Working Paper Series
ISSN 1177-777X

SEVEN ABSTRACTION RULES PRESERVING GENERALISED NONBLOCKING

Robi Malik & Ryan Leduc

Working Paper: 07/2009
September 1, 2009

Department of Computer Science
The University of Waikato
Private Bag 3105
Hamilton, 3240
New Zealand

SEVEN ABSTRACTION RULES PRESERVING GENERALISED NONBLOCKING

Robi Malik
Department of Computer Science
University of Waikato
Hamilton, New Zealand
robi@cs.waikato.ac.nz

Ryan Leduc
Department Computing and Software
McMaster University
Hamilton, ON, Canada
leduc@mcmaster.ca

September 1, 2009

Abstract

This working paper proposes a compositional approach to verify the generalised nonblocking property of discrete-event systems. Generalised nonblocking is introduced in [15] to overcome weaknesses of the standard nonblocking check in discrete-event systems and increase the scope of liveness properties that can be handled. This paper addresses the question of how generalised nonblocking can be verified efficiently. The explicit construction of the complete state space is avoided by first composing and simplifying individual components in ways that preserve generalised nonblocking. The paper extends and generalises previous results about compositional verification of standard nonblocking and lists a new set of computationally feasible abstraction rules for standard and generalised nonblocking.

1 Introduction

Blocking or *conflicts* are common faults in the design of concurrent programs that can be very subtle and hard to detect [6, 21]. They have long been studied in the field of *discrete-event systems* [3, 19], which is applied to the modelling of complex, safety-critical systems [1, 2, 14]. To improve the reliability of such systems, techniques are needed to detect the presence or verify the absence of blocking in models of an ever increasing size.

In discrete-events theory, the absence of blocking is formalised using the *nonblocking* property, which is used very successfully for *synthesis* [3, 19]. A lot of research has been conducted to study the compositional semantics [12, 16] of nonblocking and its verification [9, 18]. Despite its widespread use, the expressive powers of nonblocking are limited. To overcome its weaknesses, nonblocking has been modified and extended in several ways [5, 7, 15].

This working paper is concerned about compositional verification of the *generalised nonblocking* property introduced in [15]. Generalised nonblocking adds to standard nonblocking the ability to restrict the set of states from which blocking needs to be checked. This is useful for the verification of software components and of certain conditions in Hierarchical Interface-Based Supervisory Control [13].

While properties such as generalised nonblocking can be verified using standard state-space exploration or CTL model checking [4], these approaches are limited by the well-known state-space explosion problem. *Compositional verification* using conflict-preserving abstractions is an alternative used with considerable success to verify standard nonblocking [9]. In an effort to extend those results for generalised nonblocking, this working paper presents seven computationally feasible abstraction rules and shows how they can reduce the size of automata during compositional verification of both generalised and standard nonblocking.

This working paper is organised as follows. Section 2 introduces the necessary background of nondeterministic automata and defines the generalised nonblocking property. Then section 3 outlines the compositional verification method, section 4 presents the abstraction rules for generalised nonblocking with proofs of correctness and analysis of complexity, and section 5 adds some concluding remarks.

2 Preliminaries

2.1 Events and Languages

Event sequences and languages are a simple means to describe discrete system behaviours [3, 19]. Their basic building blocks are *events*, which are taken from a finite *alphabet* Σ. In addition, the *silent event* $\tau \notin \Sigma$ is used, with the notation $\Sigma_\tau = \Sigma \cup \{\tau\}$.

Σ^* denotes the set of all finite *strings* of the form $\sigma_1\sigma_2 \ldots \sigma_n$ of events from Σ, including the *empty string* ε. The *concatenation* of two strings $s, t \in \Sigma^*$ is written as st. A subset $\mathcal{L} \subseteq \Sigma^*$ is called a *language*. The *natural projection* $P_\tau \colon \Sigma_\tau^* \to \Sigma^*$ is the operation that deletes all silent (τ) events from strings.

2.2 Multi-coloured Automata

Nondeterministic multi-coloured automata are used to model dynamic system behaviours. *Nondeterminism* is essential for the abstraction techniques in this working paper. *Multi-coloured* automata extend the traditional concept of *marked states* to multiple simultaneous marking conditions, by labelling states with different *colours* or *propositions*. The generalised nonblocking condition is defined using these propositions. The following definition is introduced in [15], and is based on similar ideas in [4, 5].

Definition 1 A *multi-coloured automaton* is a tuple $G = \langle \Sigma, \Pi, X, \rightarrow, X^\circ, \Xi \rangle$ where Σ is a finite set of *events*, Π is a finite set of *propositions* or *colours*, X is a set of *states*, $\rightarrow \subseteq X \times \Sigma_\tau \times X$ is the *state transition relation*, $X^\circ \subseteq X$ is the set of *initial states*, and $\Xi \colon \Pi \rightarrow 2^X$ defines the set of marked states for each proposition in Π.

The transition relation is written in infix notation $x \xrightarrow{\sigma} y$, and is extended to strings in Σ_τ^* in the standard way. For state sets $X_1, X_2 \subseteq X$, the notation $X_1 \xrightarrow{s} X_2$ denotes the existence of $x_1 \in X_1$ and $x_2 \in X_2$ such that $x_1 \xrightarrow{s} x_2$. Also, $x \rightarrow y$ denotes the existence of a string $s \in \Sigma_\tau^*$ such that $x \xrightarrow{s} y$, and $x \xrightarrow{s}$ denotes the existence of a state $y \in X$ such that $x \xrightarrow{s} y$. Finally, $G \xrightarrow{s} x$ stands for $X^\circ \xrightarrow{s} x$.

To support hiding of silent events, another transition relation $\Rightarrow \subseteq X \times \Sigma^* \times X$ is introduced, where $x \stackrel{s}{\Rightarrow} y$ denotes the existence of a string $t \in \Sigma_\tau^*$ such that $P_\tau(t) = s$ and $x \xrightarrow{t} y$. That is, $x \xrightarrow{s} y$ denotes a path with *exactly* the events in s, while $x \stackrel{s}{\Rightarrow} y$ denotes a path with an arbitrary number of τ events shuffled with the events of s. Notations such as $X_1 \stackrel{s}{\Rightarrow} X_2$, $x \Rightarrow y$, and $x \stackrel{s}{\Rightarrow}$ are defined analogously to $\rightarrow$.

Synchronous composition models the parallel execution of two or more automata, and is done using lock-step synchronisation in the style of [10].

Definition 2 Let $G_1 = \langle \Sigma, \Pi, X_1, \rightarrow_1, X_1^\circ, \Xi_1 \rangle$ and $G_2 = \langle \Sigma, \Pi, X_2, \rightarrow_2, X_2^\circ, \Xi_2 \rangle$ be multi-coloured automata. The *synchronous product* of G_1 and G_2 is

$$G_1 \parallel G_2 = \langle \Sigma, \Pi, X_1 \times X_2, \rightarrow, X_1^\circ \times X_2^\circ, \Xi \rangle \tag{1}$$

where

$$\begin{array}{ll} (x_1, x_2) \xrightarrow{\sigma} (y_1, y_2) & \text{if } \sigma \in \Sigma,\ x_1 \xrightarrow{\sigma}_1 y_1,\ x_2 \xrightarrow{\sigma}_2 y_2; \\ (x_1, x_2) \xrightarrow{\tau} (y_1, x_2) & \text{if } x_1 \xrightarrow{\tau}_1 y_1; \\ (x_1, x_2) \xrightarrow{\tau} (x_1, y_2) & \text{if } x_2 \xrightarrow{\tau}_2 y_2; \end{array}$$

and $\Xi(\pi) = \Xi_1(\pi) \times \Xi_2(\pi)$ for each $\pi \in \Pi$.

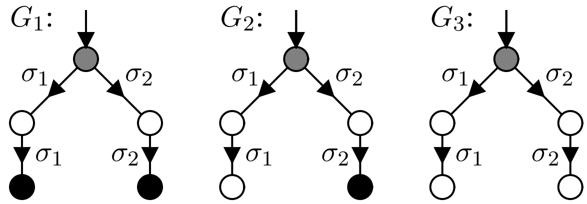

Figure 1: Generalised nonblocking vs. standard nonblocking.

This definition assumes that the two composed automata share the same event and proposition alphabets. This is sufficient for the purpose of this working paper. Automata with different alphabets can also be composed by lifting them to common alphabets first: when an event σ is added to the alphabet Σ, selfloop transitions $x \xrightarrow{\sigma} x$ are added for all states $x \in X$, and when a proposition π is added to Π, it is defined that $\Xi(\pi) = X$.

Hiding is the process-algebraic operation that generalises natural projection of languages when nondeterministic automata are considered. Events that are not of interest are replaced by silent (τ) transitions or ε-*moves* [11].

Definition 3 Let $G = \langle \Sigma, \Pi, X, \rightarrow, X^\circ, \Xi \rangle$ be a multi-coloured automaton, and let $\Upsilon \subseteq \Sigma$. The result of *hiding* Υ in G is

$$G \setminus \Upsilon \;=\; \langle \Sigma \setminus \Upsilon, \Pi, X, \rightarrow \setminus \Upsilon, X^\circ, \Xi \rangle \,, \tag{2}$$

where $\rightarrow \setminus \Upsilon$ is obtained from $\rightarrow$ by replacing all events in Υ with the silent event τ.

2.3 Generalised Nonblocking

It is a desirable for control systems to be free from *livelock* or *deadlock*. This is typically expressed and checked by designating certain states of an automaton as *success* or *terminal* states and checking their reachability. The *generalised nonblocking* property introduced in [15] uses two propositions, called α and ω, for this purpose. The intended meaning is that ω represents terminal states and corresponds to the traditional *marked* states [19], while α specifies a set of states from which terminal states are required to be reachable.

Definition 4 Let $G = \langle \Sigma, \Pi, X, \rightarrow, X^\circ, \Xi \rangle$ with $\alpha, \omega \in \Pi$ be a multi-coloured automaton. G is (α, ω)-*nonblocking*, if for all states $x \in \Xi(\alpha)$ such that $G \Rightarrow x$ it also holds that $x \Rightarrow \Xi(\omega)$. Otherwise, G is (α, ω)-*blocking*.

Generalised nonblocking requires that, from all reachable states marked α, it is possible to reach a state marked ω. This extends the standard definition of nonblocking, which requires that a state marked ω be reachable from *all* states [19], by adding the ability to specify the set of states from which termination must be possible. Clearly, if an automaton is standard nonblocking, it is also (α, ω)-nonblocking. The converse is not true in general.

Example 1 Consider the automata in figure 1. States marked α are grey, and states marked ω are black. Only automaton G_3 is (α, ω)-blocking, although G_2 and G_3 are both blocking according to the standard definition [19].

The relationship between generalised nonblocking and standard nonblocking along with some applications is discussed in [15].

3 Compositional Verification

The straightforward approach to verify whether a composed system

$$G_1 \parallel G_2 \parallel \cdots \parallel G_n \tag{3}$$

is (α, ω)-nonblocking consists of explicitly constructing the synchronous product and checking whether a state marked ω can be reached from every state marked α. This can be done using CTL model checking, and models of substantial size can be analysed if the state space is represented symbolically [4]. Yet, the technique remains limited by the amount of memory available to store representations of the synchronous product.

As an alternative, *compositional* reasoning [9] attempts to rewrite individual components of a composed system such as (3) and, e.g., replace G_1 by a simpler version G_1', to analyse the simpler system

$$G_1' \parallel G_2 \parallel \cdots \parallel G_n \ . \tag{4}$$

Such compositional reasoning requires that G_1 and G_1' are related in some way. An appropriate notion of equivalence has been identified for the verification of standard nonblocking in [16], and adapted to (α, ω)-nonblocking in [15].

Definition 5 Let G_1 and G_2 be two multi-coloured automata with $\alpha, \omega \in \Pi$. Then G_1 and G_2 are called (α, ω)*-nonblocking equivalent*, written $G_1 \simeq_{(\alpha,\omega)} G_2$, if for any multi-coloured automaton T, it holds that $G_1 \parallel T$ is (α, ω)-nonblocking if and only if $G_2 \parallel T$ is (α, ω)-nonblocking.

To be feasible for compositional verification, the equivalence used must be well-behaved with respect to synchronous composition and hiding. These so-called *congruence* properties can easily be shown for (α,ω)-nonblocking equivalence [15].

Proposition 1 Let G_1, G_2, T be multi-coloured automata with $\alpha, \omega \in \Pi$. If $G_1 \simeq_{(\alpha,\omega)} G_2$, then $G_1 \parallel T \simeq_{(\alpha,\omega)} G_2 \parallel T$.

Proof. Let G_1, G_2, and T be such that $G_1 \simeq_{(\alpha,\omega)} G_2$, and let T' be an arbitrary multi-coloured automaton. Since $G_1 \simeq_{(\alpha,\omega)} G_2$, it holds that $(G_1 \parallel T) \parallel T' = G_1 \parallel (T \parallel T')$ is (α,ω)-nonblocking if and only if $G_2 \parallel (T \parallel T') = (G_2 \parallel T) \parallel T'$ is. $\square$

Proposition 2 Let $G = \langle \Sigma, \Pi, X, \rightarrow, X^\circ, \Xi \rangle$ be a multi-coloured automaton with $\alpha, \omega \in \Pi$, and let $\Upsilon \subseteq \Sigma$. Then G is (α,ω)-nonblocking if and only if $G \setminus \Upsilon$ is (α,ω)-nonblocking.

Note that, if given two automata G and H such that H does not use any events in alphabet Υ, then $(G \parallel H) \setminus \Upsilon = (G \setminus \Upsilon) \parallel H$. In combination with proposition 2 this means that abstractions can be applied in a compositional way, as long as only events local to the subsystem considered are abstracted away. Subsystems can be simplified individually or composed as needed, and the verification and simplification strategies outlined in [9, 16] can be used.

4 Abstractions that Preserve Generalised Nonblocking

This section follows the previous work [9] on standard nonblocking and proposes a set of *simplification rules* that can be used to rewrite an automaton to an equivalent but simpler version. The rules are not *complete*, as no attempt is made to ensure that any two (α,ω)-nonblocking equivalent automata can be transformed into each other. Instead, the focus is to provide computationally feasible rewrite rules that can achieve a fair reduction of the state space.

Some of the following results are similar and closely related to corresponding results about abstractions for standard nonblocking [9]. Yet, although (α,ω)-nonblocking seems to be more complicated then standard nonblocking at first glance, it is a weaker property and different kinds of abstraction are possible. Markings can be removed from certain states, and some states that are not coreachable can be removed. Furthermore, unlike in standard nonblocking, a large proportion of the states encountered in generalised nonblocking may be *not* marked α, and these can often be simplified more aggressively than states marked α.

4.1 Observation Equivalence

One of the strongest known equivalences of nondeterministic automata is known as *observation equivalence* or *weak bisimulation* [17]. Observation equivalence considers two states as equivalent if they have exactly the same structure of nondeterministic future behaviour.

Definition 6 Let $G_1 = \langle \Sigma, \Pi, X_1, \rightarrow_1, X_1^\circ, \Xi_1 \rangle$ and $G_2 = \langle \Sigma, \Pi, X_2, \rightarrow_2, X_2^\circ, \Xi_2 \rangle$ be two multi-coloured automata. A relation $\approx \subseteq X_1 \times X_2$ is a *weak bisimulation* between G_1 and G_2 if, for all states $x_1 \in X_1$ and $x_2 \in X_2$ such that $x_1 \approx x_2$,

- if $x_1 \stackrel{s}{\Rightarrow}_1 y_1$ for some $s \in \Sigma^*$, then there exists $y_2 \in X_2$ such that $y_1 \approx y_2$ and $x_2 \stackrel{s}{\Rightarrow}_2 y_2$;
- if $x_2 \stackrel{s}{\Rightarrow}_2 y_2$ for some $s \in \Sigma^*$, then there exists $y_1 \in X_1$ such that $y_1 \approx y_2$ and $x_1 \stackrel{s}{\Rightarrow}_1 y_1$;
- if $x_1 \in \Xi_1(\pi)$ for some $\pi \in \Pi$, then $x_2 \stackrel{\varepsilon}{\Rightarrow}_2 \Xi_2(\pi)$;
- if $x_2 \in \Xi_2(\pi)$ for some $\pi \in \Pi$, then $x_1 \stackrel{\varepsilon}{\Rightarrow}_1 \Xi_1(\pi)$.

G_1 and G_2 are *observation equivalent* or *weakly bisimilar*, $G_1 \approx G_2$, if there exists a weak bisimulation $\approx$ between G_1 and G_2 such that, for each initial state $x_1^\circ \in X_1^\circ$ there exists $x_2^\circ \in X_2$ such that $X_2^\circ \stackrel{\varepsilon}{\Rightarrow}_2 x_2^\circ$ and $x_1^\circ \approx x_2^\circ$, and vice versa.

Observation equivalence has been studied extensively in process algebra. It is known to preserve all temporal properties, and as such it is finer than (α, ω)-nonblocking equivalence. The following result is straightforward to prove.

Proposition 3 Let G_1 and G_2 be two multi-coloured automata with $\alpha, \omega \in \Pi$. If $G_1 \approx G_2$ then $G_1 \simeq_{(\alpha,\omega)} G_2$.

Proof. Let $G_1 \approx G_2$. To show $G_1 \simeq_{(\alpha,\omega)} G_2$, let T be such that $G_1 \parallel T$ is (α, ω)-nonblocking, and let $G_2 \parallel T \stackrel{s}{\Rightarrow} (x_2, x_T) \in \Xi_{G_2 \parallel T}(\alpha)$. Then there exists an initial state $x_2^\circ \in X_2^\circ$ such that $x_2^\circ \stackrel{s}{\Rightarrow}_2 x_2$. Since $G_1 \approx G_2$, by definition 6, there exists $x_1^\circ \in X_1$ such that $X_1^\circ \stackrel{\varepsilon}{\Rightarrow}_1 x_1^\circ$ and $x_1^\circ \approx x_2^\circ$, and since $x_2^\circ \stackrel{s}{\Rightarrow}_2 x_2$, there exists $x_1 \in X_1$ such that $x_1^\circ \stackrel{s}{\Rightarrow}_1 x_1$ and $x_1 \approx x_2$. Again by definition 6, since $x_2 \in \Xi_2(\alpha)$, there exists $x_1' \in \Xi_1(\alpha)$ such that $x_1 \stackrel{\varepsilon}{\Rightarrow}_1 x_1'$. Thus, $G_1 \parallel T \stackrel{s}{\Rightarrow} (x_1, x_T) \stackrel{\varepsilon}{\Rightarrow} (x_1', x_T) \in \Xi_{G_1 \parallel T}(\alpha)$. Since $G_1 \parallel T$ is (α, ω)-nonblocking, there exist $t \in \Sigma^*$, $y_1 \in \Xi_1(\omega)$, and $y_T \in \Xi_T(\omega)$ such that $G_1 \parallel T \stackrel{s}{\Rightarrow} (x_1, x_T) \stackrel{\varepsilon}{\Rightarrow} (x_1', x_T) \stackrel{t}{\Rightarrow} (y_1, y_T)$. Then $x_1 \stackrel{t}{\Rightarrow}_1 y_1$, and since $x_1 \approx x_2$, by definition 6, there exists $y_2 \in X_2$ such

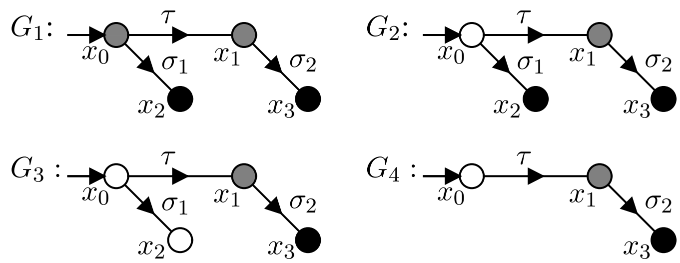

Figure 2: Example application of α-Removal Rule, followed by ω-Removal Rule, and Coreachability Rule

that $x_2 \stackrel{t}{\Rightarrow}_2 y_2$ and $y_1 \approx y_2$, and since $y_1 \in \Xi_1(\omega)$, there exists $y_2' \in \Xi_2(\omega)$ such that $y_2 \stackrel{\varepsilon}{\Rightarrow}_2 y_2'$. Hence, $G_2 \parallel T \stackrel{s}{\Rightarrow} (x_2, x_T) \stackrel{t}{\Rightarrow} (y_2, y_T) \stackrel{\varepsilon}{\Rightarrow} (y_2', y_T) \in \Xi_{G_2 \parallel T}(\omega)$. Since such t can be constructed for any $s \in \Sigma^*$, $G_2 \parallel T$ is (α, ω)-nonblocking. Analogously it can be shown that, if $G_2 \parallel T$ is (α, ω)-nonblocking, then $G_1 \parallel T$ is (α, ω)-nonblocking. It follows that $G_1 \simeq_{(\alpha,\omega)} G_2$. □

Observation equivalence comes with efficient simplification algorithms [8] and has been used successfully to simplify automata for the verification of standard nonblocking, where this abstraction alone is responsible for a substantial reduction in the number of states [9]. Proposition 3 confirms that an automaton can be replaced by an observation equivalent version when verifying generalised nonblocking.

Rule 1 (Observation Equivalence Rule) If two automata G_1 and G_2 are observation equivalent, then G_1 can be replaced by G_2 (and vice versa).

Complexity. A coarsest observation equivalence relation can be computed in $O(|{\rightarrow}| \log |X|)$ using the algorithm in [8]. However, since this algorithm is designed for bisimulation, the automaton has to be augmented such that, for all $\sigma \in \Sigma$, $x \stackrel{\sigma}{\Rightarrow} y$ always implies $x \stackrel{\sigma}{\rightarrow} y$. Thus, the number $|{\rightarrow}|$ of transitions may be very large—on the order of $|X|^2|\Sigma|$, resulting in the overall complexity $O(|X|^2|\Sigma| \log |X|)$.

4.2 Removal of α-Markings

While observation equivalence achieves a good reduction of the state space and is easy to implement, there are several examples of (α, ω)-nonblocking equivalent automata that are not observation equivalent. The following sections propose a selection of simplification rules that are applied directly to the states and transitions of an automaton. The first of these rules simply removes α-markings from certain states.

Rule 2 (α-Removal Rule) If an automaton contains two different states x and y both marked α, such that $x \xrightarrow{\tau} y$, then the α-marking can be removed from state x.

Example 2 Automata G_1 and G_2 in figure 2 are (α, ω)-nonblocking equivalent. States marked α are grey, and states marked ω are black. Since state x_1 is marked α, any test that is to be (α, ω)-nonblocking in combination with G_1 needs to be able to execute σ_2 initially. This implicitly includes the condition for state x_0, which says that a test needs to be able to execute σ_1 or σ_2 initially. As the test must satisfy both, the condition simplifies to just executing σ_2. Testing for state x_1 alone is thus sufficient, so the α-marking of state x_0 can be removed as shown in G_2.

Proposition 4 Let $G = \langle \Sigma, \Pi, X, \rightarrow, X^\circ, \Xi_G \rangle$ be a multi-coloured automaton with $\alpha, \omega \in \Pi$ and states $x, y \in X$ such that $x \xrightarrow{\tau} y$, $x \neq y$, and $x, y \in \Xi_G(\alpha)$. Define $H = \langle \Sigma, \Pi, X, \rightarrow, X^\circ, \Xi_H \rangle$ where Ξ_H is identical to Ξ_G except $\Xi_H(\alpha) = \Xi_G(\alpha) \setminus \{x\}$. Then $G \simeq_{(\alpha,\omega)} H$.

Proof. Let T be an arbitrary multi-coloured automaton. It is sufficient to show that $G \parallel T$ is (α, ω)-nonblocking if and only if $H \parallel T$ is (α, ω)-nonblocking.

First assume that $G \parallel T$ is (α, ω)-nonblocking, and let $H \parallel T \Rightarrow (q, q_T) \in \Xi_{H\parallel T}(\alpha)$. Since the transition relations of G and H are equal, it follows that $G \parallel T \Rightarrow (q, q_T) \in \Xi_{H\parallel T}(\alpha) \subseteq \Xi_{G\parallel T}(\alpha)$. Since $G \parallel T$ is (α, ω)-nonblocking, it follows that $G \parallel T \Rightarrow (q, q_T) \Rightarrow \Xi_{G\parallel T}(\omega)$. Again, since the transition relations of G and H are equal, $H \parallel T \Rightarrow (q, q_T) \Rightarrow \Xi_{G\parallel T}(\omega) = \Xi_{H\parallel T}(\omega)$.

Second assume that $H \parallel T$ is (α, ω)-nonblocking, and let $G \parallel T \Rightarrow (q, q_T) \in \Xi_{G\parallel T}(\alpha)$. Since the transition relations of G and H are equal, it follows that $H \parallel T \Rightarrow (q, q_T)$. Consider two cases.

Case 1: $q \neq x$. In this case $q \in \Xi_G(\alpha) \setminus \{x\} = \Xi_H(\alpha)$. Hence, $H \parallel T \Rightarrow (q, q_T) \in \Xi_{H\parallel T}(\alpha)$, and since $H \parallel T$ is (α, ω)-nonblocking, $H \parallel T \Rightarrow (q, q_T) \Rightarrow \Xi_{H\parallel T}(\omega)$.

Case 2: $q = x$. In this case, $q = x \xrightarrow{\tau} y \in \Xi_G(\alpha)$, and since $x \neq y$ also $y \in \Xi_G(\alpha) \setminus \{x\} = \Xi_H(\alpha)$. Hence, $H \parallel T \Rightarrow (q, q_T) \xrightarrow{\tau} (y, q_T) \in \Xi_{H\parallel T}(\alpha)$. Since $H \parallel T$ is (α, ω)-nonblocking, $H \parallel T \Rightarrow (q, q_T) \xrightarrow{\tau} (y, q_T) \Rightarrow \Xi_{H\parallel T}(\omega)$.

In both cases, $H \parallel T \Rightarrow (q, q_T) \Rightarrow \Xi_{H\parallel T}(\omega)$, and since the transition relations of G and H are equal, it follows that $G \parallel T \Rightarrow (q, q_T) \Rightarrow \Xi_{H\parallel T}(\omega) = \Xi_{G\parallel T}(\omega)$. $\square$

Complexity. To check the applicability of the α-Removal Rule to an automaton, it is enough to visit and check the source and target states of all τ-transitions. There are at most $|X|^2$ τ-transitions, and this leads to the overall complexity of $O(|X|^2)$ to check and apply the α-Removal Rule to all states where it is applicable.

While the removal of markings does not reduce the number of states of an automaton, it can make it simpler and enable other abstractions. Only states marked α need to satisfy nonblocking conditions, so verification is expected to be easier with less states marked α. The α-Removal Rule can also be considered when verifying standard nonblocking, where all states are marked α initially, treating standard nonblocking as generalised nonblocking and making some of the rules for generalised nonblocking applicable.

4.3 Removal of ω-Markings

Similar to the case of α-markings, ω-markings can also be removed under certain conditions, namely if the state marked ω is not reachable from any state marked α.

Rule 3 (ω-Removal Rule) If a state x is not reachable from any state marked α, then an ω-marking can be removed from (or added to) state x.

Example 3 Automata G_2 and G_3 in figure 2 are (α, ω)-nonblocking equivalent. Only for states marked α, it is required that a state marked ω is reachable, but state x_2 in G_2 cannot be reached from any state marked α. Therefore, the fact that x_2 is marked ω is irrelevant, and this marking can be removed as shown in G_3.

Proposition 5 Let $G = \langle \Sigma, \Pi, X, \rightarrow, X^\circ, \Xi_G \rangle$ be a multi-coloured automaton with $\alpha, \omega \in \Pi$, and let $x \in X$ such that $\Xi_G(\alpha) \rightarrow x$ does not hold. Define $H = \langle \Sigma, \Pi, X, \rightarrow, X^\circ, \Xi_H \rangle$ where Ξ_H is identical to Ξ_G except $\Xi_H(\omega) = \Xi_G(\omega) \setminus \{x\}$. Then $G \simeq_{(\alpha,\omega)} H$.

Proof. Let T be an arbitrary multi-coloured automaton. It is sufficient to show that $G \parallel T$ is (α, ω)-nonblocking if and only if $H \parallel T$ is (α, ω)-nonblocking.

First assume that $G \parallel T$ is (α, ω)-nonblocking, and let $H \parallel T \Rightarrow (q, q_T) \in \Xi_{H\parallel T}(\alpha)$. Since the transition relations of G and H are equal, it follows that $G \parallel T \Rightarrow (q, q_T) \in \Xi_{H\parallel T}(\alpha) = \Xi_{G\parallel T}(\alpha)$. Since $G \parallel T$ is (α, ω)-nonblocking, there are states $p \in \Xi_G(\omega)$ and $p_T \in \Xi_T(\omega)$ such that $G \parallel T \Rightarrow (q, q_T) \Rightarrow (p, p_T)$. Again, since the transition relations of G and H are equal, it follows that $H \parallel T \Rightarrow (q, q_T) \Rightarrow (p, p_T)$. Also note $p \neq x$ as $q \in \Xi_G(\alpha)$ and $q \rightarrow p$, and thus $p \in \Xi_G(\omega) \setminus \{x\} = \Xi_H(\omega)$. This implies $H \parallel T \Rightarrow (q, q_T) \Rightarrow (p, p_T) \in \Xi_{H\parallel T}(\omega)$.

Second assume that $H \parallel T$ is (α, ω)-nonblocking, and let $G \parallel T \Rightarrow (q, q_T) \in \Xi_{G\parallel T}(\alpha)$. Since the transition relations of G and H are equal, it follows that $H \parallel$

$T \Rightarrow (q, q_T) \in \Xi_{G\|T}(\alpha) = \Xi_{H\|T}(\alpha)$. Since $H \parallel T$ is (α,ω)-nonblocking, it follows that $H \parallel T \Rightarrow (q, q_T) \Rightarrow \Xi_{H\|T}(\omega)$, and since the transition relations of G and H are equal, also $G \parallel T \Rightarrow (q, q_T) \Rightarrow \Xi_{H\|T}(\omega) \subseteq \Xi_{G\|T}(\omega)$. □

Complexity. To apply the ω-Removal Rule to an automaton, it needs to be checked for all states whether they are reachable from an α-marked state. This can be done by a standard graph search visiting each transition at most once. There are at most $|X|^2|\Sigma_\tau|$ transitions, and this leads to the overall complexity of $O(|X|^2|\Sigma|)$ to check and apply the ω-Removal Rule to all states where it is applicable.

Again, the removal of ω-markings does not directly reduce the state space, but it can make other rules applicable. In particular, it may increase the number of non-coreachable states, which can be deleted according to the following rule.

4.4 Removal of Non-coreachable States

Following is the first abstraction that actually removes states from an automaton. While the following rule seems superficially similar to the *Certain Conflicts Rule* of [9], it is quite different. The Certain Conflicts Rule merges blocking states into a single state when verifying standard nonblocking. Here, in the case of generalised nonblocking, states that are not coreachable can be removed entirely.

Rule 4 (Coreachability Rule) States that are not α/ω-coreachable, i.e., from which neither a state marked α nor a state marked ω can be reached, can be removed.

Example 4 Automata G_3 and G_4 in figure 2 are (α,ω)-nonblocking equivalent. State x_2 in G_3 is neither α-coreachable nor ω-coreachable, and therefore it is not needed to reach an ω-marked state, nor does it lead to any further conditions (α-marked state) that need to be satisfied. This state can be removed as shown in G_4.

Proposition 6 Let $G = \langle \Sigma, \Pi, X, \rightarrow, X^\circ, \Xi \rangle$ be a multi-coloured automaton with $\alpha, \omega \in \Pi$, and let C be the set of α/ω-coreachable states for G, namely $C = \{\, x \in X \mid x \rightarrow \Xi(\alpha) \cup \Xi(\omega) \,\}$. Define $H = \langle \Sigma, \Pi, C, \rightarrow_{|C}, X^\circ \cap C, \Xi_{|C} \rangle$ where $\rightarrow_{|C} = \{\, (x, \sigma, y) \in \rightarrow \mid x, y \in C \,\}$ and $\Xi_{|C}(\pi) = \Xi(\pi) \cap C$ for all $\pi \in \Pi$. Then $G \simeq_{(\alpha,\omega)} H$.

Proof. Let T be an arbitrary multi-coloured automaton. It is sufficient to show that $G \parallel T$ is (α,ω)-nonblocking if and only if $H \parallel T$ is (α,ω)-nonblocking.

First assume that $G \parallel T$ is (α,ω)-nonblocking, and let $H \parallel T \Rightarrow (q, q_T) \in \Xi_{H\|T}(\alpha)$. Obviously, since $\rightarrow_{|C} \subseteq \rightarrow$ and $X^\circ \cap C \subseteq X^\circ$, this implies $G \parallel T \Rightarrow (q, q_T) \in \Xi_{H\|T}(\alpha) \subseteq \Xi_{G\|T}(\alpha)$. Since $G \parallel T$ is (α,ω)-nonblocking, it holds that $G \parallel T \Rightarrow (q, q_T) \stackrel{t}{\Rightarrow} \Xi_{G\parallel T}(\omega)$ for some $t \in \Sigma^*$. Then there exist events $\sigma_1, \dots, \sigma_n \in \Sigma_\tau$, $n \geq 0$, such that $t = P_\tau(\sigma_1 \dots \sigma_n)$ and states $p_0, \dots, p_n \in X$ such that

$$G \Rightarrow q = p_0 \xrightarrow{\sigma_1} p_1 \xrightarrow{\sigma_2} \cdots \xrightarrow{\sigma_n} p_n \in \Xi(\omega) \ . \tag{5}$$

Then $p_0, p_1, \dots, p_n \in C$ by construction of C, and hence $H \parallel T \Rightarrow (q, q_T) \Rightarrow (\Xi(\omega) \cap C) \times \Xi_T(\omega) = \Xi_{H\parallel T}(\omega)$.

Second assume that $H \parallel T$ is (α, ω)-nonblocking, and let $G \parallel T \stackrel{s}{\Rightarrow} (q, q_T) \in \Xi_{G\parallel T}(\alpha)$. Then there exist events $\sigma_1, \dots, \sigma_n \in \Sigma_\tau$, $n \geq 0$, such that $s = P_\tau(\sigma_1 \dots \sigma_n)$ and states $q_0, \dots, q_n \in X$ such that $q_0 \in X^\circ$ and

$$q_0 \xrightarrow{\sigma_1} q_1 \xrightarrow{\sigma_2} \cdots \xrightarrow{\sigma_n} q_n = q \in \Xi(\alpha) \ . \tag{6}$$

Then $q_0, q_1, \dots, q_n \in C$ by construction of C, and hence $H \parallel T \stackrel{s}{\Rightarrow} (q, q_T) \in (\Xi(\alpha) \cap C) \times \Xi_T(\alpha) = \Xi_{H\parallel T}(\alpha)$. Since $H \parallel T$ is (α, ω)-nonblocking, it follows that $H \parallel T \stackrel{s}{\Rightarrow} (q, q_T) \Rightarrow \Xi_{H\parallel T}(\omega)$, and since $\rightarrow_{|C} \subseteq \rightarrow$ also $G \parallel T \stackrel{s}{\Rightarrow} (q, q_T) \Rightarrow \Xi_{H\parallel T}(\omega) \subseteq \Xi_{G\parallel T}(\omega)$. □

Complexity. α/ω-coreachability of all states in an automaton can be checked by a standard graph search visiting each transition at most once. There are at most $|X|^2|\Sigma_\tau|$ transitions, and this leads to the overall complexity of $O(|X|^2|\Sigma|)$ to check and apply the Coreachability Rule.

4.5 Determinisation of Non-α States

In generalised nonblocking, there are two different kinds of states. States marked α carry nonblocking requirements, which means that their precise nondeterministic future may be relevant. These states can only be simplified using rules preserving conflict equivalence such as those in [9]. On the other hand, non-α states do not carry nonblocking requirements, and only the language associated with these states is important. These states can be treated using language equivalence, and determinisation algorithms [11] can be used to merge them.

Rule 5 (Non-α Determinisation Rule) Two non-α states that are reachable by exactly the same strings from initial states and from each state marked α, can be merged into a single state.

Example 5 Automata G and H in figure 3 are (α, ω)-nonblocking equivalent. States g_1 and g_3 are only reachable via string σ_1 from the initial state or from the only α-marked state and therefore can be merged into a single state h_{13} as shown in H. Note that this simplification is not possible for standard nonblocking, or if

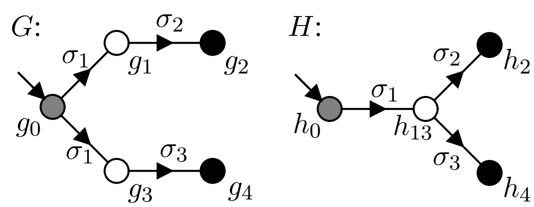

Figure 3: Example application of Non-α Determinisation Rule.

one of the two states is marked α, because in this case it is important that the two states have different continuations to states marked ω.

To describe this rule formally, the concept of automaton abstraction with respect to an *equivalence relation* is used. The idea is to identify certain groups of states as equivalent and merge each group into a single state. The following definitions are standard.

Definition 7 Let X be an arbitrary set. A binary relation $\sim \subseteq X \times X$ is an *equivalence relation*, if $\sim$ is reflexive, symmetric, and transitive. If $\sim$ is an equivalence relation on X, the *equivalence class* of $x \in X$ is $[x] = \{\, y \in X \mid x \sim y \,\}$, and the set of equivalence classes modulo $\sim$ is written as $X/\!\sim \; = \{\, [x] \mid x \in X \,\}$.

Definition 8 Let $G = \langle \Sigma, \Pi, X, \rightarrow, X^\circ, \Xi \rangle$ be a multi-coloured automaton, and let $\sim \subseteq X \times X$ be an equivalence relation. The *abstraction* of G with respect to $\sim$ is

$$G/\!\sim \;=\; \langle \Sigma, \Pi, X/\!\sim, \rightarrow/\!\sim, \tilde{X}^\circ, \tilde{\Xi} \rangle \; , \tag{7}$$

where

$$\begin{aligned} \rightarrow/\!\sim \; &= \; \{\, ([x], \sigma, [y]) \mid x \xrightarrow{\sigma} y \,\} \; ; \\ \tilde{X}^\circ \; &= \; \{\, [x^\circ] \mid x^\circ \in X^\circ \,\} \; ; \\ \tilde{\Xi}(\pi) \; &= \; \{\, [x] \mid x \in \Xi(\pi) \,\} \quad \text{for all } \pi \in \Pi \; . \end{aligned}$$

The Non-α Determinisation Rule is described using a particular equivalence relation, namely a *reverse weak bisimulation* [20]: two states are considered as equivalent if they can be reached via the same traces from the initial states.

Definition 9 Let $G = \langle \Sigma, \Pi, X, \rightarrow, X^\circ, \Xi \rangle$. An equivalence relation $\sim \subseteq X \times X$ is a *reverse weak bisimulation* on G, if the following conditions hold for all $x_1, x_2 \in X$ with $x_1 \sim x_2$.

- If $x_1 \in X^\circ$, then $X^\circ \overset{\varepsilon}{\Rightarrow} x_2$.

- For all states $w_1 \in X$ and all events $\sigma \in \Sigma_\tau$ such that $w_1 \xrightarrow{\sigma} x_1$ there exists a state $w_2 \in X$ such that $w_2 \overset{P_\tau(\sigma)}{\Longrightarrow} x_2$ and $w_1 \sim w_2$.

Given these definitions, the Non-α Determinisation Rule can be described in a more precise way.

Rule 5 (Non-α Determinisation Rule) If $\sim$ is a reverse weak bisimulation on an automaton G such that states marked α are only equated to themselves by $\sim$, then G can be replaced by $G/{\sim}$.

To prove the validity of this rule, the relationship between the traces in an automaton G and its abstraction $G/{\sim}$ needs to be established first. It is well-known that every trace in G also has a corresponding trace in $G/{\sim}$. The following result quoted from [9] holds for every equivalence relation.

Lemma 1 Let $G = \langle \Sigma, \Pi, X, \rightarrow, X^\circ, \Xi \rangle$, and let ${\sim} \subseteq X \times X$ be an equivalence relation. Then, for all states $w, x \in X$ and all strings $s \in \Sigma^*$ such that $w \overset{s}{\Rightarrow} x$ in G, it holds that $[w] \overset{s}{\Rightarrow} [x]$ in $G/{\sim}$.

Proof. Let $w \overset{s}{\Rightarrow} x$ in G. Then there exists $t = \sigma_1 \dots \sigma_n \in \Sigma_\tau^*$ such that $P_\tau(t) = s$ and $w \xrightarrow{t} x$. Also, there exist states $x_0, \dots, x_n \in X$ such that

$$w = x_0 \xrightarrow{\sigma_1} x_1 \xrightarrow{\sigma_2} \cdots \xrightarrow{\sigma_n} x_n = x \, . \tag{8}$$

By definition 8, it holds that $[x_{k-1}] \xrightarrow{\sigma_k} [x_k]$ for each $k = 1, \dots, n$, which implies $[w] \overset{s}{\Rightarrow} [x]$ in $G/{\sim}$. □

Conversely, for a trace in the abstracted automaton $G/{\sim}$, there does not always exist a corresponding trace in the original automaton. This only holds under additional conditions, in this case that a reverse weak bisimulation is used.

Lemma 2 Let $G = \langle \Sigma, \Pi, X, \rightarrow, X^\circ, \Xi \rangle$, and let ${\sim} \subseteq X \times X$ be a reverse weak bisimulation on G. Then, for all states $w, x \in X$ and all strings $s \in \Sigma^*$ such that $[w] \overset{s}{\Rightarrow} [x]$ in $G/{\sim}$, there exists $w' \in [w]$ such that $w' \overset{s}{\Rightarrow} x$ in G.

Proof. Let $w, x \in X$ and $s \in \Sigma^*$ such that $[w] \overset{s}{\Rightarrow} [x]$ in $G/{\sim}$. Then there exists $s' \in \Sigma_\tau^*$ such that $[w] \xrightarrow{s'} [x]$ and $P_\tau(s') = s$. It is shown by induction on $n = |s'|$ that for $[w] \xrightarrow{s'} [x]$ there exists $w' \in [w]$ such that $w' \overset{P_\tau(s')}{\Longrightarrow} x$.

Base case: $s' = \varepsilon$. $[w] \xrightarrow{\varepsilon} [x]$ implies $[w] = [x]$, and with $x \in [x] = [w]$ it follows that $x \overset{\varepsilon}{\Rightarrow} x$ in G.

Inductive step: $s' = t'\sigma$. Assume that $[w] \xrightarrow{t'} [y] \xrightarrow{\sigma} [x]$. Since $[y] \xrightarrow{\sigma} [x]$ in $G/\!\sim$, by definition of $\rightarrow/\!\sim$ there exist states $x' \in [x]$ and $y' \in [y]$ such that $y' \xrightarrow{\sigma} x'$. Then $x' \sim x$, and since $\sim$ is a reverse weak bisimulation, there exists $y'' \in X$ such that $y'' \stackrel{P_\tau(\sigma)}{\Longrightarrow} x$ and $y'' \sim y'$. By inductive assumption, since $[w] \xrightarrow{t'} [y] = [y'] = [y'']$, there exists $w' \in [w]$ such that $w' \stackrel{P_\tau(t')}{\Longrightarrow} y'' \stackrel{P_\tau(\sigma)}{\Longrightarrow} x$. Since $t'\sigma = s'$, it follows that $w' \stackrel{P_\tau(s')}{\Longrightarrow} x$ in G. □

Having thus established a relationship between the traces in an automaton and its abstraction, the validity of the Non-α Determinisation Rule can be proved.

Proposition 7 Let $G = \langle \Sigma, \Pi, X, \rightarrow, X^\circ, \Xi \rangle$ be a multi-coloured automaton with $\alpha, \omega \in \Pi$, and let $\sim \subseteq X \times X$ be a reverse weak bisimulation on G such that $[x] = \{x\}$ for all $x \in \Xi(\alpha)$. Then $G \simeq_{(\alpha,\omega)} G/\!\sim$.

Proof. Let T be an arbitrary multi-coloured automaton. It is sufficient to show that $G \parallel T$ is (α, ω)-nonblocking if and only if $(G/\!\sim) \parallel T$ is (α, ω)-nonblocking.

First assume that $G \| T$ is (α, ω)-nonblocking, and let $(G/\!\sim) \| T \stackrel{s}{\Rightarrow} ([x], x_T) \in \Xi_{(G/\sim)\|T}(\alpha)$. Then $\tilde{X^\circ} \stackrel{s}{\Rightarrow} [x] \in \tilde{\Xi}(\alpha)$, i.e., $[x^\circ] \stackrel{s}{\Rightarrow} [x]$ for some $x^\circ \in X^\circ$, and by lemma 2, there exists $x' \in [x^\circ]$ such that $x' \stackrel{s}{\Rightarrow} x$ in G. Then $x' \sim x^\circ \in X^\circ$, and thus $X^\circ \stackrel{\varepsilon}{\Rightarrow} x'$ since $\sim$ is a reverse weak bisimulation. Hence, $G \stackrel{s}{\Rightarrow} x$ and $G \parallel T \stackrel{s}{\Rightarrow} (x, x_T)$. As $[x] \in \tilde{\Xi}(\alpha)$, by construction of $\tilde{\Xi}$ there exists $x'' \in [x]$ such that $x'' \in \Xi(\alpha)$. By assumption, it follows that $[x''] = \{x''\}$ and therefore $x = x''$. Thus, $G \parallel T \stackrel{s}{\Rightarrow} (x, x_T) = (x'', x_T) \in \Xi_{G\|T}(\alpha)$. Since $G \parallel T$ is (α, ω)-nonblocking, there exist $t \in \Sigma^*$, $y \in X$, and $y_T \in X_T$ such that $G \parallel T \stackrel{s}{\Rightarrow} (x, x_T) \stackrel{t}{\Rightarrow} (y, y_T) \in \Xi_{G\|T}(\omega)$. By lemma 1, it holds that $[x] \stackrel{t}{\Rightarrow} [y]$ in $G/\!\sim$. Also, $(y, y_T) \in \Xi_{G\|T}(\omega)$ implies $y \in \Xi(\omega)$, i.e., $[y] \in \tilde{\Xi}(\omega)$. It follows that $(G/\!\sim) \parallel T \stackrel{s}{\Rightarrow} ([x], x_T) \stackrel{t}{\Rightarrow} ([y], y_T) \in \Xi_{(G/\sim)\|T}(\omega)$.

Second assume $(G/\!\sim) \parallel T$ is (α, ω)-nonblocking, and let $G \parallel T \stackrel{s}{\Rightarrow} (x, x_T) \in \Xi_{G\|T}(\alpha)$. Then $x^\circ \stackrel{s}{\Rightarrow} x$ for some $x^\circ \in X^\circ$, and by lemma 1 it follows that $[x^\circ] \stackrel{s}{\Rightarrow} [x]$, i.e., $(G/\!\sim) \stackrel{s}{\Rightarrow} [x]$. As $x \in \Xi(\alpha)$, it holds that $[x] \in \tilde{\Xi}(\alpha)$. Thus, $(G/\!\sim) \parallel T \stackrel{s}{\Rightarrow} ([x], x_T) \in \Xi_{(G/\sim)\|T}(\alpha)$. Since $(G/\!\sim) \parallel T$ is (α, ω)-nonblocking, there exist $t \in \Sigma^*$, $y \in X$, and $y_T \in X_T$ such that $(G/\!\sim) \parallel T \stackrel{s}{\Rightarrow} ([x], x_T) \stackrel{t}{\Rightarrow} ([y], y_T) \in \Xi_{(G/\sim)\|T}(\omega)$. Then $[y] \in \tilde{\Xi}(\omega)$, and by construction of $\tilde{\Xi}$ there exists $y' \in [y]$ such that $y' \in \Xi(\omega)$. Then $[x] \stackrel{t}{\Rightarrow} [y] = [y']$ in $G/\!\sim$, and by lemma 2, there exists $x' \in [x]$ such that $x' \stackrel{t}{\Rightarrow} y'$ in G. Since $x \in \Xi(\alpha)$, it holds that $x' = x$ by assumption. It then follows that $G \parallel T \stackrel{s}{\Rightarrow} (x, x_T) = (x', x_T) \stackrel{t}{\Rightarrow} (y', y_T) \in \Xi_{G\|T}(\omega)$. □

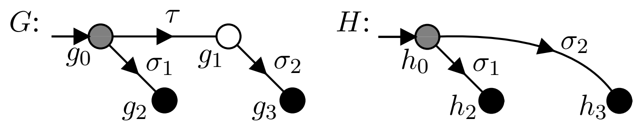

Figure 4: Example application of Silent Continuation Rule.

Complexity. A reverse weak bisimulation relation can be computed in the same way as an observation equivalence using the algorithm in [8], also under the additional constraint that states marked α cannot be merged. Like in the case of observation equivalence, the transition relation first needs to be augmented to bypass any τ-transitions, resulting in the overall complexity $O(|X|^2|\Sigma| \log |X|)$.

4.6 Removal of τ-Transitions Leading to Non-α States

Silent (τ) transitions provide a significant potential for abstraction. If a silent transition links two states that are both *not* marked α, Non-α Determinisation can be used to merge the source and target states of this transition. If both states are marked α, the α-Removal Rule can be used to remove the α-marking of the source state. The Silent Continuation Rule in this section and the Only Silent Outgoing Rule in the following section can address cases where at most one of the two states linked by a silent transition is marked α.

Rule 6 (Silent Continuation Rule) A transition $x \xrightarrow{\tau} y$ with $y \notin \Xi(\alpha)$ can be removed if all transitions originating from state y are copied to state x.

Example 6 Automata G and H in figure 4 are (α, ω)-nonblocking equivalent. The transition $g_0 \xrightarrow{\tau} g_1$ in G leads to a non-α state, so it can be removed after copying the σ_2-transition originating from the target state g_1 to the source state g_0. As a result, the target state g_1 becomes unreachable and can be removed as shown in H.

This simplification relies on the fact that the target state g_1 is *not* marked α, so there is no nonblocking requirement associated with that state. Therefore it can be merged into the source state, leading to much stronger simplification than the Silent Continuation Rule for standard nonblocking [9].

The following definition describes the construction of the simplified automaton formally. To prove the validity of the rule, it is once again necessary to establish the relationship between traces in the original and reduced automaton in Lemma 3 and 4 below. Afterwards the validity of the Silent Continuation Rule is established in proposition 8.

Definition 10 Let $G = \langle \Sigma, \Pi, X, \to, X^\circ, \Xi \rangle$ be a multi-coloured automaton with states $x, y \in X$ such that $x \xrightarrow{\tau} y$. The *target bypass* of transition $x \xrightarrow{\tau} y$ in G is the automaton $G_{x \curvearrowleft y} = \langle \Sigma, \Pi, X, \to_{x \curvearrowleft y}, X^\circ, \Xi_{x \curvearrowleft y} \rangle$ where

$$\begin{aligned}
\to_{x \curvearrowleft y} &= (\to \setminus \{(x, \tau, y)\}) \cup \{\, (x, \sigma, z) \mid y \xrightarrow{\sigma} z \,\} \ ; \\
\Xi_{x \curvearrowleft y}(\pi) &= \begin{cases} \Xi(\pi) \cup \{x\}, & \text{if } y \in \Xi(\pi); \\ \Xi(\pi), & \text{otherwise.} \end{cases}
\end{aligned}$$

Lemma 3 Let $G = \langle \Sigma, \Pi, X, \to, X^\circ, \Xi \rangle$ be a multi-coloured automaton with states $x, y \in X$ such that $x \xrightarrow{\tau} y$. For all states $p, q \in X$ and all strings $s \in \Sigma^*$, if $p \overset{s}{\Rightarrow}_{x \curvearrowleft y} q$ in $G_{x \curvearrowleft y}$, then $p \overset{s}{\Rightarrow} q$ in G.

Proof. Given $p \overset{s}{\Rightarrow}_{x \curvearrowleft y} q$, there exist $\sigma_1, \ldots, \sigma_n \in \Sigma_\tau$, $n \geq 0$ such that $s = P_\tau(\sigma_1 \ldots \sigma_n)$ and $p_0, \ldots, p_n \in X$ such that

$$p = p_0 \xrightarrow{\sigma_1}_{x \curvearrowleft y} p_1 \xrightarrow{\sigma_2}_{x \curvearrowleft y} \cdots \xrightarrow{\sigma_n}_{x \curvearrowleft y} p_n = q \ . \tag{9}$$

It suffices to show $p_{i-1} \overset{P_\tau(\sigma_i)}{\Longrightarrow} p_i$ for $i = 1, \ldots, n$. If $p_{i-1} \xrightarrow{\sigma_i} p_i$, this is trivial. Otherwise, $p_{i-1} = x$ and $y \xrightarrow{\sigma_i} p_i$ by definition 10. This implies $p_{i-1} = x \xrightarrow{\tau} y \xrightarrow{\sigma_i} p_i$, i.e., $p_{i-1} \overset{P_\tau(\sigma_i)}{\Longrightarrow} p_i$ as required. □

Lemma 4 Let $G = \langle \Sigma, \Pi, X, \to, X^\circ, \Xi \rangle$ be a multi-coloured automaton with states $x, y \in X$ such that $x \xrightarrow{\tau} y$. Furthermore, let $p, q \in X$ and $s \in \Sigma^*$ such that $p \overset{s}{\Rightarrow} q$ in G. Then the following statements hold for $G_{x \curvearrowleft y}$.

(i) If $q \neq y$, then $p \overset{s}{\Rightarrow}_{x \curvearrowleft y} q$;

(ii) If $q = y$, then $p \overset{s}{\Rightarrow}_{x \curvearrowleft y} \{q, x\}$.

Proof. Given $p \overset{s}{\Rightarrow} q$, there exists $s' \in \Sigma_\tau^*$ such that $p \xrightarrow{s'} q$ and $P_\tau(s') = s$. It suffices to show that $p \xrightarrow{s'} q$ implies $p \overset{P_\tau(s')}{\Longrightarrow}_{x \curvearrowleft y} F(q)$, where the map $F\colon X \to 2^X$ is defined as $F(z) = \{z\}$ for $z \neq y$ and $F(y) = \{x, y\}$. This claim is proved using induction on the length of the string s'.

Base case: $s' = \varepsilon$. In this case, $P_\tau(s') = \varepsilon$ and $p = q$, and it follows immediately that $p \overset{\varepsilon}{\Rightarrow}_{x \curvearrowleft y} p = q \in F(q)$.

Inductive step: $s' = \sigma t'$ *for* $\sigma \in \Sigma_\tau$ *and* $t' \in \Sigma_\tau^*$. In this case, there is a state $r \in X$ such that $p \xrightarrow{\sigma} r \xrightarrow{t'} q$.

If $p \xrightarrow{\sigma} r$ is *not* the transition $x \xrightarrow{\tau} y$, then $p \xrightarrow{\sigma}_{x \curvearrowleft y} r$ by definition 10, and it follows from the inductive assumption that $r \overset{P_\tau(t')}{\Longrightarrow}_{x \curvearrowleft y} F(q)$. Thus,

$$p \overset{P_\tau(\sigma)}{\Longrightarrow}_{x \curvearrowleft y} r \overset{P_\tau(t')}{\Longrightarrow}_{x \curvearrowleft y} F(q) \ , \tag{10}$$

i.e., $p \overset{P_\tau(s')}{\Longrightarrow}_{x \curvearrowleft y} F(q)$.

Otherwise, if $p \xrightarrow{\sigma} r$ is the transition $x \xrightarrow{\tau} y$, i.e., $p = x$, $r = y$, and $\sigma = \tau$, two more cases need to be considered.

If $t' = \varepsilon$, then $P_\tau(s') = P_\tau(\sigma t') = P_\tau(\tau\varepsilon) = \varepsilon$ and $q = r = y$. In this case, the claim follows because $p = x \overset{\varepsilon}{\Rightarrow}_{x \curvearrowleft y} x \in \{x, y\} = F(y) = F(q)$.

If $t' \neq \varepsilon$, let $t' = \sigma' u'$ for $\sigma' \in \Sigma_\tau$ and $u' \in \Sigma_\tau^*$. Then $y = r \xrightarrow{\sigma'} r' \xrightarrow{u'} q$ for some state $r' \in X$. By definition 10, $y \xrightarrow{\sigma'} r'$ implies that $x \xrightarrow{\sigma'}_{x \curvearrowleft y} r'$, and by inductive assumption, $r' \xrightarrow{u'} q$ implies $r' \overset{P_\tau(u')}{\Longrightarrow}_{x \curvearrowleft y} F(q)$. Thus,

$$p = x \xrightarrow{\sigma'}_{x \curvearrowleft y} r' \overset{P_\tau(u')}{\Longrightarrow}_{x \curvearrowleft y} F(q) \ , \tag{11}$$

and given $P_\tau(\sigma' u') = P_\tau(t') = P_\tau(\tau t') = P_\tau(s')$, it follows that $p \overset{P_\tau(s')}{\Longrightarrow}_{x \curvearrowleft y} F(q)$. □

Proposition 8 Let $G = \langle \Sigma, \Pi, X, \rightarrow, X^\circ, \Xi \rangle$ be a multi-coloured automaton with $\alpha, \omega \in \Pi$ and states $x, y \in X$ such that $x \xrightarrow{\tau} y$, and $y \notin \Xi(\alpha)$. Then $G \simeq_{(\alpha,\omega)} G_{x \curvearrowleft y}$.

Proof. Let T be an arbitrary multi-coloured automaton. It is sufficient to show that $G \parallel T$ is (α, ω)-nonblocking if and only if $G_{x \curvearrowleft y} \parallel T$ is (α, ω)-nonblocking.

First assume that $G \parallel T$ is (α, ω)-nonblocking, and let $G_{x \curvearrowleft y} \parallel T \overset{s}{\Rightarrow} (q, q_T) \in \Xi_{G_{x \curvearrowleft y} \parallel T}(\alpha)$. Then $X^\circ \overset{s}{\Rightarrow}_{x \curvearrowleft y} q \in \Xi_{x \curvearrowleft y}(\alpha)$. By lemma 3 it holds that $X^\circ \overset{s}{\Rightarrow} q$ in G, and by definition 10 it holds that $\Xi_{x \curvearrowleft y}(\alpha) = \Xi(\alpha)$ as $y \notin \Xi(\alpha)$. Thus,

$$G \parallel T \overset{s}{\Rightarrow} (q, q_T) \in \Xi_{x \curvearrowleft y}(\alpha) \times \Xi_T(\alpha) = \Xi_{G \parallel T}(\alpha) \ . \tag{12}$$

Since $G \parallel T$ is (α, ω)-nonblocking, there exist $t \in \Sigma^*$, $r \in X$, and $r_T \in X_T$ such that $G \parallel T \overset{s}{\Rightarrow} (q, q_T) \overset{t}{\Rightarrow} (r, r_T) \in \Xi_{G \parallel T}(\omega)$. Hence, $G \overset{s}{\Rightarrow} q \overset{t}{\Rightarrow} r$. By lemma 4, it follows that either $q \overset{t}{\Rightarrow}_{x \curvearrowleft y} r$ or $q \overset{t}{\Rightarrow}_{x \curvearrowleft y} x$ with $r = y$. In the first case, note $r \in \Xi(\omega) \subseteq \Xi_{x \curvearrowleft y}(\omega)$. In the second case, note that for $y = r \in \Xi(\omega)$, it also holds that $x \in \Xi_{x \curvearrowleft y}(\omega)$ by definition 10. Hence, in both cases $q \overset{t}{\Rightarrow}_{x \curvearrowleft y} \Xi_{x \curvearrowleft y}(\omega)$. It follows that $G_{x \curvearrowleft y} \parallel T \overset{s}{\Rightarrow} (q, q_T) \overset{t}{\Rightarrow} \Xi_{x \curvearrowleft y}(\omega) \times \Xi_T(\omega)$.

Second assume that $G_{x \curvearrowleft y} \parallel T$ is (α, ω)-nonblocking, and let $G \parallel T \overset{s}{\Rightarrow} (q, q_T) \in \Xi_{G \parallel T}(\alpha)$. Then $X^\circ \overset{s}{\Rightarrow} q \in \Xi(\alpha)$ in G. As $q \in \Xi(\alpha)$ and $y \notin \Xi(\alpha)$ by assumption, it holds that $q \neq y$. Then it follows from lemma 4 that $X^\circ \overset{s}{\Rightarrow}_{x \curvearrowleft y} q$, and $q \in \Xi(\alpha) \subseteq \Xi_{x \curvearrowleft y}(\alpha)$ by definition 10. Thus,

$$G_{x \curvearrowleft y} \parallel T \overset{s}{\Rightarrow} (q, q_T) \in \Xi_{x \curvearrowleft y}(\alpha) \times \Xi_T(\alpha) \ . \tag{13}$$

Since $G_{x \frown y} \parallel T$ is (α, ω)-nonblocking, there exist $t \in \Sigma^*$, $r \in X$, and $r_T \in X_T$ such that

$$G_{x \frown y} \parallel T \stackrel{s}{\Rightarrow} (q, q_T) \stackrel{t}{\Rightarrow} (r, r_T) \in \Xi_{x \frown y}(\omega) \times \Xi_T(\omega) \; . \tag{14}$$

Then $q \stackrel{t}{\Rightarrow}_{x \frown y} r$, and by lemma 3 it follows that $q \stackrel{t}{\Rightarrow} r$ in G.

If $r \in \Xi(\omega)$, it follows immediately that $q \stackrel{t}{\Rightarrow} \Xi(\omega)$. Otherwise, if $r \notin \Xi(\omega)$, note that $r \in \Xi_{x \frown y}(\omega)$, which means that $r = x$ and $y \in \Xi(\omega)$ by definition 10. This implies $q \stackrel{t}{\Rightarrow} r = x \stackrel{\tau}{\rightarrow} y \in \Xi(\omega)$, and it again follows that $q \stackrel{t}{\Rightarrow} \Xi(\omega)$.

It follows that $G \parallel T \stackrel{s}{\Rightarrow} (q, q_T) \stackrel{t}{\Rightarrow} \Xi_{G \parallel T}(\omega)$. □

Complexity. The Silent Continuation Rule can be applied at most once to every τ-transition in an automaton, i.e., at most $|X|^2$ applications. Each application involves the copying of all transitions from the target state to the source state, and there may be up to $|\Sigma_\tau||X|$ transitions outgoing from every state. Therefore, the overall complexity to check the applicability of this rule and apply it to all applicable transitions is $O(|X|^3|\Sigma|)$.

It should be noted that the removal of a τ-transition alone does not necessarily lead to a reduction in state space or complexity. Indeed, the Silent Continuation Rule is likely to increase the number of transitions. Its major benefit is that the target state may become unreachable, perhaps after multiple application of the rule, and then can be removed. Also, the rule produces a more regular structure of transitions, which may pave the way for other simplifications.

4.7 Removal of τ-Transitions Originating from Non-α States

The final rule considers the case of a silent transition originating from a non-α state. This case is more difficult, and the following rule is more restrictive than its companion for standard nonblocking [9], because α-markings need to be taken into account in addition to other conditions.

Rule 7 (Only Silent Outgoing Rule) A state x that is not marked α or ω can be removed, if $x \stackrel{\tau}{\rightarrow}$, and x has only τ-transitions outgoing. Incoming transitions to x must be redirected to all the τ-successor states of x.

Example 7 Automata G and H in figure 5 are (α, ω)-nonblocking equivalent. State g_1 in G is not marked α or ω and has only τ-transitions outgoing, so it can be bypassed and removed as shown in H. This simplification is only possible because state g_1 is not marked α or ω. If the state is marked, the nonblocking conditions associated with it need to be retained, and there is no easy way to merge these into one or both of the successor states.

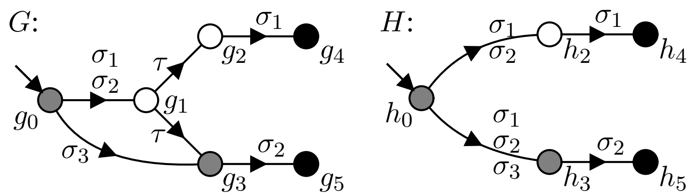

Figure 5: Example application of Only Silent Outgoing Rule.

Following is a formal description of the Only Silent Outgoing Rule. Again, the relationship between traces in the original and the simplified automaton is established before proving the validity of the rule.

Definition 11 Let $G = \langle \Sigma, \Pi, X, \rightarrow, X^\circ, \Xi \rangle$ be a multi-coloured automaton, and let $x \in X$. The *silent outgoing bypass* of state x in G is the automaton $G_{x\frown} = \langle \Sigma, \Pi, X, \rightarrow_{x\frown}, X^\circ_{x\frown}, \Xi \rangle$ where

$$\rightarrow_{x\frown} \; = \; (\rightarrow \setminus \{ (w,\sigma,x) \mid w \xrightarrow{\sigma} x \}) \; \cup \; \{ (w,\sigma,y) \mid w \xrightarrow{\sigma} x \xrightarrow{\tau} y \} \; ;$$

$$X^\circ_{x\frown} \; = \; \begin{cases} (X^\circ \setminus \{x\}) \cup \{ y \in X \mid x \xrightarrow{\tau} y \}, & \text{if } x \in X^\circ, \\ X^\circ, & \text{otherwise.} \end{cases}$$

No state is explicitly removed in this construction. However, the bypassed state x becomes unreachable and can be removed, provided that $x \xrightarrow{\tau} x$ does not hold. If $x \xrightarrow{\tau} x$, then x remains reachable (consider $w \xrightarrow{\sigma} x \xrightarrow{\tau} x$ in the definition of $\rightarrow_{x\frown}$), but such τ-selfloops can be removed first using observation equivalence.

Lemma 5 Let $G = \langle \Sigma, \Pi, X, \rightarrow, X^\circ, \Xi \rangle$, and let $x \in X$. For all states $p, q \in X$ and all strings $s \in \Sigma^*$, if $p \stackrel{s}{\Rightarrow}_{x\frown} q$ in $G_{x\frown}$, then $p \stackrel{s}{\Rightarrow} q$ in G.

Proof. Given $p \stackrel{s}{\Rightarrow}_{x\frown} q$, there exist $\sigma_1, \ldots, \sigma_n \in \Sigma_\tau$, $n \geq 0$ such that $s = P_\tau(\sigma_1 \ldots \sigma_n)$ and $p_0, \ldots, p_n \in X$ such that

$$p = p_0 \xrightarrow{\sigma_1}_{x\frown} p_1 \xrightarrow{\sigma_2}_{x\frown} \cdots \xrightarrow{\sigma_n}_{x\frown} p_n = p \; . \tag{15}$$

It suffices to show $p_{i-1} \stackrel{P_\tau(\sigma_i)}{\Longrightarrow} p_i$ for $i = 1, \ldots, n$. If $p_{i-1} \xrightarrow{\sigma_i} p_i$, this is trivial. Otherwise, $p_{i-1} \xrightarrow{\sigma_i} x \xrightarrow{\tau} p_i$ by definition 11, which also implies $p_{i-1} \stackrel{P_\tau(\sigma_i)}{\Longrightarrow} p_i$.
□

Lemma 6 Let $G = \langle \Sigma, \Pi, X, \rightarrow, X^\circ, \Xi \rangle$, and let $x \in X$ be a state with $x \xrightarrow{\tau}$, which has only τ-transitions outgoing, i.e., $x \xrightarrow{\sigma}$ implies $\sigma = \tau$. Furthermore, let $p, q \in X$ and $s \in \Sigma^*$ such that $p \stackrel{s}{\Rightarrow} q$ in G and $q \neq x$. Then it holds that $p \stackrel{s}{\Rightarrow}_{x\frown} q$ in $G_{x\frown}$.

Proof. Given $p \stackrel{s}{\Rightarrow} q$, there exists $s' \in \Sigma_\tau^*$ such that $p \stackrel{s'}{\to} q$ and $P_\tau(s') = s$. It is shown by induction on the length of s' that $p \stackrel{s'}{\to} q$ with $q \neq x$ in G implies $p \stackrel{P_\tau(s')}{\Longrightarrow}_{x\frown} q$ in $G_{x\frown}$.

Base case: $s' = \varepsilon$. In this case, $P_\tau(s') = \varepsilon$ and $p = q$. It follows immediately that $p \stackrel{\varepsilon}{\Rightarrow}_{x\frown} p = q$.

Inductive step: $s' = \sigma t'$ *for* $\sigma \in \Sigma_\tau$ *and* $t' \in \Sigma_\tau^*$. In this case, there is a state $r \in X$ such that $p \stackrel{\sigma}{\to} r \stackrel{t'}{\to} q$.

If $r \neq x$, then $p \stackrel{\sigma}{\to}_{x\frown} r$ by definition 11, and it follows from the inductive assumption that $r \stackrel{P_\tau(t')}{\Longrightarrow}_{x\frown} q$. Thus,

$$p \stackrel{P_\tau(\sigma)}{\Longrightarrow}_{x\frown} r \stackrel{P_\tau(t')}{\Longrightarrow}_{x\frown} q \ , \tag{16}$$

i.e., $p \stackrel{P_\tau(s')}{\Longrightarrow}_{x\frown} q$.

Otherwise, if $r = x$, note that $x = r \stackrel{t'}{\to} q \neq x$ and hence $t' \neq \varepsilon$. Then let $t' = \sigma' u'$ for $\sigma' \in \Sigma_\tau$ and $u' \in \Sigma_\tau^*$. Then $r \stackrel{\sigma'}{\to} r' \stackrel{u'}{\to} q$ for some state $r' \in X$. Also $\sigma' = \tau$ since $x = r$ has only τ-transitions outgoing, i.e., $p \stackrel{\sigma}{\to} r \stackrel{\tau}{\to} r'$. By definition 11, it follows that $p \stackrel{\sigma}{\to}_{x\frown} r'$. By inductive assumption, $r' \stackrel{u'}{\to} q$ implies $r' \stackrel{P_\tau(u')}{\Longrightarrow}_{x\frown} q$. Thus, $p \stackrel{\sigma}{\to}_{x\frown} r' \stackrel{P_\tau(u')}{\Longrightarrow}_{x\frown} q$, and given $P_\tau(\sigma u') = P_\tau(\sigma\tau u') = P_\tau(\sigma\sigma' u') = P_\tau(\sigma t') = P_\tau(s')$, it follows that $p \stackrel{P_\tau(s')}{\Longrightarrow}_{x\frown} q$. $\square$

Proposition 9 Let $G = \langle \Sigma, \Pi, X, \to, X^\circ, \Xi \rangle$, and let $x \in X$ be a state with $x \stackrel{\tau}{\to}$ and $x \notin \Xi(\alpha) \cup \Xi(\omega)$, which has only τ-transitions outgoing, i.e., $x \stackrel{\sigma}{\to}$ implies $\sigma = \tau$. Then $G \simeq_{(\alpha,\omega)} G_{x\frown}$.

Proof. Let T be an arbitrary multi-coloured automaton. It is sufficient to show that $G \parallel T$ is (α,ω)-nonblocking if and only if $G_{x\frown} \parallel T$ is (α,ω)-nonblocking.

First assume that $G \parallel T$ is (α,ω)-nonblocking, and let $G_{x\frown} \parallel T \stackrel{s}{\Rightarrow} (q, q_T) \in \Xi_{G_{x\frown} \parallel T}(\alpha)$. Then $X^\circ_{x\frown} \stackrel{s}{\Rightarrow}_{x\frown} q \in \Xi(\alpha)$. Then there exists $x^\circ \in X^\circ_{x\frown}$ such that $x^\circ \stackrel{s}{\Rightarrow}_{x\frown} q \in \Xi(\alpha)$. Applying lemma 5 it follows that $x^\circ \stackrel{s}{\Rightarrow} q \in \Xi(\alpha)$. Also note that, if $x^\circ \notin X^\circ$, then $x^\circ \in X^\circ_{x\frown}$ means that $x \in X^\circ$ and $x \stackrel{\tau}{\to} x^\circ$ by definition 11. Thus, $G \stackrel{s}{\Rightarrow} q$ and therefore $G \parallel T \stackrel{s}{\Rightarrow} (q, q_T) \in \Xi_{G\parallel T}(\alpha)$. Since $G \parallel T$ is (α,ω)-nonblocking, there exist $t \in \Sigma^*$, $r \in X$, and $r_T \in X_T$ such that $G \parallel T \stackrel{s}{\Rightarrow} (q, q_T) \stackrel{t}{\Rightarrow} (r, r_T) \in \Xi_{G\parallel T}(\omega)$. Hence, $G \stackrel{s}{\Rightarrow} q \stackrel{t}{\Rightarrow} r$. Since $r \in \Xi(\omega)$ and $x \notin \Xi(\omega)$, it holds that $r \neq x$. Then $q \stackrel{t}{\Rightarrow}_{x\frown} r \in \Xi(\omega)$ by lemma 6, and this means $G_{x\frown} \parallel T \stackrel{s}{\Rightarrow} (q, q_T) \stackrel{t}{\Rightarrow} \Xi_{G_{x\frown}\parallel T}(\omega)$.

Second assume that $G_{x\frown} \parallel T$ is (α,ω)-nonblocking, and let $G \parallel T \stackrel{s}{\Rightarrow} (q, q_T) \in \Xi_{G\parallel T}(\alpha)$. Then there exists $x^\circ \in X^\circ$ such that $x^\circ \stackrel{s}{\Rightarrow} q \in \Xi(\alpha)$ in G. As $q \in \Xi(\alpha)$

and $x \notin \Xi(\alpha)$ by assumption, it holds that $q \neq x$, which implies $x^\circ \stackrel{s}{\Rightarrow}_{x\frown} q$ by lemma 6. If $x^\circ \neq x$, it also holds that $x^\circ \in X^\circ_{x\frown}$ by definition 11. If $x^\circ = x$, note that x has only τ-transitions outgoing, and $q \neq x$, so the path $x = x^\circ \stackrel{s}{\Rightarrow} q$ is not empty and has the form $x \stackrel{\tau}{\rightarrow} y \stackrel{s}{\Rightarrow} q$ with $y \in X^\circ_{x\frown}$. Thus, $X^\circ_{x\frown} \stackrel{s}{\Rightarrow}_{x\frown} q \in \Xi(\alpha)$ and

$$G_{x\frown} \parallel T \stackrel{s}{\Rightarrow} (q, q_T) \in \Xi_{G\parallel T}(\alpha) = \Xi_{G_{x\frown}\parallel T}(\alpha) \; . \tag{17}$$

Since $G_{x\frown} \parallel T$ is (α, ω)-nonblocking, there exists $t \in \Sigma^*$ such that $G_{x\frown} \parallel T \stackrel{s}{\Rightarrow} (q, q_T) \stackrel{t}{\Rightarrow} \Xi_{G_{x\frown}\parallel T}(\omega)$. Then there exists a state $r \in \Xi(\omega)$ such that $q \stackrel{t}{\Rightarrow}_{x\frown} r$, and by lemma 5 also $q \stackrel{t}{\Rightarrow} r$ in G. Hence, $G \parallel T \stackrel{s}{\Rightarrow} (q, q_T) \stackrel{t}{\Rightarrow} \Xi_{G\parallel T}(\omega)$. $\square$

Complexity. To check whether the Only Silent Outgoing Rule is applicable to a state, it must be confirmed that it is not marked and has at least one and only τ-transitions outgoing. Using appropriate data structures, this can be done in constant complexity. Applying the rule requires all incoming transitions to be copied to all τ-successor states. There can be up to $|X||\Sigma_\tau|$ incoming transitions and up to $|X|$ τ-successors per state. Then the complexity to check and apply the Only Silent Outgoing Rule to all states of an automaton is $O(|X|^3|\Sigma|)$.

5 Conclusions

This working paper shows how generalised nonblocking can be verified compositionally by simplifying individual components of a system before or while composing them. Seven rewrite rules preserving generalised nonblocking have been proposed, which can substantially reduce the number of states of the automata encountered during verification. The rules have been chosen to be computationally feasible, while still covering a wide range of situations encountered in nondeterministic automata. Although developed specifically for generalised nonblocking, the results presented here are also applicable to standard nonblocking.

References

[1] S. Balemi, G. J. Hoffmann, P. Gyugyi, H. Wong-Toi, and G. F. Franklin. Supervisory control of a rapid thermal multiprocessor. *IEEE Transactions on Automatic Control*, 38(7):1040–1059, July 1993.

[2] Bertil Brandin and François Charbonnier. The supervisory control of the automated manufacturing system of the AIP. In *Proceedings of Rensselaer's 4th International Conference on Computer Integrated Manufacturing and Automation Technology*, pages 319–324, Troy, NY, USA, 1994.

[3] C. G. Cassandras and S. Lafortune. *Introduction to Discrete Event Systems*. Kluwer Academic Publishers, September 1999.

[4] Edmund M. Clarke, Jr., Orna Grumberg, and Doron A. Peled. *Model Checking*. MIT Press, 1999.

[5] Max H. de Queiroz, José E. R. Cury, and W. M. Wonham. Multi-tasking supervisory control of discrete-event systems. In *Proceedings of 7th International Workshop on Discrete Event Systems, WODES '04*, pages 175–180, Reims, France, September 2004.

[6] P. Dietrich, R. Malik, W. M. Wonham, and B. A. Brandin. Implementation considerations in supervisory control. In B. Caillaud, P. Darondeau, L. Lavagno, and X. Xie, editors, *Synthesis and Control of Discrete Event Systems*, pages 185–201. Kluwer Academic Publishers, 2002.

[7] M. Fabian and R. Kumar. Mutually nonblocking supervisory control of discrete event systems. In *Proceedings of 36th IEEE Conference on Decision and Control, CDC '97*, pages 2970–2975, San Diego, CA, USA, 1997.

[8] Jean-Claude Fernandez. An implementation of an efficient algorithm for bisimulation equivalence. *Science of Computer Programming*, 13:219–236, 1990.

[9] Hugo Flordal and Robi Malik. Compositional verification in supervisory control. *SIAM Journal of Control and Optimization*, 48(3):1914–1938, 2009.

[10] C. A. R. Hoare. *Communicating Sequential Processes*. Prentice-Hall, 1985.

[11] John E. Hopcroft, Rajeev Motwani, and Jeffrey D. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley, 2001.

[12] R. Kumar and M. A. Shayman. Non-blocking supervisory control of nondeterministic discrete event systems. In *Proceedings of American Control Conference*, pages 1089–1093, Baltimore, MD, USA, 1994.

[13] Ryan J. Leduc, Bertil A. Brandin, Mark Lawford, and W. M. Wonham. Hierarchical interface-based supervisory control—part I: Serial case. *IEEE Transactions on Automatic Control*, 50(9):1322–1335, September 2005.

[14] Petra Malik. *From Supervisory Control to Nonblocking Controllers for Discrete Event Systems*. PhD thesis, University of Kaiserslautern, Kaiserslautern, Germany, 2003.

[15] Robi Malik and Ryan Leduc. Generalised nonblocking. In *Proceedings of the 9th International Workshop on Discrete Event Systems, WODES'08*, pages 340–345, Göteborg, Sweden, May 2008.

[16] Robi Malik, David Streader, and Steve Reeves. Conflicts and fair testing. *International Journal of Foundations of Computer Science*, 17(4):797–813, 2006.

[17] R. Milner. *A Calculus of Communicating Systems*, volume 92 of *LNCS*. Springer-Verlag, 1980.

[18] Patrícia N. Pena, José E. R. Cury, and Stéphane Lafortune. New results on testing modularity of local supervisors using abstractions. In *Proceedings of the 11th IEEE International Conference on Emerging Technologies and Factory Automation, ETFA '06*, pages 950–956, Prague, Czech Republic, September 2006.

[19] Peter J. G. Ramadge and W. Murray Wonham. The control of discrete event systems. *Proceedings of the IEEE*, 77(1):81–98, January 1989.

[20] Yanjun Wen, Ji Wang, and Zhichang Qi. Reverse observation equivalence between labelled state transition systems. In *Proceedings of 1st International Colloquium on Theoretical Aspects of Computing, ICTAC '04*, pages 204–219, Guiyang, China, September 2004.

[21] K. C. Wong, J. G. Thistle, R. P. Malhame, and H.-H. Hoang. Supervisory control of distributed systems: Conflict resolution. *Discrete Event Dynamic Systems: Theory and Applications*, 10:131–186, 2000.