Model abstraction of nondeterministic finite state automata in supervisor synthesis

Citation for published version (APA):

Document status and date:
Published: 01/01/2008

Document Version:
Publisher’s PDF, also known as Version of Record (includes final page, issue and volume numbers)

Please check the document version of this publication:
• A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher's website.
• The final author version and the galley proof are versions of the publication after peer review.
• The final published version features the final layout of the paper including the volume, issue and page numbers.

Link to publication

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

• Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
• You may not further distribute the material or use it for any profit-making activity or commercial gain
• You may freely distribute the URL identifying the publication in the public portal.

If the publication is distributed under the terms of Article 25fa of the Dutch Copyright Act, indicated by the “Taverne” license above, please follow below link for the End User Agreement:
www.tue.nl/taverne

Take down policy
If you believe that this document breaches copyright please contact us at:
openaccess@tue.nl
providing details and we will investigate your claim.
Model Abstraction of Nondeterministic Finite State Automata in Supervisor Synthesis

Rong Su, Jan H. van Schuppen and Jacobus E. Rooda\textsuperscript{1, 2}

ISSN: 1872-1567
Abstract

Blockingness is one of the major obstacles that need to be overcome in the Ramadge-Wonham supervisory synthesis paradigm, especially for large systems. In this paper we propose an abstraction technique to overcome this difficulty. We first provide details of this abstraction technique, then describe how to apply it to supervisor synthesis, in which plant models are nondeterministic but specifications and supervisors are deterministic. We show that a nonblocking state-controllable state-observable (or state-normal) supervisor for an abstraction of a plant under a specification is guaranteed to be a nonblocking state-controllable state-observable (or state-normal) supervisor of the original plant under the same specification. The reverse statement is also true, if every observable event is contained in the abstraction and the plant model is marking aware with respect to the alphabet of the abstraction.
1 Introduction

The automaton-based Ramadge-Wonham (RW) supervisory control paradigm first appeared in the control literature in 1982, which was subsequently summarized in the well-known journal papers [1, 2]. Since then there has been a large volume of literature under the same paradigm. In the RW paradigm the main difficulty of supervisor synthesis for a large system is to achieve the nonblockingness. The reason is that the total number of states of a plant model increases quickly when the number of local components increases, due to the synchronous product which incurs cartesian product over automata. To overcome this difficulty, some authors attempt to introduce sufficient conditions, which allow local supervisor synthesis. For example, in [3] the authors propose the concept of modularity, which is then extended to the concept of local modularity in [4]. When local supervisors are (locally) modular, a globally nonblocking supervisory control is achieved. Nevertheless, testing (local) modularity itself usually imposes prohibitive computational complexity. Another notable work is presented in [7, 8], where, by imposing interface consistency and level-wise controllability among subsystems and local supervisors in a hierarchical setup, a very large nonblocking control problem may be solved, e.g. the system size reaches $10^{21}$ in the AIP example [8]. But the approach does not tell how to deliberately and systematically design interfaces that allow synthesis of local supervisors that satisfy those properties. Instead, it assumes that those interfaces are given before synthesis, as mentioned in [9]. In [10] the authors present an interesting approach, which is aimed to synthesize a state-feedback supervisor. By introducing the concept of state tree structures, the authors propose to represent product states in binary decision diagrams (BDDs), upon which the power of symbolic computation (as manifested by the manipulation of BDDs) is fully utilized. It has been shown in [10] that a system with $10^{24}$ states can be accommodated. Nevertheless, this approach is essentially a centralized approach. No matter how efficient the symbolic computational technique is, such efficiency can never completely overcome the complexity issue in an industrial system that usually consists of hundreds or thousands of components. Besides, the proposed approach does not deal with partial observation.

In this paper we will discuss how to synthesize a supervisor by using an appropriate abstraction of a system. Our first contribution is to present a novel automaton-based abstraction technique. The idea of abstraction has been known in the literature, e.g. in [11] abstraction is used in the modular and hierarchical supervisor synthesis; it is also used in [12] for testing the nonblocking property, and in [13] for decentralized control. Nevertheless, their approaches are language-based, which use natural projections. To make sure nonblocking information will not be incorrectly masked out by abstraction, natural projections have to possess the observer property [6], which may not always hold by a natural projection. Although a natural projection can always be modified to become an observer (with respect to a specific language) [14], such a modification has a potential drawback in the sense that the alphabet of the codomain of the projection may be fairly large for the sake of achieving the observer property, and the consequence is that the size of the projected image may not be small enough to allow supervisor synthesis for large systems. Our abstraction technique is automaton-based, thus different from those language-based abstraction techniques. There have been several research works on automaton abstraction, e.g. [15] [24] [25] [26] [27]. [15] aims to achieve weak bisimilarity between an automaton and its abstraction. [24] [25] [26] [27] first use silence events to replace internal events, then apply rewriting rules to ensure that appropriate equivalence relations, e.g. conflict equivalence in [24] [27], supervision equivalence in [25] and synthesis equivalence in [26], hold between automata before and after rewriting. Our approach does not use silence events, and its primary goal is to create an abstraction for an automaton $G$, which is not necessarily weak bisimilar to $G$, such that any automaton $S$,
whose alphabet is the same as that of the abstraction and is nonconflicting with the abstraction, must be nonconflicting with \( G \). If \( G \) is marking aware, then it is automatically true that \( S \) is nonconflicting with \( G \) implies \( S \) is nonconflicting with the abstraction - at this point, our approach is close to achieving conflict equivalence, but with a procedure much simpler than those rewriting rules and no silence events are needed. Our second contribution is to utilize the automaton-based abstraction in supervisor synthesis and provide conditions under which the existence of a nonblocking supervisor for an abstraction of a plant guarantees the existence of a nonblocking supervisor for the original plant, and vice versa. Since abstraction usually results in nondeterministic automata, we consider supervisor synthesis with a nondeterministic plant model but with a deterministic specification. The supervisor is required to be deterministic as well. There have been a large volume of work on supervisor synthesis for nondeterministic systems, but most do not use abstraction, e.g. \([23]\) \([18]\) \([5]\) \([19]\) \([20]\) \([21]\). Although \([15]\) \([24]\) \([25]\) \([26]\) \([27]\) utilize abstraction in synthesis, their abstraction techniques are different from ours. Our third contribution is to introduce the concept of state normality, which allows computing a supremal nonblocking supervisor for a nondeterministic system, which is important for synthesis, especially considering that nondeterministic systems always behave like systems under partial observation.

This paper is organized as follows. In Section II we introduce an abstraction technique over nondeterministic automata. Then in Section III we describe how to apply it to synthesize a deterministic supervisor for a nondeterministic plant model under a deterministic specification. After an illustrative example in Section IV, conclusions are stated in Section V. Long proofs are presented in the Appendix.

### 2 Automaton Abstraction and Relevant Properties

In this section we follow the notations used in \([16]\). We first briefly review concepts of languages, natural projection, synchronous product and automaton product, then introduce the concept of automaton abstraction. After that, we present properties of abstraction which will be used in supervisor synthesis.

#### 2.1 Concepts of Languages, Automaton Product and Abstraction

Let \( \Sigma \) be a finite alphabet, and \( \Sigma^* \) the Kleene closure of \( \Sigma \), i.e. the collection of all finite sequences of events taken from \( \Sigma \). Given two strings \( s, t \in \Sigma^* \), \( s \) is called a prefix substring of \( t \), written as \( s \preceq t \), if there exists \( s' \in \Sigma^* \) such that \( ss' = t \), where \( ss' \) denotes the concatenation of \( s \) and \( s' \). We use \( \epsilon \) to denote the empty string of \( \Sigma^* \) such that for any string \( s \in \Sigma^* \), \( \epsilon s = se = s \). A subset \( L \subseteq \Sigma^* \) is called a language. \( L \) is called the prefix closure of \( L \). \( L \) is called prefix closed if \( L = \overline{L} \). Given two languages \( L, L' \subseteq \Sigma^* \), \( LL' := \{ ss' \in \Sigma^* | s \in L \land s' \in L' \} \).

Let \( \Sigma' \subseteq \Sigma \). A map \( P : \Sigma^* \rightarrow \Sigma'^* \) is the natural projection with respect to \((\Sigma, \Sigma')\), if

1. \( P(\epsilon) = \epsilon \)
2. \( (\forall \sigma \in \Sigma) P(\sigma) := \begin{cases} \sigma & \text{if } \sigma \in \Sigma' \\ \epsilon & \text{otherwise} \end{cases} \)
3. \((\forall \sigma \in \Sigma^*) P(\sigma) = P(s)P(\sigma)\)

Given a language \(L \subseteq \Sigma^*, P(L) := \{P(s) \subseteq \Sigma^* | s \in L\}\). For any two languages \(L, L' \subseteq \Sigma^*,\) we can show that \(P(LL') = P(L)P(L')\). The inverse image mapping of \(P\) is

\[
P^{-1} : 2^{\Sigma^*} \rightarrow 2^{\Sigma^*} : L \mapsto P^{-1}(L) := \{s \in \Sigma^* | P(s) \in L\}
\]

Given \(L_1 \subseteq \Sigma_1^*\) and \(L_2 \subseteq \Sigma_2^*\), the \textit{synchronous product} of \(L_1\) and \(L_2\) is defined as:

\[
L_1 \parallel L_2 := P_1^{-1}(L_1) \cap P_2^{-1}(L_2) = \{s \in (\Sigma_1 \cup \Sigma_2)^* | P_1(s) \in L_1 \land P_2(s) \in L_2\}
\]

where \(P_1 : (\Sigma_1 \cup \Sigma_2)^* \rightarrow \Sigma_1^*\) and \(P_2 : (\Sigma_1 \cup \Sigma_2)^* \rightarrow \Sigma_2^*\) are natural projections. It has been shown [16] that \(\parallel\) is commutative and associative. Next, we introduce automaton product and abstraction.

Given a nondeterministic finite-state automaton \(G = (X, \Sigma, \xi, x_0, X_m)\), \(X\) stands for the state set, \(\Sigma\) for the alphabet, \(\xi : X \times \Sigma \rightarrow 2^X\) for the nondeterministic transition function, \(x_0\) for the initial state and \(X_m\) for the marker state set. As usual, we extend the domain of \(\xi\) from \(X \times \Sigma\) to \(X \times \Sigma^*\), where \(\xi(x, s) := \{x' \in X | \exists x'' \in \xi(x,s) \land x'' \in \xi(x'', s)\}\). We bring in a new event symbol \(\tau\). An automaton \(G = (X, \Sigma \cup \{\tau\}, \xi, x_0, X_m)\) is \textit{standardized} if

\[
x_0 \notin X_m \land (\forall x \in X) [\xi(x, \tau) \neq \emptyset \iff x = x_0] \land (\forall x \in X - \{x_0\}) (\forall \sigma \in \Sigma) x_0 \notin \xi(x, \sigma)
\]

A standardized automaton is nothing but an automaton, whose initial state \(x_0\) only has outgoing transitions with the same label \(\tau\), and no incoming transitions. For an ordinary automaton \(G = (X, \Sigma, \xi, x_0, X_m)\) we can convert it into a standardized automaton by simply: (1) extend the alphabet to \(\Sigma \cup \{\tau\}\); (2) add a new state \(x'_0\); (3) define a new transition map \(\xi'\) such that \(\xi'(x'_0, \tau) = \{x_0\}\) and for any \((x, \sigma) \in X \times \Sigma\) we have \(\xi'(x, \sigma) = \xi(x, \sigma)\). The resultant automaton \(G' = (X \cup \{x'_0\}, \Sigma \cup \{\tau\}, \xi', x'_0, X_m)\) is a standardized automaton. From now on, unless specified explicitly, we assume that each alphabet \(\Sigma\) contains \(\tau\), and \(\phi(\Sigma)\) is the collection of all standardized finite state automata, whose alphabet is \(\Sigma\). The role of \(\tau\) will be explained shortly. We now introduce automaton product.

Given two nondeterministic automata \(G_i = (X_i, \Sigma_i, \xi_i, x_{0,i}, X_{m,i}) \in \phi(\Sigma_i)\) \((i = 1, 2)\), the \textit{product} of \(G_1\) and \(G_2\), written as \(G_1 \times G_2\), is an automaton in \(\phi(\Sigma_1 \cup \Sigma_2)\) such that

\[
G_1 \times G_2 = (X_1 \times X_2, \Sigma_1 \cup \Sigma_2, \xi_1 \times \xi_2, (x_{0,1}, x_{0,2}), X_{m,1} \times X_{m,2})
\]

where \(\xi_1 \times \xi_2 : X_1 \times X_2 \times (\Sigma_1 \cup \Sigma_2) \rightarrow 2^{X_1 \times X_2}\) is defined as follows,

\[
(\xi_1 \times \xi_2)((x_1, x_2), \sigma) := \begin{cases} 
\xi_1(x_1, \sigma) \times \{x_2\} & \text{if } \sigma \in \Sigma_1 - \Sigma_2 \\
\{x_1\} \times \xi_2(x_2, \sigma) & \text{if } \sigma \in \Sigma_2 - \Sigma_1 \\
\xi_1(x_1, \sigma) \times \xi_2(x_2, \sigma) & \text{if } \sigma \in \Sigma_1 \cap \Sigma_2 
\end{cases}
\]

Clearly, \(\times\) is commutative and associative. By a slight abuse of notations, from now on we use \(G_1 \times G_2\) to denote its reachability part. \(\xi_1 \times \xi_2\) is extended to \(X_1 \times X_2 \times (\Sigma_1 \cup \Sigma_2)^* \rightarrow 2^{X_1 \times X_2}\).

We can easily see that product of two standardized automata is still a standardized automaton. Next, we introduce automaton abstraction, which requires the following concept
of marking weak bisimilarity.

**Definition 2.1.** Given \( G = (X, \Sigma, \xi, x_0, X_m) \), let \( \Sigma' \subseteq \Sigma \) and \( P : \Sigma^* \rightarrow \Sigma'^* \) be the natural projection. A **marking weak bisimulation** relation on \( X \) with respect to \( \Sigma' \) is an equivalence relation \( R \subseteq \{ (x, x') \in X \times X | x \in X_m \iff x' \in X_m \} \) such that,

\[
(\forall (x, x') \in R)(\forall s \in \Sigma^*)(\forall y \in \xi(x, s))(\exists s' \in \Sigma^*)P(s) = P(s') \land (\exists y' \in \xi(x', s'))(y, y') \in R
\]

The largest marking weak bisimulation relation on \( X \) with respect to \( \Sigma' \) is called **marking weak bisimilarity** on \( X \) with respect to \( \Sigma' \), written as \( \approx_{\Sigma', G} \).

Marking weak bisimulation relation is the same as weak bisimulation relation described in [17], except for the special treatment on marker states. We now introduce abstraction.

**Definition 2.2.** Given \( G = (X, \Sigma, \xi, x_0, X_m) \), let \( \Sigma' \subseteq \Sigma \). The **automaton abstraction** of \( G \) with respect to \( \approx_{\Sigma', G} \) is an automaton \( G' = (Y, \Sigma', \eta, y_0, Y_m) \) where

1. \( Y := X/ \approx_{\Sigma', G} := \{ < x > := \{ x' \in X | (x, x') \in \approx_{\Sigma', G} \} | x \in X \} \)
2. \( y_0 := < x_0 > \)
3. \( Y_m := \{ y \in Y | y \cap X_m \neq \emptyset \} \)
4. \( \eta : Y \times \Sigma' \rightarrow 2^Y \), where for any \( (y, \sigma) \in Y \times \Sigma' \),

\[
\eta(y, \sigma) := \{ y' \in Y | (\exists x \in y)(\exists u, u' \in (\Sigma - \Sigma')^* \xi(x, u\sigma u') \cap y' \neq \emptyset) \}
\]

We can easily check that, if \( G \) is standardized, then \( G' = \approx_{\Sigma', G} \) is also standardized. The time complexity of computing \( G' = \approx_{\Sigma', G} \) is mainly resulted from computing \( X/ \approx_{\Sigma', G} \), which can be estimated as follows. We first define a new automaton \( G'' = (X, \Sigma', \xi'', x_0, X_m) \), where for any \( x, x' \in X \) and \( \sigma \in \Sigma \), \( x' \in \xi''(x, \sigma) \) if there exist \( u, u' \in (\Sigma - \Sigma')^* \) such that \( x' \in \xi(x, u\sigma u') \). Then we compute \( X/ \approx_{\Sigma', G''} \), and we can show that the result is equal to \( X/ \approx_{\Sigma', G} \). The total number of transitions in \( G'' \) is no more than \( mn^2 \), where \( n = |X| \) and \( m \) is the number of transitions in \( G \). Based on a result shown in [22], the time complexity of computing \( X/ \approx_{\Sigma', G''} \) is \( O(mn^2 \log n) \) if we ignore the complexity caused by checking the condition \( "x \in X_m \iff x' \in X_m" \) in Def. 2.1. If we consider this extra condition, then the overall complexity is \( O(n(n - 1) + mn^2 \log n) \), because we need to check at most \( n(n - 1) \) pairs of states.

From now on, when \( G \) is clear from the context, we simply use \( \approx_{\Sigma'} \) to denote \( \approx_{\Sigma', G} \), and use \( < x >_{\Sigma'} \) for an element of \( X/ \approx_{\Sigma', G} \). If \( \Sigma' \) is also clear from the context, then we simply use \( < x >_G \) for \( < x >_{\Sigma'} \).

As an illustration, suppose a standardized automaton \( G \in \phi(\Sigma) \) is depicted in Figure 1, where the alphabet \( \Sigma = \{ \tau, a, b, c \} \). We take \( \Sigma' = \{ \tau, c \} \). Then we have

\[
X/ \approx_{\Sigma'} = \{ < 0 > := \{ 0 \}, < 1 > := \{ 1, 2, 3 \}, < 4 > := \{ 4 \} \}
\]

The abstraction \( G' = \approx_{\Sigma'} \) is depicted in Figure 1. Next, we present properties of automaton abstraction.

5 Automaton Abstraction and Relevant Properties
2.2 Properties of Automaton Abstraction

We first define a map $B : \phi(\Sigma) \rightarrow 2^{\Sigma^*}$ with

$$(\forall G \in \phi(\Sigma)) \ B(G) := \{ s \in \Sigma^* | \xi(x_0, s) \neq \emptyset \land (\exists x \in \xi(x_0, s))(\forall s' \in \Sigma^*) \xi(x, s') \cap X_m = \emptyset \}$$

Any string $s \in B(G)$ can lead to a state $x$, from which no marker state is reachable, i.e. for any $s \in \Sigma^*$, $\xi(x, s) \cap X_m = \emptyset$. Such a state $x$ is called a blocking state of $G$, and we call $B(G)$ the blocking set of $G$. A state that is not a blocking state is called a nonblocking state. We say $G$ is nonblocking if $B(G) = \emptyset$. Similarly, define another map $N : \phi(\Sigma) \rightarrow 2^{\Sigma^*}$ with

$$(\forall G \in \phi(\Sigma)) \ N(G) := \{ s \in \Sigma^* | \xi(x_0, s) \cap X_m \neq \emptyset \}$$

We call $N(G)$ the nonblocking set of $G$, which is simply the language recognized by $G$. It is possible that $B(G) \cap N(G) \neq \emptyset$, due to nondeterminism.

**Definition 2.3.** An automaton $G = (X, \Sigma, \xi, x_0, X_m)$ is marking aware with respect to $\Sigma' \subseteq \Sigma$, if

$$(\forall x \in X - X_m)(\forall s \in \Sigma^*) \xi(x, s) \cap X_m \neq \emptyset \Rightarrow P(s) \neq \epsilon$$

where $P : \Sigma^* \rightarrow \Sigma'^*$ is the natural projection. $\square$

If $G$ is marking aware with respect to $\Sigma'$, then any string $s$ reaching a marker state from a non-marker state must contain at least one event in $\Sigma'$. A sufficient and necessary condition to make $G$ marking aware with respect to $\Sigma'$ is to put in $\Sigma'$ every event that labels a transition from a non-marker state to a marker state, namely $\{ \sigma \in \Sigma | (\exists x \in X - X_m)(\exists x' \in X_m) x' \in \xi(x, \sigma) \} \subseteq \Sigma'$. We have the following result, which will be extensively used in the rest of this paper.

**Proposition 2.4.** Given $G \in \phi(\Sigma)$, let $\Sigma' \subseteq \Sigma$, and $P : \Sigma^* \rightarrow \Sigma'^*$ be the natural projection. Then

1. $P(B(G)) \subseteq B(G/\approx_{\Sigma'})$ and $P(N(G)) = N(G/\approx_{\Sigma'})$.

2. If $G$ is marking aware with respect to $\Sigma'$, then $P(B(G)) = B(G/\approx_{\Sigma'})$. $\square$

As an illustration of Prop. 2.4, Figure 2 depicts an example, where $\Sigma = \{ \tau, a, b \}$ and $\Sigma' = \{ \tau, b \}$. We can check that $P(N(G)) = \{ \tau \} = N(G/\approx_{\Sigma'})$. But $P(B(G)) = \{ \tau b \}$.
Def. 2.5 says that, if \( P(B(G)) \subseteq B(G/ \approx_{\Sigma'}) \). In this example, to make \( G \) marking aware with respect to \( \Sigma' \), \( a \) must be included in \( \Sigma' \). If we set \( \Sigma' = \{ \tau, a \} \), then \( P(B(G)) = B(G/ \approx_{\Sigma'}) \), as predicted in Prop. 2.4.

We now explain why we introduce the event \( \tau \) and standardized automata. If we don’t use it, then we may not always have \( P(\tau) \subseteq B(\approx_{\Sigma'}) \) and \( P(\tau) \subseteq B(G/ \approx_{\Sigma'}) \) (let alone \( P(\tau) = B(G/ \approx_{\Sigma'}) \)), which may cause supervisor synthesis to fail, in the sense that a nonblocking supervisor (whose precise definition will be given later) of \( G/ \approx_{\Sigma'} \) may not be a nonblocking one for \( G \). This will be clear when we introduce supervisor synthesis.

Next, we want to answer the question whether \( (G/ \approx_{\Sigma'}) \times S \) for some automaton \( S \) is nonblocking if and only if \( (G/ \approx_{\Sigma'}) \times S \) is nonblocking. Given an automaton \( G = (X, \Sigma, \xi, x_0, X_m) \), for each \( x \in X \), let

\[
N_G(x) := \{ s \in \Sigma^* | [x, s] \cap X_m \neq \emptyset \}
\]

We now introduce the following concept, which will be extensively used in this paper.

**Definition 2.5.** Given automata \( G_i = (X_i, \Sigma_i, \xi_i, x_{i,0}, X_{i,m}) \) \( (i = 1, 2) \), we say \( G_1 \) is nonblocking preserving with respect to \( G_2 \), denoted as \( G_1 \subseteq G_2 \), if \( B(G_1) \subseteq B(G_2) \), \( N(G_1) = N(G_2) \) and for any \( s \in N(G_1) \),

\[
(\forall x_1 \in \xi_1(x_{1,0}, s)) (\exists x_2 \in \xi_2(x_{2,0}, s)) N_{G_2}(x_2) \subseteq N_{G_1}(x_1) \land [x_1 \in X_{1,m} \iff x_2 \in X_{2,m}]
\]

\( G_1 \) is nonblocking equivalent to \( G_2 \), denoted as \( G_1 \equiv G_2 \), if \( G_1 \subseteq G_2 \) and \( G_2 \subseteq G_1 \).

Def. 2.5 says that, if \( G_1 \) is nonblocking preserving with respect to \( G_2 \) then their individual nonblocking parts are equal, but \( G_2 \)’s blocking behavior may be larger. If blocking behaviors are also equal, then \( G_1 \) and \( G_2 \) are nonblocking equivalent. We now present a few results.

**Proposition 2.6.** \( (\forall G_1, G_2 \in \phi(\Sigma)) (\forall G_3 \in \phi(\Sigma')) G_1 \subseteq G_2 \Rightarrow G_1 \times G_3 \subseteq G_2 \times G_3 \) \( \square \)

**Corollary 2.7.** \( (\forall G_1, G_2 \in \phi(\Sigma)) (\forall G_3 \in \phi(\Sigma')) G_1 \equiv G_2 \Rightarrow G_1 \times G_3 \equiv G_2 \times G_3 \). \( \square \)

Proof: Since \( G_1 \equiv G_2 \), by Def. 2.5 we have \( G_1 \subseteq G_2 \) and \( G_2 \subseteq G_1 \). Then by Prop. 2.6 we get \( G_1 \times G_3 \subseteq G_2 \times G_3 \) and \( G_2 \times G_3 \subseteq G_1 \times G_3 \), namely \( G_1 \times G_3 \equiv G_2 \times G_3 \). \( \square \)
Prop. 2.6 and Cor. 2.7 say nonblocking preserving and equivalence are invariant under automaton product.

**Proposition 2.8.** Given $G_i \in \phi(\Sigma_i)$ with $i = 1, 2$, let $\Sigma' \subseteq \Sigma_1 \cup \Sigma_2$. If $\Sigma_1 \cap \Sigma_2 \subseteq \Sigma'$, then

1. $(G_1 \times G_2)/ \approx_{\Sigma'} \subseteq (G_1/ \approx_{\Sigma_1 \cap \Sigma'}) \times (G_2/ \approx_{\Sigma_2 \cap \Sigma'})$.

2. If $G_i (i = 1, 2)$ is marking aware with respect to $\Sigma_i \cap \Sigma'$, then

$$(G_1 \times G_2)/ \approx_{\Sigma'} \equiv (G_1/ \approx_{\Sigma_1 \cap \Sigma'}) \times (G_2/ \approx_{\Sigma_2 \cap \Sigma'})$$

Proposition 2.8 is about the distribution of automaton abstraction over automaton product. As an illustration we present a simple example. Suppose we have $\Sigma_1 = \{\tau, a, \mu\}$ and $\Sigma_2 = \{\tau, b, c, \mu\}$. Let $G_1 \in \phi(\Sigma_1)$ and $G_2 \in \phi(\Sigma_2)$ be shown in Figure 3. Suppose we pick $\Sigma' = \{\tau, a, b, \mu\} \supseteq \Sigma_1 \cap \Sigma_2$. The results of $G_1 \times G_2$ and $(G_1 \times G_2)/ \approx_{\Sigma'}$ are depicted in Figure 4, and $(G_1/ \approx_{\Sigma_1 \cap \Sigma'}) \times (G_2/ \approx_{\Sigma_2 \cap \Sigma'})$ are in Figure 5. Clearly, $(G_1 \times G_2)/ \approx_{\Sigma'} \subseteq (G_1/ \approx_{\Sigma_1 \cap \Sigma'}) \times (G_2/ \approx_{\Sigma_2 \cap \Sigma'})$. But we can check that $(G_1/ \approx_{\Sigma_1 \cap \Sigma'}) \times (G_2/ \approx_{\Sigma_2 \cap \Sigma'}) \subseteq (G_1 \times G_2)/ \approx_{\Sigma'}$ does not hold. If we set $\Sigma' = \{\tau, a, c\}$, then $G_i (i = 1, 2)$ is marking aware with respect to $\Sigma_i \cap \Sigma'$. We can check that, indeed $(G_1 \times G_2)/ \approx_{\Sigma'} \equiv (G_1/ \approx_{\Sigma_1 \cap \Sigma'}) \times (G_2/ \approx_{\Sigma_2 \cap \Sigma'})$, as predicted by Prop. 2.8.
Figure 5: Example 3: $G_1/\approx_{\Sigma_1 \cap \Sigma'}$, $G_2/\approx_{\Sigma_2 \cap \Sigma'}$ and $(G_1/\approx_{\Sigma_1 \cap \Sigma'}) \times (G_2/\approx_{\Sigma_2 \cap \Sigma'})$

**Theorem 2.9.** Given $\Sigma$ and $\Sigma' \subseteq \Sigma$, let $G \in \phi(\Sigma)$ and $S \in \phi(\Sigma')$. Then

1. $B((G/\approx_{\Sigma'}) \times S) = \emptyset \Rightarrow B(G \times S) = \emptyset$
2. $G$ is marking aware w.r.t. $\Sigma'$ \[\Rightarrow \] $B((G/\approx_{\Sigma'}) \times S) = \emptyset \iff B(G \times S) = \emptyset$ □

**Proof:** Let $P : \Sigma^* \to \Sigma'^*$ be the natural projection.

$B((G/\approx_{\Sigma'}) \times S) = \emptyset$

$\iff B((G/\approx_{\Sigma'}) \times (S/\approx_{\Sigma'})) = \emptyset$ because $S/\approx_{\Sigma'} \subseteq S$ and by Cor. 2.7

$\Rightarrow B((G \times S)/\approx_{\Sigma'}) = \emptyset$ because by Prop. 2.8, $(G \times S)/\approx_{\Sigma'} \subseteq (G/\approx_{\Sigma'}) \times (S/\approx_{\Sigma'})$

$\Rightarrow P(B(G \times S)) = \emptyset$ by Prop. 2.4

$\iff B(G \times S) = \emptyset$

Thus, $B((G/\approx_{\Sigma'}) \times S) = \emptyset \Rightarrow B(G \times S) = \emptyset$.

Clearly, $S$ is marking aware with respect to $\Sigma'$ because $S \in \phi(\Sigma')$. If $G$ is also marking aware with respect to $\Sigma'$, then by Prop. 2.8, we have

$B((G \times S)/\approx_{\Sigma'}) = B((G/\approx_{\Sigma'}) \times (S/\approx_{\Sigma'}))$ \hspace{1cm} (1)

Furthermore, $G \times S$ is also marking aware with respect to $\Sigma'$ because both $G$ and $S$ are marking aware with respect to $\Sigma'$. By Prop. 2.4 we get that

$P(B(G \times S)) = B((G \times S)/\approx_{\Sigma'})$ \hspace{1cm} (2)

Thus we have

$B((G/\approx_{\Sigma'}) \times S) = \emptyset \iff B((G/\approx_{\Sigma'}) \times (S/\approx_{\Sigma'})) = \emptyset$

$\iff B((G \times S)/\approx_{\Sigma'}) = \emptyset$ by Equation 1

$\iff P(B(G \times S)) = \emptyset$ by Equation 2

$\iff B(G \times S) = \emptyset$

Thus, if $G$ is marking aware with respect to $\Sigma'$, then $B((G/\approx_{\Sigma'}) \times S) = \emptyset$ if and only if $B(G \times S) = \emptyset$. □

What Theorem 2.9 says can be interpreted informally as follows: if the abstraction of $G$ is ‘nonconflicting’ with $S$, i.e. $B((G/\approx_{\Sigma'}) \times S) = \emptyset$, then $G$ is ‘nonconflicting’ with $S$. But to make the opposite implication true, we need to impose the marking awareness condition. We will see how to use this result in supervisor synthesis shortly. But before that, we need to address some computational issue: if $G$ is very large, e.g. $G = G_1 \times \cdots \times G_n$ for some very large number $n \in \mathbb{N}$, how to compute...
G/ ≈_{\Sigma'}? To overcome this difficulty, we propose the following algorithm.

Suppose \( I = \{1, 2, \ldots, n\} \) for some \( n \in \mathbb{N} \). For any \( J \subseteq I \), let \( \Sigma_J := \cup_{j \in J} \Sigma_j \). Let \( \Sigma' \subseteq \cup_{i \in I} \Sigma_i \).

**Sequential Abstraction over Product: (SAP)**

1. Input of SAP: a collection of automata \( \{G_i | i \in I\} \).
2. For \( k = 1, 2, \ldots, n \), we perform the following computation.
   - Set \( J_k := \{1, 2, \ldots, k\} \), \( T_k := \Sigma_{J_k} \cap (\Sigma_{I - J_k} \cup \Sigma') \).
   - If \( k = 1 \) then \( W_1 := G_1/ \approx_{T_1} \)
   - If \( k > 1 \) then \( W_k := (W_{k-1} \times G_k)/ \approx_{T_k} \)
3. Output of SAP: \( W_n \).

To show that SAP fulfills our expectation, we first need some preparations.

**Proposition 2.10.** \((\forall \Sigma' \subseteq \Sigma)(\forall G_1, G_2 \in \phi(\Sigma)) G_1 \subsetneq G_2 \Rightarrow G_1/ \approx_{\Sigma'} \subsetneq G_2/ \approx_{\Sigma'} \).

**Corollary 2.11.** \((\forall \Sigma' \subseteq \Sigma)(\forall G_1, G_2 \in \phi(\Sigma)) G_1 \cong G_2 \Rightarrow G_1/ \approx_{\Sigma'} \cong G_2/ \approx_{\Sigma'} \).

**Proof:** Use Prop. 2.10 and Def. 2.5, the corollary follows.

Prop. 2.10 and Cor. 2.11 are about nonblocking preserving and equivalence being invariant under abstraction.

**Proposition 2.12.** \((\forall \Sigma'' \subseteq \Sigma')(\forall G \in \phi(\Sigma)) G/ \approx_{\Sigma''} \cong (G/ \approx_{\Sigma'})/ \approx_{\Sigma''} \).

**Proof:** We use induction to show that

\[
(\forall k : 1 \leq k \leq n)(\times_{j \in J_k} G_j)/ \approx_{T_k} \subseteq W_k
\]

It is clear that \( G_1/ \approx_{T_1} \subseteq W_1 \). Suppose Equation (3) is true for \( k \leq l \in \mathbb{N} \). Then we need to show that it also holds for \( k = l + 1 \). By the procedure,

\[
(\times_{j \in J_{l+1}} G_j)/ \approx_{T_{l+1}} \cong (\times_{j \in J_{l+1}} G_j)/ \approx_{T_{l+1}} \subseteq (\times_{j \in J_{l+1}} G_j)/ \approx_{T_{l+1}} \] by Prop. 2.12

\[
\subseteq (W_l \times G_{l+1})/ \approx_{T_{l+1}}
\]

because \( \Sigma_{l+1} \cap \Sigma_{J_l} \subseteq T_l \cup \Sigma_{l+1} \) and Prop. 2.8 and Prop. 2.10

\[
= W_{l+1}
\]
Therefore Equation (3) holds for all $k$, particularly $k = n$. The proposition follows.

SAP allows us to obtain an abstraction of the entire system $G = \times_{i \in I} G_i$ in a sequential way. Thus, we can avoid computing $G$ explicitly, which may be prohibitively large for systems of industrial size. Next, we discuss how to synthesize a supervisor based on automaton abstractions.

### 3 Supervisor Synthesis over Nondeterministic Finite-State Automata

As described in the introduction section, it has been a large volume of work on supervisor synthesis for nondeterministic systems. In this section we will mainly discuss how to apply automaton abstraction in such synthesis. We assume that the plant model $G \in \phi(\Sigma)$ is nondeterministic, but the requirement $H \in \phi(\Delta)$ with $\Delta \subseteq \Sigma$ is deterministic. Here $H$ need not necessarily be standardized. Our goal is to synthesize a deterministic supervisor $S \in \phi(\Sigma ')$ with $\Delta \subseteq \Sigma ' \subseteq \Sigma$. We first introduce some concepts.

Given $G = (X, \Sigma, \xi, x_0, X_m)$, for each $x \in X$ let $E_G : X \rightarrow 2^\Sigma : x \mapsto E_G(x) := \{\sigma \in \Sigma | \xi(x, \sigma) \neq \emptyset\}$

Thus, $E_G(x)$ is simply the set of all events allowable at $x$ in $G$. We now bring the concept of state controllability. Let $\Sigma = \Sigma_c \cup \Sigma_{uc}$, where $\Sigma_c$ is the set of controllable events, $\Sigma_{uc}$ is the set of uncontrollable events and $\tau \in \Sigma_{uc}$. Let $L(G) := \{s \in \Sigma^* | \xi(x_0, s) \neq \emptyset\}$.

**Definition 3.1.** Let $G = (X, \Sigma, \xi, x_0, X_m)$, $\Sigma ' \subseteq \Sigma$, $A = (Y, \Sigma ' , \eta, y_0, Y_m) \in \phi(\Sigma ')$ and $P : \Sigma^* \rightarrow \Sigma^*$ be the natural projection. $A$ is state-controllable with respect to $G$ and $\Sigma_{uc}$ if $(\forall s \in L(G \times A))(\forall x \in \xi(x_0, s))(\forall y \in \eta(y_0, P(s))) E_G(x) \cap \Sigma_{uc} \cap \Sigma ' \subseteq E_A(y)$

The concept of state controllability is slightly different from the one used in the literature, e.g. [23], because of the involvement of $\Sigma '$. We can check that, $A$ is state controllable implies that $L(G \times A)\Sigma_{uc} \cap L(G \times A) \subseteq L(G \times A)$. Thus, it is always true that state controllability implies language controllability described in the RW paradigm. But the reverse statement is not true unless both $A$ and $G$ are deterministic. We now introduce the concept of state observability. Let $\Sigma = \Sigma_o \cup \Sigma_{uo}$, where $\Sigma_o$ is the set of observable events and $\Sigma_{uo}$ is the set of unobservable events. Let $P_o : \Sigma^* \rightarrow \Sigma_o$ be the natural projection.

**Definition 3.2.** Let $G = (X, \Sigma, \xi, x_0, X_m) \in \phi(\Sigma)$, $\Sigma ' \subseteq \Sigma$ and $A = (Y, \Sigma ' , \eta, y_0, Y_m) \in \phi(\Sigma ' )$. $A$ is state-observable with respect to $G$ and $P_o$ if for any $s, s' \in L(G \times A)$ with $P_o(s) = P_o(s')$, we have $(\forall (x, y) \in \xi \times \eta((x_0, y_0), s))(\forall (x', y') \in \xi \times \eta((x_0, y_0), s')) E_G(x, y) \cap E_G(x') \cap \Sigma ' \subseteq E_A(y)$
We now introduce the concept of supervisors. State normality implies state observability. But the reverse statement is not true. A statement is not true unless both observable (or state-normal) supervisor of nonblocking state-controllable state-observable (or state-normal) supervisor

Proposition 3.5. Let \( G = (X, \Sigma, \xi, x_0, X_m) \in \phi(\Sigma) \) and \( \Sigma' \subseteq \Sigma \), \( A = (Y, \Sigma', \eta, y_0, Y_m) \in \phi(\Sigma') \) and \( P : \Sigma^* \rightarrow \Sigma^* \) be the natural projection. \( A \) is state-normal with respect to \( G \) and \( P_o \) if for any \( s \in L(G \times A) \) and \( s' \in P_o^{-1}(P_o(s)) \cap L(G \times A) \), we have

\[
(\forall (x, y) \in \xi \times \eta((x_0, y_0), s'))((\forall s'' \in \Sigma^*) P_o(s's'') = P_o(s) \Rightarrow [\xi(x, s'') \neq \emptyset \Rightarrow \eta(y, P(s''))) \neq \emptyset] \]

\( \square \)

We can check that, if \( A \) is state-normal with respect to \( G \) and \( P_o \), then

\[
L(G) \cap P_o^{-1}(P_o(L(G \times A))) \subseteq L(G \times A)
\]

which means \( L(G \times A) \) is language normal with respect to \( L(G) \) and \( P_o \). The reverse statement is not true unless both \( A \) and \( G \) are deterministic. Furthermore, we can check that state normality implies state observability. But the reverse statement is not true. We now introduce the concept of supervisors.

Definition 3.4. Given \( G = (X, \Sigma, \xi, x_0, X_m) \in \phi(\Sigma) \) and \( \Sigma' \subseteq \Sigma \), a deterministic finite-state automaton \( S = (Y, \Sigma', \eta, y_0, Y_m) \in \phi(\Sigma') \) is a nonblocking state-controllable state-observable (or state-normal) supervisor of \( G \) with respect to a specification \( H \in \phi(\Delta) \) with \( \Delta \subseteq \Sigma' \) under \( P_o \), if the following hold:

1. \( N(G \times S) \subseteq N(G \times H) \)
2. \( B(G \times S) = \emptyset \)
3. \( S \) is state-controllable with respect to \( G \) and \( \Sigma_{uc} \)
4. \( S \) is state-observable (or state-normal) with respect to \( G \) and \( P_o \)

\( \square \)

The first condition of Def. 3.4 says that the closed-loop behavior (CLB) satisfies the specification \( H \) and the second one says CLB must be nonblocking. The third and fourth ones are self-explanatory.

Proposition 3.5. Given \( G \in \phi(\Sigma) \) and \( H \in \phi(\Delta) \) with \( \Delta \subseteq \Sigma' \subseteq \Sigma \), there exists a nonblocking state-controllable state-observable (or state-normal) supervisor \( S \in \phi(\Sigma') \) of \( G \) with respect to \( H \) if and only if there exists an automaton \( A \in \phi(\Sigma') \) such that
1. \( N(G \times A) \subseteq N(G \times H) \)
2. \( B(G \times A) = \emptyset \)
3. \( A \) is state-controllable with respect to \( G \) and \( \Sigma_{uc} \)
4. \( A \) is state-observable (or state-normal) with respect to \( G \) and \( P_o \)

Prop. 3.5 is about the existence of a nonblocking state-controllable state-observable (or state-normal) supervisor, whose proof indicates that such a supervisor is simply a (canonical) recognizer of an automaton \( A \) such that those four conditions hold. In the language-based framework we know that controllability and normality are closed under language union. The following result shows that state controllability and state normality bear a similar feature.

**Proposition 3.6.** Given \( G \in \phi(\Sigma) \) and \( \Delta \subseteq \Sigma' \subseteq \Sigma \), let \( S_i \in \phi(\Sigma') \) (\( i = 1, 2 \)) be a nonblocking state-controllable state-normal supervisor of \( G \) w.r.t. \( H \in \phi(\Delta) \). Let \( S \in \phi(\Sigma') \) be a deterministic automaton such that \( N(S) = N(S_1) \cup N(S_2) \) and \( L(S) = N(S) \). Then \( S \) is a nonblocking state-controllable state-normal supervisor of \( G \) w.r.t. \( H \). □

Prop. 3.6 says that the ‘union’ of two nonblocking state-controllable state-normal (NSCSN) supervisors is still a NSCSN supervisor. We define a set

\[ \mathcal{CN}(G,H) := \{ S \in \phi(\Sigma') | S \text{ is a NSCSN supervisor of } G \text{ w.r.t. } H \land L(S) \subseteq L(G) \} \]

From Prop. 3.6 we can derive that \( \mathcal{CN}(G,H) \) has a unique element \( \hat{S} \) such that for any \( S \in \mathcal{CN}(G,H) \), we have \( N(S) \subseteq N(\hat{S}) \). We call \( \hat{S} \) the supremal nonblocking state-controllable state-normal supervisor of \( G \) with respect to \( H \). In reality we will be interested in such a supremal NSCSN supervisor, which can be computed from \( G \times H \). In this paper we will not discuss this issue. Instead, we will only focus on two questions: (1) under what conditions is a nonblocking supervisor for an abstract plant model \( G/ \approx_{\Sigma'} \) with \( \Sigma' \subseteq \Sigma \) also a nonblocking supervisor for the original plant model \( G \)? (2) under what conditions is a nonblocking supervisor \( S \in \phi(\Sigma') \) for \( G \) also a nonblocking supervisor for the abstract model \( G/ \approx_{\Sigma'} \)? To answer these questions, we need the following results.

**Lemma 3.7.** Given \( G \in \phi(\Sigma) \) and \( \Sigma' \subseteq \Sigma \), let \( S \in \phi(\Sigma') \). Then \( S \) is state-controllable with respect to \( G/ \approx_{\Sigma'} \) and \( \Sigma_{uc} \cap \Sigma' \) if and only if \( S \) is state-controllable with respect to \( G \) and \( \Sigma_{uc} \).

**Lemma 3.8.** Given \( G \in \phi(\Sigma) \) and \( \Sigma' \subseteq \Sigma \), let \( S \in \phi(\Sigma') \) and \( P'_o : \Sigma'^* \rightarrow (\Sigma' \cap \Sigma_o)^* \) be the natural projection. Then (1) If \( S \) is state-observable w.r.t. \( G/ \approx_{\Sigma'} \) and \( P'_o \), then \( S \) is state-observable w.r.t. \( G \) and \( P_o \). (2) If \( \Sigma_o \subseteq \Sigma' \) and \( S \) is state-observable w.r.t. \( G \) and \( P_o \), then \( S \) is state-observable w.r.t. \( G/ \approx_{\Sigma'} \) and \( P'_o \).

**Lemma 3.9.** Given \( G \in \phi(\Sigma) \) and \( \Sigma' \subseteq \Sigma \), let \( S \in \phi(\Sigma') \) and \( P'_o : \Sigma'^* \rightarrow (\Sigma' \cap \Sigma_o)^* \) be the natural projection. Then (1) If \( S \) is state-normal w.r.t. \( G/ \approx_{\Sigma'} \) and \( P'_o \), then \( S \) is state-normal w.r.t. \( G \) and \( P_o \). (2) If \( \Sigma_o \subseteq \Sigma' \) and \( S \) is state-normal w.r.t. \( G \) and \( P_o \), then \( S \) is state-normal w.r.t. \( G/ \approx_{\Sigma'} \) and \( P'_o \). □
**Theorem 3.10.** Given $G \in \phi(\Sigma)$ and a deterministic automaton $H \in \phi(\Delta)$ with $\Delta \subseteq \Sigma' \subseteq \Sigma$, if there exists a nonblocking state-controllable state-observable (or state-normal) supervisor $S \in \phi(\Sigma')$ for $G/\approx_{\Sigma'}$ with respect to $H$, then $S$ is also a nonblocking state-controllable state-observable (or state-normal) supervisor for $G$ with respect to $H$. □

Proof: Since $S$ is a nonblocking state-controllable state-observable (or state-normal) supervisor of $G/\approx_{\Sigma'}$ with respect to $H$, by Def. 3.4,

1. $N((G/\approx_{\Sigma'}) \times S) \subseteq N((G/\approx_{\Sigma'}) \times H)$
2. $B((G/\approx_{\Sigma'}) \times S) = \emptyset$
3. $S$ is state-controllable w.r.t. $G/\approx_{\Sigma'}$ and $\Sigma_{uc} \cap \Sigma'$
4. $S$ is state-observable (or state-normal) w.r.t. $G/\approx_{\Sigma'}$ and $P_o : \Sigma'^* \rightarrow (\Sigma_{\text{o}} \cap \Sigma')^*$

By Lemma 3.7, $S$ is state-controllable with respect to $G$ and $\Sigma_{uc}$. By Lemma 3.8, $S$ is state observable with respect to $G$ and $P_o$, or by Lemma 3.9, $S$ is state-normal with respect to $G$ and $P_o$. Since

$$B((G/\approx_{\Sigma'}) \times S) = \emptyset$$

By Theorem 2.9 we get that $B(G \times S) = \emptyset$. Finally, we show that $N(G \times S) \subseteq N(G \times H)$ as follows.

$$N((G/\approx_{\Sigma'}) \times S) \subseteq N((G/\approx_{\Sigma'}) \times H) \text{ by (1)}$$
$$\Rightarrow N(G/\approx_{\Sigma'}) || N(S) \subseteq N(G/\approx_{\Sigma'}) || N(H)$$
$$\Rightarrow N(G)||N(G/\approx_{\Sigma'}) || N(S) \subseteq N(G)||N(G/\approx_{\Sigma'}) || N(H)$$
$$\Rightarrow N(G)||P(N(G)) || N(S) \subseteq N(G)||P(N(G)) || N(H) \text{ by Prop. 2.4}$$
$$\Rightarrow N(G)||N(S) \subseteq N(G)||N(H) \text{ because } N(G) = N(G)||P(N(G))$$
$$\Rightarrow N(G \times S) \subseteq N(G \times H)$$

Therefore, the theorem is true. ■

Theorem 3.10 says that a nonblocking supervisor $S$ for $G/\approx_{\Sigma'}$ is also a nonblocking supervisor for $G$.

**Theorem 3.11.** Given $G \in \phi(\Sigma)$ and a deterministic automaton $H \in \phi(\Delta)$ with $\Delta \subseteq \Sigma' \subseteq \Sigma$, suppose $G$ is marking aware with respect to $\Sigma'$ and $\Sigma_{\text{o}} \subseteq \Sigma'$. Then a nonblocking state-controllable state-observable (or state-normal) supervisor $S \in \phi(\Sigma')$ for $G$ with respect to $H$ and $\Sigma_{\text{o}} \subseteq \Sigma'$ is also a nonblocking state-controllable state-observable (or state-normal) supervisor for $G/\approx_{\Sigma'}$ with respect to $H$. □

Proof: Since $S$ is a nonblocking state-controllable state-observable (or state-normal) supervisor of $G$ with respect to $H$, by Def. 3.4,

1. $N(G \times S) \subseteq N((G/\approx_{\Sigma'}) \times H)$
2. $B(G \times S) = \emptyset$
3. $S$ is state-controllable with respect to $G$ and $\Sigma_{uc}$
4. $S$ is state-observable (or state-normal) with respect to $G$ and $P_o$
By Lemma 3.7, $S$ is state-controllable with respect to $G/\approx_{\Sigma'}$ and $\Sigma_{uc} \cap \Sigma'$. By Lemma 3.8, $S$ is state-observable with respect to $G/\approx_{\Sigma'}$ and $P'_o$, or by Lemma 3.9, $S$ is state-normal with respect to $G/\approx_{\Sigma'}$ and $P'_o$. Since $B(G \times S) = \emptyset$ and $G$ is marking aware with respect to $\Sigma'$, by Theorem 2.9 we get that $B((G/\approx_{\Sigma'}) \times S) = \emptyset$. Finally, we show that $N((G/\approx_{\Sigma'}) \times S) \subseteq N((G/\approx_{\Sigma'}) \times H)$ as follows.

$$N((G/\approx_{\Sigma'}) \times S) = N((G/\approx_{\Sigma'}) \times (S/\approx_{\Sigma'}))$$

because $S/\approx_{\Sigma'} \approx S$ and by Cor. 2.7

$$= N((G \times S)/\approx_{\Sigma'})$$

by Prop. 2.8

$$= P(N(G \times S))$$

by Prop. 2.4

$$\subseteq P(N(G \times H))$$

By (1)

$$= N((G \times H)/\approx_{\Sigma'})$$

by Prop. 2.4

$$= N((G/\approx_{\Sigma'}) \times (H/\approx_{\Sigma'}))$$

by Prop. 2.8

$$= N((G/\approx_{\Sigma'}) \times H)$$

because $H/\approx_{\Sigma'} \approx H$ and by Cor. 2.7

Therefore, the theorem is true.

Theorem 3.11 says that, if $G$ is marking aware with respect to $\Sigma'$ and $\Sigma_o \subseteq \Sigma'$, then a nonblocking supervisor of $G$ is also a nonblocking supervisor of $G/\approx_{\Sigma'}$.

4 Example

As an illustration we present the following example. Suppose we have two machines, which are functionally identical, except for individual event labels. The system is depicted in Figure 6. Each machine $G_i$ ($i = 1, 2$) has the following standard operations: (1)

$$\text{fetching a work piece} \ (a_i); \ (2) \ \text{preprocessing} \ (b_i); \ (3) \ \text{postprocessing} \ (c_i); \ (4) \ \text{polishing} \ (e_i); \ (5) \ \text{packaging} \ (d_i).$$

After preprocessing $b_i$, there are two choices: to be postprocessed directly ($c_i$) or to be polished first ($e_i$) before postprocessing. The latter gives a product with better quality. The negative aspect is that polishing may cause the machine $G_1$ to fail ($f_1$). If failure does happen, $G_1$ will stop automatically and wait for repair. Among

Figure 6: Example 4: A Simple Processing Unit
each alphabet Σ_i, the controllable alphabet is Σ_{i,c} = \{a_i, e_i\}, and the observable alphabet Σ_{i,o} = Σ_i, namely every event is observable (for the purpose of simplicity). There is one specification H ∈ φ(∆) with ∆ = \{e_1, e_2\}, depicted in Figure 7, saying that if a work piece is polished in G_1 (e_1), then a work piece must be polished in G_2 afterwards (e_2). We now start to synthesize a deterministic nonblocking state-controllable state-normal supervisor that enforces the specification H.

First, we create an appropriate abstraction of G_1 × G_2. We pick Σ' = \{τ, a_1, a_2, e_1, e_2\}. The motivation is that, since ∆ ⊆ Σ', the abstraction (G_1 × G_2)/≈_{Σ'} can capture the specification H; and since all controllable events are in Σ', the abstraction (G_1 × G_2)/≈_{Σ'} also contains all means of control available to G_1 × G_2 itself. Since Σ_1 ∩ Σ_2 = \{τ\} ⊆ Σ', by Prop. 2.8, (G_1 × G_2)/≈_{Σ'} ⊑ (G_1/≈_{Σ_1 ∩ Σ'}) × (G_2/≈_{Σ_2 ∩ Σ'}).

The results of G_1/≈_{Σ_1 ∩ Σ'} and G_2/≈_{Σ_2 ∩ Σ'} are depicted in Figure 8. The product of two abstractions G' := (G_1/≈_{Σ_1 ∩ Σ'}) × (G_2/≈_{Σ_2 ∩ Σ'}) is depicted in Figure 9, We now use G' and H to synthesize a supervisor. The product G' × H is depicted in Figure 9. Clearly, the transitions e_1 between states (2, 0) and (3, 1), and between states (5, 0) and (4, 1) in G' × H must be disabled. Otherwise, blocking states (3, 1) and (4, 1) will be reached. Once these two transitions are disabled, transitions e_1 between states (2, 0) and (1, 1), and between states (5, 0) and (6, 1) must be disabled as well because, otherwise, the remaining automaton is not state-normal (and state-observable). After removing transitions e_1 at states (2, 0) and (5, 0) in Figure 9, the remaining reachable part A is depicted in Figure 10, which is nonblocking, state-controllable, state-normal (and state-observable). By Prop. 3.5 we get that, the canonical recognizer S of the marked behavior N(A), depicted in Figure 11, is a nonblocking state-controllable and state-normal supervisor of G' with respect to H. We can see that S does not allow events e_1 and e_2 to happen. It is not difficult to check that S is a nonblocking state-controllable state-normal supervisor of G_1 × G_2 with respect to the specification H, as predicted by Theorem 3.10. We can...
verify that the maximum number of states of any intermediate automata is 13, which occurs when we compute $G' \times H$. Clearly, abstractions help to reduce the computational complexity in this example because otherwise we will have to face the product $G_1 \times G_2 \times H$ directly, which has 61 states.

5 Conclusions

In this paper we first present a new abstraction technique and provide some properties. Then we apply this technique in supervisor synthesis. We consider the problem of synthesizing a deterministic supervisor for a nondeterministic plant model and a deterministic specification. After introducing the concepts of state controllability, state observability and state normality, we show that a nonblocking state-controllable state-observable (or state-normal) supervisor of an abstraction $G/ \approx_{\Sigma'}$ under a specification $H$ is also a nonblocking state-controllable state-observable (or state-normal) supervisor of the original
we have that
\[ G \times / G \]
Finally, suppose
\[ t \in t \]
which means, there exists
\[ T \]
To show that
\[ P \]
Thus,
\[ B \approx G/ \]
Because
\[ \xi \]
1. Proof of Prop. 2.4: Let \( \xi' \) be the transition map of \( G/ \approx_{\Sigma'} \). First we show that \( P(B(G)) \subseteq B(G/ \approx_{\Sigma'}) \). For each string \( s \in P(B(G)) \), there exists \( t \in B(G) \) with \( P(t) = s \) such that
\[ (\exists x \in \xi(x_0, t))(\forall t' \in \Sigma^*) \xi(x, t') \cap X_m = \emptyset \]
Since \( G \) is standardized, \( P(t) \neq \epsilon \) iff \( t \neq \epsilon \). Thus, we get that \( < x ,P(t) \rangle \). Because
\[ (\forall t' \in \Sigma^*) \xi(x, t') \cap X_m = \emptyset \]
we have that
\[ (\forall s' \in \Sigma^*) \xi'( < x ,s' ) \cap (X_m/ \approx_{\Sigma'}) = \emptyset \]
Thus, \( s = P(t) \in B(G/ \approx_{\Sigma'}) \).
To show that \( P(N(G)) \subseteq N(G/ \approx_{\Sigma'}) \), let \( s \in P(N(G)) \). Then
\[ (\exists t \in N(G)) P(t) = s \land \xi(x_0, t) \cap X_m \neq \emptyset \]
Since \( G \) is standardized, \( \xi(x_0, t) \cap X_m \neq \emptyset \) implies \( \xi'( < x_0 ,P(t) \rangle \) \( \approx_{\Sigma'} \). Thus, \( s \in N(G/ \approx_{\Sigma'}) \). To show \( N(G/ \approx_{\Sigma'}) \subseteq P(N(G)) \), let \( s \in N(G/ \approx_{\Sigma'}) \). Then we have
\[ \xi'( < x_0 ,s ) \cap (X_m/ \approx_{\Sigma'}) \neq \emptyset \]
which means, there exists \( t \in \Sigma^* \) with \( P(t) = s \) such that \( \xi(x_0, t) \cap X_m \neq \emptyset \). Thus, \( t \in N(G) \), namely \( P(t) = s \in P(N(G)) \). Therefore, we have \( P(N(G)) = N(G/ \approx_{\Sigma'}) \).
Finally, suppose \( G \) is marking aware with respect to \( \Sigma' \). To show \( B(G/ \approx_{\Sigma'}) = P(B(G)) \), we only need to show that \( B(G/ \approx_{\Sigma'}) \subseteq P(B(G)) \). For each string \( s \in B(G/ \approx_{\Sigma'}) \), we have
\[ (\exists x \in \xi'( < x_0 ,s ) \forall s' \in \Sigma^* ) \xi'( < x ,s' ) \cap (X_m/ \approx_{\Sigma'}) = \emptyset \]
from which we can derive that, there exists \( t \in \Sigma^* \) such that \( P(t) = s \) and
\[ x \in \xi(x_0, t) \land (\forall t' \in \Sigma^*) \xi(x, t') \cap X_m \neq \emptyset \Rightarrow t' \in (\Sigma - \Sigma')^* \]
Clearly, \( x \notin X_m \), because otherwise \( \xi'( < x ,, \epsilon ) \) \( \approx_{\Sigma'} \) \( \neq \emptyset \). We claim that \( x \) is a blocking state of \( G \). Otherwise, there exists \( t' \in \Sigma^* \) such that \( \xi(x, t') \cap X_m \neq \emptyset \). Since \( G \) is marking aware with respect to \( \Sigma' \), we have that \( t' \notin (\Sigma - \Sigma')^* \), which contradicts the fact that
\[ (\forall t' \in \Sigma^*) \xi(x, t') \cap X_m \neq \emptyset \Rightarrow t' \in (\Sigma - \Sigma')^* \]
Let's now show that we claim that $(2)$. If $t \in (\Sigma_1 \cup \Sigma_2)^*$ is a blocking state of $(G_1 \times G_2)/\approx_{\Sigma'}$ and $\xi'$ be the transition map of $G_1/\approx_{\Sigma_i}$ $(i = 1, 2)$. First, we have the following,

$$N((G_1 \times G_2)/\approx_{\Sigma'}) = P(N(G_1 \times G_2)) \text{ by Prop. 2.4}$$

$$= P(N(G_1)||N(G_2))$$

$$= P_1(N(G_1)||P_2(N(G_2))) \text{ because } \Sigma_1 \cup \Sigma_2 \subseteq \Sigma'$$

$$= N(G_1/\approx_{\Sigma_1})||N(G_2/\approx_{\Sigma_2}) \text{ by Prop. 2.4}$$

$$= N((G_1/\approx_{\Sigma_1}) \times (G_2/\approx_{\Sigma_2}))$$

Next, we show that $B((G_1 \times G_2)/\approx_{\Sigma'}) \subseteq B((G_1/\approx_{\Sigma_1}) \times (G_2/\approx_{\Sigma_2}))$. Let $s \in B((G_1 \times G_2)/\approx_{\Sigma'})$. Then there exists $(x_1, x_2) \in X_1 \times X_2$ such that

$$<(x_1, x_2) >_{\Sigma'} \in \xi'((< (x_1,0,x_2,0), s) \wedge \forall s' \in \Sigma'') \xi'(< (x_1, x_2) >_{\Sigma'}, s') \cap (X_{1,m} \times X_{2,m})/\approx_{\Sigma'} = \emptyset$$

which means $(x_1, x_2) \notin X_{1,m} \times X_{2,m}$ and there exists $t \in (\Sigma_1 \cup \Sigma_2)^*$ with $P(t) = s$ such that

$$(x_1, x_2) \in \xi_1 \times \xi_2((x_1,0,x_2,0), t) \wedge (\forall t' \in \Sigma^*) \xi_1 \times \xi_2((x_1, x_2), t') \cap (X_{1,m} \times X_{2,m}) \neq \emptyset \Rightarrow t' \in (\Sigma_1 \cup \Sigma_2)^*$$

Since $G_1$ and $G_2$ are standardized, from $(x_1, x_2) \in \xi_1 \times \xi_2((x_1,0,x_2,0), t)$ and the fact that $\Sigma_1 \cap \Sigma_2 \subseteq \Sigma'$ we can derive that

$$< (x_1, x_2) >_{\Sigma_1}, < (x_2, x_2) >_{\Sigma_2} \in \xi_1 \times \xi_2((< x_1,0,x_2,0 >_{\Sigma_1}, < x_2,0 >_{\Sigma_2}, s)$$

We claim that $< (x_1, x_2) >_{\Sigma_1}, < (x_2, x_2) >_{\Sigma_2}$ is a blocking state of $(G_1/\approx_{\Sigma_1}) \times (G_2/\approx_{\Sigma_2})$. Otherwise, there exists $s' \in \Sigma''$ such that

$$\xi'_1 \times \xi'_2((< x_1 >_{\Sigma_1}, < x_2 >_{\Sigma_2}), s') \cap (X_{1,m} \times X_{2,m}/\approx_{\Sigma_1} \times (X_{2,m} \times X_{2,m}/\approx_{\Sigma_2}) \neq \emptyset$$

Since $(x_1, x_2) \notin X_{1,m} \times X_{2,m}$, $(< x_1 >_{\Sigma_1}, < x_2 >_{\Sigma_2}) \notin (X_{1,m} \times (X_{2,m}/\approx_{\Sigma_1}) \times (X_{2,m}/\approx_{\Sigma_2})$.

Thus, $s' \neq \epsilon$, which means there exists $t' \in \Sigma^*$ with $P(t') = s' \notin (\Sigma_1 \cup \Sigma_2) = \Sigma'^*$ such that $\xi_1 \times \xi_2((x_1, x_2), t') \cap (X_{1,m} \times X_{2,m}) \neq \emptyset$ - contradict the fact that

$$\forall t' \in \Sigma^* \xi_1 \times \xi_2((x_1, x_2), t') \cap (X_{1,m} \times X_{2,m}) \neq \emptyset \Rightarrow t' \in (\Sigma_1 \cup \Sigma_2)^*$$

From the claim we get that $s \in B((G_1/\approx_{\Sigma_1}) \times (G_2/\approx_{\Sigma_2}))$. Let $s \in N((G_1 \times G_2)/\approx_{\Sigma'})$. For any $(x_1, x_2) \in X_1 \times X_2$ with

$$<(x_1, x_2) >_{\Sigma'} \in \xi'(< (x_1,0,x_2,0) >_{\Sigma'}, s)$$

we have

$$\exists t' \in \Sigma^* \xi(t') = s \wedge (x_1, x_2) \in \xi'((x_1,0,x_2,0), t)$$

Since $G_1$ and $G_2$ are standardized, if $s = \epsilon$, then $t = \epsilon$, which means $(x_1, x_2) = (x_1,0,x_2,0)$. Clearly, we have the following expression:

$$<(x_1,0,x_2,0) >_{\Sigma_1} \times < (x_1,0,x_2,0) >_{\Sigma_2} \in \xi'_1 \times \xi'_2((< x_1,0,x_2,0 >_{\Sigma_1}, < x_2,0 >_{\Sigma_2}), \epsilon)$$

If $s \neq \epsilon$, then by the definition of automaton abstraction and the assumption that $\Sigma_1 \cap \Sigma_2 \subseteq \Sigma'$, we get

$$<(x_1,0,x_2,0) >_{\Sigma_1} \times < (x_2,0 >_{\Sigma_2}) \in \xi'_1 \times \xi'_2((< x_1,0,x_2,0 >_{\Sigma_1}, < x_2,0 >_{\Sigma_2}), s)$$

Thus, in either case we have

$$<(x_1,0,x_2,0) >_{\Sigma_1} \times < (x_2,0 >_{\Sigma_2}) \in \xi'_1 \times \xi'_2((< x_1,0,x_2,0 >_{\Sigma_1}, < x_2,0 >_{\Sigma_2}), s)$$

We now show that

$$N(G_1/\approx_{\Sigma_1}) \times (G_2/\approx_{\Sigma_2})<(x_1,0,x_2,0) >_{\Sigma_1} \times < (x_2,0 >_{\Sigma_2}) \subseteq N((G_1 \times G_2)/\approx_{\Sigma'})<(x_1, x_2) >_{\Sigma'}$$
Let \( s' \in N_{G_1/\approx_{\Sigma_1}} \times (G_2/\approx_{\Sigma_2}) (\langle x_1 >_{\Sigma_1}, < x_2 >_{\Sigma_2}) \). If \( s' = \epsilon \), then
\[
(\langle x_1 >_{\Sigma_1}, < x_2 >_{\Sigma_2}) \in (X_{1,m}/\approx_{\Sigma_1}) \times (X_{2,m}/\approx_{\Sigma_2})
\]
from which we get \((x_1, x_2) \in X_{1,m} \times X_{2,m}\). Thus, \( (x_1, x_2) >_{\Sigma'} (X_{1,m} \times X_{2,m}) / \approx_{\Sigma'} \), namely \( \epsilon \in N_{G_1 \times G_2}(\approx_{\Sigma'}) (\langle x_1, x_2 >_{\Sigma'}) \). If \( s' \neq \epsilon \), then there exists \( t' \in \Sigma^* \) with \( P(t') = s' \) such that \( \xi_1 \times \xi_2((x_1, x_2), t') \cap (X_{1,m} \times X_{2,m}) \neq \emptyset \). Since \( P(t') \neq \epsilon \), by the definition of abstraction, we get \( \xi'(\langle x_1, x_2 >_{\Sigma'}, s') \cap (X_{1,m} \times X_{2,m}) / \approx_{\Sigma'} \neq \emptyset \). Thus, \( s' \in N_{G_1 \times G_2}(\approx_{\Sigma'}) (\langle x_1, x_2 >_{\Sigma'}) \). In either case, we have
\[
N_{G_1 \times G_2}(\approx_{\Sigma'}) \cup (G_2/\approx_{\Sigma_2}) (\langle x_1 >_{\Sigma_1}, < x_2 >_{\Sigma_2}) \subseteq N_{G_1 \times G_2}(\approx_{\Sigma'}) (\langle x_1, x_2 >_{\Sigma'})
\]
Thus, \((G_1 \times G_2) / \approx_{\Sigma'} \subseteq (G_1 / \approx_{\Sigma_1}) \times (G_2 / \approx_{\Sigma_2})\).

Suppose \( G_i (i = 1, 2) \) is marking aware with respect to \( \Sigma_i \cap \Sigma' \). To show
\[
B((G_1 / \approx_{\Sigma_1}) \times (G_2 / \approx_{\Sigma_2})) = B((G_1 \times G_2) / \approx_{\Sigma'})
\]
we only need to prove one direction (\( \subseteq \)), because the other direction (\( \supseteq \)) has been proved above. Let \( s \in B((G_1 / \approx_{\Sigma_1}) \times (G_2 / \approx_{\Sigma_2})) \). Then there exists \( (x_1, x_2) \in X_1 \times X_2 \) such that
\[
(\langle x_1 >_{\Sigma_1}, < x_2 >_{\Sigma_2}) \in \xi_1' \times \xi_2' (\langle x_1, x_2 >_{\Sigma'}, s) \quad (4)
\]
and
\[
(\forall s' \in \Sigma'^*) \xi_1' \times \xi_2' ((\langle x_1 >_{\Sigma_1}, < x_2 >_{\Sigma_2}), s') \cap ((X_{1,m}/\approx_{\Sigma_1}) \times (X_{2,m}/\approx_{\Sigma_2})) = \emptyset \quad (5)
\]
From Expression (4) we get that
\[
(\exists t \in (\Sigma_1 \cup \Sigma_2)^*) P(t) = s \land (x_1, x_2) \in \xi_1 \times \xi_2((x_{1,0}, x_{2,0}), t) \quad (6)
\]
From Expression (5) we get that \((x_1, x_2) \notin X_{1,m} \times X_{2,m}\). Since \( G_1 \) and \( G_2 \) are standardize, from Expression (6) and the fact that \( \Sigma_1 \cap \Sigma_2 \subseteq \Sigma' \) we have
\[
(\langle x_1, x_2 >_{\Sigma'}) \notin \xi'(\langle x_1, x_2 >_{\Sigma'}, s)
\]
We claim that \( (x_1, x_2) >_{\Sigma'} \) is a blocking state of \((G_1 \times G_2) / \approx_{\Sigma'}\). Otherwise, there exists \( s' \in \Sigma'^* \) such that
\[
\xi'(\langle x_1, x_2 >_{\Sigma'}, s') \cap (X_{1,m} \times X_{2,m}) / \approx_{\Sigma'} \neq \emptyset
\]
Since \((x_1, x_2) \notin X_{1,m} \times X_{2,m}\), we get that \( (x_1, x_2) >_{\Sigma'} \notin (X_{1,m} \times X_{2,m}) / \approx_{\Sigma'}\). Thus, \( s' \neq \epsilon \). Furthermore, since \( G_i (i = 1, 2) \) is marking aware with respect to \( \Sigma_i \cap \Sigma' \), we have \( \hat{P}_i(s') \neq \epsilon \). Thus, there exists \( t' \in \Sigma^* \) with \( P(t') = s' \) such that \( \xi_1 \times \xi_2((x_1, x_2), t') \cap (X_{1,m} \times X_{2,m}) \neq \emptyset \). Since \( \hat{P}_i(s') \neq \epsilon \) for \( i = 1, 2 \) and \( \Sigma_1 \cap \Sigma_2 \subseteq \Sigma' \), we have
\[
\xi_1' \times \xi_2' ((\langle x_1 >_{\Sigma_1}, < x_2 >_{\Sigma_2}), s') \cap ((X_{1,m}/\approx_{\Sigma_1}) \times (X_{2,m}/\approx_{\Sigma_2})) \neq \emptyset
\]
which contradicts Expression (5). Thus, the claim is true, namely \( s \in B((G_1 \times G_2) / \approx_{\Sigma'})\).

Let \( s \in N((G_1 / \approx_{\Sigma_1}) \times (G_2 / \approx_{\Sigma_2})) \). For any \((x_1, x_2) \in X_1 \times X_2\) with
\[
(\langle x_1 >_{\Sigma_1}, < x_2 >_{\Sigma_2}) \in \xi_1' \times \xi_2' ((\langle x_1, x_2 >_{\Sigma'}, s), s)
\]
we have
\[
(\exists t \in \Sigma'^*) P(t) = s \land (x_1, x_2) \in \xi((x_{1,0}, x_{2,0}), t)
\]
Since \( G_1 \) and \( G_2 \) are standardize, if \( s = \epsilon \), then \( t = \epsilon \), which means \((x_1, x_2) = (x_{1,0}, x_{2,0})\).

Clearly, we have the following expression:
\[
(\langle x_{1,0}, x_{2,0} >_{\Sigma'}) \notin \xi'(\langle x_{1,0}, x_{2,0} >_{\Sigma'}, \epsilon)
\]
If \( s \neq \epsilon \), then by the definition of automaton abstraction and the assumption that \( \Sigma_1 \cap \Sigma_2 \subseteq \Sigma' \), we get
\[
(\langle x_1, x_2 >_{\Sigma'}) \notin \xi'(\langle x_{1,0}, x_{2,0} >_{\Sigma'}, s)
\]
Thus, in either case we have
\[
(\langle x_1, x_2 >_{\Sigma'}) \notin \xi'(\langle x_{1,0}, x_{2,0} >_{\Sigma'}, s)
\]
We now show that
\[ N_{(G_1 \times G_2) / \tilde{\Sigma}'_x}(<x_1, x_2 > \tilde{\Sigma}) \subseteq N_{(G_1 / \approx_{\xi_1}) \times (G_2 / \approx_{\xi_2})}(<x_1 > \tilde{\Sigma}_1, <x_2 > \tilde{\Sigma}_2) \]
Let \( s' \in N_{(G_1 \times G_2) / \tilde{\Sigma}'_x}(<x_1, x_2 > \tilde{\Sigma}) \). If \( s' = \epsilon \), then
\[ <x_1, x_2 > \tilde{\Sigma} \in (X_1, m \times X_2, m) / \approx \tilde{\Sigma}'_x \]
from which we can derive that \( (x_1, x_2) \in X_1, m \times X_2, m \). Thus,
\[ <x_1 > \tilde{\Sigma}_1, <x_2 > \tilde{\Sigma}_2 \in (X_1, m / \approx \tilde{\Sigma}_1) \times (X_2, m / \approx \tilde{\Sigma}_2) \]
which means \( s' \in N_{(G_1 / \approx_{\xi_1}) \times (G_2 / \approx_{\xi_2})}(<x_1 > \tilde{\Sigma}_1, <x_2 > \tilde{\Sigma}_2) \). If \( s' \neq \epsilon \), then
\[ <x_1, x_2 > \tilde{\Sigma} \notin (X_1, m \times X_2, m) / \approx \tilde{\Sigma}'_x \]
which means \( s' \notin (X_1, m \times X_2, m) / \approx \tilde{\Sigma}'_x \). Furthermore, there exists \( t' \in \Sigma^* \) with \( P(t') = s' \) such that \( \xi_1 \times \xi_2((x_1, x_2), t') \cap (X_1, m \times X_2, m) \neq \emptyset \). We consider three cases. Case 1: \( \hat{P}_1(s') \neq \epsilon \) \((i = 1, 2)\), namely \( x_1 \notin X_1, m \) and \( x_2 \notin X_2, m \). By the definition of automaton abstraction, we get that
\[ \xi'_1 \times \xi'_2((<x_1 > \tilde{\Sigma}_1, <x_2 > \tilde{\Sigma}_2), s') \cap (X_1, m / \approx \tilde{\Sigma}_1) \times (X_2, m / \approx \tilde{\Sigma}_2) \neq \emptyset \]
Thus, \( s' \in N_{(G_1 / \approx_{\xi_1}) \times (G_2 / \approx_{\xi_2})}(<x_1 > \tilde{\Sigma}_1, <x_2 > \tilde{\Sigma}_2) \). Case 2: \( \hat{P}_1(s') = \epsilon \) and \( \hat{P}_2(s') = s' \neq \epsilon \). Since \( G_1 \) is marking aware with respect to \( \Sigma_1 \cap \Sigma' \), \( \hat{P}_1(s') = \epsilon \) implies that \( x_1 \in X_1, m \). Since \( \hat{P}_2(s') = s' \neq \epsilon \), we have
\[ (\exists <\hat{x}_2 > \tilde{\Sigma}_2 <\hat{x}_2 > \tilde{\Sigma}_2) \subseteq (\hat{P}_2(s') \neq \tilde{\Sigma'}) \]
Thus,
\[ <x_1 > \tilde{\Sigma}_1, <\hat{x}_2 > \tilde{\Sigma}_2 \in \xi'_1 \times \xi'_2((<x_1 > \tilde{\Sigma}_1, <x_2 > \tilde{\Sigma}_2), s') \]
which means \( s' \in N_{(G_1 / \approx_{\xi_1}) \times (G_2 / \approx_{\xi_2})}(<x_1 > \tilde{\Sigma}_1, <x_2 > \tilde{\Sigma}_2) \). Case 3: \( \hat{P}_1(s') \neq \epsilon \) and \( \hat{P}_2(s') = \epsilon \). This case is similar to Case 2. In either case, we have
\[ N_{(G_1 \times G_2) / \tilde{\Sigma}'_x}(<x_1, x_2 > \tilde{\Sigma}) \subseteq N_{(G_1 / \approx_{\xi_1}) \times (G_2 / \approx_{\xi_2})}(<x_1 > \tilde{\Sigma}_1, <x_2 > \tilde{\Sigma}_2) \]
Thus, \( (G_1 / \approx_{\xi_1}) \times (G_2 / \approx_{\xi_2}) \subseteq (G_1 \times G_2) / \approx \tilde{\Sigma}'_x \).

3. Proof of Prop. 2.6: Let \( G_i = (X_i, \Sigma_i, \xi_i, X_i, m_i) \) with \( i = 1, 2, 3 \), where \( \Sigma_1 = \Sigma_2 = \Sigma \) and \( \Sigma_3 = \Sigma' \). Let \( P : (\Sigma \cup \Sigma')^* \rightarrow \Sigma^* \) and \( P' : (\Sigma \cup \Sigma')^* \rightarrow \Sigma'^* \) be natural projections. We first show that \( N(G_1 \times G_3) = N(G_2 \times G_3) \). Clearly, we have \( N(G_1 \times G_3) = N(G_1)||N(G_3) \). Since \( G_1 \subseteq G_2 \), we have \( N(G_1) = N(G_2) \). Thus, we have
\[ N(G_1 \times G_3) = N(G_1)||N(G_3) = N(G_2)||N(G_3) = N(G_2 \times G_3) \]
To show that \( B(G_1 \times G_3) \subseteq B(G_2 \times G_3) \), let \( s \in B(G_1 \times G_3) \). By the definition of automaton product, there exists \( x_1 \in X_1 \) such that \( x_1 \in \xi_1(x_1, m_i, P(s)) \). There are two cases to consider. Case 1: \( x_1 \) is a blocking state. Then \( P(s) \in B(G_1) \subseteq B(G_2) \). Thus, \( s \in B(G_2 \times G_3) \). Case 2: \( x_1 \) is a nonblocking state. Since \( G_1 \subseteq G_2 \), there exists \( x_2 \in \xi_2(x_2, m_2, P(s)) \) such that \( N_{G_2}(x_1) \subseteq N_{G_2}(x_2) \). Since \( s \in B(G_1 \times G_3) \), there exists \( x_3 \in X_3 \) such that \( (x_1, x_3) \in \xi_1 \times \xi_3((x_1, 0, x_3, 0), s) \) and \( N_{G_1 \times G_3}(x_1, x_3) = \emptyset \). We have
\[ N_{G_2 \times G_3}(x_2, x_3) = N_{G_2}(x_2)||N_{G_3}(x_3) \subseteq N_{G_1}(x_1)||N_{G_3}(x_3) = N_{G_1 \times G_3}(x_1, x_3) = \emptyset \]
Thus, \( (x_2, x_3) \) is a blocking state of \( G_2 \times G_3 \), which means \( s \in B(G_2 \times G_3) \). Therefore, in either case we have \( B(G_1 \times G_3) \subseteq B(G_2 \times G_3) \). Finally, follow the argument of Case 2, for any \( s \in (\Sigma \cup \Sigma')^* \) and \( (x_1, x_3) \in \xi_1 \times \xi_3((x_1, 0, x_3, 0), s) \), we have \( (x_2, x_3) \in \xi_2 \times \xi_3((x_2, 0, x_3, 0), s) \) such that \( N_{G_2 \times G_3}(x_2, x_3) \subseteq N_{G_1 \times G_3}(x_1, x_3) \).
4. Proof of Prop. 2.10: Let \( G_i = (X_i, \Sigma_i, \xi_i, x_{i,0}, X_{i,m}) \), where \( i = 1, 2 \), and \( P : \Sigma^* \to \Sigma^* \) be the natural projection. Since \( G_1 \subseteq G_2 \), by Prop. 2.4 we have
\[
N(G_1/\approx_{\Sigma'}) = P(N(G_1)) = P(N(G_2)) = N(G_2/\approx_{\Sigma'})
\]
To show \( B(G_1/\approx_{\Sigma'}) \subseteq B(G_2/\approx_{\Sigma'}) \), let \( \xi_i' (i = 1, 2) \) be the transition map of \( G_i/\approx_{\Sigma'} \). For any \( s \in B(G_1/\approx_{\Sigma'}) \), there exists \( x_1 \in X_1 \) such that
\[
\langle x_1 \rangle \in \xi_1'(<x_{1,0}, s>) \land (\exists s' \in \Sigma^*) \xi_1'(<x_{1,0}, s') \cap X_{1,m} \approx_{\Sigma'} = \emptyset
\]
which means there exists \( t \in \Sigma^* \) such that
\[
P(t) = s \land x_1 \in \xi_1(x_{1,0}, t) \land (\forall t' \in \Sigma^*) \xi_1(x_1, t') \cap X_{1,m} \neq \emptyset \Rightarrow t' \in (\Sigma - \Sigma')^*
\]
Thus, \( N(G_1(x_1) \subseteq (\Sigma - \Sigma')^* \). There are two cases. Case 1: \( x_1 \) is a blocking state of \( G_1 \). Then \( t \in B(G_1) \), which means \( s = P(t) \in P(B(G_1)) \). Since \( G_1 \subseteq G_2 \), we have \( B(G_1) \subseteq B(G_2) \). Thus, by Prop. 2.4, we have \( s \in B(G_1) \subseteq B(G_2/\approx_{\Sigma'}) \). Case 2: \( x_1 \) is a nonblocking state. Thus \( t \in N(G_1) \). Clearly \( x_1 \notin X_{1,m} \). Since \( G_1 \subseteq G_2 \), there exists \( x_2 \in X_2 \) such that
\[
x_2 \in \xi_2(x_{2,0}, t) \land N(G_1(x_1) = N(G_2(x_2) \land \langle x_1 \rangle \in X_{1,m} \iff x_2 \in X_{2,m}]
\]
Since \( N(G_1(x_1) \subseteq (\Sigma - \Sigma')^* \), we have
\[
(\forall t' \in \Sigma^*) \xi_2(x_2, t') \cap X_{2,m} \neq \emptyset \Rightarrow t' \in (\Sigma - \Sigma')^*
\]
Thus, \( x_1 \notin X_{1,m} \). We have \( x_2 \notin X_{2,m} \). Thus, by the definition of automaton abstraction,
\[
\langle x_2 \rangle \in \xi_2'(<x_{2,0}, P(t)>) \land (\forall s' \in \Sigma^*) \xi_2'(<x_2, s'>) \cap X_{2,m} \approx_{\Sigma'} = \emptyset
\]
which means \( s = P(t) \in B(G_2/\approx_{\Sigma'}) \). Thus, in either case we have \( B(G_1/\approx_{\Sigma'}) \subseteq B(G_2/\approx_{\Sigma'}) \).

Finally, for each \( s \in N(G_1/\approx_{\Sigma'}) \), there exists \( x_1 \in X_1 \) such that
\[
\langle x_1 \rangle \in \xi_1'(x_{1,0}, t) \land (\exists t' \in \Sigma^*) P(t') = s' \land \xi_1(x_1, t') \cap X_{1,m} \neq \emptyset
\]
which means there exists \( t \in \Sigma^* \) with \( P(t) = s \) such that
\[
x_1 \in \xi_1'(x_{1,0}, t) \land (\exists t' \in \Sigma^*) P(t') = s' \land \xi_1(x_1, t') \cap X_{1,m} \neq \emptyset
\]
Clearly, \( t \in N(G_1) \). Thus, by \( G_1 \subseteq G_2 \), we have
\[
(\exists x_2 \in \xi_2(x_{2,0}, t)) N(G_1(x_1) \supseteq N(G_2(x_2) \land x_1 \in X_{1,m} \iff x_2 \in X_{2,m}]
\]
Since \( G_2 \) is standardized, \( < x_2 > \in \xi_2'(<x_{2,0}, s>) \). For any \( s' \in N(G_2/\approx_{\Sigma'}) \), we have \( \xi_2'(<x_{2,0}, s'>) \cap X_{2,m} \approx_{\Sigma'} \neq \emptyset \). If \( s' = \epsilon \), then \( x_2 \in X_{2,m} \), which means \( x_1 \in X_{1,m} \), namely \( \epsilon \in N(G_2/\approx_{\Sigma'}) \). If \( s' \neq \epsilon \), then there exists \( t' \in \Sigma^* \) with \( P(t') = s' \) such that \( \xi_2(x_2, t') \cap X_{2,m} \neq \emptyset \), which means \( t' \in N(G_2(x_2) \subseteq N(G_1(x_1)) \). Thus,
\[
\xi_1'(<x_{1,0}, t'), s'> \cap X_{1,m} \approx_{\Sigma'} \neq \emptyset
\]
namely \( s' \in N(G_1/\approx_{\Sigma'}) \). Thus, \( N(G_2/\approx_{\Sigma'}) \subseteq N(G_1/\approx_{\Sigma'}) \).
We have two cases to consider. Case 1: $x$ is a blocking state. Then clearly $t \in B(G)$. By Prop. 2.4, $P_{13}(t) = s = P_{23}(P_{12}(t)) \in P_{23}(P_{12}(B(G))) \subseteq B((G/\approx_{\Sigma'})/\approx_{\Sigma''})$. Case 2: $x$ is a nonblocking state. Clearly $x \notin X_m$, which means $<x >_{\Sigma'} \notin (X_m/\approx_{\Sigma'})/\approx_{\Sigma''}$.

Thus, from Expression (7) and the definition of automaton abstraction, we get that

$<x >_{\Sigma'} \notin \xi''(<x_0 >_{\Sigma'} \notin (X_m/\approx_{\Sigma'})/\approx_{\Sigma''})$. Thus, if $x \notin X_m$, which means $<x >_{\Sigma'} \notin (X_m/\approx_{\Sigma'})/\approx_{\Sigma''}$.

Thus, we have two cases to consider. Case 1: $x \notin X_m$, which means $<x >_{\Sigma'} \notin (X_m/\approx_{\Sigma'})/\approx_{\Sigma''}$. Clearly, we have the following expression: $<x >_{\Sigma'} \notin (X_m/\approx_{\Sigma'})/\approx_{\Sigma''}$.

Thus, we have two cases to consider. Case 1: $x \notin X_m$, which means $<x >_{\Sigma'} \notin (X_m/\approx_{\Sigma'})/\approx_{\Sigma''}$. Clearly, we have the following expression: $<x >_{\Sigma'} \notin (X_m/\approx_{\Sigma'})/\approx_{\Sigma''}$.

Thus, we have two cases to consider. Case 1: $x \notin X_m$, which means $<x >_{\Sigma'} \notin (X_m/\approx_{\Sigma'})/\approx_{\Sigma''}$.

Thus, we have two cases to consider. Case 1: $x \notin X_m$, which means $<x >_{\Sigma'} \notin (X_m/\approx_{\Sigma'})/\approx_{\Sigma''}$.

Thus, we have two cases to consider. Case 1: $x \notin X_m$, which means $<x >_{\Sigma'} \notin (X_m/\approx_{\Sigma'})/\approx_{\Sigma''}$.

Thus, we have two cases to consider. Case 1: $x \notin X_m$, which means $<x >_{\Sigma'} \notin (X_m/\approx_{\Sigma'})/\approx_{\Sigma''}$.
Thus, \(s' \in N(G/\approx_{\Sigma''})\). In either case, we have

\[
N(G/\approx_{\Sigma''}) \subseteq N(G/\approx_{\Sigma''})(< x >_{\Sigma''})
\]

The proposition follows.

6. Proof of Prop. 3.5: The ONLY IF part is obvious. So we only need to show the IF part.

Let \(S\) be a (canonical) recognizer of \(N(A)\), i.e. \(N(S) = N(A)\) and \(L(S) = \overline{N(A)} = L(A)\) (because \(B(G \times A) = \emptyset\)). Then we have

\[
N(G \times S) = N(G) \cap N(S) = N(G) \cap N(A) = N(G \times A) \subseteq N(G \times H)
\]

Next, we show \(B(G \times S) = \emptyset\). Let \(G = (X, \Sigma, \xi, x_0, X_m)\), \(A = (Z, \Sigma', \delta, z_0, Z_m)\) and \(S = (Y, \Sigma', \eta, y_0, Y_m)\). Suppose \(B(G \times S) \neq \emptyset\). Then there exists \(s \in B(G \times S)\) such that

\[
(\exists (x, y) \in \xi \times \eta((x_0, y_0), s))(\forall s' \in \Sigma^*) \xi \times \eta((x, y), s') \cap (X_m \times Y_m) = \emptyset
\]

Let \(P : \Sigma^* \rightarrow \Sigma^*\) be the natural projection. Then \(P(s) \in L(S) = \overline{N(A)}\). Then there exists \(z \in \delta(z_0, P(s))\), namely \((x, z) \in \xi \times \delta((x_0, z_0), s)\). Since \(B(G \times A) = \emptyset\), we get that

\[
(\exists s' \in \Sigma^*) \xi \times \delta((x, z), s') \cap (X_m \times Y_m) \neq \emptyset
\]

Thus, \(\xi(x, s') \cap X_m \neq \emptyset\) and \(P(ss') \in N(A) = N(S)\). Since \(S\) is deterministic, \(\eta(y, P(s')) \cap Y_m \neq \emptyset\). Therefore, \(\xi \times \eta((x, y), s') \cap (X_m \times Y_m) \neq \emptyset\) - contradicting the fact that \((x, y)\) is a blocking state. Thus, \(B(G \times S) = \emptyset\).

For each \(s \in L(G \times S)\), let \(x \in \xi(x_0, s)\) and \(y \in \eta(y_0, P(s))\). Since \(A\) is state-controllable, for any \(z \in \delta(z_0, P(s))\), we have

\[
E_G(x) \cap \Sigma_{uc} \cap \Sigma' \subseteq E_A(z)
\]

Since \(E_S(y) = \bigcup_{z \in \delta(z_0, P(s))} E_A(z)\), we have

\[
E_G(x) \cap \Sigma_{uc} \cap \Sigma' \subseteq E_S(y)
\]

Thus, \(S\) is state controllable with respect to \(G\) and \(\Sigma_{uc}\).

Next, we show that \(S\) is state observable w.r.t. \(G\) and \(P_o\). Suppose it is not true. Then there exist \(s, s' \in L(G \times S) \subseteq L(A)\) with \(P_o(s) = P_o(s')\), \((x, y) \in \xi \times \eta((x_0, y_0), s)\) and \((x', y') \in \xi \times \eta((x_0, y_0), s')\) such that

\[
E_{G \times S}(x, y) \cap E_G(x') \cap \Sigma' \nsubseteq E_S(y')
\]

Since \(S\) is deterministic, we have that

\[
(\exists s \in \Sigma') s \sigma \in L(G) \land s' \sigma \in L(G) \land P(s) \sigma \in L(S) \land P(s') \sigma \notin L(S)
\]

Since \(L(S) = L(A)\), we have that there exist \(s, s' \in L(A)\) with \(P_o(s) = P_o(s')\) such that

\[
s \sigma \in L(G) \land s' \sigma \in L(G) \land P(s) \sigma \in L(A) \land P(s') \sigma \notin L(A)
\]

Pick \(z \in \delta(z_0, P(s))\) and \(z' \in \delta(z_0, P(s'))\), then \((x, z) \in \xi \times \delta((x_0, z_0), s)\) and \((x', z') \in \xi \times \delta((x_0, z_0), s')\). Furthermore, we have that \(\sigma \in E_{G \times A}(x, z) \cap E_G(x') \cap \Sigma'\) but \(\sigma \notin E_A(z')\), namely

\[
E_{G \times A}(x, z) \cap E_G(x') \cap \Sigma' \nsubseteq E_A(z')
\]

which contradicts that \(A\) is state observable w.r.t. \(G\) and \(P_o\). Thus, \(S\) is state observable w.r.t. \(G\) and \(P_o\).

Finally, we show that \(S\) is state normal w.r.t. \(G\) and \(P_o\) if \(A\) is state normal w.r.t. \(G\) and \(P_o\). Let \(s \in L(G \times S)\) and \(s' \in P_o^{-1}(P_o(s)) \cap L(G \times S)\). For any \((x, y) \in \xi \times \eta((x_0, y_0), s')\) and \(s'' \in \Sigma^*\) with \(P_o(s''') = P_o(s)\), we need to show that

\[
\xi(x, s'') \neq \emptyset \Rightarrow \eta(y, P(s'')) \neq \emptyset
\]
Suppose it is not true. Then there exist \( x \in X \) and \( s'' \in \Sigma^* \) such that \( \xi(x, s'') \neq \emptyset \) but \( \eta(y, P(s'')) = \emptyset \). Since \( S \) is deterministic, \( P(s' s'') \notin L(S) \). If \( s \in L(G \times S) \), we get that \( P(s) \in L(S) = L(A) \). Let \( s \preceq P(s' s'') \) such that \( s \in L(A) \) but \( s \neq L(A) \). Such \( s \) must exists because at least \( \varepsilon \leq P(s' s'') \) and \( \varepsilon \in P_{\varepsilon}^{-1}(P_o(s)) \cap L(G \times A \) \) and \( P(s' s'') \notin L(A) \).

If \( P(s') \leq P(s) \), then let \( z \in \delta(z_0, P(s')) \), and we have \( (x, z) \in \xi((x_0, z_0), s') \). But \( \delta(x, P(s'')) = \emptyset \), which contradicts the fact that \( A \) is state-normal with respect to \( G \) and \( P_o \). If \( P(s) \leq P(s') \) and \( P(s) \neq P(s') \), let \( z \in \delta(z_0, P(s)) \). There exist \( x' \in \xi(x_0, s) \) and \( s'' \in \Sigma^* \) such that \( ss'' = s' \) and \( x \in \xi(x', s') \). Then we have \( (x, z) \in \xi((x_0, z_0), s) \), \( \xi(x', s' s'') \neq \emptyset \) but \( \delta((z, s's'') = \emptyset \), which still contradicts the fact that \( A \) is state-normal with respect to \( G \) and \( P_o \).

Thus,

\[
\xi(x, s'') \neq \emptyset \Rightarrow \eta(y, P(s'')) \neq \emptyset
\]

which means \( S \) is state-normal with respect to \( G \) and \( P_o \).

7. Proof of Prop. 3.6: Since \( N(G \times S_i) \subseteq N(G \times H) \) for \( i = 1, 2 \), we have

\[
N(G \times S) = N(G)||N(S) = N(G)||N(S_1 \cup N(S_2)) = N(G \times S_1) \cup N(G \times S_2) \subseteq N(G \times H)
\]

Next, we show \( B(G \times S) = \emptyset \). Let \( G = (X, \Sigma, \xi, x_0, M) \), \( S = (Y, \Sigma', \eta, y_0, M) \) and \( S = (Y, \Sigma_1, \eta, y_0, M) \). Suppose \( B(G \times S) \neq \emptyset \). Then there exists \( s \in B(G \times S) \) such that

\[
(\exists (x, y) \in \xi \times \eta(x_0, y_0, s)) (\exists s' \in \Sigma^* \xi \times \eta((x, y), s') \cap (M \times M) = \emptyset)
\]

Let \( P : \Sigma^* \rightarrow \Sigma^* \) be the natural projection. Then \( P(s) \in L(S) = N(S_1) \cup N(S_2) \). Thus, either \( P(s) \in N(S_1) \) or \( P(s) \in N(S_2) \). Without loss of generality, suppose \( P(s) \in N(S_1) \). Then there exists \( y_0 \in \eta((x_0, y_0, P(s))) \), namely \( (x, y_1) \in \xi \times \eta((x_0, y_0, s), s) \). Since \( B(G \times S_1) = \emptyset \), we get that

\[
(\exists s' \in \Sigma^*) \xi \times \eta_1((x, y_1), s') \cap (M \times M) = \emptyset
\]

Thus, \( \xi(x, s') \cap M \neq \emptyset \) and \( P(s) \notin N(S_1) \subseteq N(S) \). Since \( S \) is deterministic, we get \( \eta(y, P(s')) \cap M \neq \emptyset \). Therefore, \( \xi \times \eta((x, y), s') \cap (M \times M) \neq \emptyset \) - contradicting the fact that \( (x, y) \) is a blocking state. Thus, \( B(G \times S) = \emptyset \).

For each \( s \in L(G \times S) \), let \( x \in \xi(x_0, s) \) and \( y \in \eta(x_0, P(s)) \). Since \( P(s) \in L(S) = N(S) \), without loss of generality, suppose \( P(s) \in N(S_1) \). Let \( B(G \times S_1) = \emptyset \). Then

\[
E_G(x) \cap \Sigma uc \subseteq \Sigma uc \subseteq E_S(1, y, P(s)) \subseteq \emptyset
\]

Thus, \( S \) is state controllable with respect to \( G \) and \( \Sigma uc \).

Finally, we show that \( S \) is state-normal with respect to \( G \) and \( P_o \). Let \( s \in L(G \times S) \) and \( s' \in P_0^{-1}(P_o(s)) \cap L(G \times S) \). For any \( (x, y) \in \xi \times \eta((x_0, y_0, s'), s') \) with \( P_0(s' s'') = P_0(s) \), we need to show that

\[
\xi(x, s'') \neq \emptyset \Rightarrow \eta(y, P(s'')) \neq \emptyset
\]

Suppose it is not true. Then there exist \( x \in X \) and \( s'' \in \Sigma^* \) such that \( \xi(x, s'') \neq \emptyset \) but \( \eta(y, P(s'')) = \emptyset \). Since \( S \) is deterministic, \( P(s' s'') \notin L(S) \). If \( s \in L(G \times S) \), we get that \( P(s) \in L(S) = L(S_1) \cup L(S_2) \). Without loss of generality, suppose \( P(s) \in L(S_1) \). Let \( s \preceq P(s' s'') \) such that \( s \in L(S_1) \) but \( s \neq L(S_1) \). Such \( s \) must exist because at least \( \varepsilon \leq P(s' s'') \) and \( \varepsilon \in P_{\varepsilon}^{-1}(P_o(s)) \cap L(G \times S) \) and \( P(s' s'') \notin L(S_1) \). If \( P(s') \leq P(s) \), then let \( y_1 \in \eta_1((y_0, P(s'))) \), and we have \( (x, y_1) \in \xi \times \eta((x_0, y_1, s'), s') \). But \( \eta_1((y_1, P(s'')) = \emptyset \), which contradicts the fact that \( S_1 \) is state-normal with respect to \( G \) and \( P_o \). If \( P(s) \leq P(s') \) and \( P(s) \neq P(s') \), let \( y_1 \in \eta_1((y_0, P(s))) \). There exist \( x' \in \xi((x_0, s), M) \) and \( s' \in \Sigma^* \) such that \( ss' = s' \) and \( x \in \xi(x', s') \). Then we have \( (x, y_1) \in \xi \times \eta((x_0, y_0, s), s), \xi(x', s' s'') \neq \emptyset \) but \( \eta_1((y_1, s's'') = \emptyset \), which still contradicts the fact that \( S_1 \) is state-normal with respect to \( G \) and \( P_o \). Thus,

\[
\xi(x, s'') \neq \emptyset \Rightarrow \eta(y, P(s'')) \neq \emptyset
\]
which means $S$ is state-normal with respect to $G$ and $P_o$.

8. Proof of Lemma 3.7: We first show the IF part. Suppose it is not true. Then $S$ is state-controllable w.r.t. $G$ and $\Sigma_{uc}$, but it is not state-controllable w.r.t. $G/\approx_{\Sigma'}$ and $\Sigma_{uc} \cap \Sigma'$. Thus

\[ (\forall s \in L(G \times S))(\forall x \in \xi(s_0, s))(\forall y \in \eta(y_0, P(s))) E_G(x) \cap \Sigma_{uc} \cap \Sigma' \subseteq E_S(y) \]  

(9)

where $P : \Sigma^* \rightarrow \Sigma'^*$ is the natural projection, and there exists $t \in L((G/\approx_{\Sigma'}) \times S)$ such that

\[ (\exists < x > \in \xi'(< x_0 >, t))((\forall y \in \eta(y_0, t)) E_{G/\approx_{\Sigma'}}(< x >) \cap \Sigma_{uc} \cap \Sigma' \nsubseteq E_S(y) \]  

(10)

where $\xi'$ is the transition map of $G/\approx_{\Sigma'}$. By the definition of automaton abstraction we have

\[ E_{G/\approx_{\Sigma'}}(< x >) = \{ \sigma \in \Sigma'| (\exists u \in (\Sigma - \Sigma'^*)) ((\exists x' \in \xi(x, u)) \sigma \in E_G(x')) \} \]

Thus, $E_{G/\approx_{\Sigma'}}(< x >) \cap \Sigma_{uc} \cap \Sigma' \nsubseteq E_S(y)$ implies that

\[ (\exists u \in (\Sigma - \Sigma'^*)) ((\exists x' \in \xi(x, u)) E_G(x') \cap \Sigma_{uc} \cap \Sigma' \nsubseteq E_S(y) \]

From expression (10) we also get that

\[ (\exists s \in \Sigma^*) P(s) = t \land x \in \xi(x_0, s) \]

Thus, $(x, y) \in \xi \times \eta((x_0, y_0), s)$. Since $u \in (\Sigma - \Sigma'^*)$, we have $P(su) = x$, from which we can get that $(x', y) \in \xi \times \eta((x_0, y_0), su)$. Thus, $su \in L(G \times S)$, which means

\[ (\exists su \in L(G \times S)) (\exists x' \in \xi(x_0, su)) (\exists y \in \eta(y_0, P(s))) E_G(x') \cap \Sigma_{uc} \cap \Sigma' \nsubseteq E_S(y) \]

which contradicts expression (9). Thus, the IF part is true.

Next, we show the ONLY IF part. Suppose it is not true. Then $S$ is state-controllable w.r.t. $G/\approx_{\Sigma'}$ and $\Sigma_{uc} \cap \Sigma'$, but it is not state-controllable w.r.t. $G$ and $\Sigma_{uc}$. Thus, for any $t \in L((G/\approx_{\Sigma'}) \times S), (\forall < x > \in \xi'(< x_0 >, t))((\forall y \in \eta(y_0, t))E_{G/\approx_{\Sigma'}}(< x >) \cap \Sigma_{uc} \cap \Sigma' \subseteq E_S(y) \]

(11)

and

\[ (\exists s \in L(G \times S)) (\exists x \in \xi(x_0, s)) (\exists y \in \eta(y_0, P(s))) E_G(x) \cap \Sigma_{uc} \cap \Sigma' \nsubseteq E_S(y) \]

(12)

Since $G$ is standardized, from expression (12) we get that $(\exists < x > \in \xi'(< x_0 >, P(s)))$. Since

\[ E_G(x) \cap \Sigma_{uc} \cap \Sigma' \nsubseteq E_S(y) \Rightarrow E_{G/\approx_{\Sigma'}}(< x >) \cap \Sigma_{uc} \cap \Sigma' \nsubseteq E_S(y) \]

we get that

\[ (\exists < x > \in \xi'(< x_0 >, P(s))) (\exists y \in \eta(y_0, P(s))) E_{G/\approx_{\Sigma'}}(< x >) \cap \Sigma_{uc} \cap \Sigma' \nsubseteq E_S(y) \]

which contradicts expression (11). Thus, the ONLY IF part must be true.

9. Proof of Lemma 3.8: (1) Let $S$ is state observable with respect to $G/\approx_{\Sigma'}$ and $P_o$. Thus, for any $s, s' \in L((G/\approx_{\Sigma'}) \times S)$ with $P_o(s') = P_o(s')$, and for any $(< x >, y) \in \xi' \times \eta((< x_0 >, y_0), s)$ and $(< x' >, y') \in \xi' \times \eta((< x_0 >, y_0), s')$, we have

\[ E_{G/\approx_{\Sigma'}}(< x >, y) \cap E_{G/\approx_{\Sigma'}}(< x' >) \cap \Sigma' \subseteq E_S(y') \]

(13)

Assume that $S$ is not state-observable w.r.t. $G$ and $P_o$. Then there are $t, t' \in L(G \times S)$ with $P_o(t) = P_o(t)$,

\[ (\exists x, y \in \xi \times \eta((x_0, y_0), t))(\exists x', y' \in \xi \times \eta((x_0, y_0), t')) E_{G \times S}(x, y) \cap E_G(x') \cap \Sigma' \nsubseteq E_S(y') \]

Since $G$ is standardized, we get $(\exists < x > \in \xi'(< x_0 >, P(t)))$ and $(\exists < x' > \in \xi'(< x_0 >, P(t')))$. We also have $y \in \eta(y_0, P(t))$ and $y' \in \eta(y_0, P(t'))$. Thus, $(< x >, y) \in \xi' \times \eta((< x_0 >, y_0), P(t))$ and $(< x' >, y') \in \xi' \times \eta((< x_0 >, y_0), P(t'))$. We also have that

\[ E_{G \times S}(x, y) \cap E_G(x') \cap \Sigma' \nsubseteq E_S(y') \Rightarrow E_{G/\approx_{\Sigma'}}(x, y) \cap E_{G/\approx_{\Sigma'}}(x') \cap \Sigma' \nsubseteq E_S(y') \]
Finally, since $P(t) = P(t')$, we have $P'_o(P(t)) = P'_o(P(t'))$. Thus, there exist $s = P(t)$ and $s' = P(t')$ with $P'_o(s) = P'_o(s')$, and there exist

$$<(x, y) \in \xi' \times \eta((<x, y_0>, P(t)) \text{ and } (<x', y) \in \xi' \times \eta((<x, y_0), P(t'))$$

such that

$$E_{G/\approx_{\Sigma'}}(x, y) \cap E_{G/\approx_{\Sigma'}}(x, y') \subseteq E_{S}(y')$$

which contradicts expression (13). Therefore, (1) is true.

(2) Suppose $\Sigma_o \subseteq \Sigma'$. Let $S$ be state observable w.r.t. $G$ and $P_o$. Thus, for any $t, t' \in L(G \times S)$ with $P_o(t) = P_o(t')$, 

$$(\forall(x, y) \in \xi \times \eta((x, y_0), t))(\forall(x', y') \in \xi \times \eta((x, y_0), t')) E_{G \times S}(x, y) \cap E_{G \times S}(x', y') \subseteq E_S(y')$$

(14)

Assume that $S$ is not state-observable w.r.t. $G/\approx \Sigma'$ and $P'_o$. Then there exist $s, s' \in L((G/\approx \Sigma') \times S)$ with $P'_o(s) = P'_o(s')$, and $(<x, y) \in \xi' \times \eta((<x, y_0), s)$ and $(<x', y) \in \xi' \times \eta((<x, y_0), s')$,

$$E_{G/\approx_{\Sigma'}}(x, y) \cap E_{G/\approx_{\Sigma'}}(x', y') \subseteq E_{S}(y')$$

(15)

Clearly, there exist $t, t' \in \Sigma'$ with $P(t) = s$ and $P(t') = s'$ such that $(x, y) \in \xi \times \eta((x, y_0), t)$ and $(x', y') \in \xi \times \eta((x, y_0), t')$. We also have that

$$E_{G/\approx_{\Sigma'}}(x, y) = \{s \in \Sigma' | (\exists u \in (\Sigma - \Sigma')^*)((\exists \hat{x} \in \xi(x, u)) s \in E_{G \times S}(\hat{x}, y)\}$$

and

$$E_{G/\approx_{\Sigma'}}(x', y') = \{s \in \Sigma' | (\exists u \in (\Sigma - \Sigma')^*)((\exists \hat{x}' \in \xi(x', u)) s \in E_{G \times S}(\hat{x}', y')\}$$

Thus, from expression (15), there exist $u, u' \in (\Sigma - \Sigma')^*$ and $\hat{x} \in \xi(x, u)$ and $\hat{x}' \in \xi(x', u')$ such that

$$E_{G \times S}(\hat{x}, y) \cap E_{G}(\hat{x}') \subseteq E_{S}(y')$$

Since $P'_o(P(t)) = P'_o(P(t'))$, we have, for each $(<x, y) \in \xi' \times \eta((<x, y_0), s')$ and $s'' \in \Sigma^*$,

$$P'_o(s'' \circ s') \Rightarrow [\xi'(<x, y, s'') \neq \varnothing \Rightarrow \eta(y, s'' \circ s') \neq \varnothing]$$

(16)

Suppose $S$ is not state-observable w.r.t. $G$ and $P_o$. Then there exist $t \in L(G \times S)$ and $t' \in P^{-1}_o(t) \cap L(G \times S)$ such that there exist $(x, y) \in \xi \times \eta((x, y_0), t)$ and $t'' \in \Sigma^*$,

$$P_o(t'' \circ t') \Rightarrow \eta(y, P(t'')) = \varnothing$$

Let $P''_o : \Sigma_o \rightarrow (\Sigma_o \cap \Sigma')^*$ be the natural projection. Since $G$ is standardized, we have

$$P(t) \in L((G/\approx_{\Sigma'}) \times S)$$

and from $(x, y) \in \xi \times \eta((x, y_0), t)$ we get $(<x, y) \in \xi' \times \eta((<x, y_0), P(t))$. Since $t' \in P^{-1}_o(t) \cap L(G \times S)$, we get that

$$P_o(t') \leq P_o(t) \land P(t) \in L((G/\approx_{\Sigma'}) \times S)$$

which means

$$P'_o(P(t')) = P''_o(P_o(t')) \leq P''_o(P_o(t)) = P'_o(P(t))$$
Thus, $P(t') \in \overline{P_o^{-1}(P_o(s)) \cap L((G/\approx_{\Sigma'}) \times S)}$. Since $P_o(t't'') = P_o(t)$, we have
\[
P_o'(P(t')) = P_o'(P_o(t't'')) = P_o(t) = P_o'(P(t))
\]
Since $\xi(x, t''') \not\in \emptyset$, if $P(t'') = \epsilon$ we have $\xi'(\langle x, \epsilon \rangle) \not\in \emptyset$; if $P(t'') \neq \epsilon$ we have $\xi(\langle x, P(t'') \rangle) \not\in \emptyset$. Thus, there exist $s = P(t) \in L((G/\approx_{\Sigma'}) \times S)$, $s' = P(t') \in \overline{P_o^{-1}(P_o(s)) \cap L((G/\approx_{\Sigma'}) \times S)}$ such that there exist $\langle x, s'' \rangle \in \xi' \times \eta(\langle x, t'' \rangle)$ and $s'' = P(t'') \in T^*$,
\[
P_o(s's'') = P_o'(s) \land \xi'(\langle x, s'' \rangle) \not\in \emptyset \land \eta(y, s'') = \emptyset
\]
which contradicts expression (16). Thus, (1) is true.

(2) Let $S$ be state-normal w.r.t. $G$ and $P_o$. Then for any $t \in L(G \times S)$, $t' \in \overline{P_o^{-1}(P_o(t)) \cap L(G \times S)}$, we have, for each $\langle x, y \rangle \in \xi \times \eta(\langle x_0, y_0 \rangle, t')$ and $t'' \in T^*$,
\[
P_o(t't'') = P_o(t) \Rightarrow [\xi(x, t'') \not\in \emptyset \Rightarrow \eta(y, P(t'')) \not\in \emptyset] \tag{17}
\]
Suppose $S$ is not state-normal w.r.t. $G/\approx_{\Sigma'}$ and $P_o'$. Then there exist $s \in L((G/\approx_{\Sigma'}) \times S)$, $s' \in \overline{P_o^{-1}(P_o(s)) \cap L((G/\approx_{\Sigma'}) \times S)}$, $\langle x, y \rangle \in \xi' \times \eta(\langle x_0, y_0 \rangle, s')$ and $s'' \in T^*$ such that
\[
P_o(s's'') = P_o'(s) \land \xi'(\langle x, s'' \rangle) \not\in \emptyset \land \eta(y, s'') = \emptyset
\]
Clearly, there exists $t \in L(G \times S)$ such that $P(t) = s$. There also exists $t' \in T^*$ such that $P(t) = s'$ and $\langle x, y \rangle \in \xi \times \eta(\langle x_0, y_0 \rangle, t')$. Thus, $t' \in L(G \times S)$. Since $s' \in \overline{P_o^{-1}(P_o(s)) \cap L((G/\approx_{\Sigma'}) \times S)}$, we have $P_o(s') \not\subseteq P_o(s)$. Since $\Sigma_o \subseteq \Sigma'$, we have $P_o(t') = P_o'(P(t')) = P_o(s) \leq P_o'(P(t)) = P_o(t)$. Thus, $t' \in \overline{P_o^{-1}(P_o(t)) \cap L(G \times S)}$. Since $\xi'(\langle x, s'' \rangle) \not\in \emptyset$, there exists $t'' \in T^*$ such that $\xi(x, t'') \not\in \emptyset$. From $P_o(s's'') = P_o'(s)$ we have $P_o(t't'') = P_o(P(t''')) = P_o(s''') = P_o(s) = P_o(t)$. Thus, there exist $t \in L(G \times S)$, $t' \in \overline{P_o^{-1}(P_o(t)) \cap L(G \times S)}$, $\langle x, y \rangle \in \xi \times \eta(\langle x_0, y_0 \rangle, t')$ and $t'' \in T^*$ such that
\[
P_o(t't'') = P_o(t) \land \xi(x, t'') \not\in \emptyset \land \eta(y, P(t'')) = \emptyset
\]
which contradicts expression (17). Thus, (2) is true.


Bibliography