Controller Synthesis for Bisimulation Equivalence

Sun Yajuan

B.S., Beijing University of Aeronautics and Astronautics, China

A THESIS SUBMITTED

FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

NATIONAL UNIVERSITY OF SINGAPORE

2013

Acknowledgments

I am grateful to many people for supporting me not only intellectually but also mentally and socially in my work and life besides work. These acknowledgements can only give a glimpse on how much I benefited and learned from all my mentors, colleagues, friends and family. Thank you so much to all of you.

First of all, I wish to sincerely thank my supervisors Assist. Prof. Hai Lin and Prof. Ben M. Chen, who supplied me with invaluable advice and guidance throughout my time at the university concerning my research, writing, organization and life. Their insights in symbolic control are always stimulating, and many chapters of this thesis were shaped by the numerous discussions.

I am also highly appreciate Prof. Qing-Guo Wang and Prof. Kai-Yew Lum for agreeing to be my comprehensive and oral qualifying exam committee; all lecturers in ECE Department and former teachers who have built my academic background, and all laboratory officers for their kindly supports. I would also like to express my gratitude to my colleagues for their inspirational inputs and my friends for their true friendship.

Last but not least, I am forever grateful to my loving parents, grandparents and

husband Geng. This thesis would not have been possible without their encouragement and love.

Contents

Summary

This thesis investigates the controller synthesis for bisimulation equivalence from both discrete and continuous aspects. From discrete perspective, supervisory control is studied to enforce bisimilarity with respect to discrete event systems. Specifically, three kinds of supervisory control problems are considered here: bisimilarity control, decentralized bisimilarity control and distributed bisimilarity control.

First, we investigate bisimilarity supervisory control, which aims to design a supervisor so that the supervised system is bisimilar to the specification. This thesis considers the most general case which allows the plant, specification and supervisor to be nondeterministic. The challenge is a lack of systematic way for the construction of bisimilarity enforcing supervisors. For this issue, this thesis introduces the notion of synchronous simulation-based state controllability as the existence condition for bisimilarity control. It is shown that a bisimilarity enforcing supervisor can be efficiently built upon the specification when the existence condition holds. Another important question that arises is how to find achievable sub-specifications when the existence condition does not hold. To answer this question, the synthesis of synchronously simulation-based state controllable sub-specifications is studied. Since

the existence condition for the most general case is sufficient only, we specialize to deterministic specifications. A necessary and sufficient condition is then provided for bisimilarity control with respect to deterministic specifications. In addition, two methods are presented to calculate maximal permissive sub-specifications.

Second, we study decentralized bisimilarity supervisory control, where a set of local supervisors jointly control the given plant to achieve the specification. Unlike language-based structure, a novel automata-based structure is proposed, where the plant, specification and supervisor are all modeled as automata. In particular, three architectures, a conjunctive architecture, a disjunctive architecture and a general architecture, are developed with respect to different decision making rules. Under these three architectures, necessary and sufficient conditions are respectively provided for the existence of a deterministic decentralized bisimilarity control. Furthermore, the synthesis of decentralized bisimilarity supervisors and achievable sup-specifications are investigated.

Third, bisimilarity supervisory control is extended to deal with distributed discrete event systems which consist of multiple interacting local modules. The objective of distributed bisimilarity control is to impose bisimulation equivalence between the globally supervised system (the parallel composition of locally supervised modules) and the specification. The concept of separable and synchronous simulation-based state controllability is introduced as the existence condition for distributed bisimilarity control. When this condition is satisfied, a set of local supervisors can be constructed to enforce bisimulation equivalence. Otherwise, the computation of achievable sub-specifications is explored to enable the existence of a distributed bisimilarity

control. In addition, we focus on deterministic supervisors for distributed bisimilarity control. The synthesis of deterministic supervisors and achievable sup-specifications are investigated, accordingly. The comparisons of our results with the centralized monolithic ones are further presented.

When it comes to continuous perspective, the control of multi-affine systems for bisimulation equivalence is presented, with its application to meet temporal logic specifications. The key is to guarantee the existence of a bisimilarly abstracted system with finite state nature for the original continuous system. However, this problem is generally undecidable. For this reason, we partition the state space into rectangles, and then study the control of multi-affine system on rectangles. Resorting to the proposed control method, a bisimilarly abstracted system is obtained. A fully automated procedure is then developed to control multi-affine systems for temporal logic specifications.

List of Figures

Chapter 1

Introduction

1.1 Motivation

Nowadays, high-level and human-like specifications, such as sequencing tasks, system synchronization and network adaptability, naturally emerge in many modern systems like software control systems [1–3], automotive industries [4–6], multi-robot systems [7–9] and biological systems [10–12], which falls beyond the traditional control tasks like stabilization and output regulation. As an expressive mechanism to describe sophisticated specifications, temporal logic has been widely adopted to formally specify the desired behavior [8, 13–15]. There are two main reasons for the usage of temporal logic. Firstly, temporal logic makes it possible to form complicated specifications in a succinct and unambiguous manner. Secondly, temporal logic is similar to natural languages and can be easily interpreted by human operators [16]. This shift in specification perspective is accompanied by a change in control methods.

A current trend in control is to use formal methods, like model checking and su-

pervisory control, to generate a symbolic path on an abstracted quotient system to satisfy more complicated temporal logic specifications [17–19]. The key is to guarantee the existence of a feasible continuous path corresponding to the synthesized symbolic sequence that satisfies the temporal logic specification. For this purpose, a bisimulation equivalence between the abstracted finite state quotient system and the continuous dynamics needs to be obtained. Additionally, a supervisor designed for the abstracted quotient system needs to enforce bisimilarity with respect to the temporal logic specifications. These bring new challenges to control society. Firstly, it is necessary to investigate the supervisor synthesis for bisimulation equivalence, since most existing results on supervisor control of discrete event systems are based on language enforcement. It is known that language equivalence cannot imply bismulation. Secondly, abstraction is still an open problem. Only a small portion of systems with simple continuous dynamics has bisimilarly abstracted finite state systems. This thesis aims to investigate the controller synthesis for bisimulation from both discrete and continuous aspects. It may provide a promising way to address above mentioned challenges. Next, we will give a brief literature review of the existing methods for supervisory control of discrete event systems and abstraction of continuous systems in Section 1.2 and Section 1.3, respectively.

1.2 Supervisory Control of Discrete Event Systems

In this thesis, we are especially interested in two types of systems: continuous and discrete event systems. While continuous systems are time-driven, represented by

differential equations and modeled by state space equations; discrete event systems are event-driven, represented by the sequence of states/events and modeled by automata. To unify continuous and discrete event systems, the notion of transition systems is employed, which is shown as below.

Definition 1.1 *A transition system is a tuple* $TS = (S, E, \beta, s_0, S_m)$*, where* S *is a (possible infinite) set of states,* E *is a (possible infinite) set of events,* $\beta : S \times E \to S$ *is a transition function,* s_0 *is an initial state and* $S_m \subseteq S$ *is a (possible infinite) set of marked states. A transition system* $TS = (S, E, \beta, s_0, S_m)$ *is called to be nondeterministic if the transition function is in the form of* $\beta : S \times E \to 2^S$*.*

It can be seen that a transition system is a graph, possibly with an infinite number of states or transitions. A special form of transition system is automaton which possesses finite state and event sets. Take automaton $G = (X, \Sigma, \alpha, x_0, X_m)$ as an example, its state set X, event set Σ and marked state set $X_m \subseteq X$ are finite. Let Σ^* be the set of all finite strings over Σ including the empty string ϵ. With an abuse of notation, the transition function α of G 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 $\sigma \in \Sigma$, $\alpha(x, s\sigma) = \alpha(\alpha(x, s), \sigma)$. The transition α can also be restricted to a smaller domain $X_1 \times \Sigma$, denoted by $\alpha|_{X_1 \times \Sigma}$. The active event set at state x is defined as $E_G(x) = \{\sigma \in \Sigma \mid \alpha(x, \sigma) \text{ is defined}\}$. The behavior of a DES can be described by languages. 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. The concept of parallel composition is presented to model the interaction

between DESs [20].

Definition 1.2 *Given* $G_1 = (X_1, \Sigma_1, \alpha_1, x_{01}, X_{m1})$ *and* $G_2 = (X_2, \Sigma_2, \alpha_2, x_{02}, X_{m2})$, *the parallel composition of* G_1 *and* G_2 *is an automaton*

$$G_1||G_2 = (X_1 \times X_2, \Sigma_1 \cup \Sigma_2, \alpha_{1||2}, (x_{01}, x_{02}), X_{m1} \times X_{m2}),$$

where for any $x_1 \in X_1$, $x_2 \in X_2$ *and* $\sigma \in \Sigma$, *the transition function is defined as:*

$$\alpha_{1||2}((x_1, x_2), \sigma) = \begin{cases} \alpha_1(x_1, \sigma) \times \alpha_2(x_2, \sigma) & \sigma \in E_{G_1}(x_1) \cap E_{G_2}(x_2); \\ \alpha_1(x_1, \sigma) \times \{x_2\} & \sigma \in E_{G_1}(x_1) \wedge \sigma \in E_1 \setminus E_2; \\ \{x_1\} \times \alpha_2(x_2, \sigma) & \sigma \in E_{G_2}(x_2) \wedge \sigma \in E_2 \setminus E_1; \\ \emptyset & \textit{otherwise.} \end{cases}$$

1.2.1 Supervisory Control

The supervisory control of DESs aims to design supervisors so that the supervised system meets the specification. In particular, the event set Σ is partitioned into a controllable event set Σ_c and an uncontrollable event set Σ_{uc} such that $\Sigma = \Sigma_c \cup \Sigma_{uc}$. The supervisor disables certain controllable events to ensure the satisfaction of the desired behavior. This form of control is captured by parallel composition with $\Sigma_1 = \Sigma_2$.

The earliest work on supervisory control was traced back to [21], in which a supervisor was developed such that the supervised system achieves the language specification. The initial work has been then extended to a variety of supervisory control approaches including control under full observation [22] or partial observation [23–26], nonblocking control [27–30], modular control [31–33], hierarchical control [34–36],

decentralized control [37, 38] and distributed control [39, 40]. We will review the last two methods in Subsection 1.2.2 and Subsection 1.2.3 respectively. In fact, most of the existing literature focused on langauge equivalence. However, language equivalence is not adequate to capture the class of temporal logics which describe branching behavior, such as CTL and CTL*. This requires us to use bisimulation equivalence instead. In particular, the notion of bisimulation is stated as below [41].

Definition 1.3 *Consider transition systems* $TS_1 = (S_1, E, \beta_1, s_{01}, S_{m1})$ *and* $TS_2 = (S_2, E, \beta_2, s_{02}, S_{m2})$. *The relation* $\phi \subseteq S_1 \times S_2$ *is a simulation relation if for any* $(s_1, s_2) \in \phi$, *the following property holds:*

(1) $(\forall e \in E)\ s_1' \in \beta_1(s_1, e) \Rightarrow \exists s_2' \in \beta_2(s_2, e)$ *such that* $(s_1', s_2') \in \phi$;

(2) $s_1 \in S_{m1} \Rightarrow s_2 \in S_{m2}$.

If there is a simulation relation $\phi \subseteq S_1 \times S_2$ such that $(s_{01}, s_{02}) \in \phi$, then TS_1 is said to be simulated by TS_2, denoted as $TS_1 \prec_\phi TS_2$. A binary relation $\phi \subseteq (S_1 \cup S_2)^2$ is called a bisimulation relation between TS_1 and TS_2, if $TS_1 \prec_\phi TS_2$, $TS_2 \prec_\phi TS_1$ and ϕ is symmetric. Further, TS_1 is said to be bisimilar (bisimulation equivalent) to TS_2, denoted as $TS_1 \cong_\phi TS_2$, if there is a bisimulation relation $\phi \subseteq (S_1 \cup S_2)^2$ such that $(s_{01}, s_{02}) \in \phi$.

It is known that bisimulation equivalence implies language equivalence and marked language equivalence, but the converse does not hold. We sometimes omit the subscript ϕ from $\prec_\phi$ or $\cong_\phi$ when it is clear from the context.

Recent years have seen research attentions on bisimilarity enforcing supervisory control. An early effort on supervisory control for bisimulation equivalence can be

found in [42], where events are treated as controllable and a supervisor was developed such that the supervised system is bisimilar to the deterministic specification. [43] studied bisimilarity supervisory control of open discrete event systems. It requires that the indistinguishable events are either all enabled or all disabled at a state, which is not reasonable in the framework of supervisory control. [44] solved the bisimilarity controller synthesis problem for various systems including continuous systems, hybrid systems and DESs, in which the bisimilarity controller is the morphism in the context of category theory. Zhou and Kumar [45] investigated bisimilarity control of DESs, where the plant, specification and supervisor are allowed to be nondeterministic. They provided a small model theorem to show that a supervisor exists to enforce bisimulation equivalence between the supervised system and the specification if and only if a state controllable automaton exists over the Cartesian product of the system and specification state spaces. The small model theorem was also extended for partial observation [46]. In both [45] and [46], the complexity of checking the existence condition of bisimilarity control is doubly exponential. To reduce the computational complexity, [47] focused on deterministic bisimilarity control.

However, apart from promising progress in bisimilarity control, three main difficulties remain to be addressed for practial applications. Firstly, for the most general case, where all the plant, specification and supervisor are nondeterministic, there does not exist a systematic way to construct the bisimilarity enforcing supervisor when it exists [45, 46]. In Chapter 2, a formal supervisor design method is proposed to solve this problem. Secondly, the computational complexity is high in the existing methods (double exponential complexity in state sizes of the plant and specification [45]

and single exponential complexity in state sizes of the plant and specification [48]). To mitigate the complexity, a novel notion is introduced as the existence condition for bisimilarity control in Chapter 2. It can be effectively verified with polynomial complexity in the state sizes of the plant and specification. Thirdly, an important issue missing in the literature is how to change the specification when there does not exist such a bisimilarity supervisor for the original specification. For this issue, the calculation of achievable sub-specifications is investigated in Chapter 2.

1.2.2 Decentralized Supervisory Control

The bisimilarity control proposed in Subsection 1.2.1 only relies on a single supervisor. However, no monolithic supervisor is likely to provide a useful solution in many engineering systems, such as an automated manufacturing systems which consists of several workstations interconnected by conveyors or automated guided vehicles. Therefore, a decentralized solution is needed. In decentralized supervisory control, a set of local supervisors (more than one) should jointly control a given plant for a global behavior. Each supervisor takes its actions based on its own observation. The control actions of the local supervisors are fused into a global control decision to control the given plant. Several architectures have been developed for decentralized supervisory control according to different fusion rules.

Rudie and Wonham [37] proposed a conjunctive architecture, where the fusion rule is based on the intersection of locally enabled events. Complementary with conjunctive architecture, a disjunctive architecture was presented in [49], where the union

of locally enabled events is adopted in fusion. In that work, a general architecture which combines the conjunctive architecture and the disjunctive architecture was further developed. Namely, the local supervisors agree a priori on choosing "fusion by intersection" for certain controllable events and "fusion by union" for certain controllable events. In these works, supervisors make unconditional decisions: "enable" or "disable". Notable exceptions were reported in [50] and [51]. [50] considered the conditional decisions of the form: "enable if nobody disables" and "disable if nobody enables". [51] provided a knowledge-based architecture to associate the supervisors' decision to a grade or level of ambiguity. Based on these architectures, recent efforts on decentralized supervisory control has devoted to hierarchical control [52], reliable control [53], [54] and communicating control with communication delays [55] or without communication delays [56], [57]. The existing literature all employed language equivalence. It is known that bisimulation is necessary to deal with branching behaviors that arise in unmodeled dynamics, model abstraction and communication delays. Moreover, bisimulation is a natural choice for temporal logic specifications. These observations motivate us to study decentralized supervisory control for bisimulation equivalence in Chapter 3.

1.2.3 Distributed Supervisory Control

A distributed DES (also called a concurrent DES) is composed of several local DESs that cooperatively perform a task or computation. A multi-agent system [58–60] is an example of distributed discrete event systems. The goal of distributed supervisory

control is to synthesize local supervisors for individual plants such that the resulting supervised behavior is identical with the global specification. Each local supervisor determines its control actions based on the locally observed behavior.

The current methods for distributed supervisor control typically involve converting the distributed DES to equivalent monolithic system [39, 61, 62]. Unfortunately, in general a monolithic control design is computationally expensive. It was formally shown in [33, 63] that the monolithic-based computation grows exponentially as the number of components in plant increases. Modular control seems a solution for complexity mitigation since the computation of local supervisor only relies on the local DES rather than the global one. [64] proposed an existence condition for the modular control of distributed DESs. The same problem was explored in [65] for a special class of distributed DESs in which all local DESs exhibit isomorphic behavior. Different from these works, [66–72] investigated the conditions under which a modular control of distributed DESs is equivalent to a maximally permissive/ minimally restrictive monolithic control. Specifically, [66] assumes that the event sets of local DESs are mutually disjoint. [67] and [68] assume that the shared events are all controllable for local DESs. [70], [71] and [72] assume that the shared events have the same control status and additionally the newly introduced property of mutual controllability holds. This assumption has been generalized in [69] by only requiring the same status for the shared events. Research efforts are all devoted into language enforcement. As pointed out in the previous subsection, bisimulation equivalence is strongly required to tackle branching behaviors and temporal logic specifications. Therefore, we consider distributed supervisory control for bisimulation equivalence in Chapter 4.

1.3 Abstraction of Continuous Systems

Abstraction is a process of extracting a discrete event model from the continuous system. It is desired that the abstracted model can either be equivalent with the original continuous system with respect to the satisfaction of the specification, or it can provide approximately guarantee that the satisfaction of the specification for the abstracted model is sufficient for the satisfaction of the specification by the original system.

Equivalent abstraction relies on the notion of bisimulation. So far, only few classes of systems can be bisimilarly abstracted. The first success is timed automata [73]. Subsequent extensions led to the results for multi-rate automata [74], rectangular hybrid automata [75] and order-minimal automata [76]. Recent work for more complex continuous dynamics-second order linear dynamics can be found in [77], where a bisimilarly abstracted model can be obtained by a triangulation of polygonal state space. Their work was refined in [17] through approaching arbitrary dimensional discrete-time linear system. It was shown that an equivalent discrete transition system exists for the controllable system with properly chosen observables. As opposed to discrete-time linear systems in [17], continuous-time linear system over polytopes was studied in [78]. Recent works of constructing sufficient abstractions focused on systems with linear dynamics and polyhedral partitions [79], and systems with polynomial dynamics and partitions given by semi-algebraic sets [80]. The construction of sufficient or equivalent abstractions (if they exist) is computationally expensive. In this thesis, we propose effective polyhedral operations to check the existence of equiv-

alent abstractions with respect to a class of nonlinear systems, multi-affine systems. This novel methodology covers more classes of system than those are addressed in [81, 82], with its successful application for temporal logic specifications.

1.4 Organization of the Thesis

This thesis investigates the controller synthesis for bisimulation equivalence from discrete and continuous aspects. From discrete perspective, supervisory control is studied to enforce bisimilarity with respect to discrete event systems. Specifically, three kinds of supervisory control problems are considered here: bisimilarity control, decentralized bisimilarity control and distributed bisimilarity control. The organization of the thesis is described as follows.

Chapter 2 studies supervisory control of DESs for bisimulation equivalence. We first focus on the most general case which allows the plant, specification and supervisor to be nondeterministic. The difficulty is that there does not exist a systematic way to construct the bisimilarity enforcing supervisor when it exists. For this issue, we introduce the notion of synchronous simulation-based state controllability as the existence condition for bisimilarity control. It is shown that a bisimilarity enforcing supervisor can be efficiently built upon the specification. Another important question is how to find achievable sub/sup-specifications when the existence condition does not hold. To answer this question, the synthesis of synchronously simulation-based state controllable sub-specifications is studied.

Since the existence condition for the most general case is sufficient only, we spe-

cialize to deterministic specifications. A necessary and sufficient condition is then provided for bisimilarity control with respect to deterministic specifications. It can be effectively verified with polynomial complexity in state sizes of the plant and specification (less than the complexity of the conditions with respect to deterministic specifications [48]). When the existence condition holds, a bisimilarity enforcing supervisor is established. Otherwise, the calculation of maximal permissive sub-specification is investigated. It is a challenging problem because the supremum of an automaton set is not closed under the upper bound (join) operator [48]. This problem is solved by converting the automaton set into equivalently expressed language sets which are closed under the corresponding upper bound (set union) operator [20].

Decentralized bisimilarity control is explored in Chapter 3, where a set of local supervisors jointly control a given plant to reach a specification. Unlike language-based structure, a novel automata-based structure is proposed, where the plant, specification and supervisor are all modeled as automata. In particular, three architectures, a conjunctive architecture, a disjunctive architecture and a general architecture, are developed with respect to different decision making rules. Under these three architectures, necessary and sufficient conditions are respectively provided for the existence of a deterministic decentralized bisimilarity control. Furthermore, the synthesis of decentralized bisimilarity supervisors and achievable sup-specifications are further developed.

Chapter 4 studies bisimilarity supervisory control of distributed discrete event systems which consist of multiple interacting modules. The objective of distributed bisimilarity control is to impose bisimulation equivalence between the globally super-

vised system (the parallel composition of locally supervised modules) and the given automaton specification. The concept of separable and synchronous simulation-based state controllability is introduced as the existence condition for distributed bisimilarity control. When this condition is satisfied, a set of local supervisors can be constructed to enforce bisimulation equivalence. Otherwise, the computation of achievable sub-specifications is explored to enable the existence of a distributed bisimilarity control. In addition, we focus on deterministic supervisors. The synthesis of deterministic distributed bisimilarity supervisors and achievable sup-specifications are investigated accordingly. The comparisons of our results with the centralized monolithic ones are further presented.

Chapter 5 investigates the control of multi-affine systems for bisimulation equivalence, with its application to meet temporal logic specifications. The key is to establish a bisimilarly abstracted system with finite state nature for the original continuous system. However, this problem is generally undecidable. For this reason, we partition the state space into rectangles, and then study the control of multi-affine system on rectangles. Resorting to the proposed control method, a bisimilarly abstracted system is obtained. It is shown that the proposed method covers more classes of systems than those are addressed in [81] and [82]. Moreover, the construction of bisimilarly abstracted system does not involve complex operators such as the integration of vector fields [83], but rather polyhedral operators. A fully automated procedure to control multi-affine systems for temporal logic specifications is developed by using the proposed bisimilarity supervisory techniques to the abstracted system and then by refining the resulting supervisor to the original multi-affine system.

The thesis is finally concluded in Chapter 6 with highlighting the contributions and outlining the future works.

Chapter 2

Supervisory Control for Bisimulation Equivalence

This chapter studies bisimilarity supervisory control of DESs. First, we focus on the most general case which allows the plant, specification and supervisor to be nondeterministic. In particular, the notion of synchronous simulation-based state controllability is introduced as the sufficient condition for the existence of a bisimilarity enforcing supervisor, and a polynomial algorithm is developed to check such a condition. When the existence condition holds, a bisimilarity enforcing supervisor is constructed. Otherwise, the synthesis of achievable sub-specifications is further studied. Then, we specialize to deterministic specifications. A necessary and sufficient condition is proposed for the existence of bisimilarity control. Accordingly, the synthesis of bisimilarity enforcing supervisors and supremal achievable sub-specifications are investigated as well.

2.1 Bisimilarity Control for Nondeterministic Specifications

This section explores bisimilarity supervisory control for the most general case. The notion of bisimilarity enforcing supervisor is introduced as below.

Definition 2.1 *Given a plant* $G = (X, \Sigma, \alpha, x_0, X_m)$ *and a specification* $R = (Q, \Sigma, \delta, q_0, Q_m)$*, a supervisor* $S = (Y, \Sigma, \beta, y_0, Y_m)$ *is said to be a bisimilarity enforcing supervisor for* G *and* R *if*

(1) There is a bisimulation relation ϕ *such that* $G||S \cong_\phi R$*;*

(2) $(\forall y \in Y$ *and* $\forall \sigma \in \Sigma_{uc})$ $\beta(y, \sigma) \neq \emptyset$*.*

It is shown that a bisimilarity enforcing supervisor always enables all uncontrollable events and achieves bisimulation equivalence. 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 plant, specification and supervisor in this chapter.

2.1.1 Existence Condition

This subsection investigates the existence condition for bisimilarity control. For sufficiency, a bisimilarity enforcing supervisor is needed. In the context language enforcing control, it is known that a controllable specification itself can work as a supervisor. This motivates us to construct a bisimilarity enforcing supervisor based on the specification. Since a bisimilarity enforcing supervisor is required to satisfy two conditions (Definition 2.1), the following concept is introduced.

Definition 2.2 *Given $G_1 = (X_1, \Sigma, \alpha_1, x_{01}, X_{m1})$, the uncontrollable augmented automaton G_{1uc} of G_1 is an automaton*

$$G_{1uc} = (X_1 \cup \{D_d\}, \Sigma, \alpha_{uc}, x_{01}, X_{m1}),$$

where for any $x \in X_1 \cup \{D_d\}$ and $\sigma \in \Sigma$, the transition function is defined as

$$\alpha_{uc}(x,\sigma) = \begin{cases} \alpha_1(x,\sigma) & \sigma \in E_{G_1}(x); \\ \{D_d\} & \sigma \in \Sigma_{uc} \backslash E_{G_1}(x) \vee (x = D_d \wedge \sigma \in \Sigma_{uc}); \\ \emptyset & otherwise. \end{cases}$$

It can been seen that if we choose the uncontrollable augmented automaton R_{uc} as the bisimilarity enforcing supervisor, it naturally satisfies the second condition of Definition 2.1. Next, we investigate the properties that makes $G||R_{uc}$ bisimilar to R, i.e., the satisfaction of the first condition of Definition 2.1. Before presenting the properties, we need the following notions.

Definition 2.3 *Given $G_1 = (X_1, \Sigma_1, \alpha_1, , x_{01}, X_{m1})$ and $G_2 = (X_2, \Sigma_2, \alpha_2, , x_{02}, X_{m2})$, the synchronized state map $X_{synG_1G_2}$: $X_1 \to 2^{X_2}$ from G_1 to G_2 is defined as*

$$X_{synG_1G_2}(x_1) = \{x_2 \in X_2 \mid (\exists s \in \Sigma^*)\ x_1 \in \alpha_1(x_{01}, s) \wedge x_2 \in \alpha_2(x_{01}, s)\}.$$

The synchronized state map can be used to find the synchronized state pairs of two automata [45]. Based on this property, the concept of synchronous simulation is stated.

Definition 2.4 *Given $G_1 = (X_1, \Sigma, \alpha_1, x_{01}, X_{m1})$, $G_2 = (X_2, \Sigma, \alpha_2, x_{02}, X_{m2})$ and a simulation relation ϕ such that $G_1 \prec_\phi G_2$, ϕ is called to be a synchronous simulation relation from G_1 to G_2 if $(x_1, x_2) \in \phi$ for any $x_1 \in X_1$ and $x_2 \in X_{synG_1G_2}(x_1)$.*

If there exists a synchronous simulation relation from G_1 to G_2, G_1 is said to be synchronously simulated by G_2, denoted as $G_1 \prec_{syn\phi} G_2$.

We can see that when the specification R is synchronously simulated by the plant G, every synchronized state pair of R and G belongs to the synchronous simulation relation ϕ. Hence, $G||R \cong R$. Furthermore, if for every state of R, its active event set includes the uncontrollable events defined in the corresponding synchronized states of G, then $G||R = G||R_{uc}$, implying $G||R_{uc} \cong R$, i.e., R_{uc} is a bisimilarity enforcing supervisor for G and R. Hence we formalize the notion of synchronous simulation-based state controllability as a property of the specification for bisimilarity control.

Definition 2.5 *Given* $G_1 = (X_1, \Sigma, \alpha_1, x_{01}, X_{m1})$ *and* $G_2 = (X_2, \Sigma, \alpha_2, x_{02}, X_{m2})$, G_1 *is said to be synchronously simulation-based state controllable with respect to* G_2 *and* Σ_{uc} *if*

(1) There is a synchronous simulation relation ϕ *such that* $G_1 \prec_{syn\phi} G_2$*;*

(2) (State Controllability) $(\forall s \in \Sigma^*$ *and* $\forall \sigma \in \Sigma_{uc})$ $s\sigma \in L(G_2) \wedge x_1 \in \alpha_1(x_{01}, s) \Rightarrow \alpha_1(x_1, \sigma) \neq \emptyset$.

Then, we present the existence condition for bisimilarity control.

Theorem 2.1 *Given a plant* G *and a specification* R*, if* R *is synchronously simulation-based state controllable with respect to* G *and* Σ_{uc}*, then there is a bisimilarity enforcing supervisor for* G *and* R.

Proof: Let $G||R = (X_{GR}, \Sigma, (x_0, q_0), \alpha_{GR}, X_{mGR})$, $G||R_{uc} = (X_{GR_{uc}}, \Sigma, (x_0, q_0), \alpha_{GR_{uc}} X_{mGR_{uc}})$ and $R_{uc} = (Q_{uc}, \Sigma, q_0, \delta_{uc}\ , Q_{muc})$. Consider R_{uc} to be a supervisor and a relation $\phi_1 = \{((x, q), q) \in X_{GR_{uc}} \times Q \mid (x, q) \in X_{GR_{uc}}\}$. It is obvious that R_{uc}

satisfies the condition (2) of Definition 2.1. Next we show that ϕ_1 is a bisimulation relation from $G||R_{uc}$ to R. For any $((x,q),q) \in \phi_1$, if there is a $\sigma-$successor $(x',q') \in \alpha_{GR_{uc}}((x,q),\sigma)$, where $\sigma \in \Sigma$, the definition of product implies $q' \in \delta_{uc}(q,\sigma)$ and $q' \in Q_m$ if $(x',q') \in X_{mGR_{uc}}$. If $\sigma \notin \Sigma_{uc}$, then $q' \in \delta(q,\sigma)$ from the definition of the uncontrollable augmented automaton. If $\sigma \in \Sigma_{uc}$, because R is state controllable w.r.t. G and Σ_{uc}, we have $q' \in \delta(q,\sigma)$. Therefore, $((x',q'),q') \in \phi_1$. For any $(q,(x,q)) \in \phi_1^{-1}$, if there is a $\sigma-$successor $q' \in \delta(q,\sigma)$, where $\sigma \in \Sigma$, we have $E_R(q) \subseteq E_G(x)$ because synchronous simulation-based state controllability of R indicates that there is a synchronous simulation relation ϕ from R to G such that every synchronized state pair of R and G belongs to ϕ. Then, there is $x' \in \alpha(x,\sigma)$ s.t. $(x',q') \in \alpha_{GR_{uc}}((x,q),\sigma)$ with $(q',x') \in \phi$, which implies $(q',(x',q')) \in \phi_1^{-1}$. In addition, $(q',x') \in \phi$ implies $x' \in X_m$ if $q' \in Q_m$, that is, $(x',q') \in X_{mGR_{uc}}$. Therefore, $G||R_{uc} \cong_{\phi_1 \cup \phi_1^{-1}} R$.

Theorem 2.1 indicates that synchronous simulation-based state controllability is the sufficient condition for the existence of a bisimilarity enforcing supervisor.

2.1.2 Test for Existence Condition

This section proposes an algorithm to test whether a given specification R is synchronously simulation-based state controllable with respect to G and Σ_{uc}. Before presenting this algorithm, the notion of synchronously simulation-based controllable product is introduced as below.

Definition 2.6 *Given* $G_1 = (X_1, \Sigma, \alpha_1, x_{01}, X_{m1})$ *and* $G_2 = (X_2, \Sigma, \alpha_2, x_{02}, X_{m2})$*, the*

synchronously simulation-based controllable product of G_1 *and* G_2 *is an automaton*

$$G_1||_{sync}G_2 = ((X_1 \times X_2) \cup \{q_d, q_d'\}, \Sigma, \alpha_{12}, (x_{01}, x_{02}), X_{m1} \times X_{m2}),$$

where for any $(x_1, x_2) \in X_1 \times X_2$ *and* $\sigma \in \Sigma$*, the transition function is defined as*

$$\alpha_{12}((x_1, x_2), \sigma) = \begin{cases} \alpha_1(x_1, \sigma) \times \alpha_2(x_2, \sigma) & \sigma \in E_{G_1}(x_1) \cap E_{G_2}(x_2); \\ q_d & \sigma \in E_{G_1}(x_1) \setminus E_{G_2}(x_2); \\ q_d' & \sigma \in (E_{G_2}(x_2) \setminus E_{G_1}(x_1)) \cap \Sigma_{uc}; \\ \emptyset & otherwise. \end{cases}$$

Then, we present the following algorithm to verify synchronous simulation-based state controllability.

Algorithm 2.1 *Given a plant* G *and a specification* R*, the algorithm for testing synchronous simulation-based state controllability of* R *with respect to* G *and* Σ_{uc} *is described as below.*

Step 1: Obtain $R||_{sync}G = (X_{sync}, \Sigma, \alpha_{sync}, (q_0, x_0), X_{msync})$*;*

Step 2: R *is synchronously simulated-based state controllable with respect to* G *and* Σ_{uc} *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* $q \in Q_m$*.*

Theorem 2.2 *Algorithm 2.1 is correct.*

Proof: It can be seen that any (q, x) satisfying $x \in X_{synRG}(q)$ is a state reachable in $R||_{sync}G$ and any $(q, x) \in X_{sync} \setminus \{q_d, q_d'\}$ satisfies that $x \in X_{synRG}(q)$ according

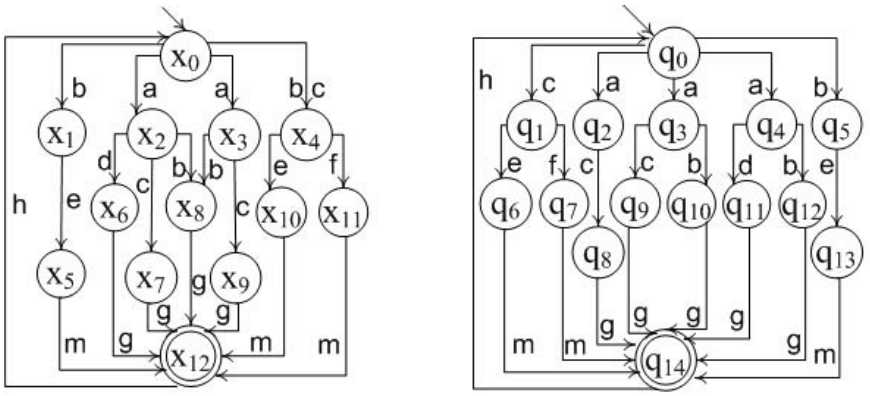

Figure 2.1: Plant G (Left) and Specification R (Right)

to the definition of synchronously simulation-based controllable product. For synchronous simulation-based state controllability to hold, condition (1) and condition (2) of Definition 2.5 should be satisfied. On the other hand, if condition (1) is violated, there are two cases. Case 1: there exist (q, x) and $\sigma \in \Sigma$ such that $x \in X_{synRG}(q)$ 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 $x \in X_{synRG}(q)$ and $x \notin X_m$ when $q \in Q_m$. If condition (2) is violated, i.e. there exist (q, x) and $\sigma \in \Sigma_{uc}$ such that $x \in X_{synRG}(q)$ 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 R is not synchronously simulated-based state controllable w.r.t. G and Σ_{uc}.

Algorithm 2.1 can be terminated because the state sets and the event sets of G and R are finite. Since G and R are nondeterministic, their numbers of transitions are $O(|X|^2|\Sigma|)$ and $O(|Q|^2|\Sigma|)$ respectively. Then, the complexity of constructing $R||_{sync}G$ is $O(|X|^2|Q|^2|\Sigma|)$. In addition, the complexity of checking the reachability of q_d and q'_d in $R||_{sync}G$ is $O(log(|X||Q|))$ [84]. So the complexity of Algorithm 2.1 is

$O(|X|^2|Q|^2|\Sigma|)$.

The following example is presented to illustrate Algorithm 2.1.

Example 2.1. Consider a plant G and a specification R configured in Fig. 2.1, where $\Sigma_{uc} = \{c\}$. It can be seen that R is not synchronously simulation-based state controllable with respect to G and Σ_{uc} because the uncontrollable event c is defined at $x_3 \in \alpha(x_0, a)$ but not $q_4 \in \delta(q_0, a)$, and d is defined at q_4 but not $x_3 \in X_{synRG}(q_4)$.

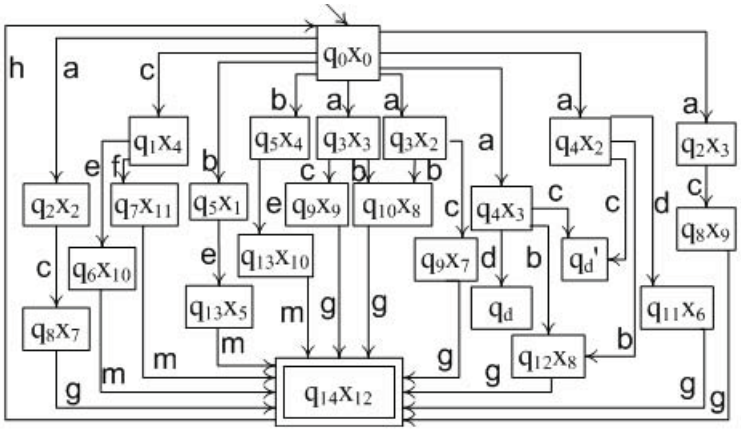

Figure 2.2: $R||_{sync}G$

Next, we apply Algorithm 2.1 to test synchronous simulation-based state controllability of R. The synchronously simulation-based controllable product $R||_{sync}G$ is obtained in Fig. 2.2 and it shows that q_d and q'_d are reachable in $R||_{sync}G$. Therefore, R is not synchronously simulation-based state controllable with respect to G and Σ_{uc}.

Remark 2.1 *Theorem 2.1 indicates that under the condition that R is synchronously simulation-based state controllable, R_{uc} is a bisimilarity enforcing supervisor, and it can be effectively constructed with polynomial complexity in the state and event sizes of the specification, i.e., $O(|Q|^2|\Sigma|)$.*

2.1.3 Synthesis of Bisimilarity Enforcing Supervisors

This section studies the synthesis of a bisimilarity enforcing supervisor when it exists. Followed by the previous results, it is natural that R_{uc} is a bisimilarity enforcing supervisor, which is shown in the following theorem.

Theorem 2.3 *Given a plant G and a deterministic specification R, if R is synchronously simulation-based state controllable with respect to G and Σ_{uc}, then R_{uc} is a bisimilarity enforcing supervisor for G and R.*

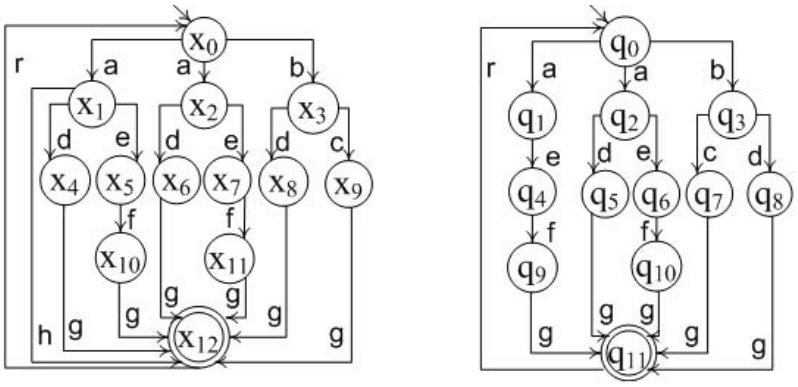

Figure 2.3: Plant G (Left) and Specification R (Right)

Now, we provide an example to illustrate bisimilarity control of discrete event systems.

Example 2.2. Consider a plant G and a specification R shown in Fig. 2.3. Assume $\Sigma_{uc} = \{e\}$. In the following, we investigate the problem whether there is a supervisor S such that the supervised system $G||S$ is bisimilar to R.

It is obvious that R is synchronously simulation-based state controllable with respect to G and Σ_{uc}. By Theorem 2.1, we can establish R_{uc} as the supervisor (Fig. 2.4

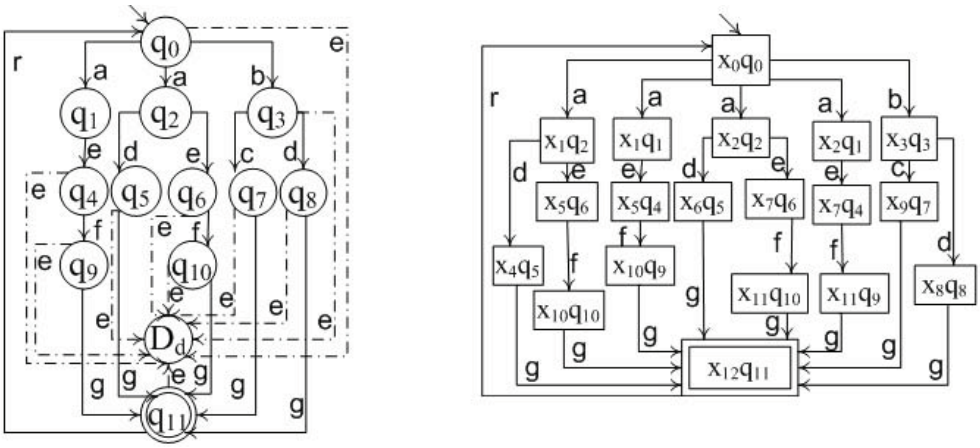

Figure 2.4: R_{uc} (Left) and $G||R_{uc}$ (Right)

(Left)). Then, the supervised system $G||R_{uc}$ is presented in Fig. 2.4 (Right). Let $\phi = \{((x_0, q_0), q_0), ((x_1, q_1), q_1), ((x_2, q_1), q_1), ((x_1, q_2), q_2), ((x_2, q_2), q_2), ((x_3, q_3), q_3), \{(x_5, q_4), (x_7, q_4), (x_5, q_6), (x_7, q_6)\} \times \{q_4, q_6\}, \{(x_4, q_5), (x_6, q_5), (x_9, q_7), (x_8, q_8), (x_{10}, q_9), (x_{11}, q_9), (x_{10}, q_{10}), (x_{11}, q_{10})\} \times \{q_5, q_7, q_8, q_9, q_{10}\}, ((x_{12}, q_{11}), q_{11})\}$. Thus, $G||R_{uc} \cong_{\phi \cup \phi^{-1}} R$.

2.1.4 Synthesis of Achievable Sub-specifications

Example 2.1 indicates that a given specification R is not always synchronously simulation-based state controllable. To guarantee the existence of a bisimilarity enforcing supervisor, this subsection aims to find synchronously simulation-based state controllable sub-specifications. Because a sub-specification is possibly nondeterministic, here we assume a sub-specification is an automaton simulated by the specification R. We start by considering synchronous simulation relation, which is required in synchronous simulation-based state controllability. Then, the following concept is introduced.

Definition 2.7 *Given a plant G, the synchronous state merger operator on G, de-*

noted by $F_{syn}(G)$*, 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*

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

In the following lemma, we show that $F_{syn}(G)$ is synchronously simulated by G.

Lemma 2.1 *Given a plant* G*, there is a synchronous simulation relation* ϕ *such that* $F_{syn}(G) \prec_{syn\phi} G$*.*

Proof: Consider a relation $\phi' = \{(A, x) \mid x \in A\}$. Next we show that ϕ' is a synchronous simulation relation from $F_{syn}(G)$ to G. The definition of synchronous state merger operator implies: (1) for any $A \in X_{syn}$ and $x \in X_{synF_{syn}(G)G}(A)$, we have $(A, x) \in \phi'$; (2) if $A \in X_{msyn}$, then $x \in X_m$ for any $x \in A$; (3) for any $(A, x) \in \phi'$, if there is a $\sigma-$successor $A' \in \alpha_{syn}(A, \sigma)$ in $F_{syn}(G)$, where $\sigma \in \Sigma$, then for any $x \in A$, there exists $x' \in \alpha(x, \sigma)$ such that $x' \in A'$. It implies $(A', x') \in \phi'$. Therefore, $F_{syn}(G) \prec_{syn\phi'} G$.

Furthermore, we illustrate that any automaton G_1 simulated by $F_{syn}(G)$ is synchronously simulated by G.

Lemma 2.2 *Given a plant* G *and an automaton* $G_1 = (X_1, \Sigma, \alpha_1, x_{01}, X_{m1})$*, if there is a simulation relation* ϕ_1 *such that* $G_1 \prec_{\phi_1} F_{syn}(G)$*, then there is a synchronous simulation relation* ϕ_2 *such that* $G_1 \prec_{syn\phi_2} G$*.*

Proof: Lemma 2.1 shows that there is a synchronous simulation relation ϕ such that $F_{syn}(G) \prec_{syn\phi} G$. Consider a relation $\phi_2' = \{(x_1, x) \in X_1 \times X \mid (\exists A \in X_{syn})(x_1, A) \in \phi_1 \wedge (A, x) \in \phi\}$, where ϕ_1 is a simulation relation from G_1 to $F_{syn}(G)$. We next show ϕ_2' is a synchronous simulation relation from G_1 to G. For any $(x_1, x) \in \phi_2$, if there is a $\sigma-$successor $x_1' \in \alpha_1(x_1, \sigma)$, where $\sigma \in \Sigma$, then we have $A' \in \alpha_{syn}(A, \sigma)$ such that $(x_1', A') \in \phi_1$. Thus there exists $x' \in \alpha(x, \sigma)$ such that $(A', x') \in \phi$. Further, if $x_1 \in X_{m1}$, then $A \in X_{msyn}$, which implies $x \in X_m$. Therefore $G_1 \prec_{\phi_2'} G$. Since $G_1 \prec_{\phi_1} F_{syn}(G)$, we have $L(G_1) \subseteq L(F_{syn}(G))$, moreover, $F_{syn}(G)$ is deterministic. It follows that for any $s \in L(G_1)$ with $x_1 \in \alpha(x_{01}, s)$, there is $A \in \alpha_{syn}(\{x_0\}, s)$ such that $(x_1, A) \in \phi_1$. Thus $(x_1, A) \in \phi_1$ for any $x_1 \in X_1$ and $A \in X_{synG_1F_{syn}(G)}(x_1)$. In addition, $F_{syn}(G) \prec_{syn\phi} G$ implies $(A, x) \in \phi$ for any $A \in X_{syn}$ and $x \in X_{synF_{syn}(G)G}(A)$. Hence $(x_1, x) \in \phi_2'$ for any $x_1 \in X_1$ and $x \in X_{synG_1G}(x_1)$, implying $G_1 \prec_{syn\phi_2'} G$.

The definition of parallel composition indicates $F_{syn}(G)||R \prec F_{syn}(G)$ and $F_{syn}(G)||R \prec R$. Then, $F_{syn}(G)||R$ is a sub-specification, moreover, Lemma 2.2 implies that $F_{syn}(G)||R$ is synchronously simulated by the plant G. If $F_{syn}(G)||R$ is further state controllable, then it is synchronously simulation-based state controllable. Next, we introduce the state controllable operator to enforce state controllability with respect to $F_{syn}(G)||R$. Before presenting this operator, we need the following concepts.

Definition 2.8 *Consider* $G_1 = (X_1, \Sigma, \alpha_1, x_{01}, X_{m1}), G_2 = (X_2, \Sigma, \alpha_2, x_{02}, X_{m2})$ *and* $G_1||_{sync}G_2 = (X_{||}, \Sigma, \alpha_{||}, (x_{01}, x_{02}),\ X_{m||})$. *The uncontrollable set of* $G_1||_{sync}G_2$ *is defined as* $X_{uc}(G_1||_{sync}G_2) = \{(x_1, s) \in X_1 \times \Sigma^* |$ *there exist* $\sigma \in \Sigma_{uc}$ *and* $x_2 \in X_2$ *s.t.* $q_d' \in \alpha_{||}((x_1, x_2), \sigma)$ *and* $(x_1, x_2) \in \alpha_{||}((x_{01}, x_{02}), s)\}$. *The post uncontrol-*

lable map $P_{G_1||_{sync}G_2} : X_1 \times \Sigma \to 2^{X_1}$ *of* $G_1||_{sync}G_2$ *is defined as* $P_{G_1||_{sync}G_2}(x,\sigma) = \{x_1 \mid$ *there exists* $(x_1, s) \in X_{uc}(G_1||_{sync}G_2)$ *s.t.* $\sigma = s(|s|)$, $x_1 \in \alpha_1(x,\sigma)$ *and* $x \in \alpha_1(x_{01}, s(1)\cdots s(|s|-1))\}$.

Intuitively, every element in $X_{uc}(G_1||_{sync}G_2)$ is of the form (x, s), where x is a state of G_1 that violates state controllability and s is a string through which x can be reached by the initial state. Further, $P_{G_1||_{sync}G_2}(x,\sigma)$ contains the states of G_1 that can transit to the states which fail to satisfy state controllability through the related string s. Then, the state controllable operator is introduced.

Definition 2.9 *Given* $G_1 = (X_1, \Sigma, \alpha_1, x_{01}, X_{m1})$, $G_2 = (X_2, \Sigma, \alpha_2, x_{02}, X_{m2})$ *and* $G_1||_{sync}G_2$, *the state controllable operator on* G_1 *with respect to* G_2, *denoted by* $F_{cG_2}(G_1)$, *is an automaton*

$$F_{cG_2}(G_1) = (X_1, \Sigma, \alpha_{cG_2}, x_{01}, X_{m1}),$$

where for any $x \in X_1$ *and* $\sigma \in \Sigma$, *the transition function is defined as*

$$\alpha_{cG_2}(x,\sigma) = \begin{cases} \alpha_1(x,\sigma) \setminus P_{G_1||_{sync}G_2}(x,\sigma) & P_{G_1||_{sync}G_2}(x,\sigma) \neq \emptyset; \\ \alpha_1(x,\sigma) & otherwise. \end{cases}$$

Then, by iteratively using the state controllable operator on G_1, we can achieve an automaton which is state controllable with respect to G_2 and Σ_{uc}.

Lemma 2.3 *Given* $G_1 = (X_1, \Sigma, \alpha_1, x_{01}, X_{m1})$, $G_2 = (X_2, \Sigma, \alpha_2, x_{02}, X_{m2})$, *if* $F^{i+1}_{cG_2}(G_1) = F^i_{cG_2}(G_1) \neq \emptyset$, *then* $F^i_{cG_2}(G_1)$ *is state controllable with respect to* G_2 *and* Σ_{uc}, *where* $i \in \mathbb{N}^+$ *and* $F^{i+1}_{cG_2}(G_1) = F_{cG_2}(F^i_{cG_2}(G_1))$.

Proof: Let $F^i_{cG_2}(G_1) = (X_i, \Sigma, \alpha_i, x_{01}, X_{mi})$, $F^{i+1}_{cG_2}(G_1) = (X_{i+1}, \Sigma, \alpha_{i+1}, x_{01}, X_{mi+1})$ and $F^i_{cG_2}(G_1)||_{sync}G_2 = (X_{F^i_{cG_2}(G_1)||_{sync}G_2}, \Sigma, (x_{01}, x_{02}), \alpha_{F^i_{cG_2}(G_1)||_{sync}G_2}, X_{mF^i_{cG_2}(G_1)||_{sync}G_2})$. We assume the automaton $F^i_{cG_2}(G_1)$ is not state controllable w.r.t. G_2 and Σ_{uc}. That is, for $s\sigma \in L(G_2)$, where $\sigma \in \Sigma_{uc}$, there is $x_1 \in X_1$ such that $x_1 \in \alpha_i(x_{01}, s)$ and $\sigma \notin E_{F^i_{cG_2}(G_1)}(x_1)$. Since $s\sigma \in L(G_2)$, there exists $x_2 \in \alpha_2(x_{02}, s)$ such that $\sigma \in E_{G_2}(x_2)$. Then, $(x_1, x_2) \in \alpha_{F^i_{cG_2}(G_1)||_{sync}G_2}((x_{01}, x_{02}), s)$ and $q'_d \in \alpha_{F^i_{cG_2}(G_1)||_{sync}G_2}((x_1, x_2), \sigma)$, moreover, there is $x'_1 \in \alpha_i(x_{01}, s(1)\cdots s(|s|-1))$ such that $x_1 \in \alpha_i(x'_1, s(|s|))$. It follows that $x_1 \in P_{F^i_{cG_2}(G_1)||_{sync}G_2}(x'_1, s(|s|)) \neq \emptyset$. Then $x_1 \notin \alpha_{i+1}(x'_1, s(|s|))$, which violates $F^{i+1}_{cG_2}(G_1) = F^i_{cG_2}(G_1)$. Therefore, the assumption is wrong. As a result, $F^i_{cG_2}(G_1)$ is state controllable with respect to G_2 and Σ_{uc}.

Moreover, the state controllable operator preserves synchronous simulation, which can be seen from the following Lemma.

Lemma 2.4 *Given $G_1 = (X_1, \Sigma, \alpha_1, x_{01}, X_{m1})$ and $G_2 = (X_2, \Sigma, \alpha_2, x_{02}, X_{m2})$, if there is a synchronous simulation relation ϕ_1 such that $G_1 \prec_{syn\phi_1} G_2$, then there is a synchronous simulation relation ϕ_2 such that $F_{cG_2}(G_1) \prec_{syn\phi_2} G_2$.*

Proof: Let $F_{cG_2}(G_1) = (X_{F_{cG_2}(G_1)}, \Sigma, \alpha_{F_{cG_2}(G_1)}, x_{01}, X_{mF_{cG_2}(G_1)})$. Consider a relation $\phi'_2 = \{(x_1, x_2) \in X_{F_{cG_2}(G_1)} \times X_2 \mid (x_1, x_2) \in \phi_1\}$, where ϕ_1 is a synchronous simulation relation from G_1 to G_2. Next we show that ϕ'_2 is a synchronous simulation relation from $F_{cG_2}(G_1)$ to G_2. For any $(x_1, x_2) \in \phi'_2$, if there is $\sigma-$successor $x'_1 \in \alpha_{F_{cG_2(G_1)}}(x_1, \sigma)$, where $\sigma \in \Sigma$, then we have $x_1 \in X_1$ and $x'_1 \in \alpha_1(x_1, \sigma)$ from the definition of state controllable operator. It implies that there exists $x'_2 \in \alpha_2(x_2, \sigma)$ such that $(x'_1, x'_2) \in \phi_1$. Then $(x_1, x_2) \in \phi'_2$ for any $x_1 \in X_{F_{cG_2}(G_1)}$ and

$x_2 \in X_{synF_{cG_2}(G_1)G_2}(x_1)$. Therefore, $F_{cG_2}(G_1) \prec_{syn\phi_2'} G_2$.

With above results, we present a theorem to synthesize achievable sub-specifications.

Theorem 2.4 *Given a plant G and a specification R, if $F_{cG}^{i+1}(F_{syn}(G)||R)$=$F_{cG}^{i}(F_{syn}(G)||R)$ $\neq \emptyset$, then $F_{cG}^{i}(F_{syn}(G)||R)$ is synchronously simulation-based state controllable with respect to G and Σ_{uc}, where $i \in \mathbb{N}^+$ and $F_{cG_2}^{i+1}(F_{syn}(G)||R) = F_{cG_2}(F_{cG_2}^{i}(F_{syn}(G)||R))$.*

Proof: Lemma 2.1 shows that there is a synchronous simulation relation ϕ_1 such that $F_{syn}(G) \prec_{syn\phi_1} G$. Moreover, $F_{syn}(G)||R \prec F_{syn}(G)$. Thus, there exists a synchronous simulation relation ϕ_2 such that $F_{syn}(G)||R \prec_{syn\phi_2} G$ by Lemma 2.2. In addition, Lemma 2.4 indicates that there is a synchronous simulation relation ϕ_3 such that $F_{cG}^{i}(F_{syn}(G)||R) \prec_{syn\phi_3} G$. Furthermore, Lemma 2.3 implies that $F_{cG}^{i}(F_{syn}(G)||R)$ is state controllable w.r.t. G and Σ_{uc} because $F_{cG}^{i+1}(F_{syn}(G)||R) = F_{cG}^{i}(F_{syn}(G)||R) \neq \emptyset$. Hence, $F_{cG}^{i}(F_{syn}(G)||R)$ is synchronously simulation-based state controllable w.r.t G and Σ_{uc}.

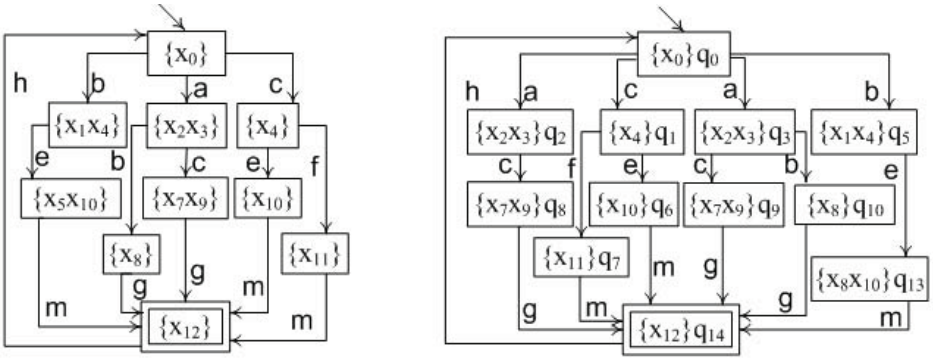

Figure 2.5: $F_{syn}(G)$ (Left) and $F_{cG}(F_{syn}(G)||R)$ (Right)

Remark 2.2 *If R fails to satisfy synchronous simulation-based state controllability and $F_{cG}^{i+1}(F_{syn}(G)||R) = F_{cG}^{i}(F_{syn}(G)||R) \neq \emptyset$, we can replace R with its sub-*

specification $F^i_{cG}(F_{syn}(G)||R)$ *to ensure the existence of a bisimilarity enforcing supervisor.*

We revisit Example 2.1 to demonstrate the synthesis of achievable sub-specifications.

Example 2.3. Consider a plant G and a specification R configured in Fig. 2.1, where $\Sigma_{uc} = \{c\}$. From Example 2.1, it is known that R is not synchronously simulation-based state controllable with respect to G and Σ_{uc}. We would like to find achievable sub-specifications of R, which satisfy synchronous simulation-based state controllability.

First, we establish $F_{syn}(G)$ as shown in Fig. 2.5. Then, we obtain $(F_{syn}(G)||R)||_{sync}G$, which indicates that $((\{x_2, x_3\}, q_4), x_2)$ and $((\{x_2, x_3\}, q_4), x_3)$ can transit to q'_d through the uncontrollable event c. Therefore, $P_{(F_{syn}(G)||R)||_{sync}G}((\{x_0\}, q_0), a) = \{(\{x_2, x_3\}, q_4)\}$ $\neq \emptyset$. Let $F_{cG}(F_{syn}(G)||R) = (X_{F_{cG}(F_{syn}(G)||R)}, \Sigma, (\{x_0\}, q_0), \alpha_{F_{cG}(F_{syn}(G)||R)}, X_{mF_{cG}(F_{syn}(G)||R)})$. Then $(\{x_2, x_3\}, q_4) \notin \alpha_{F_{cG}(F_{syn}(G)||R)}((\{x_0\}, q_0), a)$. Hence, $F_{cG}(F_{syn}(G)||R)$ is achieved in Fig. 2.5 (Right). Further, $F^2_{cG}(F_{syn}(G)||R) = F_{cG}(F_{syn}(G)||R)$. So $F_{cG}(F_{syn}(G)||R)$ is synchronously simulation-based state controllable with respect to G and Σ_{uc} by Theorem 2.4.

2.2 Specialization to Deterministic Specifications

The existence condition proposed in the previous section is sufficient only. This motivates us to consider bisimilarity supervisory control for deterministic specifications. In this section, a necessary and sufficient condition is provided for the existence of a bisimilarity enforcing supervisor with respect to deterministic specifications. It can

be verified in polynomial complexity (less than the complexity of the conditions with respect to deterministic specifications in [48]). When the existence condition holds, a bisimilarity enforcing supervisor can be constructed. Furthermore, when the existence condition does not hold, two different methods are provided for synthesizing maximal permissive sub-specifications.

2.2.1 Existence Condition

This subsection investigates the existence condition for bisimilarity enforcing supervisors. Since bisimulation implies language equivalence, the necessary condition such as language controllability for the existence of language enforcing supervisors still works for the existence of bisimilarity enforcing supervisors. Thus, we introduce the notion of langauge controllability as below.

Definition 2.10 *Given languages K and $M \in \Sigma^*$ with $K \subseteq M$, K is said to be language controllable with respect to M and Σ_{uc} if*

$$\overline{K}\Sigma_{uc} \cap M \subseteq \overline{K}.$$

Denote $det(G)$ as a minimal deterministic automaton such that $L(det(G)) = L(G)$ and $L_m(det(G)) = L_m(G)$. The result of [48] indicated that $G||det(R) \cong R$ and language controllability of $L(R)$ are necessary and sufficient conditions for the existence of a deterministic bisimilarity supervisor. In particular, $G||det(R) \cong R$ is reduced to $G||R \cong R$ if R is deterministic. In these conditions, R gets entangled with G, which fails to provide an insight about the character of R for bisimilarity control. Moreover,

the complexity of checking these condition is high. To address these problems, we will introduce synchronous simulation-based controllability as a property for R.

Definition 2.11 *Given* $G_1 = (X_1, \Sigma, \alpha_1, x_{01}, X_{m1})$ *and* $G_2 = (X_2, \Sigma, \alpha_2, x_{02}, X_{m2})$, G_1 *is said to be synchronously simulation-based controllable with respect to* G_2 *and* Σ_{uc} *if it satisfies*

(1) There is a synchronous simulation relation ϕ *such that* $G_1 \prec_{syn\phi} G_2$*;*

(2) $L(G_1)$ *is language controllable with respect to* $L(G_2)$ *and* Σ_{uc}*.*

For a deterministic specification R, R is synchronously simulated by G implies that G possesses the branches which are bisimilar to R and the branches which are outside $L(R)$. Hence, when R is deterministic, R is synchronously simulated by G iff $G||R \cong R$. Next, we show that synchronous simulation-based controllability is a necessary and sufficient condition for the existence of a bisimilarity enforcing supervisor with respect to deterministic specifications.

Theorem 2.5 *Given a plant* G *and a deterministic specification* R*, there exists a bisimilarity enforcing supervisor* S *for* G *and* R *if and only if* R *is synchronously simulation-based controllable with respect to* G *and* Σ_{uc}*.*

Moreover, synchronous simulation-based controllability offers computation advantages compared to the conditions in [48]. An algorithm will be proposed in the next subsection for testing synchronous simulation-based controllability.

2.2.2 Test for Existence Condition

This subsection proposes an algorithm to test synchronous simulation-based controllability for a deterministic R, i.e. the existence of a bisimilarity enforcing supervisor. In fact, synchronous simulation-based controllability is equivalent to synchronous simulation-based state controllability when R is deterministic. Thus, similar to Algorithm 2.1, the following algorithm is proposed.

Algorithm 2.2 *Given a plant G and a deterministic specification R, the algorithm for testing synchronous simulation-based controllability of R with respect to G and Σ_{uc} is described as below.*

Step 1: Obtain $R||_{sync}G = (X_{sync}, \Sigma, \alpha_{sync}, (q_0, x_0), X_{msync})$;

Step 2: R is synchronously simulated-based controllable with respect to G and Σ_{uc} 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 $q \in Q_m$.

Theorem 2.6 *Algorithm 2.2 is correct.*

Proof: The proof is similar to Theorem 2.2.

Compared to the results in [48], we can see that when R is deterministic, synchronously simulation-based controllability of R is equivalent to the conditions in [48] ($G||R \cong R$ and language controllability of R). In addition, the complexity of verifying synchronously simulation-based controllability and the conditions ($G||R \cong R$ and language controllability of R) are $O(|X|^2|Q|^2|\Sigma|)$ (Algorithm 2.2) and $O(|X|^2|Q|^2|\Sigma|^3 log(|X||Q|^2))$ (Remark 2 in [48]) respectively. Hence, we argue that the proposed method in this thesis is more effective.

Next, we give an example to illustrate Algorithm 2.2.

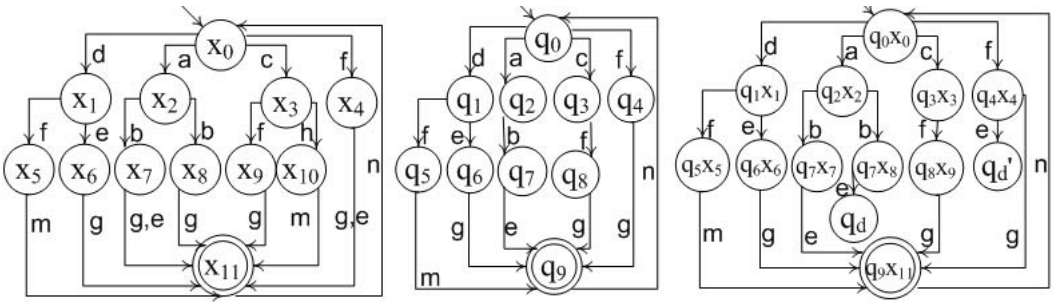

Figure 2.6: Plant G (Left), Specification R (Middle) and $R||_{sync}G$ (Right)

Example 2.4. Consider a plant G and a specification R configured in Fig. 2.6. Assure $\Sigma_{uc} = \{b, e\}$. We can see that R is not synchronously simulation-based controllable with respect to G and Σ_{uc} because for $f \in L(G) \cap L(R)$ and $e \in \Sigma_{uc}$, $fe \in L(G) \setminus L(R)$, and e is defined at q_7 but not $x_8 \in X_{synRG}(q_7)$. Next, we use Algorithm 2.2 to test synchronously simulation-based controllability of R. The synchronously simulation-based controllable product $R||_{sync}G$ is shown in Fig. 2.6 (Right). It can be seen that q_d and q'_d are reachable in $R||_{sync}G$. Hence, R is not synchronously simulation-based controllable with respect to G and Σ_{uc}.

2.2.3 Synthesis of Bisimilarity Enforcing Supervisors

In this subsection, we investigate how to synthesize a bisimilarity enforcing supervisor when R is synchronously simulation-based controllable. It is immediate to see that R_{uc} can be chosen as a candidate of bisimilarity enforcing supervisors, which is shown as below.

Theorem 2.7 *Given a plant G and a deterministic specification R, if R is synchronously simulation-based controllable with respect to G and Σ_{uc}, then R_{uc} is a bisimilarity enforcing supervisor for G and R.*

Proof: Since R is synchronously simulation-based controllable with respect to G and Σ_{uc}, we obtain that: (1) $G||R_{uc} = G||R$; and (2) there exists a synchronous simulation relation ϕ from R to G such that $R \prec_{syn\phi} G$. Moreover, $R \prec_{syn\phi} G$ implies $G||R \cong R$. Therefore, $G||R_{uc} = G||R \cong R$, i.e., R_{uc} is a bisimilarity enforcing supervisor for G and R.

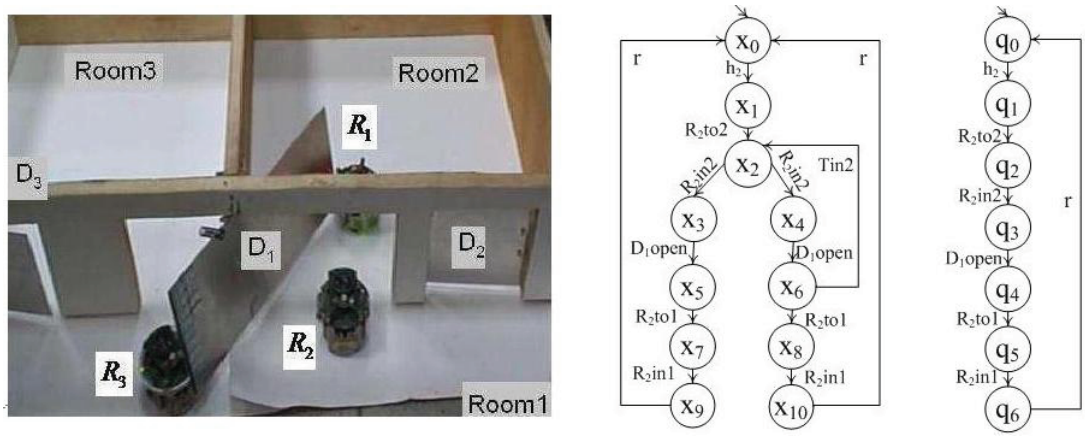

Figure 2.7: Multi-robot system (Left), Plant A_{P_2} (Middle) and Local Task Automaton $P_2(A_s)$ (Right)

Now, we give an example to illustrate the synthesis of bisimilarity enforcing supervisors for deterministic specifications.

Example 2.5. Consider a cooperative multi-robot system (MRS) example, which is adopted from [85]. In order to achieve a global task A_s, each robot R_i (with plant A_{P_i}) should accomplish the local task $P_i(A_s)$ obtained by decomposing A_s, where $i = 1, 2, 3$ and $||_{i\in\{1,2,3\}} P_i(A_s) \cong A_s$. According to Theorem 6 in [85], if we can

design a local supervisor A_{C_i} such that $A_{C_i}||A_{P_i} \cong P_i(A_s)$, this multi-robot system will achieve the global task, i.e., $||_{i\in\{1,2,3\}}A_{C_i}||A_{P_i} \cong A_s$. Here, we take robot 2 as an example. The plant A_{P_2} and the local specification $P_2(A_s)$ of robot 2 are given in Fig. 2.7. In particular, the behavior of robot 2 is described as below: After R_2 receives the help request (event h_2), it will go to Room 2 by moving towards the position on D_2 (event R_2to2). Once the robot 2 is in Room 2, it can nondeterministically goes along pre-defined paths (event R_2in2). In one path, the robot 2 can go to position on D_1 (event R_2to1) after Door 1 is opened (event D_1open). Then, it is in Room 1 (event R_2in1) after that it can go to the initial state for the next implementation (event r). In another path, the robot 2 behaves similarly. The difference is that it can take a surveillance tour (event Tin2) in Room 2. All events except h_2 are controllable in this example.

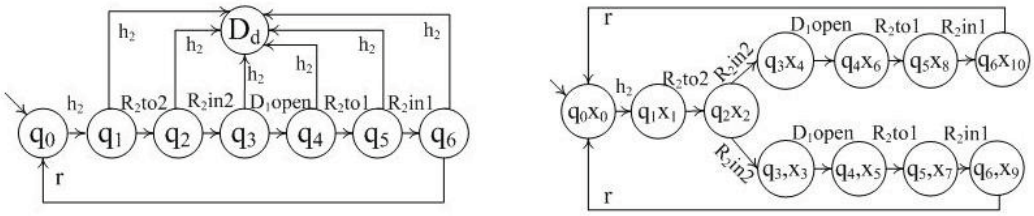

Figure 2.8: Supervisor A_{C_2} (Left) and Supervised System $A_{C_2}||A_{P_2}$ (Right)

We would like to design a bisimilarity enforcing supervisor A_{C_2} (Fig. 2.8) for A_{P_2} and $P_2(A_s)$. It can be seen that $P_2(A_s)$ is synchronously simulation-based controllable w.r.t. A_{P_2} and $\Sigma_{uc} = \{h_2\}$. By designing the supervisor A_{C_2} to be $P_2(A_s)_{uc}$, we can obtain the supervised system in Fig. 2.8 (Right). It can be seen that $A_{C_2}||A_{P_2} \cong_\phi P_2(A_s)$, where $\phi = \{((q_0, x_0), q_0), ((q_1, x_1), q_1), ((q_2, x_2), q_2), ((q_3, x_4), q_3), ((q_3, x_3), q_3), ((q_4, x_6)$

$, q_4), ((q_4, x_5), q_4), ((q_5, x_8), q_5), ((q_5, x_7), q_5), ((q_6, x_{10}), q_6), ((q_6, x_9), q_6), (q_0, (q_0, x_0)), (q_1,$ $(q_1, x_1)), (q_2, (q_2, x_2)), (q_3, (q_3, x_4)), (q_3, (q_3, x_3)), (q_4, (q_4, x_6)), (q_4, (q_4, x_5)), (q_5, (q_5, x_8)),$ $(q_5, (q_5, x_7)), (q_6, (q_6, x_{10})), (q_6, (q_6, x_9))\}$. Similarly, the bisimilarity enforcing supervisors for robot 1 and robot 3 can also be obtained. This control scenario has been implemented on a team of three robots (Fig. 2.7 (Left)).

2.2.4 Synthesis of Supremal Achievable Sub-specifications

When a given specification fails to satisfy synchronous simulation-based controllability, a natural question that arises is how to find a maximal permissive specification which guarantees the existence of a bisimilarity enforcing supervisor. To answer this question, the synthesis of supremal synchronously simulation-based controllable sub-specifications is studied in this subsection. We start by introducing the notion of supremum [86].

Definition 2.12 *Given a set A, a preorder over A, denoted $\leq \subseteq A \times A$, is a transitive and reflexive relation, in which case the pair $(A, \leq)$ is called a preordered set. Given $A' \subseteq A$, $x \in A$ is said to be a supremum of A', denoted by $supA'$, if*

(1) $\forall y \in A'$: $y \leq x$;

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

When we define the supremum of A', a set $(A, \leq)$ should be given with respect to the elements of A'. If the elements of A' are languages, the set $(2^{\Sigma^*}, \subseteq)$ should be applied because 2^{Σ^*} includes all languages over alphabet Σ and language inclusion fully captures the comparison between two languages. However, if the elements

of A' are automata, the set $(B, \prec)$ should be applied, where B is a full set of automata with alphabet Σ and $\prec \subseteq B \times B$ is the simulation relation, since B includes all automata over alphabet Σ and the simulation relation is adequate for automata (possibly nondeterministic) comparison. Please note that the supremum defined on $(A, \leq)$ 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 $A_1 = A_2$.

Consider the class of sub-specifications satisfying synchronous simulation-based controllability as below.

$$\begin{aligned} C_1 \ := \ & \{R' \mid R' \textit{ is deterministic}, R' \prec R \textit{ and } R' \textit{ is synchronous} \\ & \textit{simulation} - \textit{based controllable w.r.t. } G \textit{ and } \Sigma_{uc}\} \end{aligned}$$

It can be seen that the supremum of C_1 with respect to $(B, \prec)$ is a supremal synchronously simulation-based controllable sub-specification. However, it is difficult to directly calculate the supremum of C_1 because C_1 is not closed under the upper bound (join) operator with respect to $(B, \prec)$ [48]. To encounter this problem, 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 with respect to $(2^{\Sigma^*}, \subseteq)$ [20]. Next, we do this conversion item by item. First, for two deterministic automata R' and R, the condition $R' \prec R$ is equivalent to the language condition $L(R') \subseteq L(R)$ and $L_m(R') \subseteq L_m(R)$. Second, language controllability required in synchronous simulation-based controllability is naturally a language description. It remains to convert synchronous simulation relation required in synchronous simulation-based controllability to an equivalent language condition. By using $F_{syn}(G)$, a synchronous

simulation relation from a deterministic automaton G_1 to a plant 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 is illustrated by the following proposition.

Proposition 2.1 *Given a plant G and a deterministic automaton G_1, there is a synchronous 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 $F_{syn}(G) = (X_f, \Sigma, \{x_0\}, \alpha_f, X_{mf})$, $G_1 = (X_1, \Sigma, x_{01}, \alpha_1, X_{m1})$ and $G_L = G_1||G = (X_L, \Sigma, (x_{01}, x_0), \alpha_L, X_{mL})$. For sufficiency, consider a relation $\phi = \{(x_1, x) \in X_1 \times X \mid x \in X_{synG_1G}(x_1)\}$. We obtain that $G_1 \prec_{syn\phi} G$. For necessity, we can use the induction method to prove $L(G_1) \subseteq L(F_{syn}(G))$. In addition, for any $s' \in L_m(G_1)$, there is $x_4 \in \alpha_1(x_{01}, s')$ such that $x_4 = X_{m1}$. Because $G_1 \prec_{syn\phi} G$ implies $(x_4, x''') \in \phi$ for any $x''' \in \alpha(x_0, s')$, we have $x''' \in X_m$. The definition of $F_{syn}(G)$ implies $s' \in L_m(F_{syn}(G))$, i.e. $L_m(G_1) \subseteq L_m(F_{syn}(G))$.

Hence, the automaton set C_1 can be converted into the language sets:

$$\begin{aligned} C_2 &:= \{L_1 \subseteq L(R) \cap L(F_{syn}(G)) \mid L_1 = \overline{L_1} \text{ and } L_1 \text{ is language controllable} \\ &\qquad w.r.t.\ L(G) \text{ and } \Sigma_{uc}\}; \\ C_3 &:= \{L_1 \cap L_m(R) \cap L_m(F_{syn}(G)) \mid L_1 \in C_2\}. \end{aligned}$$

The computation of supremal synchronously simulation-based controllable subspecification, i.e., $supC_1$, with respect to $(B, \prec)$, can be achieved through the computation of the supremal languages of C_2 and C_3 with respect to $(2^{\Sigma^*}, \subseteq)$ as shown in the following theorem. For two languages $K_1, K_2 \in \Sigma^*$ with $K_2 \subseteq K_1 \neq \emptyset$, let $G_{(K_1,K_2)}$ be a deterministic automaton such that $L(G_{(K_1,K_2)}) = K_1$ and $L_m(G_{(K_1,K_2)}) = K_2$.

Theorem 2.8 *Given a plant G and a deterministic specification R, if $supC_2 \neq \emptyset$, then $G_{(supC_2,supC_3)} \in supC_1$.*

Proof: Let $L_1 = supC_2 \neq \emptyset$ and $L_1' = supC_2 \cap L_m(R) \cap L_m(F_{syn}(G)) = supC_3$. First we 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 w.r.t. $L(G)$ and Σ_{uc} and $L_1 \subseteq L(F_{syn}(G))$. Further, $L_1' \subseteq L_m(F_{syn}(G))$. From Proposition 2.1, $G_{(L_1,L_1')}$ is synchronously simulation-based controllable w.r.t. G and Σ_{uc}. Since the determinism of R and $G_{(L_1,L_1')}$ 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 synchronous simulation-based controllability of R_1. Hence $L(R_1)$ is language controllable w.r.t. $L(G)$ and Σ_{uc} and $R_1 \prec_{syn\phi} G$, which implies $L(R_1) \subseteq L(F_{syn}(G))$ and $L_m(R_1) \subseteq L_m(F_{syn}(G))$ by Proposition 2.1. Hence $L(R_1) \in C_2$. Moreover, $L_m(R_1) \subseteq L(R_1)$. We have $L(R_1) \subseteq supC_2 = L_1$ and $L_m(R_1) \subseteq supC_3 = L_1'$, further, R_1 and $G_{(L_1,L_1')}$ are deterministic. It follows that $R_1 \prec G_{(L_1,L_1')}$, which introduces a contradiction. Hence, the assumption is not correct. That is, $R_1 \prec G_{(L_1,L_1')}$ for any $R_1 \in C_1$. So $G_{(L_1,L_1')} = G_{(supC_2,supC_3)} \in supC_1$.

Next, we present a recursive algorithm and a formula-based method for computing the supremal synchronously simulation-based controllable sub-specification. For an automaton $G' = (X', \Sigma, \alpha', x_0', X_m')$ and $X_1 \subseteq X'$, the subautomaton of G' with respect to X_1, denoted by $F_{G'}(X_1)$, is defined as $F_{G'}(X_1) = (X_1, \Sigma, \alpha_1, x_0, X_{m1})$,

where $\alpha_1 = \alpha'|_{X_1 \times \Sigma}$ and $X_{m1} = X_1 \cap X'_m$.

Algorithm 2.3 *Given a plant G and a deterministic specification R, the algorithm for computing the supremal synchronously simulation-based controllable sub-specification with respect to G and Σ_{uc} is described as follows:*

Step 1: Obtain $det(G) = (X_{det}, \Sigma, \alpha_{det}, x_{0det}, X_{mdet})$, $G' = (F_{syn}(G)||R)_{uc} = (X', \Sigma, \alpha', x'_0, X'_m)$ and $G'' = G'||\ det(G) = (X'', \Sigma, \alpha'', x''_0, X''_m)$;

Step 2: $Z_0 := \{(x'_1, x_2) \in X' \times X_{det} \mid x'_1 = D_d\}$;

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 $Z_{k+1} = Z_k \neq Z$, then the subautomaton $F_{G''}(X'' - Z_k)$ of G'' is a supremal synchronously simulation-based controllable sub-specification with respect to G and Σ_{uc}.

The correctness of Algorithm 2.3 is obvious according to Theorem 2.8. Because the state set X'' is finite and the state numbers of $F_{syn}(G)$ and $det(G)$ are both $O(2^{|X|})$, Algorithm 2.3 can be terminated with complexity $O(2^{2|X|}|Q||\Sigma|)$.

In addition to the recursive algorithm, the supremal synchronously simulation-based controllable sub-specification can also be calculated by formulas. Before presenting the formula-based method, we need the following notations. 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 $\overline{K} = \{s \in \Sigma^* \mid (\exists t \in \Sigma^*)\ st \in K\}$. The quotient of K_1 with respect to K_2, denoted as K_1/K_2, is the language $K_1/K_2 = \{s \in \Sigma^* \mid (\exists t \in K_2)\ st \in K_1\}$.

Theorem 2.9 *Given a plant G and a deterministic specification R, if $M = L(R) \cap L(F_{syn}(G)) - [(L(G) - L(R) \cap L(F_{syn}(G)))/\Sigma_{uc}^*]\Sigma^* \neq \emptyset$, then $G_{(M,M')}$ is a supremal synchronously simulation-based controllable sub-specification with respect to G and Σ_{uc}, where $M' = M \cap L_m(R) \cap L_m(F_{syn}(G))$.*

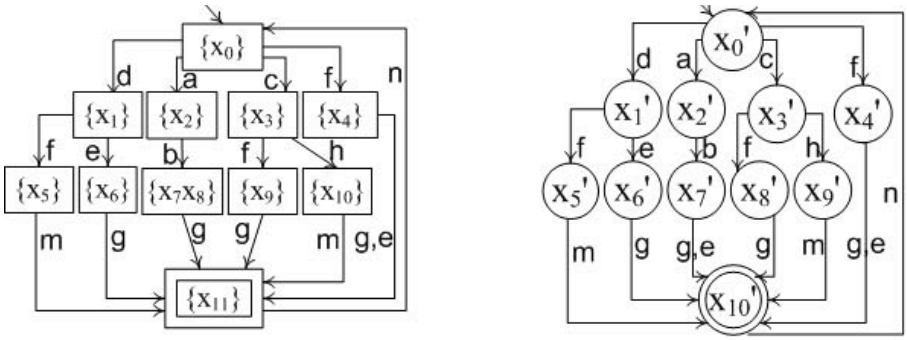

Figure 2.9: $F_{syn}(G)$ (Left) and $det(G)$ (Right)

Example 2.6: We revisit Example 2.4, which indicates that R is not synchronously simulation-based controllable. Here, we would like to calculate the supremal synchronously simulation-based controllable sub-specification w.r.t. G and Σ_{uc}.

(1) Recursive Method: From Algorithm 2.3, we establish $F_{syn}(G)$ and $det(G)$, shown in Fig. 2.9. Then $G'' = (X'', \Sigma, \alpha'', x_0'', X_m'') = (F_{syn}(G)||R)_{uc}||det(G)$ is achieved in (Fig. 2.10 (Left)). We obtain $Z_0 = \{(D_d, x'_{10})\}$, $Z_1 = Z_0 \cup \{(\{x_7, x_8\}, q_7, x'_7), (\{x_4\}, q_4, x'_4)\}$ and $Z_2 = Z_1 \cup \{(\{x_2\}, q_2, x'_2)\} = Z_3$. Therefore, the supremal synchronously simulation-based controllable sub-specification $F_{G''}(X'' - Z_2)$ is obtained in Fig. 2.10 (Right).

(2) Formula-based Method: First, we construct $F_{syn}(G)$, which can be seen in Fig. 2.9 (Left). Hence $L(R) \cap L(F_{syn}(G)) = \overline{(d(fm + eg)n + cfgn + fgn)^* ab}$. Thus,

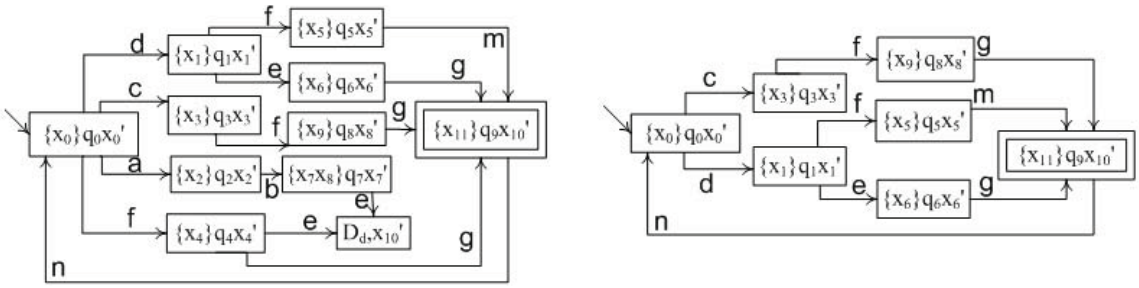

Figure 2.10: $(F_{syn}(G)||R)_{uc}||det(G)$ (Left) and $F_{G''}(X'' - Z_2)$ (Right)

$M = L(R) \cap L(F_{syn}(G)) - [(L(G) - L(R) \cap L(F_{syn}(G)))/\Sigma_{uc}^*]\Sigma^* = \overline{(d(fm + eg)n + cfgn + fgn)^*ab}$-$(d(fm + eg)n + cfgn + fgn)^*ab\Sigma^*$ -$(d(fm + eg)n + cfgn + fgn)^*a\Sigma^*$-$(d(fm+eg)n+cfgn+fgn)^*f\Sigma^* = \overline{(d(fm + eg)n + cfgn)^*} \neq \emptyset$ and $M' = M \cap L_m(R) \cap L_m(F_{syn}(G)) = (d(fm+eg)n+cfgn)^*(d(fm+eg)+cfg)$. The supremal synchronously simulation-based controllable sub-specification $G_{(M,M')} = F_{G''}(X'' - Z_2)$ is achieved in Fig. 2.10 (Right).

2.3 Conclusion

This chapter explored bisimilarity supervisor control of DESs under a centralized framework. We firstly allowed the plant, specification and supervisor to be nondeterministic. The notion of synchronous simulation-based state controllability was introduced as the sufficient condition for the existence of a bisimilarity enforcing supervisor, which can be verified by a polynomial algorithm. When the existence condition holds, a bisimilarity enforcing supervisor can be constructed. When the existence condition does not hold, the synthesis of achievable sub-specifications was further studied. Then, we put our efforts to deterministic specifications. The notion of

synchronous simulation-based controllability was presented as the necessary and sufficient condition for bisimilarity control. Such a condition can also be checked polynomially in both state sizes of the plant and specification. In addition, two constructive methods were provided for synthesizing supremal achievable sub-specifications.

Chapter 3

Decentralized Supervisory Control for Bisimulation Equivalence

This chapter extends bisimilarity supervisory control from centralized framework to decentralized framework. Firstly, an automata-based control framework is formalized, upon which we develop three architectures with respect to different decision making rules for decentralized bisimilarity control, named a conjunctive architecture, a disjunctive architecture and a general architecture. Under these three architectures, necessary and sufficient conditions for the existence of decentralized bisimilarity control are derived respectively, which extend traditional results of decentralized supervisory control from language equivalence to bisimulation equivalence. It is shown that these conditions can be verified with exponential complexity. The synthesis of decentralized bisimilarity supervisors is presented when the existence condition holds. When the specification does not satisfy the existence condition, the computation of achievable sup-specifications has been further investigated.

3.1 Automata-based Framework

We start by introducing an automata-based framework for decentralized bisimilarity control. A system G is jointly controlled by n local supervisors $\mathbf{S}_1$, $\mathbf{S}_2$ $\cdots$ $\mathbf{S}_n$ so that the supervised system is bisimilar to the given specification R. A priori information available to each local supervisor includes the desired behavior R and the decision fusion rule. Further, each local supervisor can observe the locally observable information and make the local control decisions.

Denote Σ_{ci}, Σ_{uci}, Σ_{oi} and Σ_{uoi} as locally controllable event set, locally uncontrollable event set, locally observable event set and local unobservable event set respectively, where $i \in I := \{1, 2, \cdots, n\}$. Then, the set of globally controllable events is defined as $\Sigma_c = \cup_{i\in I}\Sigma_{ci}$ and the set of globally observable events is defined as $\Sigma_o = \cup_{i\in I}\Sigma_{oi}$. The globally uncontrollable event set is given by $\Sigma_{uc} = \Sigma - \Sigma_c$ and the globally uncontrollable event set is given by $\Sigma_{uo} = \Sigma - \Sigma_o$. When a string of events occurs, the sequence of observed events is filtered by a projection P_{Σ_o}: $\Sigma^* \to \Sigma_o^*$, which is defined inductively as follows: $P_{\Sigma_o}(\epsilon) = \epsilon$, for $\sigma \in \Sigma$ and $s \in \Sigma^*$, $P_{\Sigma_o}(s\sigma) = P(s)\sigma$ if $\sigma \in \Sigma_o$, otherwise, $P_{\Sigma_o}(s\sigma) = P(s)$.

The local supervisor $\mathbf{S}_i$ is a tuple

$$\mathbf{S}_i = (S_i, \psi_i), \tag{3.1}$$

where $S_i = (Y_i, \Sigma, \beta_i, y_{0i}, Y_{mi})$ is an automaton with $Y_{mi} = Y_i$ and $\psi_i : Y_i \to \Gamma := \{\gamma \in 2^\Sigma : \Sigma_{uc} \subseteq \gamma\}$ is a local decision map.

It can be seen that a local supervisor consists of an automaton S_i and a local decision map ψ_i, where S_i dynamically tracks and synchronizes the behaviors of the

plant and ψ_i determines whether enables the events defined at the state of S_i or not. Since a local supervisor can not disable globally uncontrollable events, we have $\Sigma_{uc} \subseteq \psi_i(y_i)$ for any $y_i \in Y_i$. A local supervisor is called to be nondeterministic if S_i is nondeterministic, otherwise, it is called to be deterministic. To reduce the implementation complexity, this chapter focuses on deterministic local supervisors.

Because a local supervisor possesses limit control and observation capabilities, an admissible local supervisor should satisfy the following properties.

Definition 3.1 *Consider a supervisor* $\boldsymbol{S}_i = ((Y_i, \Sigma, \beta_i, y_{0i}, Y_{mi}), \psi_i)$. *Then,*

- $\boldsymbol{S}_i$ *is called* $\Sigma_{uoi} -$ *compatible if* $\forall y \in Y_i$ *and* $\forall \sigma \in \Sigma_{uoi}$, $\beta_i(y, \sigma) = y$;
- $\boldsymbol{S}_i$ *is called* $\Sigma_{uci} -$ *compatible if* $\forall y \in Y_i$ *and* $\forall \sigma \in \Sigma_{uci}$, $\beta_i(y, \sigma) \neq \emptyset$;
- $\boldsymbol{S}_i$ *is called* $(\Sigma_{uoi}, \Sigma_{uci}) -$ *compatible if it is* $\Sigma_{uoi} -$ *compatible and* $\Sigma_{uci} -$ *compatible.*

It is shown that a $\Sigma_{uoi} -$ *compatible* supervisor does the same control actions for the indistinguishable events, and a $\Sigma_{uci} -$ *compatible* supervisor always enables all locally uncontrollable events. Further, the decisions from local supervisors can be synthesized through the decision fusion rule, which is stated as follows.

Definition 3.2 *Given supervisors* $\boldsymbol{S}_i = (S_i, \psi_i)$ *with* $||_{i \in I} S_i = (Y_{||}, \Sigma, \beta_{||}, y_{0||}, Y_{m||})$, *where* $i \in I$, *the decision fusion rule* ψ_f *is defined as*

$$\psi_f : Y_{||} \to \Gamma := \{\gamma \in 2^{\Sigma} : \Sigma_{uc} \subseteq \gamma\}. \tag{3.2}$$

Then, the supervised system generated by decentralized bisimilarity supervisors is introduced.

Definition 3.3 *Consider a plant* $G = (X, \Sigma, \alpha, x_0, X_m)$*, a specification* $R = (Q, \Sigma, \delta, q_0, Q_m)$*, supervisors* $\boldsymbol{S}_i = (S_i, \psi_i)$ *with* $||_{i\in I} S_i = (Y_{||}, \Sigma, \beta_{||}, y_{0||}, Y_{m||})$ *and a decision fusion rule* ψ_f*, where* $i \in I$*. The supervised system* $cl_{i\in I}(S_i, \psi_i)/_{\psi_f} G$ *is defined as an automaton*

$$cl_{i\in I}(S_i, \psi_i)/_{\psi_f} G = (X', \Sigma, \alpha', x'_0, X'_m), \tag{3.3}$$

where $X' \subseteq X \times Y_{||}$*,* $x'_0 = (x_0, y_{0||})$*,* $X'_m \subseteq X_m \cap Y_{m||}$ *and the transition function* $\alpha' : X' \times \Sigma^* \to 2^{X'}$ *is defined inductively*

(1) $(x, y) \in \alpha'(x'_0, \sigma) \Leftrightarrow x \in \alpha(x_0, \sigma) \wedge y \in \beta_{||}(y_{0||}, \sigma) \wedge \sigma \in \psi_f(y_{0||})$*;*

(2) If $(x, y) \in \alpha'(x'_0, s)$*, then* $(x', y') \in \alpha'((x, y), \sigma) \Leftrightarrow x' \in \alpha(x, \sigma) \wedge y' \in \beta_{||}(y, \sigma) \wedge \sigma \in \psi_f(y)$*.*

This supervision framework can be easily implemented. When a certain event occurs in the plant, the local supervisors will update to new states based on their own observation. At these states, local decisions are made and then fuse a global decision which will be delivered to the plant through a communication channel to enforce a desired behavior.

Remark 3.1 *The supervised system is traditionally defined based on strings, e.g., [37] and [49], with respect to language equivalence. However, we focus on bisimulation equivalence rather than language equivalence, and allow the plant, specification and*

supervisor to be nondeterministic. Thus, the string-based description is generalized to the automata-based description.

Based on the proposed frameworks, this chapter aims to tackle the following decentralized bisimilarity control problem:

Given a plant G and a specification R with $L(R) \subseteq L(G)$, does there exist $(\Sigma_{uoi}, \Sigma_{uci})-$compatible supervisors $\boldsymbol{S}_i = (S_i, \psi_i)$ such that $cl_{i\in I}(S_i, \psi_i)/_{\psi_f} G \cong R$, where $i \in I$? If so, how to construct $\boldsymbol{S}_i$? If not, how to find an achievable sup-specification?

In the rest of this chapter, we will use $G = (X, \Sigma, \alpha, x_0, X_m)$, $R = (Q, \Sigma, \delta, q_0, Q_m)$, $\mathbf{S}_i = (S_i, \psi_i) = ((Y_i, \Sigma, \beta_i, y_{0i}, Y_{mi}), \psi_i)$ and $||_{i\in I} S_i = (Y_{||}, \Sigma, \beta_{||}, y_{0||}, Y_{m||})$ to denote the nondeterministic plant, nondeterministic specification, local supervisor and parallel composition of S_i respectively unless otherwise stated.

3.2 Conjunctive Architecture

In this section, a conjunctive architecture is presented for the decentralized bisimilarity control of DESs. For a conjunctive architecture, a local supervisor $\mathbf{S}_i$ enables $\Sigma_c \setminus \Sigma_{ci}$ by default, i.e., $\Sigma_c \setminus \Sigma_{ci} \subseteq \psi_i(y)$ for any $y \in Y_i$. Then, we present the conjunctive decision fusion rule.

Definition 3.4 *Given local supervisors $\boldsymbol{S}_i = (S_i, \psi_i)$ with $||_{i\in I} S_i = (Y_{||}, \Sigma, \beta_{||}, y_{0||}, Y_{m||})$, where $i \in I$, the conjunctive decision fusion rule $\psi_{fc} : Y_{||} \to 2^{\Sigma}$ is defined as*

$$\psi_{fc}(y_1, y_2, \cdots, y_n) = \cap_{i\in I} \psi_i(y_i). \tag{3.4}$$

3.2.1 Existence Condition

Next, we investigate the existence condition of decentralized bisimilarity control with respect to the conjunctive architecture. For necessity, it is known that the marking only depends on the plant because the decentralized bisimilarity supervisor plays no role in the marking. Thus, we introduce the following concept.

Definition 3.5 *Given a plant* $G = (X, \Sigma, \alpha, x_0, X_m)$ *and a specification* $R = (Q, \Sigma, \delta, q_0, Q_m)$ *with* $L(R) \subseteq L(G)$, *R is marked language closed with respect to G if*

$$(\forall s \in L(R))s \in L_m(G) \Rightarrow s \in L_m(R) \tag{3.5}$$

Moreover, bisimulation implies language equivalence, and the notion of $C\&P$ co-observability is the necessary condition for the existence of a set of decentralized language enforcing supervisors [37]. So we need the following notion for achieving bisimulation equivalence.

Definition 3.6 *Given languages* K *and* $M \in \Sigma^*$ *with* $K \subseteq M$, K *is said to be* $C\&P$ *co-observable with respect to* M, Σ_{oi} *and* Σ_{ci}, *where* $i \in I$, *if* $\forall s \in \overline{K}$ *and* $\forall \sigma \in \Sigma_c$ *such that* $s\sigma \in M \setminus \overline{K}$,

$$(\exists i \in I)[(P^{-1}_{\Sigma_{oi}} P_{\Sigma_{oi}}(s)\sigma \cap \overline{K} = \emptyset) \wedge (\sigma \in \Sigma_{ci})]. \tag{3.6}$$

For sufficiency, we present the notion of projected automaton to construct local bisimilarity enforcing supervisors. Before presenting this notion, we need following concepts. Given $\Sigma_1 \subseteq \Sigma$, Σ_1-reach of a state x is $R_{\Sigma_1}(x) = \{x' \mid (\exists s \in \Sigma_1^*)\ x' \in \alpha(x, s)\}$. In the case of a set of states $B \subseteq X$, $R_{\Sigma_1}(B) = \cup_{x \in B} R_{\Sigma_1}(x)$.

Definition 3.7 *Given $G = (X, \Sigma, \alpha, x_0, X_m)$ and $\Sigma_1 \subseteq \Sigma$, the projected automaton of G with respect to Σ_1, denoted by $P^A_{\Sigma_1}(G)$, is a tuple*

$$P^A_{\Sigma_1}(G) = (X_A, \Sigma_A, \alpha_A, x_{0A}, X_{mA})$$

where $X_A = 2^X$, $\Sigma_A = \Sigma_1$, $x_{0A} = R_{\Sigma_1}(x_0)$, $X_{mA} = X_A$ and for any $x_A \in X_A$ and $\sigma \in \Sigma_A$, the transition function is defined as

$$\alpha_A(x_A, \sigma) = \begin{cases} x_A & \sigma \in \Sigma_1; \\ R_{\Sigma_1}(\{x_1' \in X_1 \mid (\exists x_1 \in x_A)\ x_1' \in \alpha(x_1, \sigma)\}) & \sigma \in \Sigma \setminus \Sigma_1. \end{cases}$$

We provide the necessary and sufficient condition for the existence of a set of $(\Sigma_{uoi}, \Sigma_{uci}) - compatible$ bisimilarity supervisors under the conjunctive architecture.

Theorem 3.1 *Given a plant $G = (X, \Sigma, \alpha, x_0, X_m)$ and a specification $R = (Q, \Sigma, \delta, q_0, Q_m)$ with $L(R) \subseteq L(G)$, there exist $(\Sigma_{uoi}, \Sigma_{uci}) - compatible$ supervisors $\boldsymbol{S}_i = (S_i, \psi_i)$ with the conjunctive decision fusion rule ψ_{fc} such that $cl_{i\in I}(S_i, \psi_i)/_{\psi_{fc}} G \cong R$, where $i \in I$, if and only if the following conditions hold:*

(1) There is a bisimulation relation ϕ such that $G||det(R) \cong_\phi R$;

(2) $L(R)$ is language controllable with respect to $L(G)$ and Σ_{uc};

(3) $L(R)$ is C&P co-observable with respect to $L(G)$, Σ_{ci} and Σ_{oi};

(4) R is marked language closed with respect to G.

Proof: Consider $det(R) = (Z, \Sigma, \delta_Z, \{q_0\}, Z_m)$, $G||det(R) = (X_{XZ}, \Sigma, \alpha_{XZ}, (x_0, \{q_0\}), X_{mXZ})$ and $cl_{i\in I}(S_i, \psi_i)/_{\psi_{fc}} G = (X', \Sigma, \alpha', x_0', X_m')$. (Necessity) Let $\mathbf{S}_i = (S_i, \psi_i)$=((($Y_i$, $\Sigma, \beta_i, y_{0i}, Y_{mi}$), ψ_i) and $||_{i\in I} S_i = (Y_{||}, \Sigma, \beta_{||}, y_{0||}, Y_{m||})$, where $i \in I$. Because there is a

bisimulation relation ϕ' such that $cl_{i\in I}(S_i,\psi_i)/_{\psi_{fc}}G \cong_{\phi'} R$. We have $L(cl_{i\in I}(S_i,\psi_i)/_{\psi_{fc}}G) = L(R)$ and $L_m(cl_{i\in I}(S_i,\psi_i)/_{\psi_{fc}}G) = L_m(R)$.

We firstly prove that $L(R)$ is language controllable with respect to $L(G)$ and Σ_{uc}. For any $s \in L(R)$ and $\sigma \in \Sigma_{uc}$ such that $s\sigma \in L(G)$, there is $x \in \alpha(x_0, s)$ with $x' \in \alpha(x,\sigma)$. Because $s \in L(R) = L(cl_{i\in I}(S_i,\psi_i)/_{\psi_{fc}}G)$ and $||_{i\in I}S_i$ is deterministic, there exists $(x,(y_1,y_2,\cdots,y_n)) \in \alpha'(x'_0,s)$. Because $\sigma \in \Sigma_{uc}$, we have $\sigma \in \psi_{fc}(y_1,y_2,\cdots,y_n) = \cap_{i\in I}\psi_i(y_i)$. Moreover, $\mathbf{S}_i$ is $\Sigma_{uci} - compatible$, which implies $\beta_i(y_i,\sigma) \neq \emptyset$ for $i \in I$. Thus, there is $(y'_1,y'_2,\cdots,y'_n) \in \beta_{||}((y_1,y_2,\cdots,y_n),\sigma)$ such that $(x',(y'_1,y'_2,\cdots,y'_n)) \in \alpha'((x,(y_1,y_2,\cdots,y_n)),\sigma)$ according to Definition 3.3. Therefore, $s\sigma \in L(cl_{i\in I}(S_i,\psi_i)/_{\psi_{fc}}G) = L(R)$.

Secondly, we check $C\&P$ co-observability of $L(R)$ with respect to $L(G)$, Σ_{ci} and Σ_{oi}, where $i \in I$. Assume there are $s \in L(R)$ and $\sigma \in \Sigma_c$ satisfying $s\sigma \in L(G)\backslash L(R)$, moreover, either $\sigma \notin \Sigma_{ci}$ or $P_i^{-1}P_i(s)\sigma \cap L(R) \neq \emptyset$ for any $i \in I$. For any $j \in I$ satisfies $\sigma \in \Sigma_{cj}$ and $P_j^{-1}P_j(s)\sigma \cap L(R) \neq \emptyset$, there exists $s' \in L(R)$ such that $P_j(s) = P_j(s')$ and $s'\sigma \in L(R)$. Because $L(R) = L(cl_{i\in I}(S_i,\psi_i)/_{\psi_{fc}}G)$, we have $s'\sigma \in L(cl_{i\in I}(S_i,\psi_i)/_{\psi_{fc}}G)$. Then, there exists $(x,(y_1,y_2\cdots y_n)) \in \alpha'(x'_0,s')$ such that $(x',y') \in \alpha'((x,(y_1,y_2\cdots y_n)),\sigma)$. By Definition 3.3 and (3.4), we have $\sigma \in \cap_{i\in I}\psi_i(y_i)$. Since $s \in L(R)$, we have $s \in L(cl_{i\in I}(S_i,\psi_i)/_{\psi_{fc}}G)$. In addition, $s\sigma \in L(G)$. Hence, there is $x'' \in \alpha(x_0,s)$ such that $(x'',(y''_1,y''_2,\cdots,y''_n)) \in \alpha'(x'_0,s)$ and $x''' \in \alpha(x'',\sigma)$. Because $||_{i\in I}S_i$ is deterministic and $\mathbf{S}_i$ is $\Sigma_{uoi} - compatible$, if $\sigma \in \Sigma_{ci}$, we have $P_i(s) = P_i(s')$ with $\beta_i(y_{oi},s) = \beta_i(y_{oi},s')$, where $i \in I$ and $P_i : \Sigma^* \to \Sigma_{oi}^*$ is the projection. Therefore, either $\sigma \notin \Sigma_{ci}$ or $y_i = y''_i$ for $i \in I$. Hence, $\sigma \in \cap_{i\in I}\psi_i(y''_i)$. Furthermore, $\mathbf{S}_i$ is $\Sigma_{uci} - compatible$. Then, there is $y''' \in \beta_{||}((y''_1,y''_2,\cdots,y''_n),\sigma)$ such

that $(x''', y''') \in \alpha'((x'', (y_1'', y_2'', \cdots, y_n'')), \sigma)$. It implies $s\sigma \in L(cl_{i\in I}(S_i, \psi_i)/_{\psi_{fc}}G) = L(R)$, which contradicts that $s\sigma \notin L(R)$. Therefore, the assumption is not correct. Hence, $L(R)$ is $C\&P$ co-observable with respect to $L(G)$, Σ_{ci} and Σ_{oi}, where $i \in I$.

Thirdly, we verify that there is a bisimulation relation ϕ such that $G||det(R) \cong_{\phi} R$. From the definition of product, we have $L(G||det(R)) = L(G) \cap L(det(R)) = L(R)$. Thus, $L(cl_{i\in I}(S_i, \psi_i)/_{\psi_{fc}}G) = L(R) = L(G||det(R))$. Let $\phi_1 = \{((x, z), q) \in X_{XZ} \times Q \mid \exists s \in L(R) \text{ s.t. } (x, z) \in \alpha_{XZ}((x_0, \{q_0\}), s),\ q \in \delta(q_0, s),\ y \in \beta_{||}(y_{0||}, s)$ and $((x, y), q) \in \phi'\}$. For any $((x, z), q) \in \phi_1$, if there is a σ-successor $(x', z') \in \alpha_{XZ}((x, z), \sigma)$, where $\sigma \in \Sigma$, we obtain $s\sigma \in L(R) = L(cl_{i\in I}(S_i, \psi_i)/_{\psi_{fc}}G)$ and $x' \in \alpha(x, \sigma)$. Because of the determinism of $||_{i\in I}S_i$, there is $y \in \beta_{||}(y_{0||}, s)$ such that $y' \in \beta_{||}(y, \sigma)$. It implies $(x', y') \in \alpha'((x, y), \sigma)$. Then, there exists $q' \in \delta(q, \sigma)$ such that $((x', y'), q') \in \phi'$. Hence, $((x', z'), q') \in \phi_1$. If $(x, z) \in X_{mXZ}$, then $x \in X_m$, which implies $(x, y) \in X_m'$. Therefore, $q \in Q_m$. For any $(q, (x, z)) \in \phi_1^{-1}$, if there is a σ-successor $q' \in \delta(q, \sigma)$, where $\sigma \in \Sigma$, we have $(x', y') \in \alpha'((x, y), \sigma)$ such that $((x', y'), q') \in \phi'$ because $((x, y), q) \in \phi'$. Thus, $x' \in \alpha(x, \sigma)$. Further, $s\sigma \in L(R)$ implies that there exists $z' \in \delta_Z(z, \sigma)$ by the definition of $det(R)$. Thus, $(x', z') \in \alpha_{XZ}((x, z), \sigma)$. Hence, $(q', (x', z')) \in \phi_1^{-1}$. If $q \in Q_m$, then $z \in Z_m$ and $x \in X_m$. Therefore, $(x, z) \in X_{mXZ}$. As a result, $G||det(R) \cong_{\phi_1 \cup \phi_1^{-1}} R$.

Fourthly, we would like to prove that R is marked language closed with respect to G. For any $s \in L(R)$, we have $s \in L(cl_{i\in I}(S_i, \psi_i)_{\psi_{fc}}/G)$. If $s \in L_m(G)$, there is $x \in X_m$ such that $x \in \alpha(x_0, s)$. Since $s \in L(cl_{i\in I}(S_i, \psi_i)/_{\psi_{fc}}G)$, we obtain $s \in L_m(cl_{i\in I}(S_i, \psi_i)/_{\psi_{fc}}G)$, which implies $s \in L_m(R)$.

(Sufficiency) Let $\mathbf{S}_i = ((P^A_{\Sigma_{uoi}}(det(R)||det(G)))_{\Sigma_{uci}}, \psi_i) = ((Y_i, \Sigma, \beta_i, y_{0i}, Y_{mi}), \psi_i)$ and $det(R)||det(G) = (X_{det}, \Sigma, \alpha_{det}, x_{0det}, X_{mdet})$. The local decision map $\psi_i(y_i)$ is

$$\psi_i(y_i) = \begin{cases} (\Sigma_c \setminus \Sigma_{ci}) \cup \Sigma_{uc} \cup \{\sigma \in \Sigma_{ci} \mid \sigma \in \cup_{m_i \in y_i} E_{det(R)||det(G)}(m_i)\} & y_i \neq D_d; \\ (\Sigma_c \setminus \Sigma_{ci}) \cup \Sigma_{uc} & y_i = D_d. \end{cases} \tag{3.7}$$

Therefore, $\mathbf{S}_i$ is $(\Sigma_{uoi}, \Sigma_{uci}) - compatible$ and ψ_i satisfies the requirement for the conjunctive architecture. Let ψ_{fc} (3.4) be the conjunctive decision fusion rule.

Firstly, we would like to prove that $s \in L(R)$ for any $s \in L(cl_{i\in I}(S_i, \psi_i)/_{\psi_{fc}} G)$ by the induction method. (1) $|s| = 0$, that is, $s = \epsilon$. We have $\epsilon \in L(R)$. (2) Suppose that $s \in L(R)$ for any $s \in L(cl_{i\in I}(S_i, \psi_i)/_{\psi_{fc}} G)$ when $|s| = n$. (3) $|s| = n + 1$ with $s = s_1\sigma$. Assume that $s_1\sigma \notin L(R)$. Since $s_1\sigma \in L(cl_{i\in I}(S_i, \psi_i)/_{\psi_{fc}} G)$, there is $(x, (y_1, y_2, \cdots, y_n)) \in \alpha'(x'_0, s_1)$ such that $(x', (y'_1, y'_2, \cdots, y'_n)) \in \alpha'((x, (y_1, y_2, \cdots, y_n)), \sigma)$. Then, $s_1\sigma \in L(G)$ and $\sigma \in \psi_{fc}(y_1, y_2, \cdots, y_n) = \cap_{i\in I}\psi_i(y_i)$. Because $|s_1| = n$, $s_1 \in L(R)$. We have the following cases. Case 1: $\sigma \in \Sigma_{uc}$. Then, $s_1\sigma \in L(R)$ since $L(R)$ is language controllable with respect to $L(G)$ and Σ_{uc}. Thus, there is a contradiction. Case 2: $\sigma \in \Sigma_c$. Since $s_1 \in L(R)$ and $\sigma \in \cap_{i\in I}\psi_i(y_i)$, for any $i \in I$ satisfying $\sigma \in \Sigma_{ci}$, there exists $m_i \in y_i$ such that $\sigma \in E_{det(R)||det(G)}(m_i)$. From the definition of projected automata, we have $m_i \in \alpha_{det}(x_{0det}, s'_1)$ with $P_{\Sigma_{oi}}(s_1) = P_{\Sigma_{oi}}(s'_1)$. Thus, $s'_1\sigma \in L(R)$. It violates $C\&P$ co-observability of $L(R)$.

Secondly, the induction method is also used to verify $s \in L(cl_{i\in I}(S_i, \psi_i)/_{\psi_{fc}} G)$ for any $s \in L(R)$. (1) $|s| = 0$, that is, $s = \epsilon$. We have $\epsilon \in L(cl_{i\in I}(S_i, \psi_i)/_{\psi_{fc}} G)$. (2) Suppose that $s \in L(cl_{i\in I}(S_i, \psi_i)/_{\psi_{fc}} G)$ for any $s \in L(R)$ when $|s| = n$. (3) $|s| = n + 1$ with $s = s_1\sigma$. Since $s_1\sigma \in L(R)$, we have $s_1\sigma \in L(G)$. Then,

there is $x \in \alpha(x_0, s_1)$ such that $x' \in \alpha(x, \sigma)$. Moreover $|s_1| = n$, we obtain $s_1 \in L(cl_{i\in I}(S_i, \psi_i)/_{\psi_{fc}}G)$. Because $||_{i\in I}S_i$ is deterministic, there is $(y_1, y_2, \cdots, y_n) \in \beta_{||}(y_{0||}, s_1)$ such that $(x, (y_1, y_2, \cdots, y_n)) \in \alpha'(x'_0, s_1)$. For any $i \in I$ and $m_i \in y_i$, there is s'_1 with $P_{\Sigma_{oi}}(s_1) = P_{\Sigma_{oi}}(s'_1)$ such that $m_i \in \alpha_{det}(x_{0det}, s'_1)$. Then, we obtain the following cases. (1) $\sigma \in \Sigma_{uc}$. Because of $\Sigma_{uci} - compatiblility$ of $\mathbf{S}_i$, we have $\beta_i(y_i, \sigma) \neq \emptyset$ for $i \in I$. Further, $\sigma \in \psi_i(y_i)$ for $i \in I$ since $\sigma \in \Sigma_{uc}$. Thus, there is $(y'_1, y'_2, \cdots, y'_n) \in \beta_{||}((y_1, y_2, \cdots, y_n), \sigma)$ such that $(x', (y'_1, y'_2, \cdots, y'_n)) \in \alpha'((x, (y_1, y_2, \cdots, y_n)), \sigma)$. Hence, $s_1\sigma \in L(cl_{i\in I}(S_i, \psi_i)/_{\psi_{fc}}G)$. (2) $\sigma \in \Sigma_c$. Because $s_1\sigma \in L(R)$ and $S_i = (P^A_{\Sigma_{uoi}}(det(R)||det(G)))_{uci}$ is deterministic, there exist $y'_i \in \beta_i(y_i, \sigma)$ and $m_i \in y_i$ such that $\sigma \in E_{det(R)||det(G)}(m_i)$. Thus, we have $\sigma \in \cap_{i\in I}\psi_i(y_i) = \psi_{fc}(y_1, y_2, \cdots, y_n)$ and $(y'_1, y'_2, \cdots, y'_n) \in \beta_{||}((y_1, y_2, \cdots, y_n), \sigma)$. Then, $(x', (y'_1, y'_2, \cdots, y'_n)) \in \alpha'((x, (y_1, y_2, \cdots, y_n)), \sigma)$ which implies $s_1\sigma \in L(cl_{i\in I}(S_i, \psi_i)/_{\psi_{fc}}G)$.

Thirdly, we would like to verify the existence of a bisimulation relation between the supervised system and the specification. Because there is a bisimulation relation such that $G||det(R) \cong_\phi R$, we have $L(G||det(R)) = L(R)$. In addition, we know $L(cl_{i\in I}(S_i, \psi_i)/_{\psi_{fc}}G) = L(R)$. Thus, $L(cl_{i\in I}(S_i, \psi_i)/_{\psi_{fc}}G) = L(G||det(R)) = L(R)$.

Let $\phi_1 = \{((x, y), q) \in X' \times Q \mid \exists s \in L(R) \text{ s.t. } y \in \beta_{||}(y_{0||}, s),\ x \in \alpha(x_0, s),\ q \in \delta(q_0, s),\ z \in \delta_z(\{q_0\}, z) \text{ and } ((x, z), q) \in \phi\}$. For any $((x, y), q) \in \phi_1$, if there is a σ-successor $(x', y') \in \alpha'((x, y), \sigma)$, where $\sigma \in \Sigma$, we obtain $s\sigma \in L(cl_{i\in I}(S_i, \psi_i)/_{\psi_{fc}}G) = L(R)$ and $x' \in \alpha(x, \sigma)$. Thus, there exists $z' \in \delta_z(z, \sigma)$ by the definition of $det(R)$. Then, $(x', z') \in \alpha_{XZ}((x, z), \sigma)$. Because $((x, z), q) \in \phi$, there exists $q' \in \delta(q, \sigma)$ such that $((x', z'), q') \in \phi$. Therefore, $((x', y'), q') \in \phi_1$. If $(x, y) \in X'_m$, then $x \in X_m$. It implies $s \in L_m(G)$. Because R is marked language closed with respect to G, we have

$s \in L_m(R)$. Then, $z \in Z_m$. Hence, $(x, z) \in X_{mXZ}$ which implies $q \in Q_m$. For any $(q, (x, y)) \in \phi_1^{-1}$, if there is a σ-successor $q' \in \delta(q, \sigma)$, where $\sigma \in \Sigma$, we have $(x', z') \in \alpha_{XZ}((x, z), \sigma)$ such that $((x', z'), q') \in \phi$ because $((x, z), q) \in \phi$. Then, $x' \in \alpha(x, \sigma)$. Further, $s\sigma \in L(R) = L(cl_{i\in I}(S_i, \psi_i)/_{\psi_{fc}}G)$, there exists $(x', y') \in \alpha'((x, y), \sigma)$ because of the determinism of $||_{i\in I}S_i$. Hence, $(q', (x', y')) \in \phi_1^{-1}$. If $q \in Q_m$, then $x \in X_m$. Therefore, $(x, y) \in X'_m$. As a result, $cl_{i\in I}(S_i, \psi_i)/_{\psi_{fc}}G \cong_{\phi_1 \cup \phi_1^{-1}} R$.

Intuitively, condition (1) depicts that the amount of nondeterminism of the plant restricted by the deterministic controller equals to the amount of nondeterminism of the specification. Moreover, condition (4) is required because bisimulation implies not only language equivalence but also marked language equivalence, i.e., $L_m(cl_{i\in I}(S_i, \psi_i)/_{\psi_{fc}}G) = L_m(R)$.

Remark 3.2 *To verify the existence of a set of decentralized bisimilarity supervisors for the conjunctive architecture, we examine the conditions of Theorem 3.1 item by item. (1)* $G||det(R) \cong_\phi R$. *Since both the plant and specification are nondeterministic, their numbers of transitions are* $O(|X|^2|\Sigma|)$ *and* $O(|Q|^2|\Sigma|)$ *respectively. Moreover,* $det(R)$ *is deterministic with* $O(2^{|Q|}|\Sigma|)$ *transitions. According to the result in [87], the complexity of checking* $G||det(R) \cong_\phi R$ *is* $O(|X|^2 2^{|Q|^2}|\Sigma| log(|X|2^{|Q|}))$. *(2)* $L(R)$ *is language controllable with respect to* $L(G)$ *and* Σ_{uc}, *which can be tested with complexity* $O(|X|^2|Q|^2|\Sigma|)$ *[20]. (3)* $L(R)$ *is C&P co-observable with respect to* $L(G)$, Σ_{ci} *and* Σ_{oi}, *where* $i \in I$. *It can be verified by polynomial complexity with respect to* $|X|$ *and* $|Q|$ *[88]. (4)* R *is marked language closed with respect to* G. *By checking the states of* $G||R$, *condition (4) can be tested with complexity* $O(|X||Q|)$. *Therefore,*

the computational complexity of verifying Theorem 3.1 is $O(|X|^2 2^{|Q|^2}|\Sigma|log(|X|2^{|Q|}))$, *which is exponential with respect to* $|X|$ *and* $|Q|$.

3.2.2 Synthesis of Decentralized Bisimilarity Supervisors

According to the result of the previous subsection, the construction of decentralized bisimilarity supervisors is naturally obtained.

Theorem 3.2 *Given a plant* $G = (X, \Sigma, \alpha, x_0, X_m)$ *and a specification* $R = (Q, \Sigma, \delta, q_0, Q_m)$ *with* $L(R) \subseteq L(G)$, *if the existence condition of Theorem 3.1 holds, then* $\{\boldsymbol{S}_i = ((P^A_{\Sigma_{uoi}}(det(R)||det(G)))_{\Sigma_{uci}}, \psi_{fc})\}$, *where* $i \in I$ *and* ψ_{fc} *defined by (3.7), is a set of decentralized bisimilarity supervisors under the conjunctive architecture.*

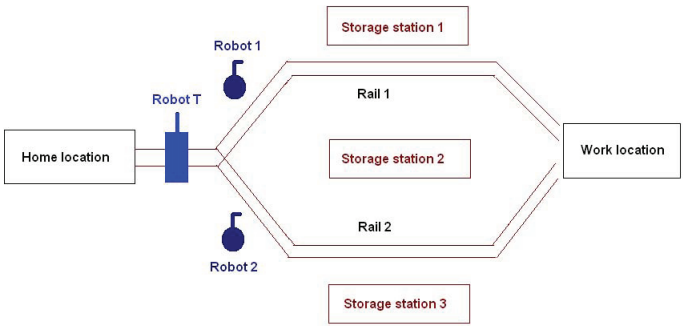

Figure 3.1: Manufacturing System

Now, we provide an example to illustrate the proposed techniques.

Example 3.1. Consider the following manufacturing example adopted from [46]. A manufacturing system consists of a home location, a work location, three storage stations and three robots, which is shown in Fig. 3.1. Robot T is available at its home

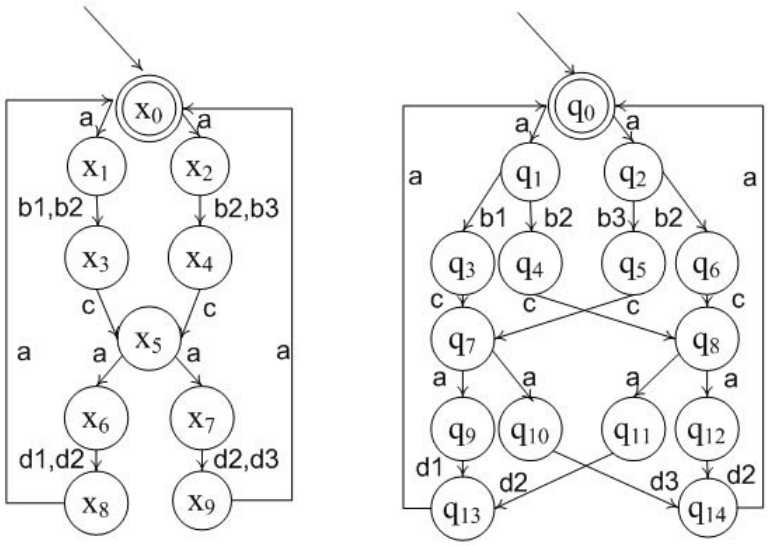

Figure 3.2: Plant G (Left) and Specification R (Right)

location to traverse on one of the two rails. Traversal on Rail i ($i = 1, 2$) is randomly chosen and is denoted by event a. While Robot T is on Rail i, it can pick a part from Storage i (event b_i) or Storage $(i+1)$ (event b_{i+1}), and then it takes the part to work location for processing (event c). When returning, Robot T can nondeterministically choose a Rail-i and drop the part to either Storage i (event d_i) or Storage $(i+1)$ (event d_{i+1}) and returns to its home location. Robot 1 and Robot 2 can monitor and supervise the manufacturing process.

The control specification requires that a part should be returned to its original pickup location except the parts picked up at Storage 1 (respectively Storage 3) can also be returned to Storage 3 (respectively, Storage 1), as those parts are exchangeable. The specification also requires that Robot T always be able to return to its home location (which means that the state representing the home location is the only

marked state). Models G and R of the manufacturing system and its specification are given in Fig. 3.2.

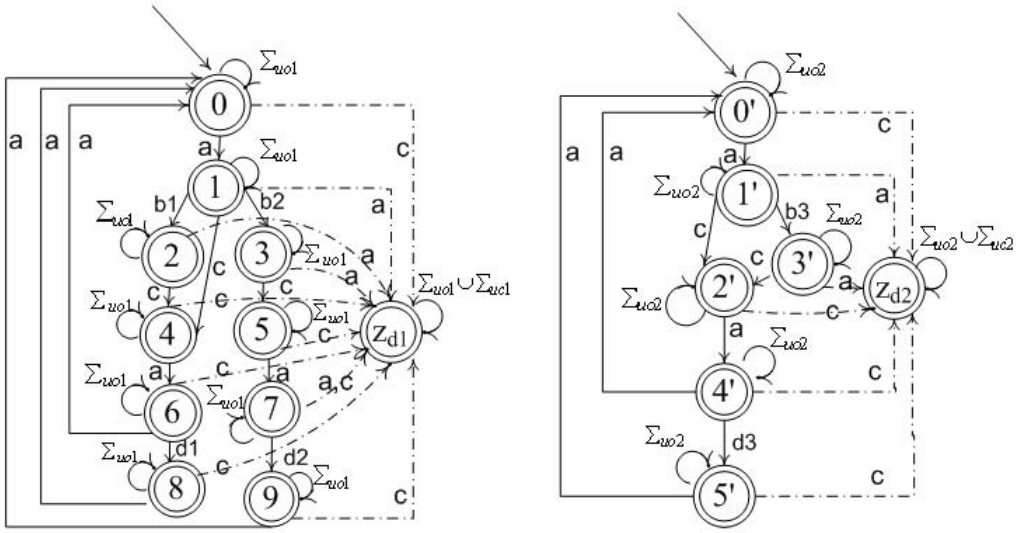

Figure 3.3: Automata S_1 (Left) and S_2 (Right)

Suppose $\Sigma_{o1} = \{a, c, b_1, b_2, d_1, d_2\}$, $\Sigma_{o2} = \{a, c, b_3, d_3\}$, $\Sigma_{c1} = \{b_1, b_2, d_1, d_2, d_3\}$ and $\Sigma_{c2} = \{b_3, d_3\}$. Then, $\Sigma_{uc} = \{a, c\}$, $\Sigma_{uc1} = \{a, c, b_3\}$ and $\Sigma_{uc2} = \{a, c, b_1, d_1, b_2, d_2\}$. For this example, we obtain that $L(G) = \overline{(ab_1cad_1a + ab_1cad_2a + ab_1cad_3a + ab_2 + cad_1a + ab_2cad_2a + ab_2cad_3a + ab_3cad_1a + ab_3cad_2a + ab_3cad_3a)^*}$ and $L(R) = \overline{(ab_1cad_1a + ab_1cad_3a + ab_2cad_2a + ab_3cad_1a + ab_3cad_3a)^*}$. It can be seen that $L(R)$ is controllable with respect to $L(G)$ and Σ_{uc} and $L(R)$ is $C\&P$ co-observable with respect to $L(G)$, Σ_{ci} and Σ_{oi}, where $i = 1, 2$. In addition, we can obtain $det(R)$, which implies there is a bisimulation ϕ such that $G||det(R) \cong_\phi R$. According to Theorem 3.1, there exist decentralized bisimilarity supervisors for the conjunctive architecture.

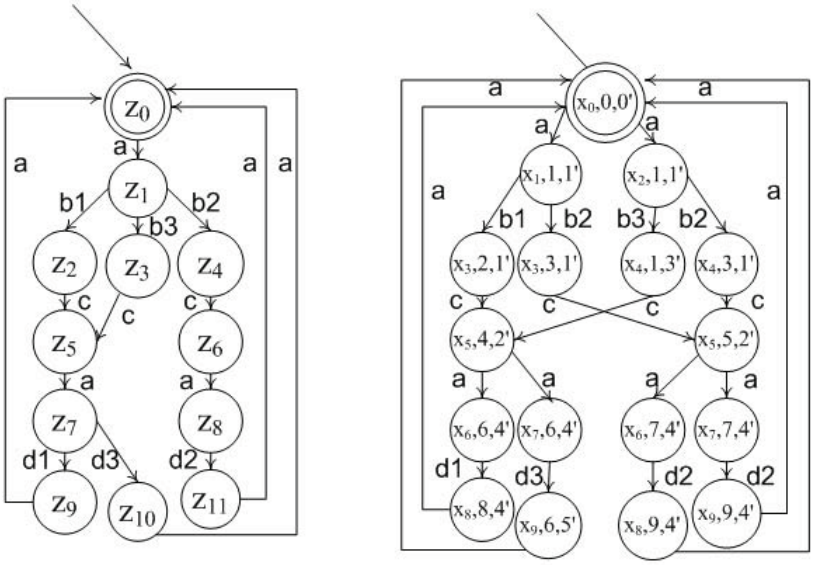

Figure 3.4: det(R) (Left) and $cl(\mathbf{S}_1, \mathbf{S}_2)/_{\psi_{fc}}G$ (Right)

Then, $\mathbf{S}_1$ and $\mathbf{S}_2$ can be constructed, where S_1 and S_2 are shown in Fig. 3.3, and the local decision maps ψ_1 and ψ_2 are described as below.

$$\psi_1(y) = \begin{cases} \{a, c, b_3\} & y = 0, 2, 3, 4, 5, 8, 9, z_{d1}; \\ \{a, c, b_1, b_2, b_3\} & y = 1; \\ \{a, c, b_3, d_1, d_3\} & y = 6; \\ \{a, c, b_3, d_2\} & y = 7. \end{cases}$$

$$\psi_2(y) = \begin{cases} \{a, c, b_1, b_2, d_1, d_2\} & y = 0', 2', 3', 5', z_{d2}; \\ \{a, c, b_1, b_2, b_3, d_1, d_2\} & y = 1'; \\ \{a, c, b_1, b_2, d_1, d_2, d_3\} & y = 4'. \end{cases}$$

Then, the supervised system is shown in Fig. 3.4 (Right). It can be verified that $cl(\mathbf{S}_1, \mathbf{S}_2)/_{\psi_{fc}}G \cong_{\phi_1 \cup \phi_1^{-1}} R$, where ψ_{fc} is defined as (3.4).

If we consider $det(R)$ (Fig. 3.4 (Left)) as the specification, it can be seen that $G||det(R)$ is not bisimilar to R. Therefore, we can not find a solution for the decentralized bisimilarity control problem. However, we can achieve language equivalence for the decentralized control problem since $L(R)$ is $C\&P$ co-observable with respect to $L(G)$, Σ_{oi} and Σ_{ci} for $i = 1, 2$. Hence, the decentralized control for language equivalence is easier than the decentralized control for bisimulation equivalence.

3.2.3 Synthesis of Achievable Sup-specifications

A given specification does not always satisfy the conditions of Theorem 3.1, a natural question arises is how to find a achievable specification which guarantees the existence of a set of decentralized bisimilarity enforcing supervisors. To answer this question, we introduce the following sets which describe the class of achievable sup-specifications with respect to conjunctive architectures.

$$\begin{aligned} CO(R) \;:=\; \{R' \,|R \prec R'\,, & L(R) \textit{ is language controllable}, C\&P \textit{ co} - \textit{observable} \\ & \textit{and marked language closed w.r.t. } L(G) \textit{and } G||det(R) \cong R\}; \end{aligned}$$

Additionally, the concept of infimum is stated [86].

Definition 3.8 *Given a preordered set $(A, \leq)$, $x \in A$ is said to be the infimum of A', denoted by $inf A'$ and $\sqcap A'$, if*

(1) $\forall y \in A'$: $x \leq y$;

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

Theorem 3.3 presents the computation of infimal achievable sup-specifications

under the conjunctive architecture. Before introducing them, the following results are needed.

Lemma 3.1 *Given languages K_1, K_2 and K_3, if K_1 and K_2 are marked language closed with respect to K_3 and Σ_{uc}, then $K_1 \cap K_2$ is marked language closed with respect to K_3 and Σ_{uc}.*

Proposition 3.1 *If $A_i \in CO(R)$, then $A_i||\{det(A_j) \mid A_j \in CO(R) - \{A_i\}\} \in CO(R)$.*

Proof: Since $A_i, A_j \in CO(R)$, then $L(A_i)$ and $L(A_j)$ are language controllable, C & P co-observable and marked langauge closed with respect to $L(G)$. Moreover, $L(A_i||\{det(A_j) \mid A_j \in CO(R) - \{A_i\}\}) = L(A_i) \cap_{A_j \in CO(R)-\{A_i\}} L(det(A_j))$. The facts that language controllability, C&P co-observability and mark language are closed under intersection and Lemma 3.1 imply $L(A_i||\{det(A_j) \mid A_j \in CO(R) - \{A_i\}\})$ is langauge controllable, C & P co-observable and marked langauge closed with respect to $L(G)$. Next, we would like to prove that $A_i||\{det(A_j) \mid A_j \in CO(R) - \{A_i\}\} \cong G||det(A_i||\{det(A_j) \mid A_j \in CO(R) - \{A_i\}\})$. Since bisimilarity preserves when "det" operator commutes with parallel composition and $G||det(A_i) \cong A_i$, $G||det(A_i||\{det(A_j) \mid A_j \in CO(R) - \{A_i\}\}) \cong G||det(A_i)||\{det(A_j) \mid A_j \in CO(R) - \{A_i\}\} \cong A_i||\{det(A_j) \mid A_j \in CO(R) - \{A_i\}\}$. Therefore, $A_i||\{det(A_j) \mid A_j \in CO(R) - \{A_i\}\} \in CO(R)$.

Theorem 3.3 *Given a plant G and a specification R, if $R \prec G$ and $L(R)$ is language controllable, C&P co-observable and marked language w.r.t. $L(G)$, then $G||det(R) \in infCO(R)$.*

Proof: Since $R \prec G$, we have $L(R) \subseteq L(G)$. Thus, $L(G||det(R)) = L(G) \cap L(det(R)) = L(G) \cap L(R) = L(R)$. Thus, $L(G||det(R))$ is language controllable, C & P co-observable and marked language w.r.t. $L(G)$. Further, since $G||det(G) \cong G$ and bisimilarity preserves when "det" operator commutes with parallel composition, $G||det(G||det(R)) = G||(det(G)||det(R)) \cong G||det(R)$. Next we show that $G||det(R) \prec A$ for any $A \in CO(R)$. Since $A \cong G||det(A)$, it suffices to show that $det(R) \prec det(A)$. This obviously holds because $R \prec A$.

Example 3.2. Example 3.1 indicates that there does not exist a set decentralized bisimilarity enforcing supervisors with respect to the specification $det(R)$ since $G||det(R) \ncong det(R)$. Thus, an achievable sup-specification is in need to enable the existence of a decentralized bisimilarity control.

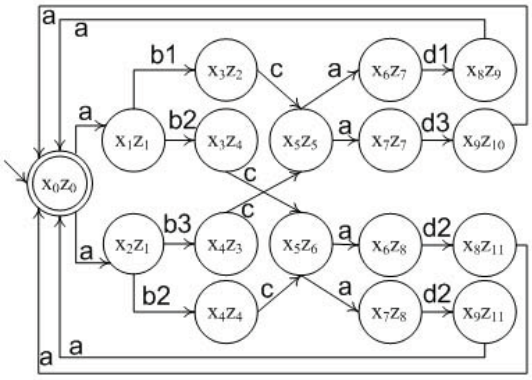

Figure 3.5: $G||det(R)$

We know that $L(R)$ is language controllable, $C\&P$ co-observable and marked language w.r.t. $L(G)$. From Theorem 3.3, we obtain that $G||det(R) \in infCO(R)$, i.e., $G||det(R)$ (Fig. 3.5) is an infimal achievable sup-specification for decentralized bisimilarity control with respect to the conjunctive architecture.

3.3 Disjunctive Architecture

This section introduces a disjunctive architecture for the decentralized bisimilarity control of DESs. With respect to the disjunctive architecture, a local supervisor $\mathbf{S}_i$ disables $\Sigma_c \setminus \Sigma_{ci}$ by default, i.e., $(\Sigma_c \setminus \Sigma_{ci}) \cap \psi_i(y) = \emptyset$ for any $y \in Y_i$. And the disjunctive fusion rule is stated as below.

Definition 3.9 *Given local supervisors* $\boldsymbol{S}_i = (S_i, \psi_i)$ *with* $||_{i\in I} S_i = (Y_{||}, \Sigma, \beta_{||}, y_{0||}, Y_{m||})$, *where* $i \in I$, *the disjunctive decision fusion rule* $\psi_{fd} : Y_{||} \to 2^{\Sigma}$ *is defined as*

$$\psi_{fd}(y_1, y_2, \cdots, y_n) = \cup_{i\in I} \psi_i(y_i). \tag{3.8}$$

3.3.1 Existence Condition

This subsection studies the necessary and sufficient condition for the existence of a decentralized bisimilarity control with respect to the disjunctive architecture. Before that, we need the notion of $D\&A$ co-observability [49], which guarantees the existence of a set of language enforcing supervisors for the disjunctive structure.

Definition 3.10 *Given languages* K *and* $M \in \Sigma^*$ *with* $K \subseteq M$, K *is said to be* $D\&A$ *co-observable with respect to* M, Σ_{oi} *and* Σ_{ci}, *where* $i \in I$, *if* $\forall s \in \overline{K}$ *and* $\forall \sigma \in \Sigma_c$ *such that* $s\sigma \in \overline{K}$,

$$(\exists i \in I)[((P^{-1}_{\Sigma_{oi}} P_{\Sigma_{oi}}(s) \cap \overline{K})\sigma \cap M \subseteq \overline{K}) \wedge (\sigma \in \Sigma_{ci})]. \tag{3.9}$$

Theorem 3.4 *Given a plant* $G = (X, \Sigma, \alpha, x_0, X_m)$ *and a specification* $R = (Q, \Sigma, \delta, q_0, Q_m)$ *with* $L(R) \subseteq L(G)$, *there exist* $(\Sigma_{uoi}, \Sigma_{uci}) -$ *compatible supervisors* $\boldsymbol{S}_i =$

(S_i, ψ_i) with the disjunctive decision fusion rule ψ_{fd} such that $cl_{i\in I}(S_i, \psi_i)/_{\psi_{fd}} G \cong R$ if and only if the following conditions hold:

(1) There is a bisimulation relation ϕ such that $G||det(R) \cong_\phi R$;

(2) $L(R)$ is language controllable with respect to $L(G)$ and Σ_{uc};

(3) $L(R)$ is D&A co-observable with respect to $L(G)$, Σ_{ci} and Σ_{oi}, where $i \in I$.

(4) R is marked language closed with respect to G.

Proof: The necessary part is similar to Theorem 3.1. Let $det(R) = (Z, \Sigma, \delta_Z, \{q_0\}, Z_m)$, $G||det(R) = (X_{XZ}, \Sigma, \alpha_{XZ}, (x_0, \{q_0\}), X_{mXZ})$ and $cl_{i\in I}(S_i, \psi_i)/_{\psi_{fd}} G = (X', \Sigma, \alpha', x'_0, X'_m)$. For sufficiency, let $\mathbf{S}_i = ((P^A_{\Sigma_{uoi}}(det(R)||det(G))_{\Sigma_{uci}}, \psi_i) = ((Y_i, \Sigma, \beta_i, y_{0i}, Y_{mi}), \psi_i)$ and $det(R)||det(G) = (X_{det}, \Sigma, \alpha_{det}, x_{0det}, X_{mdet})$. Further, the local decision map $\psi_i(y_i)$ is defined as

$$\psi_i(y_i) = \begin{cases} \Sigma_{uc} \cup \{\sigma \in \Sigma_{ci} \mid \sigma \in \cap_{m_i \in y_i} E_{det(R)||det(G)}(m_i)\} & y_i \neq D_d; \\ \Sigma_{uc} & y_i = D_d. \end{cases} \tag{3.10}$$

It is obvious that $cl_{i\in I}(S_i, \psi_i)/_{\psi_{fd}} G \cong R$.

Remark 3.3 *It is shown that D&A co-observability of $L(R)$ can be verified by polynomial complexity with respect to $|X|$ and $|Q|$ from [49]. By using Remark 3.2, the computational complexity of verifying the existence of a set of decentralized bisimilarity supervisors with respect to disjunctive architecture is $O(|X|^2 2^{|Q|^2} |\Sigma| log(|X| 2^{|Q|}))$.*

3.3.2 Synthesis of Decentralized Bisimilarity Supervisors

This subsection presents the establish of a set of decentralized bisimilarity supervisors for the disjunctive architecture.

Theorem 3.5 *Given a plant* $G = (X, \Sigma, \alpha, x_0, X_m)$ *and a specification* $R = (Q, \Sigma, \delta, q_0, Q_m)$ *with* $L(R) \subseteq L(G)$*, if the existence condition of Theorem 3.4 holds, then* $\{\boldsymbol{S}_i = ((P^A_{\Sigma_{uoi}}(det(R)||det(G)))_{\Sigma_{uci}}, \psi_{fd})\}$*, where* $i \in I$ *and* ψ_{fc} *defined by (3.10), is a set of decentralized bisimilarity supervisors under the disjunctive architecture.*

Moreover, we provide examples to illustrate the proposed synthesis methods.

Example 3.3. Consider a plant G and a specification R configured in Fig. 3.6. Let $i = 1, 2$, $\Sigma_{o1} = \{a, c, d, e, f, g, h\}$, $\Sigma_{o2} = \{b, c, d, e, f, g, h\}$, $\Sigma_{c1} = \{c, e, f, g, h\}$ and $\Sigma_{c2} = \{c, d, e, f, g, h\}$. The aim of control is to design decentralized supervisors $\mathbf{S}_1$ and $\mathbf{S}_2$ with a global decision fusion rule ψ_f such that $cl(\mathbf{S}_1, \mathbf{S}_2)/_{\psi_f} G \cong R$.

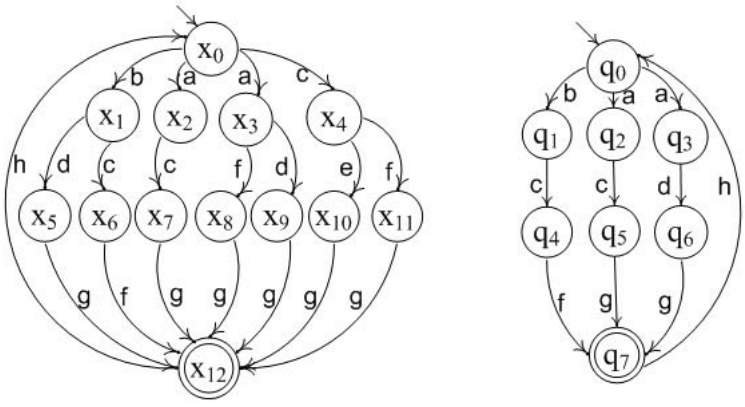

Figure 3.6: Plant G (Left) and Specification R (Right)

For $c \in \Sigma_{c1} \cap \Sigma_{c2}$, we have $c \notin L(R)$. On the other hand, there exist $b \in P_1^{-1}P_1(\epsilon)$ and $a \in P_2^{-1}P_2(\epsilon)$ such that $ac, bc \in L(R)$. Thus, $L(R)$ is not $C\&P$ co-observable

with respect to $L(G)$, Σ_{oi} and Σ_{ci}, where $i = 1, 2$. However, $L(R)$ is $D\&A$ co-observable with respect to $L(G)$, Σ_{oi} and Σ_{ci}, where $i = 1, 2$. Moreover, $L(R)$ is language controllable with respect to $L(G)$ and Σ_{uc} and $G||det(R) \cong R$. It implies that there is a set of decentralized supervisors for the disjunctive architecture to achieve bisimulation equivalence between the supervised system and the specification.

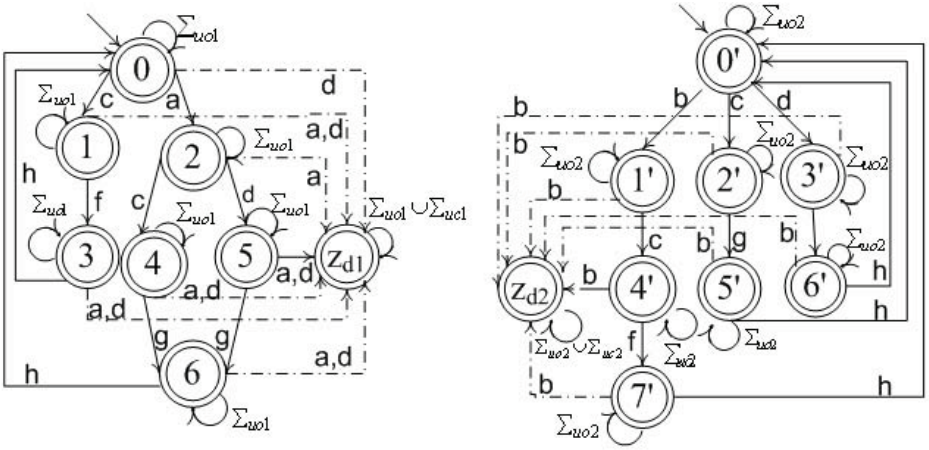

Figure 3.7: Automata S_1 (Left) and S_2 (Right)

We design decentralized bisimilarity supervisors $\mathbf{S}_1 = (S_1, \psi_1)$ and $\mathbf{S}_2 = (S_2, \psi_2)$, in which S_1 and S_2 are shown in Fig. 3.7 and ψ_1 and ψ_2 are presented as below.

$$\psi_1(y) = \begin{cases} \{a, b\} & y = 0, z_{d1}; \\ \{a, b, f\} & y = 1; \\ \{a, b, c, d\} & y = 2; \\ \{a, b, g\} & y = 4, 5; \\ \{a, b, h\} & y = 3, 6. \end{cases}$$

$$\psi_2(y) = \begin{cases} \{a,b,d\} & y = 0'; \\ \{a,b,c\} & y = 1'; \\ \{a,b,g\} & y = 2', 3'; \\ \{a,b,f\} & y = 4'; \\ \{a,b,h\} & y = 5', 6', 7'; \\ \{a,b\} & y = z_{d2}. \end{cases}$$

By using $\mathbf{S}_1$ and $\mathbf{S}_2$, the supervised system $cl(\mathbf{S}_1, \mathbf{S}_2)/_{\psi_{fd}}G$ is obtained in Fig. 3.8 (Right), where ψ_{fd} is defined as (3.8). Let $\phi_1 = \{((x_0, 0, 0'), q_0), ((x_1, 0, 1'), q_1), ((x_2, 2, 0'), q_2), ((x_3, 2, 0'), q_3), ((x_6, 1, 4'), q_4), ((x_7, 4, 2'), q_5), ((x_7, 4, 2'), q_6), ((x_9, 5, 3'), q_5), ((x_9, 5, 3'), q_6), ((x_{12}, 3, 7'), q_7), ((x_{12}, 6, 5'), q_7), ((x_{12}, 6, 6'), q_7)\}$. Thus, $cl(\mathbf{S}_1, \mathbf{S}_2)/_{\psi_{fd}}G \cong_{\phi_1 \cup \phi_1^{-1}} R$.

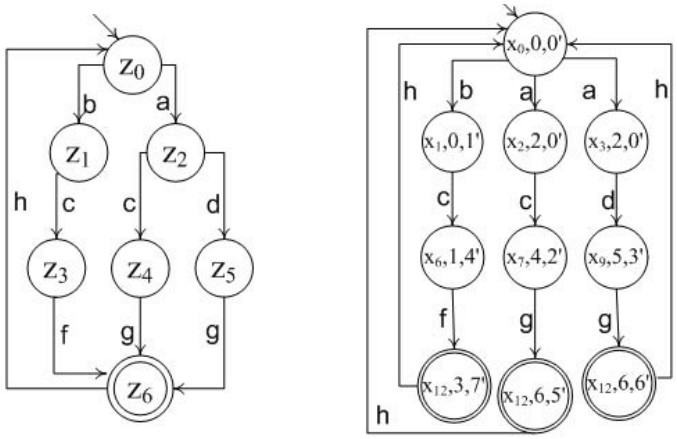

Figure 3.8: det(R) (Left) and $cl(\mathbf{S}_1, \mathbf{S}_2)/_{\psi_{fd}}G$ (Right)

Example 3.4. Consider a plant G and a specification R, which are shown in Fig. 3.9. Assume $i = 1, 2$, $\Sigma_{o1} = \{c, d, e, f, g, m\}$, $\Sigma_{o2} = \{b, d, e, f, g, m\}$, $\Sigma_{c1} = \{a, c, d, f, g, m\}$ and $\Sigma_{c2} = \{a, c, d, e, f, g, m\}$. We would like to design decentralized supervisors $\mathbf{S}_1$ and $\mathbf{S}_2$ with a global decision rule ψ_f such that $cl(\mathbf{S}_1, \mathbf{S}_2)/_{\psi_f}G \cong R$.

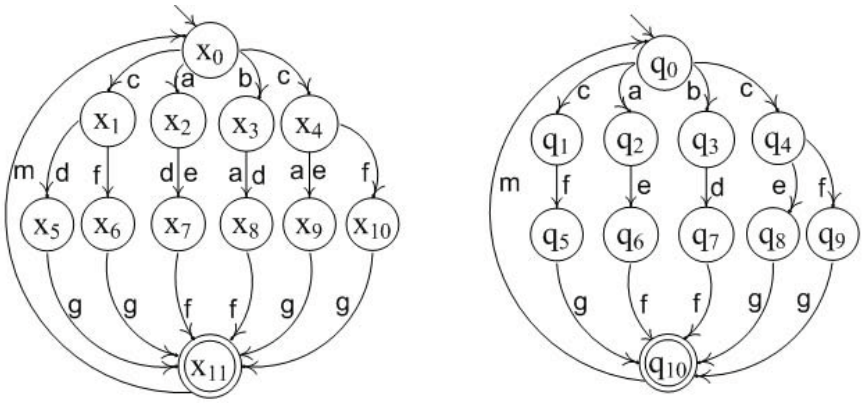

Figure 3.9: Plant G (Left) and Specification R (Right)

It can be seen that $L(R)$ is controllable with respect to $L(G)$ and Σ_{uc} and $C\&P$ co-observable with respect to $L(G)$, Σ_{ci} and Σ_{oi}, where $i = 1, 2$. In addition, $det(R)$ is obtained (Fig. 3.11 (Left)), which shows that $G||det(R) \cong R$. According to Theorem 3.1, there exist decentralized bisimilarity supervisors for the conjunctive architecture.

We establish $\mathbf{S}_1 = (S_1, \psi_1)$ and $\mathbf{S}_2 = (S_2, \psi_2)$, where S_1 and S_2 are shown in Fig. 3.10, and the local decision maps ψ_1 and ψ_2 are described as follows.

$$\psi_1(y) = \begin{cases} \{a, b, c, d, e\} & y = 0; \\ \{b, e, f\} & y = 1, 2, 3; \\ \{b, e, g\} & y = 4, 5; \\ \{b, e, m\} & y = 6, 7; \\ \{b, e\} & y = z_{d1}. \end{cases}$$

$$\psi_2(y) = \begin{cases} \{a, b, c, e, f\} & y = 0'; \\ \{b, f, g\} & y = 1'; \\ \{b, g\} & y = 2'; \\ \{b, d\} & y = 3'; \\ \{b, m\} & y = 4', 6'; \\ \{b, f\} & y = 5'; \\ \{b\} & y = z_{d2}. \end{cases}$$

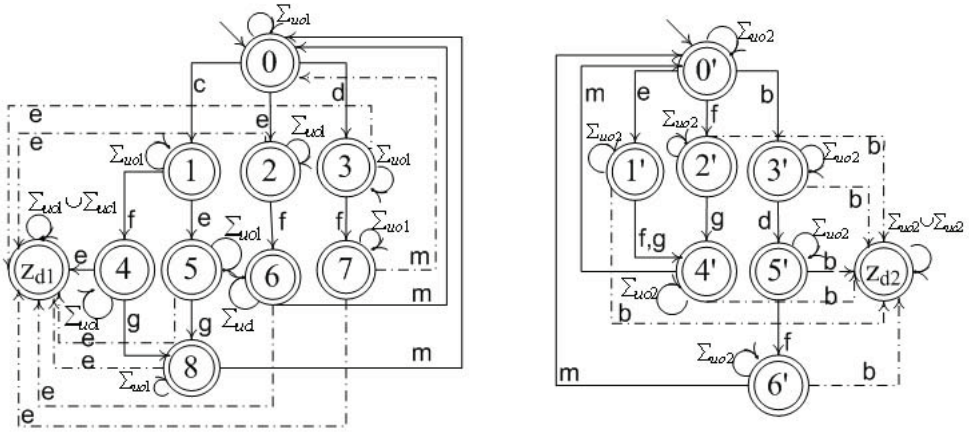

Figure 3.10: Automata S_1 (Left) and S_2 (Right)

The supervised system is presented in Fig. 3.11 (Right). It can be verified that $cl(\mathbf{S}_1, \mathbf{S}_2)/_{\psi_{fc}} G \cong_{\phi_1 \cup \phi_1^{-1}} R$, where $\phi_1 = \{((x_0, 0, 0'), q_0), ((x_2, 0, 0'), q_2), ((x_3,\ 0, 3'), q_3), ((x_1, 1, 0'), q_1), ((x_4, 1, 0'), q_4), ((x_7, 2, 1'), q_6), ((x_7, 2, 1'), q_7), ((x_8, 3, 5'), q_6), ((\ x_8, 3, 5'), q_7), ((x_6, 4, 2'), q_5), ((x_6, 4, 2'), q_8), ((x_6, 4, 2'), q_9), ((x_9, 5, 1'), q_5), ((x_9, 5, 1'), q_8), ((x_9, 5, 1'), q_9), ((x_{10}, 4, 2'), q_5), ((x_{10}, 4, 2'), q_8), ((x_{10}, 4, 2'), q_9), ((x_{11}, 6, 4'), q_{10}), ((x_{11}, 7, 6'), q_{10}), ((x_{11}, 8, 4'), q_{10})\}$ and ψ_{fc} is defined as (3.4).

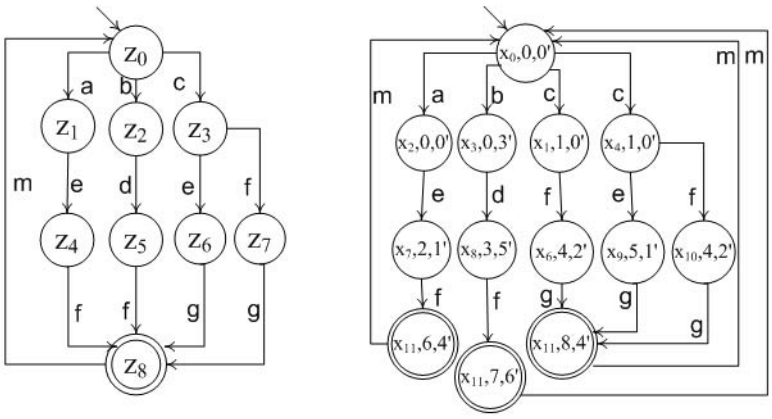

Figure 3.11: det(R) (Left) and $cl(\mathbf{S}_1, \mathbf{S}_2)/_{\psi_{fc}}G$ (Right)

Next, we check whether there exist decentralized bisimilarity supervisors with respect to the disjunctive architecture. For $a \in L(R)$, we have $ba \in [(P_1^{-1}P_1(\epsilon) \cap L(R))a \cap L(G)] \setminus L(R)$ and $ca \in [(P_2^{-1}P_2(\epsilon) \cap L(R))a \cap L(G)] \setminus L(R)$ for $a \in \Sigma_{c1} \cap \Sigma_{c2}$, implying $L(R)$ is not $D\&A$ co-observable with respect to $L(G)$, Σ_{oi} and Σ_{ci}, where $i = 1, 2$. So we cannot find decentralized bisimilarity supervisors for the disjunctive architecture. On the other hand, decentralized bisimilarity supervisors exist for the conjunctive architecture. Therefore, the conjunctive architecture is complementary with respect to the disjunctive architecture.

3.3.3 Synthesis of Achievable Sup-specifications

The class of achievable sup-specifications for disjunctive architecture is described as below.

$$\begin{aligned} DO(R) \;:=\; \{R' \,|R \prec R' \,, L(R) \textit{ is language controllable}, D\&A \textit{ co} - \textit{observable} \\ \textit{and marked language closed w.r.t. } L(G) \textit{and } G||det(R) \cong R\}; \end{aligned}$$

Subsequently, the calculation of achievable sup-specifications is presented.

Theorem 3.6 *Given a plant G and a specification R, if $R \prec G$ and $L(R)$ is language controllable, D&A co-observable and marked language w.r.t. $L(G)$, then $G||det(R) \in DO(R)$.*

Proof: Since $R \prec G$ implies $L(R) \subseteq L(G)$, then $L(G||det(R)) = L(G) \cap L(det(R)) = L(G) \cap L(R) = L(R)$. Thus, $L(G||det(R))$ is language controllable, D & A co-observable and marked language closed w.r.t. $L(G)$. By using the facts that $G||det(G) \cong G$ and bisimilarity preserves when "det" operator commutes with parallel composition, $G||det(G||det(R)) = G||(det(G)||det(R)) \cong G||det(R)$. Hence, $A_i||\{det(A_j) \mid A_j \in CO(R) - \{A_i\}\} \in DO(R)$.

3.4 General Architecture

In this section, we present a general architecture for decentralized bisimilarity control. The event set Σ_c is further partitioned into $\Sigma_c = \Sigma_{ce} \cup \Sigma_{cd}$, where Σ_{ce} is the set of controllable events enabled by default from the local decision aspect and Σ_{cd} is the set of controllable events disabled by default from the local decision aspect. That is, a local supervisor $\mathbf{S}_i$ for the general architecture satisfies $\Sigma_{ce} \setminus \Sigma_{ci} \subseteq \psi_i(y)$ and $(\Sigma_{cd} \setminus \Sigma_{ci}) \cap \psi_i(y) = \emptyset$ for any $y \in Y_i$. Denote $\Sigma_{cei} = \Sigma_{ci} \cap \Sigma_{ce}$ and $\Sigma_{cdi} = \Sigma_{ci} \cap \Sigma_{cd}$. The decision fusion rule of the general architecture is captured by the following definition.

Definition 3.11 *Consider local supervisors $\boldsymbol{S}_i = (S_i, \psi_i)$ with $||_{i \in I} S_i = (Y_{||}, \Sigma, \beta_{||},$*

$y_{0||}$, $Y_{m||}$), *where* $i \in I$. *The general decision fusion rule* $\psi_{fg} : Y_{||} \rightarrow 2^{\Sigma}$ *is defined as*

$$\psi_{fg}(y_1, y_2, \cdots, y_n) = [\Sigma_{ce} \cap (\cap_{i \in I} \psi_i(y_i))] \cup [\Sigma_{cd} \cap (\cup_{i \in I} \psi_i(y_i))] \cup \Sigma_{uc}. \tag{3.11}$$

3.4.1 Existence Condition

Under the general structure, the following concept is employed as the existence condition of a set of decentralized supervisors to achieve language equivalence between the plant and the specification [49].

Definition 3.12 *Given languages* K *and* $M \in \Sigma^*$ *with* $K \subseteq M$, K *is said to be co-observable with respect to* M, Σ_{oi} Σ_{cei} *and* Σ_{cdi}, *where* $i \in I$ *if*

(1) K *is C&P co-observable with respect to* M, Σ_{oi} *and* Σ_{cei};

(2) K *is D&A co-observable with respect to* M, Σ_{oi} *and* Σ_{cdi}.

The following theorem depicts the necessary and sufficient condition for the existence of a set of decentralized bisimilarity supervisors with respect to the general architecture.

Theorem 3.7 *Given a plant* $G = (X, \Sigma, \alpha, x_0, X_m)$ *and a specification* $R = (Q, \Sigma, \delta, q_0, Q_m)$ *with* $L(R) \subseteq L(G)$, *there exist* $(\Sigma_{uoi}, \Sigma_{uci}) -$ *compatible supervisors* $\boldsymbol{S}_i = (S_i, \psi_i)$ *with the general decision fusion rule* ψ_{fg} *such that* $cl_{i \in I}(S_i, \psi_i)/_{\psi_{fg}} G \cong R$, *where* $i \in I$, *if and only if the following conditions hold:*

(1) There is a bisimulation relation ϕ *such that* $G||det(R) \cong_{\phi} R$;

(2) $L(R)$ *is language controllable with respect to* $L(G)$ *and* Σ_{uc};

(3) $L(R)$ is co-observable with respect to $L(G)$, Σ_{oi}, Σ_{cei} and Σ_{cdi};

(4) R is marked language closed with respect to G.

Proof: Consider $det(R) = (Z, \Sigma, \delta_Z, \{q_0\}, Z_m)$, $G||det(R) = (X_{XZ}, \Sigma, \alpha_{XZ}, (x_0, \{q_0\}), X_{mXZ})$ and $cl_{i\in I}(S_i, \psi_i)/_{\psi_{fg}}G = (X', \Sigma, \alpha', x'_0, X'_m)$. For Necessity, the proof is similar to Theorem 3.1 and Theorem 3.4. For sufficiency, let $\mathbf{S}_i = ((P^A_{\Sigma_{uoi}}(det(R)||det(G))_{\Sigma_{uci}}, \psi_i) = ((Y_i, \Sigma, \beta_i, y_{0i}, Y_{mi}), \psi_i)$ and $det(R)||det(G) = (X_{det}, \Sigma, \alpha_{det}, x_{0det}, X_{mdet})$. The local decision map $\psi_i(y_i)$ is

$$\psi_i(y_i) = \begin{cases} \Sigma_{uc} \cup \Sigma_{ce} \setminus \Sigma_{cei} \cup \{\sigma \in \Sigma_{cei} \mid \sigma \in \cup_{m_i \in y_i} E_{det(R)||det(G)}(m_i)\} & \\ \qquad \{\sigma \in \Sigma_{cdi} \mid \sigma \in \cap_{m_i \in y_i} E_{det(R)||det(G)}(m_i)\} & y_i \neq D_d; \\ \qquad \Sigma_{uc} \cup \Sigma_{ce} \setminus \Sigma_{cei} & y_i = D_d. \end{cases} \tag{3.12}$$

We can easily prove that $cl_{i\in I}(S_i, \psi_i)/_{\psi_{fg}}G \cong R$.

Since $C\&P$ co-observability and $D\&A$ co-observability are the special cases of co-observability, the result (Theorem 3.7) for the general architecture is a generation of Theorem 3.1 and Theorem 3.4 which consider the conjunctive architecture and disjunctive architecture respectively.

Remark 3.4 *Refer to Remark 3.2 and Remark 3.3, the computational complexity of verifying the existence condition of a set of decentralized bisimilarity supervisors for the general architecture is $O(|X|^2 2^{|Q|^2} |\Sigma| log(|X|2^{|Q|}))$.*

3.4.2 Synthesis of Decentralized Bisimilarity Supervisors

This subsection investigates the synthesis of decentralized bisimilarity supervisors with respect to the general architecture.

Theorem 3.8 *Given a plant* $G = (X, \Sigma, \alpha, x_0, X_m)$ *and a specification* $R = (Q, \Sigma, \delta, q_0, Q_m)$ *with* $L(R) \subseteq L(G)$*, if the existence condition of Theorem 3.7 holds, then* $\{\boldsymbol{S}_i = ((P^A_{\Sigma_{uoi}}(det(R)||det(G)))_{\Sigma_{uci}}, \psi_{fg})\}$*, where* $i \in I$ *and* ψ_{fg} *defined by (3.12), is a set of decentralized bisimilarity supervisors under the general architecture.*

This synthesis procedure is shown in the following example.

Example 3.5. Consider a plant G and a specification R shown in Fig. 3.12. Let $i = 1, 2$, $\Sigma_{o1} = \{a, b, d, e, f, g, h\}$, $\Sigma_{o2} = \{a, c, d, f, g, h\}$, $\Sigma_{c1} = \{a, e, f\}$ and $\Sigma_{c2} = \{a, f\}$. The aim of control is to design decentralized supervisors $\mathbf{S}_1$ and $\mathbf{S}_2$ with a global decision fusion rule ψ_f such that $cl(\mathbf{S}_1, \mathbf{S}_2)/_{\psi_f} G \cong R$.

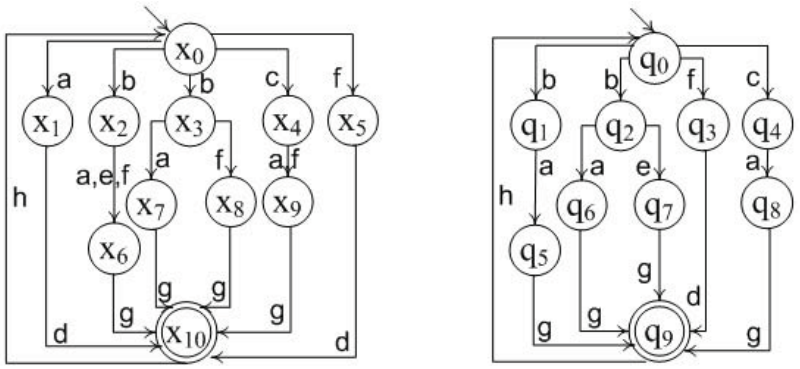

Figure 3.12: Plant G (Left) and Specification R (Right)

We can see that $c \in \Sigma_{c1} \cap \Sigma_{c2}$ and $c \notin L(R)$. And there exist $c \in P_1^{-1}P_1(\epsilon)$ and $a \in P_2^{-1}P_2(\epsilon)$ such that $ca, ba \in L(R)$. Thus, $L(R)$ is not $C\&P$ co-observable with

respect to $L(G)$, Σ_{oi} and Σ_{ci}, where $i = 1, 2$. Moreover, $f \in \Sigma_{c1} \cap \Sigma_{c2}$, $f \in L(R)$, $cf \in [(P_1^{-1}P_1(\epsilon) \cap L(R))f \cap L(G)] \setminus L(R)$ and $bf \in [(P_2^{-1}P_2(\epsilon) \cap L(R))f \cap L(G)] \setminus L(R)$. Thus, $L(R)$ is not $D\&A$ co-observable with respect to $L(G)$, Σ_{oi} and Σ_{ci}, where $i = 1, 2$.

By Theorem 3.1 and Theorem 3.4, there does not exist a set of decentralized bisimilarity supervisors for both the conjunctive and disjunctive architectures.

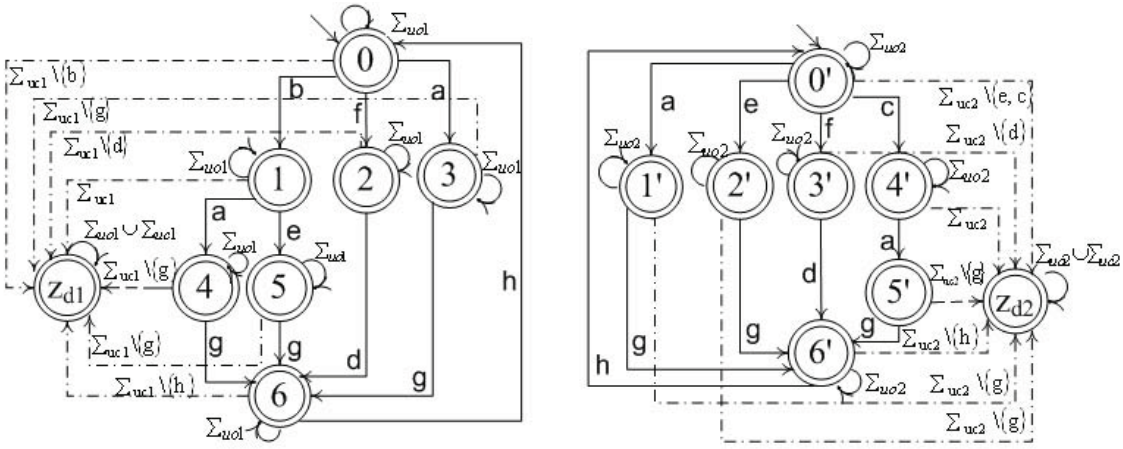

Figure 3.13: Automata S_1 (Left) and S_2(Right)

Let $\Sigma_c = \Sigma_{ce} \cup \Sigma_{cd}$, where $\Sigma_{ce} = \{f, e\}$ and $\Sigma_{cd} = \{a\}$. Then, $\Sigma_{ce1} = \Sigma_{ce} \cap \Sigma_{c1} = \{f, e\}$, $\Sigma_{ce2} = \Sigma_{ce} \cap \Sigma_{c2} = \{f\}$, $\Sigma_{cd1} = \Sigma_{cd} \cap \Sigma_{c1} = \{a\}$ and $\Sigma_{cd2} = \Sigma_{cd} \cap \Sigma_{c2} = \{a\}$. It can be easily verified that $L(R)$ is co-observable with respect to $L(G)$, Σ_{oi}, Σ_{cei} and Σ_{cdi}, where $i = 1, 2$. In addition, $L(R)$ is language controllable with respect to $L(G)$ and Σ_{uc} and $G||det(R) \cong R$. Therefore, we can find decentralized bisimilarity enforcing supervisors $\mathbf{S}_1 = (S_1, \psi_1)$ and $\mathbf{S}_2 = (S_2, \psi_2)$ for the general architecture. In particular, S_1 and S_2 are shown in Fig. 3.7 and ψ_1 and ψ_2 are presented as below.

$$\psi_1(y) = \begin{cases} \{b,c,d,g,h\} & y = 2,3,4,5,6,z_{d1}; \\ \{b,c,d,f,g,h\} & y = 0; \\ \{a,b,c,d,e,g,h\} & y = 1. \end{cases}$$

$$\psi_2(y) = \begin{cases} \{b,c,d,e,g,h\} & y = 1',2',3',5',6',z_{d2}; \\ \{b,c,d,e,f,g,h\} & y = 0'; \\ \{a,b,c,d,e,g,h\} & y = 4'. \end{cases}$$

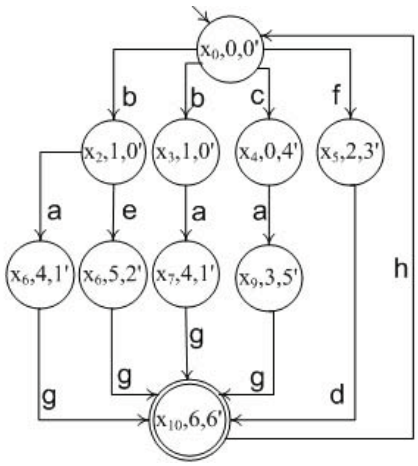

Figure 3.14: Supervised System $cl(\mathbf{S}_1, \mathbf{S}_2)/_{\psi_{fg}}G$

Therefore, $cl(\mathbf{S}_1, \mathbf{S}_2)/_{\psi_{fg}}G \cong R$ (Fig. 3.14), where ψ_{fg} is defined as (3.11).

3.4.3 Synthesis of Achievable Sup-specifications

We state the set of achievable sup-specifications enabling the existence of a decentralized bisimilarity control for the general architecture.

$$GO(R) := \{R' \mid R \prec R', L(R) \textit{ is language controllable, co} - \textit{observable and marked language closed w.r.t. } L(G), \textit{and } G||det(R) \cong R\}.$$

The computation of achievable sup-specifications is stated as follows.

Theorem 3.9 *Given a plant G and a specification R, if $R \prec G$ and $L(R)$ is language controllable, co-observable and marked language w.r.t. $L(G)$, then $G||det(R) \in GO(R)$.*

Proof: It is known that $R \prec G$ implies $L(R) \subseteq L(G)$. Therefore, $L(G||det(R)) = L(G) \cap L(det(R)) = L(G) \cap L(R) = L(R)$, which implies $L(G||det(R))$ is language controllable, D & A co-observable and marked language w.r.t. $L(G)$. Moreover, $G||det(G) \cong G$ and bisimilarity preserves when "det" operator commutes with parallel composition, implying $G||det(G||det(R)) = G||(det(G)||det(R)) \cong G||det(R)$. As a result, $A_i||\{det(A_j) \mid A_j \in CO(R) - \{A_i\}\} \in GO(R)$.

3.5 Conclusion

The decentralized bisimilarity control of discrete event systems was studied in this chapter, where the plant and specification are modeled as nondeterministic automata. To formally capture the branching information, we propose an automata-based framework, upon which a conjunctive architecture, a disjunctive architecture and a general architecture were constructed for decentralized bisimilarity control with respect to different decision making rules. Then, necessary and sufficient conditions for the existence of a set of $\Sigma_{uci} - compatible$ and $\Sigma_{uoi} - compatible$ deterministic bisimilarity supervisors were presented under these three architectures, respectively. It was shown that these conditions can be verified with exponential complexity. When the existence

condition holds, we have provided the synthesis of decentralized bisimilarity supervisors accordingly. When the specification does not satisfy the existence condition, the computation of achievable sup-specifications has been further investigated.

Chapter 4

Distributed Supervisory Control for Bisimulation Equivalence

This chapter studies bisimilarity control of distributed discrete event systems which consist of multiple interacting modules. The goal is to design local supervisors such that the globally supervised system (the parallel composition of locally supervised modules) is bisimilar to the given specification. Here, each local supervisor determines the control actions upon its observation on the local module to be controlled and its neighbors. The notion of separably and synchronously simulation-based state controllability is introduced as the existence condition for distributed bisimilarity control. When the given specification satisfies this condition, a set of local supervisors can be constructed to enforce bisimulation equivalence. Otherwise, the calculation of achievable sub-specifications is further investigated. In addition, we focus on deterministic supervisors for distributed bisimilarity control. The synthesis of deterministic supervisors and achievable sup-specifications are investigated. The comparisons

of our results with the centralized monolithic ones are presented.

A distributed DES is composed of serval concurrent modules. Denote $G_i = (X_i, \Sigma_i, x_{0i}, \alpha_i, X_{mi})$ as the local module with $i \in I := \{1, 2, \cdots, n\}$. The global system $G = (X, \Sigma, x_0, \alpha, X_m)$ is the parallel composition of local components, i.e., $G = ||_{i\in I} G_i$ with $\Sigma = \cup_{i\in I}\Sigma_i$. The local event set Σ_i is partitioned into $\Sigma_i = \Sigma_{ci} \cup \Sigma_{uci}$, where Σ_{ci} is the controllable event set of G_i and Σ_{uci} is the uncontrollable event set of G_i. Then, the globally controllable and uncontrollable event sets are given by $\Sigma_c = \cup_{i\in I}\Sigma_{ci}$ and $\Sigma_{uc} = \Sigma - \Sigma_c$ respectively. This chapter aims to solve the following problems.

Problem 1: Given a distributed discrete event system $G = ||_{i\in I} G_i$ and a specification R, where $i \in I$, does there exist a set of $\Sigma_{uci}-$compatible supervisors S_i such that the globally supervised system $||_{i\in I}(S_i||G_i)$ is bisimilar to the desired specification R?

Problem 2: If so, how to design $\Sigma_{uci}-$compatible supervisors S_i?

Problem 3: If not, how to find an achievable sub/sup-specification?

In the rest of this chapter, we will use $G = ||_{i\in I} G_i$, $R = (Q, \Sigma, \delta, q_0, Q_m)$ and $S_i = (Y_i, \Sigma_i, \beta_i, y_{0i}, Y_{mi})$ to denote the distributed plant, specification and local supervisor (possibly nondeterministic) respectively unless otherwise stated.

4.1 Existence Condition

This section investigates Problem 1. Since the alphabet of the local plant is $\Sigma_i \subset \Sigma$, it is natural to decompose the global specification R into local specifications R_i with alphabet Σ_i. Then, the following concept is introduced to capture the decomposition.

Definition 4.1 *Given a specification R and event sets $\{\Sigma_i\}$ for $i \in I$, R is $\{\Sigma_i\}-$separable if there exists a group of R_i with alphabet Σ_i such that $R \cong ||_{i\in I}R_i$.*

In [85], a method was provided to verify $\{\Sigma_i\}-$separability by using the projected automata. The interested reader can refer to [85] for more details. Next, we present the notion of $\{\Sigma_i, G_i, \Sigma_{uci}\}-$separable and synchronous simulation-based state controllability which servers as the sufficient condition for the existence of distributed bisimilarity control. It is indicated in Theorem 4.1.

Definition 4.2 *Given a distributed plant $G = ||_{i\in I}G_i$ and a specification R, where $i \in I$, R is $\{\Sigma_i, G_i, \Sigma_{uci}\}-$separably and synchronously simulation-based state controllable if there exists a group of R_i with alphabet Σ_i such that*

(1) $R \cong ||_{i\in I}R_i$;

(2) R_i is synchronously simulation-based state controllable w.r.t. G_i and Σ_{uci}.

Before presenting Theorem 4.1, the following result is needed.

Proposition 4.1 *Given $G_i = (X_i, \Sigma_i, \alpha_i, x_{0i}, X_{mi})$, where $i \in \{1, 2, 3, 4\}$, if $G_1 \cong G_2$, $G_3 \cong G_4$, $\Sigma_1 = \Sigma_2$ and $\Sigma_3 = \Sigma_4$, then $G_1||G_3 \cong G_2||G_4$.*

Proof: Let $G_i = (X_i, \Sigma_i, x_{0i}, \alpha_i, X_{mi})$ $(i = 1, 2, 3, 4)$, $G_1||G_3 = (X_{13}, \Sigma_{13}, (x_{01}, x_{03}), \alpha_{13}, X_{m13})$, $G_2||G_4 = (X_{24}, \Sigma_{24}, (x_{02}, x_{04}), \alpha_{24}, X_{m24})$, $G_1 \cong_{\phi_1} G_2$ and $G_3 \cong_{\phi_2} G_4$. Consider $\phi = \{((x_1, x_3), (x_2, x_4)) \in X_{13} \times X_{24} \mid (x_1, x_2) \in \phi_1 \wedge (x_3, x_4) \in \phi_2\}$. For any $((x_1, x_3), (x_2, x_4)) \in \phi$, if there is $(x_1', x_3') \in \alpha_{13}((x_1, x_3), \sigma)$ for any $\sigma \in \Sigma$. We have the following cases. (1) $\sigma \in \Sigma_1 \setminus \Sigma_3$. Then, $x_1 \in \alpha_1(x_1, \sigma)$ and $x_3' = x_3$.

Since $(x_1, x_2) \in \phi_1$, there exists $x_2' \in \alpha_2(x_2, \sigma)$ such that $(x_1', x_2') \in \phi_1$. It implies $(x_2', x_4) \in \alpha_{24}((x_2, x_4), \sigma)$. Thus, $((x_1', x_3), (x_2', x_4)) \in \phi$. (2) $\sigma \in \Sigma_1 \cap \Sigma_2$. Then, $x_1 \in \alpha_1(x_1, \sigma)$ and $x_3' \in \alpha_3(x_3, \sigma)$. Since $(x_1, x_2) \in \phi_1$ and $(x_3, x_4) \in \phi_1$, there exist $x_2' \in \alpha_2(x_2, \sigma)$ and $x_4' \in \alpha_4(x_4, \sigma)$ such that $(x_1', x_2') \in \phi_1$ and $(x_3', x_4') \in \phi_2$, indicating $((x_1', x_3'), (x_2', x_4')) \in \phi$. (3) $\sigma \in \Sigma_3 \setminus \Sigma_1$. The proof is similar to (1). In addition, if $(x_1, x_3) \in X_{m13}$, then $x_1 \in X_{m1}$ and $x_3 \in X_{m3}$, which implies $x_2 \in X_{m1}$ and $x_4 \in X_{m4}$, which in turn implies $(x_2, x_4) \in X_{m24}$. As a result, $G_1||G_3 \cong_{\phi \cup \phi^{-1}} G_2||G_4$.

Theorem 4.1 *Given a distributed plant $G = ||_{i \in I} G_i$ and a specification $R = (Q, \Sigma, q_0, \delta, Q_m)$, there exist $\Sigma_{uci}-$compatible supervisors S_i such that $||_{i \in I}(S_i||G_i) \cong R$ if R is $\{\Sigma_i, G_i, \Sigma_{uci}\}-$separably and synchronously simulation-based state controllable.*

Proof: For $i \in I$, let $S_i = (R_i)_{uci}$. It is obvious that $(R_i)_{uci}$ is $\Sigma_{uci}-$compatible. Moreover, since R_i is synchronously simulation-based state controllable with respect to G_i and Σ_{uci}, we have $G_i||S_i = G||(R_i)_{uci} \cong R_i$. From Proposition 4.1, we obtain $||_{i \in I}(G_i||(R_i)_{uci}) \cong ||_{i \in I} R_i$. Moreover, the fact that R is $\{\Sigma_i\}-$separable implies $||_{i \in I} R_i \cong R$. Therefore, $||_{i \in I}(G_i||S_i) = ||_{i \in I}(G_i||(R_i)_{uci}) \cong ||_{i \in I} R_i \cong R$.

Since Theorem 4.1 is sufficient only, we focus on deterministic supervisors. Accordingly, the following notion is presented to characterize the class of specifications that are achievable by deterministic distributed bisimilarity control.

Definition 4.3 *Given a distributed plant $G = ||_{i \in I} G_i$ and a specification $R = (Q, \Sigma, q_0, \delta, Q_m)$, where $i \in I$, R is $\{\Sigma_i, G_i, \Sigma_{uci}\}-$separably controllable if there exists a group of R_i with alphabet Σ_i such that*

(1) $R \cong ||_{i \in I} R_i$;

(2) $L(R_i)$ *is language controllable with respect to* $L(G_i)$ *and* Σ_{uci}*;*

(3) $G_i||det(R_i) \cong R_i$.

If conditions (1) and (2) are satisfied, R is said to be $\{\Sigma_i, G_i, \Sigma_{uci}\}-$separably language controllable. Then, we provide the necessary and sufficient condition for the existence of deterministic distributed bisimilarity control as below.

Theorem 4.2 *Given a distributed plant* $G = ||_{i\in I}G_i$ *and a specification* $R = (Q, \Sigma, q_0, \delta, Q_m)$*, where* $i \in I$*, there exist* $\Sigma_{uci}-$*compatible deterministic supervisors* S_i *such that* $||_{i\in I}(S_i||G_i) \cong R$ *if and only if* R *is* $\{\Sigma_i, G_i, \Sigma_{uci}\}-$*separably controllable.*

Proof: (Necessity) Suppose there exist $\Sigma_{ui}-$compatible deterministic supervisors S_i such that $||_{i\in I}(S_i||G_i) \cong R$, where $i \in I$. Let $R_i := S_i||G_i$. Then, $R \cong ||_{i\in I}R_i$. Because S_i is $\Sigma_{ui}-$compatible, we obtain that $L(R_i) = L(S_i||G_i)$ is langue controllable with respect to $L(G_i)$ and Σ_{uci}. Furthermore, $G_i||det(R_i) \cong G_i||det(G_i||S_i) \cong G_i||det(G_i)||det(S_i) \cong G_i||det(S_i) \cong G_i||S_i \cong R_i$. Here we have use the facts (1) bisimilarity is preserved when the det operator commutes with parallel composition. (2) $G||det(G) \cong G$. (3) S_i is deterministic. Therefore, R is $\{\Sigma_i, G_i, \Sigma_{uci}\}-$separably controllable.

(Sufficiency) Let $S_i = det(R_i)_{uci}$. Since R_i is langauge controllable with respect to $L(G_i)$ and Σ_{uci}, $S_i = (det(R_i))_{uci}$ is $\Sigma_{uci}-$compatible. From Σ_i-separability of R and Proposition 4.1, we have $||_{i\in I}(G_i||S_i) = ||_{i\in I}(G_i||(det(R_i))_{uci} \cong ||_{i\in I}R_i \cong R$.

4.1.1 Comparison with Monolithic Bisimilarity Control

This subsection compares the proposed existence conditions for distributed bisimilarity control with those for the centralized monolithic bisimilarity control. The following proposition shows that $\{\Sigma_i, G_i, \Sigma_{uci}\}-$separable and synchronous simulation-based state controllability is strictly stronger than the properties of $\{\Sigma_i\}-$separability and $\{G, \Sigma_{uc}\}-$synchronous simulation-based state controllability combined.

Proposition 4.2 *If R is $\{\Sigma_i, G_i, \Sigma_{uci}\}-$separably and synchronously simulation-based state controllable, where $i \in I$, then R is $\{\Sigma_i\}-$separable and synchronously simulation-based state controllable with respect to G and Σ_{uc}.*

Proof: Let $G = ||_{i\in I}G_i = (X, \Sigma, \alpha, x_0, X_m)$ and $R \cong ||_{i\in I}R_i = (Q_{||}, \Sigma_{||}, \delta_{||}, q_{0||}, Q_{m||})$, where $R_i = (Q_i, \Sigma_i, \delta_i, q_{0i}, Q_{mi})$ is synchronously simulation-based state controllable with respect to G_i and Σ_{uci}. Then, there is a synchronous simulation relation ϕ_i such that $R_i \prec_{syn\phi_i} G_i$. Consider $\phi = \{((q_1, q_2, \cdots, q_n), (x_1, x_2, \cdots, x_n)) \in Q_{||} \times X \mid (q_i, x_i) \in \phi_i\}$. It is obvious that ϕ is a synchronous simulation relation such that $||_{i\in I}R_i \prec_{syn\phi} ||_{i\in I}G_i$, i.e., $R \prec_{syn\phi} G$. On the other hand, for any $(q_1, q_2, \cdots, q_n) \in \delta_{||}(q_{0||}, s)$ and $\sigma \in \Sigma_{uc}$, if $s\sigma \in L(G) = \cap_{i\in I}P^{-1}_{\Sigma i}(L(G_i))$, then $P_{\Sigma_i}(s)\sigma \in L(G_i)$ with $\sigma \in \Sigma_{uci} \subseteq \Sigma_i$ and $q_i \in \delta_i(q_{0i}, P_{\Sigma_i}(s))$. Since R_i is state controllable with respect to G_i and Σ_{uci}, we have $\delta_i(q_i, \sigma) \neq \emptyset$, which implies $\delta_{||}((q_1, q_2, \cdots, q_n), \sigma) \neq \emptyset$. As a result, $R \cong ||_{i\in I}R_i$ is $\{\Sigma_i\}-$separable and synchronously simulation-based state controllable with respect to G and Σ_{uc}.

Moreover, $\{\Sigma_i, G_i, \Sigma_{uci}\}-$separable controllability is strictly stronger than the properties of $\{\Sigma_i\}-$separability, $\{G, \Sigma_{uc}\}-$language controllability and $G||det(R) \cong$

R combined, as it is indicated by Proposition 4.3.

Proposition 4.3 *If R is $\{\Sigma_i, G_i, \Sigma_{uci}\}-$separably controllable, then $G||det(R) \cong R$ and R is $\{\Sigma_i\}-$separable and language controllable with respect to G and Σ_{uc}, where $i \in I$.*

Proof: Let $R \cong ||_{i\in I}R_i$, where R_i is language controllable with respect to G_i and Σ_{uci} and $G_i||det(R_i) \cong R_i$. Thus, $L(R) = L(||_{i\in I}R_i)$ is langauge controllable with respect to $L(G)$ and Σ_{uc}. By using the fact that bisimilarity is preserved when the det operator commutes with parallel composition, we have $G||det(R) \cong (||_{i\in I}G_i)||det(||_{i\in I}R_i) \cong (||_{i\in I}G_i)||(||_{i\in I}det(R_i)) \cong ||_{i\in I}(G_i||det(R_i)) \cong ||_{i\in I}R_i \cong R$.

It is known that $\{G, \Sigma_{uc}\}-$synchronous simulation-based state controllability ($\{G, \Sigma_{uc}\}$ -language controllability and $G||det(R) \cong R$) guarantees the existence of a centralized monolithic bisimilarity control (a deterministic centralized monolithic bisimilarity control). Therefore, Proposition 4.2 (Proposition 4.3) implies whenever a specification R can be achieved by distributed bisimilarity control (deterministic distributed bisimilarity control), it can also be achieved by centralized monolithic bisimilarity control (deterministic centralized monolithic control). However, the converse does not hold in general.

4.2 Synthesis of Distributed Bisimilarity Supervisors

This section investigates Problem 2. We first present the synthesis of distributed bisimilarity supervisors based on the result of Theorem 4.1.

Theorem 4.3 *Given a distributed plant* $G = ||_{i\in I}G_i$ *and a specification* $R = (Q, \Sigma, q_0, \delta, Q_m)$*, if* R *is* $\{\Sigma_i, G_i, \Sigma_{uci}\}-$*separably and synchronously simulation-based state controllable such that* $R \cong ||_{i\in I}R_i$ *and* R_i *is* $\{G_i, \Sigma_{uci}\}-$*synchronously simulation-based state controllable, then* $\{S_i = (R_i)_{uci}\}$ *is a set of* $\Sigma_{uci}-$*compatible supervisors such that* $||_{i\in I}(S_i||G_i) \cong R$*, where* $i \in I$*.*

Second, the following theorem provides the synthesis of deterministic distributed bisimilarity supervisors.

Theorem 4.4 *Given a distributed plant* $G = ||_{i\in I}G_i$ *and a specification* $R = (Q, \Sigma, q_0, \delta, Q_m)$*, if* R *is* $\{\Sigma_i, G_i, \Sigma_{uci}\}-$*separably controllable such that* $R \cong ||_{i\in I}R_i$*,* R_i *is* $\{G_i, \Sigma_{uci}\}-$*language controllable and* $G_i||det(R_i) \cong R_i$*, then* $\{S_i = (det(R_i))_{uci}\}$ *is a set of deterministic* $\Sigma_{uci}-$*compatible supervisors such that* $||_{i\in I}(S_i||G_i) \cong R$*, where* $i \in I$*.*

Next, we present an example to illustrate bisimilarity control of distributed discrete event systems.

Example 4.1. Consider a cooperative multi-robot system (MRS) configured in Fig. 4.1 (Left). The MRS consists of two robots R_1 and R_2. Both of them

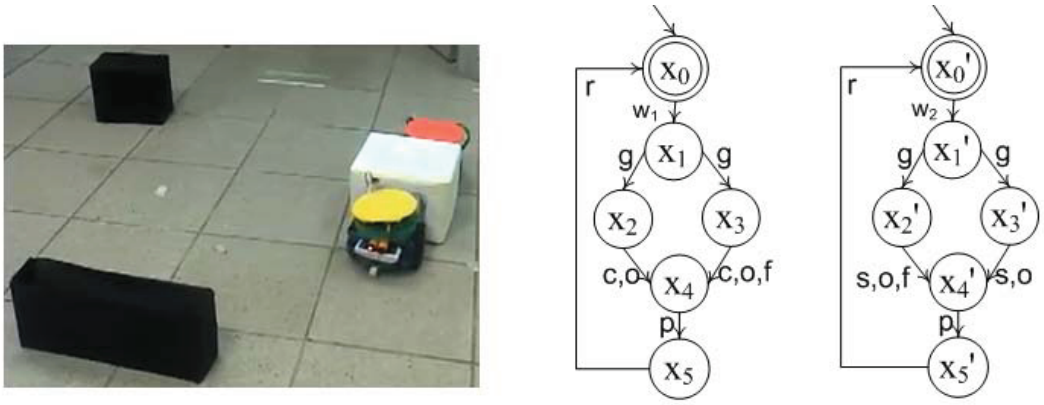

Figure 4.1: Multi-robot system (MRS) (Left), G_1 (Middle) and G_2(Right)

have the same communication, position, pushing, scent-sensing and frequency-sensing capabilities. Furthermore, R_1 has color-sensing capabilities, while R_2 has shape-sensing capability. R_1 and R_2 can cooperatively search and clear a dangerous object (the white cube) in the workspace. Initially, R_1 and R_2 are positioned outside the workspace. Let $i = 1, 2$. When the work request announces (event w_i), R_i is required to enter the workspace. Due to actuator limitations, it nondeterministically goes along one of two pre-defined paths (event g). In the first path, R_1 activates color-sensing (event c) or scent-sensing (event o) capabilities to detect the dangerous object; whereas in the second path, R_1 activates color-sensing, scent-sensing capabilities or frequency-sensing (event f) capabilities for detection.

Similarly, R_2 activates shape-sensing (event s), scent-sensing or frequency-sensing capabilities in the first path, while in the second path it activates shape-sensing or scent-sensing capabilities. After detecting the dangerous object, R_i pushes the dangerous object outward the workspace (event p), and then returns to the initial position (event r) for the next implementation.

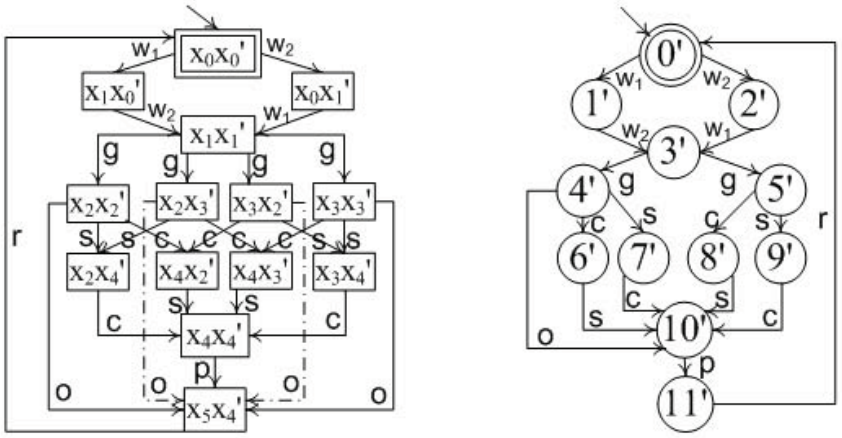

Figure 4.2: $G_1||G_2$ (Left) and R (Right)

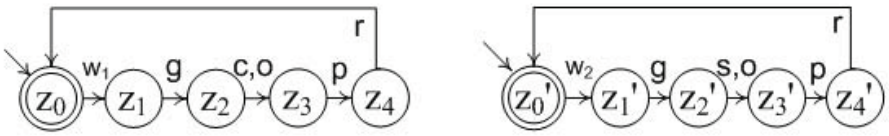

Figure 4.3: S_1 (Left) and S_2 (Right)

The automaton model G_i of R_i with alphabet Σ_i is shown in Fig. 4.1, where $\Sigma_1 = \{w_1, g, c, o, f, p, r\}$ and $\Sigma_2 = \{w_2, g, s, o, f, p, r\}$. Since R_i cannot disable the broadcast from the host computer, the event w_i is deemed uncontrollable, that is $w_i \in \Sigma_{uci}$. The rest of events are controllable. The cooperative behavior of R_1 and R_2 can be represented by $G_1||G_2$ (Fig. 4.2(Left)), and the specification R (Fig. 4.2 (Right)) is given in order to restrict the cooperative behavior $G_1||G_2$.

According to the specification, after R_1 and R_2 receive the work command and go to the workspace, two possible states are reached by the MRS. In the first state, either the color sensor, the shape sensor or the scent sensors are adopted to confirm

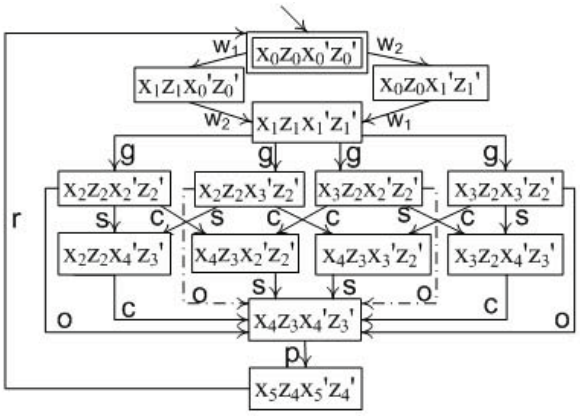

Figure 4.4: $||_{i\in\{1,2\}}G_i||S_i$

an objective is dangerous. However, to save the energy, in the second state only the color sensor and the shape sensor can be adopted for dangerous object detection. After the detection, the dangerous object is cleared from the workspace.

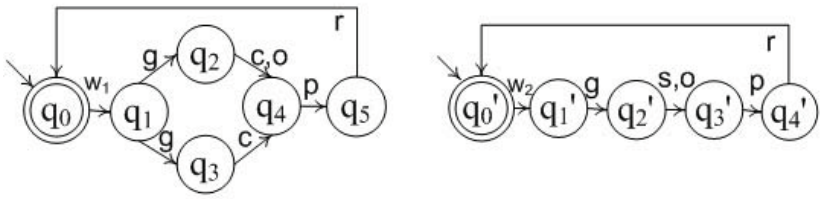

Figure 4.5: R_{s_1} (Left) and R_{s_2} (Right)

For such a MRS, if we use language equivalence as behavior equivalence, the control target is to design supervisors S_1 and S_2 such that $L(\|_{i\in\{1,2\}}\ G_i||S_i) = L(R)$. Consider $\Sigma = \Sigma_1 \cup \Sigma_2$. According to the results in [67], this problem can be solved by designing S_i such that $L(G_i||S_i) = P_{\Sigma_i}(L(R))$. Since $P_{\Sigma_i}(L(R))$ is language controllable with respect to $L(G_i)$ and Σ_{uci}, we can construct S_i as shown in Fig. 4.3. So the supervised system $||_{i\in\{1,2\}}G_i||S_i$ (Fig. 4.4) is language equivalent to $L(R)$.

However, it can be seen that $||_{i\in\{1,2\}}G_i||S_i$ violates the energy saving requirement in the specification.

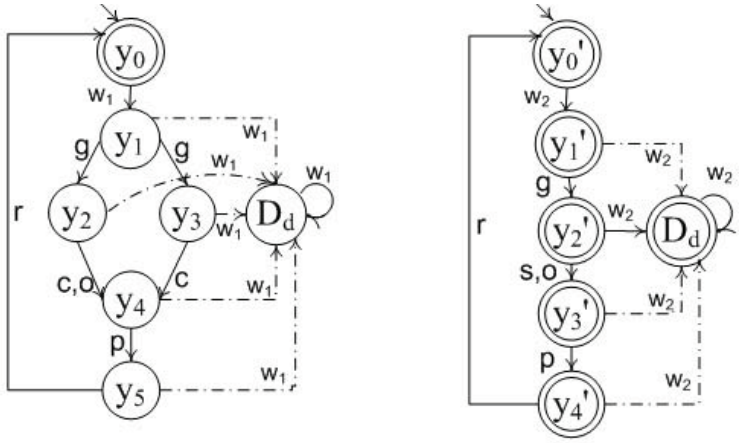

Figure 4.6: S_1' (Left) and S_2' (Right)

Hence, langauge equivalence is not adequate for this case, which calls for the use of bisimulation equivalence instead. That is, we need design supervisor S_i' such that $||_{i\in\{1,2\}}G_i||S_i' \cong R$. By using the proposed techniques in this chapter, we firstly decompose the global specification R into sub-specifications R_{s_i} with alphabet Σ_i for R_i (Fig. 4.5) such that $||_{i\in\{1,2\}}R_{s_i} \cong R$. Secondly, if we can design S_i' such that $G_i||S_i' \cong R_{s_i}$, then $||_{i\in\{1,2\}}G_i||S_i' \cong R$.

Since R_{s_1} is synchronously simulation-based state controllable with respect to G_2 and $\Sigma_{uc1} = \{w_1\}$, we design $S_1' = (R_{s_1})_{uc}$ (Fig. 4.6) according to Theorem 2.7. Then, $G_1||S_1' \cong_{\phi\cup\phi^{-1}} R_{s_1}$, where $\phi = \{(q_0,(x_0,y_0)),(q_1,(x_1,y_1)),(q_2,(x_2,y_2)),(q_2,(x_3,y_2)),(q_3,(x_2,y_3)),(q_3,(x_3,y_3)),(q_4,(x_4,y_4)),(q_5,(x_5,y_5))\}$. On the other hand, R_{s_2} is deterministic and synchronously simulation-based controllable with respect to G_2 and $\Sigma_{uc2} = \{w_2\}$. Theorem 2.5 indicates that we can construct $S_2' = (R_{s_2})_{uc2}$

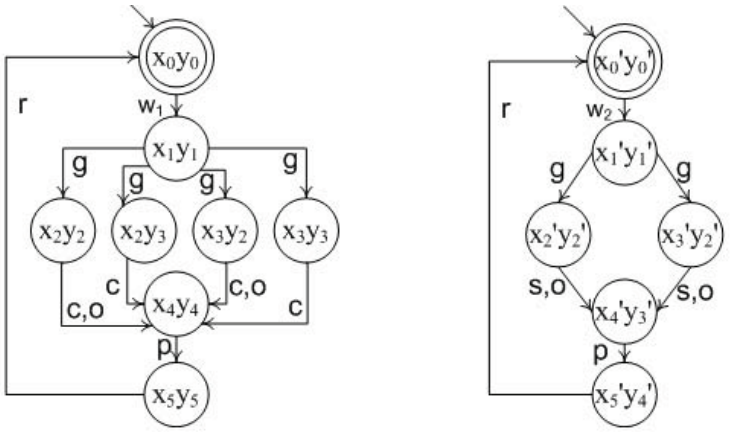

Figure 4.7: $G_1||S_1'$ (Left) and $G_2||S_2'$ (Right)

(Fig. 4.6). The supervised system $G_2||S_2'$ is shown in Fig. 4.7. It can be seen that $G_2||S_2' \cong_{\phi_1 \cup \phi_1^{-1}} R_{s_2}$, where $\phi_1 = \{(q_0', (x_0', y_0')), (q_1', (x_1', y_1')), (q_2', (x_2', y_2')), (q_2', (x_3', y_2')), (q_3', (x_4', y_3')), (q_4', (x_5', y_4'))\}$. As a result, $||_{i \in \{1,2\}} G_i||S_i' \cong R$.

4.3 Synthesis of Achievable Sub-specifications /Sup-specifications

This section investigates the synthesis of achievable sub-specifications/sup-specifications to enable the existence of a distributed bisimilarity control/a deterministic distributed bisimilarity control in case of the given specification is not $\{\Sigma_i, G_i, \Sigma_{uci}\}-$separably and synchronously simulation-based state controllable/$\{\Sigma_i, G_i, \Sigma_{uci}\}-$separably controllable. We present the classes of achievable sub/sup-specifications for distributed

bisimilarity control as follows.

$$DC_1(R) := \{R' \mid R' \prec R \text{ and } R' \text{ is } \{\Sigma_i, G_i, \Sigma_{uci}\} - \textit{separably and synchronouly simulation} - \textit{based state controllable}\};$$

$$\overline{DC_1(R)} := \{R' \mid R \prec R' \text{ and } R' \text{ is } \{\Sigma_i, G_i, \Sigma_{uci}\} - \textit{separably and synchronouly simulation} - \textit{based state controllable}\};$$

Some preliminary results are presented before we give the main results of this section.

Lemma 4.1 *Given $G_i = (X_i, \Sigma, \alpha_i, x_{0i}, X_{mi})$, where $i \in \{1, 2, 3, 4\}$, if $G_1 \prec G_2$ and $G_3 \prec G_4$, then $G_1||G_3 \prec G_2||G_4$.*

Proof: Let $G_1 \prec_{\phi_1} G_2$, $G_3 \prec_{\phi_2} G_4$, $G_1||G_3 = (X_{13}, \Sigma, \alpha_{13}, x_{013}, X_{m13})$ and $G_2||G_4 = (X_{24}, \Sigma, \alpha_{24}, x_{024}, X_{m24})$. Consider a relation $\phi = \{((x_1, x_3), (x_2, x_4)) \mid (x_1, x_2) \in \phi_1 \wedge (x_3, x_4) \in \phi_2\}$. Let $((x_1, x_3), (x_2, x_4)) \in \phi$. It is obvious that $((x_{01}, x_{03}), (x_{02}, x_{04})) \in \phi$. If there exists $(x_1', x_3') \in \alpha_{13}((x_1, x_3), \sigma)$ for any $\sigma \in \Sigma$, we have $x_1' \in \alpha_1(x_1, \sigma)$ and $x_3' \in \alpha_3(x_3, \sigma)$. Since $(x_1, x_2) \in \phi_1$ and $(x_3, x_4) \in \phi_2$, there exist $x_2' \in \alpha_2(x_2, \sigma)$ and $x_4' \in \alpha_4(x_4, \sigma)$ such that $(x_1', x_2') \in \phi_1$ and $(x_3', x_4') \in \phi_2$. Then, $(x_2', x_4') \in \alpha_{24}((x_2, x_4), \sigma)$ and $((x_1', x_3), (x_2, x_4)) \in \phi$. Thus, ϕ is a simulation relation from $G_1||G_3$ to $G_2||G_4$.

The following lemma follows from Lemma 4.1.

Lemma 4.2 *Given $G_i = (X_i, \Sigma, \alpha_i, x_{0i}, X_{mi})$, where $i \in 1, 2, 3$, if $G_1 \prec G_2$ and $G_1 \prec G_3$, then $G_1 \prec G_2||G_3$.*

Proof: Let $G_1 \prec_{\phi_1} G_2$ and $G_1 \prec_{\phi_2} G_3$. It is obvious that $\phi = \{(x_1, (x_2, x_3)) \mid (x_1, x_2) \in \phi_1 \wedge (x_1, x_3) \in \phi_2\}$ is a simulation relation from G_1 to $G_2||G_3$.

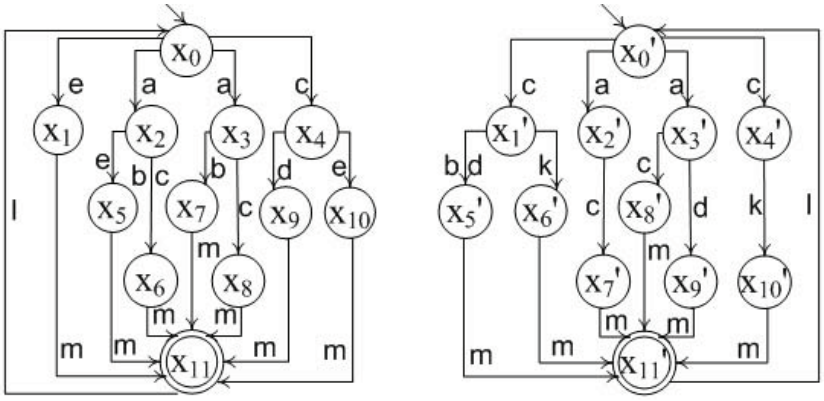

Figure 4.8: G_1 (Left) and G_2 (Right)

Then, the synthesis of achievable sub-specifications is presented as below. Here we define $S(R) = \{R' \prec R \mid R' \textit{ is } \{\Sigma_i\}\text{-separable}\}$ as the set of sub-specifications satisfying the property of separability.

Theorem 4.5 *Given a distributed plant $G = ||_{i \in I} G_i$ and a specification R, if there exists $R' \in S(R)$ with $R' \cong ||_{i \in I} R'_i$ such that $F^{j+1}_{cG}(F_{syn}(G_i)||R'_i) = F^{j}_{cG}(F_{syn}(G_i)||R'_i) \neq \emptyset$, then $||_{i \in I} F^{j}_{cG}(F_{syn}(G_i)||R'_i) \in DC_1(R)$, where $i \in I$ and $j \in N^+$.*

Proof: Since $F^{j+1}_{cG}(F_{syn}(G_i)||R'_i) = F^{j}_{cG}(F_{syn}(G_i)||R'_i) \neq \emptyset$, we obtain that $F^{j}_{cG}(F_{syn}(G_i)||R'_i)$ is synchronously simulation-based state controllable with respect to G_i and Σ_{uci}. Moreover, according to Lemma 4.1 and Lemma 4.2 $F^{j}_{cG}(F_{syn}(G_i)||R'_i) \prec R'_i$ implies $||_{i \in I} F^{j}_{cG}(F_{syn}(G_i)||R'_i) \prec ||_{i \in I} R'_i \cong R' \prec R$. We conclude that $||_{i \in I} F^{j}_{cG}(F_{syn}(G_i)||R'_i) \in DC_1(R)$.

We can see that if the condition of Theorem 4.5 holds, $||_{i \in I} F^{j}_{cG}(F_{syn}(G_i)||R'_i)$ is

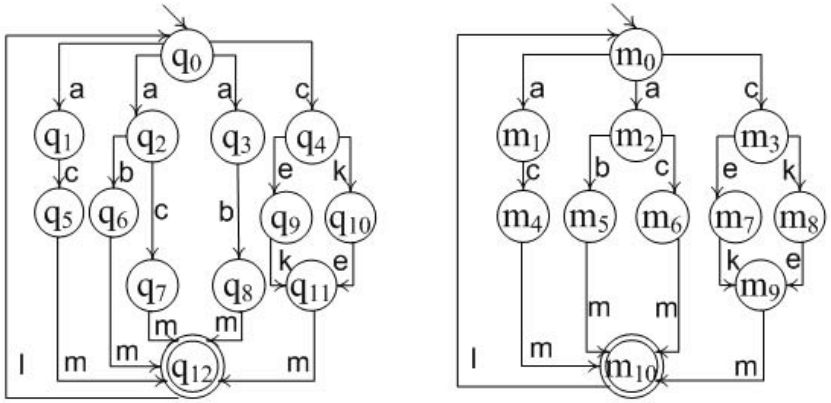

Figure 4.9: R (Left) and R' (Right)

an achievable sub-specification for distributed bisimilarity control. This synthesis method is illustrated by the following example.

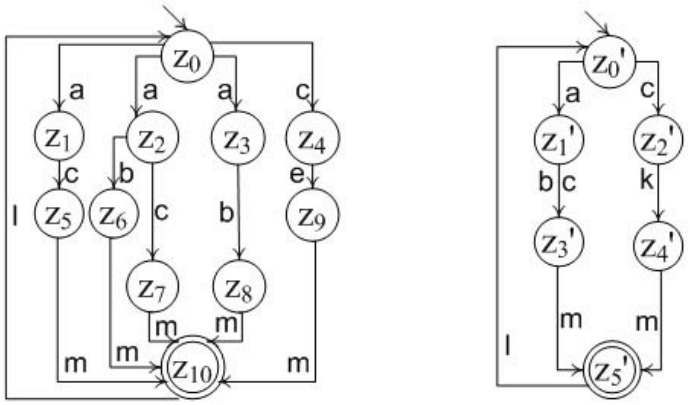

Figure 4.10: R_1 (Left) and R_2 (Right)

Example 4.2. Consider a set of plants $G_i = (X_i, \Sigma_i, \alpha_i, x_{0i}, X_{mi})$ (Fig. 4.8) and a specification R (Fig. 4.9 (Left)), where $i \in \{1, 2\}$, $\Sigma_1 = \{a, b, c, d, e, m, l\}$, $\Sigma_2 = \{a, b, c, d, k, m, l\}$, $\Sigma_{uc1} = \{c\}$ and $\Sigma_{uc2} = \{b, c\}$. It is obvious that R is not $\{\Sigma_i, G_i, \Sigma_{uci}\}-$separably and synchronously simulation-based state controllable. Thus, there does not exist a distributed bisimilarity control with respect to G_i

and R according to Theorem 4.1. To address this problem, we would like to find a sub-specification of R, which satisfies $\{\Sigma_i, G_i, \Sigma_{uci}\}-$separable and synchronous simulation-based state controllability.

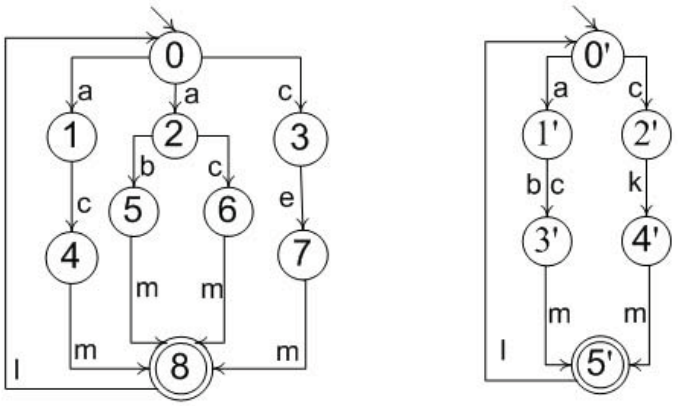

Figure 4.11: R_1' (Left) and R_2' (Right)

First, we decompose the global specification R into local specifications R_1 with alphabet Σ_1 and R_2 with alphabet Σ_2 such that $R_1||R_2 \cong R$, which are shown in Fig. 4.10. By using Theorem 4.5, we obtain that $F^2_{cG}(F_{syn}(G_1)||R_1) = F_{cG}(F_{syn}(G_1)||R_1) \cong R_1'$ and $F_{cG}(F_{syn}(G_2)||R_2) = R_2 \cong R_2'$, where R_1' and R_2' are presented in Fig. 4.11. Therefore, the achievable sub-specification $R' \cong R_1'||R_2' \in DC_1(R)$ is obtained in Fig. 4.9 (Right).

Subsequently, we give the classes of achievable sup-specifications for deterministic distributed bisimilarity control.

$$DC_2(R) := \{R' \mid R' \prec R \text{ and } R' \text{ is } \{\Sigma_i, G_i, \Sigma_{uci}\} - \text{separable and controllable}\};$$

$$\overline{DC_2(R)} := \{R' \mid R \prec R' \text{ and } R' \text{ is } \{\Sigma_i, G_i, \Sigma_{uci}\} - \text{separable and controllable}\};$$

Then, the following theorem is presented to show the synthesis of infimal sup-

specifications for deterministic distributed bisimilarity control.

Theorem 4.6 *Given a distributed plant $G = ||_{i\in I}G_i$ and a specification R, if there exists a set of $\{R_i\}$ such that $R \cong ||_{i\in I}R_i$, $R_i \prec G_i$ and $L(R_i)$ is language controllable with respect to $L(G_i)$ and Σ_{uci}, then $||_{i\in I}(G_i||det(R_i)) \in inf\overline{DC_2(R)}$, where $i \in I$.*

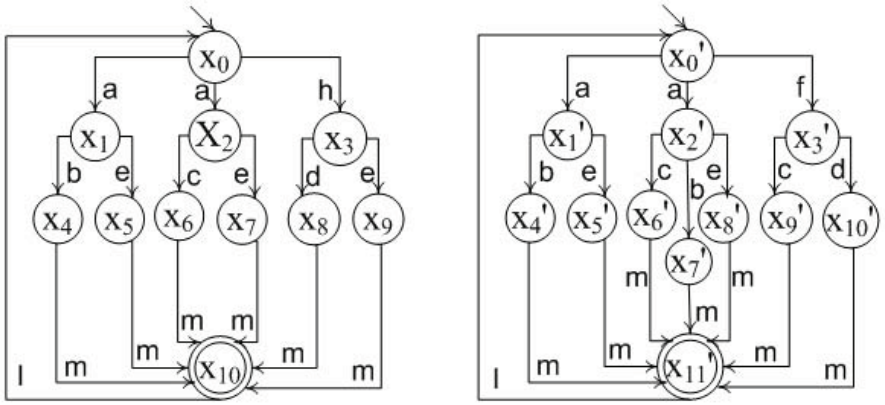

Figure 4.12: G_1 (Left) and G_2 (Right)

Proof: Because commuting the det operator with parallel composition preserves bisimilarity, we have $G_i||det(G_i||det(R_i)) \cong G_i||det(G_i)||det(R_i) \cong G_i||det(R_i)$ for $i \in I$. Moreover, $L(G_i||det(R_i)) = L(G_i)\cap L(det(R_i)) = L(G_i)\cap L(R_i) = L(R_i)$ and $L(R_i)$ is $\{L(G_i), \Sigma_{uci}\}-$controllable. Thus, $L(G_i||det(R_i))$ is $\{L(G_i), \Sigma_{uci}\}-$controllable. In addition, according to Lemma 4.2 $R_i \prec det(R_i)$ and $R_i \prec G_i$ implies $R_i \prec G_i||det(R_i)$, which in turns implies $||_{i\in I}R_i \prec ||_{i\in I}(G_i||det(R_i))$ by using Lemma 4.1. Since $R \cong ||_{i\in I}R_i$, we obtain $R \prec ||_{i\in I}(G_i||det(R_i))$. Therefore, $||_{i\in I}(G_i||det(R_i)) \in \overline{DC_2(R)}$.

Next, we would like to show $||_{i\in I}(G_i||det(R_i)) \prec R_1$ holds for any $R_1 \in \overline{DC_2}$. Since R_1 is $\{\Sigma_i, G_i, \Sigma_{uci}\}$-separably controllable, there exists a set of $\{R_{1i}\}$ for $i \in I$ such that $R_1 \cong ||_{i\in I}R_{1i}$ and $G_i||det(R_{1i}) \cong R_{1i}$. Thus, $R_1 \cong ||_{i\in I}(G_i||det(R_{1i}))$. It suffices

to show that $||_{i\in I}det(R_i) \prec ||_{i\in I}det(R_{1i})$. This holds because the following facts: (1) $||_{i\in I}det(R_i)$ and $||_{i\in I}det(R_{1i})$ are deterministic; (2) $R \prec R_1$ implies $L(R) \subseteq L(R_1)$, i.e.,$L(||_i R_i) \subseteq L(||_i R_{1i})$, which in turn implies $L(||_{i\in I}det(R_i)) \subseteq L(||_{i\in I}det(R_{1i}))$.

It follows that when the condition of Theorem 4.6 is satisfied, $||_{i\in I}(G_i||det(R_i))$ is an infimal achievable sup-specification that can be synthesized by deterministic distributed bisimilarity control. Moreover, the result of Theorem 4.6 is demonstrated by the following example.

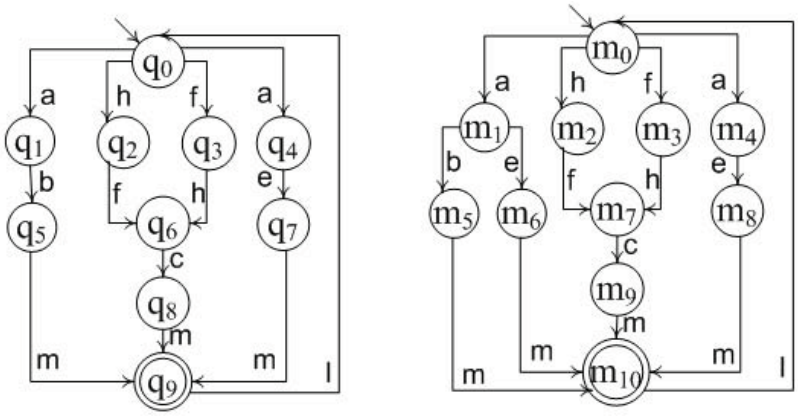

Figure 4.13: R (Left) and R' (Right)

Example 4.3. Consider plants $G_i = (X_i, \Sigma_i, \alpha_i, x_{0i}, X_{mi})$ (Fig. 4.12) and a specification R (Fig. 4.13 (Left)), where $i \in \{1, 2\}$, $\Sigma_1 = \{a, b, c, d, e, h, m, l\}$, $\Sigma_2 = \{a, b, c, d, e, f, m, l\}$, $\Sigma_{uc1} = \{e\}$ and $\Sigma_{uc2} = \{a\}$.

We can see that R is not $\{\Sigma_i, G_i, \Sigma_{uci}\}-$separably controllable. It results in the nonexistence of deterministic distributed bisimilarity control from the result of Theorem 4.2. For this issue, the calculation of sup-specifications satisfying $\{\Sigma_i, G_i, \Sigma_{uci}\}-$separable controllability with respect to R is presented as follows.

The specification R is firstly decomposed into R_1 and R_2 (Fig. 4.14) such that

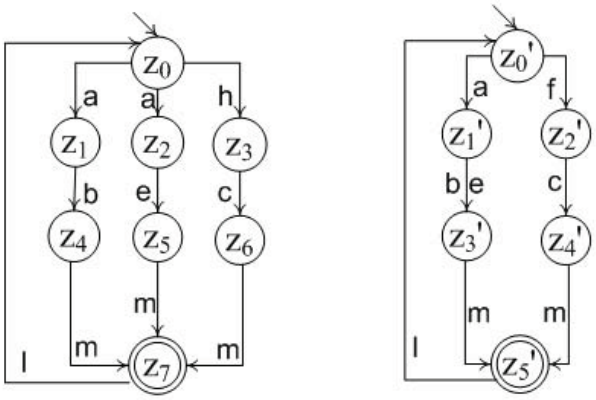

Figure 4.14: R_1 (Left) and R_2 (Right)

$R_1||R_2 \cong R$ and $L(R_i)$ is language controllable with respect to $L(G_i)$ and Σ_{uci}. Then, we have $R_1' \cong G_1||det(R_1)$ and $R_2' \cong G_2||det(R_2)$. By using Theorem 4.6, we further obtain that $R' \cong R_1'||R_2' \in inf\overline{DC_2(R)}$, where R_1' and R_2' are shown in Fig. 4.15. That is, R' (Fig. 4.13 (Right)) is an infimal achievable sup-specification of R for deterministic distributed bisimilarity control.

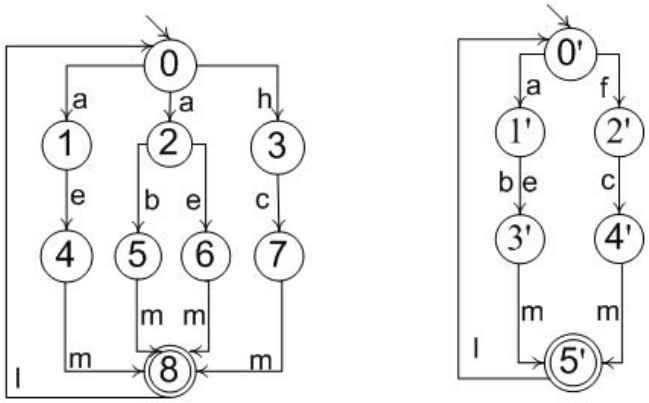

Figure 4.15: R_1' (Left) and R_2' (Right)

4.3.1 Comparison with Monolithic Bisimilarity Control

This subsection compares the upper and lower bounds of the behavior achieved by distributed bisimilarity control/deterministic distributed bisimilarity control with those are for monolithic bisimilarity control. Firstly, the classes of achievable sub/sup-specifications for monolithic bisimilarity control are presented.

$$\begin{aligned} DC_1'(R) &:= \{R' \mid R' \prec R \text{ and } R' \text{ is synchronouly simulation} - \text{based} \\ &\qquad \text{state controllable w.r.t. } G \text{ and } \Sigma_{uc}\}; \\ \overline{DC_1'(R)} &:= \{R' \mid R \prec R' \text{ and } R' \text{ is synchronouly simulation} - \text{based} \\ &\qquad \text{state controllable w.r.t. } G \text{ and } \Sigma_{uc}\}; \end{aligned}$$

It is known that an element of $supDC_1'(R)$ ($inf\overline{DC_1'(R)}$) provides an upper bound (a lower bound) to achievable sub-specifications (sup-specifications) for monolithic bisimilarity control. Denote $maxDC_1(R)$ and $min\overline{DC_1(R)}$ as the set of maximal elements of $DC_1(R)$ and the set of minimal elements of $\overline{DC_1(R)}$. Then, necessary and sufficient conditions are presented to show the relationships between $supDC_1'(R)$ and $maxDC_1(R)$ (resp. $infDC_1'(R)$ and $minDC_1(R)$).

Proposition 4.4 *Consider a distributed plant $G = ||_{i\in I}G$ and a specification R, where $i \in I$. For any $A \in supDC_1'(R)$, $A \in maxDC_1(R)$ if and only if A is $\{\Sigma_i, G_i, \Sigma_{uci}\}$-separably and synchronously simulation-based state controllable.*

Proof: (Necessity) Since $A \in maxDC_1(R)$, we have $A \in DC_1(R)$, which implies A is $\{\Sigma_i, G_i, \Sigma_{uci}\}$-separably and synchronously simulation-based state controllable.

(Sufficiency) Because A is $\{\Sigma_i, G_i, \Sigma_{uci}\}$-separably and synchronously simulation-based state controllable, we have $A \in DC_1(R)$. Suppose there exists $A_1 \in DC_1(R)$ such that $A \prec A_1$. Proposition 4.2 indicates that A_1 is synchronously simulation-based state controllable with respect to G and Σ_{uc}. Then, $A_1 \in DC_1'(R)$. According to the definition of supremum, we obtain $A_1 \prec A$, which introduces a contradiction. Therefore, the assumption is not correct. As a result, $A \in maxDC_1(R)$.

Proposition 4.5 *Consider a distributed plant $G = ||_{i\in I}G_i$ and a specification R, where $i \in I$. For any $A \in inf\overline{DC_1'(R)}$, $A \in min\overline{DC_1(R)}$ if and only if A is $\{\Sigma_i, G_i, \Sigma_{uci}\}$-separably and synchronously simulation-based state controllable.*

Proof: The proof is similar to Proposition 4.4.

Proposition 4.4 and Proposition 4.5 indicate that under the condition of $\{\Sigma_i, G_i, \Sigma_{uci}\}$-separable and synchronous simulation-based state controllability, an element in $supDC_1'(R)$ ($inf\overline{DC_1'(R)}$) is an upper bound (a lower bound) which can be achieved by restrictive distributed bisimilarity control (relaxer distributed bisimilarity control). Next, we pay attention to deterministic control. The classes of achievable sup-specifications for monolithic deterministic bisimilarity control are stated as below.

$$\begin{aligned} DC_2'(R) &:= \{R' \mid R' \prec R, R' \textit{ is language controllable w.r.t. } G \textit{ and } \Sigma_{uc}, \textit{and} \\ &\qquad G||det(R') \cong R\}; \\ \overline{DC_2'(R)} &:= \{R' \mid R \prec R', R' \textit{ is language controllable w.r.t. } G \textit{ and } \Sigma_{uc}, \textit{and} \\ &\qquad G||det(R') \cong R\}; \end{aligned}$$

Similar results can also be obtained with respect to deterministic distributed bisimilarity control.

Proposition 4.6 *Consider a distributed plant* $G = ||_{i \in I} G_i$ *and a specification* R, *where* $i \in I$. *For any* $A \in supDC_2'(R)$, $A \in maxDC_2(R)$ *if and only if* A *is* $\{\Sigma_i, G_i, \Sigma_{uci}\}$*-separably controllable.*

Proof: (Necessity) Since $A \in maxDC_2(R)$, we get $A \in DC_2(R)$, which implies A is $\{\Sigma_i, G_i, \Sigma_{uci}\}$-separably controllable.

(Sufficiency) Since A is $\{\Sigma_i, G_i, \Sigma_{uci}\}$-separable controllable, we obtain that $G||det(A) \cong A$ and $L(A)$ is language controllable with respect to $L(G)$ and Σ_{uc} from Proposition 4.3. Thus, we have $A \in DC_2(R)$. Suppose there exists $A_1 \in DC_2(R)$ such that $A \prec A_1$. Proposition 4.3 implies $G||det(A_1) \cong A_1$ and $L(A_1)$ is language controllable with respect to $L(G)$ and Σ_{uc}, which in turn implies $A_1 \in DC_2'(R)$. Definition of supremum indicates $A_1 \prec A$, introducing a contradiction. Then, the assumption is not correct. It follows that $A \in maxDC_2(R)$.

Proposition 4.7 *Consider a distributed plant* $G = ||_{i \in I} G_i$ *and a specification* R, *where* $i \in I$. *For any* $A \in inf\overline{DC_2'(R)}$, $A \in min\overline{DC_2(R)}$ *if and only if* A *is* $\{\Sigma_i, G_i, \Sigma_{uci}\}$*-separably controllable.*

4.4 Conclusion

We investigated supervisory control of distributed discrete event systems for bisimulation equivalence in this chapter. The notion of separable and synchronous simulation-based state controllability, which combines separability with synchronous simulation-based state controllability, was introduced for the existence of distributed bisimilarity

control. When the given specification satisfies this condition, a set of local supervisors enforcing bisimilarity can be constructed. Otherwise, we further explored the calculation of achievable sub-specifications which enable the existence of a distributed bisimilarity control. In particular, we focused on deterministic supervisors for distributed bisimilarity control. Accordingly, the notion of separable controllability was introduced as the necessary and sufficient condition for the existence of a set of deterministic local supervisors with respect to distributed bisimiarity control. In addition, the synthesis of deterministic supervisors and achievable sup-specifications were investigated. The comparisons of our results with the centralized monolithic ones were presented as well.

Chapter 5

Control of Multi-Affine Systems for Bisimulation Equivalence

This chapter studies bisimilarity control of a particular class of nonlinear systems, multi-affine systems. This kind of continuous dynamics is widely used in system modelling. The celebrated Euler [89], Volterra [90] and Lotka-Volterra [91] equations, the control systems for aircraft and underwater vehicles [92] and the models of genetic regulatory networks [93] are examples of multi-affine systems. First, we partition the state space into rectangles. Then, we investigate the control of multi-affine systems on rectangles, including the control based on the exit sub-region to drive all trajectories starting from a rectangle to exit through a facet and the control to stabilize the multi-affine system towards a desired point. With the proposed controllers, a finitely abstracted transition system is constructed, and it is shown to be bisimilar to the rectangular transition system of the multi-affine system. Since bisimulation preserves temporal logic properties, the controller synthesis of the multi-affine system for tem-

poral logic specifications is achieved by designing a bisimilarity enforcing supervisor for the abstracted transition system and by implementing the resulting supervisor to the original multi-affine system. We start by reviewing the notions of multi-affine function and multi-affine control system [81].

Definition 5.1 *A function* $f = (f_1, f_2, ..., f_m) : R^n \to R^m$ *(with* $n, m \in N$*) is said to be multi-affine, if every* $f_i(x) : R^n \to R$*, where* $x = (x_1, x_2, \cdots, x_n)$ *and* $i = 1, \cdots, m$*, is a polynomial in the indeterminates* $x_1, x_2, \cdots, x_n$*, with the property that the degree of* f_i *in any of indeterminates* $x_1, x_2, \cdots, x_n$ *is less or equal to 1. That is,* f *has the form*

$$f(x) = f(x_1, x_2, \cdots, x_n) = \sum_{i_1, \cdots, i_n \in \{0,1\}} c_{i_1 i_2 \cdots i_n} (x_1)^{i_1} (x_2)^{i_2} \cdots (x_n)^{i_n}$$

where $c_{i_1 i_2 \cdots i_n} \in R^m$ *for all* $i_1, i_2, \cdots, i_n \in \{0, 1\}$.

For example, for $n = 2$ and arbitrary m, the multi-affine function is in terms of $f(x_1, x_2) = c_{00} + c_{10}x_1 + c_{01}x_2 + c_{11}x_1x_2$, where $c_{ij} \in R^m$ for $i, j \in \{0, 1\}$.

Definition 5.2 *A control system* $\Sigma : \dot{x} = f(x, u) = g(x) + Bu$ *with* $B \in R^{n \times m}$ *is said to be multi-affine if* $g : R^n \to R^n$ *is a multi-affine function.*

For a multi-affine control system, we write $\chi_{x_0,u}(t)$ to denote the point reached at time t under the control input u from initial condition x_0.

5.1 Rectangular Partition

In this chapter, the state space of the multi-affine system is assumed to be bounded and rectangular, which holds in lots of engineering applications [81, 94]. Given such a

state space, we would like to rectangularly partition it with respect to the coordinates. Then, the following concepts are provided.

An n-rectangle is described by $E = \prod_{i=1}^{n}(a_i, b_i)$, where $a_i, b_i \in \mathbb{R}$ satisfy $a_i < b_i$ for $i = 1, 2, \cdots, n$. The closure of E is defined as $\overline{E}=\prod_{i=1}^{n}[a_i, b_i]$. A facet of E is the intersection of $\overline{E}$ with one of its supporting hyperplanes. The set of facets of E is denoted by $F(E)$. The set of vertices of E, denoted by $V(E)$, is $V(E)=\{(x_1, x_2, \cdots, x_n) \mid x_i \in \{a_i, b_i\}, i = 1, 2, \cdots, n\}$. Given $v \in V(E)$, we denote $F(v)$ the set of all facets containing v.

The state space can be partitioned into $\prod_{i=1}^{n} n_i$ rectangles as follows. Let $x_i \in \bigcup_{j=1}^{n_i}(a_i^j, b_i^j)$, where $a_i^j < b_i^j$ and $a_i^{j+1} = b_i^j$. Then, $R_{k_1k_2\cdots k_n} = \prod_{i=1}^{n}(a_i^{k_i}, b_i^{k_i})$ is a rectangle in the partitioned state space, where $1 \leq k_i \leq n_i$. The facet of $R_{k_1k_2\cdots k_n}$ is described by

$$F^{j,d}_{k_1k_2\cdots k_n} = \begin{cases} \overline{R_{k_1k_2\cdots k_n}} \bigcap \left\{x \in R^n \mid x_j = b_j^{k_j}\right\} \ if \ d = + \\ \overline{R_{k_1k_2\cdots k_n}} \bigcap \left\{x \in R^n \mid x_j = a_j^{k_j}\right\} \ if \ d = - \end{cases}$$

where $d \in \{+, -\}$ and $j = 1, \cdots, n$.

The outer normal of $F^{j,d}_{k_1k_2\cdots k_n}$ is given by

$$n^{j,d} = \begin{cases} e_j^\top \ \ if \ \ d = + \\ -e_j^\top \ \ if \ \ d = - \end{cases}$$

where $d \in \{+, -\}$, $j = 1, \cdots, n$ and e_j is the Euclidian basis of R^n.

Given $w = (w_1, w_2, \cdots, w_n) \in V(R_{k_1k_2\cdots k_n})$, the vertex membership function $S : \{w_1, \cdots, w_n\} \to \{0, 1\}$ is defined as

$$S(w_j) = \begin{cases} 1 \ if \ w_j = b_j^{k_j} \\ 0 \ if \ w_j = a_j^{k_j} \end{cases}$$

Denote ξ as the set of rectangles generated by rectangularly partitioning the state space. The rectangular projection map $\pi_Q : R^n \to \xi$ is defined as $\pi_Q(x) = \{R_{k_1k_2\cdots k_n} \in \xi \mid x \in R_{k_1k_2\cdots k_n}\}$. Subsequently, the rectangular transition system is established, and it can be understood as a transition system form of the multi-affine control system over a rectangularly partitioned state space.

Definition 5.3 *Given a multi-affine control system $\Sigma : \dot{x} = g(x)+Bu$, a rectangle set ξ generated by rectangularly partitioning the state space and a rectangular projection map π_Q defined by ξ, the rectangular transition system of Σ associated with ξ, denoted as $S_{\Sigma,Q}$, is a tuple*

$$S_{\Sigma,Q} = (X_Q, X_{Q0}, U_Q, \to_Q, X_{mQ})$$

- $X_Q = R^n = X_{mQ}$;
- $X_{Q0} = \{x \mid x$ *is an initial state of the multi-affine control system*$\}$;
- $U_Q = \{U_{R_{k_1k_2\cdots k_n}} \mid U_{R_{k_1k_2\cdots k_n}}$ *is an invariant controller w.r.t.* $R_{k_1k_2\cdots k_n}$ *or an exit controller w.r.t.* $F^{j,d}_{k_1k_2\cdots k_n}$, $R_{k_1k_2\cdots k_n} \in \xi$ *and* $F^{j,d}_{k_1k_2\cdots k_n}\}$;
- $x \xrightarrow{U_{\pi_Q(x)}}_Q x'$ *if any of the following two conditions is satisfied:*

  (1) $\pi_Q(x) = \pi_Q(x')$ *holds and there exists* $\tau \in R^+$ *such that* $\chi_{x,k(x)}(\tau) = x'$ *and* $\pi_Q(\chi_{x,k(x)}(t)) = \pi_Q(x)$, *where* $t \in [0, +\infty)$.

  (2) $\pi_Q(x) \neq \pi_Q(x')$ *holds and there exist* $\tau, \epsilon \in R^+$ *such that* $\chi_{x,k(x)}(\tau) = x'$, $\pi_Q(\chi_{x,k(x)}(t_1)) = \pi_Q(x)$ *and* $\pi_Q(\chi_{x,k(x)}(t_2)) = \pi_Q(x')$, *where* $t_1 \in [0, \epsilon)$ *and* $t_2 \in [\epsilon, \tau]$.

Next, we present the property of the multi-affine function on rectangles [81].

Lemma 5.1 *Consider a multi-affine function f and a rectangle $R_{k_1k_2\cdots k_n}$. In every point $x \in R_{k_1k_2\cdots k_n}$, the value $f(x)$ is uniquely determined by the values of f at vertices of $R_{k_1k_2\cdots k_n}$:*

$$f(x) = \sum_{w \in V(R_{k_1k_2\cdots k_n})} \lambda_w(x) f(w) \tag{5.1}$$

where for any $w = (w_1, \cdots, w_n) \in V(R_{k_1k_2\cdots k_n})$ and $x = (x_1, x_2, \cdots, x_n) \in R_{k_1k_2\cdots k_n}$, the coefficient $\lambda_w(x)$ is defined as

$$\lambda_w(x) = \prod_{j=1}^{n} \left(\frac{x_j - a_j^{k_j}}{b_j^{k_j} - a_j^{k_j}} \right)^{S(w_j)} \left(\frac{b_j^{k_j} - x_j}{b_j^{k_j} - a_j^{k_j}} \right)^{(1-S(w_j))} \tag{5.2}$$

5.2 Control of Multi-Affine Systems on Rectangles

In the previous section, the state space has been rectangularly partitioned into several rectangles. Now, we investigate the control of multi-affine systems on rectangles. First, the notion of state-based switch multi-affine function is introduced.

Definition 5.4 *Given multi-affine functions $U : \mathbb{R}^n \to \mathbb{R}^m$ and $U' : R^n \to R^m$, $x_f \in R^n$ and $\varepsilon \in R^+$, a function $U \diamond U' : R^n \to R^m$ is said to be a state-based switch multi-affine function from U to U' with respect to x_f and ε if*

$$U \diamond U'(x) = \begin{cases} U(x) \ \ if \ \ x \notin B_\varepsilon(x_f) \\ U'(x) \ \ if \ \ x \in B_\varepsilon(x_f) \end{cases}$$

where $B_\varepsilon(x_f) = \{x \mid \ ||x - x_f|| \leq \varepsilon\}$ with $||\ ||$ denotes the Euclidean norm.

In this chapter, the control input for a multi-affine system $\dot{x} = g(x) + Bu$ is in terms of $u = K(x)$, where K a multi-affine function or a state-based switch multi-affine function. Therefore, the feedback law is automatically bounded on $R_{k_1k_2\cdots k_n}$.

Next, we review the results on the existence of a multi-affine feedback controller for a multi-affine system to keep the system in a rectangular invariant (Lemma 5.2) and to drive all initial states in a rectangle through a desired fact in finite time (Lemma 5.3) [81].

Lemma 5.2 *Given a multi-affine control system* $\Sigma : \dot{x} = g(x) + Bu$ *and a rectangle* $R_{k_1k_2\cdots k_n}$*, there exists a multi-affine feedback controller* $K(x)$ *such that* $u = K(x)$ *and all trajectories of the closed-loop system that start from* $R_{k_1k_2\cdots k_n}$ *remain in* $R_{k_1k_2\cdots k_n}$ *for all times if and only if for any* $w \in V(R_{k_1k_2\cdots k_n})$*, the following set is nonempty:*

$$U_I(w) = \bigcap_{F^{j,d}_{k_1k_2\cdots k_n} \in F(w)} \{v \in R^m \mid n^{j,d}(g(w) + Bv) \leq 0\}. \tag{5.3}$$

In this chapter, the multi-affine function U which keeps the system in a rectangular $R_{k_1k_2\cdots k_n}$ invariant is called as an invariant controller with respect to $R_{k_1k_2\cdots k_n}$.

Lemma 5.3 *Given a multi-affine control system* $\Sigma : \dot{x} = g(x) + Bu$ *and a rectangle* $R_{k_1k_2\cdots k_n}$*, there exists a multi-affine feedback controller* $K(x)$ *such that* $u = K(x)$ *and all trajectories of the closed-loop system that start from* $R_{k_1k_2\cdots k_n}$ *are driven only through* $F^{j,d}_{k_1k_2\cdots k_n}$ *in finite time if for any* $w \in V(R_{k_1k_2\cdots k_n})$*, the following set is nonempty:*

$$U_E(w) = \bigcap_{F^{j',d'}_{k_1k_2\cdots k_n} \in F(w) \setminus (j,d) \neq (j',d')} \{v \in R^m \mid n^{j,d}(g(w)+Bv) > 0 \wedge n^{j',d'}(g(w)+Bv) \leq 0\} \tag{5.4}$$

In the rest of this subsection, we propose a control method based on the exit sub-region to drive all trajectories of the closed-loop system starting from $R_{k_1k_2\cdots k_n}$ to exit through a desired facet of $R_{k_1k_2\cdots k_n}$, where the exit sub-region is defined as follows.

Definition 5.5 *Let $\Sigma : \dot{x} = g(x) + Bu$ be a multi-affine control system, $K(x)$ be a multi-affine feedback controller, $R_{k_1k_2\cdots k_n}$ be a rectangle and $F^{j,d}_{k_1k_2\cdots k_n}$ be a facet of $R_{k_1k_2\cdots k_n}$. A region $[K]^{j,d}_{k_1k_2\cdots k_n} \subseteq R_{k_1k_2\cdots k_n}$ is called to be an exit sub-region with respect to $F^{j,d}_{k_1k_2\cdots k_n}$ and $K(x)$ if for any $x_0 \in [K]^{j,d}_{k_1k_2\cdots k_n}$, there exists $\tau \in R^+$ such that*

(1) $\chi_{x_0,K(x)}(t_1) \in R_{k_1k_2\cdots k_n}$ for $t_1 \in [0, \tau)$;

(2) $\chi_{x_0,K(x)}(t_2) \in F^{j,d}_{k_1k_2\cdots k_n}$ for $t_2 = \tau$;

(3) $\chi_{x_0,K(x)}(t_3) \notin R_{k_1k_2\cdots k_n} \bigcup F^{j,d}_{k_1k_2\cdots k_n}$ for $t_3 \in (\tau, \tau + \varepsilon)$ and $\varepsilon \in \mathbb{R}^+$.

We can see that all trajectories of the closed-loop system $\dot{x} = g(x) + BK(x)$ originating in the sub-region $[K]^{j,d}_{k_1k_2\cdots k_n}$ will leave $R_{k_1k_2\cdots k_n}$ only through $F^{j,d}_{k_1k_2\cdots k_n}$. It implies that if we can find a controller $K'(x)$ such that all trajectories of the closed-loop system $\dot{x} = g(x) + BK'(x)$ starting from $R_{k_1k_2\cdots k_n}$ can reach the exit sub-region $[K]^{j,d}_{k_1k_2\cdots k_n}$ in finite time, then the control of multi-affine systems with respect to the exit facet $F^{j,d}_{k_1k_2\cdots k_n}$ can be realized by using $K(x)$ together with $K'(x)$. That is, we can first apply the controller $K'(x)$ to the multi-affine system and then update the controller to $K(x)$ once the trajectories arrive in $[K]^{j,d}_{k_1k_2\cdots k_n}$. To implement this idea, the following problems should be addressed.

Problem 1: how to find a controller $K(x)$ to guarantee the existence of an exit sub-region $[K]^{j,d}_{k_1k_2\cdots k_n}$? If there exists an exit sub-region $[K]^{j,d}_{k_1k_2\cdots k_n}$, how to compute it?

Problem 2: how to design a controller $K'(x)$ to drive all trajectories of the closed-loop system starting from $R_{k_1k_2\cdots k_n}$ towards $[K]^{j,d}_{k_1k_2\cdots k_n}$?

For Problem 1, we provide the following proposition.

Proposition 5.1 *Given a multi-affine control system* $\Sigma : \dot{x} = g(x) + Bu$*, a multi-affine feedback controller* $K(x)$*, a rectangle* $R_{k_1k_2\cdots k_n}$ *and a facet* $F^{j,d}_{k_1k_2\cdots k_n}$ *of* $R_{k_1k_2\cdots k_n}$*, there exists an exit sub-region* $[K]^{j,d}_{k_1k_2\cdots k_n}$ *with respect to* $F^{j,d}_{k_1k_2\cdots k_n}$ *and* $K(x)$ *if*

(1) $\exists w \in V(F^{j,d}_{k_1k_2\cdots k_n})$*:*

$$n^{j,d}[g(w) + BK(w)] > 0; \tag{5.5}$$

(2) $\forall v \in V(R_{k_1k_2\cdots k_n})\backslash V(F^{j,d}_{k_1k_2\cdots k_n})$, $\forall F^{j',d'}_{k_1k_2\cdots k_n} \in F(v)$*:*

$$n^{j',d'}[g(v) + BK(v)] \leq 0; \tag{5.6}$$

(3) $\forall x \in R_{k_1k_2\cdots k_n}$*:*

$$g(x) + BK(x) \neq 0. \tag{5.7}$$

Proof: We have $n^{j,d}[g(w) + BK(w)] > 0$ at the vertex $w \in V(F^{j,d}_{k_1k_2\cdots k_n})$. Because the vector field is continuous, there exist some points at the neighborhood of w that have strictly positive vector field outwards $R_{k_1k_2\cdots k_n}$ through $F^{j,d}_{k_1k_2\cdots k_n}$. Moreover, (5.6) implies that the trajectories of the closed-loop system can not leave through the facets whose vertices all satisfy the condition (5.6), and (5.7) implies there does not exist an equilibrium point inside $R_{k_1k_2\cdots k_n}$. We conclude that some trajectories of the closed-loop system starting from $R_{k_1k_2\cdots k_n}$ will leave through $F^{j,d}_{k_1k_2\cdots k_n}$. That is, there is an exit sub-region $[K]^{j,d}_{k_1k_2\cdots k_n}$ of $R_{k_1k_2\cdots k_n}$ with respect to $F^{j,d}_{k_1k_2\cdots k_n}$ and $K(x)$.

It intuitively states that there exists an exit sub-region $[K]^{j,d}_{k_1k_2\cdots k_n}$ with respect to $F^{j,d}_{k_1k_2\cdots k_n}$ and $K(x)$ if the multi-affine feedback controller $K(x)$ is such that: (1) there exists a vertex w on the exit facet such that the velocity of the closed-loop system $g(w) + BK(w)$ at w has a strictly positive projection along the outer normal

of the exit facet; (2) for any vertex v which is not on the exit facet, the velocity of the closed-loop system $g(v) + BK(v)$ at v has a negative projection along the outer normal of the facet containing v. (3) there does not exist an equilibrium point inside $R_{k_1k_2\cdots k_n}$. Moreover, the exit sub-region can be computed by using the result of [94]. Thus, Problem 1 is solved. Then, we consider Problem 2. The following proposition is introduced.

Proposition 5.2 ***(Control to a Fixed Point)*** *Given a multi-affine control system $\Sigma : \dot{x} = g(x) + Bu$, a rectangle $R_{k_1k_2\cdots k_n}$ and a desired point $x_f \in R_{k_1k_2\cdots k_n}$, there exists a multi-affine feedback controller $K'(x)$ such that $u = K'(x)$ and all trajectories of the closed-loop system starting from $R_{k_1k_2\cdots k_n}$ remain in $R_{k_1k_2\cdots k_n}$ for all times and converge to x_f if for any $w \in V(R_{k_1k_2\cdots k_n})$, $U_I(w) \neq \emptyset$ holds and there exists $u'(w) \in U_I(w)$ such that x_f is a unique point in $R_{k_1k_2\cdots k_n}$:*

$$g(x_f) + B \sum_{w \in V(R_{k_1k_2\cdots k_n})} \lambda_w(x_f) u'(w) = 0. \tag{5.8}$$

Proof: Because $U_I(w) \neq \emptyset$ for any $w \in V(R_{k_1k_2\cdots k_n})$, there exists a multi-affine feedback controller such that all trajectories of the closed-loop system starting from $R_{k_1k_2\cdots k_n}$ remain in $R_{k_1k_2\cdots k_n}$ for all times by Lemma 5.2. Let $u'(w) \in U_I(w)$ be the control input at w such that x_f is a unique point in $R_{k_1k_2\cdots k_n}$ satisfying (5.8). Then, we design $K'(x) = \sum_{w \in V(R_{k_1k_2\cdots k_n})} \lambda_w(x) u'(w)$. For all rectangle $\alpha R_{k_1k_2\cdots k_n}$, where $\alpha \in [0, 1]$, the vertex set $V(\alpha R_{k_1k_2\cdots k_n}) = \{\alpha w + (1 - \alpha)x_f\}$. It can be seen that $\alpha R_{k_1k_2\cdots k_n}$ is just a shrunken version of $R_{k_1k_2\cdots k_n}$ by multiplying $R_{k_1k_2\cdots k_n}$ from x_f by the factor α. Thus, the velocity vector of closed-loop system at the vertex of $\alpha R_{k_1k_2\cdots k_n}$ is just α-multiple the velocity vector at the corresponding vertex of $R_{k_1k_2\cdots k_n}$. Since

the vector field of the closed-loop system in all vertices of $\alpha R_{k_1k_2\cdots k_n}$ is pointing inside to $\alpha R_{k_1k_2\cdots k_n}$, there exist $t_0 > 0$ and $\alpha' \in [0,1)$ such that $\chi_{w,K'(x)}(t_0) \in \alpha' R_{k_1k_2\cdots k_n}$. Then, $\chi_{x_0,K'(x)}(t) \in \alpha' R_{k_1k_2\cdots k_n}$ for all $x_0 \in R_{k_1k_2\cdots k_n}$ and $t \geq t_0$. Similarly, we obtain $\chi_{x_0,K'(x)}(t) \in (\alpha')^n R_{k_1k_2\cdots k_n}$ for $t \geq nt_0$. Therefore, $\lim_{t\to\infty} \chi_{x_0,K'(x)}(t) = x_f$.

It indicates that if we can construct a controller of the form $u = K'(x) = \sum_{w\in V(R_{k_1k_2\cdots k_n})} \lambda_w(x)u'(w)$, where $u'(w) \in U_I(w) \neq \emptyset$, such that x_f is a unique equilibrium point inside $R_{k_1k_2\cdots k_n}$, then all trajectories of the closed-loop system starting from $R_{k_1k_2\cdots k_n}$ are driven towards x_f. This kind of multi-affine function K' is called a fixed point controller with respect to x_f. By putting x_f inside the exit sub-region $[K]^{j,d}_{k_1k_2\cdots k_n}$, the fixed point controller yields a solution for Problem 2. Now, we are ready to present the result on the control with respect to a desired exit facet.

Proposition 5.3 *(**Control to an Exit Facet**) Given a multi-affine control system $\Sigma : \dot{x} = g(x) + Bu$, a rectangle $R_{k_1k_2\cdots k_n}$ and a facet $F^{j,d}_{k_1k_2\cdots k_n}$ of $R_{k_1k_2\cdots k_n}$, there exists a feedback controller such that all trajectories of the closed-loop system starting from $R_{k_1k_2\cdots k_n}$ are driven only through $F^{j,d}_{k_1k_2\cdots k_n}$ in finite time if any of the following two conditions is satisfied:*

(1) $U_E(w) \neq \emptyset$ holds for any $w \in V(R_{k_1k_2\cdots k_n})$;

(2) $U_E(w) \neq \emptyset$ does not hold for any $w \in V(R_{k_1k_2\cdots k_n})$ and there exist $x_f \in R_{k_1k_2\cdots k_n}$, $\varepsilon \in \mathbb{R}^+$ and multi-affine functions U and U' such that $B_\varepsilon(x_f) \subseteq [U]^{j,d}_{k_1k_2\cdots k_n}$ and $U'(x)$ is a fixed point controller with respect to x_f.

Proof: As for condition (1), it obviously guarantees the existence of a controller with respect to an exit facet according to Lemma 5.3. As for condition (2), because U'

is a fixed point controller with respect to x_f, all trajectories of the closed-loop system $\dot{x} = g(x) + BU'(x)$ starting from $R_{k_1k_2\cdots k_n}$ will converge towards x_f. Moreover, there is $\varepsilon \in \mathbb{R}^+$ such that $B_\varepsilon(x_f) \subseteq [U]^{j,d}_{k_1k_2\cdots k_n}$, where $[U]^{j,d}_{k_1k_2\cdots k_n}$ is an exit sub-region with respect to $F^{j,d}_{k_1k_2\cdots k_n}$ and $U(x)$. By using the state-based switch multi-affine feedback controller $U' \diamond U(x)$ (w.r.t. x_f and ε), all trajectories of the corresponding closed-loop system starting from $R_{k_1k_2\cdots k_n}$ will exit only through $F^{j,d}_{k_1k_2\cdots k_n}$ in finite time.

Proposition 5.3 provides two different ways to drive the trajectories of the corresponding closed-loop system starting from $R_{k_1k_2\cdots k_n}$ to exit only through a desired facet. One (condition (1)) is based on the result of Lemma 5.3 and the other (condition (2)) is based on the exit sub-region. Thus, the proposed control method for an exit facet covers more classes of systems than those are addressed in [81, 82]. We call the multi-affine function or the state-based switch multi-affine function U which drives all trajectories of the closed-loop system starting from $R_{k_1k_2\cdots k_n}$ to exit only through $F^{j,d}_{k_1k_2\cdots k_n}$ as an exit controller with respect to $F^{j,d}_{k_1k_2\cdots k_n}$. Such an exit controller can be obtained by the following algorithm.

Proposition 5.4 *Algorithm 1 is correct.*

Proof: From the definition of $U^2_{R_{k_1k_2\cdots k_n}}$ and Proposition 5.2, we have $U^2_{R_{k_1k_2\cdots k_n}}$ is a fixed point controller with respect to x'. In addition, there exist $\varepsilon \in \mathbb{R}^+$ and a unique point $x' \in R_{k_1k_2\cdots k_n}$ s.t. $g(x') + BU^3_{R_{k_1k_2\cdots k_n}}(x') = 0$ and $B_\varepsilon(x') \subseteq [U^2_{R_{k_1k_2\cdots k_n}}]^{j,d}$. Then, Proposition 5.3 implies that $U^3_{R_{k_1k_2\cdots k_n}} \diamond U^2_{R_{k_1k_2\cdots k_n}}$ w.r.t. x' and ε is an exit controller for $F^{j,d}_{k_1k_2\cdots k_n}$. On the other hand, Proposition 5.3 further implies that $U^1_{R_{k_1k_2\cdots k_n}}$ is an exit controller w.r.t. $F^{j,d}_{k_1k_2\cdots k_n}$. Therefore, Algorithm 1 is correct.

Algorithm 1: Synthesis of Exit Controllers

input : a multi-affine control system $\Sigma : \dot{x} = g(x) + Bu$, a rectangle $R_{k_1k_2\cdots k_n}$, a facet $F^{j,d}_{k_1k_2\cdots k_n}$ of $R_{k_1k_2\cdots k_n}$ and $|u| \leq \eta$.

output: an exit controller with respect to $F^{j,d}_{k_1k_2\cdots k_n}$.

Let $V(R_{k_1k_2\cdots k_n}) = \{w_j \mid j = 1, 2, \cdots, 2^n\}$;

if $U_E(w_j) \neq \emptyset$ *for any* $w_j \in V(R_{k_1k_2\cdots k_n}) \setminus V(F^{j,d}_{k_1k_2\cdots k_n})$ **then**

  $V_1 := \{j \in \{1, 2, \cdots, 2^n\} \mid U_E(w_j) = \emptyset, w_j \in V(R_{k_1k_2\cdots k_n})\}$;

  if $V_1 = \emptyset$ **then**

    $U_1 := \{\{U^1_{R_{k_1k_2\cdots k_n}}(w_j) \mid j = 1, 2, \cdots, 2^n\} \mid w_j \in V(R_{k_1k_2\cdots k_n}) \wedge U^1_{R_{k_1k_2\cdots k_n}}(w_j) \in U_E(w_j) \wedge |\sum_{j=1,2,\cdots,2^n} \lambda_{w_j}(x) U^1_{R_{k_1k_2\cdots k_n}}(w_j)| \leq \eta, x \in R_{k_1k_2\cdots k_n}\}$;

    if $U_1 \neq \emptyset$ **then**

      $U^1_{R_{k_1k_2\cdots k_n}}(x) = \sum_{j=1,2,\cdots,2^n} \lambda_{w_j}(x) U^1_{R_{k_1k_2\cdots k_n}}(w_j)$, where $\{U^1_{R_{k_1k_2\cdots k_n}}(w_j) | j = 1, 2, \cdots, 2^n\} \in U_1$;

      The multi-affine function $U^1_{R_{k_1k_2\cdots k_n}}$ is an exit controller w.r.t. $F^{j,d}_{k_1k_2\cdots k_n}$;

  if $V_1 \subset \{1, 2, \cdots, 2^n\}$ **then**

    if $U_I(w_j) \neq \emptyset$ *for any* $w_j \in V(R_{k_1k_2\cdots k_n})$ **then**

      $U_3 := \{\{U^3_{R_{k_1k_2\cdots k_n}}(w_j) \mid j = 1, 2, \cdots, 2^n\} \mid w_j \in V(R_{k_1k_2\cdots k_n}) \wedge U^3_{R_{k_1k_2\cdots k_n}}(w_j) \in U_I(w_j) \wedge |\sum_{j=1,2,\cdots,2^n} \lambda_{w_j}(x) U^3_{R_{k_1k_2\cdots k_n}}(w_j)| \leq \eta, x \in R_{k_1k_2\cdots k_n}\}$;

      if $U_3 \neq \emptyset$ **then**

        $U_2 := \{\{U^2_{R_{k_1k_2\cdots k_n}}(w_j) \mid j = 1, 2, \cdots, 2^n\} \mid n^{j,d}[g(w_m) + B \times U^2_{R_{k_1k_2\cdots k_n}}(w_m)] > 0 \wedge w_j \in V(R_{k_1k_2\cdots k_n}) \wedge U^2_{R_{k_1k_2\cdots k_n}}(w_l) \in U_E(w_l) \wedge |\sum_{j=1,2,\cdots,2^n} \lambda_{w_j}(x) \times U^2_{R_{k_1k_2\cdots k_n}}(w_j)| \leq \eta, m \in V_1, l \in \{1, 2, \cdots, 2^n\} \setminus V_1$ and $x \in R_{k_1k_2\cdots k_n}\}$;

        if $U_2 \neq \emptyset$ **then**

          for *any* $\{U^2_{R_{k_1k_2\cdots k_n}}(w_j) \mid j = 1, 2, \cdots, 2^n\} \in U_2$ **do**

            $U^2_{R_{k_1k_2\cdots k_n}}(x) = \sum_{j=1,2,\cdots,2^n} \lambda_{w_j}(x) U^2_{R_{k_1k_2\cdots k_n}}(w_j)$;

            Obtain the exit sub-region $[U^2_{R_{k_1k_2\cdots k_n}}]^{j,d}$ w.r.t. $F^{j,d}_{k_1k_2\cdots k_n}$ and $U^2_{R_{k_1k_2\cdots k_n}}(x)$;

            for *any* $\{U^3_{R_{k_1k_2\cdots k_n}}(w_j) \mid j = 1, \cdots, 2^n\} \in U_3$ **do**

              $U^3_{R_{k_1k_2\cdots k_n}}(x) = \sum_{j=1,2,\cdots,2^n} \lambda_{w_j}(x) U^3_{R_{k_1k_2\cdots k_n}}(w_j)$;

              if $\exists \varepsilon \in \mathbb{R}^+$ *and a unique point* $x' \in R_{k_1k_2\cdots k_n}$ *s.t.* $g(x') + BU^3_{R_{k_1k_2\cdots k_n}}(x') = 0$ *and* $B_\varepsilon(x') \subseteq [U^2_{R_{k_1k_2\cdots k_n}}]^{j,d}$ **then**

                The state-based switch multi-affine function $U^3_{R_{k_1k_2\cdots k_n}} \diamond U^2_{R_{k_1k_2\cdots k_n}}$ w.r.t. x' and ε is an exit controller for $F^{j,d}_{k_1k_2\cdots k_n}$.

5.3 Bisimilarly Abstracted Discrete Event System

The control of multi-affine systems on rectangles enables the construction of a finitely abstracted transition system for the multi-affine system. Here we assume that any initial state of the multi-affine system is inside the rectangles and the duration that the trajectories stay on the boundary of the rectangle is ignored. These assumptions result in no loss of generality since they always hold in the implementations.

Definition 5.6 *Given a multi-affine control system* $\Sigma : \dot{x} = g(x) + Bu$ *and a rectangle set* ξ *generated by rectangularly partitioning the state space, the abstracted transition system of* Σ *associated with* ξ*, denoted as* $S_{\Sigma,\xi}$*, is a tuple*

$$S_{\Sigma,\xi} = (X_\xi, U_\xi, \rightarrow_\xi, X_{\xi 0}, X_{m\xi})$$

- $X_\xi = \xi = X_{m\xi}$;
- $U_\xi = \{U_{R_{k_1k_2\cdots k_n}} \mid U_{R_{k_1k_2\cdots k_n}}$ *is a multi-affine function or a state-based switch multi-affine function,* $R_{k_1k_2\cdots k_n} \in \xi\}$;
- $R_{k_1k_2\cdots k_n} \xrightarrow{U_{R_{k_1k_2\cdots k_n}}}_\xi R_{k'_1k'_2\cdots k'_n}$ *if any of the following two conditions is satisfied:*

  (1) $R_{k_1k_2\cdots k_n} = R_{k'_1k'_2\cdots k'_n}$ *holds and for any* $w \in V(R_{k_1k_2\cdots k_n})$, $U_I(w) \neq \emptyset$ *and* $U_{R_{k_1k_2\cdots k_n}}(w) \in U_I(w)$.

  (2) $R_{k_1k_2\cdots k_n} \neq R_{k'_1k'_2\cdots k'_n}$ *with* $\overline{R_{k_1k_2\cdots k_n}} \cap \overline{R_{k'_1k'_2\cdots k'_n}} = F^{j,d}_{k_1k_2\cdots k_n}$ *holds and* $U_{R_{k_1k_2\cdots k_n}}$ *is an exit controller with respect to* $F^{j,d}_{k_1k_2\cdots k_n}$.

- $X_{\xi 0} = \{R_{k_1k_2\cdots k_n} \in \xi \mid R_{k_1k_2\cdots k_n}$ *contains an initial state of the multi-affine control system*$\}$.

An abstracted transition system is a finite state system, therefore it facilitates controller synthesis for finite-state requirements while accommodating to infinite-state dynamics. Next, we investigate the relationship between the abstracted transition system and the regular transition system of the multi-affine control system.

Theorem 5.1 *Given a multi-affine control system* $\Sigma : \dot{x} = g(x) + Bu$, *a rectangle set* ξ *generated by rectangularly partitioning the state space and a rectangular projection map* π_Q *defined by* ξ, *there exists a bisimulation relation* ϕ_1 *such that* $S_{\Sigma,\xi} \cong_{\phi_1} S_{\Sigma,Q}$.

Proof: Consider $\phi = \{(R_{k_1k_2\cdots k_n}, x) \mid x \in R_{k_1k_2\cdots k_n}\}$. *For any* $(R_{k_1k_2\cdots k_n}, x) \in \phi$ *if there is a transition* $R_{k_1k_2\cdots k_n} \xrightarrow{U_{R_{k_1k_2\cdots k_n}}}_{\xi} R_{k'_1k'_2\cdots k'_n}$, *we have the following two cases: (a)* $R_{k_1k_2\cdots k_n} \neq R_{k'_1k'_2\cdots k'_n}$ *with* $F^{j,d}_{k_1k_2\cdots k_n} = \overline{R_{k_1k_2\cdots k_n}} \cap \overline{R_{k'_1k'_2\cdots k'_n}}$. *According to the construction of* $S_{\Sigma,\xi}$, *there exists a controller* $U_{R_{k_1k_2\cdots k_n}}$ *such that all trajectories of the closed-loop system* $\dot{x} = g(x) + BU_{R_{k_1k_2\cdots k_n}}(x)$ *starting from* $R_{k_1k_2\cdots k_n}$ *are driven only through* $F^{j,d}_{k_1k_2\cdots k_n}$. *Then, for any* $x \in R_{k_1k_2\cdots k_n}$, *there is* $x' \in R_{k'_1k'_2\cdots k'_n}$ *such that* $x \xrightarrow{U_{R_{k_1k_2\cdots k_n}}}_{Q} x'$ *and* $(R_{k'_1k'_2\cdots k'_n}, x') \in \phi$. *(b)* $R_{k_1k_2\cdots k_n} = R_{k'_1k'_2\cdots k'_n}$. *The controller* $U_{R_{k_1k_2\cdots k_n}}$ *satisfying* $U_{R_{k_1k_2\cdots k_n}}(w) \in U_I(w) \neq \emptyset$ *for any* $w \in V(R_{k_1k_2\cdots k_n})$ *drives all trajectories of the closed-loop system* $\dot{x} = g(x) + BU_{R_{k_1k_2\cdots x_n}}(x)$ *starting from* $R_{k_1k_2\cdots k_n}$ *to remain in* $R_{k_1k_2\cdots k_n}$ *for all times [81]. Therefore, there exists* $x' \in R_{k_1k_2\cdots k_n}$ *such that* $x \xrightarrow{U_{R_{k_1k_2\cdots k_n}}}_{Q} x'$ *and* $(R_{k'_1k'_2\cdots k'_n}, x') \in \phi$. *Moreover, the definition of* $X_{\xi 0}$ *indicates that for any* $R_{k_1k_2\cdots k_n} \in X_{\xi 0}$, *there exists* $x \in X_{Q0}$ *such that* $(R_{k_1k_2\cdots k_n}, x) \in \phi$. *Similarly, we can prove that for any* $(x, R_{k_1k_2\cdots k_n}) \in \phi^{-1}$, *if there is a transition* $x \xrightarrow{U_{R_{k_1k_2\cdots k_n}}}_{Q} x'$, *then we have* $R_{k_1k_2\cdots k_n} \xrightarrow{U_{R_{k_1k_2\cdots k_n}}}_{\xi} R_{k'_1k'_2\cdots k'_n}$ *such that* $(x', R_{k'_1k'_2\cdots k'_n}) \in \phi^{-1}$. *As a result,* $S_{\Sigma,\xi} \cong_{\phi\cup\phi^{-1}} S_{\Sigma,Q}$.

5.4 Controller Synthesis

This section provides a bisimulation-based approach for the controller synthesis of the multi-affine system with respect to temporal logic specifications.

5.4.1 Linear Temporal Logic

The syntax and semantics of linear temporal logic (LTL) formulas over the words of the transition system are introduced [95].

Definition 5.7 *An LTL formula over* Π *is recursively defined as*

- *Every proposition* $\pi \in \Pi$ *is a formula.*
- *If* φ_1 *and* φ_2 *are formulas, then* $\varphi_1 \wedge \varphi_2$, $\neg\varphi_1$, $\circ\varphi$ *and* $\varphi_1 \mathcal{U} \varphi_2$ *are also formulas.*

Definition 5.8 *The satisfaction of an LTL formula* φ *at position* $i = 1, 2, 3, \cdots$ *of the word* W*, denoted by* $W(i) \vDash \varphi$*, is recursively defined as*

- $W(i) \vDash \pi$*, if* $\pi \in W(i)$*;*
- $W(i) \vDash \neg\varphi$*, if* $W(i) \nvDash \varphi$*, where* $\nvDash$ *denotes the negation of* $\vDash$*;*
- $W(i) \vDash \circ\varphi$ *if* $W(i+1) \vDash \varphi$*;*
- $W(i) \vDash \varphi_1 \wedge \varphi_2$*, if* $W(i) \vDash \varphi_1$ *and* $W(i) \vDash \varphi_2$*;*
- $W(i) \vDash \varphi_1 \mathcal{U} \varphi_2$*, if there exists* $j > i$ *such that* $W(j) \vDash \varphi_2$ *and for all* $i \leq k < j$ *we have* $W(k) \vDash \varphi_1$*.*

If $W(1) \vDash \varphi$, we say that the word W satisfies φ, written as $W \vDash \varphi$. The symbols $\wedge$ and $\neg$ stand for conjunction and negation. The other Boolean connectors $\vee$ (disjunction), $\Rightarrow$ (implication), and $\Leftrightarrow$ (equivalence) are defined in the usual way. The temporal operator $\circ$ is called the next operator. Formula $\circ\varphi$ specifies that φ will be true in the next step. The temporal operator $\mathcal{U}$ is called the until operator. Formula $\varphi_1 \mathcal{U} \varphi_2$ means that φ_1 must hold until φ_2 holds. Two additional operators, "eventually" and "always" are defined as $\diamondsuit\varphi$=**true**$\mathcal{U}\phi$ and $\square\varphi$=$\neg\diamondsuit\neg\varphi$. Formula $\diamondsuit\varphi$ means that φ becomes eventually true whereas $\square\varphi$ indicates that φ is true at all positions of W. This set of operators can be employed to express many interesting specifications such as system synchronization [17] and obstacle avoidance (See Example 5.1).

It is well known that a linear temporal logic formula φ over a proposition set Π can be effectively converted into a B*ü*chi automaton which accepts every infinite string over Π satisfying φ [96]. This kind of B*ü*chi automaton is described as follows.

Definition 5.9 *Given a linear temporal logic formula φ over a proposition set Π, the B*ü*chi automaton with respect to φ, denoted as B_φ, is a tuple*

$$B_\varphi = (B, B_0, 2^\Pi, \rightarrow_B, B_m)$$

- *B, $B_0 \subseteq B$ and $B_m \subseteq B$ are finite sets of states, initial states and marked states respectively;*
- *2^Π is an input alphabet;*
- *$\rightarrow_B \subseteq B \times 2^\Pi \times 2^B$ is a transition relation.*

5.4.2 Supervisor Synthesis

Since the abstracted transition system $S_{\Sigma,\xi}$ is bisimilar the rectangular transition system $S_{\Sigma,Q}$, if there exists a supervisor (discrete controller) S_c for $S_{\Sigma,\xi}$ enforcing temporal logic specifications, then such a supervisor also works for $S_{\Sigma,Q}$, i.e., the implementation of S_c drives the multi-affine system to fulfill temporal logic specifications. Thus, we focus on the synthesis of S_c. In particular, the supervisor S_c can restrict the behaviors of $S_{\Sigma,\xi}$ which fail to satisfy the LTL specifications. This observation motivates us to construct S_c by working with $S_{\Sigma,\xi}$ and B_φ. Hence, we introduce the notion of product automaton. Given a proposition set Π, the label function $L : \xi \to 2^{\Pi}$ assigns each rectangle a set of atomic propositions satisfied by this rectangle.

Definition 5.10 *Given an abstracted transition system* $S_{\Sigma,\xi}=(X_\xi, U_\xi, \to_\xi, X_{\xi 0}, X_{m\xi})$, *a Büchi automaton* $B_\varphi = (B, B_0, 2^{\Pi}, \to_B, B_m)$ *and a label function* $L : Y_\xi \to 2^{\Pi}$, *the product automaton of* $S_{\Sigma,\xi}$ *and* B_φ, *denoted as* $S_{\Sigma,\xi} \times_A B_\varphi$, *is a transition system*

$$S_{\Sigma,\xi} \times_A B_\varphi = (A, U_A, \to_A, A_0, A_m)$$

- $A = X_\xi \times B$;
- $U_A = U_\xi$;
- $(x_\xi, b) \xrightarrow{u}_A (x'_\xi, b')$ *iff* $x_\xi \xrightarrow{u}_\xi x'_\xi$ *and* $b \xrightarrow{L(x'_\xi)}_B b'$;
- $A_0 = \{(x_\xi, b) \in X_{\xi 0} \times B \mid (\exists b_0 \in B_0)\ b_0 \xrightarrow{L(x_\xi)}_B b\}$;
- $A_m = X_{m\xi} \times B_m$.

The result provided by [97] indicates that if we can design a supervisor S_c such that the supervised system $S_c||S_{\Sigma,Q}$ is bisimilar to $S_{\Sigma,\xi} \times_A B_\varphi$, then $S_c||S_{\Sigma,Q}$ satisfies the LTL formula φ. In fact, S_c is a bisimilarity enforcing supervisor for $S_{\Sigma,Q}$ and $S_{\Sigma,\xi} \times_A B_\varphi$. The existence condition of S_c is stated as below.

Theorem 5.2 *Given a rectangular transition system $S_{\Sigma,Q}$ and a product automaton $S_{\Sigma,\xi} \times_A B_\varphi$, there exists a supervisor S_c for $S_{\Sigma,Q}$ such that $S_c||S_{\Sigma,Q} \cong S_{\Sigma,\xi} \times_A B_\varphi$ if $S_{\Sigma,\xi} \times_A B_\varphi \neq \emptyset$.*

Proof: Since $S_{\Sigma,\xi} \times_A B_\varphi \neq \emptyset$ and any event in $S_{\Sigma,\xi}$ and $S_{\Sigma,Q}$ is controllable, let $S_c = S_{\Sigma,\xi} \times_A B_\varphi$. It is known that $S_{\Sigma,\xi} \times_A B_\varphi \prec S_{\Sigma,\xi}$ and $S_{\Sigma,\xi}$ is deterministic. Then, $S_c||S_{\Sigma,\xi} = (S_{\Sigma,\xi} \times_A B_\varphi)||S_{\Sigma,\xi} \cong S_{\Sigma,\xi} \times_A B_\varphi$. Moreover, Theorem 5.1 indicates $S_{\Sigma,\xi} \cong S_{\Sigma,Q}$. Thus, $S_c||S_{\Sigma,Q} \cong S_c||S_{\Sigma,\xi} \cong S_{\Sigma,\xi} \times_A B_\varphi$.

Remark 5.1 *Theorem 5.2 is constructive since if $S_{\Sigma,\xi} \times_A B_\varphi \neq \emptyset$, we can build $S_c = S_{\Sigma,\xi} \times_A B_\varphi$ as the supervisor to achieve the LTL formula φ.*

5.4.3 Implementation of Supervisor to Multi-Affine Systems

Let S_c be a supervisor which enforces the satisfaction of temporal logic specifications with respect to $S_{\Sigma,Q}$. Then, we discuss the implementation of S_c to the multi-affine system. Let $R_{k_1k_2\cdots k_n}R_{k'_1k'_2\cdots k'_n}\cdots$ be a string in $L(S_c||S_{\Sigma,Q})$ and $R_{k_1k_2\cdots k_n}U_{R_{k_1k_2\cdots k_n}}$ $R_{k'_1k'_2\cdots k'_n}U_{R_{k'_1k'_2\cdots k'_n}}\cdots$ be the corresponding transitions. To realize $R_{k_1k_2\cdots k_n}R_{k'_1k'_2\cdots k'_n}\cdots$, we can apply the controller $U_{R_{k_1k_2\cdots k_n}}(x)$ to the multi-affine system as long as $x \in R_{k_1k_2\cdots k_n}$. When and if $x \notin R_{k_1k_2\cdots k_n}$, the string is updated to $R_{k'_1k'_2\cdots k'_n}$, then the

process continues. Therefore, the implementation of S_c drives the multi-affine system to satisfy temporal logic specifications.

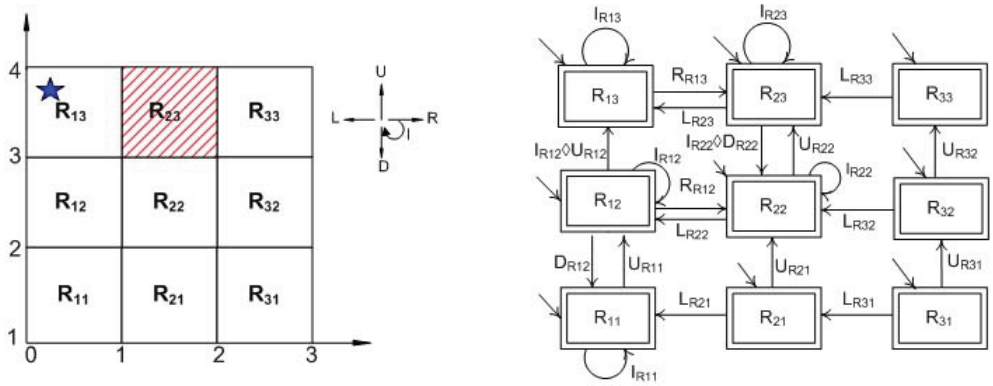

Figure 5.1: Rectangularly Partitioned State Space (Left) and Abstracted Transition System $S_{\Sigma,\xi}$ (Right)

Now, we present an example to illustrate the proposed methodology.

Example 5.1. Consider a path planning example, where a robot with detection and positioning capabilities moves inside a rectangular region $[0,3] \times [1,4]$. In particular, the robot system takes the form of the following differential equation

$$\dot{x} = \begin{bmatrix} \dot{x_1} \\ \\ \dot{x_2} \end{bmatrix} = \begin{bmatrix} -6x_1 + x_2 + x_1x_2 \\ \\ 3x_1 - 2x_2 + x_1x_2 \end{bmatrix} + \begin{bmatrix} 1 \\ \\ 4 \end{bmatrix} u \tag{5.9}$$

where x is the position of the robot and u is the control input. The rectangular region is partitioned into 9 small rectangular sub-regions with respect to the coordinates (Fig. 5.1 (Left)). Let R_{23} be a dangerous sub-region and R_{13} be a goal sub-region. Thus, for each sub-region we define the label function L: $L(R_{23}) = \{Danger, \neg Goal\}$, $L(R_{13}) = \{\neg Danger, Goal\}$ and $L(R_i) = \{\neg Danger, \neg Goal\}$ $(i =$

$11, 12, 21, 22, 31, 32, 33$), where *Danger* represents the dangerous sub-region and *Goal* represents the goal sub-region. In this example, the specification is to eventually go to the goal sub-region ($\diamondsuit Goal$) while avoiding the dangerous sub-region ($\square\neg Danger$). Such an obstacle avoidance specification can be naturally expressed by the linear temporal logic formula φ: $\square\neg Danger \wedge \diamondsuit Goal$.

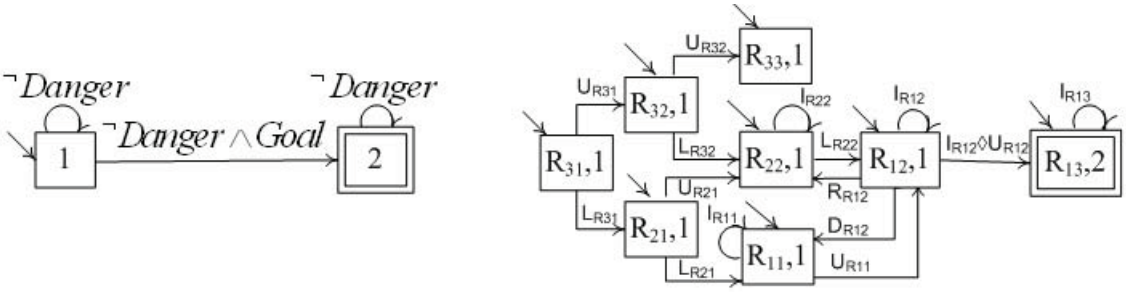

Figure 5.2: $\mathcal{B}_\varphi$ (Left) and $S_{\Sigma,\xi} \times_A \mathcal{B}_\varphi$ (Right)

To achieve the specification, we first explore the control of the robot on sub-regions. Take R_{12} as an example. If we would like to control the robot to exit from R_{12} to R_{13} through the facet $F_{R_{12}}^{2,+}$, then $U_E(1,3)$=$\{v \mid [0,1][-6+3+3+v, 3-6+3+4v]^\top > 0 \wedge [1,0][-6+3+3+v, 3-6+3+4v]^\top \leq 0\}$=$\{v > 0 \wedge v \leq 0\}$= $\emptyset$. Obviously, such a controller does not exist according to Lemma 5.3 [81, 82]. However, by using the proposed method in this chapter, we can obtain a controller for the exit problem. Here we assume the accuracy limitation $\varepsilon = 10^{-4}$ and the control limitation $|u| \leq 10^7$. By Algorithm 1, we can design a state-based switch multi-affine controller:

$$I_{R_{12}} \diamond U_{R_{12}}(x) = \begin{cases} -30x_1 - 12x_2 + 10x_1x_2 + 34 \;\; if \;\; x \notin B_{0.01}(0.767, 2.494) \\ -11x_1 + x_1x_2 + 10 \;\; if \;\; x \in B_{0.01}(0.767, 2.494) \end{cases}$$

to drive the robot to exit only through $F_{R_{12}}^{2,+}$. Similarly, for each sub-region

$R_{mn}(m, n = 1, 2, 3)$ we can establish the controllers that steer the robot from R_{mn} to its neighborhood sub-region (Algorithm 1) or to be invariant (Lemma 5.2) in R_{mn} respectively. Thus, an abstracted transition system $S_{\Sigma,\xi}$ is constructed, as it is shown in Fig. 5.1 (Right).

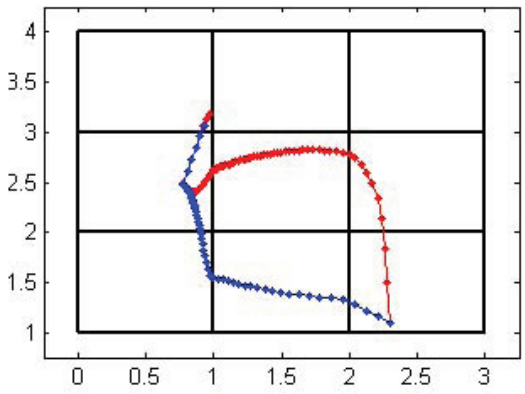

Figure 5.3: Simulation Results

On the other side, we convert the LTL formula φ to a Büchi automaton (Fig. 5.2 (Left)) and then establish the product automaton $S_{\Sigma,\xi} \times_A B_\varphi$ (Fig. 5.2 (Right)). We design $S_{\Sigma,\xi} \times_A B_\varphi$ to be the supervisor for $S_{\Sigma,\xi}$. After the implementation of $S_{\Sigma,\xi} \times_A B_\varphi$ to the robot system, the controlled system achieves the LTL formula φ. Moreover, the simulation results of two feasible paths initializing from R_{31} and satisfying φ are shown in Fig. 5.3.

5.5 Conclusion

This chapter provided a bisimulation-based approach to controlling the multi-affine system for temporal logic specifications in a rectangularly partitioned state space.

Two novel methods were derived to control the multi-affine system on rectangles. One is based on the exit sub-region to drive all trajectories starting from a rectangle to exit only through a facet, which enlarges the classes of control systems in the context of existing literature [81]. The other provides a solution for the convergence problem by stabilizing the multi-affine system towards a desired point. With the proposed control methods, a finitely abstracted transition system was constructed and it was shown to be bisimilar to the rectangular transition system of the multi-affine system. Therefore, the controller synthesis for the multi-affine system to enforce the temporal logic specification can be achieved by designing a bisimilarity enforcing supervisor for the abstracted transition system and then mapped into continuous control signals.

Chapter 6

Conclusions and Future Works

This thesis investigated the controller synthesis for bisimulation equivalence with respect to both discrete event systems and continuous systems. For discrete event systems, bisimilarity supervisory control has been studied under centralized, decentralized and distributed frameworks, respectively. For continuous systems, a bisimulation-based approach has been proposed to control a class of nonlinear systems for temporal logic specifications.

The main contributions of the thesis are described as follows.

- A systematic way was presented for bisimilarity supervisory control with respect to the most general case which allows the plant, specification and supervisor to be nondeterministic. Specifically, we proposed the notion of synchronous simulation-based state controllability as the sufficient condition for the existence of a bisimilarity enforcing supervisor. When the existence condition holds, a bisimilarity enforcing supervisor can be effectively constructed. When the

existence condition does not hold, the synthesis of achievable sub-specifications has been further explored.

- We investigated bisimilarity supervisory control for deterministic specifications. The notion of synchronous simulation-based controllability was introduced as the necessary and sufficient condition for the existence of bisimilarity control with respect to deterministic specifications. This condition can be checked with polynomial complexity. Moreover, the synthesis of supremal synchronously simulation-based controllable sub-specifications has been studied.

- Bisimilarity supervisory control has been extended from centralized framework to decentralized framework. A novel automata-based structure was proposed for decentralized bisimilarity control, under which three architectures, a conjunctive architecture a disjunctive architecture and a general architecture, were built up according to different decision making rules. For these three architectures, necessary and sufficient conditions are provided for the existence of deterministic decentralized bisimilarity control, respectively. The synthesis of achievable sup-specifications was further developed.

- We studied bisimilarity control of distributed discrete event systems which consist of multiple interacting local modules. A sufficient condition was stated for the existence of distributed bisimilarity control. When the given specification satisfies this condition, a set of local supervisors enforcing bisimilarity can be constructed. Otherwise, we explored the calculation of achievable sub-specifications to enable the existence of distributed bisimilarity control. In

particular, we focused on deterministic supervisors. The notion of separable controllability was introduced as the necessary and sufficient condition for the existence of deterministic distributed bisimilarity control. In addition, the synthesis of infimal sup-specifications was investigated. The comparisons of our results with the centralized monolithic ones were presented as well.

- A unified and automatic control framework has been proposed to control of a class of nonlinear systems for temporal logic specifications. It provides not only a tighter connection between continuous dynamics and discrete dynamics but also a practical complexity for implementation. A novel control method based on exit sub-regions has been established for controlling the system on rectangles. It enlarges the class of systems which possesses bisimilarly abstracted models. Furthermore, the calculation of bisimilarly abstracted systems is not expensive, involving polyhedral operator only.

This thesis may represent an interesting step towards the controller synthesis for bisimulation equivalence and its application to achieve temporal logic specifications. Here, the work on bisimilarity control of discrete event systems is limited to the class of plants with all events are observable. Although it is true in lots of engineering systems, in some applications, e.g. a lack of measurement, only some of the events are observable. Therefore, bisimilarity enforcing supervisor should be developed in such a way that bisimulation equivalence can always be achieved irrespective of partial observation. Moreover, decentralized bisimilarity control proposed in this thesis concerns with supervisors without communication. By establishing new communica-

tion architectures, it is possible to allow the interaction among supervisors to resolve ambiguities and determine correct control actions. This would be important for the applications with limited sensing and actuation of the events.

Furthermore, bisimilarity control could be extended to deal with more general systems, such as polynomial dynamics [98] through a relaxed notion of bisimulation. Another interesting direction is to incorporate the time information into the specification. In this case, based on Metric Temporal Logic (MTL) specifications [99], the occurrence time of the events can be characterized, with its great application to task optimization emerging in manufacturing systems, cloud computing, smart grids, transportation, parallel computing, multi-processing, and distributed process control.

List of Publications

- Journal Papers:

  1. Y. Sun, H. Lin and Ben M. Chen, "An Input-Output Simulation Approach to Controlling Multi-Affine Systems for Linear Temporal Logic Specifications," International Journal of Control, vol. 85, no. 10, October 2012, pages 1464-1476.
  2. Y. Sun, H. Lin and Ben M. Chen, "Bisimilarity Enforcing Supervisory Control for Deterministic Specifications", submitted for publication, 2012.
  3. Y. Sun, H. Lin and Ben M. Chen, "Decentralized Supervisory Control of Discrete Event Systems for Bisimulation Equivalence", submitted for publication, 2012.
  4. Y. Sun, H. Lin and Ben M. Chen, "Supervisory Control of Distributed Discrete Event Systems for Bisimulation Equivalence", submitted for publication, 2012.

- Conference Papers:

  1. Y. Sun and H. Lin, "Bisimilarity Enforcing Supervisory control of Nonde-

terministic Discrete Event Systems", American Control Conference (ACC 2012), pages 6102-6107, 2012.

2. Y. Sun, H. Lin and Ben M. Chen, "Computation for Supremal Simulation-based Controllable Subautomata", IEEE International Conference on Control and Automation (ICCA 2010), pages 1450 - 1455, 2010.

3. Y. Sun, H. Lin and Ben M. Chen, "Decentralized Bisimilarity Supervisory Control of Discrete Event Systems", Chinese Control Conference (CCC2012), accepted for publication, 2012.

4. Y. Sun, H. Lin and Ben M. Chen, "Computation for Supremal Simulation-Based Controllable and Strong Observable Subautomata", Chinese Control Conference (CCC2012), accepted for publication, 2012.

Bibliography

[1] P. Dewan and J. Riedl, "Toward computer-supported concurrent software engineering," *Computer*, vol. 26, no. 1, pp. 17–27, 1993.

[2] A. Aarsten, D. Brugali, and G. Menga, "Designing concurrent and distributed control systems," *Communications of the ACM*, vol. 39, no. 10, pp. 50–58, 1996.

[3] J. Teutsch and E. Hoffman, "Aircraft in the future atm system-exploiting the 4d aircraft trajectory," in *Proceeding of Digital Avionics Systems Conference*, vol. 1. IEEE, 2004, pp. 3–B.

[4] H. Hansson, L. Lawson, O. Bridal, C. Eriksson, S. Larsson, H. Lon, and M. Stromberg, "Basement: an architecture and methodology for distributed automotive real-time systems," *IEEE Transactions on Computers*, vol. 46, no. 9, pp. 1016–1027, 1997.

[5] J. Cook, J. Sun, and J. Grizzle, "Opportunities in automotive powertrain control applications," in *Proceedings of the International Conference on Control Applications*, vol. 1. IEEE, 2002, pp. xlii–xlli.

[6] A. Davare, Q. Zhu, M. Di Natale, C. Pinello, S. Kanajan, and A. Sangiovanni-

Vincentelli, "Period optimization for hard real-time distributed automotive systems," in *Proceedings of the 44th annual Design Automation Conference*. ACM, 2007, pp. 278–283.

[7] A. Winfield, J. Sa, M. Fernandez-Gago, C. Dixon, and M. Fisher, "On formal specification of emergent behaviours in swarm robotic systems," *International journal of advanced robotic systems*, vol. 2, no. 4, pp. 363–370, 2005.

[8] C. Belta, A. Bicchi, M. Egerstedt, E. Frazzoli, E. Klavins, and G. Pappas, "Symbolic planning and control of robot motion [grand challenges of robotics]," *Robotics & Automation Magazine, IEEE*, vol. 14, no. 1, pp. 61–70, 2007.

[9] H. Kress-Gazit, G. Fainekos, and G. Pappas, "Where's waldo? sensor-based temporal logic motion planning," in *Proceedings of the IEEE International Conference on Robotics and Automation*. IEEE, 2007, pp. 3116–3121.

[10] N. Chabrier-Rivier, M. Chiaverini, V. Danos, F. Fages, and V. Schächter, "Modeling and querying biomolecular interaction networks," *Theoretical Computer Science*, vol. 325, no. 1, pp. 25–44, 2004.

[11] G. Bernot, J. Comet, A. Richard, and J. Guespin, "Application of formal methods to biological regulatory networks: extending Thomas' asynchronous logical approach with temporal logic," *Journal of Theoretical Biology*, vol. 229, no. 3, pp. 339–347, 2004.

[12] L. Calzone, F. Fages, and S. Soliman, "Biocham: an environment for model-

ing biological systems and formalizing experimental knowledge," *Bioinformatics*, vol. 22, no. 14, pp. 1805–1807, 2006.

[13] J. Thistle and W. Wonham, "Control problems in a temporal logic framework," *International Journal of Control*, vol. 44, no. 4, pp. 943–976, 1986.

[14] J. Knight and K. Passino, "Decidability for a temporal logic used in discrete-event system analysis," *International Journal of Control*, vol. 52, no. 6, pp. 1489–1506, 1990.

[15] A. Ulusoy, S. Smith, C. Xu, and C. Belta, "Robust multi-robot optimal path planning with temporal logic constraints," in *Proceedings of IEEE International Conference on Robotics and Automation, to appear*, 2012.

[16] S. Eker, M. Knapp, K. Laderoute, P. Lincoln, J. Meseguer, and K. Sonmez, "Pathway logic: Symbolic analysis of biological signaling," in *Proceedings of the Pacific Symposium on Biocomputing*, vol. 7, 2002, pp. 400–412.

[17] P. Tabuada and G. Pappas, "Linear time logic control of discrete-time linear systems," *IEEE Transactions on Automatic Control*, vol. 51, no. 12, pp. 1862–1877, 2006.

[18] C. Belta, V. Isler, and G. Pappas, "Discrete abstractions for robot motion planning and control in polygonal environments," *IEEE Transactions on Robotics*, vol. 21, no. 5, pp. 864–874, 2005.

[19] C. Belta, A. Bicchi, M. Egerstedt, E. Frazzoli, E. Klavins, and G. Pappas,

"Symbolic planning and control of robot motion [grand challenges of robotics]," *Robotics and Automation Magazine, IEEE*, vol. 14, no. 1, pp. 61–70, March 2007.

[20] C. Cassandras and S. Lafortune, *Introduction to discrete event systems*. Springer, 2008.

[21] P. Ramadge and W. Wonham, "Supervisory control of a class of discrete event processes," *SIAM journal on control and optimization*, vol. 25, pp. 206–230, 1987.

[22] ——, "The control of discrete event systems," *Proceedings of the IEEE*, vol. 77, no. 1, pp. 81 –98, Jan. 1989.

[23] R. Cieslak, C. Desclaux, A. Fawaz, and P. Varaiya, "Supervisory control of discrete-event processes with partial observations," *IEEE Transactions on Automatic Control*, vol. 33, no. 3, pp. 249 –260, Mar. 1988.

[24] J. Prosser, M. Kam, and H. Kwatny, "Online supervisor synthesis for partially observed discrete-event systems," *IEEE Transactions on Automatic Control*, vol. 43, no. 11, pp. 1630–1634, 1998.

[25] T. Yoo and S. Lafortune, "On the computational complexity of some problems arising in partially-observed discrete-event systems," in *Proceedings of American Control Conference*, vol. 1. IEEE, 2001, pp. 307–312.

[26] R. Kumar, S. Jiang, C. Zhou, and W. Qiu, "Polynomial synthesis of supervisor for partially observed discrete-event systems by allowing nondeterminism in control," *IEEE Transactions on Automatic Control*, vol. 50, no. 4, pp. 463–475, 2005.

[27] E. Chen and S. Lafortune, "Dealing with blocking in supervisory control of discrete-event systems," *IEEE Transactions on Automatic Control*, vol. 36, no. 6, pp. 724–735, 1991.

[28] R. Kumar and M. Shayman, "Nonblocking supervisory control of nondeterministic systems via prioritized synchronization," *IEEE Transactions on Automatic Control*, vol. 41, no. 8, pp. 1160–1175, 1996.

[29] C. Ma and W. Wonham, "Nonblocking supervisory control of state tree structures," *IEEE Transactions on Automatic Control*, vol. 51, no. 5, pp. 782–793, 2006.

[30] R. Su, J. van Schuppen, and J. Rooda, "Model Abstraction of Nondeterministic Finite-State Automata in Supervisor Synthesis," *IEEE Transactions on Automatic Control*, vol. 55, no. 99, pp. 2527–2541.

[31] W. Wonham and P. Ramadge, "Modular supervisory control of discrete-event systems," *Mathematics of Control, Signals, and Systems (MCSS)*, vol. 1, no. 1, pp. 13–30, 1988.

[32] Y. Chen, S. Lafortune, and F. Lin, "Modular supervisory control with priorities for discrete event systems," in *Proceedings of IEEE Conference on Decision and Control*, 1995, pp. 409–415.

[33] K. Rohloff and S. Lafortune, "On the computational complexity of the verification of modular discrete-event systems," in *Proceedings of IEEE Conference on Decision and Control*, vol. 1. IEEE, 2002, pp. 16–21.

[34] Y. Brave, "Control of discrete event systems modeled as hierarchical state machines," *IEEE Transactions on Automatic Control*, vol. 38, no. 12, pp. 1803–1819, 1993.

[35] K. Wong and W. Wonham, "Hierarchical control of discrete-event systems," *Discrete Event Dynamic Systems*, vol. 6, no. 3, pp. 241–273, 1996.

[36] H. Zhong and W. Wonham, "On the consistency of hierarchical supervision in discrete-event systems," *IEEE Transactions on Automatic Control*, vol. 35, no. 10, pp. 1125–1134, 1990.

[37] K. Rudie and W. Wonham, "Think globally, act locally: Decentralized supervisory control," *IEEE Transactions on Automatic Control*, vol. 37, no. 11, pp. 1692–1708, 1992.

[38] S. Tripakis, "Undecidable problems of decentralized observation and control on regular languages," *Information Processing Letters*, vol. 90, no. 1, pp. 21–28, 2004.

[39] Y. Li and W. Wonham, "On supervisory control of real-time discrete-event systems," *Information sciences*, vol. 46, no. 3, pp. 159–183, 1988.

[40] K. Wong, J. Thistle, R. Malhame, and H. Hoang, "Supervisory control of distributed systems: Conflict resolution," *Discrete Event Dynamic Systems*, vol. 10, no. 1, pp. 131–186, 2000.

[41] R. Milner, *Communication and concurrency*, 1989.

[42] H. Qin and P. Lewis, "Factorization of finite state machines under observational equivalence," in *Proceedings of International Conference on Concurrency Theory*, pp. 427–441, 1990.

[43] P. Madhusudan and P. Thiagarajan, "Branching time controllers for discrete event systems," *Theoretical Computer Science*, vol. 274, no. 1-2, pp. 117–149, 2002.

[44] P. Tabuada, "Controller synthesis for bisimulation equivalence," *Systems & Control Letters*, vol. 57, no. 6, pp. 443–452, 2008.

[45] C. Zhou, R. Kumar, and S. Jiang, "Control of nondeterministic discrete-event systems for bisimulation equivalence," *IEEE Transactions on Automatic Control*, vol. 51, no. 5, pp. 754–765, 2006.

[46] C. Zhou and R. Kumar, "A small model theorem for bisimilarity control under partial observation," *IEEE Transactions on Automation Science and Engineering*, vol. 4, no. 1, pp. 93–97, 2007.

[47] ——, "Bisimilarity enforcement for discrete event systems using deterministic control," *IEEE Transactions on Automatic Control, (In Press)*.

[48] ——, "Bisimilarity enforcement for discrete event systems using deterministic control," *IEEE Transactions on Automatic Control*, vol. 56, no. 12, pp. 2986 – 2991, 2011.

[49] T. Yoo and S. Lafortune, "A general architecture for decentralized supervisory

control of discrete-event systems," *Discrete Event Dynamic Systems*, vol. 12, no. 3, pp. 335–377, 2002.

[50] ——, "Decentralized supervisory control with conditional decisions: Supervisor existence," *IEEE Transactions on Automatic Control*, vol. 49, no. 11, pp. 1886–1904, 2004.

[51] S. Ricker and K. Rudie, "Know means no: Incorporating knowledge into discrete-event control systems," *IEEE Transactions on Automatic Control*, vol. 45, no. 9, pp. 1656–1668, 2000.

[52] K. Schmidt, T. Moor, and S. Perk, "Nonblocking hierarchical control of decentralized discrete event systems," *IEEE Transactions on Automatic Control*, vol. 53, no. 10, pp. 2252–2265, 2008.

[53] S. Takai and T. Ushio, "Reliable decentralized supervisory control of discrete event systems," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 30, no. 5, pp. 661–667, 2000.

[54] F. Liu and H. Lin, "Reliable supervisory control for general architecture of decentralized discrete event systems," *Automatica*, vol. 46, no. 9, pp. 1510–1516, 2010.

[55] S. Park and K. Cho, "Technical communique: Decentralized supervisory control of discrete event systems with communication delays based on conjunctive and permissive decision structures," *Automatica* , vol. 43, no. 4, pp. 738–743, 2007.

[56] G. Barrett and S. Lafortune, "Decentralized supervisory control with communicating controllers," *IEEE Transactions on Automatic Control*, vol. 45, no. 9, pp. 1620–1638, 2000.

[57] J. van Schuppen, "Decentralized control with communication between controllers," *Unsolved problems in mathematical systems and control theory*, p. 144, 2004.

[58] E. Bonabeau, M. Dorigo, and G. Theraulaz, *Swarm Intelligence*. England: Oxford University Press: Oxford, 1999.

[59] E. Yang and D. Gu, "Multiagent reinforcement learning for multi-robot systems: A survey," *Department of Computer Science, Univeristy of Essex, Tech. Rep*, 2004.

[60] X. Li and Y. Xi, "Distributed connected coverage control for groups of mobile agents," *International Journal of Control*, vol. 83, no. 7, pp. 1347–1363, 2010.

[61] S. Takai and T. Ushio, "Supervisory control of a class of concurrent discrete event systems under partial observation," *Discrete Event Dynamic Systems*, vol. 15, no. 1, pp. 7–32, 2005.

[62] R. Su, "Supervisory control of concurrent discrete-event systems," in *Proceedings of the IEEE Conference on Decision and Control*. IEEE, 2011, pp. 1811–1816.

[63] P. Gohari and W. Wonham, "On the complexity of supervisory control design in the rw framework," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 30, no. 5, pp. 643–652, 2000.

[64] C. Zhou, R. Kumar, and R. Sreenivas, "Decentralized modular control of concurrent discrete event systems," in *Proceedings of the IEEE Conference on Decision and Control*. IEEE, 2007, pp. 5918–5923.

[65] K. Rohloff and S. Lafortune, "The verification and control of interacting similar discrete-event systems," *SIAM Journal on Control and Optimization*, vol. 45, no. 2, pp. 634–667, 2006.

[66] M. De Queiroz and J. Cury, "Modular control of composed systems," in *Proceedings of the American Control Conference*, vol. 6. IEEE, 2000, pp. 4051–4055.

[67] Y. Willner and M. Heymann, "Supervisory control of concurrent discrete-event systems," *International Journal of Control*, vol. 54, no. 5, pp. 1143–1169, 1991.

[68] B. Gaudin and H. March, "Modular supervisory control of a class of concurrent discrete event systems," in *Proceedings WODES'04, Workshop on Discrete-Event Systems*, 2004, pp. 181–186.

[69] S. Jiang and R. Kumar, "Decentralized control of discrete event systems with specializations to local control and concurrent systems," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 30, no. 5, pp. 653 –660, Oct. 2000.

[70] K. Wong and S. Lee, "Structual decentralized control of concurrent discrete-event systems," *European Journal of Control*, vol. 8, no. 1, pp. 477–491, 2002.

[71] J. Komenda and J. van Schuppen, "Supremal sublanguages of general specifica-

tion languages arising in modular control of discrete-event systems," in *Proceedings of the IEEE Conference on Decision and Control*, 2005, pp. 2775 – 2780.

[72] J. Komenda, J. van Schuppen, B. Gaudin, and H. Marchand, "Supervisory control of modular systems with global specification languages," *Automatica*, vol. 44, no. 4, pp. 1127–1134, 2008.

[73] R. Alur and D. Dill, "A theory of timed automata," *Theoretical computer science*, vol. 126, no. 2, pp. 183–235, 1994.

[74] R. Alur, C. Courcoubetis, T. Henzinger, and P. Ho, "Hybrid automata: An algorithmic approach to the specification and verification of hybrid systems," *Hybrid systems*, pp. 209–229, 1993.

[75] A. Puri and P. Varaiya, "Decidability of hybrid systems with rectangular differential inclusions," in *Computer Aided Verification*. Springer, 1994, pp. 95–104.

[76] G. Lafferriere, G. Pappas, and S. Sastry, "O-minimal hybrid systems," *Mathematics of Control, Signals, and Systems*, vol. 13, no. 1, pp. 1–21, 2000.

[77] G. Fainekos, H. Kress-Gazit, and G. Pappas, "Hybrid controllers for path planning: A temporal logic approach," in *Proceedings of the IEEE Conference on Decision and Control and European Control Conference*, 2005, pp. 4885–4890.

[78] L. Habets and J. van Schuppen, "A control problem for affine dynamical systems on a full-dimensional polytope," *Automatica*, vol. 40, no. 1, pp. 21–35, 2004.

[79] R. Alur, T. Dang, and F. Ivancic, "Predicate abstraction for reachability analysis

of hybrid systems," *ACM Transactions on Embedded Computing Systems*, vol. 5, no. 1, pp. 152–199, 2006.

[80] A. Tiwari and G. Khanna, "Series of abstractions for hybrid automata," in *Hybrid Systems: Computation and Control*. Springer, 2002, pp. 465–478.

[81] C. Belta and L. Habets, "Controlling a class of nonlinear systems on rectangles," *IEEE Transactions on Automatic Control*, vol. 51, no. 11, pp. 1749–1759, 2006.

[82] L. Habets, M. Kloetzer, and C. Belta, "Control of rectangular multi-affine hybrid systems," in *Proceedings of the IEEE Conference on Decision and Control*, 2006, pp. 2619–2624.

[83] R. Alur, T. Henzinger, G. Lafferriere, and G. Pappas, "Discrete abstractions of hybrid systems," *Proceedings of the IEEE*, vol. 88, no. 7, pp. 971–984, 2002.

[84] N. Jones, "Space-bounded reducibility among combinatorial problems*," *Journal of Computer and System Sciences*, vol. 11, no. 1, pp. 68–85, 1975.

[85] M. Karimadini and H. Lin, "Guaranteed global performance through local coordinations," *Automatica*, vol. 47, no. 5, pp. 890–898, 2011.

[86] R. Kumar and V. Garg, "Modeling and control of logical discrete event systems(Book)," *Boston, MA: Kluwer Academic Publishers, 1995.*, 1995.

[87] J. Fernandez, "An implementation of an efficient algorithm for bisimulation equivalence," *Science of Computer Programming*, vol. 13, no. 2-3, pp. 219–236, 1990.

[88] K. Rudie and J. Willems, "The computational complexity of decentralized discrete-event control problems," *IEEE Transactions on Automatic Control*, vol. 40, no. 7, pp. 1313–1319, 1995.

[89] K. Ogawa, "Economic development and time preference schedule: The case of japan and east asian nics," *Journal of Development Economics*, vol. 42, no. 1, pp. 175–195, 1993.

[90] V. Volterra, "Fluctuations in the abundance of a species considered mathematically," *Nature*, vol. 118, no. 2972, pp. 558–560, 1926.

[91] A. Lotka, *Elements of physical biology*. Williams & Wilkins company, 1925.

[92] C. Belta, "On controlling aircraft and underwater vehicles," in *Proceedings of the IEEE International Conference on Robotics and Automation*, vol. 5, 2004, pp. 4905–4910.

[93] S. Sastry, *Nonlinear systems: analysis, stability, and control*. Springer Verlag, 1999.

[94] S. Berman, Á. Halász, and V. Kumar, "MARCO: a reachability algorithm for multi-affine systems with applications to biological systems," *Hybrid Systems: Computation and Control*, pp. 76–89, 2007.

[95] M. Kloetzer and C. Belta, "A fully automated framework for control of linear systems from temporal logic specifications," *IEEE Transactions on Automatic Control*, vol. 53, no. 1, pp. 287–297, 2008.

[96] P. Wolper, M. Vardi, and A. Sistla, "Reasoning about infinite computation paths," in *24th Annual Symposium on Foundations of Computer Science*, 1983, pp. 185–194.

[97] G. De Giacomo and M. Vardi, "Automata-theoretic approach to planning for temporally extended goals," *Recent Advances in AI Planning*, pp. 226–238, 2000.

[98] R. Benedetto, "Examples of wandering domains in p-adic polynomial dynamics," *Comptes Rendus Mathematique*, vol. 335, no. 7, pp. 615–620, 2002.

[99] R. Koymans, "Specifying real-time properties with metric temporal logic," *Real-Time Systems*, vol. 2, no. 4, pp. 255–299, 1990.