of practical interest are needed for an empirical evaluation of the proposed techniques.

Acknowledgment. We thank Richard Mayr and Patrick Totzke for helpful discussions, and two anonymous reviewers for their valuable feedback.

References

1. Abdulla, P., Chen, Y.-F., Holik, L., Vojnar, T.: Mediating for Reduction. In: FSTTCS, pp. 1–12. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik (2009)
2. Aziz, A., Singhal, V., Swamy, G.M., Brayton, R.K.: Minimizing Interacting Finite State Machines. Technical Report UCB/ERL M93/68, UoC, Berkeley (1993)
3. Clemente, L.: Büchi Automata can have Smaller Quotients. Technical Report EDI-INF-RR-1399, LFCS, University of Edinburgh (April 2011)
4. Clemente, L., Mayr, R.: Multipebble Simulations for Alternating Automata. In: Gastin, P., Laroussinie, F. (eds.) CONCUR 2010. LNCS, vol. 6269, pp. 297–312. Springer, Heidelberg (2010)
5. Dill, D.L., Hu, A.J., Wont-Toi, H.: Checking for Language Inclusion Using Simulation Preorders. In: Larsen, K.G., Skou, A. (eds.) CAV 1991. LNCS, vol. 575, Springer, Heidelberg (1992)
6. Etessami, K.: A Hierarchy of Polynomial-Time Computable Simulations for Automata. In: Brim, L., Jančar, P., Křetínský, M., Kučera, A. (eds.) CONCUR 2002. LNCS, vol. 2421, pp. 131–144. Springer, Heidelberg (2002)
7. Etessami, K., Wilke, T., Schuller, R.A.: Fair Simulation Relations, Parity Games, and State Space Reduction for Büchi Automata. SIAM J. Comput. 34(5), 1159–1175 (2005)
8. Fritz, C., Wilke, T.: Simulation Relations for Alternating Büchi Automata. Theor. Comput. Sci. 338(1-3), 275–314 (2005)
9. Gramlich, G., Schnitger, G.: Minimizing NFA's and Regular Expressions. Journal of Computer and System Sciences 73(6), 908–923 (2007)
10. Gurumurthy, S., Bloem, R., Somenzi, F.: Fair Simulation Minimization. In: Brinksma, E., Larsen, K.G. (eds.) CAV 2002. LNCS, vol. 2404, pp. 610–624. Springer, Heidelberg (2002)
11. Henzinger, T.A., Kupferman, O., Rajamani, S.K.: Fair Simulation. Information and Computation 173, 64–81 (2002)
12. Henzinger, T.A., Rajamani, S.K.: Fair Bisimulation. In: Graf, S. (ed.) TACAS 2000. LNCS, vol. 1785, pp. 299–314. Springer, Heidelberg (2000)
13. Jiang, T., Ravikumar, B.: Minimal NFA Problems are Hard. In: Leach Albert, J., Monien, B., Rodríguez-Artalejo, M. (eds.) ICALP 1991. LNCS, vol. 510, pp. 629–640. Springer, Heidelberg (1991)
14. Juvekar, S., Piterman, N.: Minimizing Generalized Büchi Automata. In: Ball, T., Jones, R.B. (eds.) CAV 2006. LNCS, vol. 4144, pp. 45–58. Springer, Heidelberg (2006)
15. Kupferman, O., Vardi, M.: Verification of Fair Transition Systems. In: Alur, R., Henzinger, T.A. (eds.) CAV 1996. LNCS, vol. 1102, pp. 372–382. Springer, Heidelberg (1996)
16. Kupferman, O., Vardi, M.: Weak Alternating Automata Are Not That Weak. ACM Trans. Comput. Logic 2, 408–429 (2001)
17. Lynch, N.A., Vaandrager, F.W.: Forward and Backward Simulations. Part I: Untimed Systems. Information and Computation 121(2), 214–233 (1995)

18. Milner, R.: Communication and Concurrency. Prentice-Hall, Englewood Cliffs (1989)
19. Raimi, R.S.: Environment Modeling and Efficient State Reachability Checking. PhD thesis, The University of Texas at Austin (1999)
20. Schewe, S.: Beyond Hyper-Minimisation—Minimising DBAs and DPAs is NP-Complete. In: FSTTCS. LIPIcs, vol. 8, pp. 400–411. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany (2010)
21. 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)
22. Vardi, M.: Alternating Automata and Program Verification. In: van Leeuwen, J. (ed.) Computer Science Today. LNCS, vol. 1000, pp. 471–485. Springer, Heidelberg (1995)