MASARYK
UNIVERSITY

FACULTY OF INFORMATICS

Simulation-Based Reduction of Modal Omega-Automata

Bachelor's Thesis

DÁVID SMOLKA

Brno, Spring 2023

MASARYK UNIVERSITY

FACULTY OF INFORMATICS

Simulation-Based Reduction of Modal Omega-Automata

Bachelor's Thesis

DÁVID SMOLKA

Advisor: prof. RNDr. Jan Strejček, Ph.D.

Brno, Spring 2023


FACULTAS ARTIS INFORMATICÆ
UNIVERSITAS MASARYKIANA

Declaration

Hereby I declare that this paper is my original authorial work, which I have worked out on my own. All sources, references, and literature used or excerpted during elaboration of this work are properly cited and listed in complete reference to the due source.

Dávid Smolka

Advisor: prof. RNDr. Jan Strejček, Ph.D.

Acknowledgements

I would like to thank Jan Strejček for guidance, comprehensive feedback and valuable advice throughout the whole process. Next, I want to thank Ďuri for his excessive troubleshooting and many helpful tips. I would also like to thank my family and friends for their constant support during my studies.

Abstract

In this thesis we introduce modal automata (automata utilizing two sets of transitions) and generalize the concepts of simulations, pruning and quotienting to make use of the modality of such automata. For particular subset of modal simulations, we characterize those that can be used with the modal pruning and modal quotienting. We then show how to reduce the problem of computing particular class these simulations to the problem of solving parity games. In the last two chapters we implement these reduction techniques in tool `Maslo` and compare it with the tool Spot.

Keywords

omega-automata, TBA, MTBA, modal automata, simulations, parity games, pruning, quotienting, Maslo, modality, reduction, Seminator, Spot

Contents

Introduction

Büchi automata are one of the fundamental structures in theoretical computer science used to describe ω-languages. They are often used in complicated computations, therefore it is generally desirable to obtain as small automata as possible. For this reason, many reduction techniques and tools are studied and used. One of the most common tools are *simulations*[1], which relate states of the automaton that are somehow similar in "behaviour". These simulations are then used with reduction techniques like *pruning*[2] and *quotienting*[3]. Pruning works on the principle of removing transitions that are in some sense redundant because some "better" transitions remain. Quotienting identifies states that are equivalent with respect to a simulation and merges them together. Both of these techniques can reduce both the number of states and the number of transitions. Pruning primarily removes transitions but can also make some states unreachable, making them redundant. Quotienting, on the other hand, primarily removes states but can also reduce the number of transitions.

When constructing ω-automata, one is often given the option to either include some transition in the automaton or not. Intuition suggests that by including as few transitions as possible, a smaller automaton will be obtained. However, this does not always have to be the case.

The concept of *modal automata* lies in not only preserving these optional transitions, but also working with the knowledge, that these transitions are optional. In this thesis, our goal is to use this information to reduce modal automata more effectively.

In Chapter 1, we go through basic definitions. Chapter 2 formally defines modal transition-based Büchi automata (MTBA) and generalizes the concept of simulations to MTBA, obtaining modal simulations which take the modality into account. After that, in Chapter 3, we define pruning on modal automata and characterize some simulations that can be used with such pruning. Chapter 4 contains two definitions of quotienting on modal automata and describes some simulations that can be used to quotient MTBA. In Chapter 5, we solve the issue with computing simulations by solving parity games. Chapter 6 contains information about our tool `Maslo`, which is a part of this thesis, and describes how some problems concerning the implementation

were solved. In Chapter 7, we compare our reduction techniques with the state-of-the-art tool Spot[4].

1 Preliminaries

This chapter contains well-known basic definitions and some notation we will use throughout this thesis. We recall the definition of *TBA*, *simulations on TBA*, TBA *pruning*, TBA *quotienting*, and the last section will define *parity games*.

A *strict partial order* is an irreflexive, asymmetric, and transitive relation. Given a finite sequence $\tau = \alpha_0\alpha_1\alpha_2\ldots\alpha_n$, an infinite sequence $\pi = \beta_0\beta_1\beta_2\ldots$ and $0 \leq i \leq j$ we define:

$$\begin{aligned} \tau[i] &= \alpha_i & \pi[i] &= \beta_i \\ \tau[i..] &= \alpha_i\alpha_{i+1}\ldots\alpha_n & \pi[i..] &= \beta_i\beta_{i+1}\ldots \\ \tau[..i] &= \alpha_0\alpha_1\ldots\alpha_{i-1} & \pi[..i] &= \beta_0\beta_1\ldots\beta_{i-1} \\ \tau[i..j] &= \alpha_i\alpha_{i+1}\ldots\alpha_{j-1} & \pi[i..j] &= \beta_i\beta_{i+1}\ldots\beta_{j-1} \\ \tau^R &= \alpha_n\alpha_{n-1}\ldots\alpha_0 \end{aligned}$$

1.1 Parity games

The *parity game arena* is a tuple $\mathcal{G} = (V_0, v_1, E, f)$, where V_0 and V_1 are the two disjoint sets of vertices, $E \subseteq V_0 \times V_1 \cup V_1 \times V_0$ is the set of edges and $f : V \to \mathbb{N}$ is the priority function. Further, $V = V_0 \cup V_1$ is the set of all vertices. Moreover, we will assume that for every vertex $v \in V$, there exists an edge beginning in v.

The *parity game* is a game of two players - Player 0 and Player 1. The players play the game beginning in vertex $v_0 \in V$ as follows: Imagine that we place a pebble on the vertex v_0. In the i-th round, when the pebble is on the vertex $v_{i-1} \in V_x$, where $x \in \{0, 1\}$, the player x chooses an edge beginning in v_{i-1} and moves the pebble onto the vertex $v_i \in V$, in which the edge ends. That is the end of i-th round, and the game continues in the $(i + 1)$-th round.

Hence the players create an infinite sequence of vertices called a play:

$$\rho = v_0v_1v_2v_3\ldots \in V^\omega.$$

Let $\gamma \in \mathbb{N}$ be maximal among the priorities of vertices that appear infinitely many times in the sequence ρ. If γ is even, the winner is Player 0, otherwise Player 1 wins.

Note that every play of a parity game has to be won by either Player 0 or Player 1.

1.2 Transition-based Büchi automata

A *transition-based Büchi automaton* (*TBA*) $\mathcal{A}$ is a tuple $(Q, \Sigma, \delta, q_0, F)$, where

- Q is a finite set of states,
- Σ is a finite alphabet,
- $\delta \subseteq Q \times \Sigma \times Q$ is the transition relation,
- $q_0 \in Q$ is the initial state, and
- $F \subseteq \delta$ is the set of accepting transitions.

A *finite* (*forward*) *trace* of a TBA $\mathcal{A}$ starting in a state $q \in Q$ over a finite word $w = \sigma_0\sigma_1 \ldots \sigma_{i-1} \in \Sigma^i$ is a finite sequence of transitions

$$\pi = (s_0, \sigma_0, d_0)(s_1, \sigma_1, d_1) \ldots (s_{i-1}, \sigma_{i-1}, d_{i-1}) \in \delta^i$$

such that $s_0 = q$ and for every $0 \leq j < (i-1)$, $d_j = s_{j+1}$.
An *infinite* (*forward*) *trace* of a TBA $\mathcal{A}$ starting in a state $q \in Q$ over an infinite word $w = \sigma_0\sigma_1 \ldots \in \Sigma^\omega$ is an infinite sequence of transitions

$$\pi = (s_0, \sigma_0, d_0)(s_1, \sigma_1, d_1) \ldots \in \delta^\omega$$

such that $s_0 = q$ and for every $i \geq 0$, $d_i = s_{i+1}$. A trace is *initial* if it starts in the initial state q_0. An infinite trace π is *fair* if $(s_i, \sigma_i, d_i) \in F$ holds for infinitely many i.

The language of a TBA $\mathcal{A}$ is

$$\mathcal{L}(\mathcal{A}) = \{w \in \Sigma^\omega | \ \mathcal{A} \text{ has an initial fair trace over } w\}.$$

A *finite backward trace* of a TBA $\mathcal{A}$ starting in a state $q \in Q$ over a finite word $w = \sigma_0\sigma_1 \ldots \sigma_{i-1} \in \Sigma^i$ is a finite sequence of transitions

$$\tau = (s_0, \sigma_0, d_0)(s_1, \sigma_1, d_1) \ldots (s_{i-1}, \sigma_{i-1}, d_{i-1}) \in \delta^i$$

such that $d_0 = q$ and for every $0 \leq j < (i-1)$, $s_j = d_{j+1}$.

An *infinite backward trace* of a TBA $\mathcal{A}$ starting in a state $q \in Q$ over an infinite word $w = \sigma_0\sigma_1 \ldots \in \Sigma^\omega$ is an infinite sequence of transitions

$$\tau = (s_0, \sigma_0, d_0)(s_1, \sigma_1, d_1) \ldots \in \delta^\omega$$

such that $d_0 = q$ and for every $i \geq 0$, $s_i = d_{i+1}$.

Given two (backward or forward) traces $\tau, \pi \in \delta^x$, where $x \in \mathbb{N} \cup \{\omega\}$, we say that τ is *at least as accepting as* π ($\tau \geq_{acc} \pi$) if $\pi[i] \in F \implies \tau[i] \in F$ for every $i < |\tau|$.

1.3 Simulation relations on TBA

Simulations [1] are relations on the states of an automaton. These relations tell us whether any behaviour of the automaton beginning in one state can somehow be replicated when beginning in another state. In particular, simulations are utilized to tell us how words are accepted. Some of these relations can be used for *pruning* and *quotienting*[3].

We will define simulations on TBA by simplifying simulations on TGBA introduced by Zbončáková [5].

Forward simulation

Let $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$ be a TBA. A (*forward*) *simulation* is a binary relation on the states of $\mathcal{A}$. Let $p_0, r_0 \in Q$ be arbitrary states. Forward simulation between p_0 and r_0 can be described as a game of two players, Spoiler (he) and Duplicator (she), where Duplicator tries to prove that state r_0 can imitate any behaviour of p_0 while Spoiler tries to disprove it. The game starts in configuration $(p_0, r_0) \in Q \times Q$. In the i-th round, when the configuration is (p_{i-1}, r_{i-1}), Spoiler chooses a transition $(p_{i-1}, \sigma_{i-1}, p_i) \in \delta$. Then it is Duplicator's turn and she chooses a transition $(r_{i-1}, \sigma_{i-1}, r_i) \in \delta$ under the same symbol and that is the end of i-th round. The game is now in configuration (p_i, r_i) and $(i+1)$-th round begins. If the game gets to a point, where one of the players cannot make a move, then the other player wins. Otherwise, if the game goes on forever, the players build two infinite traces. Spoiler builds a trace

$$\pi_S = (p_0, \sigma_0, p_1)(p_1, \sigma_1, p_2) \ldots \in \delta^\omega$$

and Duplicator builds trace

$$\pi_D = (r_0, \sigma_0, r_1)(r_1, \sigma_1, r_2) \ldots \in \delta^\omega.$$

The outcome of such game is described by the winning condition $\mathcal{C}^{di}(\pi_S, \pi_D)$ for Duplicator:

$$\mathcal{C}^{di}(\pi_S, \pi_D) \iff \pi_D \geq_{acc} \pi_S$$

We define a forward simulation as a relation $\sqsubseteq \subseteq Q \times Q$, where $p_0 \sqsubseteq r_0$ (r_0 simulates p_0) holds if Duplicator has a winning strategy in a forward simulation game beginning in a configuration (p_0, r_0).

Backward simulation

Backward simulation is similar to forward simulation, but in the game which describes them, players go by the transitions in the opposite direction. The backward simulation between states $p_0 \in Q$ and $r_0 \in Q$ is again defined in terms of a game of Spoiler and Duplicator, beginning in configuration $(p_0, r_0) \in Q \times Q$. In the i-th round, when the game is in configuration (p_{i-1}, r_{i-1}), Spoiler chooses a transition $(p_i, \sigma_{i-1}, p_{i-1}) \in \delta$, to which Duplicator reacts by choosing a transition $(r_i, \sigma_{i-1}, r_{i-1}) \in \delta$ under the same symbol and the game continues with new configuration (p_i, r_i). Again, if at some point in the game, one of the players fails to make a move, the other player wins. Otherwise, Spoiler and Duplicator create two infinite backward traces

$$\pi_S = (p_1, \sigma_0, p_0)(p_2, \sigma_1, p_1) \ldots \in \delta^\omega$$
$$\pi_D = (r_1, \sigma_0, r_0)(r_2, \sigma_1, r_1) \ldots \in \delta^\omega$$

respectively. In backward simulation, it is not sufficient for Duplicator to match only accepting transitions, she also has to match initial state. Hence the winning condition $\mathcal{C}^{di}_{bw}(\pi_S, \pi_D)$ for Duplicator is

$$\mathcal{C}^{di}_{bw}(\pi_S, \pi_D) \iff \pi_D \geq_{acc} \pi_S \wedge \forall (i \geq 0).(p_i = q_0 \Rightarrow r_i = q_0).$$

We define a backward simulation as a relation $\preceq \subseteq Q \times Q$, where $p_0 \preceq r_0$ holds if Duplicator has a winning strategy in a backward simulation game beginning in a configuration (p_0, r_0).

1.4 Pruning of TBA

Pruning [2, 3] is a method which reduces the size of an automaton by removing transitions. It removes certain transitions, because there are some other transitions which "dominate" them.

Consider a TBA $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$ and a relation on its transitions $< \subseteq \delta \times \delta$. We define the *pruned automaton* $Prune(\mathcal{A}, <) = (Q, \Sigma, \delta', q_0, F \cap \delta')$, where

$$\delta' = max <= \{(p, \sigma, r) \in \delta \mid \nexists (p', \sigma', r') \in \delta \,.\, (p, \sigma, r) < (p', \sigma', r')\}.$$

We say that a relation $< \subseteq \delta \times \delta$ is *good for pruning* (*GFP*) if $\mathcal{L}(\mathcal{A}) = \mathcal{L}(Prune(\mathcal{A}, <))$, for each automaton $\mathcal{A}$.

Since simulations are relations on states and pruning utilizes relations on transitions, we have to somehow transform these relations on states to relations on transitions. Given two binary relations $R_b, R_f \subseteq Q \times Q$ on states, we define

$$P(R_b, R_f) = \big\{\big((p, \sigma, q), (p', \sigma, q')\big) \in \delta_{\mathsf{h}} \times \delta_{\mathsf{h}} \mid (p, p') \in R_b \wedge \\ \wedge (q, q') \in R_f \wedge (p', \sigma, q') \geq_{acc} (p, \sigma, q)\big\}.$$

Note that in publication by Clemente and Mayr [3], similar definition was stated, however since we work with transition based acceptance, the additional condition $(p', \sigma, q') \geq_{acc} (p, \sigma, q)$ needs to be added.

Theorem 1. *For every strict partial order $< \subseteq X$ for $X \in \{\sqsubseteq, \preceq\}$, $P(id, <)$ is GFP.* [2, 3]

1.5 Quotienting of TBA

Another technique how to obtain smaller automata is quotienting. This method identifies states that are equal (based on a given relation) and "merges them together".

Let $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$ be a TBA and $\approx \subseteq Q \times Q$ an equivalence on its states. For any state $q \in Q$, by $[q] = \{p \in Q \mid p \approx q\}$ we denote the equivalence class of q with respect to $\approx$. Further, for a set $P \subseteq Q$, $[P] = \{[p] \mid p \in P\}$ is the set of equivalence classes of elements from the set P.

Then we define the *quotient* of an automaton $\mathcal{A}$ by a relation $\approx$ as the tuple $\mathcal{A}/\approx = ([Q], \Sigma, \delta', [q_0], F')$, where

$$\delta' = \{([p], \sigma, [r]) \mid (p, \sigma, r) \in \delta\},$$
$$F' = \{([p], \sigma, [r]) \mid (p, \sigma, r) \in F\}.$$

We say that a relation $\approx \subseteq Q \times Q$ is *good for quotienting* (*GFQ*) if $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{A}/\approx)$, for each automaton $\mathcal{A}$.

Theorem 2. *For every equivalence $\approx \subseteq X$ for $X \in \{\sqsubseteq, \preceq\}$, $\approx$ is GFQ.* [6]

2 Definitions

2.1 Modal transition-based Büchi automata

A *modal transition-based Büchi automaton* (*MTBA*) $\mathcal{A}$ is a tuple

$$(Q, \Sigma, \delta_{\mathsf{l}}, \delta_{\mathsf{h}}, q_0, F),$$

where

- Q is the set of states,
- Σ is a finite alphabet,
- $\delta_{\mathsf{l}} \subseteq \delta_{\mathsf{h}} \subseteq Q \times \Sigma \times Q$ are the transition relations.
- $q_0 \in Q$ is the initial state, and
- $F \subseteq \delta_{\mathsf{h}}$ is the set of accepting transitions.

We further define the *low-automaton* $\mathcal{A}_{\mathsf{l}} = (Q, \Sigma, \delta_{\mathsf{l}}, q_0, F \cap \delta_{\mathsf{l}})$ of the MTBA $\mathcal{A}$ and the *high-automaton* $\mathcal{A}_{\mathsf{h}} = (Q, \Sigma, \delta_{\mathsf{h}}, q_0, F)$ of the MTBA $\mathcal{A}$.

A *finite/infinite trace* of an MTBA $\mathcal{A}$ over a word w starting in a state $q \in Q$ is a finite/infinite trace of $\mathcal{A}_{\mathsf{h}}$ over w starting in q. A *finite/infinite low-trace* of $\mathcal{A}$ starting in q over w is a finite/infinite trace of $\mathcal{A}_{\mathsf{l}}$ over w starting in q.

The *high-language* of an MTBA $\mathcal{A}$ is $\mathcal{L}(\mathcal{A}_{\mathsf{h}})$ and the *low-language* of an MTBA $\mathcal{A}$ is $\mathcal{L}(\mathcal{A}_{\mathsf{l}})$.

A *finite/infinite backward trace* of an MTBA $\mathcal{A}$ over a word w starting in a state $q \in Q$ is a finite/infinite backward trace of $\mathcal{A}_{\mathsf{h}}$ over w starting in q. A *finite/infinite backward low-trace* of $\mathcal{A}$ starting in q over w is a finite/infinite backward trace of $\mathcal{A}_{\mathsf{l}}$ over w starting in q.

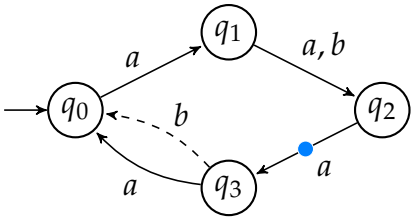
We graphically represent MTBA by a directed graph, where states are vertices, transitions from δ_{l} are solid edges, transitions from $\delta_{\mathsf{h}} \setminus \delta_{\mathsf{l}}$ are dashed edges, symbols belonging to a transition are drawn right beside its edge. The initial state is marked by an incoming arrow and accepting transitions are marked by a blue D.O.T.[1]

Example 2.1.1. Let us look at a simple MTBA and its languages.

1. Dwarfish Orbicular Token

$\mathcal{A} = (Q, \Sigma, \delta_\mathsf{l}, \delta_\mathsf{h}, q_0, F)$, where

$$
\begin{aligned}
Q &= \{q_0, q_1, q_2, q_3\},\\
\Sigma &= \{a, b\},\\
\delta_\mathsf{l} &= \{(q_0, a, q_1), (q_1, a, q_2), (q_1, b, q_2), (q_2, a, q_3), (q_3, a, q_0)\},\\
\delta_\mathsf{h} &= \delta_\mathsf{l} \cup \{(q_3, b, q_0)\},\\
F &= \{(q_2, a, q_3)\}.
\end{aligned}
$$



$$
\begin{aligned}
\mathcal{L}(\mathcal{A}_\mathsf{l}) &= (\{a\}.\{a, b\}.\{aa\})^\omega\\
\mathcal{L}(\mathcal{A}_\mathsf{h}) &= (\{a\}.\{a, b\})^\omega
\end{aligned}
$$

2.2 Simulations on MTBA

It is expected that we want our simulation relations to be as large as possible. We also want those simulations to retain certain properties, so we can use them for reducing automata. There are currently multiple ways how to achieve this:

- change Duplicator's winning condition, obtaining *delayed* [7] and *fair* [8] simulations,
- increase Duplicator's lookahead on Spoiler's trace and eventually come to *trace inclusions*[3],
- increase Duplicator's lookahead only on the input word rather than Spoiler's trace and discover *fixed-word* simulations [9].

Another approach is to allow Duplicator to use more transitions than Spoiler. By working with two sets of transitions, modal automata provide us with a natural way how to achieve this.

Let $\mathcal{A} = (Q, \Sigma, \delta_\mathsf{l}, \delta_\mathsf{h}, q_0, F)$ be an MTBA. For $\mathsf{X} \in \{\mathsf{l}, \mathsf{h}\}^i$ we define

$$\delta_\mathsf{X} = \delta_{\mathsf{X}[0]}.\delta_{\mathsf{X}[1]}.\delta_{\mathsf{X}[2]} \ldots \delta_{\mathsf{X}[i-1]}.$$

Similarly, for $\mathsf{X}' \in \{\mathsf{l}, \mathsf{h}\}^\omega$ we define

$$\delta_{\mathsf{X}'} = \delta_{\mathsf{X}'[0]}.\delta_{\mathsf{X}'[1]}.\delta_{\mathsf{X}'[2]} \dots$$

In the rest of the section, let us fix an MTBA $\mathcal{A} = (Q, \Sigma, \delta_\mathsf{l}, \delta_\mathsf{h}, q_0, F)$ and $\mathsf{X}, \mathsf{Y} \in \{\mathsf{l}, \mathsf{h}\}^\omega$.

Forward X/Y-simulations

We define a (*forward*) X/Y-*simulation* as a relation between states of $\mathcal{A}$. Forward X/Y-simulation between states $p_0 \in Q$ and $r_0 \in Q$ can be described as a game between Spoiler and Duplicator, similarly to the forward simulation on TBA. The game begins in configuration $(p_0, r_0) \in Q \times Q$. In the i-th round, when the game is in configuration (p_{i-1}, r_{i-1}), Spoiler chooses a transition $(p_{i-1}, \sigma_{i-1}, p_i) \in \delta_{\mathsf{X}[i-1]}$ and Duplicator replies by choosing a transition $(r_{i-1}, \sigma_{i-1}, r_i) \in \delta_{\mathsf{Y}[i-1]}$. That is the end of i-th round and the next round begins. If at some point in the game, one of the players cannot make a move, the other player wins. Otherwise, Spoiler and Duplicator build two infinite traces:

$$\pi_S = (p_0, \sigma_0, p_1)(p_1, \sigma_1, p_2) \dots \in \delta_\mathsf{X}$$
$$\pi_D = (r_0, \sigma_0, r_1)(r_1, \sigma_1, r_2) \dots \in \delta_\mathsf{Y},$$

respectively. The winning condition for Duplicator remains the same as in forward simulation on TBA:

$$\mathcal{C}^{di}(\pi_S, \pi_D) \iff \pi_D \geq_{acc} \pi_S$$

We define a forward X/Y-simulation as a relation $\sqsubseteq^\mathsf{X}_\mathsf{Y} \subseteq Q \times Q$, where $p_0 \sqsubseteq^\mathsf{X}_\mathsf{Y} q_0$ holds, if Duplicator has a winning strategy in an X/Y-simulation game starting from configuration (p_0, q_0). Additionally, we define its strict version $\sqsubset^\mathsf{X}_\mathsf{Y} \subseteq Q \times Q$ and symmetric version $\equiv^\mathsf{X}_\mathsf{Y} \subseteq Q \times Q$, where

$$p_0 \sqsubset^\mathsf{X}_\mathsf{Y} q_0 \overset{\text{def}}{\iff} p_0 \sqsubseteq^\mathsf{X}_\mathsf{Y} q_0 \wedge q_0 \not\sqsubseteq^\mathsf{X}_\mathsf{Y} p_0,$$
$$p_0 \equiv^\mathsf{X}_\mathsf{Y} q_0 \overset{\text{def}}{\iff} p_0 \sqsubseteq^\mathsf{X}_\mathsf{Y} q_0 \wedge q_0 \sqsubseteq^\mathsf{X}_\mathsf{Y} p_0.$$

Backward X/Y-simulations

We can also describe a *backward* X/Y-*simulation* between states $p_0 \in Q$ and $r_0 \in Q$ as a game between the well known players. The game again starts in configuration $(p_0, r_0) \in Q \times Q$ and in the i-th round, Spoiler chooses a transition $(p_i, \sigma_{i-1}, p_{i-1}) \in \delta_{\mathsf{X}[i-1]}$ and Duplicator chooses a transition $(r_i, \sigma_{i-1}, r_{i-1}) \in \delta$ under the same symbol and the game continues from configuration (p_i, r_i). If one of the players fails to make a move, the other player wins. Otherwise, the players create two infinite backward traces

$$\pi_S = (p_1, \sigma_0, p_0)(p_2, \sigma_1, p_1) \ldots \in \delta_{\mathsf{X}}$$
$$\pi_D = (r_1, \sigma_0, r_0)(r_2, \sigma_1, r_1) \ldots \in \delta_{\mathsf{Y}}$$

respectively. The winning condition for Duplicator is the same as in backward simulation on TBA:

$$\mathcal{C}^{di}_{bw}(\pi_S, \pi_D) \iff \pi_D \geq_{acc} \pi_S \wedge \forall (i \geq 0).(p_i = q_0 \Rightarrow r_i = q_0).$$

We define a backward X/Y-simulation as a relation $\preceq^{\mathsf{X}}_{\mathsf{Y}} \subseteq Q \times Q$, where $p_0 \preceq^{\mathsf{X}}_{\mathsf{Y}} q_0$ holds, if Duplicator has a winning strategy in a backward X/Y-simulation game starting from configuration (p_0, q_0). Additionally, we define its strict version $\prec^{\mathsf{X}}_{\mathsf{Y}} \subseteq Q \times Q$ and symmetric version $\approx^{\mathsf{X}}_{\mathsf{Y}} \subseteq Q \times Q$, where

$$p_0 \prec^{\mathsf{X}}_{\mathsf{Y}} q_0 \overset{\text{def}}{\iff} p_0 \preceq^{\mathsf{X}}_{\mathsf{Y}} q_0 \wedge q_0 \not\preceq^{\mathsf{X}}_{\mathsf{Y}} p_0,$$
$$p_0 \approx^{\mathsf{X}}_{\mathsf{Y}} q_0 \overset{\text{def}}{\iff} p_0 \preceq^{\mathsf{X}}_{\mathsf{Y}} q_0 \wedge q_0 \preceq^{\mathsf{X}}_{\mathsf{Y}} p_0.$$

Intuitively, in X/Y-simulation, the word X tells Spoiler which transitions can he use in which rounds of the game and Y tells Duplicator which transitions she can use.

Example 2.2.1. $\mathsf{l}^\omega/\mathsf{h}^\omega$-simulation is a simulation, where Spoiler can use only low-transitions and Duplicator can use all the transitions - $\pi_S \in \delta_{\mathsf{l}^\omega} = \delta^\omega_{\mathsf{l}}$, $\pi_D \in \delta_{\mathsf{h}^\omega} = \delta^\omega_{\mathsf{h}}$. In Figure 2.1 is a simple MTBA with the simulation relation denoted.

Since we want to obtain simulation relations that are as large as possible, it comes naturally that we want to make Spoiler "weaker" by allowing him to use less transitions than Duplicator. Hence in this thesis we will mainly focus on X/Y-simulations, where $\delta_{\mathsf{X}} \subseteq \delta_{\mathsf{Y}}$. Especially, we would like to utilize such simulations, where $\delta_{\mathsf{X}} \subsetneq \delta_{\mathsf{Y}}$.

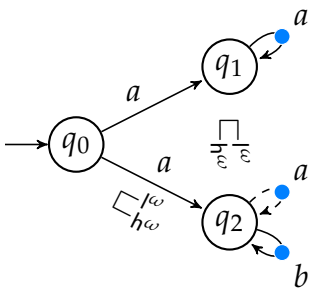


Figure 2.1: Example of $\mathsf{l}^\omega/\mathsf{h}^\omega$-simulation.

3 MTBA pruning

In this chapter we will generalize the definition of pruning to MTBA, explore the X/Y-simulations and their usability with pruning. By the end of the chapter, we will give equivalent conditions for X/Y-simulations to be usable for pruning.

The definition of MTBA pruning is almost identical to the definition of pruning of TBA, given in Section 1.4. Let $\mathcal{A} = (Q, \Sigma, \delta_\mathsf{l}, \delta_\mathsf{h}, q_0, F)$ be an MTBA and $R \subseteq \delta_\mathsf{h} \times \delta_\mathsf{h}$ a relation on its transitions.

The *pruned modal automaton* is

$$Prune_M(\mathcal{A}, R) = Prune(\mathcal{A}_\mathsf{h}, R).$$

A relation $R \subseteq \delta_\mathsf{h} \times \delta_\mathsf{h}$ is *good for modal pruning* (*GFMP*) if

$$\mathcal{L}(\mathcal{A}_\mathsf{l}) \subseteq \mathcal{L}(Prune_M(\mathcal{A}, R)) \subseteq \mathcal{L}(\mathcal{A}_\mathsf{h}).$$

As mentioned earlier, the motivation behind modal simulations is that we want to obtain larger simulation relations by making Spoiler weaker and Duplicator stronger. It is then quite logical that we would like to make use of the $\mathsf{l}^\omega/\mathsf{h}^\omega$-simulation, which forces Spoiler to use only the low-transitions and allows Duplicator to use all of the transitions. Unfortunately, as can be seen in Figure 3.1[1], this simulation is not good for modal pruning. From the simulation game beginning in configuration (q_1, q_2), Spoiler can build the low-trace

1. After pruning, edges marked by $\times$ will be removed and dashed edges will become solid.

$(q_1, a, q_3)(q_3, a, q_3)^\omega$, to which Duplicator will reply with building the trace $(q_2, a, q_4)(q_4, a, q_4)^\omega$. This ensures that after pruning the transition (q_0, a, q_1), the word a^ω can still be accepted. However, in a simulation game beginning in configuration (q_4, q_5), Spoiler cannot utilize the transition $(q_4, a, q_4) \in \delta_{\mathsf{h}} \setminus \delta_{\mathsf{l}}$ meaning that Duplicator does not need to have any transition under the symbol a available. Hence after pruning the transition (q_2, a, q_4), the word a^ω is no longer accepted.

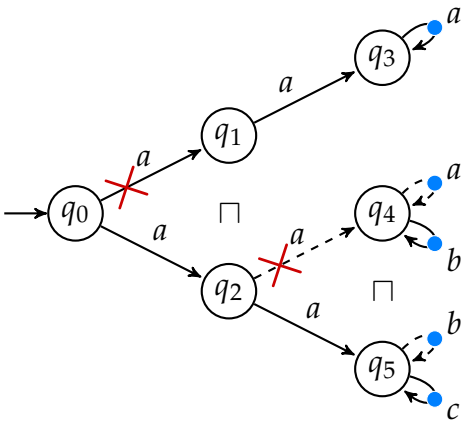


Figure 3.1: There exists a strict partial order $\sqsubset \subseteq \sqsubseteq^{\mathsf{l}^\omega}_{\mathsf{h}^\omega}$ such that $P(id, \sqsubset)$ is not GFMP.

Further analysis of this counterexample invokes the following thought: If we want a simulation that will be good for modal pruning, then Duplicator has to use only such traces whose proper suffixes can be utilized by Spoiler. This means that we want the inclusion $\delta_{\mathsf{X}} \supseteq \delta_{\mathsf{Y}[i..]}$ to hold for every $i > 0$. As it turns out, given $\delta_{\mathsf{X}} \subseteq \delta_{\mathsf{Y}}$, the condition $\delta_{\mathsf{X}} \supseteq \delta_{\mathsf{Y}[1..]}$ is equivalent to the previous one. Moreover, as will be shown, this condition is not only necessary, but also sufficient. We start with three auxiliary lemmas.

Lemma 3. *Let* $\mathsf{X}, \mathsf{Y} \in \{\mathsf{l}, \mathsf{h}\}^\omega$. *If* $\delta_{\mathsf{X}} \not\supseteq \delta_{\mathsf{Y}}$, *then there exists an index* $j \geq 0$ *such that* $\mathsf{X}[j] = \mathsf{l}$ *and* $\mathsf{Y}[j] = \mathsf{h}$.

Proof. The proof will be done by contraposition. Let us assume that for every $j \geq 0$, $(\mathsf{X}[j], \mathsf{Y}[j]) \in \{(\mathsf{l}, \mathsf{l}), (\mathsf{h}, \mathsf{l}), (\mathsf{h}, \mathsf{h})\}$. Therefore for every $j \geq 0$, $\delta_{\mathsf{X}[j]} \supseteq \delta_{\mathsf{Y}[j]}$, thus $\delta_{\mathsf{X}} \supseteq \delta_{\mathsf{Y}}$. □

Lemma 4. *Let* $\mathsf{X}, \mathsf{Y} \in \{\mathsf{l}, \mathsf{h}\}^\omega$ *such that* $\delta_\mathsf{X} \subseteq \delta_\mathsf{Y}$. *If* $\delta_\mathsf{X} \supseteq \delta_{\mathsf{Y}[1..]}$, *then* $\delta_\mathsf{h}.\delta_\mathsf{X} \supseteq \delta_\mathsf{X}$.

Proof. The proof will be done by contradiction. Let $\delta_\mathsf{X} \supseteq \delta_{\mathsf{Y}[1..]}$ and $\delta_\mathsf{h}.\delta_\mathsf{X} = \delta_{\mathsf{h}.\mathsf{X}} \not\supseteq \delta_\mathsf{X}$. From Lemma 3 there exists an index $j \geq 0$ such that $(\mathsf{h}.\mathsf{X})[j] = \mathsf{l}$ and $\mathsf{X}[j] = \mathsf{h}$, hence $\mathsf{X}[j-1] = \mathsf{l}$. From the inclusion $\delta_\mathsf{X} \supseteq \delta_{\mathsf{Y}[1..]}$ we obtain $\mathsf{Y}[j] = \mathsf{Y}[1..][j-1] = \mathsf{l}$. Therefore $\delta_{\mathsf{X}[j]} = \delta_\mathsf{h}$ and $\delta_{\mathsf{Y}[j]} = \delta_\mathsf{l}$ implying $\delta_\mathsf{X} \not\subseteq \delta_\mathsf{Y}$. $\lightning$ □

Corollary 5. *Let* X, Y *be as in Lemma 4. If* $\delta_\mathsf{X} \supseteq \delta_{\mathsf{Y}[1..]}$, *then for every* $j > 0$, $\delta_{\mathsf{X}[..j]^R} \subseteq \delta_{\mathsf{X}[..(j-1)]^R}.\delta_\mathsf{h}$.

Proof.

$$\begin{aligned}
&\delta_\mathsf{h}.\delta_\mathsf{X} \supseteq \delta_\mathsf{X} \implies \delta_{\mathsf{h}.\mathsf{X}} \supseteq \delta_\mathsf{X} \implies \delta_{(\mathsf{h}.\mathsf{X})[..j]} \supseteq \delta_{\mathsf{X}[..j]} \implies \\
&\quad \implies \delta_{\mathsf{h}.\mathsf{X}[..(j-1)]} \supseteq \delta_{\mathsf{X}[..j]} \implies \delta_{(\mathsf{h}.\mathsf{X}[..(j-1)])^R} \supseteq \delta_{\mathsf{X}[..j]^R} \implies \\
&\quad\quad \implies \delta_{\mathsf{X}[..(j-1)]^R.\mathsf{h}} \supseteq \delta_{\mathsf{X}[..j]^R} \implies \delta_{\mathsf{X}[..(j-1)]^R}.\delta_\mathsf{h} \supseteq \delta_{\mathsf{X}[..j]^R}.
\end{aligned}$$

□

Lemma 6. *Let* $\mathcal{A} = (Q, \Sigma, \delta, q_0, F), \mathcal{B} = (Q', \Sigma, \delta', q_0', F')$ *be TBAs,* $w \in \mathcal{L}(\mathcal{A})$ *an infinite word,* $\rho \in \delta^\omega$ *an initial fair trace of* $\mathcal{A}$ *over* w *and*

$$\{\xi_i \in \delta'^i \mid i \geq 0\}$$

an infinite set of initial finite traces of $\mathcal{B}$ *such that* $\xi_i = \xi_{i+1}[..i], \xi_i \geq_{acc} \rho[..i]$ *and* ξ_i *is a trace over* $w[..i]$. *Then* $w \in \mathcal{L}(\mathcal{B})$.

Proof. Let us construct the following infinite sequence of transitions

$$\zeta = \left(\xi_{i+1}[i]\right)_{i=0}^{\infty}.$$

- ζ is an infinite trace of $\mathcal{B}$, since for every $i \geq 0, \zeta[i] = \xi_{i+1}[i] = \xi_{i+2}[i]$, $\zeta[i+1] = \xi_{i+2}[i+1]$ and ξ_{i+2} is a finite trace, thus $\zeta[i]$ ends in the same state in which $\zeta[i+1]$ starts.
- ζ is initial, because $\zeta[0] = \xi_1[0]$ and ξ_1 is an initial trace.
- ζ is a trace over w, since for every $i \geq 0, \zeta[i] = \xi_{i+1}[i]$ is a transition under $w[i]$.

- ζ is fair, because for every $i \geq 0, \zeta[i] = \xi_{i+1}[i] \geq_{acc} \rho[i]$ and ρ is fair.

Hence ζ is an initial fair trace of $\mathcal{B}$ over w, therefore $w \in \mathcal{L}(\mathcal{B})$ □

Let us consider an MTBA $\mathcal{A} = (Q, \Sigma, \delta_\mathsf{l}, \delta_\mathsf{h}, q_0, F)$ and a fixed relation $R \subseteq \delta_\mathsf{h} \times \delta_\mathsf{h}$ on its transitions and $n \geq 0$. Then let

$$\mathcal{B} = Prune_M(\mathcal{A}, R) = (Q, \Sigma, \delta', q_0, F').$$

We say that a transition $(p, \sigma, q) \in \delta_\mathsf{h}$ is *safe* if $(p, \sigma, q) \in \delta'$.
We say that a trace $\rho \in \delta_\mathsf{h}^\omega$ is *n-safe* if $\rho \in \delta'^n.\delta_\mathsf{h}^\omega$.
We say that a trace $\rho \in \delta_\mathsf{h}^\omega$ is *safe* if $\rho \in \delta'^\omega$.

Theorem 7. *Let $\mathsf{X}, \mathsf{Y} \in \{\mathsf{l}, \mathsf{h}\}^\omega$ such that $\delta_\mathsf{X} \subseteq \delta_\mathsf{Y}$. Then for every strict partial order $\sqsubset \subseteq \sqsubset_\mathsf{Y}^\mathsf{X}$, $P(id, \sqsubset)$ is GFMP if and only if $\delta_\mathsf{X} \supseteq \delta_{\mathsf{Y}[1..]}$.*

Proof. „ $\Longrightarrow$ ”

We will prove the contraposition of the implication: *if $\delta_\mathsf{X} \not\supseteq \delta_{\mathsf{Y}[1..]}$, then there exists a strict partial order $\sqsubset \subseteq \sqsubset_\mathsf{Y}^\mathsf{X}$ that is not GFMP*. From Lemma 3 it follows that there exists an index $k \geq 0$ such that $\delta_{\mathsf{X}[k]} = \delta_\mathsf{l}$, $\delta_{\mathsf{Y}[1..][k]} = \delta_{\mathsf{Y}[k+1]} = \delta_\mathsf{h}$. Let us then construct the automaton in Figure 3.2. The initial automaton contains the word a^ω in its low-language; however, after pruning the transitions (q_0, a, q_1) and (p_1, a, r_0), the resulting TBA no longer accepts a^ω.

„ $\Longleftarrow$ ”

Let $\mathcal{A} = (Q, \Sigma, \delta_\mathsf{l}, \delta_\mathsf{h}, q_0, F)$ be an MTBA and

$$\mathcal{B} = Prune_M(\mathcal{A}, P(id, \sqsubset)) = (Q, \Sigma, \delta', q_0, F).$$

The inclusion $\mathcal{L}(\mathcal{B}) \subseteq \mathcal{L}(\mathcal{A}_\mathsf{h})$ holds trivially, since by removing transitions from an automaton one cannot expand its language.

Let us show $\mathcal{L}(\mathcal{A}_\mathsf{l}) \subseteq \mathcal{L}(\mathcal{B})$. Let $w = \sigma_0\sigma_1\sigma_2 \ldots \in \mathcal{L}(\mathcal{A}_\mathsf{l})$ be arbitrary and $\rho = (q_0, \sigma_0, q_1)(q_1, \sigma_1, q_2)(q_2, \sigma_2, q_3) \ldots \in \delta_\mathsf{l}^\omega$ be an initial fair low-trace of $\mathcal{A}$ over w. We will show by induction that for every $i \geq 0$, there exists an initial i-safe trace $\xi_i \in \delta'^i.\delta_\mathsf{h}.\delta_\mathsf{X}$ of $\mathcal{A}$ over w such that $\xi_i \geq_{acc} \rho$.

For the base of induction, let $i = 0$ and $\xi_0 = \rho \in \delta_\mathsf{l}^\omega \subseteq \delta_\mathsf{h}.\delta_\mathsf{X}$.

For the induction step, let us assume that $i > 0$ and there exists an initial $(i-1)$-safe trace[2]

$$\xi_{i-1} = (p_0, \sigma_0, p_1)\dots(p_{i-2}, \sigma_{i-2}, p_{i-1}).(p_{i-1}, \sigma_{i-1}, p_i).\\ .(p_i, \sigma_i, p_{i+1})(p_{i+1}, \sigma_{i+1}, p_{i+2})\dots \in \delta'^{i-1}.\delta_{\mathsf{h}}.\delta_{\mathsf{X}},$$

that is at least as accepting as ρ. If ξ_{i-1} is also i-safe ($(p_{i-1}, \sigma_{i-1}, p_i) \in \delta'$), then let $\xi_i = \xi_{i-1}$. From Lemma 4 we obtain

$$\xi_i = \xi_{i-1} \in \delta'^{i-1}.\delta'.\delta_{\mathsf{X}} \subseteq \delta'^{i}.\delta_{\mathsf{h}}.\delta_{\mathsf{X}}.$$

Otherwise, there exists (from transitivity of $\sqsubset$) a safe transition $(p_{i-1}, \sigma_{i-1}, r_i) \geq_{acc} (p_{i-1}, \sigma_{i-1}, p_i)$ such that $p_i \sqsubseteq^{\mathsf{X}}_{\mathsf{Y}} r_i$, which means that for the trace

$$\xi_{i-1}[i..] = (p_i, \sigma_i, p_{i+1})(p_{i+1}, \sigma_{i+1}, p_{i+2})\dots \in \delta_{\mathsf{X}}$$

there exists a trace

$$\pi = (r_i, \sigma_i, r_{i+1})(r_{i+1}, \sigma_{i+1}, r_{i+2})\dots \in \delta_{\mathsf{Y}}$$

over the same word, that is at least as accepting as $\xi_{i-1}[i..]$. Thus we may construct a trace

$$\xi_i = \xi_{i-1}[..(i-1)].(p_{i-1}, \sigma_{i-1}, r_i).\pi \in \delta'^{i-1}.\delta'.\delta_{\mathsf{Y}}$$

and since $\delta_{\mathsf{X}} \supseteq \delta_{\mathsf{Y}[1..]}$, then $\delta_{\mathsf{h}}.\delta_{\mathsf{X}} \supseteq \delta_{\mathsf{h}}.\delta_{\mathsf{Y}[1..]} \supseteq \delta_{\mathsf{Y}[0]}.\delta_{\mathsf{Y}[1..]} = \delta_{\mathsf{Y}}$, hence $\xi_i \in \delta'^{i}.\delta_{\mathsf{h}}.\delta_{\mathsf{X}}$. Moreover, ξ_i will be initial, i-safe and $\xi_i \geq_{acc} \rho$, since

- $\xi_{i-1}[..(i-1)] \geq_{acc} \rho[..(i-1)]$,
- $(p_{i-1}, \sigma_{i-1}, r_i) \geq_{acc} (p_{i-1}, \sigma_{i-1}, p_i) \geq_{acc} \rho[i-1]$, and
- $\pi \geq_{acc} \xi_{i-1}[i..] \geq_{acc} \rho[i..]$.

Moreover note that ξ_{i-1} and ξ_i are identical on the first $i-1$ transitions.

Considering the following set $\{\xi_i[..i] \in \delta'^{i} \mid i \geq 0\}$, from Lemma 6 it follows that $w \in \mathcal{L}(\mathcal{B})$.

□

2. *Note:* The sole purpose of colors is to increase readability.

Similarly, it can be shown that the same statement holds for backward X/Y-simulations.

We start with a lemma.

Lemma 8. *Let* $\mathsf{X}, \mathsf{Y} \in \{\mathsf{l}, \mathsf{h}\}^\omega$ *such that* $\delta_\mathsf{X} \subseteq \delta_\mathsf{Y}$ *and* $\delta_\mathsf{X} \supseteq \delta_{\mathsf{Y}[1..]}$. *Let* $\mathcal{A} = (Q, \Sigma, \delta_\mathsf{l}, \delta_\mathsf{h}, q_0, F)$ *be an MTBA,* $\prec \subseteq \preceq^\mathsf{X}_\mathsf{Y}$ *be a strict partial order,*

$$\mathcal{B} = Prune_M(\mathcal{A}, P(\prec, id)) = (Q, \Sigma, \delta', q_0, F'),$$

$w = \sigma_0\sigma_1\sigma_2 \ldots \in \mathcal{L}(\mathcal{A}_\mathsf{l})$ *and* $\rho = (q_0, \sigma_0, q_1)(q_1, \sigma_1, q_2) \ldots \in \delta_\mathsf{l}^\omega$ *an initial fair low-trace of* $\mathcal{A}$ *over* w. *Then for every* $i \geq 0$, *there exists an initial* i*-safe trace* ρ_i *of* $\mathcal{A}$ *over* w *such that* ρ_i *is at least as accepting as* ρ.

Proof. Let $i \geq 0$ be arbitrary. We will show by induction that for every $0 < j \leq i$, there exists an initial trace $\tau_{j,i} \in \delta_{\mathsf{X}[..(j-1)]^R}.\delta_\mathsf{h}.\delta'^{i-j}.\delta_\mathsf{l}^\omega$ over w, that is at least as accepting as ρ.

For the base of induction, let $j = i$ and

$$\tau_{i,i} = \rho \in \delta_\mathsf{l}^\omega \subseteq \delta_{\mathsf{X}[..(i-1)]^R}.\delta_\mathsf{h}.\delta_\mathsf{l}^\omega.$$

For the induction step, let $0 < j < i$ and let us assume, that there exists an initial trace[3]

$$\tau_{j+1,i} = (p_0, \sigma_0, p_1)(p_1, \sigma_1, p_2) \ldots (p_{j-1}, \sigma_{j-1}, p_j).(p_j, \sigma_j, p_{j+1}).$$
$$.(p_{j+1}, \sigma_{j+1}, p_{j+2}) \ldots \in \delta_{\mathsf{X}[..j]^R}.\delta_\mathsf{h}.\delta'^{i-j-1}.\delta_\mathsf{l}^\omega$$

with $\tau_{j+1,i} \geq_{acc} \rho$. If (p_j, σ_j, p_{j+1}) is safe, then let $\tau_{j,i} = \tau_{j+1,i}$ and from Corollary 5 we obtain $\delta_{\mathsf{X}[..j]^R} \subseteq \delta_{\mathsf{X}[..(j-1)]^R}.\delta_\mathsf{h}$, hence

$$\tau_{j,i} = \tau_{j+1,i} \in \delta_{\mathsf{X}[..(j-1)]^R}.\delta_\mathsf{h}.\delta'.\delta'^{i-j-1}.\delta_\mathsf{l}^\omega.$$

If (p_j, σ_j, p_{j+1}) is not safe, then there exists a transition $(r_j, \sigma_j, p_{j+1}) \geq_{acc} (p_j, \sigma_j, p_{j+1})$ such that $p_j \prec r_j$. From transitivity of $\prec$, we may assume that (r_j, σ_j, p_{j+1}) is safe. Since $\prec \subseteq \preceq^\mathsf{X}_\mathsf{Y}$, then for the finite initial trace

$$\tau_{j+1,i}[..j] = (p_0, \sigma_0, p_1)(p_1, \sigma_1, p_2) \ldots (p_{j-1}, \sigma_{j-1}, p_j) \in \delta_{\mathsf{X}[..j]^R}$$

there exists a finite initial trace

$$\pi = (r_0, \sigma_0, r_1)(r_1, \sigma_1, r_2) \ldots (r_{j-1}, \sigma_{j-1}, r_j) \in \delta_{\mathsf{Y}[..j]^R}$$

3. *Note:* The sole purpose of colors is to increase readability.

for which $\pi \geq_{acc} \tau_{j+1,i}[..j]$. And since

$$\begin{aligned}\delta_{\mathsf{X}} \supseteq \delta_{\mathsf{Y}[1..]} &\implies \delta_{\mathsf{h}}.\delta_{\mathsf{X}} \supseteq \delta_{\mathsf{Y}[0]}.\delta_{\mathsf{Y}[1..]} \implies \delta_{\mathsf{h}.\mathsf{X}} \supseteq \delta_{\mathsf{Y}} \implies \\ &\implies \delta_{(\mathsf{h}.\mathsf{X})[..j]} \supseteq \delta_{\mathsf{Y}[..j]} \implies \delta_{\mathsf{h}.\mathsf{X}[..(j-1)]} \supseteq \delta_{\mathsf{Y}[..j]} \implies \\ &\implies \delta_{\mathsf{X}[..(j-1)]^R.\mathsf{h}} \supseteq \delta_{\mathsf{Y}[..j]^R} \implies \delta_{\mathsf{X}[..(j-1)]^R}.\delta_{\mathsf{h}} \supseteq \delta_{\mathsf{Y}[..j]^R},\end{aligned}$$

then $\pi \in \delta_{\mathsf{X}[..(j-1)]^R}.\delta_{\mathsf{h}}$ and we may construct a new initial trace

$$\begin{aligned}\tau_{j,i} &= \pi.(r_j, \sigma_j, p_{j+1}).\tau_{j+1,i}[(j+1)..] = \\ &= (r_0, \sigma_0, r_1)(r_1, \sigma_1, r_2)\dots(r_{j-1}, \sigma_{j-1}, r_j).(r_j, \sigma_j, p_{j+1}). \\ &\quad .(p_{j+1}, \sigma_{j+1}, p_{j+2})\dots \in \delta_{\mathsf{X}[..(j-1)]^R}.\delta_{\mathsf{h}}.\delta'.\delta'^{i-j-1}.\delta_{\mathsf{l}}^{\omega},\end{aligned}$$

that will be at least as accepting as ρ, since

- $\pi \geq_{acc} \tau_{j+1,i}[..j] \geq_{acc} \rho[..j]$,
- $(r_j, \sigma_j, p_{j+1}) \geq_{acc} (p_j, \sigma_j, p_{j+1}) \geq_{acc} \rho[j]$,
- $\tau_{j+1,i}[(j+1)..] \geq_{acc} \rho[(j+1)..]$.

Because we assume only one initial state q_0, there does not exist any state $s_0 \neq q_0$ such that $q_0 \preceq_{\mathsf{Y}}^{\mathsf{X}} s_0$. Hence, there cannot exist any transition that would "dominate" the transition $\tau_{1,i}[0]$. Therefore, $\tau_{1,i}[0]$ is safe and $\tau_{1,i}$ is i-safe. Thus we may set $\rho_i = \tau_{1,i}$. □

Theorem 9. *Let $\mathsf{X}, \mathsf{Y} \in \{\mathsf{l}, \mathsf{h}\}^{\omega}$ such that $\delta_{\mathsf{X}} \subseteq \delta_{\mathsf{Y}}$. Then for every strict partial order $\prec \subseteq \preceq_{\mathsf{Y}}^{\mathsf{X}}$, $P(id, \prec)$ is GFMP if and only if $\delta_{\mathsf{X}} \supseteq \delta_{\mathsf{Y}[1..]}$.*

Proof. „ $\implies$ ”

We will prove the contraposition of this implication: *if $\delta_{\mathsf{X}} \not\supseteq \delta_{\mathsf{Y}[1..]}$, then there exists a strict partial order $\prec \subseteq \preceq_{\mathsf{Y}}^{\mathsf{X}}$ that is not GFMP*. From Lemma 3 it follows that there exists an index $k \geq 0$ such that $\delta_{\mathsf{X}[k]} = \delta_{\mathsf{l}}$, $\delta_{\mathsf{Y}[1..][k]} = \delta_{\mathsf{Y}[k+1]} = \delta_{\mathsf{h}}$. Let us then construct the automaton in Figure 3.3. The initial automaton contains the word a^{ω} in its low-language; however, after pruning the transitions (q_1, a, q_3) and (r_k, a, q_2), the resulting TBA no longer accepts a^{ω}.

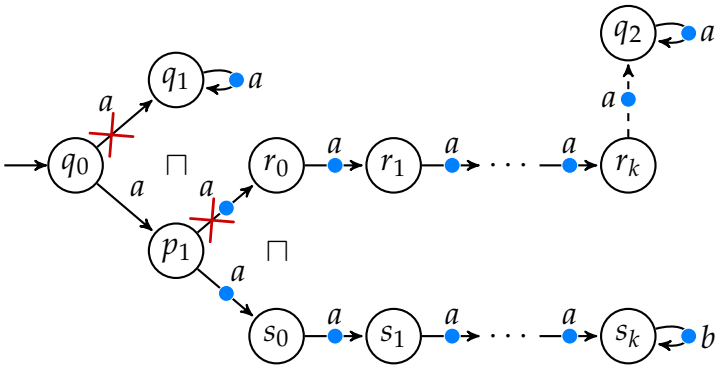


Figure 3.2: If $\mathsf{X}[k] = \mathsf{l}$ and $\mathsf{Y}[k+1] = \mathsf{h}$ for some k and some $\mathsf{X}, \mathsf{Y} \in \{\mathsf{l}, \mathsf{h}\}^\omega$, then there exists a strict partial order $\sqsubset \subseteq \sqsubseteq^{\mathsf{X}}_{\mathsf{Y}}$ such that $P(id, \sqsubset)$ is not GFMP.

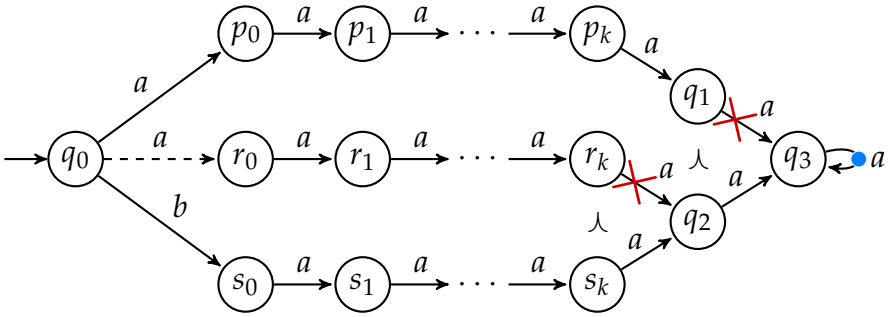


Figure 3.3: If $\mathsf{X}[k] = \mathsf{l}$ and $\mathsf{Y}[k+1] = \mathsf{h}$ for some k and some $\mathsf{X}, \mathsf{Y} \in \{\mathsf{l}, \mathsf{h}\}^\omega$, then there exists a strict partial order $\prec \subseteq \preceq^{\mathsf{X}}_{\mathsf{Y}}$ such that $P(\prec, id)$ is not GFMP.

„ $\Longleftarrow$ ”

Let $\mathcal{A} = (Q, \Sigma, \delta_{\mathsf{l}}, \delta_{\mathsf{h}}, q_0, F)$ be an MTBA and

$$\mathcal{B} = Prune_M(\mathcal{A}, P(\prec, id)) = (Q, \Sigma, \delta', q_0, F').$$

The inclusion $\mathcal{L}(\mathcal{B}) \subseteq \mathcal{L}(\mathcal{A}_{\mathsf{h}})$ holds trivially, since by removing transitions from an automaton one cannot expand its language.

Let us then show $\mathcal{L}(\mathcal{A}_{\mathsf{l}}) \subseteq \mathcal{L}(\mathcal{B})$. Let $w = \sigma_0\sigma_1 \ldots \in \mathcal{L}(\mathcal{A}_{\mathsf{l}})$ be an arbitrary word and let $\rho = (q_0, \sigma_0, q_1)(q_1, \sigma_1, q_2) \ldots \in \delta_{\mathsf{l}}^{\omega}$ be an initial fair low-trace of $\mathcal{A}$ over w.

We say that a finite initial trace $\xi_i \in \delta_{\mathsf{h}}^{i}$ over $w[..i]$ is an *i-good prefix* if $\xi_i \geq_{acc} \rho[..i]$ and for infinitely many j's there exists an inital j-safe trace $\varphi_j \in \delta_{\mathsf{h}}^{\omega}$ over w beginning with the prefix ξ_i ($\varphi_j = \xi_i.\psi$ for suitable $\psi \in \delta_{\mathsf{h}}^{\omega}$) such that $\varphi_j \geq_{acc} \rho$.

We will now show by induction that for every $i \geq 0$, there exists an i-good prefix ξ_i.

For the base of induction, let $i = 0$ and $\xi_0 = \varepsilon$. Using Lemma 8, we obtain that ξ_0 is a 0-good prefix.

For the induction step, let us assume that $i > 0$ and there exists an $(i-1)$-good prefix ξ_{i-1}. Since the automaton is finite, from the pigeonhole principle there is a transition $(p, \sigma_{i-1}, q) \in \delta_{\mathsf{h}}$ such that $\xi_{i-1}.(p, \sigma_{i-1}, q)$ is a finite trace over $w[..i]$ and there are infinitely many j's for which there exists an initial j-safe trace $\varphi_j \in \delta_{\mathsf{h}}^{\omega}$ beginning with the prefix $\xi_{i-1}.(p, \sigma_{i-1}, q)$ such that $\varphi_j \geq_{acc} \rho$, which also implies $(p, \sigma_{i-1}, q) \geq_{acc} \rho[i-1]$ and further $\xi_{i-1}.(p, \sigma_{i-1}, q) \in \delta'^{i}$. Let then $\xi_i = \xi_{i-1}.(p, \sigma_{i-1}, q)$. Moreover, notice that $\xi_i[..(i-1)] = \xi_{i-1}$.

Considering the following set $\{\xi_i \in \delta'^{i} \mid i \geq 0\}$, from Lemma 6 it follows that $w \in \mathcal{L}(\mathcal{B})$. □

For all $\mathsf{X}, \mathsf{Y} \in \{\mathsf{l}, \mathsf{h}\}^{\omega}$ such that $\delta_{\mathsf{X}} \subseteq \delta_{\mathsf{Y}}$, we have successfully characterized all the forward and backward X/Y-simulations that are good for modal pruning. Let us then study modal automata quotienting.

4 MTBA quotienting

In this chapter we will generalize the concept of quotienting to MTBA, introducing two kinds of MTBA quotienting and afterwards, we will investigate the possibilities of using our X/Y-simulations for such quotientings. By the end of the chapter, we will describe all of the simulations usable for one of the quotientings.

We define the two versions of quotienting depending on the type of transitions they preserve. The first version will keep all the transitions and the second one will keep only the low-transitions.

Let $\mathcal{A} = (Q, \Sigma, \delta_\mathsf{l}, \delta_\mathsf{h}, q_0, F)$ be an MTBA and $\approx \subseteq Q \times Q$ an equivalence on its states.

The *high-quotient* of $\mathcal{A}$ by $\approx$ is $\mathcal{A}_\mathsf{h}/\approx$. The *low-quotient* $\mathcal{A}$ by $\approx$ is $\mathcal{A}_\mathsf{l}/\approx$.

An equivalence $\approx$ is *good for high-quotienting* $(GFHQ)$ if

$$\mathcal{L}(\mathcal{A}_\mathsf{l}) \subseteq \mathcal{L}(\mathcal{A}_\mathsf{h}/\approx) \subseteq \mathcal{L}(\mathcal{A}_\mathsf{h}).$$

Likewise, $\approx$ is *good for low-quotienting* $(GFLQ)$ if

$$\mathcal{L}(\mathcal{A}_\mathsf{l}) \subseteq \mathcal{L}(\mathcal{A}_\mathsf{l}/\approx) \subseteq \mathcal{L}(\mathcal{A}_\mathsf{h}).$$

It can be easily shown that low-quotienting "preserves" the low-language of the initial automaton.

Lemma 10. *Let $\mathcal{A} = (Q, \Sigma, \delta_\mathsf{l}, \delta_\mathsf{h}, q_0, F)$ be an MTBA, $\approx \subseteq Q \times Q$ an equivalence on its states and $\mathcal{B} = \mathcal{A}_\mathsf{l}/\approx = ([Q], \Sigma, \delta', [q_0], F')$ the low-quotient of $\mathcal{A}$ by $\approx$. Then $\mathcal{L}(\mathcal{A}_\mathsf{l}) \subseteq \mathcal{L}(\mathcal{B})$.*

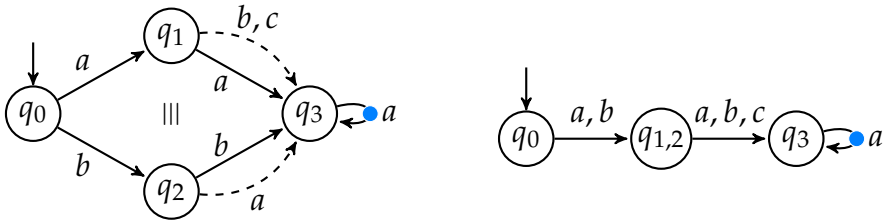
Proof. For every low-trace $\rho = (q_0, \sigma_0, q_1)(q_1, \sigma_1, q_2) \ldots \in \delta_\mathsf{l}^\omega$ of $\mathcal{A}$ there exists by definition a trace $[\rho] = ([q_0], \sigma_0, [q_1])([q_1], \sigma_1, [q_2]) \ldots \in \delta'^\omega$ of $\mathcal{B}$ over the same word, and for every accepting low-transition (q_i, σ_i, q_{i+1}) there exists by definition an accepting transition $([q_i], \sigma_i, [q_{i+1}]) \in F'$.

□

First, we will look at the high-quotienting and discover that it does not work very well with our simulations. After that, we will focus on low-quotienting and formulate theorems characterizing simulations that are GFLQ.

High-quotienting

Similarly to pruning, it is naturally desirable to make use of the $\mathsf{l}^\omega/\mathsf{h}^\omega$-simulation for quotienting. Unfortunately, this simulation is also not GFHQ, as can be seen in the counterexample in Figure 4.1. In the initial automaton, the word $w = bca^\omega$ is not in the high-language of the automaton; however, it is accepted by the high-quotient.



Initial automaton High-quotient

Figure 4.1: There exists an equivalence $\equiv \subseteq \sqsubseteq^{\mathsf{l}^\omega}_{\mathsf{h}^\omega}$ such that $\equiv$ is not GFHQ.

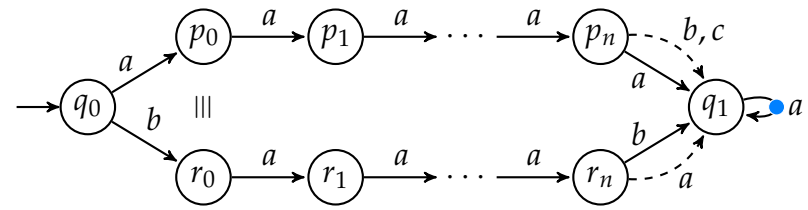
This raises a question what simulations are then good for high-quotienting. Figure 4.2 shows a strong counterexample about every $\mathsf{h}^n.\mathsf{l}.\mathsf{h}^\omega/\mathsf{l}^n.\mathsf{h}.\mathsf{l}^\omega$-simulation for $n \geq 0$. In the initial automaton, the word ba^nca^ω is not in the high-language, but is accepted in the high-quotient.

Corollary 11. *For every* $\mathsf{X}, \mathsf{Y} \in \{\mathsf{l}, \mathsf{h}\}^\omega$ *such that* $\delta_\mathsf{X} \subseteq \delta_\mathsf{Y}$*, if* $\delta_\mathsf{X} \subsetneq \delta_\mathsf{Y}$*, then* $\sqsubseteq^\mathsf{X}_\mathsf{Y}$ *is not good for high-quotienting.*

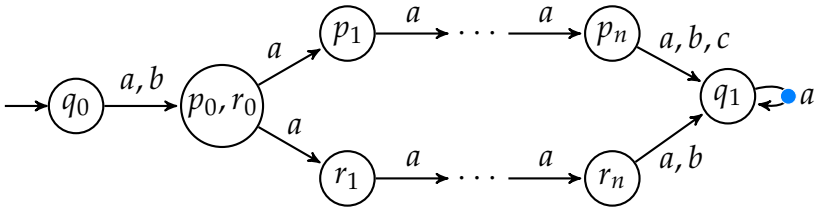
Proof. If $\delta_\mathsf{X} \subsetneq \delta_\mathsf{Y}$, then there exists an index $n \geq 0$ such that $\delta_{\mathsf{X}[n]} = \delta_\mathsf{l}$ and $\delta_{\mathsf{Y}[n]} = \delta_\mathsf{h}$. Note then that in this X/Y-simulation, Spoiler cannot be stronger than in the $\mathsf{h}^n.\mathsf{l}.\mathsf{h}^\omega/\mathsf{l}^n.\mathsf{h}.\mathsf{l}^\omega$-simulation and Duplicator cannot be weaker than in the $\mathsf{h}^n.\mathsf{l}.\mathsf{h}^\omega/\mathsf{l}^n.\mathsf{h}.\mathsf{l}^\omega$-simulation. Hence the inclusion $\sqsubseteq^\mathsf{X}_\mathsf{Y} \supseteq \sqsubseteq^{\mathsf{h}^n.\mathsf{l}.\mathsf{h}^\omega}_{\mathsf{l}^n.\mathsf{h}.\mathsf{l}^\omega}$ holds and thus $\sqsubseteq^\mathsf{X}_\mathsf{Y}$ is not GFHQ. □

The only X/Y-simulations, that are not covered by the previous corollary are those where $\mathsf{X} = \mathsf{Y}$. Let us then inspect the two most basic such simulations: $\mathsf{l}^\omega/\mathsf{l}^\omega$-simulation and $\mathsf{h}^\omega/\mathsf{h}^\omega$-simulation.

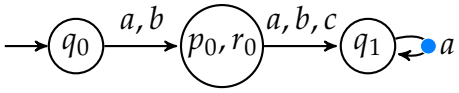
Theorem 12. *For every equivalence* $\equiv \subseteq \sqsubseteq^{\mathsf{h}^\omega}_{\mathsf{h}^\omega}$*,* $\equiv$ *is GFHQ.*



Initial automaton



High-quotient (for $n > 0$)



High-quotient (for $n = 0$)

Figure 4.2: For every $n \geq 0$, there exists an equivalence $\equiv \subseteq \sqsubseteq^{\mathsf{h}^n.\mathsf{l}.\mathsf{h}^\omega}_{\mathsf{l}^n.\mathsf{h}.\mathsf{l}^\omega}$ such that $\equiv$ is not GFHQ.

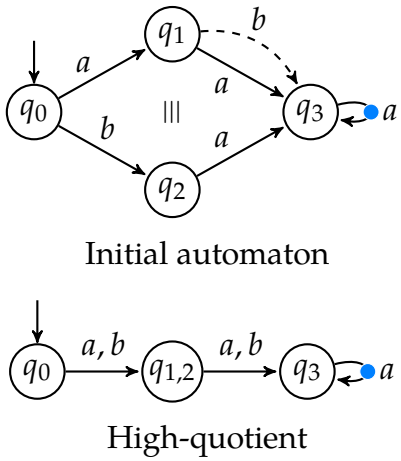


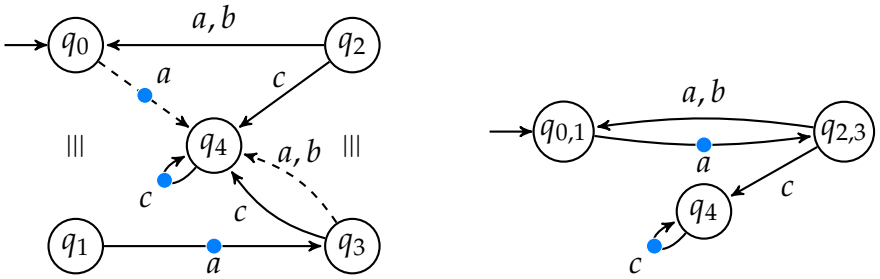
Figure 4.3: There exists an equivalence $\equiv \subseteq \sqsubseteq^{\mathsf{I}^\omega}_{\mathsf{I}^\omega}$ such that $\equiv$ is not GFHQ.

Proof. Let $\mathcal{A}$ be an MTBA. It can be easily observed that the $\mathsf{h}^\omega/\mathsf{h}^\omega$-simulation is equivalent to the forward simulation on the TBA $\mathcal{A}_\mathsf{h}$, which is GFQ (Theorem 2), hence from the inclusion $\equiv \subseteq \sqsubseteq^{\mathsf{h}^\omega}_{\mathsf{h}^\omega} = \sqsubseteq$ follows $\mathcal{L}(\mathcal{A}_\mathsf{h}/\equiv) = \mathcal{L}(\mathcal{A}_\mathsf{h})$. □

Unlike the $\mathsf{h}^\omega/\mathsf{h}^\omega$-simulation, the $\mathsf{I}^\omega/\mathsf{I}^\omega$-simulation is not GFHQ as can be seen in the Figure 4.3. The word bba^ω is accepted by the high-quotient even though it is not contained in the high-language of the initial automaton.

Low-quotienting

In every counterexample in the previous section, we have seen that the problem was, that the high-quotient of an automaton exceeded the high-language of the initial automaton. Since low-quotienting preserves less transitions, it sparks a flame of hope for more positive results. Unfortunately, Figure 4.4 shows a counterexample to why the $\mathsf{I}^\omega/\mathsf{h}^\omega$-simulation is also not good for low-quotienting. The initial automaton does not contain the word a^ω in its high-language; however, this word is accepted by the low-quotient.



Initial automaton Low-quotient

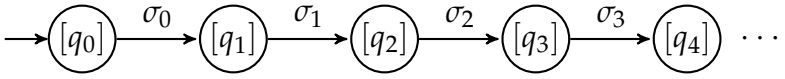
Figure 4.4: There exists an equivalence $\equiv \subseteq \sqsubseteq^{\omega}_{\mathsf{h}^\omega}$ such that $\equiv$ is not GFLQ.

Let us look as the intuition behind how to obain an X/Y-simulation that is GFLQ. We have to be concerned only about the inclusion $\mathcal{L}(\mathcal{B}) \subseteq \mathcal{L}(\mathcal{A}_\mathsf{h})$.

Let $\mathcal{A} = (Q, \Sigma, \delta_\mathsf{l}, \delta_\mathsf{h}, q_0, F)$ be an MTBA, $\equiv \subseteq \sqsubseteq^{\mathsf{X}}_{\mathsf{Y}}$ an equivalence on its states and $\mathcal{B} = \mathcal{A}_\mathsf{l}/\equiv = ([Q], \Sigma, \delta', [q_0], F')$ its low-quotient. Further let $w = \sigma_0\sigma_1\sigma_2 \ldots \in \mathcal{L}(\mathcal{B})$ be arbitrary and let

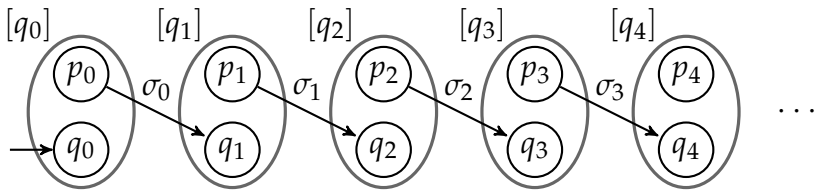
$$\rho = ([q_0], \sigma_0, [q_1])([q_1], \sigma_1, [q_2])([q_2], \sigma_2, [q_3]) \ldots \in \delta'^{\omega}$$

be an initial fair trace of $\mathcal{B}$ over w.

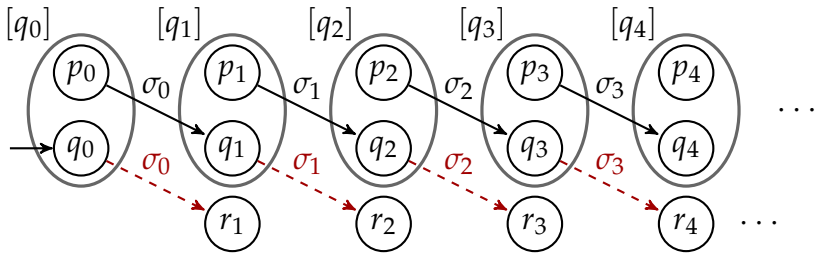


Our goal is to restrict the X/Y-simulation in such a way that there will exist a trace of $\mathcal{A}$ over w beginning in q_0.

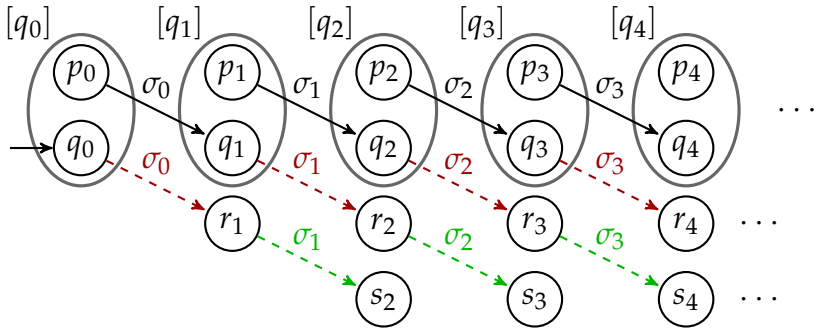
By the definition of low-quotienting, there exists an initial state $q_0 \in [q_0]$ and for every $i \geq 0$ there exist states $p_i \in [q_i]$ and $q_{i+1} \in [q_{i+1}]$ such that there exists a low-transition (p_i, σ_i, q_{i+1}).



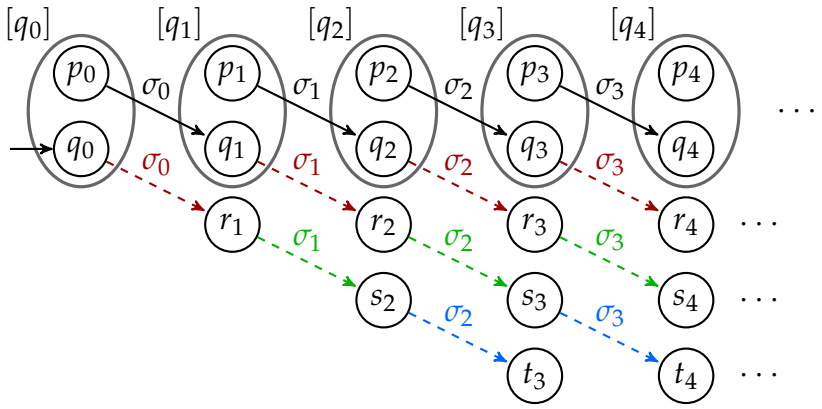
Since Spoiler can always use low-transitions, then for every state p_i, Spoiler can in the first round use the transition (p_i, σ_i, q_{i+1}) and hence, for $p_i \equiv_{\mathsf{Y}}^{\mathsf{X}} q_i$ to hold, Duplicator has to mimic this behaviour by using transition $(q_i, \sigma_i, r_{i+1}) \in \delta_{\mathsf{Y}[0]}$ for suitable $r_{i+1} \in Q$.



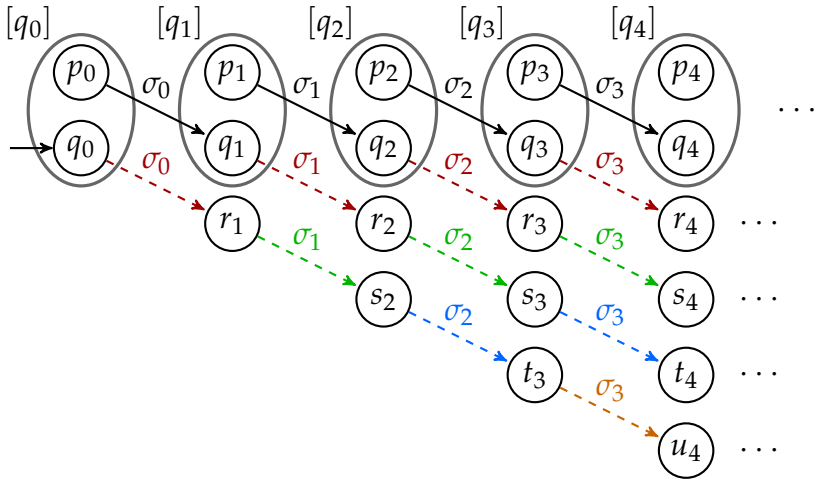
Next, we would like there to be a finite initial trace over $\sigma_0\sigma_1$. If in the simulation game beginning in (p_0, q_0), we allow Spoiler to use the transition $(q_1, \sigma_1, r_2) \in \delta_{\mathsf{Y}[0]}$ in the second round (introducing a restriction $\delta_{\mathsf{X}[1]} \supseteq \delta_{\mathsf{Y}[0]}$), then Duplicator is forced to replicate this by using a transition $(r_1, \sigma_1, s_2) \in \delta_{\mathsf{Y}[1]}$ for suitable $s_2 \in Q$. Likewise, for every game beginning in configuration (p_i, q_i), Duplicator will be forced to use a transition $(r_{i+1}, \sigma_{i+1}, s_{i+2}) \in \delta_{\mathsf{Y}[1]}$ in the second round.



If we want a finite initial trace over $\sigma_0\sigma_1\sigma_2$ beginning in q_i, we allow Spoiler to use in the third round the transition that Duplicator used in the second round in game beginning in configuration (p_{i+1}, q_{i+1}), introducing restriction $\delta_{\mathsf{X}[2]} \supseteq \delta_{\mathsf{Y}[1]}$. This forces Duplicator to find and use transition $(s_{i+2}, \sigma_{i+2}, t_{i+3}) \in \delta_{\mathsf{Y}[2]}$ in the third round.



By restricting $\delta_{\mathsf{X}[3]} \supseteq \delta_{\mathsf{Y}[2]}$, Duplicator is forced to use transitions $(t_{i+3}, \sigma_{i+3}, u_{i+4})$ for $i \geq 0$ in the fourth round.



Inductively, we give restriction $\delta_{\mathsf{X}[i]} \supseteq \delta_{\mathsf{Y}[i-1]}$ for $i > 0$. This can be further modified to obtain $\delta_{\mathsf{X}[1..]} \supseteq \delta_{\mathsf{Y}}$. We will show that this condition is not only necessary but also sufficient. First, we start with two auxiliary lemmas.

Lemma 13. *Let $\mathcal{A} = (Q, \Sigma, \delta_{\mathsf{I}}, \delta_{\mathsf{h}}, q_0, F)$ be an MTBA, $\mathcal{B} = (Q', \Sigma, \delta', q'_0, F')$ a TBA, $w = \sigma_0\sigma_1\sigma_2 \ldots \in \mathcal{L}(\mathcal{B})$, and let $\rho \in \delta'^{\omega}$ be an initial fair trace of $\mathcal{B}$ over w. Additionally, for every $i \geq 0$, there exists a finite initial trace $\tau_i \in \delta_{\mathsf{h}}^i$ of $\mathcal{A}$ over $w[..i]$ that is at least as accepting as $\rho[..i]$. Then $w \in \mathcal{L}(\mathcal{A}_{\mathsf{h}})$.*

Proof. We say that a finite initial trace $\xi_i \in \delta_{\mathsf{h}}^i$ over $w[..i]$ is an *i-good prefix* if $\xi_i \geq_{acc} \rho[..i]$ and for infinitely many $j \geq i$, there exists a finite initial trace $\varphi_j \in \delta_{\mathsf{h}}^j$ beginning with the prefix ξ_i ($\varphi_j = \xi_i.\psi$ for suitable $\psi \in \delta_{\mathsf{h}}^{j-i}$) over $w[..j]$ such that $\varphi_j \geq_{acc} \rho[..j]$. We will show by induction that for every $i \geq 0$, there exists an i-good prefix ξ_i.

For the base of induction, let $i = 0$ and $\xi_0 = \varepsilon$. ξ_0 is a 0-good prefix, following from the assumption of the theorem.

For the induction step, let us assume that $i > 0$ and there exists an $(i-1)$-good prefix ξ_{i-1}. Since the automaton $\mathcal{A}$ is finite, it follows from the pigeonhole principle that there exists a transition $(p, \sigma_{i-1}, q) \in \delta_{\mathsf{h}}$ such that $\xi_{i-1}.(p, \sigma_{i-1}, q)$ is a finite initial trace and there are infinitely many j's for which there exists a finite trace $\varphi_j \in \delta_{\mathsf{h}}^j$ beginning with the prefix $\xi_{i-1}.(p, \sigma_{i-1}, q)$ such that $\varphi_j \geq_{acc} \rho[..j]$, which also implies $(p, \sigma_{i-1}, q) \geq_{acc} \rho[i-1]$. Hence $\xi_i = \xi_{i-1}.(p, \sigma_{i-1}, q)$ is an i-good prefix. Moreover, note that $\xi_i[..(i-1)] = \xi_{i-1}$.

Considering the following set $\{\xi_i \in \delta_{\mathsf{h}}^i \mid i \geq 0\}$, from Lemma 6 it follows that $w \in \mathcal{L}(\mathcal{A}_{\mathsf{h}})$. □

Lemma 14. *Let* $\mathsf{X}, \mathsf{Y} \in \{\mathsf{l}, \mathsf{h}\}^\omega$ *such that* $\delta_{\mathsf{X}} \subseteq \delta_{\mathsf{Y}}$ *and* $\delta_{\mathsf{X}[1..]} \supseteq \delta_{\mathsf{Y}}$. *Let* $\mathcal{A} = (Q, \Sigma, \delta_{\mathsf{l}}, \delta_{\mathsf{h}}, q_0, F)$ *be an MTBA,* $\equiv \subseteq \sqsubseteq_{\mathsf{Y}}^{\mathsf{X}}$ *be an equivalence,* $\mathcal{B} = \mathcal{A}_{\mathsf{l}}/\equiv = ([Q], \Sigma, \delta', [q_0], F')$, $w = \sigma_0\sigma_1\sigma_2 \ldots \in \mathcal{L}(\mathcal{B})$ *and let* $\rho = ([q_0], \sigma_0, [q_1])([q_1], \sigma_1, [q_2]) \ldots \in \delta'^\omega$ *be an initial fair trace of* $\mathcal{B}$ *over* w. *Then for every* $i \geq 0$, *there exists a finite initial trace* $\varphi_i \in \delta_{\mathsf{h}}^i$ *of* $\mathcal{A}$ *over* $w[..i]$ *such that* $\varphi_i \geq_{acc} \rho[..i]$.

Proof. Let $i \geq 0$ be arbitrary. We will show that for every $0 \leq j \leq i$ and every $p_j \in [q_j]$, there exists a finite trace

$$\tau_{j,i} = (p_j, \sigma_j, p_{j+1})(p_{j+1}, \sigma_{j+1}, p_{j+2}) \ldots (p_{i-1}, \sigma_{i-1}, p_i) \in \delta_{\mathsf{Y}[..(i-j)]}$$

over $w[j..i]$ beginning in p_j such that $\tau_{j,i} \geq_{acc} \rho[j..i]$. The proof will be done by induction.

For the base of induction, let $j = i$. This case holds trivially with $\tau_{i,i} = \varepsilon$.

For the induction step, let us assume $0 \leq j < i$. Following the definition of low-quotienting, there exist suitable states $p'_j \in [q_j]$, $p'_{j+1} \in [q_{j+1}]$ such that $(p'_j, \sigma_j, p'_{j+1}) \in \delta_{\mathsf{l}}$ is a low-transition that is at least as

accepting as $([q_j], \sigma_j, [q_{j+1}])$. Further, by the induction hypothesis, we assume that there exists a finite trace

$$\tau_{j+1,i} = (p'_{j+1}, \sigma_{j+1}, p'_{j+2}) \dots (p'_{i-1}, \sigma_{i-1}, p'_i) \in \delta_{\mathsf{Y}[..(i-j-1)]}$$

such that $\tau_{j+1,i} \geq_{acc} \rho[(j+1)..i]$. Let $t_j \in [q_j]$ be arbitrary. Since $t_j \equiv p'_j$ (thus $p'_j \sqsubseteq^{\mathsf{X}}_{\mathsf{Y}} t_j$), then for the finite trace

$$(p'_j, \sigma_j, p'_{j+1}).\tau_{j+1,i} \in \delta_{\mathsf{l}}.\delta_{\mathsf{Y}[..(i-j-1)]} \subseteq \delta_{\mathsf{X}[0]}.\delta_{\mathsf{X}[1..][..(i-j-1)]} = \delta_{\mathsf{X}[..(i-j)]}$$

there exists a finite trace

$$\psi = (t_j, \sigma_j, t_{j+1}) \dots (t_{i-1}, \sigma_{i-1}, t_i) \in \delta_{\mathsf{Y}[..(i-j)]}$$

over the same word. Moreover,

$$\psi \geq_{acc} (p'_j, \sigma_j, p'_{j+1}).\tau_{j+1,i} \geq_{acc} ([q_j], \sigma_j, [q_{j+1}]).\rho[(j+1)..i] = \rho[j..i].$$

By taking $q_0 \in [q_0]$ and $\varphi_i = \tau_{0,i}$ we obtain a finite initial trace of $\mathcal{A}$ over $w[..i]$ for which $\varphi_i \geq_{acc} \rho[..i]$. □

Theorem 15. *Let* $\mathsf{X}, \mathsf{Y} \in \{\mathsf{l}, \mathsf{h}\}^\omega$ *such that* $\delta_{\mathsf{X}} \subseteq \delta_{\mathsf{Y}}$. *Then for every equivalence* $\equiv \subseteq \sqsubseteq^{\mathsf{X}}_{\mathsf{Y}}$, $\equiv$ *is GFLQ if and only if* $\delta_{\mathsf{X}[1..]} \supseteq \delta_{\mathsf{Y}}$.

Proof. „ $\Longrightarrow$ ”

We will prove the contraposition of the implication: if $\delta_{\mathsf{X}[1..]} \not\supseteq \delta_{\mathsf{Y}}$, then there exists an equivalence $\equiv \subseteq \sqsubseteq^{\mathsf{X}}_{\mathsf{Y}}$ that is not GFLQ. From Lemma 3 it follows that there exists an index $j \geq 0$ such that $\delta_{\mathsf{X}[1..][j]} = \delta_{\mathsf{X}[j+1]} = \delta_{\mathsf{l}}$ and $\delta_{\mathsf{Y}[j]} = \delta_{\mathsf{h}}$. Let us then construct the automaton in Figure 4.5. The initial automaton does not contain the word bba^ω in its high-language; however, after merging the two pairs of states (q_1, q_2) and (p_0, r_0), the resulting TBA does accept the word bba^ω.

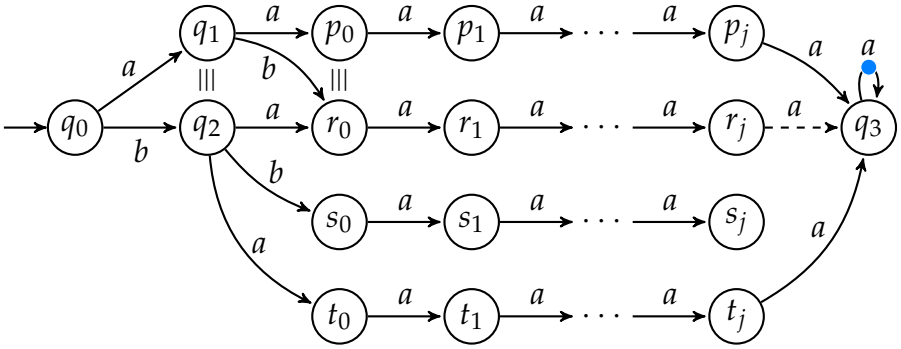
„ $\Longleftarrow$ ”

Let $\mathcal{A} = (Q, \Sigma, \delta_{\mathsf{l}}, \delta_{\mathsf{h}}, q_0, F)$ be an MTBA and

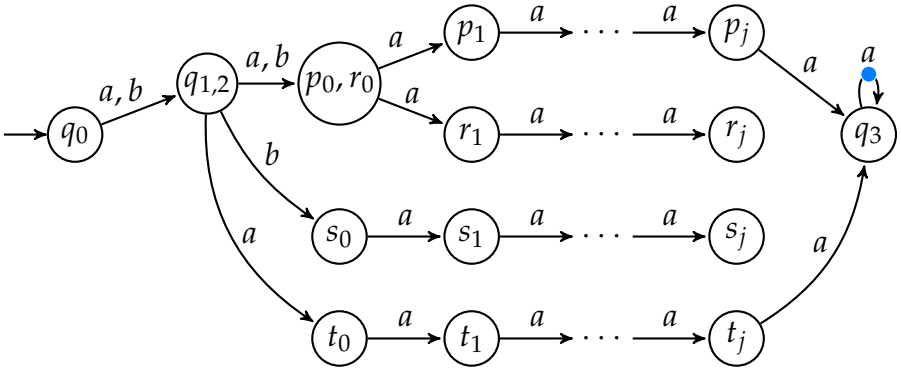
$$\mathcal{B} = \mathcal{A}_{\mathsf{l}}/{\equiv} = ([Q], \Sigma, \delta', [q_0], F').$$

Let us show that $\mathcal{L}(\mathcal{B}) \subseteq \mathcal{L}(\mathcal{A}_{\mathsf{h}})$ holds. Let $w = \sigma_0\sigma_1\sigma_2 \dots \in \mathcal{L}(\mathcal{B})$ be arbitrary and let

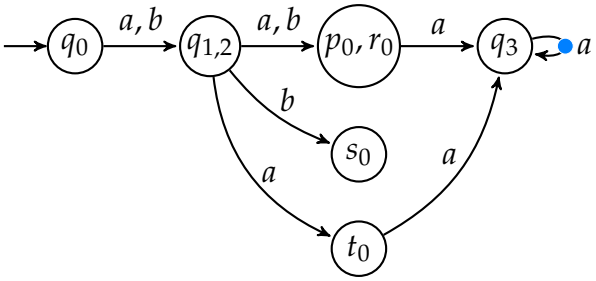
$$\rho = ([q_0], \sigma_0, [q_1])([q_1], \sigma_1, [q_2])([q_2], \sigma_2, [q_3]) \dots \in \delta'^\omega$$



Initial automaton



Low-quotient (for $j > 0$)



Low-quotient (for $j = 0$)

Figure 4.5: If $\mathsf{X}[j+1] = \mathsf{l}$ and $\mathsf{Y}[j] = \mathsf{h}$ for some j and some $\mathsf{X}, \mathsf{Y} \in \{\mathsf{l}, \mathsf{h}\}^\omega$, then there exists an equivalence $\equiv \subseteq \sqsubseteq_{\mathsf{Y}}^{\mathsf{X}}$ such that $\equiv$ is not GFLQ.

be an initial fair trace of $\mathcal{B}$ over w.

Lemma 14 implies that for every $i \geq 0$, there exists a finite initial trace $\varphi_i \in \delta_{\mathsf{h}}^i$ of $\mathcal{A}$ over $w[..i]$ such that $\varphi_i \geq_{acc} \rho[..i]$.

Further by using Lemma 13 we obtain $w \in \mathcal{L}(\mathcal{A}_{\mathsf{h}})$. □

Next, we will show that the same condition holds for backward X/Y-simulations.

Theorem 16. *Let* $\mathsf{X}, \mathsf{Y} \in \{\mathsf{l}, \mathsf{h}\}^\omega$ *such that* $\delta_{\mathsf{X}} \subseteq \delta_{\mathsf{Y}}$. *Then for every equivalence* $\approx\!\!\!\approx \subseteq \preceq_{\mathsf{Y}}^{\mathsf{X}}$, $\equiv$ *is GFLQ if and only if* $\delta_{\mathsf{X}[1..]} \supseteq \delta_{\mathsf{Y}}$.

Proof. „ $\Longrightarrow$ ”

We will prove the contraposition of the implication: if $\delta_{\mathsf{X}[1..]} \not\supseteq \delta_{\mathsf{Y}}$, then there exists an equivalence $\approx\!\!\!\approx \subseteq \preceq_{\mathsf{Y}}^{\mathsf{X}}$ that is not GFLQ. From Lemma 3 it follows that there exists an index $j \geq 0$ such that $\delta_{\mathsf{X}[1..][j]} = \delta_{\mathsf{X}[j+1]} = \delta_{\mathsf{l}}$ and $\delta_{\mathsf{Y}[j]} = \delta_{\mathsf{h}}$. Let us then construct the automaton in Figure 4.6. The initial automaton does not contain the word $a^{j+1}bba^\omega$ in its high-language; however, after merging the two pairs of states (q_1, q_2) and (p_0, r_0), the resulting TBA does accept the word $a^{j+1}bba^\omega$.

„ $\Longleftarrow$ ”

Let $\mathcal{A} = (Q, \Sigma, \delta_{\mathsf{l}}, \delta_{\mathsf{h}}, q_0, F)$ be an MTBA and

$$\mathcal{B} = \mathcal{A}_{\mathsf{l}}/\approx\!\!\!\approx = ([Q], \Sigma, \delta', [q_0], F').$$

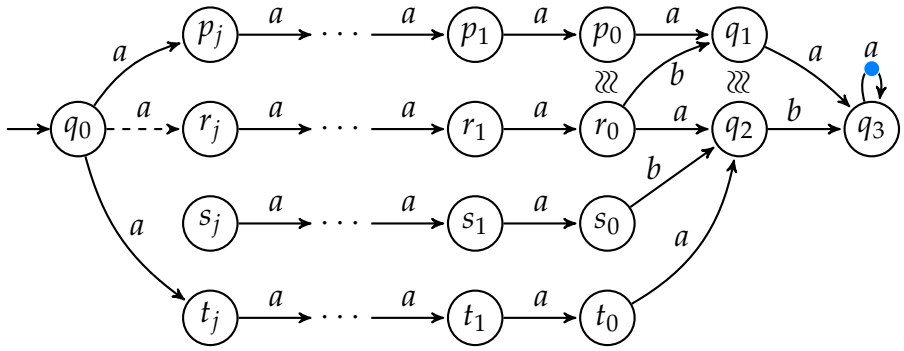
Let us prove $\mathcal{L}(\mathcal{B}) \subseteq \mathcal{L}(\mathcal{A}_{\mathsf{h}})$. Let $w = \sigma_0\sigma_1\sigma_2 \ldots \in \mathcal{L}(\mathcal{B})$ be arbitrary and $\rho = ([q_0], \sigma_0, [q_1])([q_1], \sigma_1, [q_2]) \ldots \in \delta'^\omega$ be an initial fair trace of $\mathcal{B}$ over w. We will show that for every $i \geq 0$, for every $p_i \in [q_i]$, there exists a finite initial trace

$$\tau_i = (t_0, \sigma_0, t_1)(t_1, \sigma_1, t_2) \ldots (t_{i-1}, \sigma_{i-1}, t_i) \in \delta_{\mathsf{Y}[..i]^R}$$

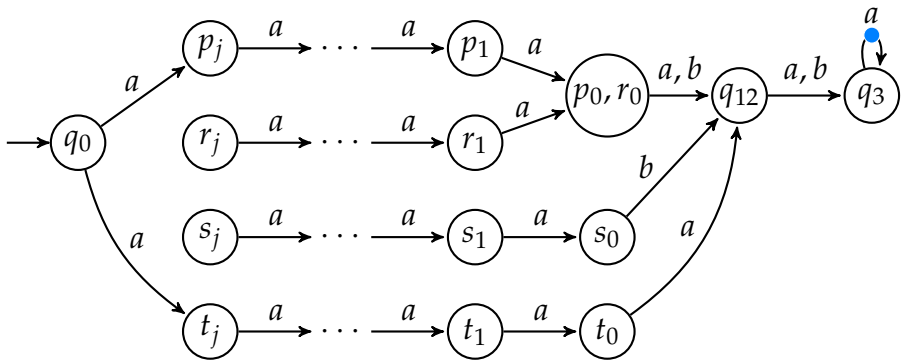
ending in p_i ($t_i = p_i$) of $\mathcal{A}$ over $w[..i]$ such that $\tau_i \geq_{acc} \rho[..i]$. The proof will be done by induction.

For the base of induction, let $i = 0$. Since we assume only one initial state, then there does not exists any state $v \neq q_0$ such that $q_0 \preceq_{\mathsf{Y}}^{\mathsf{X}} v$, hence $[q_0] = \{q_0\}$.

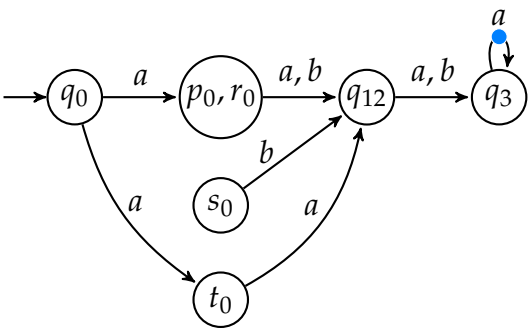
For the induction step, let $i > 0$. From the definition of low-quotienting, there exist suitable states $t_{i-1} \in [q_{i-1}], t_i \in [q_i]$ such that $(t_{i-1}, \sigma_{i-1}, t_i) \in \delta_{\mathsf{l}}$ and $(t_{i-1}, \sigma_{i-1}, t_i) \geq_{acc} ([q_{i-1}], \sigma_{i-1}, [q_i])$. By



Initial automaton



Low-quotient (for $j > 0$)



Low-quotient (for $j = 0$)

Figure 4.6: If $\mathsf{X}[j+1] = \mathsf{l}$ and $\mathsf{Y}[j] = \mathsf{h}$ for some j and some $\mathsf{X}, \mathsf{Y} \in \{\mathsf{l}, \mathsf{h}\}^\omega$, then there exists an equivalence $\approx\!\!\!\approx \subseteq \preceq^{\mathsf{X}}_{\mathsf{Y}}$ such that $\approx\!\!\!\approx$ is not GFLQ.

the induction hypothesis, for the state t_{i-1} there exists a finite initial trace

$$\tau_{i-1} = (r_0, \sigma_0, r_1)(r_1, \sigma_1, r_2) \dots (r_{i-2}, \sigma_{i-2}, r_{i-1}) \in \delta_{\mathsf{Y}[..(i-1)]^R}$$

ending in t_{i-1} that is at least as accepting as $\rho[..(i-1)]$. Let $p_i \in [q_i]$ be arbitrary. Since $p_i \approx t_i$ (and $\approx \subseteq \preceq^{\mathsf{X}}_{\mathsf{Y}}$), then for the finite initial trace

$$\tau_{i-1}.(t_{i-1}, \sigma_{i-1}, t_i) = (r_0, \sigma_0, r_1) \dots (r_{i-2}, \sigma_{i-2}, r_{i-1}).(t_{i-1}, \sigma_{i-1}, t_i) \in$$
$$\in \delta_{\mathsf{Y}[..(i-1)]^R}.\delta_{\mathsf{I}} \subseteq \delta_{\mathsf{X}[1..][..(i-1)]^R.\mathsf{X}[0]} = \delta_{\mathsf{X}[..i]^R}$$

there exists a finite initial trace

$$\tau_i = (p_0, \sigma_0, p_1)(p_1, \sigma_1, p_2) \dots (p_{i-1}, \sigma_{i-1}, p_i) \in \delta_{\mathsf{Y}[..i]^R}$$

that is at least as accepting as $\rho[..i]$, since

$$\tau_i \geq_{acc} \tau_{i-1}.(t_{i-1}, \sigma_{i-1}, t_i) \geq_{acc} \rho[..(i-1)].\rho[i-1].$$

Hence for every $i \geq 0$, there exists a finite initial trace $\tau_i \in \delta^i_{\mathsf{h}}$ of $\mathcal{A}$ over $w[..i]$ that is at least as accepting as $\rho[..i]$ and from Lemma 13 it follows that $w \in \mathcal{L}(\mathcal{A}_{\mathsf{h}})$. □

5 Parity game construction

In this chapter we will show how to reduce the problem of computing particular class of X/Y-simulation to the problem of solving a parity game. The construction of parity game arena is inspired by the construction introduced by Zbončáková [5].

Let $\mathcal{A} = (Q, \Sigma, \delta_{\mathsf{l}}, \delta_{\mathsf{h}}, q_0, F)$ be an MTBA, $\mathsf{W}, \mathsf{Y} \in \{\mathsf{l}, \mathsf{h}\}^n$ and $\mathsf{X}, \mathsf{Z} \in \{\mathsf{l}^\omega, \mathsf{h}^\omega\}$. The idea behind construcion of parity game arena for WX/YZ-simulation is as follows. The vertices V_0 correspond to the configuration at the beginning of any round of the simulation game. The vertices in V_1 correspond to the part of the round when it is Duplicators turn. We will then construct the edges between these vertices in such a way that they correspond to the moves that players can make in the simulation game. We will also define the priority function f, so that it copies the winning condition of Duplicator. Moreover, we will organize the arena into "layers". Each layer will correspond to a different round of the simulation game. This is because we need to allow the players to use different sets of transitions in different rounds. Hence, for the WX/YZ-simulation, we will organize the parity game arena into $n + 1$ layers. Each $1 \leq i \leq n$, the i-th layer will corespond to the i-th round of the game, where Spoiler and Duplicator will be allowed to use $\delta_{\mathsf{W}[i-1]}$ and $\delta_{\mathsf{Y}[i-1]}$, respectively. Any following round of the simulation game will be translated to the $(n+1)$-th layer of the parity game, since in any of these rounds, Spoiler an Duplicator will be able to use only one set of transition for the rest of the game, either δ_{l} or δ_{h}.

For the forward WX/YZ-simulation, on the automaton $\mathcal{A}$, we define the following parity game arena $\mathcal{G}^{\mathsf{WX}}_{\mathsf{YZ}} = (V_0, V_1, E, f)$, where:

$$
\begin{aligned}
V_0 =& \{ v^i_{(p,q)} \mid p,q \in Q,\ 0 \le i \le n \} \\
V_1 =& \{ v^i_{(p',q,\sigma,A)} \mid p',q \in Q,\ \sigma \in \Sigma,\ \exists p \in Q \,.\, \big((p,\sigma,p') \in \delta_{\mathsf{WX}[i]},\ A \Leftrightarrow (p,\sigma,p') \in F\big),\ 0 \le i \le n \} \\
E =& \{ (v^i_{(p,q)}, v^i_{(p',q,\sigma,A)}) \mid (p,\sigma,p') \in \delta_{\mathsf{WX}[i]},\ A \Leftrightarrow (p,\sigma,p') \in F,\ 0 \le i \le n \} \cup \\
& \cup \{ (v^i_{(p',q,\sigma,A)}, v^{min(i+1,n)}_{(p',q')}) \mid (q,\sigma,q') \in \delta_{\mathsf{YZ}[i]},\ A \Rightarrow (q,\sigma,q') \in F,\ 0 \le i \le n \}
\end{aligned}
$$

$$
f(v) = \begin{cases} 0 & \text{if } v \in V_0, \\ 1 & \text{if } v \in V_1 \end{cases}
$$

For the backward WX/YZ-simulation, on the automaton $\mathcal{A}$, the construction is very similar to the forward simulation; however, transitions must be taken backwards and we also introduce the restriction for matching initial states. The parity game arena then is ${}^{bw}\mathcal{G}^{\mathsf{WX}}_{\mathsf{YZ}} = (V^{bw}_0, V^{bw}_1, E^{bw}, f^{bw})$, where:

$$
\begin{aligned}
V^{bw}_0 =& \{ v^i_{(p,q)} \mid p,q \in Q,\ 0 \le i \le n \} \\
V^{bw}_1 =& \{ v^i_{(p',q,\sigma,A)} \mid p',q \in Q,\ \sigma \in \Sigma,\ \exists p \in Q \,.\, \big((p',\sigma,p) \in \delta_{\mathsf{WX}[i]},\ A \Leftrightarrow (p',\sigma,p) \in F\big),\ 0 \le i \le n \} \\
E^{bw} =& \{ (v^i_{(p,q)}, v^i_{(p',q,\sigma,A)}) \mid (p',\sigma,p) \in \delta_{\mathsf{WX}[i]},\ A \Leftrightarrow (p',\sigma,p) \in F,\ 0 \le i \le n \} \cup \\
& \cup \{ (v^i_{(p',q,\sigma,A)}, v^{min(i+1,n)}_{(p',q')}) \mid (q',\sigma,q) \in \delta_{\mathsf{YZ}[i]},\ A \Rightarrow (q',\sigma,q) \in F,\ p' = q_0 \Rightarrow q' = q_0, 0 \le i \le n \}
\end{aligned}
$$

$$
f^{bw}(v) = \begin{cases} 0 & \text{if } v \in V_0, \\ 1 & \text{if } v \in V_1 \end{cases}
$$

6 Implementation

In this chapter we focus on the implementation of the reduction techniques introduced in this thesis. We also deal with the problems of storing modal automata in the HOA format [10].

Tool `Maslo`

The reduction techniques introduced in this thesis have been implemented in the tool `Maslo`[1], which is part of this thesis and is publicly available at `https://gitlab.fi.muni.cz/xsmolka/maslo` under the WTF Public Licence Version 2. Implementation of this tool is again inspired by the tool `Reduce` by Zbončáková. `Maslo` reades MTBA in HOA format from standard input and returns reduced automata in HOA format on the standard output. The implementation is done in Python and utilizes the Spot [4] library, version 2.11.4.

`Maslo` is able to reduce automata in four ways. Given an input MTBA $\mathcal{A}$, the tool can return one of the following:

- $Prune_M(\mathcal{A}, P(id, \sqsubset))$, where $\sqsubset \subseteq \sqsubseteq^{\mathsf{h}^n.\mathsf{l}^\omega}_{\mathsf{h}^n.\mathsf{h}.\mathsf{l}^\omega}$ is a strict partial order.
- $Prune_M(\mathcal{A}, P(\prec, id))$, where $\prec \subseteq \preceq^{\mathsf{h}^n.\mathsf{l}^\omega}_{\mathsf{h}^n.\mathsf{h}.\mathsf{l}^\omega}$ is a strict partial order.
- $\mathcal{A}_\mathsf{l}/\equiv$, where $\equiv \subseteq \sqsubseteq^{\mathsf{l}^n.\mathsf{l}.\mathsf{h}^\omega}_{\mathsf{l}^n.\mathsf{h}^\omega}$ is an equivalence.
- $\mathcal{A}_\mathsf{l}/\approx$, where $\approx \subseteq \preceq^{\mathsf{l}^n.\mathsf{l}.\mathsf{h}^\omega}_{\mathsf{l}^n.\mathsf{h}^\omega}$ is an equivalence.

Procedure, how particular strict partial orders and equivalences are obtained, is described in the control flow diagram in Figure 6.1.

To run the tool, simply write:

```
python3 maslo.py [-h] [-n N] {prune,quotient} {fw,bw}
```

The positional and optional arguments taken by `Maslo` are described in Table 6.1.

1. **M**odal **A**utomata reduced by the use of **S**imu**L**ati**O**ns

Table 6.1: Arguments for `Maslo`.

Arguments	Description
`{prune,quotient}`	which reduction technique to use
`{fw,bw}`	direction of simulation
`-n`	specifies n in the simulation; default is 0
`-h`	prints help message

Figure 6.1 shows the control flow diagram for `Maslo`. First, MTBA from the input is loaded and after that, based on the arguments, corresponding simulation is computed. In case of pruning, either the relation $P(id, \sqsubseteq)$ or the relation $P(\preceq, id)$ is computed, after which antisymmetric and transitive subrelation is taken. Next, the automaton is pruned and state that became unreachable are removed and the final automaton is printed. In case of quotienting, equivalence is computed and quotient is created and printed.

Computation of simulation is done by reducing them to parity games and solving those with Spot. Symmetric and antisymmetric subrelations are computed by trivial algorithms: Algorithm 1 and Algorithm 2, respectively. After pruning, unreachable states are removed by using Spot's function `purge_dead_states`, which removes states that cannot be part of an inifinite run.

Implementation details

Modal automata are a little more complicated than standard automata, so we need to solve a few problems, which are trivial in classical automata and their simulations.

Storing MTBA in HOAF

The HOA format in its current version does not support modal automata. To go around this issue, for an MTBA $\mathcal{A}$, we use Spot's named property `highlight-edges`, to mark modal transitions (i.e. transitions from the set $\delta_{\mathsf{h}} \setminus \delta_{\mathsf{l}}$). We then store the high-automaton $\mathcal{A}_{\mathsf{h}}$ in HOA format, version 1.1, which supports storage of Spot's named properties.

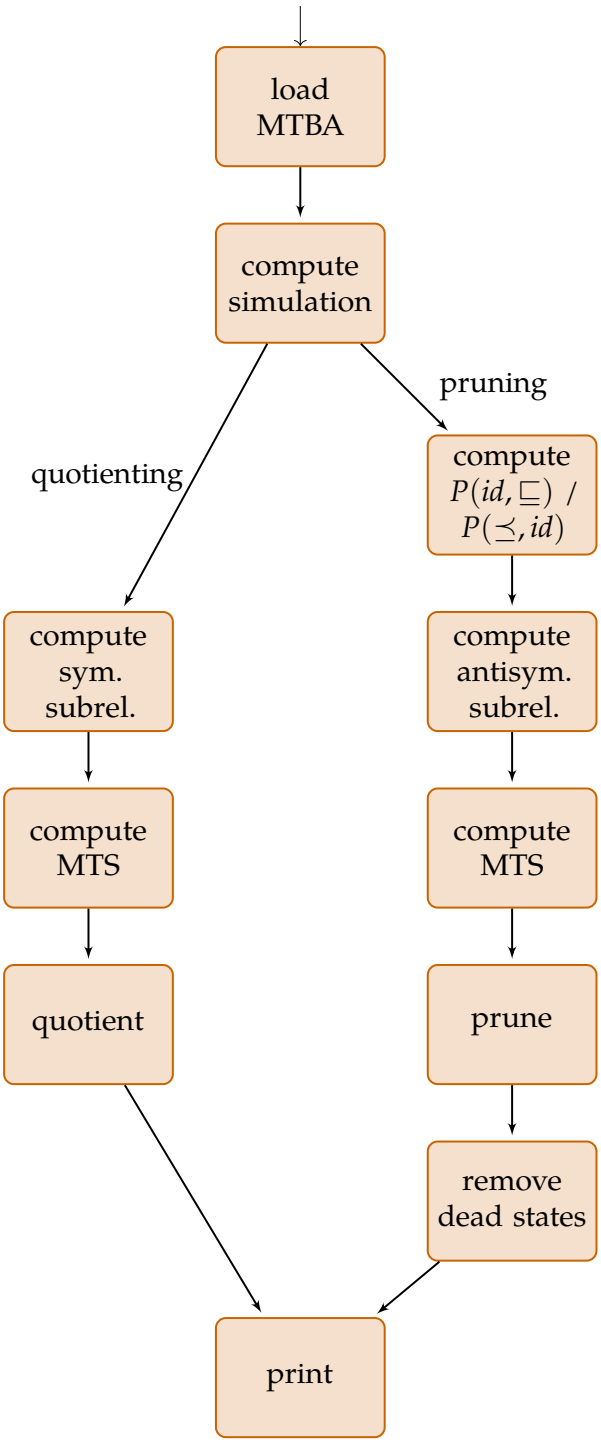


Figure 6.1: `Maslo` control flow diagram.

Algorithm 1: Finding maximal symmetric subrelation

Input : An $n \times n$ matrix $A = (a_{ij})$ representing a relation.
Output: A matrix $B = (b_{ij})$, which is the maximal symmetric subrelation contained in A.

$B \leftarrow 0_{n,n}$
for $i \leftarrow 1$ **to** n **do**
  for $j \leftarrow 1$ **to** n **do**
    if $a_{ij} = 1$ *and* $a_{ji} = 1$ **then**
      $b_{ij} \leftarrow 1$
      $b_{ji} \leftarrow 1$
    end
  end
end
return B

Algorithm 2: Finding maximal antisymmetric subrelation

Input : An $n \times n$ matrix $A = (a_{ij})$ representing a relation.
Output: A matrix $B = (b_{ij})$, which is a maximal antisymmetric subrelation contained in A.

$B \leftarrow 0_{n,n}$
for $i \leftarrow 1$ **to** n **do**
  for $j \leftarrow 1$ **to** $n,\ j \neq i$ **do**
    if $a_{ij} = 1$ *and* $b_{ji} = 0$ **then**
      $b_{ij} \leftarrow 1$
    end
  end
end
return B

Obtaining modal automata

One potential source of modal automata is a development version of Spot. From private communication with Alexandre Duret-Lutz (mantainer of Spot), we know that modal automata are implemented in a development version of Spot; however, their export is not yet supported and creating relevant modal automata is a non-trivial task.

Modal automata are also internally produced in an unpublished development version of the tool *Seminator* [11]. However, Seminator produces a very specific class of modal automata, i.e., automata, where the low-language equals the high-language.

Obtainting transitive subrelation

Modal simulations, by their nature, create asymmetry between Spoiler and Duplicator, which causes them to not be transitive, as can be seen on the $\mathsf{l}^\omega/\mathsf{h}^\omega$-simulation in Figure 6.2. However, for pruning and quotienting, it is necessary to obtain a transitive relation. We do this by implementing an algorithm for obtaining maximal transitive subrelation, described by Chakraborty, et al. [12].

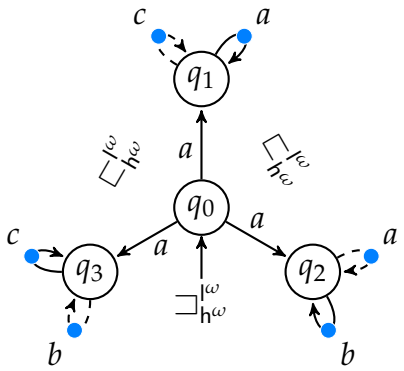


Figure 6.2: $\sqsubseteq^{\mathsf{l}^\omega}_{\mathsf{h}^\omega}$ is not transitive.

Computing simulations with parity games

As mentioned earlier, we reduce the problem od computing simulations to the problem of computing parity games. However, Spot's

parity games solver uses a different winning condition. In Spot, every finite play is won by Player 0. Hence, if we want to use our parity game construction, we need to modify the input automaton in such a way that every simulation game will be infinite. For forward simulations, we do this by introducing a new state and adding a transition from every state (even the new state itself) under every symbol to the new state. For backward simulation, we again introduce a new state an dadd a transition to every state under every symbol from the new state.

7 Experiments

In this chapter, we describe what experiments were ran and how they were evaluated. Given an MTBA $\mathcal{A}$, we will run our reduction algorithms and compare them with reduction algorithms implemented in Spot on the high-automaton $\mathcal{A}_h$ and low-automaton $\mathcal{A}_l$.

Data

For generating data, we used Spot's tool `genltl` and the unpublished version of Seminator. First we took a set of 1769 LTL formulae generated by `genltl` and used those as input for `ltl2tgba` which gave us a set of automata. These automata we then fed to Seminator, which was modified so that it printed desired modal automata in earlier mentioned HOA format with modal transitions highlighted. Out of these 1769 automata, 1264 of them didn't contain any modal transitions. We are then left with 505 automata, which we used for our experiments.

Results

All our experiments were done by comparing Spot's reductions with our reductions, which were combined with Spot's reducitons (Spot vs. `Maslo` + Spot). We did two rounds of experiments. In the first round, we compared our reductions with Spot's simulation-based reductions (`reduce_direct_sim` and `reduce_direct_cosim`). In the second round, we compared our reductions with Spot's strongest reduction algorithms.

Spot's simulation-based reductions

Given an MTBA $\mathcal{A}$, we ran each of our four different reduction techniques on $\mathcal{A}$, after which we applied Spot's reduction based on simulation in the same direction, obtaining the final TBA, e.g. after running `maslo prune bw`, we reduced its output with `reduce_direct_cosim`. Moreover, we ran `Maslo` six times, with flag `-n i` for `i` from 0 to 5 and picked the best (with least number of states) among them.

Each of our four reductions was then compared with Spot's reduction technique based on simulation in the same direction. Given an MTBA $\mathcal{A}$, we reduced both $\mathcal{A}_h$ and $\mathcal{A}_l$ with Spot's simulation-based reduction and picked the one with least states.

Scatter plots of these comparisons can be seen in Figure 7.1 and comparisons of number of smaller automata are in Table 7.1. Most interesting about these data is the diversity. While in the pruning with backward simulation `Maslo` beats Spot's reduction more than the other way around, in pruning with forward simulation, Spot works much better. However, in quotienting it is a more even score.

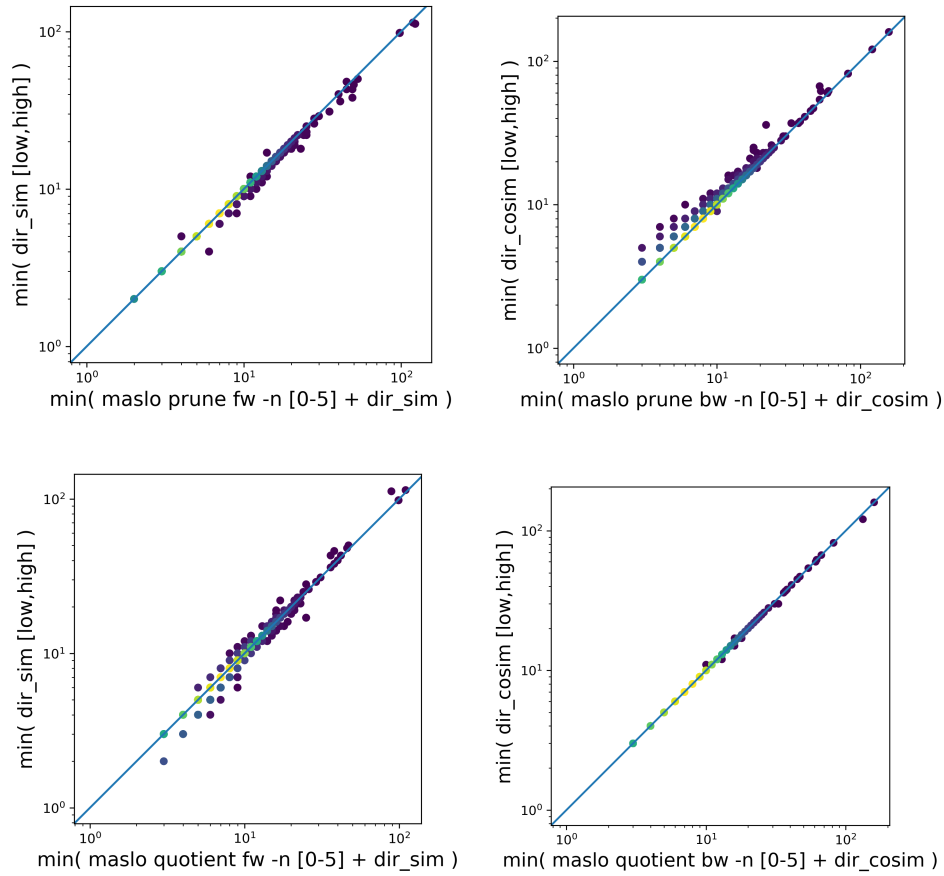


Figure 7.1: Best of `Maslo` with `-n i` vs. Spot's simulation-based reductions (for $0 \leq \mathtt{i} \leq 5$).

Table 7.1: Comparison of better reduced automata.

Reduction technique	Number of states		
	< Spot	= Spot	> Spot
`prune fw -n [0-5] + dir_sim`	4	174	327
`prune bw -n [0-5] + dir_cosim`	181	222	102
`quotient fw -n [0-5] + dir_sim`	36	386	83
`quotient bw -n [0-5] + dir_cosim`	18	407	80

Spot's `autfilt`

Given an MTBA $\mathcal{A}$, we again ran each of our modal reduction techniques. Afterwards, we processed the output with `autfilt --high --small`. As before, we ran `Maslo` with `-n i` for each `i` from 0 to 5 and picked the smallest among them.

Then, we took both $\mathcal{A}_h$ and $\mathcal{A}_l$, reduced them with `autfilt --high --small` and picked the smaller one.

Figure 7.2 contains scatter plots and Table 7.2 numerical comparisons. Here, the situation is quite different. In both pruning and quotienting combined with forward simulations, `autfilt` can reduce either the high-automaton or the low-automaton much more effectively than `Maslo`. However, there are still a few automata, where `Maslo` can make some reductions, that Spot cannot.

After examining these experiments, a few natural questions arise. Why do our reductins combined with `autfilt` not return smaller automata? If we tried six different values of `i` for the flag `-n i`, which one works the best? When using classical reduction technique on the low-automaton and the high-automaton, which one reduces to smaller automaton? We will go through these questions one by one.

Why do our reductions not return smaller automata?

We do not really know yet. Our experiments were run only on modal automata from Seminator, where the high-language equals the low-

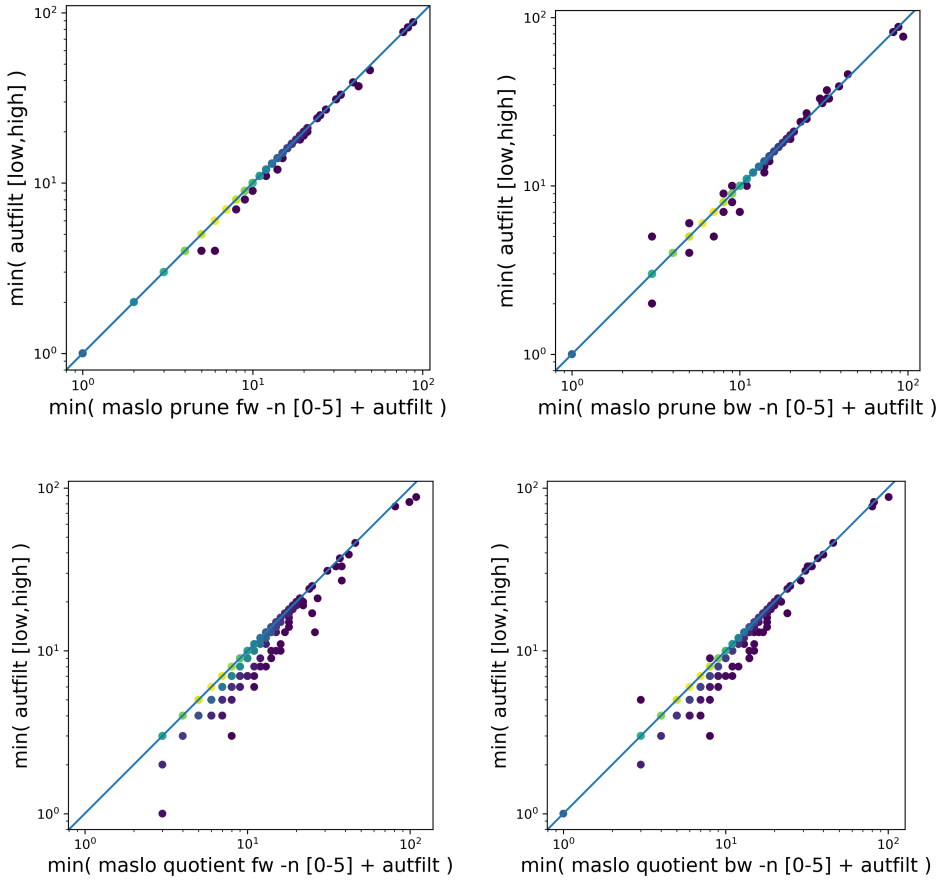


Figure 7.2: Best of `Maslo` with `-n i` vs. `autfilt` (for $0 \leq$ `i` ≤ 5).

Table 7.2: Comparison of better reduced automata.

Reduction technique	Number of states		
	< Spot	= Spot	> Spot
`prune fw -n [0-5] + autfilt`	0	439	66
`prune bw -n [0-5] + autfilt`	28	413	64
`quotient fw -n [0-5] + autfilt`	0	311	194
`quotient bw -n [0-5] + autfilt`	11	319	175

language. Unfortunately, for now we are not able to run experiments on real-life automata, where the two languages are different.

Which value of the flag `-n` works the best?

After examining all six of the values, we noticed that the differences are minimal. In Figure 7.3, Table 7.3, Figure 7.4, Table 7.4 are scatter plots and tables for the same experiments as before; however, this time only for the value 0 of `-n`. Most notable differences, compared to the previous experiments, are in the reduction by modal pruning with backward simulation followed by `autfilt`. Even though the value 0 might not always give the best results, it is the most reasonable option, since the size of parity game arena for particular simulation used in `Maslo` grows linearly with the value of the flag `-n`.

Low-automaton vs. high-automaton for classical reduction

Since in our experiments, classical reductions on low-automata and high-automata turned out to beat our modal reductions, we would like to know which is better. We compared sizes of both of these automata reduced by `autofilt --high --small` and discovered that in 244 cases, the resulting automata have exactly the same size. In 67 cases the low-automaton was reduced to a smaller size and surprisingly in 194 cases, the high-automaton was reduced more effectively. This shows that the inutitive approach of keeping as few transitions as possible, is not always the best. However, we cannot omit the fact that we are working only with modal automata, where the low-language and the high-language are the same. This may have significant influence on the results of our experiments.

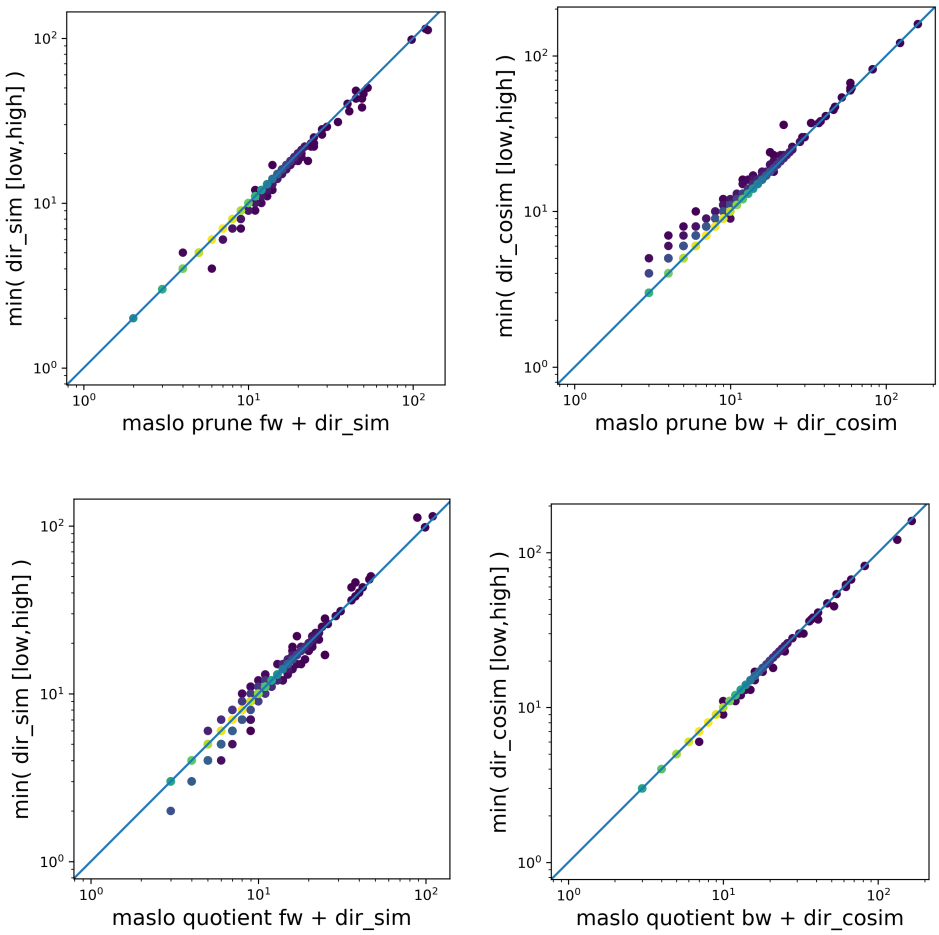


Figure 7.3: `Maslo` with `-n 0` vs. Spot's simulation-based reductions.

Table 7.3: Comparison of better reduced automata.

Reduction technique	Number of states		
	< Spot	= Spot	> Spot
`prune fw + dir_sim`	4	174	327
`prune bw + dir_cosim`	175	222	108
`quotient fw + dir_sim`	36	386	83
`quotient bw + dir_cosim`	18	360	127

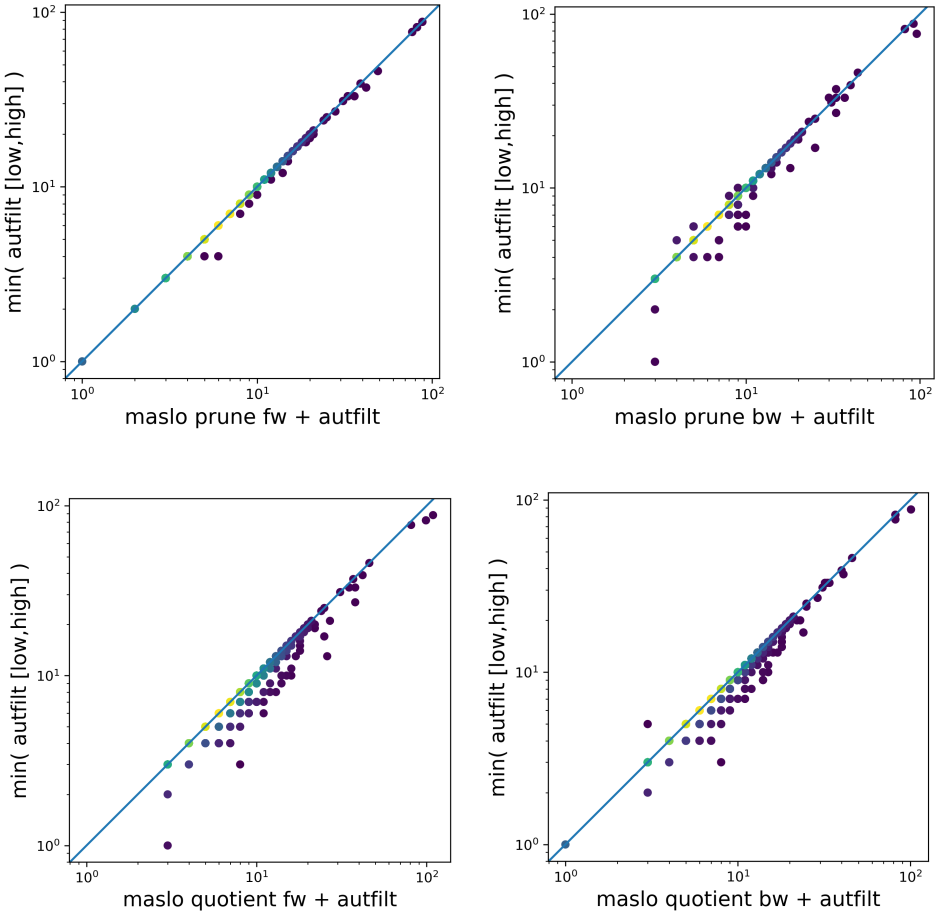


Figure 7.4: `Maslo` with `-n 0` vs. `autfilt`.

Table 7.4: Comparison of better reduced automata.

Reduction technique	Number of states		
	< Spot	= Spot	> Spot
`prune fw + autfilt`	0	437	68
`prune bw + autfilt`	26	391	88
`quotient fw + autfilt`	0	311	194
`quotient bw + autfilt`	10	292	203

Conclusion

In this thesis, we have defined modal transition-based Büchi automata and generalized the concepts of pruning and quotienting to MTBA. Further, we have defined X/Y-simulations, which take modality into account, and for the most interesting of these simulations, we gave equivalent conditions to whether they can be used with pruning or quotienting.

We have shown that for X/Y-simulations to be usable, we need to introduce strong restrictions which reduce the difference between Spoiler and Duplicator.

We have also implemented the modal pruning and quotienting and experimentally compared our reduction algorithms against Spot. Unfortunately, we were able to run our experiments only on a very specific class of modal automata, where the low-language equals the high-language, which greatly limits the room for improvement.

Future work

Since we were not able to run experiments on any real-world automata, where the low-language does not equal the high-language, this would be our first goal. After evaluating such experiments, we can then analyse them and see if our reductions yield better results and if not, we may identify room for improvement.

Next, we want to study so-called "modality preserving pruning," i.e., pruning that produces modal automata.

After that, we would like to study concepts like delayed simulations or trace inclusion and generalize them to modal automata.

Another approach would be to try to explore a different kind of modal simulations, for example, simulations where Spoiler and Duplicator do not have predetermined what transitions they can use in which round, but Duplicator "reacts" to what transitions Spoiler is using.

Bibliography

1. PARK, David Michael Ritchie. Concurrency and Automata on Infinite Sequences. In: *Theoretical Computer Science*. 1981.
2. BUSTAN, Doron; GRUMBERG, Orna. Simulation-Based Minimization. *ACM Trans. Comput. Logic*. 2003, vol. 4, no. 2, pp. 181–206. ISSN 1529-3785. Available from DOI: 10.1145/635499.635502.
3. CLEMENTE, Lorenzo; MAYR, Richard. Efficient reduction of nondeterministic automata with application to language inclusion testing. *Logical Methods in Computer Science*. 2019, vol. Volume 15, Issue 1. Available from DOI: 10.23638/LMCS-15(1:12)2019.
4. ALEXANDRE DURET-LUTZ et al. From Spot 2.0 to Spot 2.10: What's New? In: *Proceedings of the 34th International Conference on Computer Aided Verification (CAV'22)*. Springer, 2022, vol. 13372, pp. 174–187. Lecture Notes in Computer Science. Available from DOI: 10.1007/978-3-031-13188-2_9.
5. ZBONČÁKOVÁ, Tatiana. *Redukcie TGBA pomocou pokročilých simulácií*. Brno, 2021. Available also from: https://is.muni.cz/th/mfeaf/. Master's thesis. Masaryk University, Faculty of Informatics. Supervised by Jan STREJČEK.
6. SOMENZI, Fabio; BLOEM, Roderick. Efficient Büchi Automata from LTL Formulae. In: EMERSON, E. Allen; SISTLA, A. Prasad (eds.). *Computer Aided Verification, 12th International Conference, CAV 2000, Chicago, IL, USA, July 15-19, 2000, Proceedings*. Springer, 2000, vol. 1855, pp. 248–263. Lecture Notes in Computer Science. Available from DOI: 10.1007/10722167_21.
7. ETESSAMI, Kousha; WILKE, Thomas; SCHULLER, Rebecca A. Fair Simulation Relations, Parity Games, and State Space Reduction for Büchi Automata. In: OREJAS, Fernando; SPIRAKIS, Paul G.; LEEUWEN, Jan van (eds.). *Automata, Languages and Programming*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 694–707. ISBN 978-3-540-48224-6.

8. HENZINGER, Thomas A.; KUPFERMAN, Orna; RAJAMANI, Sriram K. Fair simulation. In: MAZURKIEWICZ, Antoni; WINKOWSKI, Józef (eds.). *CONCUR '97: Concurrency Theory*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1997, pp. 273–287. ISBN 978-3-540-69188-4.

9. CLEMENTE, Lorenzo. Büchi Automata Can Have Smaller Quotients. In: ACETO, Luca; HENZINGER, Monika; SGALL, Jiří (eds.). *Automata, Languages and Programming - 38th International Colloquium, ICALP 2011, Zurich, Switzerland, July 4-8, 2011, Proceedings, Part II*. Springer, 2011, vol. 6756, pp. 258–270. Lecture Notes in Computer Science. Available from DOI: 10.1007/978-3-642-22012-8_20.

10. BABIAK, Tomáš; BLAHOUDEK, František; DURET-LUTZ, Alexandre; KLEIN, Joachim; KŘETÍNSKÝ, Jan; MÜLLER, David; PARKER, David; STREJČEK, Jan. The Hanoi Omega-Automata Format. In: KROENING, Daniel; PASAREANU, Corina S. (eds.). *Computer Aided Verification - 27th International Conference, CAV 2015, San Francisco, CA, USA, July 18-24, 2015, Proceedings, Part I*. Springer, 2015, vol. 9206, pp. 479–486. Lecture Notes in Computer Science. Available from DOI: 10.1007/978-3-319-21690-4_31.

11. BLAHOUDEK, František; DURET-LUTZ, Alexandre; STREJČEK, Jan. Seminator 2 Can Complement Generalized Büchi Automata via Improved Semi-Determinization. In: *Proceedings of the 32nd International Conference on Computer-Aided Verification (CAV'20)*. Springer, 2020, vol. 12225, pp. 15–27. Lecture Notes in Computer Science. Available from DOI: 10.1007/978-3-030-53291-8_2.

12. CHAKRABORTY, Sourav; GHOSH, Shamik; JHA, Nitesh; ROY, Sasanka. Maximal and Maximum Transitive Relation Contained in a Given Binary Relation. In: XU, Dachuan; DU, Donglei; DU, Ding-Zhu (eds.). *Computing and Combinatorics - 21st International Conference, COCOON 2015, Beijing, China, August 4-6, 2015, Proceedings*. Springer, 2015, vol. 9198, pp. 587–600. Lecture Notes in Computer Science. Available from DOI: 10.1007/978-3-319-21398-9_46.

A Appendix

A.1 Included archive *attachment.zip*

- *maslo* – directory containing the implementation of the tool `Maslo`.
- *experiments* – directory containing the experiments.
- *README.txt* – manual on how to run the experiments.

A.2 Running `Maslo`

To run the tool `Maslo`, you need to have Spot installed (at least version 2.11.4).

After that, simply run `python3 maslo.py` with desired arguments and feed automata in HOA format to standard input.