Automatica 50 (2014) 287–290

Contents lists available at ScienceDirect

Automatica

journal homepage: www.elsevier.com/locate/automatica

Technical communique

Bisimilarity enforcing supervisory control for deterministic specifications☆

Yajuan Sun [a,1], Hai Lin [b], Ben M. Chen [a]

[a] Department of Electrical and Computer Engineering, National University of Singapore, Singapore
[b] Department of Electrical Engineering, University of Notre Dame, USA

ARTICLE INFO

Article history:
Received 18 March 2013
Received in revised form
23 June 2013
Accepted 10 September 2013
Available online 11 November 2013

Keywords:
Supervisory control
Bisimulation
Discrete event systems

ABSTRACT

This paper studies the supervisory control of nondeterministic discrete event systems to achieve a bisimulation equivalence between the controlled system and the deterministic specification. In particular, a necessary and sufficient condition is given for the existence of a bisimilarity enforcing supervisor, and a polynomial algorithm is developed to verify such a condition. When the existence condition holds, a bisimilarity enforcing supervisor is constructed. Otherwise, two methods are provided for synthesizing supremal feasible sub-specifications.

1. Introduction

The notion of bisimulation, introduced by Milner (1989) and Park (1981), has been successfully used as a behavior equivalence in model checking, software verification and analysis of continuous, hybrid and discrete event systems. What makes bisimulation appealing is its capability in complexity mitigation, especially when we deal with large or infinite states. In the recent literature of robotic symbolic motion planning, a bisimulation equivalence between the abstracted finite state quotient system and the continuous robotic dynamics needs to be obtained (Belta et al., 2007; Pappas & Tabuada, 2006), which is the key to guarantee the existence of a feasible continuous path corresponding to the synthesized symbolic sequence that satisfies logical specifications. Additionally, a supervisor designed for the abstracted quotient system needs to enforce bisimilarity with respect to logical specifications. This brings a new challenge to the discrete event supervisory control society since most existing results on supervisor synthesis are based on language enforcement. However, it is known that language equivalence cannot imply bisimulation equivalence.

An early effort on bisimilarity supervisory control can be found in Qin and Lewis (1990), where a supervisor was developed to enforce bisimulation equivalence between the supervised system and the deterministic specification, under the assumption that all the events are controllable. Tabuada (2008) solved the controller synthesis problem for bisimulation equivalence in a wide variety of scenarios including continuous systems, hybrid systems and discrete event systems, in which the bisimilarity controller is a morphism in the context of category theory. Zhou, Kumar, and Jiang (2006) proposed a necessary and sufficient condition for the existence of a bisimilarity supervisor, where the plant and specification are nondeterministic. However, this work does not provide a systematic way to design the bisimilarity supervisor when it exists. To address this problem, Zhou and Kumar (2011) focused on deterministic supervisors, where the construction of deterministic bisimilarity supervisors and infimal superspecifications was developed.

Despite these progresses, the main obstacle of the existing literature to real applications is the expansively computational complexity (double exponential complexity in state sizes of the plant and the specification (Zhou et al., 2006) and single exponential complexity in state sizes of the plant and the specification (Zhou & Kumar, 2011)). Due to these concerns, we investigate bisimilarity control for deterministic specifications in this paper, for which a necessary and sufficient condition is proposed for the existence of a bisimilarity supervisor that can be effectively verified in polynomial complexity. Another important issue missing in the literature is how to change the specification when there does not exist such a

☆ Financial supports from NSF-CNS-1239222 and NSF-EECS-1253488 for this work are greatly acknowledged. The material in this paper was not presented at any conference. This paper was recommended for publication in revised form by Associate Editor Jan Komenda under the direction of Editor André L. Tits.

E-mail addresses: elesuya@nus.edu.sg (Y. Sun), hlin1@nd.edu (H. Lin), bmchen@nus.edu.sg (B.M. Chen).

[1] Tel.: +65 9152 8061; fax: +65 6779 1103.

0005-1098/$ – see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.automatica.2013.09.025

bisimilarity supervisor for the original specification. To address this problem, the calculation of supremal feasible sub-specifications is investigated in this paper.

Compared to the literature, the contributions of this paper mainly lie on the following aspects. First, a new necessary and sufficient condition is introduced for the existence of a bisimilarity enforcing supervisor, which helps to shed light on what characters should a deterministic specification possess for bisimilarity control. Second, a test algorithm is proposed to verify the existence condition, which is shown to be of polynomial complexity. When the existence condition holds, we further present a systematic way to construct bisimilarity enforcing supervisors. Third, when a given specification does not always guarantee the existence of a bisimilarity enforcing supervisor, an important question arises that is how to find a maximally permissive specification which enables the synthesis of bisimilarity enforcing supervisors. Two methods are proposed to answer the question. One is based on a recursive algorithm to calculate the supremal feasible sub-specifications and the other directly computes such sub-specifications by formulas.

The rest of the paper is organized as follows. Section 2 presents the existence condition for bisimilarity enforcing supervisors. The algorithm for checking the existence condition can be found in Section 3. Section 4 explores the calculation of maximally permissive sub-specifications. This paper concludes with Section 5.

2. Existence condition for bisimilarity enforcing supervisors

The plant is modeled by a nondeterministic automaton $G = (X, \Sigma, x_0, \alpha, X_m)$, where X is a finite state set, Σ is a finite event set, $\alpha : X \times \Sigma \to 2^X$ is a transition function, x_0 is an initial state and $X_m \subseteq X$ is a marked state set. The active event set at state x is defined as $E_G(x) = \{\sigma \in \Sigma \mid \alpha(x, \sigma)\text{is defined}\}$. Let Σ^* be the set of all the finite strings over Σ including the empty string ϵ. The transition function α can be extended from events to traces, $\alpha : X \times \Sigma^* \to 2^X$, which is defined inductively as: for any $x \in X, \alpha(x, \epsilon) = \{x\}$; for any $s \in \Sigma^*$ and $\sigma \in \Sigma, \alpha(x, s\sigma) = \bigcup_{x' \in \alpha(x,s)} \alpha(x', \sigma)$. If the transition function is a partial map $\alpha : X \times \Sigma \to X$, G is said to be deterministic. For $X_1 \subseteq X$, the notation $\alpha|_{X_1 \times \Sigma}$ means α is restricted from a smaller domain $X_1 \times \Sigma$ to 2^{X_1}. Consider three languages $K, K_1, K_2 \subseteq \Sigma^*$. The Kleene closure of K, denoted as K^*, is the language $K^* = \cup_{n \in \mathbb{N}} K^n$, where $K^0 = \{\epsilon\}$ and for any $n \geq 0, K^{n+1} = K^n K$. The prefix closure of K, denoted as $\overline{K}$, is the language $\overline{K} = \{s \in \Sigma^* \mid (\exists t \in \Sigma^*)\, st \in K\}$. If $K = \overline{K}$, K is called to be prefix closed. The quotient of K_1 with respect to K_2, denoted as K_1/K_2, is the language $K_1/K_2 = \{s \in \Sigma^* \mid (\exists t \in K_2)\, st \in K_1\}$. The language and the marked language generated by G are defined by $L(G) = \{s \in \Sigma^* \mid \alpha(x_0, s)\text{is defined}\}$ and $L_m(G) = \{s \in \Sigma^* \mid \alpha(x_0, s) \cap X_m \neq \emptyset\}$ respectively. For a nondeterministic G, let $det(G)$ be a minimal deterministic automaton such that $L(det(G)) = L(G)$ and $L_m(det(G)) = L_m(G)$.

The plant G is controlled by a supervisor $S = (Y, \Sigma, y_0, \beta, Y_m)$. The supervisor can disable the events inside the controllable event set. This form of control can be captured by the parallel composition of the plant and the supervisor which is denoted by $G \parallel S$ (Cassandras, 2008). To find the synchronized state pairs of G and S, the synchronized state map $X_{synGS}: X \to 2^Y$ from G to S is defined as $X_{synGS}(x_1) = \{x_2 \in Y \mid (\exists s \in \Sigma^*)\, x_1 \in \alpha(x_0, s) \wedge x_2 \in \beta(y_0, s)\}$ (Zhou et al., 2006). Most literature on supervisory control aims to achieve a language equivalence between the supervised system and the specification. The necessary and sufficient condition for the existence of a language enforcing supervisor is language controllability. We denote Σ_c and $\Sigma_{uc} := \Sigma \setminus \Sigma_c$ as the controllable event set and the uncontrollable event set respectively. A language $K' \subseteq L(G)$ is said to be language controllable with respect to $L(G)$ and Σ_{uc} if $\overline{K'}\Sigma_{uc}^* \cap L(G) \subseteq \overline{K'}$. As a stronger behavior equivalence than language equivalence, bisimulation is stated as follows (Milner, 1989).

Given $G_1 = (X_1, \Sigma, x_{01}, \alpha_1, X_{m1})$ and $G_2 = (X_2, \Sigma, x_{02}, \alpha_2, X_{m2})$, a simulation relation is a binary relation $\phi \subseteq X_1 \times X_2$ such that $(x_1, x_2) \in \phi$ implies

(1) $(\forall \sigma \in \Sigma)[\forall x_1' \in \alpha_1(x_1, \sigma) \Rightarrow \exists x_2' \in \alpha_2(x_2, \sigma)$ such that $(x_1', x_2') \in \phi]$;
(2) $x_1 \in X_{m1} \Rightarrow x_2 \in X_{m2}$.

If there is a simulation relation $\phi \subseteq X_1 \times X_2$ such that $(x_{01}, x_{02}) \in \phi$, G_1 is said to be simulated by G_2, denoted by $G_1 \prec_\phi G_2$. For $\phi \subseteq (X_1 \cup X_2) \times (X_1 \cup X_2)$, if $G_1 \prec_\phi G_2$, $G_2 \prec_\phi G_1$ and ϕ is symmetric, ϕ is called a bisimulation relation between G_1 and G_2. If there is a bisimulation relation ϕ between G_1 and G_2, G_1 is called to be bisimilar to G_2, denoted by $G_1 \cong_\phi G_2$. We sometimes omit the subscript ϕ from $\prec_\phi$ or $\cong_\phi$ when it is clear from the context.

This paper investigates supervisory control to enforce a bisimulation equivalence between the supervised system and the given deterministic specification R. The supervisor which enables all the uncontrollable events and enforces bisimulation equivalence is formalized by the following notion. Unless otherwise stated we will use $G = (X, \Sigma, \alpha, x_0, X_m)$, $R = (Q, \Sigma, \delta, q_0, Q_m)$ and $S = (Y, \Sigma, \beta, y_0, Y_m)$ to denote the nondeterministic plant, the deterministic specification and the supervisor (possibly nondeterministic) respectively in the rest of the paper.

Definition 1. Given a plant G and a specification R, a supervisor S is said to be a bisimilarity enforcing supervisor for G and R if

(1) there is a bisimulation relation ϕ such that $G \parallel S \cong_\phi R$;
(2) $(\forall y \in Y)(\forall \sigma \in \Sigma_{uc})\, \beta(y, \sigma) \neq \emptyset$.

First, we focus on the following problem: given a nondeterministic plant G and a deterministic specification R, what property should R possess to guarantee the existence of a bisimilarity enforcing supervisor S for G and R? The result of Zhou and Kumar (2011) indicates that $G \parallel det(R) \cong R$ and language controllability of $L(R)$ are necessary and sufficient conditions for the existence of a deterministic bisimilarity supervisor. In particular, $G \parallel det(R) \cong R$ is reduced to $G \parallel R \cong R$ when R is deterministic. However, in these conditions, R gets entangled with G, which fails to provide an insight into the character of R for bisimilarity control. Moreover, the complexity of checking these condition is high. To address these problems, we introduce the following property.

Theorem 1. *Given a plant G and a deterministic specification R, there exists a bisimulation relation ϕ such that $G \parallel R \cong_\phi R$ if and only if there exists a simulation relation ϕ' such that $(q, x) \in \phi'$ for any $q \in Q$ and $x \in X_{synRG}(q)$.*

Proof. For sufficiency, let $\phi_1 = \{((x, q), q) \mid (\forall q \in Q)\, x \in X_{synRG}(q)\}$, it is obvious that $G \parallel R \cong_{\phi_1 \cup \phi_1^{-1}} R$. For necessity, we consider a relation $\phi_2 = \{(q, x) \mid (((x, q), q), (q, (x, q))) \in \phi\}$. Since R is deterministic, ϕ_2 is a simulation relation such that $(q, x) \in \phi_2$ for any $q \in Q$ and $x \in X_{synRG}(q)$.

If there exists a simulation relation ϕ such that $(q, x) \in \phi$ for any $q \in Q$ and $x \in X_{synRG}(q)$, then R is called to be synchronously simulated by G, denoted as $R \prec_{syn\phi} G$. The subscript ϕ can be omitted from $\prec_{syn\phi}$ when it is clear from the context. We can see that if $R \prec_{syn\phi} G$, then G possesses the branches which are bisimilar to R and the branches outside $L(R)$. Hence, $R \prec_{syn\phi} G$ iff $G \parallel R \cong R$. Therefore, the specification satisfying language controllability and synchronous simulation guarantees the existence of a bisimilarity enforcing supervisor. Moreover, the result of Theorem 1 offers computational advantages for checking the existence of a bisimilarity enforcing supervisor, which will be illustrated in the next section. Here, we show how to synthesize a bisimilarity enforcing supervisor when it exists. The following concept is developed.

Definition 2. Given $G = (X, \Sigma, x_0, \alpha, X_m)$, the Σ_{uc} completion of G, denoted by G_{uc}, is an automaton

$$G_{uc} = (X \cup \{D\}, \Sigma, x_0, \alpha_{uc}, X_m),$$

where for any $x \in X \cup \{D\}$ and $\sigma \in \Sigma$, the transition function is defined as

$$\alpha_{uc}(x,\sigma) = \begin{cases} \alpha(x,\sigma) & \sigma \in E_G(x); \\ \{D\} & (\sigma \in \Sigma_{uc} \setminus E_G(x)) \vee (x = D \wedge \sigma \in \Sigma_{uc}); \\ \emptyset & \text{otherwise.} \end{cases}$$

It is immediate to see that R_{uc} can be chosen as a candidate of bisimilarity enforcing supervisors because it naturally satisfies the condition (2) required in Definition 1. Further, since R is deterministic and synchronously simulated by G, we have $G \parallel R \cong R$. Moreover, language controllability of R indicates $G\|R = G\|R_{uc}$. Thus, $G\|R_{uc} = G\|R \cong R$, i.e., R_{uc} is a bisimilarity enforcing supervisor. The proposed bisimilarity enforcing supervisor has been successfully implemented on a multi-robot system cooperative tasking scenario (Karimadini & Lin, 2011).

3. Algorithm for checking the existence condition

Based on the result of Theorem 1, this section proposes an algorithm to test the existence of a bisimilarity enforcing supervisor. First, we introduce the notion of synchronously simulation-based controllable product, which will be used in the test algorithm.

Definition 3. Given a plant G and a specification R, the synchronously simulation-based controllable product of R and G is an automaton

$$R \|_{sync} G = ((Q \times X) \cup \{q_d, q'_d\}, \Sigma, \alpha_{sync}, (q_0, x_0), Q_m \times X_m),$$

where for any $(q,x) \in Q \times X$ and $\sigma \in \Sigma$, the transition function is defined as

$$\alpha_{sync}((q,x),\sigma) = \begin{cases} \delta(q,\sigma) \times \alpha(x,\sigma) & \sigma \in E_R(q) \cap E_G(x); \\ \{q_d\} & \sigma \in E_R(q) \setminus E_G(x); \\ \{q'_d\} & \sigma \in \Sigma_{uc} \cap (E_G(x) \setminus E_R(q)); \\ \emptyset & \text{otherwise.} \end{cases}$$

Then, we present the following theorem to verify the existence of a bisimilarity enforcing supervisor.

Theorem 2. *Given a plant G and a deterministic specification R, there exists a bisimilarity enforcing supervisor for G and R if and only if q_d and q'_d are not reachable in $R \|_{sync} G$ and $x \in X_m$ for any reachable state (q,x) in $R \|_{sync} G$ with $q \in Q_m$.*

Proof. Theorem 1 and the result of Zhou and Kumar (2011) imply that there exists a bisimilarity enforcing supervisor iff R satisfies: (1) there exists a simulation relation ϕ s.t. $(q,x) \in \phi$ for any $q \in Q$ and $x \in X_{synRG}(q)$, and (2) $L(R)$ is language controllable. It is obvious that any (q,x) satisfying $x \in X_{synRG}(q)$ is a state reachable in $R \|_{sync} G$, and any $(q,x) \notin \{q_d, q'_d\} \times X$ satisfies $x \in X_{synRG}(q)$. If (1) is violated, we have two cases. Case 1: there exist (q,x) and $\sigma \in \Sigma$ s.t. $x \in X_{synRG}(q)$ and $\sigma \in E_R(q) \setminus E_G(x)$. So $q_d \in \alpha_{sync}((q,x),\sigma)$. Case 2: there is (q,x) such that $x \in X_{synRG}(q)$, $x \notin X_m$ and $q \in Q_m$. If (2) is violated, there exist (q,x) and $\sigma \in \Sigma_{uc}$ s.t. $x \in X_{synRG}(q)$ and $\sigma \in E_G(x) \setminus E_R(q)$. So $q'_d \in \alpha_{sync}((q,x),\sigma)$. It follows that q_d and q'_d are reachable in $R \|_{sync} G$ or $x \notin X_m$ for any reachable state (q,x) in $R \|_{sync} G$ with $q \in Q_m$ iff there does not exist a bisimilarity enforcing supervisor for G and R.

Next, we discuss the complexity of Theorem 2. Since G is nondeterministic and R is deterministic, their numbers of transitions are $O(|X|^2|\Sigma|)$ and $O(|Q||\Sigma|)$ respectively. Then, the complexity of constructing $R \|_{sync} G$ is $O(|X|^2|Q|^2|\Sigma|)$. So the complexity of Theorem 2 is $O(|X|^2|Q|^2|\Sigma|)$. On the other hand, the complexity of verifying the existence condition ($G \parallel R \cong R$ and language controllability of R) in Zhou and Kumar (2011) is $O(|X|^2|Q|^2|\Sigma| \log(|X||Q|))$. Hence, the method proposed in this paper is more effective.

4. Supremal feasible sub-specifications

Since a given specification does not always satisfy the conditions like synchronous simulation and language controllability, a natural question arises: how to find a maximally permissive specification, for which there exists a bisimilarity enforcing supervisor? To answer this question, the synthesis of supremal feasible subspecifications is studied in this section. We start by introducing the notion of supremum.

Given two sets A and $A' \subseteq A$ and a transitive and reflexive relation $\preceq \subseteq A \times A$ over A, $x \in A$ is said to be a supremum of A', denoted by $supA'$, if

(1) $\forall y \in A'$: $y \preceq x$;
(2) $\forall z \in A : [\forall y \in A' : y \preceq z] \Rightarrow [x \preceq z]$.

When we define the supremum of A', a pair of $(A, \preceq)$ should be given according to the elements of A'. If the elements of A' are languages, then $(2^{\Sigma^*}, \subseteq)$ should be applied since 2^{Σ^*} includes all the languages over alphabet Σ and language inclusion fully captures the comparison between two languages. However, if the elements of A' are automata, we should apply $(B, \prec)$, where B is a full set of automata with alphabet Σ and $\prec \subseteq B \times B$ is a simulation relation. This is because language inclusion is not adequate for automata (possibly nondeterministic) comparison, whereas a finer simulation relation is needed. Please note that the supremum defined over $(2^{\Sigma^*}, \subseteq)$ is unique. However, such uniqueness does not hold with respect to $(B, \prec)$ because $A_1 \prec A_2$ and $A_2 \prec A_1$ do not imply $A_1 = A_2$.

The class of feasible sub-specifications which guarantee the existence of a bisimilarity enforcing supervisor is described as follows:

$$C_1 := \{R' \mid R' \text{ is deterministic, } R' \prec R,\ R' \prec_{syn} G \text{ and } L(R) \text{ is language controllable w.r.t. } L(G) \text{ and } \Sigma_{uc}\}.$$

It is difficult to directly calculate the supremum of C_1 because C_1 is not closed under the upper bound (join) operator over $(B, \prec)$ (Zhou & Kumar, 2011). Thus, we would like to convert the automaton set C_1 into equivalently expressed language sets which are closed under the upper bound (set union) operator over $(2^{\Sigma^*}, \subseteq)$ (Cassandras, 2008). The conversion can be done item by item. First, for two deterministic automata R' and R, the condition $R' \prec R$ is equivalent to the language conditions $L(R') \subseteq L(R)$ and $L_m(R') \subseteq L_m(R)$. Second, language controllability is naturally a language description. It remains to convert $R' \prec_{syn} G$ to an equivalent language condition. The following concept is in need to complete the conversion.

Definition 4. Given $G = (X, \Sigma, x_0, \alpha, X_m)$, the synchronous state merger operator on G is an automaton

$$F_{syn}(G) = (X_{syn}, \Sigma, \{x_0\}, \alpha_{syn}, X_{msyn}),$$

where $X_{syn} = 2^X$, $X_{msyn} = \{Y_1 \mid Y_1 \subseteq X_m\}$, and for any $A \in X_{syn}$ and $\sigma \in \Sigma$, the transition function is defined as

$$\alpha_{syn}(A,\sigma) = \begin{cases} \bigcup\limits_{x \in A} \alpha(x,\sigma) & \sigma \in \bigcap\limits_{x \in A} E_G(x); \\ \text{undefined} & \text{otherwise.} \end{cases}$$

Intuitively, the construction of $F_{syn}(G)$ starts with the initial state $\{x_0\}$. It then identifies the event that is defined in the active event set of any state in the set $\{x_0\}$. A collection of this kind of events is the active event set of $\{x_0\}$. For each event σ in this active event set, the initial state $\{x_0\}$ can transit to a new state A, where A consists of all the states in X that are reached starting from a state in $\{x_0\}$. Then, the process continues from the updated state A. By using $F_{syn}(G)$, the requirement $G_1 \prec_{syn} G$ is equivalent to language conditions $L(G_1) \subseteq L(F_{syn}(G))$ and $L_m(G_1) \subseteq L_m(F_{syn}(G))$, which can be described as the following proposition.

Proposition 1. *Given a plant G and a deterministic automaton G_1, there exists a simulation relation ϕ such that $G_1 \prec_{syn\phi} G$ iff $L(G_1) \subseteq L(F_{syn}(G))$ and $L_m(G_1) \subseteq L_m(F_{syn}(G))$.*

Proof. Let $F_{syn}(G) = (X_f, \Sigma, \{x_0\}, \alpha_f, X_{mf})$ and $G_1 = (X_1, \Sigma, x_{01}, \alpha_1, X_{m1})$. For sufficiency, consider $\phi_1 = \{(x_1, x) \in X_1 \times X \mid x \in X_{synG_1G}(x_1)\}$. Obviously, $G_1 \prec_{syn\phi_1} G$. For necessity, we can use the induction method to prove $L(G_1) \subseteq L(F_{syn}(G))$. Further, we would like to prove $L_m(G_1) \subseteq L_m(F_{syn}(G))$. For any $s' \in L_m(G_1)$, there is $x_1 \in \alpha_1(x_{01}, s')$ such that $x_1 \in X_{m1}$. Because $G_1 \prec_{syn\phi} G$, we have $x' \in X_m$ for any $x' \in \alpha(x_0, s')$. It implies $s' \in L_m(F_{syn}(G))$. Thus, $L_m(G_1) \subseteq L_m(F_{syn}(G))$.

Hence, we can convert the automaton set C_1 into the following language sets.

$$C_2 := \{L_1 \subseteq L(R) \cap L(F_{syn}(G)) \mid L_1 = \overline{L_1} \text{ and } L_1 \text{ is language controllable w.r.t. } L(G) \text{ and } \Sigma_{uc}\};$$

$$C_3 := \{L_1 \cap L_m(R) \cap L_m(F_{syn}(G)) \mid L_1 \in C_2\}.$$

The computation of $supC_1$ over $(B, \prec)$ is then achieved through the computation of the supremal languages of C_2 and C_3 over $(2^{\Sigma^*}, \subseteq)$, which is stated in the following proposition. For two languages $K_1, K_2 \in \Sigma^*$, where K_1 is nonempty and prefix closed and $K_2 \subseteq K_1$, let $G_{(K_1,K_2)}$ be a deterministic automaton such that $L(G_{(K_1,K_2)}) = K_1$ and $L_m(G_{(K_1,K_2)}) = K_2$.

Proposition 2. *Given a plant G and a deterministic specification R, the automaton $G_{(supC_2,supC_3)} \in supC_1$ iff $supC_2 \neq \emptyset$.*

Proof. The necessity holds due to the existence of $G_{(supC_2,supC_3)}$. For sufficiency, let $L_1 = supC_2 \neq \emptyset$ and $L_1' = supC_2 \cap L_m(R) \cap L_m(F_{syn}(G)) = supC_3$. We first show that $G_{(L_1,L_1')} \in C_1$. Since $L_1 = supC_2$, we have $L_1 \in C_2$, which implies L_1 is language controllable and $L_1 \subseteq L(F_{syn}(G))$. Further, $L_1' \subseteq L_m(F_{syn}(G))$. Proposition 1 implies $G_{(L_1,L_1')} \prec_{syn} G$. Since R and $G_{(L_1,L_1')}$ are deterministic and $L_1 \in C_2$ implies $L_1 \subseteq L(R)$ and $L_1' \subseteq L_m(R)$, we have $G_{(L_1,L_1')} \prec R$. Therefore, $G_{(L_1,L_1')} \in C_1$. Next, we show that $R_1 \prec G_{(L_1,L_1')}$ for any $R_1 \in C_1$. Suppose there is $R_1 \in C_1$ such that $R_1 \not\prec G_{(L_1,L_1')}$. Since $R_1 \in C_1$, it implies $R_1 \prec R$. Moreover, R_1 and R are deterministic. It follows that $L(R_1) \subseteq L(R)$ and $L_m(R_1) \subseteq L_m(R)$. In addition, $R_1 \in C_1$ implies $L(R_1)$ is language controllable and $R_1 \prec_{syn} G$, which implies $L(R_1) \subseteq L(F_{syn}(G))$ and $L_m(R_1) \subseteq L_m(F_{syn}(G))$ by Proposition 1. Hence, $L(R_1) \in C_2$. Moreover, $L_m(R_1) \subseteq L(R_1)$. Then, $L(R_1) \subseteq supC_2 = L_1$ and $L_m(R_1) \subseteq supC_3 = L_1'$. Further, R_1 and $G_{(L_1,L_1')}$ are deterministic. Thus, $R_1 \prec G_{(L_1,L_1')}$, which introduces a contradiction. Hence, the assumption is not correct. Therefore, $R_1 \prec G_{(L_1,L_1')}$ for any $R_1 \in C_1$, i.e., $G_{(L_1,L_1')} = G_{(supC_2,supC_3)} \in supC_1$.

Next, we present a recursive algorithm method to compute the supremal feasible sub-specification. For an automaton $G' = (X', \Sigma, x_0', \alpha', X_m')$ and a state set $X_1 \subseteq X'$ with $x_0' \in X_1$, the sub-automaton of G' with respect to X_1 is defined as $F_{G'}(X_1) = (X_1, \Sigma, x_0', \alpha_1, X_{m1})$, where $\alpha_1 = \alpha'|_{X_1 \times \Sigma}$ and $X_{m1} = X_1 \cap X_m'$.

Algorithm 1. Given a plant G and a deterministic specification R, the algorithm for computing the supremal feasible sub-specification with respect to G and Σ_{uc} is described as follows:

Step 1: obtain $det(G) = (X_{det}, \Sigma, x_{0det}, \alpha_{det}, X_{mdet})$, $G' = (F_{syn}(G) \parallel R)_{uc} = (X', \Sigma, x_0', \alpha', X_m')$ and $G'' = G' \parallel det(G) = (X'', \Sigma, x_0'', \alpha'', X_m'')$;

Step 2: $Z_0 := \{(x_1', x_2) \in X' \times X_{det} \mid x_1' = D\}$;

Step 3: $\forall k \geq 0, Z_{k+1} = Z_k \cup \{z \in X'' - Z_k \mid (\exists \sigma \in \Sigma_{uc})\, \alpha''(z, \sigma) \in Z_k\}$;

Step 4: if there exists $k \in \mathcal{N}$ such that $Z_{k+1} = Z_k \neq X''$ and $x_0'' \notin Z_k$, then $F_{G''}(X'' - Z_k)$ of G'' is the supremal feasible sub-specification with respect to G and Σ_{uc}.

The correctness of Algorithm 1 is obvious according to Proposition 2. Because the state set X'' is finite and the state numbers of $F_{syn}(G)$ and $det(G)$ are both $O(2^{|X|})$, Algorithm 1 is terminated with complexity $O(2^{2|X|}|Q||\Sigma|)$. Moreover, a formula-based method induced by Propositions 1 and 2 is presented to compute supremal feasible sub-specifications.

Proposition 3. *Given a plant G, a deterministic specification R and a language $M = L(R) \cap L(F_{syn}(G)) - [(L(G) - L(R) \cap L(F_{syn}(G)))/\Sigma_{uc}^*]\,\Sigma^*$, the automaton $G_{(M,M')}$ is the supremal feasible sub-specification w.r.t. G and Σ_{uc} iff $M \neq \emptyset$, where $M' = M \cap L_m(R) \cap L_m(F_{syn}(G))$.*

5. Conclusion

In this paper, we investigated bisimilarity enforcing supervisory control of nondeterministic plants for deterministic specifications. A necessary and sufficient condition for the existence of a bisimilarity enforcing supervisor was deduced from synchronous simulation and language controllability of the specification, which can be verified by a polynomial algorithm. For those specifications fulfilling the existence condition, a bisimilarity enforcing supervisor has been constructed. In addition, when the existence condition does not hold, a recursive method and a formula-based method have been developed to calculate supremal feasible sub-specifications.

References

Belta, C., Bicchi, A., Egerstedt, M., Frazzoli, E., Klavins, E., & Pappas, G. (2007). Symbolic planning and control of robot motion (grand challenges of robotics). *IEEE Robotics and Automation Magazine*, *14*, 61–70.

Cassandras, C. (2008). *Introduction to discrete event systems*. Spinger.

Karimadini, M., & Lin, H. (2011). Guaranteed global performance through local coordinations. *Automatica*, *47*, 890–898.

Milner, R. (1989). *Communication and concurrency*. Prentice-Hall.

Pappas, G., & Tabuada, P. (2006). Linear time logic control of discrete-time linear systems. *IEEE Transactions on Automatic Control*, *51*, 1862–1877.

Park, D. (1981). *Concurrency and automata on infinite sequences*. Spinger.

Qin, H., & Lewis, P. (1990). Factorization of finite state machines under observational equivalence. In *CONCUR'90: theories of concurrency–unification and extension* (pp. 427–441).

Tabuada, P. (2008). Controller synthesis for bisimulation equivalence. *Systems & Control Letters*, *57*, 443–452.

Zhou, C., & Kumar, R. (2011). Bisimilarity enforcement for discrete event systems using deterministic control. *IEEE Transactions on Automatic Control*, *56*, 2986–2991.

Zhou, C., Kumar, R., & Jiang, S. (2006). Control of nondeterministic discrete-event systems for bisimulation equivalence. *IEEE Transactions on Automatic Control*, *51*, 754–765.