Bidirectional and Multiple Reconfiguration of Discrete-Event Systems

by

Jiachen Zhang

Technical Report
Systems Control Group
Department of Electrical and Computer Engineering
University of Toronto

Abstract

Bidirectional and Multiple Reconfiguration of Discrete-Event Systems

Jiachen Zhang
Systems Control Group
Department of Electrical and Computer Engineering
University of Toronto
2019-07-14

Owing to the complexity and flexibility of modern industry, a system often has multiple operation modes and the reconfiguration mechanism is often required to be bidirectional. The unidirectional reconfiguration approach for discrete-event systems has been proved to effectively solve one-way reconfiguration problems modeled from real scenarios but fails to handle bidirectional reconfiguration problems. In this report, we study the fundamentals of bidirectional reconfiguration and multiple reconfiguration of discrete-event systems. Specifically, we introduce a bidirectional reconfiguration specification by analyzing states and events in each plant component and extend similar techniques to multiple reconfiguration in both monolithic and localized architectures. The proposed approaches are compatible with supervisory control theory, so the resulting supervisors can regulate both reconfiguration and normal behaviors of the system. Finally, the trigger requirement and guaranteed reachability are studied as notable issues in applications.

Contents

List of Symbols and Abbreviations

L	Language	8
$L(\mathbf{G})$	Closed behavior of $\mathbf{G}$	10
$L_m(\mathbf{G})$	Marked behavior of $\mathbf{G}$	10
P_i^G	Aggregated mode predicate of $\mathbf{Mode}_i$ on Q of $\mathbf{G}$	31
P_i^{RG}	Segregated mode predicate of $\mathbf{Mode}_i$ on Q^{RG} of $\mathbf{RG}$	33
$Pwr(\Sigma)$	Power set associated with event alphabet Σ	10
Q_m	Subset of marked states of DES	8
Γ	Control pattern	10
Σ	Event alphabet of DES	8
Σ^*	Set of all finite strings over Σ	8
Σ^+	Set of all nonempty finite strings over Σ	8
Σ_c	Event alphabet of controllable events	8
Σ_u	Event alphabet of uncontrollable events	8
δ	Transition function of DES	8
ϵ	Empty string	8
$\overline{L}$	Prefix closure of language L	8
$\mathbf{G}$	Plant DES	8
$\mathbf{G}^k$	The k^{th} component of the plant	11

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

This section introduces the subject of this report. First, the importance of the problem studied is explained. We provide examples in Section 1.1 to clarify the notion of bidirectional reconfiguration and highlight the complexity of the bidirectional reconfiguration in systems with multiple modes to justify the report subject. Section 1.2 reviews literature regarding the proposed reconfiguration approaches and pinpoints the strengths and weaknesses of each existing approach. Finally, Section 1.3 outlines the report organization.

1.1 Motivation

Adaptability and evolution are critical notions of natural self-organizing systems [3]. Any natural system should adapt itself to changes in its environment in order to extend its life or increase its productivity. Similarly, artificial systems need to be subject to changes. Reconfiguration aims to plan and to execute systematic strategies in response to those changes.

Reconfiguration in a system can be informally defined as the operation to switch from one mode of the system to another [4]. A mode is an organization of components or operations of a system. Thus, a reconfiguration refers to components replacement or operational reorganization in a system. Consider a car running on a road while a tire suddenly blows. To manually put on the spare tire is a reconfiguration which refers to component replacement. On the other hand, consider a fitness treadmill. To switch from

the energy saving mode to the running mode is also a reconfiguration which refers to operational reorganization.

In a system, the purpose of reconfiguration may be to fulfill multifunctional flexibility and fault tolerance requirements of relatively complex systems [5]. In the absence of reconfiguration, extra mechanisms must be designed and applied to the system for these requirements. Such mechanisms would increase the complexity of the system.

Reconfiguration can be generally divided into static reconfiguration, where the system must be shut down, and dynamic reconfiguration [6], where reconfiguration can be operated during the runtime of the system. Still consider the fitness treadmill example. The power of the treadmill is always on. The trainer just needs to press a button to wake up the treadmill to reconfigure to the running mode. This is a dynamic reconfiguration. For the running car example, if the driver wants to put on a spare tire, he has to stop the car for obvious physical reasons. This is a static reconfiguration. However, stopping the car to replace a tire will waste time and money for gasoline. That's why in practice, dynamic reconfiguration is usually desired as it's less disruptive.

For reconfiguration, users are not necessarily interested in the bottom-level details of a system, since they care more about different components and operations [7]. Therefore, there needs to be a convenient abstract framework to represent a system in a high level. That's why discrete-event systems models may be appropriate.

A discrete-event system (DES) is a dynamic system [1, 8, 9]; it has a discrete state space and a state transition structure. The state space of a DES is usually a finite set of elements, called states. The state transition structure of a DES is not given by differential or difference equations, which serves as one of the properties that distinguish DES from other types of dynamic systems. Besides, a DES is considered asynchronous and event-driven since it can reside at a state for a period of time until it goes to another state by the occurrence of an event. Namely in such a system the logical order of events is of concern but not the time at which the events occur [10]. Therefore, DES can represent high-level logical behaviors of a system. Logistic systems, computer-controlled traffic systems, communication protocols, and database management systems are a few practical examples of systems whose behaviors can be modeled with DES.

Based on finite state automata, supervisory control theory (SCT) framework is a

popular framework for DES. supervisory control theory divides events in a DES into controllable events and uncontrollable events. By doing so, a DES can be controlled in SCT framework.

Different modes of a discrete-event system in SCT framework can be defined according to different event alphabets and all modes can be modeled altogether in a DES or in the synchronous product of several DES. The reconfiguration of DES can then be defined as the switching from one mode distinguished by an event alphabet to another. However, this type of reconfiguration has been formulated in a unidirectional fashion so that the bidirectional reconfiguration of DES has to be realized by remodeling and repeating the approach [1].

In modern industry, since systems tend to be complex and have multiple modes [11, 12, 13, 14, 15], multiple reconfiguration is in demand. Besides, owing to the flexibility and adaptability requirements, systems also require bidirectional reconfiguration since once they go to another modes, they still need an option to go back [16, 17, 18, 19]. DES can model lots of systems in an abstract high level. That's why bidirectional and multiple reconfiguration of DES is an important topic to study on.

Specifically, this report aims at a general solution to two relatively complicated reconfiguration problems of DES in SCT framework, i.e., the bidirectional dynamic reconfiguration and the multiple dynamic reconfiguration of discrete-event systems.

1.2 Literature Review

Efforts to study the notion of reconfiguration started in the 1970s [20]. Reconfiguration was explored in classical control systems [21], power systems [22], embedded systems [23], computer networks [24], hybrid systems [25], manufacturing systems [26], and discrete-event systems [27, 28, 29].

There have been several works about reconfiguration on Petri-net-based discrete-event systems [30, 31, 32].

Finite state automata and Petri nets [33] have competed with each other for years to represent the better tool to model discrete-event systems. Thus, there are a number of reconfiguration strategies applied to Petri-net-based discrete-event systems, e.g., [34, 35].

Petri net reconfiguration has extensively concentrated on reconfigurable manufacturing systems (see, e.g., [36, 37, 38, 39, 40, 41]). Along this way, [42] employs recursive Petri nets [43] to solve the reconfiguration problems of DES. This approach proposes the concept of "feature" to manage dynamic reconfiguration. Specifically, a feature has two functions that are essential for a successful reconfiguration: transition activation (deactivation) and interruption. This work has a contribution to the behavioral verifiability but fails to handle the nonblocking issue. Though with high complexity, this approach seems to surpass other Petri-net-driven reconfiguration approaches and realize the multiple reconfiguration mechanism.

The reconfiguration of finite-automaton-based DES is defined for plants and their close-loop interconnections with supervisory controllers, whereas Petri net reconfiguration is mostly applied to controllers. Logic controllers [44] based on Petri nets are quite small and localized in coordination with supervisors but suffer from poor performance in highly complex systems with more states.

Recently, [45] proposes a reconfiguration control method, based on the assumption that all modes of a dynamic reconfigurable discrete-event system are designed in advance and dynamic reconfiguration can be implemented only at some predefined reconfigurable states. This method is able to implement a required reconfiguration before the maximum permissible reconfiguration delay if a shortest legal firing sequence exists. However, this work has strongly assumed that all events are controllable, which is too restrictive in reality.

There are also several applications of Petri nets to the dynamic reconfiguration of complex systems in [46].

Apart from the Petri-net-based discrete-event systems, there are some works about the reconfiguration of automata-based discrete-event systems.

For instance, [47] proposes a state-based approach to reconfigure automata-based DES. Specifically, for each particular mode of the system, a supervisor is designed to associate with it. A specific coordinator is applied to switch from one of these supervisors to another when reconfiguration is in demand. Evidently, this approach requires many supervisors for a system with multiple modes and needs a complicated coordinator to activate and/or deactivate a particular subset of supervisors' events in a demand for

reconfiguration. However, this approach fails to consider the situations when a component of the system belongs to more than one mode. Moreover, [47] strongly assumes that the controllable and uncontrollable events are synchronous with respect to an internal clock, which is not applicable to many cases.

The notion of user operation modes [48] has been used for the reconfiguration of DES. In this scheme, each system has a number of specific transition graphs, namely the user operating modes, each of which represents a particular functional scenario corresponding to the coordination of the system's components. As a result, the reconfiguration here is to switch between distinct user operating modes. Though this idea is logically straightforward, no algorithm is designed to deal with the interaction of an arbitrarily large number of user operating modes. Thus, the scalability of this approach is problematic.

Another attempt to solve the reconfiguration problems of DES is the coordinated approaches [49]. These methods assign to each subsystem a coordinator, which is able to indirectly communicate with other coordinators to manage reconfiguration. The flaw still lies in the scalability owing to the use of large non-sparse matrices.

Furthermore, specifications separation is a strategy to handle reconfiguration in fault-tolerant systems [50]. In this scheme, a system under control has separate specifications before and after a fault. For the two specifications, there must be two different controllers, which have to be synthesized offline.

Though of great diversity, the above works fail to realize bidirectional reconfiguration of automata-based DES, let alone multiple reconfiguration.

The multi-modal approach can be used to address bidirectional reconfiguration of DES. This approach divides the states of a variety of supervisors into three classes: incompatible states [51], forbidden states, and pre-forbidden states [52]. The reconfiguration and behavioral requirements are then modeled by intramodal and intermodal specifications. The approach uses bidirectional reconfiguration events in intermodal specifications, but this might lead to blocking issues [4]. This approach also suffers from its complexity and its lack of extendibility to distributed systems.

The authors in [4] introduce a unidirectional reconfiguration approach for reconfiguration problems of DES in the SCT framework based on automata. However, naively applying this approach to bidirectional reconfiguration problems would result in blocking

issues. Fortunately, this approach has the potential to be refined, so that bidirectional reconfiguration and multiple reconfiguration mechanisms can be achieved without blocking. This report will expand this approach to some extent and overcome those blocking issues.

1.3 report Outline

In Chapter 2, we introduce necessary mathematical preliminaries about discrete-event systems and the unidirectional reconfiguration approach of DES.

In Chapter 3, we first fix bidirectional reconfiguration problems on parallel-mode systems for practicability and convenience. We then analyze the inadequacy of the unidirectional approach when solving bidirectional reconfiguration problems and discuss potential solutions intuitively. States and events of each plant component are then classified according to the modes to which they belong. Based on the classification of events, a bidirectional reconfiguration specification is constructed, under five reasonable assumptions. The adequacy of the problem formulation is discussed in Appendix C.

In Chapter 4, we briefly explain the inefficacy of the bidirectional reconfiguration approach for a system with more than two modes. A series of modifications of definitions in Chapter 3 are then employed to refine the reconfiguration specification. A core multiple reconfiguration specification and several extra multiple reconfiguration specifications for each pair of modes are then designed. This approach is proved to be a generalized version of the bidirectional reconfiguration approach. As the next step, behavioral specifications are managed by the supervisory control synthesis, which is proved to be compatible with the proposed bidirectional and multiple reconfiguration approaches. Finally, trigger behaviors and guaranteed reachability are studied as valuable issues in applications.

The approach in Appendix A is localized, while that in Chapter 4 is monolithic. The purpose of localization is to relax two of the six assumptions. This approach localizes an extra multiple reconfiguration specification with respect to each pair of modes on each plant component. As a result, the number of reconfiguration specifications increases and the structure of them is much simpler. The localized bidirectional reconfiguration approach is also briefly introduced.

Examples at the end of Chapter 3,4 and Appendix A illustrate the procedure to apply the proposed approaches. All formal proofs are provided in Appendix B. Moreover, the proposed reconfiguration approaches are applied to two benchmark production systems FESTO and EnAS in Appendix D.

Finally, discussion and conclusions are presented in Chapter 5.

Chapter 2

Mathematical Preliminaries

2.1 Discrete-Event Systems

Supervisory Control Theory (SCT) [1] employs the notion of regular languages to model discrete-event systems (DES). Regular languages can be represented by deterministic finite state automata, which form the backbone of efficient computations with the corresponding regular languages.

Given a finite alphabet Σ, Σ^+ is the set of all nonempty finite strings over Σ. Considering the empty string ϵ, the set of all finite strings over Σ is represented by $\Sigma^* := \Sigma^+ \dot{\cup} \{\epsilon\}$. The prefix relation is defined between two strings based on a partial order as $s_1 \leq s_2 \Leftrightarrow (\exists t \in \Sigma^*) s_2 := s_1 t$, so the *prefix closure* of a regular language L is

$$\overline{L} := \{s \in \Sigma^* | (\exists t \in L)\ s \leq t\} \tag{2.1}$$

Also, a regular language L is (*prefix*) *closed* if $L = \overline{L}$.

Supervisory control theory provides a theoretical solution to control DES modeled by finite deterministic automata. In particular, a DES is formally represented by a string generator, say

$$\mathbf{G} := (Q, \Sigma, \delta, q_o, Q_m)$$

Here Q is the set of states; Σ is the event alphabet, i.e., the set of event labels comprised of two disjoint sets: the *controllable* event set Σ_c, and the *uncontrollable* event set Σ_u,

namely $\Sigma = \Sigma_c \dot{\cup} \Sigma_u$; $\delta : Q \times \Sigma \to Q$ is the *partial transition function* [1]; q_o. is the *initial state*; and $Q_m \subseteq Q$ is the subset of *marked* states.

The following example **MACH** shows a DES in the SCT framework.

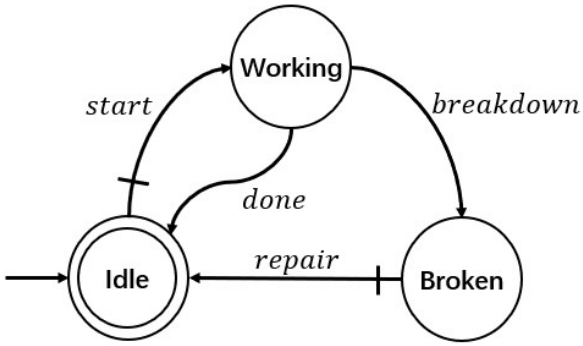

Figure 2.1: MACH

In Figure 2.1, a dash on a transition arrow means that the corresponding event is a controllable event; otherwise it is an uncontrollable event. This DES can also be written as the 5-tuple **MACH** = ({Idle, Working, Broken}, {*start, done, breakdown, repair*}, δ, Idle, {Idle}).

Note that in the supervisory controller synthesis software TCT, there is no dash on a transition arrow for controllable event labels. Instead, the controllable and uncontrollable event labels are represented by odd and even numbers, respectively.

The *closed behavior* $L(\mathbf{G})$ and the *marked behavior* $L_m(\mathbf{G})$ of $\mathbf{G}$ are the following languages

$$L(\mathbf{G}) := \{s \in \Sigma^* | \delta(q_o, s)!\} \tag{2.2}$$

$$L_m(\mathbf{G}) := \{s \in L(\mathbf{G}) | \delta(q_o, s) \in Q_m\} \tag{2.3}$$

Here $\delta(q_o, s)!$ means that $\delta(q_o, s)$ is defined.

In a DES $\mathbf{G} = (Q, \Sigma, \delta, q_o, Q_m)$, a state q is *reachable* if there is a string $s \in \Sigma^*$ with $\delta(q_o, s) = q$. A state q is *coreachable* if there is a string $s \in \Sigma^*$ such that $\delta(q, s) \in Q_m$. $\mathbf{G}$ is *nonblocking* if every reachable state is coreachable, or equivalently

$$L(\mathbf{G}) = \overline{L_m(\mathbf{G})}. \tag{2.4}$$

[1] For convenience, this one-step transition function could also be extended to a string-wise version, i.e. $\delta : Q \times \Sigma^* \to Q$

If $S \subset \Sigma^*$ and $\Sigma_o \subseteq \Sigma$, let $S\Sigma_o$ denote the set of strings of form $s\sigma$ with $s \in S$ and $\sigma \in \Sigma_o$. Then given a closed regular language M, a regular language K is *controllable* with respect to (M, Σ_u) if

$$\overline{K}\Sigma_u \cap M \subseteq \overline{K} \tag{2.5}$$

A *supervisory control* for $\mathbf{G}$ is a function $\mathbf{V} : L(\mathbf{G}) \to \Gamma$, where $\Gamma := \{\gamma \in Pwr(\Sigma) | \gamma \supseteq \Sigma_u\}$ is the set of *control patterns*. '$\mathbf{G}$ under supervision of $\mathbf{V}$' is written as $\mathbf{V}/\mathbf{G}$. Given a sublanguage $S \subseteq L_m(\mathbf{G})$, the marked behavior of $\mathbf{V}/\mathbf{G}$ is defined as $L_m(\mathbf{V}/\mathbf{G}) := L(\mathbf{V}/\mathbf{G}) \cap S$. $\mathbf{V}$ is a *marking nonblocking supervisory control* for the pair $(E, \mathbf{G})$ if

$$\overline{L_m(\mathbf{V}/\mathbf{G})} = L(\mathbf{V}/\mathbf{G}). \tag{2.6}$$

In practice, $\mathbf{V}$ is implemented by a *supervisor*, which can be taken as a generator **SUP** that represents the maximally permissive controlled behavior $L_m(\mathbf{V}/\mathbf{G})$ subject to a generator specification, say **SPEC**. In TCT notation, we denote this computation by

$$\mathbf{SUP} = \mathbf{supcon}(\mathbf{G}, \mathbf{SPEC}) \tag{2.7}$$

For details see [1].

For more than one DES, the standard operation to combine them is *synchronization*. Consider two alphabets Σ_1 and Σ_2, where it is allowed that $\Sigma_1 \cap \Sigma_2 \neq \emptyset$. Let $\Sigma = \Sigma_1 \cup \Sigma_2$. The *natural projection* of Σ^* onto Σ_i^*, $P_i : \Sigma^* \to \Sigma_i^*$ $(i = 1, 2)$, is defined according to

$$\begin{aligned} P_i(\epsilon) &:= \epsilon; \\ P_i(\sigma) &:= \begin{cases} \epsilon & \text{if } \sigma \notin \Sigma_i; \\ \sigma & \text{if } \sigma \in \Sigma_i; \end{cases} \\ P_i(s\sigma) &:= P_i(s)P_i(\sigma) \quad \text{if } s \in \Sigma^*, \sigma \in \Sigma. \end{aligned} \tag{2.8}$$

The action of P_i on a string s is just to erase all occurrences of σ in s such that $\sigma \notin \Sigma_i$. The inverse image function of P_i is defined as

$$P_i^{-1} : Pwr(\Sigma_i^*) \to Pwr(\Sigma^*) \tag{2.9}$$

Namely for $H \subseteq \Sigma_i^*$,

$$P_i^{-1}(H) = \{s \in \Sigma^* | P_i(s) \in H\} \tag{2.10}$$

For $L_1 \subseteq \Sigma_1^*, L_2 \subseteq \Sigma_2^*$, the *synchronous product* $L_1||L_2 \subseteq \Sigma^*$ is defined according to

$$L_1||L_2 := P_1^{-1}L_1 \cap P_2^{-1}L_2 \tag{2.11}$$

Here $s \in L_1||L_2$ iff $P_1(s) \in L_1$ and $P_2(s) \in L_2$. If $L_1 = L_m(\mathbf{G}_1)$ and $L_2 = L_m(\mathbf{G}_2)$, $\mathbf{G}_1$ and $\mathbf{G}_2$ can be thought as generating $L_1||L_2$ 'cooperatively' by agreeing to synchronize those events with labels σ which they possess in common.

Let

$$P_{i0} : \Sigma_i^* \to (\Sigma_1 \cap \Sigma_2)^* \qquad i = 1, 2 \tag{2.12}$$

be the natural projections; then it is true that

$$\begin{aligned} P_1(L_1||L_2) &= L_1 \cap P_{10}^{-1}(P_{20}L_2) \\ P_2(L_1||L_2) &= L_2 \cap P_{20}^{-1}(P_{10}L_1) \end{aligned} \tag{2.13}$$

In TCT notation, the computation to generate synchronous product is denoted by

$$\mathbf{G} = \mathbf{Sync}(\mathbf{G}_1, \mathbf{G}_2) \tag{2.14}$$

Here $L_m(\mathbf{G}) = L_m(\mathbf{G}_1)||L_m(\mathbf{G}_2)$, and $L(\mathbf{G}) = L(\mathbf{G}_1)||L(\mathbf{G}_2)$.

2.2 Unidirectional Reconfiguration of DES

A reconfiguration operation is a transition to transfer the system from one mode to another. Let $\mathbf{G} = (Q, \Sigma, \delta, q_o, Q_m)$ be a DES. In particular, $\mathbf{G} = \mathbf{G}^1||...||\mathbf{G}^h$ could be a synchronous product of several DES components. A *mode (configuration)* $\mathbf{Mode}_i$ $(i = 1, ..., n)$ of $\mathbf{G}$ is considered as a partial system of $\mathbf{G}$, which is also distinguished by Σ_i. Here $\Sigma_i \subseteq \Sigma$ is the event alphabet of $\mathbf{Mode}_i$. In practice, this event-based definition of mode applies to a wide range of systems. For instance, the modes might be distinguished by different plant components as long as their event alphabets are distinct, the modes might also be distinguished by different states of the same plant component according to

the events defined at those states.

Formally, for each plant component $\mathbf{G}^k = (Q^k, \Sigma^k, \delta^k, q_o^k, Q_m^k)$ $(k = 1, ..., h)$, the $\mathbf{Mode}_i$ $(i = 1, ..., n)$ involves $\mathbf{G}^k$, if $\Sigma_i \cap \Sigma^k \neq \emptyset$. Then $\mathbf{G}^k$ as a participant in $\mathbf{Mode}_i$ is represented by a partial system

$$\mathbf{G}_i^k := (Q_i^k, \Sigma_i^k, \delta_i^k, q_{o,i}^k, Q_{m,i}^k)$$

where $Q_i^k \subseteq Q^k$ is the set of states for $\mathbf{Mode}_i$ in $\mathbf{G}^k$; $\Sigma_i^k = \Sigma_i \cap \Sigma^k \subseteq \Sigma^k$ is the alphabet of events for $\mathbf{Mode}_i$ in $\mathbf{G}^k$; $q_{o,i}^k \in Q_i^k$ is the initial state of $\mathbf{Mode}_i$ in $\mathbf{G}^k$; $Q_{m,i}^k \subseteq Q_m^k$ is the set of marked states for $\mathbf{Mode}_i$ in $\mathbf{G}^k$; and $\delta_i^k : Q_i^k \times \Sigma_i^k \to Q_i^k$ is the partial transition function

$$\delta_i^k(q, \sigma) = \begin{cases} \delta^k(q, \sigma) & \text{if } q \in Q_i^k, \sigma \in \Sigma_i^k; \\ \text{not defined} & \text{otherwise.} \end{cases} \tag{2.15}$$

The $\mathbf{Mode}_i$ is then represented by $\mathbf{G}_i$, the synchronous product of these $\mathbf{G}_i^k$ such that $\Sigma_i \cap \Sigma^k \neq \emptyset$, i.e.

$$\mathbf{G}_i := ||\{\mathbf{G}_i^k | (k = 1, ..., h)\ \Sigma_i \cap \Sigma^k \neq \emptyset\} \tag{2.16}$$

The definition of mode (configuration) is illustrated in Figure 2.2.

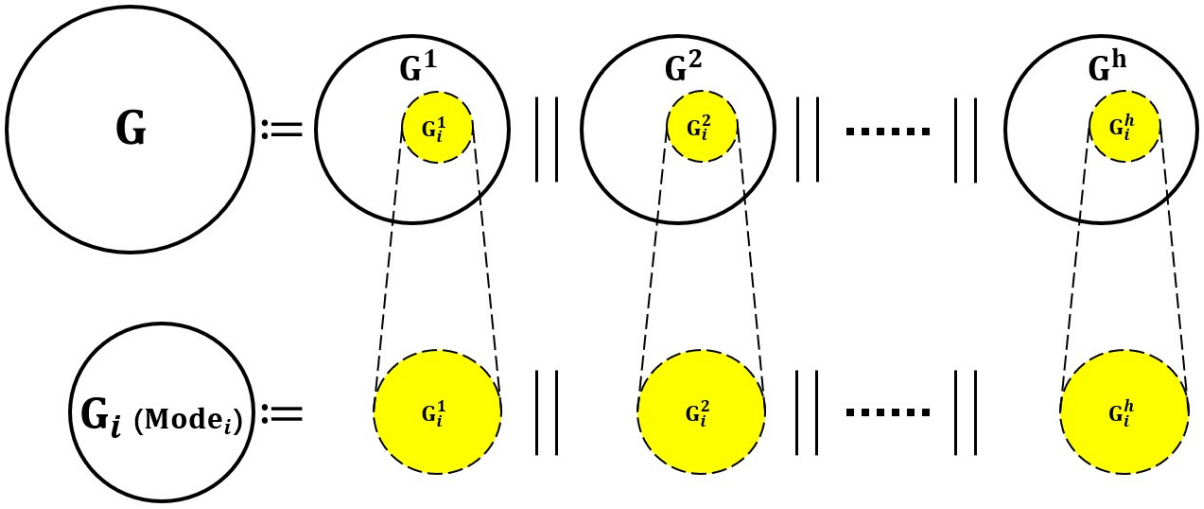

Figure 2.2: Mode (configuration)

In particular, a DES with three modes [2] is depicted in Figure 2.3. When the plant $\mathbf{G}$

[2]Imagine $\mathbf{G}$ represents a machine. The three modes are "Light-load", "Heavy-load", and "Maintenance". State q_0 means "ready to start". State q_1 means "start". State q_2 means "working in the light load conditions". State q_3 means

is the synchronization of multiple plant components, it is hard to directly recognize a mode from **G**. The reason is that different modes are intertwined in a complicated way in **G** owing to the synchronization. The purpose of the reconfiguration approach is to decouple the intertwined modes, preserve the dynamics of each mode, and introduce the reconfiguration mechanism.

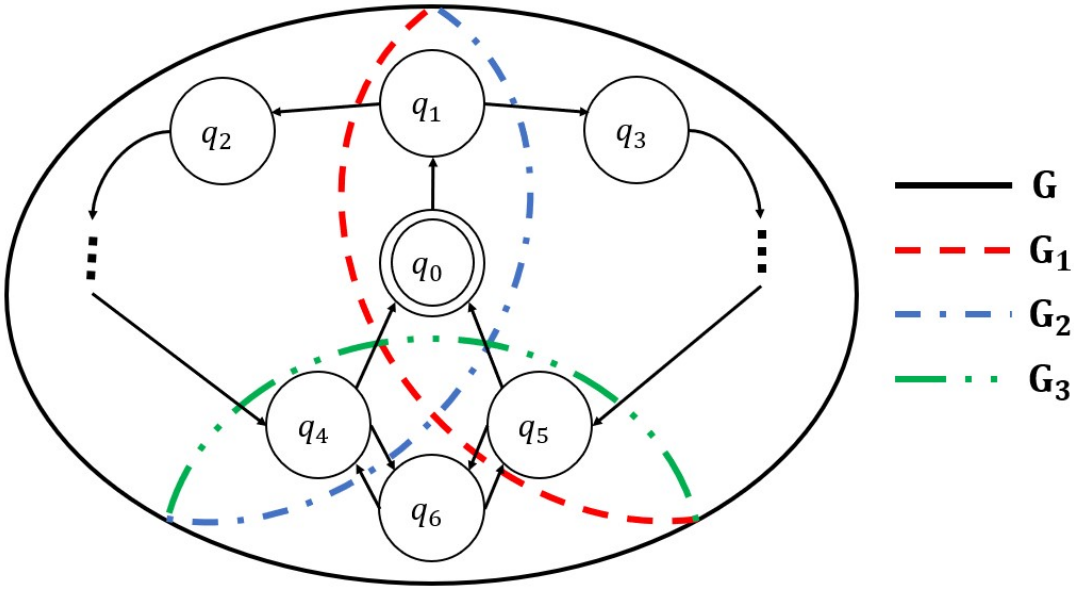

Figure 2.3: DES with three modes

Different modes are distinguished by the event alphabets, which are not necessarily pairwise disjoint. For $\mathbf{Mode}_i$ and $\mathbf{Mode}_j, i \neq j$, it is possible that $\Sigma_i = \Sigma_j$. But in most cases, it is true that

$$(\Sigma_i - \Sigma_j \neq \emptyset) \text{ or } (\Sigma_j - \Sigma_i \neq \emptyset). \tag{2.17}$$

At any time, **G** can only operate in one of the different modes, say $\mathbf{Mode}_i$; then the partial systems $\mathbf{G}_i^k$ and $\mathbf{G}_j^k$ $(\forall j \neq i)$ are active and inactive in each plant component $\mathbf{G}^k$, respectively.

A *unidirectional reconfiguration* specifies a source mode and a target mode. The source and target modes, corresponding to a reconfiguration task, are connected through a *reconfiguration event* (RE) which specifies a transition from the former mode to the latter. A RE is always controllable [3] since a reconfiguration operation needs to be under

"working in the heavy mode conditions". State q_4 means "work is done in the light load conditions". State q_5 means "work is done in the heavy load conditions". State q_6 means "maintenance".

[3]The reader who is interested in this aspect may consult Appendix C

control [1]. The set of reconfiguration events is denoted by Σ_{RE}, where $\Sigma_{RE} \cap \Sigma = \emptyset$.

Given a DES $\mathbf{G}$ with n modes, let $\mathbf{Mode}_i, \mathbf{Mode}_j$ be two modes of $\mathbf{G}$, where $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ must be deactivated and activated respectively in the course of reconfiguration. Then, the occurrence of reconfiguration event $\sigma_{i,j}$ deactivates $\mathbf{Mode}_i$ and activates $\mathbf{Mode}_j$.

In the unidirectional reconfiguration, a *reconfiguration specification* (RS) describes the reconfiguration requirements and specifies the source and target modes. For $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ of $\mathbf{G}$, with event alphabets Σ_i and Σ_j, assume that reconfiguration requires switching from $\mathbf{Mode}_i$ to $\mathbf{Mode}_j$, with the system $\mathbf{G}$ initially in $\mathbf{Mode}_i$. Bringing in a new corresponding switching event $\sigma_{i,j}$, the RS is defined as the DES

$$\mathbf{R} := (Q^R, \Sigma^R, \delta^R, q_o^R, Q_m^R)$$

where $Q^R := \{q_i, q_j\}$ and $Q^R \cap Q = \emptyset$; $\Sigma^R := \Sigma_i \cup \Sigma_j \dot{\cup} \{\sigma_{i,j}\}$ (note that $\Sigma_i \cup \Sigma_j$ is disjoint from $\{\sigma_{i,j}\}$); $q_o^R = q_i$; $Q_m^R = \{q_i, q_j\}$; and

$$\begin{aligned} \delta^R(q_i, \sigma) &:= \begin{cases} q_i & \text{if } \sigma \in \Sigma_i; \\ q_j & \text{if } \sigma = \sigma_{i,j}; \end{cases} \\ \delta^R(q_j, \sigma) &:= q_j \text{ if } \sigma \in \Sigma_j. \end{aligned} \tag{2.18}$$

$\mathbf{R}$ is depicted in Figure 2.4.

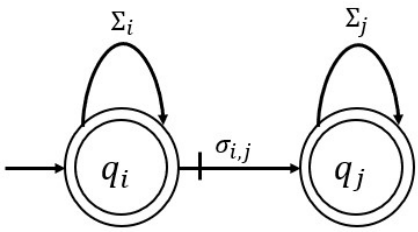

Figure 2.4: Unidirectional reconfiguration specification R

All the reconfiguration behaviors will then be incorporated in a *reconfiguration plant* (RP), which is the synchronous product of the plant and the reconfiguration specification. Given a plant DES $\mathbf{G}$ and a corresponding reconfiguration specification $\mathbf{R}$, the

reconfiguration plant is defined as a DES **RG**, where

$$\mathbf{RG} := \mathbf{G}||\mathbf{R} \tag{2.19}$$

In a reconfiguration plant, the one-way reconfiguration is represented by transitions labeled by the specific reconfiguration event that is defined in the reconfiguration specification.

For example, Consider a plant DES $\mathbf{G}_{toy}$ as shown in Figure 2.5. There are only two modes $\mathbf{Mode}_1$ and $\mathbf{Mode}_2$ in the system, which are distinguished by $\Sigma_1 = \{1, 3, 5\}$ and $\Sigma_2 = \{1, 7, 9\}$, respectively.

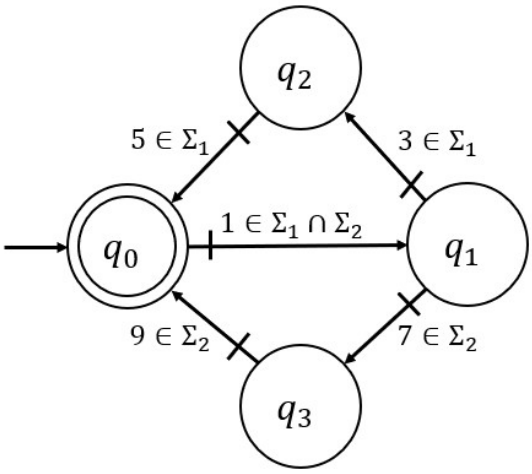

Figure 2.5: Plant DES $\mathbf{G}_{toy}$ of a toy example

To model the unidirectional reconfiguration behavior, it is natural to use one reconfiguration specification as follows.

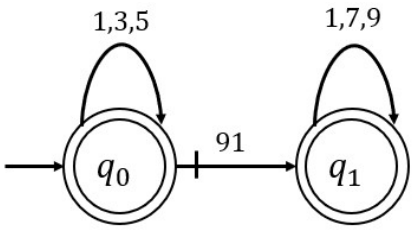

Figure 2.6: Reconfiguration specification $\mathbf{R}_{12}$ of a toy example

We then compute

$$\mathbf{RG}_{12} := \mathbf{Sync}(\mathbf{G}_{toy}, \mathbf{R}_{12})$$

The resulting reconfiguration plant $\mathbf{RG}_{12}$ generated by the DES computing software TCT is shown below.

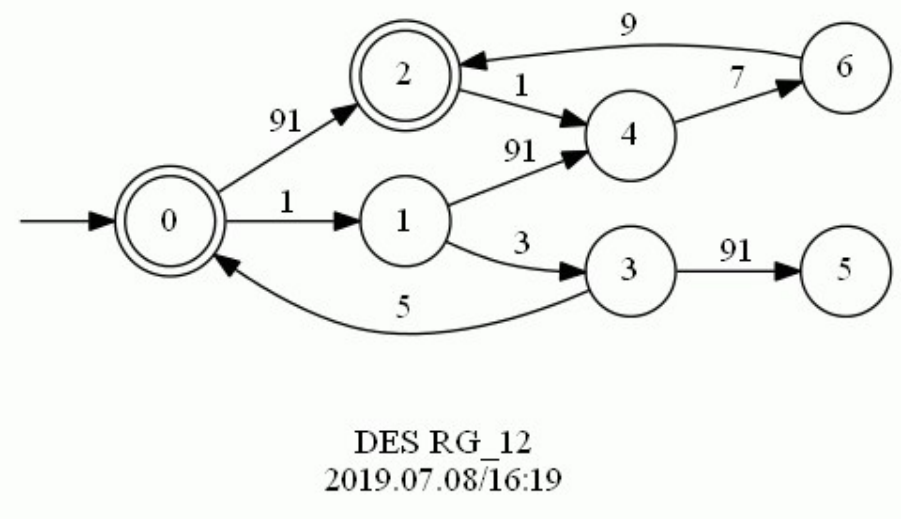

Figure 2.7: Reconfiguration plant $\mathbf{RG}_{12}$ corresponding to $\mathbf{R}_{12}$

However, in $\mathbf{RG}_{12}$, state 5 is blocking; to handle that we can simply compute:

$$\mathbf{ALLRG}_{12} := \mathbf{Allevents}(\mathbf{RG}_{12})$$
$$\mathbf{RGSUP}_{12} := \mathbf{Supcon}(\mathbf{RG}_{12}, \mathbf{ALLRG}_{12})$$

The resulting $\mathbf{RGSUP}_{12}$ is as follows.

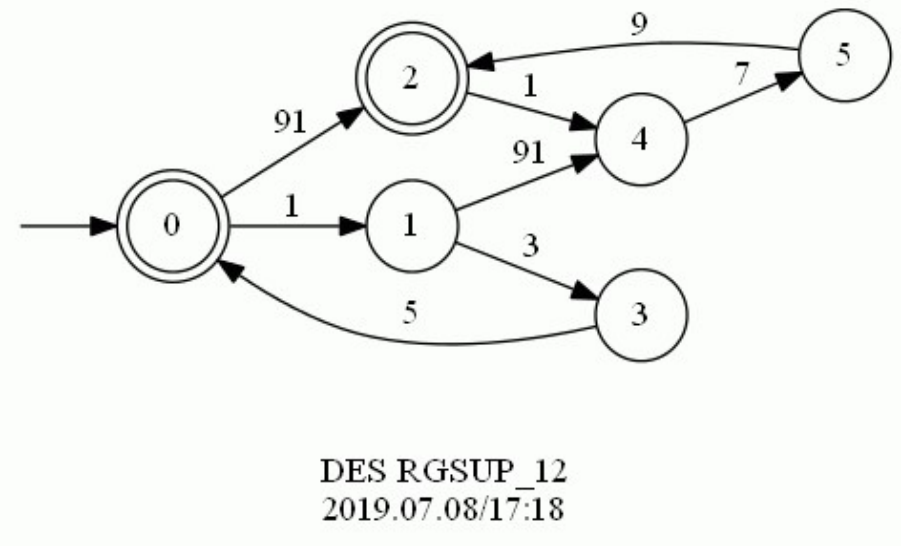

Figure 2.8: Reconfiguration plant $\mathbf{RGSUP}_{12}$ corresponding to $\mathbf{R}_{12}$

In the resulting reconfiguration plant $\mathbf{RGSUP}_{12}$, the initial mode of the system is distinguished by $\Sigma_1 = \{1, 3, 5\}$. At state 0 and 1 of $\mathbf{G}_{toy}$, reconfiguration can be defined since the two states belong to both modes. That's why reconfiguration event 91 is defined at state 0 and 1 of $\mathbf{RGSUP}_{12}$. $\mathbf{RGSUP}_{12}$ is nonblocking as expected.

Chapter 3

Bidirectional Reconfiguration of DES

3.1 Introduction

The approach to unidirectional reconfiguration of discrete-event systems has been proved effective to automatically and dynamically reconfigure a system from one mode to another mode [1]. However, to solve a bidirectional reconfiguration problem by this approach, a naive modeling in both directions could lead to blocking. In fact, the unidirectional reconfiguration can only deal with the one-way reconfiguration problem, namely it fails to realize the mechanism of bidirectional reconfiguration. Moreover, bidirectional reconfiguration forms the backbone for multiple reconfiguration that will be discussed in Chapter 4, where a system with more than two modes can be switched from any mode to any other mode. In a complex real system, there might be more than two modes, so realizing the bidirectional reconfiguration mechanism is of notable importance.

This chapter presents a procedure to dynamically and bidirectionally reconfigure discrete-event systems by supervisory control theory. To this end, the scope of the problem studied in this chapter will be fixed first. Then we explain why the unidirectional reconfiguration approach reviewed in Chapter 2 doesn't work. The next step is to analyze the system components to distinguish different states and events, followed by the procedure to construct a new reconfiguration specification. With this reconfiguration specification, the bidirectional reconfiguration of DES is always managed at a state which

is in both the source mode and the target mode.

This chapter is organized as follows. Section 3.2 selects problems on which we will focus in the rest of this report, demonstrates the inadequacy of a unidirectional reconfiguration approach for these problems, and formally defines the bidirectional reconfiguration problem. Section 3.3 elaborates on the construction of the new reconfiguration specification and its restrictions. Section 3.4 illustrates the proposed method with an example and draws conclusions.

3.2 Bidirectional Reconfiguration Problem

This section aims to select the reconfiguration problem such that the problem is meaningful in practice and valuable if solved, but not trivial to solve. In addition, cumbersome notations could also be avoided. Finally, this section will provide a formal problem definition along with reasonable assumptions.

3.2.1 Problem Selection

Reconfiguration problems are a class of problems with rich diversity, hence the solution approaches to them are highly problem-specific. The approach mentioned in this chapter is a relatively general approach to solve a number of reconfiguration problems in a bidirectional and dynamic fashion, but it is not general enough to solve all kinds of reconfiguration problems in the SCT framework.

Without loss of generality, systems with more than one mode can be roughly divided into two types: sequential-mode systems (SMS) and parallel-mode systems (PMS). In SMS, some modes cannot be activated until the system finishes operating in certain other modes. But in PMS, any mode can be activated at the beginning. The two types of reconfiguration problems along with their combination seem to cover most if not all reconfiguration problems.

Different modes in a SMS obey a logical operating order according to the structure, features and purposes of the system. The SMS is very common in daily life. For example, consider an unmanned aircraft as a system. The aircraft has seven modes for taxiing, take off, climbing, cruising, descending, approach, and landing. At the beginning, the

aircraft must operate in the taxiing mode, and the system cannot activate the cruising mode until it deactivates the climbing mode. Of course, when emergencies occur, the system is able to switch to the descending and the approach mode immediately, but they are all expected to be activated after the taxiing mode.

The following Figure 3.1 illustrates a sequential-mode system in the SCT framework. In this SMS, the $\mathbf{Mode}_1$ is the initial mode in which the system starts initially. The $\mathbf{Mode}_2$ cannot be activated unless the system resides at state q_1, and the $\mathbf{Mode}_3$ can be activated only when the system leaves the $\mathbf{Mode}_2$. The sequence of modes is $\mathbf{Mode}_1 \rightarrow \mathbf{Mode}_2 \rightarrow \mathbf{Mode}_3 \rightarrow \mathbf{Mode}_1 \rightarrow \ldots$. A noteworthy characteristic of SMS in the SCT framework is that the initial state of the DES is not in every mode of the system.

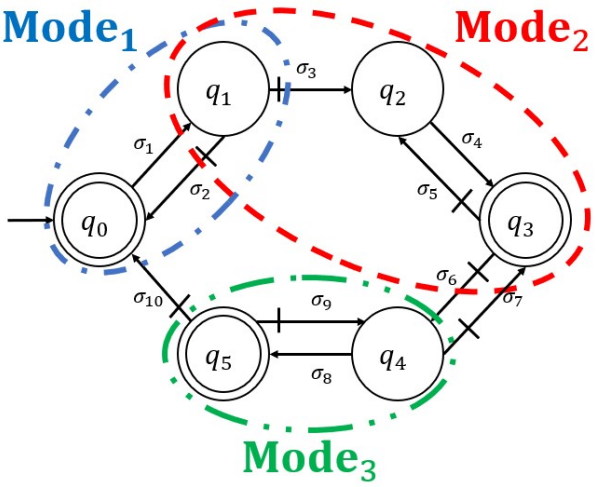

Figure 3.1: Sequential-mode system in the SCT framework

In a PMS, each mode can be activated initially. The PMS is also common in daily life. For example, consider a fitness treadmill as a system. The treadmill has four modes, for fat burning, cardio, cross-country and Fartlek training. The system can start in any mode according to the user's wish, namely there is no prerequisite for entering a particular mode. When the machine is running in one mode, the system can be reconfigured to any other mode according to the user's demand.

The following Figure 3.2 illustrates a parallel-mode system in the SCT framework. In this PMS, the initial mode in which the system starts can be any one of the three modes. There is no explicit sequence of modes in this PMS, namely the three modes are in parallel. In contrast to SMS, a conspicuous characteristic of PMS in the SCT

framework is that the initial state of the DES is always in all modes of the system, in order to allow the system to start in any mode.

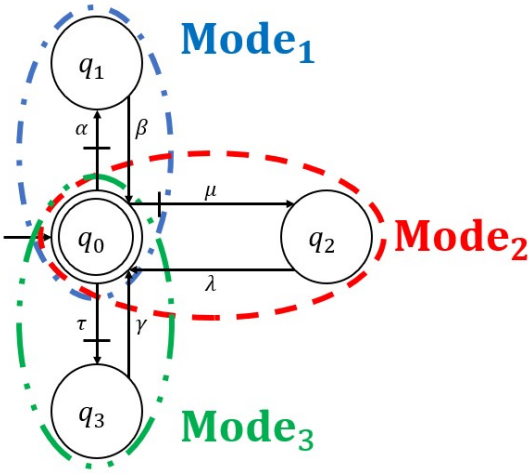

Figure 3.2: Parallel-mode system in the SCT framework

There are also many reconfiguration problems that are partly sequential-mode and partly parallel-mode. For example, consider a car as a system. The car has three modes, for low-speed running, high-speed running, and reverse running. The car system can start in low-speed running mode or the reverse running mode. But the high-speed running mode can be activated only after the system finishes operating in the low-speed mode. The following Figure 3.3 illustrates a partly sequential-mode and partly parallel-mode system.

In the example illustrated in Figure 3.3, $\mathbf{Mode}_1$, $\mathbf{Mode}_2$, and $\mathbf{Mode}_3$ are in parallel, while $\mathbf{Mode}_3$ and $\mathbf{Mode}_4$ are in sequence.

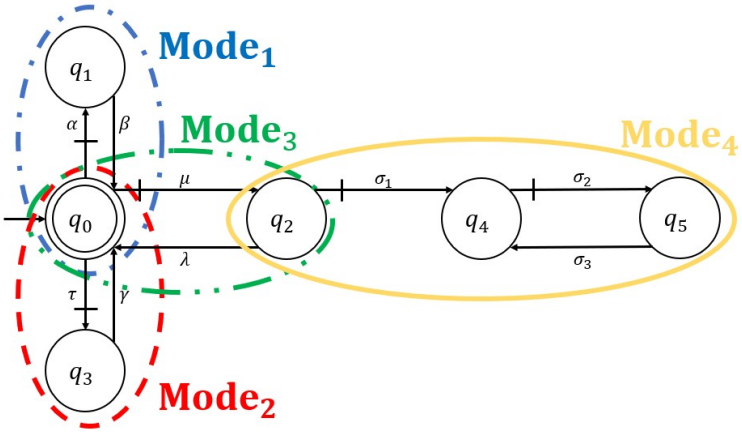

Figure 3.3: Partly sequential-mode and partly parallel-mode system in the SCT framework

In this report, we focus on parallel-mode systems rather than sequential-mode systems or their combination. There are three main reasons.

Firstly, reconfiguration in sequential-mode systems is often unidirectional, while this report studies bidirectional and multiple reconfiguration. In a SMS, the modes are set in sequence mainly because the order of modes is irreversible. Consider the aircraft example with seven modes of taxiing, take off, climbing, cruising, descending, approach, and landing. Imagine the taxiing mode is activated after the take off mode. Such a reversal might will be unfortunate.

Second, from another perspective, still consider the aircraft example. Since every time the system is reset, the system has to start in the taxiing mode, it is harmless to consider the seven modes as one mode, say "default mode". Then the reconfiguration and switching behaviors between previous modes can be seen as the dynamics in the new default mode.

Third, the sequential-mode mechanism can also be achieved in parallel-mode systems. In a PMS, since the system can start in any mode, then the system could also incorporate all mode sequences with the help of a suitable reconfiguration approach. In other words, a PMS contains more possibilities than a SMS in terms of the order of modes. Guided by some behavioral specifications, a PMS with reconfiguration mechanism embedded is able to imitate a SMS. For example, in the treadmill example, though the system is a PMS, the order of modes can still be "fat burning $\rightarrow$ cardio $\rightarrow$ cross-country $\rightarrow$ Fartlek training" according to the user's demand. On the contrary, the parallel-mode mechanism cannot be achieved in a SMS owing to the restrictive structure of SMS.

Fourth, working on reconfiguration problems in SMS is inconvenient to some extent. In a SMS, from one mode to another mode, there might not exist direct reconfiguration, namely the system has to enter other modes first, so some notations are required to distinguish them. In particular, consider the aircraft example. There doesn't exist direct reconfiguration from taxiing mode to cruising mode. In fact, any two modes in sequential-mode part of a system but are not adjacent don't support direct reconfiguration. On the other hand, since the system is not able to start in some modes in a SMS, we also need to distinguish them.

Therefore, for convenience of notation and the first three reasons, we confine atten-

tion to the parallel-mode systems and fix our bidirectional and multiple reconfiguration problems on them.[1]

Note that when the system is composed of several components, it is required that every component is a PMS, namely each component is able to start in any mode. Thus, for each mode whose alphabet has a non-empty intersection with the alphabet of a component, the component is able to start in this mode, i.e.

$$(\forall i = 1, ..., n)(\forall k = 1, ..., h)\ [\Sigma_i \cap \Sigma^k \neq \emptyset \Rightarrow (\exists \sigma \in \Sigma_i \cap \Sigma^k)\delta^k(q_o^k, \sigma)!] \tag{3.1}$$

However, the event alphabet of some mode may not be represented in some components, i.e.

$$(\exists j = 1, ..., n)(\exists k = 1, ..., h)\Sigma_j \cap \Sigma^k = \emptyset. \tag{3.2}$$

In this case, we can simply set $\mathbf{G}_j^k$ as a partial system with only the initial state of $\mathbf{G}^k$ and no events defined, i.e.

$$\mathbf{G}_j^k := (\{q_o^k\}, \emptyset, \delta_j^k, q_o^k, \{q_o^k\} \cap Q_m^k)$$

Here $\delta_j^k : \{q_o^k\} \times \emptyset \to \{q_o^k\}$ doesn't incorporate any transition. Then the component $\mathbf{G}^k$ can also start in $\mathbf{Mode}_j$, which is simply staying at the initial state. Thus, we can conclude that the initial state of a component belongs to every mode, i.e.

$$(\forall i = 1, ..., n)(\forall k = 1, ..., h)\ q_{o,i}^k = q_o^k \tag{3.3}$$

This statement serves as an important assumption as we work on parallel-mode systems. Here we also need another assumption that each mode is required to be reachable and coreachable separately in each component. For components $\mathbf{G}^k$ $(k = 1, ..., h)$ such that $\Sigma^k \cap \Sigma_i \neq \emptyset$, we just need to guarantee that $\mathbf{G}_i^k$ is reachable and coreachable separately, i.e.

$$\begin{aligned}&(\forall i \in 1, ..., n)(\forall k \in 1, ..., h)\ [\Sigma_i \cap \Sigma^k \neq \emptyset \Rightarrow \\ &(\forall q \in Q_i^k)(\exists q^{'} \in Q_{m,i}^k)(\exists s, s^{'} \in (\Sigma_i \cap \Sigma^k)^*)\delta_i^k(q_o^k, s) = q \wedge \delta_i^k(q, s^{'}) = q^{'}]\end{aligned} \tag{3.4}$$

However, for components $\mathbf{G}^r$ $(r = 1, ..., h)$ such that $\Sigma^r \cap \Sigma_i = \emptyset$, since $\mathbf{G}_i^r :=$

[1]The proposed bidirectional reconfiguration approaches in this chapter and the proposed multiple reconfiguration approach in Chapter 4 can also be applied to SMS, with slight modifications.

$(\{q_o^r\}, \emptyset, \delta_i^r, q_o^r, \{q_o^r\} \cap Q_m^r)$, in order to make sure the synchronous product $\mathbf{G}_i$ is coreachable by itself, it is required that q_o^r also be marked, i.e.

$$(\forall i \in 1, ..., n)(\forall r \in 1, ..., h)\ [\Sigma_i \cap \Sigma^r = \emptyset \Rightarrow q_o^r \in Q_m^r] \tag{3.5}$$

The assumption that every $\mathbf{G}_i^k$ is reachable and coreachable by itself is not strong. In a plant component, if there are some unreachable states of a mode, then the mode is not well-defined. In a real PMS, a mode is required to be able to start from the initial state of the system and lead to every state of this mode. Besides, if there are some uncoreachable states in a mode, then there are some blocking states in this mode, given that every state in the mode is reachable. Therefore, for the nonblocking and the proper definition issue, each mode is reachable and coreachable by itself in a component.

According to the above assumption, it is obvious that every mode is nonblocking in plant components where the mode is defined according to [1]. The reachability and coreachability are guaranteed for a mode in a plant component separately. In other words, these two properties are ensured locally in $\mathbf{G}_i^k$ for $\mathbf{Mode}_i$ in $\mathbf{G}^k$.

Therefore, according to the foregoing discussion, it is necessary to reformulate the definition of mode (configuration) in the context of PMS.

Definition 1. [Mode (Configuration)].Denote by $\mathbf{G} = (Q, \Sigma, \delta, q_o, Q_m)$ the plant DES ($\mathbf{G}$ is a PMS). In particular, $\mathbf{G} = \mathbf{G}^1||...||\mathbf{G}^h$ is the synchronous product of several DES components. A *mode (configuration)* $\mathbf{Mode}_i$ $(i = 1, ..., n)$ of $\mathbf{G}$ is represented by $\mathbf{G}_i$, the synchronous product of $\mathbf{G}_i^k$, i.e.

$$\mathbf{G}_i := \mathbf{G}_i^1||...||\mathbf{G}_i^h$$

where

$$(\forall k = 1, ..., h)\ \mathbf{G}_i^k := (Q_i^k, \Sigma_i^k, \delta_i^k, q_{o,i}^k, Q_{m,i}^k)$$

and

- Q_i^k is the subset of states for $\mathbf{Mode}_i$ in $\mathbf{G}^k$ and $q_o^k \in Q_i^k \subseteq Q^k$;
- $\Sigma_i^k = \Sigma_i \cap \Sigma^k \subseteq \Sigma^k$ is the alphabet of events for $\mathbf{Mode}_i$ in $\mathbf{G}^k$;
- $\delta_i^k : Q_i^k \times \Sigma_i^k \to Q_i^k$ is the partial transition function [2]

[2]For convenience, this one-step transition function could also be extended to string-wise, i.e. $\delta_i^k : Q_i^k \times \Sigma_i^{k*} \to Q_i^k$

$$\delta_i^k(q,\sigma) = \begin{cases} \delta^k(q,\sigma) & \text{if } q \in Q_i^k, \sigma \in \Sigma_i^k; \\ \text{not defined} & \text{otherwise.} \end{cases};$$

- $q_{o,i}^k = q_o^k \in Q_i^k$ is the initial state of $\mathbf{Mode}_i$ in $\mathbf{G}^k$;
- $Q_{m,i}^k \subseteq Q_m^k$ is the set of marked states for $\mathbf{Mode}_i$ in $\mathbf{G}^k$.

◇

Remark. For $\mathbf{Mode}_i$ and $\mathbf{Mode}_j, i \neq j$, it is possible that $\Sigma_i = \Sigma_j$. But in most cases, it is true that

$$(\Sigma_i - \Sigma_j \neq \emptyset) \text{ or } (\Sigma_j - \Sigma_i \neq \emptyset). \tag{3.6}$$

At any time, $\mathbf{G}$ can only operate in one of the different modes, say $\mathbf{Mode}_i$, when the partial systems $\mathbf{G}_i^k$ and $\mathbf{G}_j^k$ $(\forall j \neq i)$ are active and inactive in each plant component $\mathbf{G}^k$, respectively.

The event alphabets of distinct modes are not necessarily pairwise disjoint. Otherwise, the proposed approach would be overly restrictive.

A plant component $\mathbf{G}^1$ with four modes in the context of PMS is depicted in Figure 3.4. In $\mathbf{G}^1$, there is no event in Σ_4 defined, i.e. $\Sigma_4 \cap \Sigma^1 = \emptyset$, so $Q_4^1 = \{q_0\}$ and q_0 is marked.

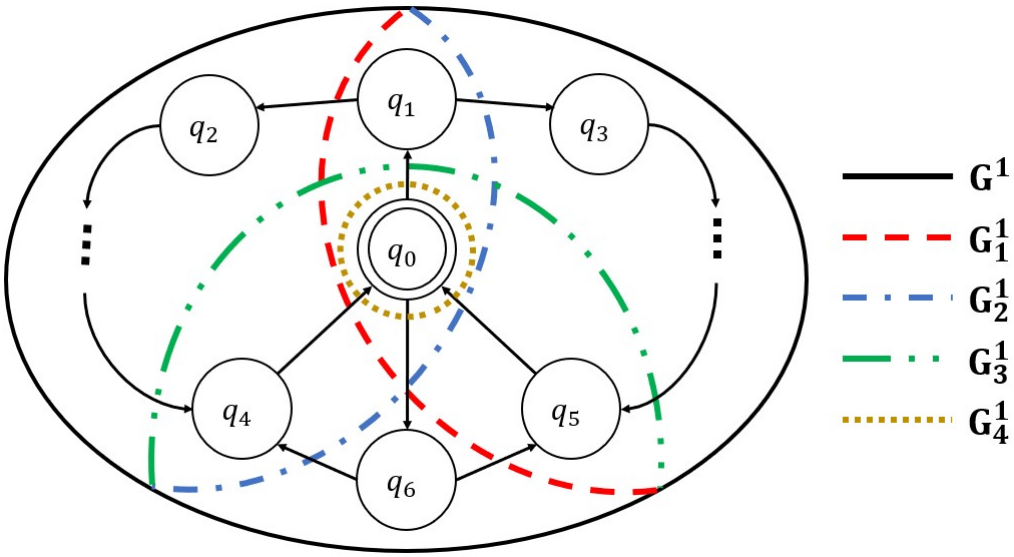

Figure 3.4: DES with three modes in the context of PMS

In most cases, the plant DES is the synchronous product of several components. As mentioned in Chapter 2, when there are multiple plant components, different modes are coupled in the synchronous product $\mathbf{G}$. Thus, it is hard to represent a mode as a partial system of $\mathbf{G}$. Similarly, the reachability and coreachability will not be discussed directly in $\mathbf{G}$ either.

In the rest of this report, the term "mode" refers to Definition 1.

With the definition of mode in this chapter, the informal definition of "bidirectional reconfiguration" problem is as follows.

Problem 1. [Bidirectional Reconfiguration of DES (informal)]. For a DES $\mathbf{G}$ with 2 modes, i.e. $\mathbf{Mode}_1, \mathbf{Mode}_2$, synthesize a reconfiguration plant represented by the DES $\mathbf{RG}$ such that in $\mathbf{RG}$:

(i) $\mathbf{RG}$ can start in either of the two modes;

(ii) From either mode, reconfiguration to the other mode is always possible when the plant is at a state shared by the two modes;

(iii) Reconfiguration can be operated back-and-forth without blocking;

(iv) Nonblocking is guaranteed for each of the two modes separately.

For requirement (i), since we study reconfiguration problems on parallel-mode systems, it is required that there be no explicit initial mode and the system can start in either of the two modes.

The requirement (ii) summarizes the legality and the safety of reconfiguration operations. Suppose that the source mode is $\mathbf{Mode}_1$ and the target mode is $\mathbf{Mode}_2$. If the reconfiguration from $\mathbf{Mode}_1$ to $\mathbf{Mode}_2$ occurs at a state that is not shared by both modes, then in some component $\mathbf{G}^k$, $\mathbf{G}_2^k$ may not be activated. Then this reconfiguration is unsuitable and should not be defined at that state.

The requirement (iii) emphasizes the essence of the bidirectional reconfiguration, which lies in the nonblocking back-and-forth reconfiguration behavior.

The requirement (iv) is to guarantee that the original system is not jeopardized by the embedded reconfiguration mechanism after the two modes are decoupled in $\mathbf{RG}$. Namely, each mode of the system needs to preserve nonblocking with the reconfiguration behavior incorporated.

Note that the problem definition with four requirements above is quite informal. In order to give a formal problem definition for bidirectional dynamic reconfiguration of DES, some more definitions are necessary. However, before supplying those definitions, the inadequacy of the unidirectional reconfiguration approach is analyzed first to provide helpful intuition.

Consider a plant DES $\mathbf{G}_{toy}$ as shown in Figure 3.5. There are only two modes $\mathbf{Mode}_1$ and $\mathbf{Mode}_2$ in the system, which are distinguished by $\Sigma_1 = \{1, 3, 5\}$ and $\Sigma_2 = \{1, 7, 9\}$, respectively.

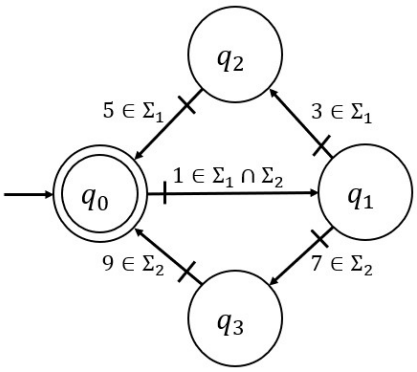

Figure 3.5: Plant DES $\mathbf{G}_{toy}$ of a toy example

To model the bidirectional reconfiguration behavior by using the unidirectional reconfiguration approach, it is natural to use two reconfiguration specifications, where each RS models one direction of reconfiguration. The two RS are as follows.

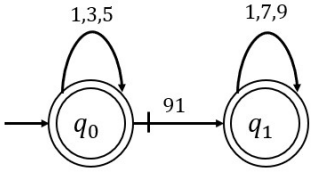

Figure 3.6: Reconfiguration specification $\mathbf{R}_{12}$ of a toy example

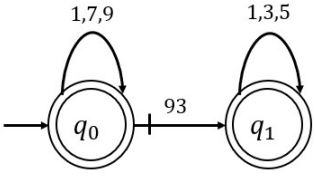

Figure 3.7: Reconfiguration specification $\mathbf{R}_{21}$ of a toy example

We then compute

$$\mathbf{RG}_{toy} := \mathbf{Sync}(\mathbf{G}_{toy}, \mathbf{R}_{12}, \mathbf{R}_{21}) \text{ Blocked events} = \text{None}$$

The resulting reconfiguration plant $\mathbf{RG}_{toy}$ generated by the DES computing software TCT is shown below.

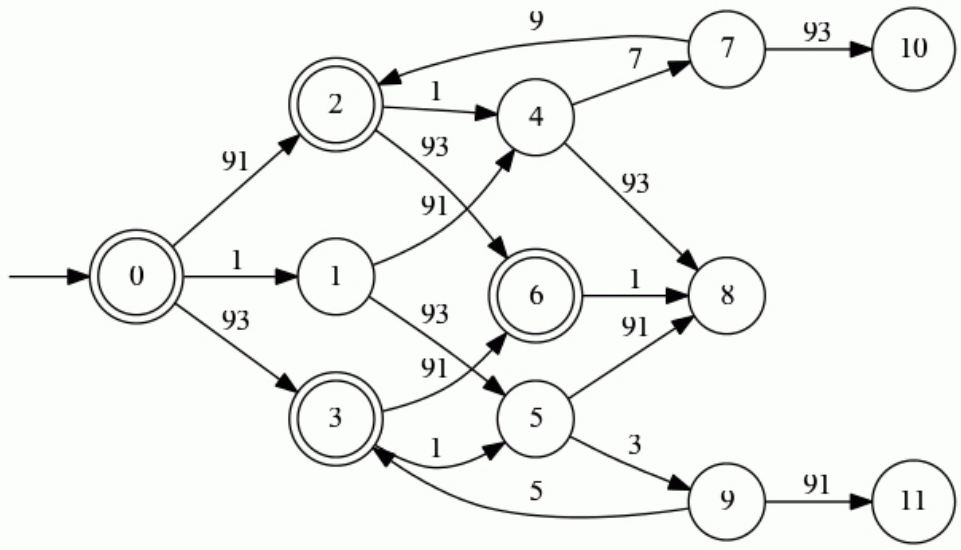

Figure 3.8: Reconfiguration plant $\mathbf{RG}_{toy}$ corresponding to $\mathbf{R}_{12}$ and $\mathbf{R}_{21}$

The reconfiguration plant $\mathbf{RG}_{toy}$ is blocking since the states 10 and 11 can never reach a marked state. The underlying reason is that the reconfiguration events are unsuitably defined at some states. For example, at state 9 of $\mathbf{RG}_{toy}$, the RE 91 is defined after the occurrences of event 1 and 3 in $\mathbf{G}_{toy}$, which lead the plant to state q_2. However, there is no event in $\mathbf{Mode}_2$ that is defined at state q_2 of $\mathbf{G}_{toy}$, so the occurrence of RE 91 at state 9 of $\mathbf{RG}_{toy}$ will result in blocking.

In fact, this blocking issue can be solved by the "**Supcon**" function in supervisory control theory, which by definition obtains the maximally permissive nonblocking controllable sublanguage of a string generator. Namely to remove blocking in the reconfiguration plant, we compute

$$\mathbf{ALLRG}_{toy} := \mathbf{Allevents}(\mathbf{RG}_{toy})$$
$$\mathbf{RGS}_{toy} := \mathbf{Supcon}(\mathbf{RG}_{toy}, \mathbf{ALLRG}_{toy})^{3}$$

[3]The "Allevents" function is used to generate the alphabets of a DES. The **Supcon**(**G**,**ALLG**) can eliminate blocking in **G**. The reader interested in this aspect may consult [1].

The resulting DES $\mathbf{RGS}_{toy}$ generated by the DES computing software TCT is shown below.

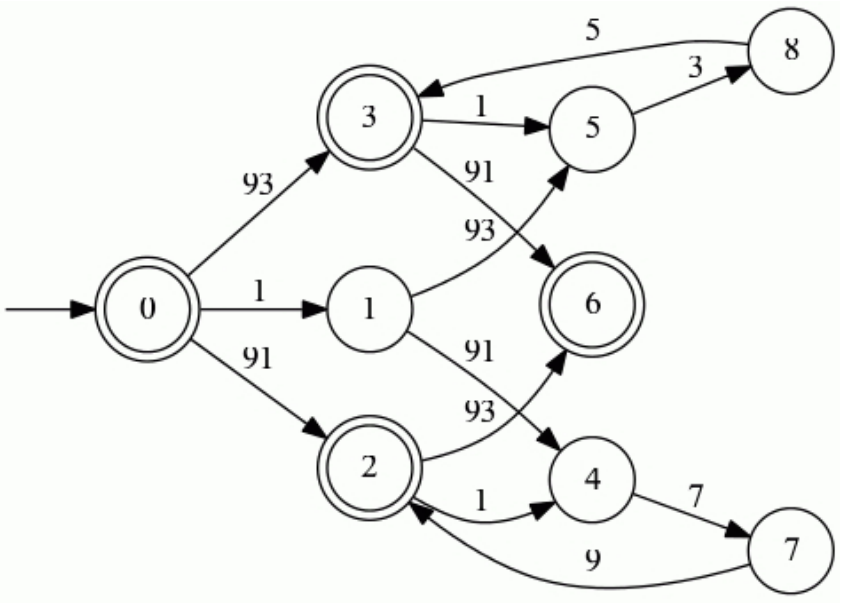

Figure 3.9: Nonblocking reconfiguration plant $\mathbf{RGS}_{toy}$

As we expect, there is no blocking state in $\mathbf{RGS}_{toy}$. However, there is another problem remaining. There is an ambiguity at the initial state 0 of $\mathbf{RGS}_{toy}$ since events 1, 91, 93 are all defined at state 0, which makes the physical interpretation of state 0 unclear. The reason for this ambiguity is that according to $\mathbf{R}_{12}$, the initial mode is $\mathbf{Mode}_1$, but according to $\mathbf{R}_{21}$, the initial mode is $\mathbf{Mode}_2$. Simply combining these two RS cannot enable the system to start in either mode.

Intuitively, there is another way to contain the two-way reconfiguration in a single RS, which just adds to $\mathbf{R}_{12}$ an additional backward transition labeled by another RE. The new RS is shown below.

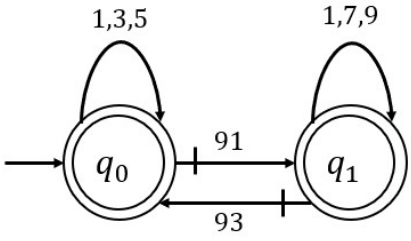

Figure 3.10: Alternative reconfiguration specification $\mathbf{R}_{new}$

In the new RS $\mathbf{R}_{new}$, we have to assign a mode to be the initial mode. Here the initial mode is $\mathbf{Mode}_1$ and the transitions labeled by RE 91 and 93 represent the bidirectional reconfiguration. We can then compute:

$$\mathbf{RG}_{new} := \mathbf{Sync}(\mathbf{G}_{toy}, \mathbf{R}_{new})$$

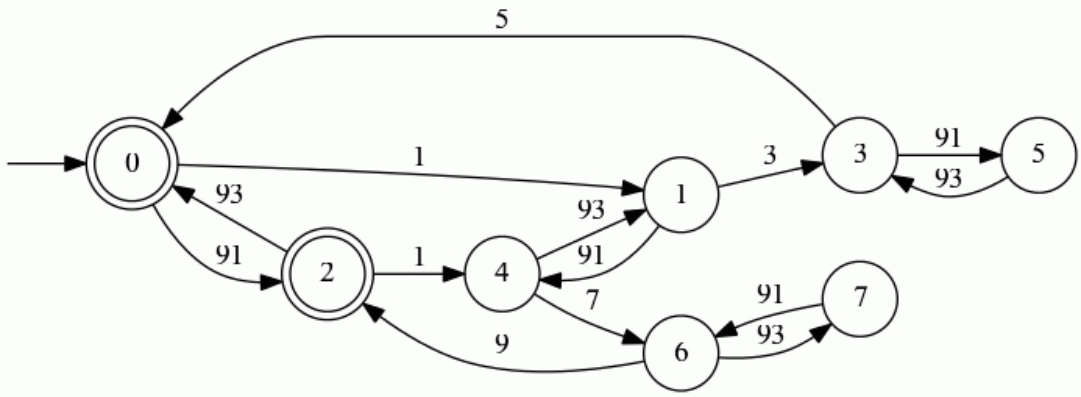

Figure 3.11: Reconfiguration plant $\mathbf{RG}_{new}$ corresponding to $\mathbf{R}_{new}$

Although there is no ambiguity or blocking issue in $\mathbf{RG}_{new}$, state 5 and state 7 with all transitions connected are redundant. For example, state 5 can only be reached from state 3 by a RE 91. However, state 3 is reached from state 0 by the string $< 1, 3 >$, which indicates that the plant is at state q_2, where no event in Σ_2 is defined. Intuitively, at state q_2 of $\mathbf{G}_{toy}$, the plant is at a state belonging to $\mathbf{Mode}_1$ but not $\mathbf{Mode}_2$, where the reconfiguration is not expected to be eligible, since a reconfiguration should occur at a state shared by both the source and the target mode, otherwise the reconfiguration would lead the system to a state where the target mode is not defined. Namely, the reconfiguration is unsuitable. Thus, a reconfiguration event cannot be defined at state 3 of $\mathbf{RG}_{new}$. Thus, state 5 is redundant. However, unlike in $\mathbf{RG}_{toy}$, state 5 in $\mathbf{RG}_{new}$ is not blocking. The reason is the bidirectional reconfiguration incorporated in $\mathbf{R}_{new}$, which eliminates the potential blocking issue. Hence, using "**Supcon**" function to further refine the RP is no longer effective.

Therefore, a new reconfiguration specification is required to avoid the ambiguity, the blocking issue, and the unsuitable reconfiguration.

3.2.2 Formal Problem Definition

The key to construct a new RS is to distinguish states belonging to different modes in the plant. Although different modes are distinguished by event alphabets, they are also relevant to states. In order to formally distinguish states of the plant according to different modes, the following definition of "aggregated mode predicate (AMP)" is needed.

Definition 2. [Aggregated Mode Predicate (AMP)]. An *aggregated mode predicate* of $\mathbf{Mode}_i$ (distinguished by Σ_i) on Q is denoted by $P_i^G : Q \to \{0,1\}$. P_i^G can be identified with the corresponding state subset $Q_i^G := \{q \in Q | P_i^G(q) = 1\} \subseteq Q$. For $\mathbf{Mode}_i$, the aggregated mode predicate P_i^G on Q is defined as follows

$$q \models P_i^G \text{ iff } \begin{cases} q = q_o; or \\ (\exists \sigma \in \Sigma_i)\ \delta(q,\sigma)!; or \\ (\exists \sigma \in \Sigma_i)(\exists q' \in Q)\ \delta(q',\sigma) = q. \end{cases} \tag{3.7}$$

Namely, consider any transition in $\mathbf{G}$ labeled by event $\sigma \in \Sigma_i$, then both the source state and the target state of the transition satisfy the aggregated mode predicate P_i^G on Q. In addition, the initial state satisfies all the aggregated mode predicates.

◇

Remark. Since in a PMS, the initial state is in every mode, it also satisfies all aggregated mode predicates, i.e. $q_o \models (P_1^G \wedge ... \wedge P_n^G)$. In practice, this statement is reasonable since the idle status of the system belongs to every mode, otherwise there would be some mode that can never start.

For any transition in $\mathbf{G}$ labeled by event $\sigma \in \Sigma_i$, both the source state and the target state of the transition satisfy the aggregated mode predicate P_i^G on Q.

The AMP can also be defined locally for different plant components, given a set of modes. For example, if the plant DES $\mathbf{G}$ is the synchronous product of h plant components $\mathbf{G}^1, ..., \mathbf{G}^h$, then there will be an AMP P_i^k $(k = 1, ..., h)$ on the state set Q^k

of $\mathbf{G}^k$ for the $\mathbf{Mode}_i$ defined as

$$q \models P_i^k \text{ iff } \begin{cases} q = q_o^k; or \\ (\exists \sigma \in \Sigma_i)\ \delta^k(q, \sigma)!; or \\ (\exists \sigma \in \Sigma_i)(\exists q^{'} \in Q^k)\ \delta^k(q^{'}, \sigma) = q. \end{cases} \tag{3.8}$$

or simply

$$q \models P_i^k \text{ iff } q \in Q_i^k \tag{3.9}$$

where Q_i^k is the state set of $\mathbf{G}_i^k$. This is true since we assume that every mode is both reachable and coreachable in each plant component.

Since the initial state of each plant component belongs to all modes, it is natural to get that the initial state of each plant component satisfies all AMP on the state set of the component for each mode. Moreover, it also makes the initial state of the synchronous product satisfies all AMP.

The example shown in Figure 3.12 illustrates the notion of AMP. In the example, there are three modes distinguished by $\Sigma_1 = \{\alpha, \beta, \mu\}, \Sigma_2 = \{\alpha, \lambda, \sigma\}, \Sigma_3 = \{\gamma, \tau\}$.

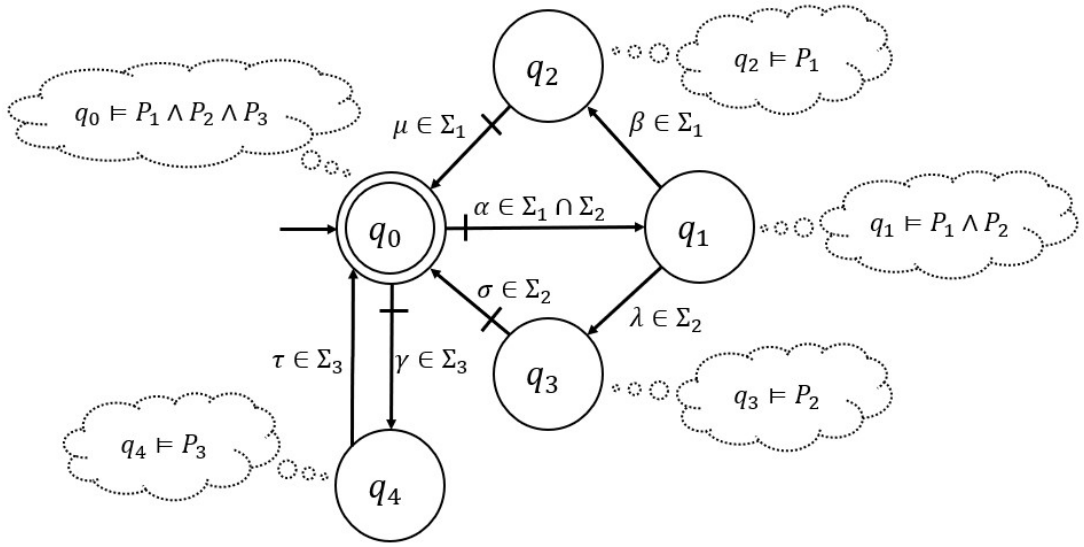

Figure 3.12: Toy example illustrating AMP

The mode predicates of the plant are "aggregated" in the sense that some modes share states, namely a state may belong to more than one mode. Then the state subsets corre-

sponding to mode predicates of distinct modes may have a non-empty intersection. Thus, the mode predicates for the plant and the plant components are defined as "aggregated" mode predicates.

Apart from the AMP, it is required that in the resulting reconfiguration plant generated by the proposed approach, the mode predicate is also defined. Different from the mode predicate of the plant, in the desired reconfiguration plant, the intersection of different mode subsets corresponding to the mode predicates must be empty, otherwise different modes cannot be completely decoupled. Since there might be some events in more than one alphabet of distinct modes, the definition of AMP is not suitable for the reconfiguration plant. Therefore, a definition of segregated mode predicate (SMP) is needed.

The formal definition of the segregated mode predicate is given in inductive fashion. In order to define the base case, a notion of mode initialization event (MIE) is required.

Recall that the initial state of the plant satisfies all AMP. In the reconfiguration plant, however, the initial state must not satisfy more than one mode predicate. Since the initial state of the plant has different physical interpretations for different modes, in the reconfiguration plant it is important to obtain n states corresponding to the initial state, where n is the number of modes[4]. However, every DES must have exactly one initial state. In order to satisfy the first requirement that the system can start in either mode, a new initial state is needed in the reconfiguration plant, otherwise there must be a mode serving as the initial mode.

Thus, the system would start from the new initial state. From the initial state, the system can reach the states corresponding to the initial state in the plant. The transitions from the initial state to those states are labeled by mode initialization events. The definitions of SMP and MIE are interdependent and are provided together as follows.

Definition 3. [Mode Initialization Event (MIE), Segregated Mode Predicate (SMP)]. Denote by **RG** the reconfiguration plant DES, and $\mathbf{Mode}_1, ..., \mathbf{Mode}_n$ the n modes distinguished by event alphabets $\Sigma_1, ..., \Sigma_n$. A *mode initialization event* of $\mathbf{Mode}_i$ is an event σ_i such that $\delta^{RG}(q_o^{RG}, \sigma_i)!$. Let the set of MIE be Σ_{MIE}; $\Sigma_{MIE} \cap (\Sigma_1 \cup ... \cup \Sigma_n \cup \Sigma_{RE}) = \emptyset$. A *segregated mode predicate* of $\mathbf{Mode}_i$ on Q^{RG} is denoted by $P_i^{RG} : Q^{RG} \to \{0, 1\}$.

[4] $n = 2$ in this chapter and $n \geq 2$ in the next chapter

For each configuration $\mathbf{Mode}_i$ $(i = 1, ..., n)$, the *segregated mode predicate* P_i^{RG} is defined inductively as follows.

$$\begin{aligned} &\text{(base case) } \delta^{RG}(q_o^{RG}, \sigma_i) \models P_i^{RG}; \\ &\text{(inductive case) } (\forall \sigma \in \Sigma_i)(\forall q \models P_i^{RG})\ [\delta^{RG}(q, \sigma)! \Rightarrow \delta^{RG}(q, \sigma) \models P_i^{RG}]. \end{aligned} \tag{3.10}$$

◇

Remark. It is obvious that for $\mathbf{Mode}_i$, the state $q \models P_i^{RG}$ if $\delta^{RG}(q_o^{RG}, \sigma_i s) = q$, where $\sigma_i \in \Sigma_{MIE}$ is the MIE for $\mathbf{Mode}_i$ and $s \in \Sigma_i^*$ is a string containing only events in $\mathbf{Mode}_i$.

The initial state of the reconfiguration plant doesn't satisfy any SMP. After the MIE occurs, the system "starts" physically.

If there are behavioral specifications, a supervisory controller can be computed from the reconfiguration plant under the restriction of behavioral specifications. The notion of SMP also applies to the supervisory controller. The formal proof will be provided later.

The example shown in Figure 3.13 illustrates the definition of MIE and SMP in a reconfiguration plant. In the example, there are three modes distinguished by $\Sigma_1 = \{\alpha, \beta, \mu\}, \Sigma_2 = \{\alpha, \lambda, \sigma\}, \Sigma_3 = \{\gamma, \tau\}$. The MIE $\sigma_1, \sigma_2, \sigma_3$ are for $\mathbf{Mode}_1, \mathbf{Mode}_2, \mathbf{Mode}_3$ respectively.

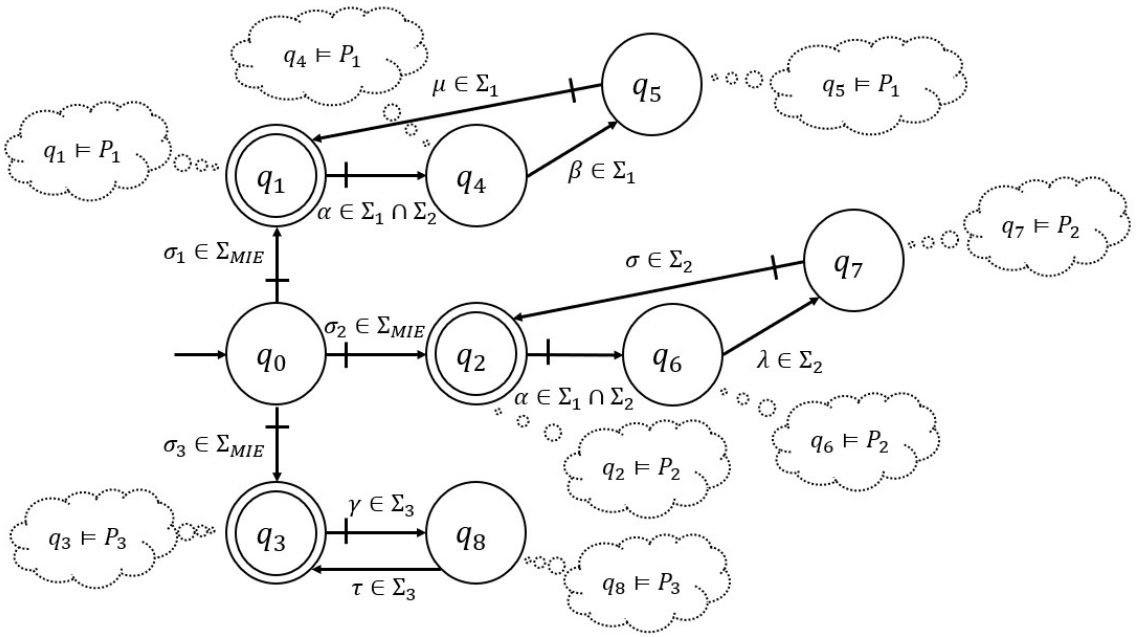

Figure 3.13: Toy example illustrating MIE and SMP

The MIE will be used in the new reconfiguration specification. In order to formally define the new reconfiguration specification, the notion of strictly public state (SPBS) needs to be introduced.

In a plant DES, a state (other than the initial state) might be reachable only by the events in one mode. More commonly, a state is reachable by events in more than one mode. Intuitively, if a state is reachable by events in all modes involved, it is a strictly public state.

In practice, the plant model might be the synchronous product of several plant components. It is useful if the users can construct the RS according to plant components since it is easier to analyze the modular parts. Moreover, if there is only one plant component, then that component is the plant itself. Thus, the definition of strictly public state can be generalized from the single plant to multiple plant components.

For each plant component DES, a strictly public state has to satisfy all AMP involved in the system. The formal definition of strictly public state is provided below.

Definition 4. [Strictly Public State (SPBS)]. Denote by $\mathbf{G} = \mathbf{G}^1||...||\mathbf{G}^h$ the plant DES formed by synchronization, and $\mathbf{Mode}_1, ..., \mathbf{Mode}_n$ the n modes. Also denote by $P_i^k : Q^k \rightarrow \{0, 1\}$ the AMP of $\mathbf{Mode}_i$ $(i = 1, ..., n)$ on Q^k $(k = 1, ..., h)$. For a state $q \in Q^k$, q is a *strictly public state* iff

$$q \models (P_1^k \wedge ... \wedge P_n^k). \tag{3.11}$$

Let Q_{SPBS}^k be the set of all SPBS in $\mathbf{G}^k$, and Q_{SPBS}^G be the set of all SPBS in $\mathbf{G}$.

◇

Remark. The initial state of each plant component $\mathbf{G}^k$ is a SPBS, since it satisfies all aggregated mode predicates, i.e.

$$q_o^k \models (P_1^k \wedge ... \wedge P_n^k). \tag{3.12}$$

The example shown in Figure 3.14 illustrates the notion of SPBS in the plant DES. In the example, there are three modes distinguished by $\Sigma_1 = \{\alpha, \beta, \mu\}, \Sigma_2 = \{\alpha, \lambda, \sigma\}, \Sigma_3 = \{\gamma, \tau\}$. Evidently, the state q_0 is the only SPBS.

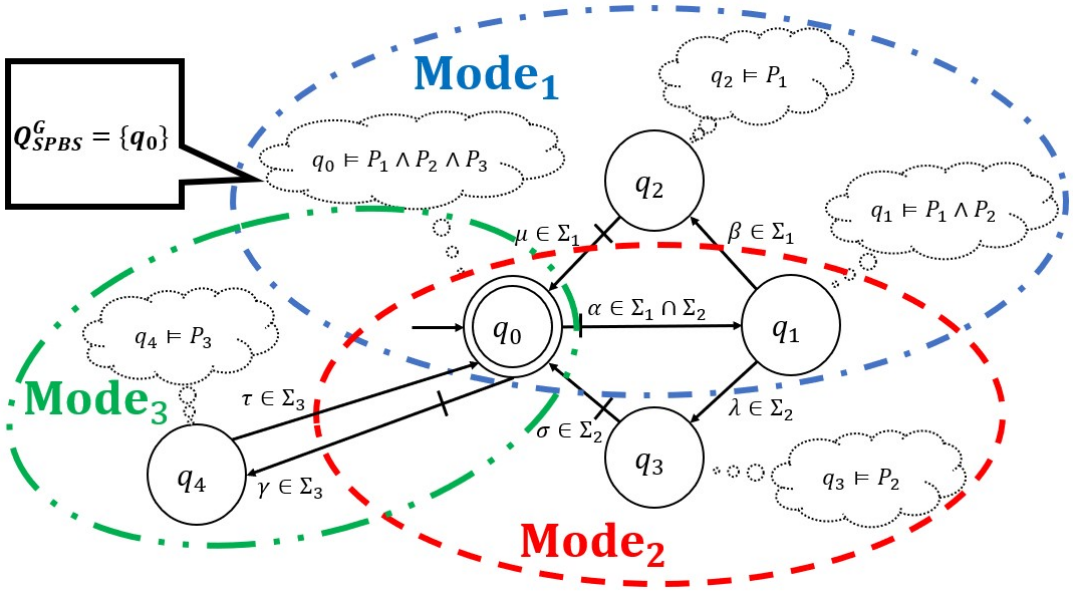

Figure 3.14: Toy example illustrating SPBS

The importance of SPBS is that the reconfiguration is always expected to be done at SPBS. Since a SPBS satisfies all AMP in the system, after the reconfiguration there would be no blocking issue in the result. Besides, the reconfiguration would be done at a state shared by both the source and the target mode. This requirement will be included in the formal definition of the bidirectional reconfiguration problem later.

With the definition of SPBS, it is natural to make the following definition of private state (PRS).

Definition 5. [Private State (PRS)]. For a state $q \in Q^k$ $(k = 1, ..., h)$, q is a *private state* iff it is not a SPBS, i.e.

$$q \not\models (P_1^k \wedge ... \wedge P_n^k) \tag{3.13}$$

Let Q^k_{PRS} be the set of all PRS in $\mathbf{G}_k$, and Q^G_{PRS} be the set of all PRS in $\mathbf{G}$.

◇

In the example illustrated by Figure 3.14, q_1, q_2, q_3 and q_4 are PRS, since they fail to satisfy all AMP.

With the definition of SPBS and PRS, it is not sufficient to construct the reconfiguration specification, since it's better to characterize an event or a string instead of a state in the SCT framework. Thus, the definitions for strictly exit event (SETE), strictly

entry event (SEYE), strictly external event (SELE) and strictly inner event (SINE) are given.

For a mode, consider the outer part as the set of PRS involved in the mode and the inner part as the set of SPBS involved in the mode. The inner part is often shared with other modes and is always used to communicate with other components of the system. Naturally, the inner parts are often the same for distinct modes.

A strictly exit event is used to label a transition from a SPBS to a PRS, which represents that the system goes to the outer part of a mode from the inner part. A strictly entry event is used to label a transition from a PRS to a SPBS, which represents that the system goes to the inner part of a mode from the outer part. A strictly external event is used to label a transition from a PRS to another PRS, which represents that the system is in the outer part. A strictly inner event is used to label a transition from a SPBS to another SPBS, which represents that the system is in the inner part.

Definition 6. [Strictly Exit Event (SETE)]. An event $\sigma \in \Sigma$ is a *strictly exit event* iff

$$(\exists k = 1, ..., h)(\exists q, q^{'} \in Q^k)\ [\delta^k(q, \sigma) = q^{'} \wedge q \in Q^k_{SPBS} \wedge q^{'} \in Q^k_{PRS}]. \tag{3.14}$$

Let Σ^k_{SETE} be the set of strictly exit events in $\mathbf{G}^k$, and Σ^G_{SETE} the set of strictly exit events in $\mathbf{G}$.

$\diamond$

Definition 7. [Strictly Entry Event (SEYE)]. An event $\sigma \in \Sigma$ is a *strictly entry event* iff

$$(\exists k = 1, ..., h)(\exists q, q^{'} \in Q^k)\ [\delta^k(q, \sigma) = q^{'} \wedge q \in Q^k_{PRS} \wedge q^{'} \in Q^k_{SPBS}]. \tag{3.15}$$

Let Σ^k_{SEYE} be the set of strictly entry events in $\mathbf{G}^k$, and Σ^G_{SEYE} the set of strictly entry events in $\mathbf{G}$.

$\diamond$

Definition 8. [Strictly External Event (SELE)]. An event $\sigma \in \Sigma$ is a *strictly external event* iff

$$(\exists k = 1, ..., h)(\exists q, q^{'} \in Q^k)\ [\delta^k(q, \sigma) = q^{'} \wedge q \in Q^k_{PRS} \wedge q^{'} \in Q^k_{PRS}]. \tag{3.16}$$

Let Σ^k_{SELE} be the set of strictly external events in $\mathbf{G}^k$, and Σ^G_{SELE} the set of strictly external events in $\mathbf{G}$.

$\diamond$

Definition 9. [Strictly Inner Event (SINE)]. An event $\sigma \in \Sigma$ is a *strictly inner event* iff

$$(\exists k = 1, ..., h)(\exists q, q^{'} \in Q^k)\ [\delta^k(q, \sigma) = q^{'} \wedge q \in Q^k_{SPBS} \wedge q^{'} \in Q^k_{SPBS}]. \tag{3.17}$$

Let Σ^k_{SINE} be the set of strictly inner events in $\mathbf{G}^k$, and Σ^G_{SINE} the set of strictly inner events in $\mathbf{G}$.

$\diamond$

Remark. Evidently, $\Sigma = \Sigma^G_{SETE} \cup \Sigma^G_{SEYE} \cup \Sigma^G_{SELE} \cup \Sigma^G_{SINE}$, and $(\forall \mathbf{G}^k)\ \Sigma^k = \Sigma^k_{SETE} \cup \Sigma^k_{SEYE} \cup \Sigma^k_{SELE} \cup \Sigma^k_{SINE}$

The following Figure 3.15 demonstrates the definition of SETE, SEYE, SELE and SINE.

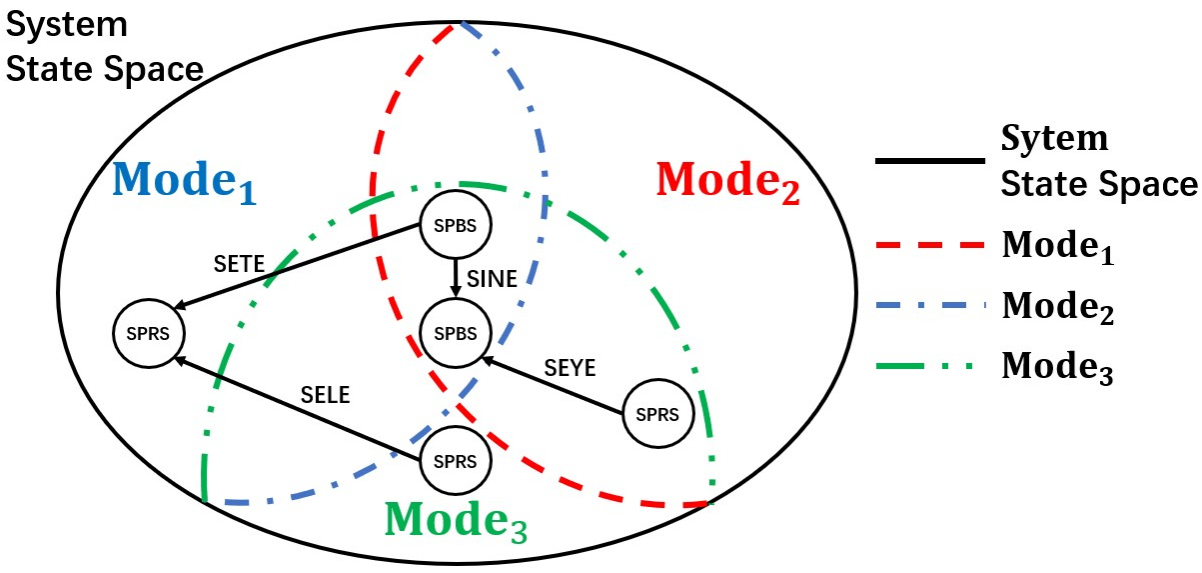

Figure 3.15: Toy example illustrating SETE, SEYE, SELE and SINE

In the later procedure of constructing a reconfiguration specification, two assumptions on strictly exit events and strictly entry events are required. The first states that the strictly exit events for different plant components are distinct, i.e.

$$(\forall k, r = 1, ..., h, k \neq r)\ \Sigma^k_{SETE} \cap \Sigma^r_{SETE} = \emptyset. \tag{3.18}$$

Similarly, the second states that the strictly entry events for different plant components are distinct, i.e.

$$(\forall k, r = 1, ..., h, k \neq r)\ \Sigma^k_{SEYE} \cap \Sigma^r_{SEYE} = \emptyset. \tag{3.19}$$

The objective of the two assumptions is to avoid overlapping physical interpretations.

In practice, neither of the two assumptions is strong. When the assumptions don't hold, there is some strictly exit (entry) event which is shared by at least two different components. Namely, the transitions labeled by this event in these components have to be synchronized to occur. In the SCT framework of DES, though there is no restriction of using the same event in more than one plant component, distinct components usually don't have shared events. On the one hand, components don't share events to avoid confusion of physical meanings, since they could be replaced by a single component otherwise. On the other hand, the objective of using shared events is to represent the relation of different components, which can also be realized by behavioral specifications.

In case there is some shared event, a relabeling procedure could be applied. For example, consider an event σ in the event alphabets of at least two plant components. This case reflects an unsuitable labeling of events since it may result in confusion of event meanings. Then relabeling can be done to distinguish them. Therefore, a new assumption is required that

$$(\forall a, b \in \{SETE, SEYE, SELE, SINE\}, a \neq b)\ \Sigma^G_a \cap \Sigma^G_b = \emptyset. \tag{3.20}$$

In summary, a necessary relabeling is needed when the original labelling is unsuitable. It is often done manually, but the procedure is not complicated.

With all the definitions above, the formal definition of bidirectional reconfiguration is provided below. In addition to the four requirements mentioned in Problem 1, one extra requirement needs to be formalized. It states that every state except for the initial state in the reconfiguration specification is required to satisfy exactly one SMP.

Problem 2. [Bidirectional Reconfiguration of DES (Formal)]. For a DES **G** with 2 modes, i.e. $\mathbf{Mode}_1$ and $\mathbf{Mode}_2$ that are distinguished by alphabets Σ_1 and Σ_2, given that $\Sigma_{MIE} = \{\sigma_1, \sigma_2\}$ and $\Sigma_{RE} = \{\sigma_{1,2}, \sigma_{2,1}\}$, synthesize a reconfiguration plant $\mathbf{RG} = (Q^{RG}, \Sigma^{RG}, \delta^{RG}, q^{RG}_o, Q^{RG}_m)$ such that in **RG**:

(i) **RG** can start in either of the two modes via a transition labeled by a mode initialization event, i.e.

$$(\forall \sigma_i = \sigma_1, \sigma_2)(\exists q \in Q^{RG})\ \delta^{RG}(q_o^{RG}, \sigma_i) = q; \tag{3.21}$$

(ii) (a) Reconfiguration can be done only when all plant components are at strictly public states, i.e.

$$\begin{aligned}&(\forall \sigma_{i,j} = \sigma_{1,2}, \sigma_{2,1})(\forall s \in L(\mathbf{RG}))\\ &[\delta^{RG}(q_o^{RG}, s\sigma_{i,j})! \Rightarrow (\forall k = 1, ..., h)\delta^{k}(q_o^{r}, P_{G^k}(s)) \in Q_{SPBS}^{k}];\end{aligned} \tag{3.22}$$

In this case $P_{G^k} : \Sigma^{RG*} \rightarrow \Sigma^{k*}$ is a natural projection [1].

(b) Each reconfiguration event appears in **RG**, i.e.

$$(\forall \sigma_{i,j} = \sigma_{1,2}, \sigma_{2,1})(\exists q_s, q_d \in Q^{RG}, q_s \neq q_d)\ \delta^{RG}(q_s, \sigma_{i,j}) = q_d; \tag{3.23}$$

(c) From one mode, reconfiguration to the other mode is possible only when the source state is in the source mode and the target state is in the target mode, i.e.

$$\begin{aligned}&(\forall \sigma_{i,j} = \sigma_{1,2}, \sigma_{2,1})(\forall q, q^{'} \in Q^{RG}, q \neq q^{'})\\ &[\delta^{RG}(q, \sigma_{i.j}) = q^{'} \Rightarrow (q \models P_i^{RG} \wedge q^{'} \models P_j^{RG})];\end{aligned} \tag{3.24}$$

(iii) Reconfiguration can be operated back-and-forth without blocking, i.e.

$$(\forall \sigma_{i,j}, \sigma_{j,i} = \sigma_{1,2}, \sigma_{2,1})\ [\delta^{RG}(q, \sigma_{i,j})! \Rightarrow (\exists s \in L(RG))\delta^{RG}(q, \sigma_{i,j}s\sigma_{j,i})!]; \tag{3.25}$$

(iv) Every reachable state of $\mathbf{Mode}_1$ or $\mathbf{Mode}_2$ in **RG** is also coreachable separately, which means that each mode is nonblocking in **RG**, i.e.

$$\begin{aligned}&(\forall i = 1, 2)(\forall q \models P_i^{RG})(\exists \sigma_i \in \Sigma_{MIE})(\exists s \in \Sigma_i^{*})\\ &[\delta^{RG}(q_o^{RG}, \sigma_i s) = q \Rightarrow (\exists q_m \in Q_m^{RG})(\exists s^{'} \in \Sigma_i^{*})(\delta^{RG}(q, s^{'}) = q_m \wedge q_m \models P_i^{RG})];\end{aligned} \tag{3.26}$$

(v) Every state except for the initial state in the **RG** satisfies exactly one SMP, i.e.

$$(\forall q \in Q^{RG}, q \neq q_o^{RG})\ [(q \models P_1^{RG} \wedge q \not\models P_2^{RG}) \vee (q \models P_2^{RG} \wedge q \not\models P_1^{RG})]. \quad (3.27)$$

$\diamond$

Remark. The requirement (v) implies that for the state sets corresponding to the SMP $(\forall i = 1, 2)\ Q_i^{RG} := \{q \in Q^{RG} | P_i^{RG}(q) = 1\} \subseteq Q^{RG}$, it is true that $Q^{RG} = Q_1^{RG} \dot{\cup} Q_2^{RG} \dot{\cup} \{q_o^{RG}\}$.

In fact, the requirement (v) is vital for the requirement (ii). If there is a state satisfying more than one SMP in the reconfiguration plant, then the requirement (ii) is not met, since the state is required to be divided into two states and there should be a reconfiguration defined between them. However, the formal definition of the requirement (ii) cannot detect whether there is some state satisfying more than one SMP, hence cannot tell the absence of reconfiguration at a state which satisfies only one SMP. If the requirement (v) is not satisfied, then requirement (ii) is not met, either.

The reconfiguration task can be solved by constructing a reconfiguration specification (RS). The reconfiguration specification can further be synchronized with the plant DES to generate the reconfiguration plant (RP). However, in order to automatically generate a structured RS without ambiguity, some assumptions mentioned above are required.

The following definition summarizes all assumptions needed in the proposed approach. Not all of them may be necessary in solving practical problems, but they are useful in establishing the theory, especially in the context of parallel-mode systems.

Definition 10. [Assumptions]. For a bidirectional reconfiguration problem, there are six assumptions.

(i) The initial state of each plant component is in both modes of the system, i.e.

$$(\forall i = 1, 2)(\forall k = 1, ..., h)\ q_{o,i}^k = q_o^k \quad (3.28)$$

(ii) Each of the two modes in each component DES is both reachable and coreachable

by itself, i.e.

$$\begin{aligned}&(\forall i \in 1,2)(\forall k \in 1,...,h)\ [\Sigma_i \cap \Sigma^k \neq \emptyset \Rightarrow \\ &(\forall q \in Q_i^k)(\exists q^{'} \in Q_{m,i}^k)(\exists s, s^{'} \in (\Sigma_i \cap \Sigma^k)^*)\delta_i^k(q_o^k, s) = q \wedge \delta_i^k(q, s^{'}) = q^{'}]\end{aligned} \tag{3.29}$$

and

$$(\forall i \in 1,2)(\forall r \in 1,...,h)\ [\Sigma_i \cap \Sigma^r = \emptyset \Rightarrow q_o^r \in Q_m^r] \tag{3.30}$$

(iii) The strictly exit events for different plant components are distinct, i.e.

$$(\forall k, r = 1,...,h, k \neq r)\ \Sigma_{SETE}^k \cap \Sigma_{SETE}^r = \emptyset \tag{3.31}$$

(iv) The strictly entry events for different plant components are distinct, i.e.

$$(\forall k, r = 1,...,h, k \neq r)\ \Sigma_{SEYE}^k \cap \Sigma_{SEYE}^r = \emptyset \tag{3.32}$$

(v) An event cannot serve as different types of events in different components, i.e.

$$(\forall a, b \in \{SETE, SEYE, SELE, SINE\}, a \neq b)\ \Sigma_a^G \cap \Sigma_b^G = \emptyset \tag{3.33}$$

(vi) Each event in the plant must belong to either (or both) of the two modes, i.e.

$$\Sigma_1 \cup \Sigma_2 = \Sigma \tag{3.34}$$

◇

Remark. The assumption (vi) is to make sure that there is no undefined behavior in the plant.

3.3 Bidirectional Reconfiguration Approach

In the last section we studied the bidirectional reconfiguration problems on parallel-mode systems. The problem is formally defined, along with six important assumptions. This section aims to solve Problem 2 within the scope of the six assumptions.

3.3.1 Bidirectional Reconfiguration Specification

A new structured bidirectional reconfiguration specification (BRS) can be used to represent the bidirectional reconfiguration behavior and solve Problem 2. The BRS includes all details for the bidirectional reconfiguration task, such as event alphabets of the two modes, RE, and MIE.

Definition 11. [Bidirectional Reconfiguration Specification (BRS)]. Denote by $\mathbf{G}$ the plant DES formed by synchronization. $\mathbf{Mode}_1$ and $\mathbf{Mode}_2$ are 2 different modes of $\mathbf{G}$ distinguished by event alphabets Σ_1 and Σ_2. The MIE and RE are $\Sigma_{MIE} = \{\sigma_1, \sigma_2\}$ and $\Sigma_{RE} = \{\sigma_{1,2}, \sigma_{2,1}\}$. The *bidirectional reconfiguration specification* is defined as the DES

$$\mathbf{R} := (Q^R, \Sigma^R, \delta^R, q_o^R, Q_m^R)$$

where

- $Q^R := \{q_o^R\} \dot{\cup} Q_M^R \dot{\cup} Q_E^R$, in which $Q_M^R := \{q_1^0, q_2^0\}$ and for each of the two modes $\mathbf{Mode}_i$ $(i = 1, 2)$, if $\Sigma_{SETE}^G \cap \Sigma_i \neq \emptyset$, then there will be b states $q_i^1, ..., q_i^b \in Q_E^i \subseteq Q_E^R$, where b is the number of plant components whose event alphabets include at least one SETE, i.e. $b = |\{r | \Sigma_{SETE}^r \neq \emptyset\}|$; $Q_E^1 \dot{\cup} Q_E^2 = Q_E^R$ and $Q_E^1 \cap Q_E^2 = \emptyset$;
- $\Sigma^R := \Sigma \dot{\cup} \Sigma_{RE} \dot{\cup} \Sigma_{MIE}$;
- $\delta^R(q_o^R, \sigma) := q_i^0 \quad$ if $\sigma = \sigma_i \in \Sigma_{MIE}$;

  $\delta^R(q_i^0, \sigma) := q_i^0 \quad$ if $\sigma = \sigma_{i,j} \in \Sigma_{RE}$;

  $$\delta^R(q_i^r, \sigma) := \begin{cases} q_i^r & \text{if } \sigma \in \Sigma_i \cap \Sigma_{SINE}^G, 0 \leq r \leq b; \\ q_i^{r+1} & \text{if } \sigma \in \Sigma_i \cap \Sigma_{SETE}^G, 0 \leq r < b; \\ q_i^r & \text{if } \sigma \in \Sigma_i \cap \Sigma_{SELE}^G, 0 < r \leq b; \\ q_i^{r-1} & \text{if } \sigma \in \Sigma_i \cap \Sigma_{SEYE}^G, 0 < r \leq b; \end{cases}$$
- q_o^R is the initial state of $\mathbf{R}$;
- $Q_m^R = Q^R - \{q_o^R\}$.

$\diamond$

Remark. The Q_M^R is the state set in which the two states have RE defined. At each state in Q_M^R, there are self-loops of transitions labeled by events in Σ_{SINE}^G, namely in inner part of the mode. It means that the reconfiguration can be done in the inner part of the system. It is reasonable since the inner part has all the strictly public states, where reconfiguration is expected to be defined.

Strictly inner events in $\mathbf{Mode}_i$ are also self-looped at q_i^r, where $1 \leq r \leq b$. After a number of SETE occur in some plant components, other components may still at SPBS, so all SINE are required to be enabled at the state q_i^r, in order to avoid unnecessary restriction.

At any state in Q_E^R, there are self-loops of transitions labeled by events in Σ_{SELE}^G, since the strictly external events are possible to occur after the system component exit the inner part of the mode.

The transitions regarding Q_E^R, Σ_{SETE}^G and Σ_{SEYE}^G simply mean that reconfiguration occurs only when every plant component of the system has returned to the inner part.

The following Figure 3.16 illustrates a typical BRS.

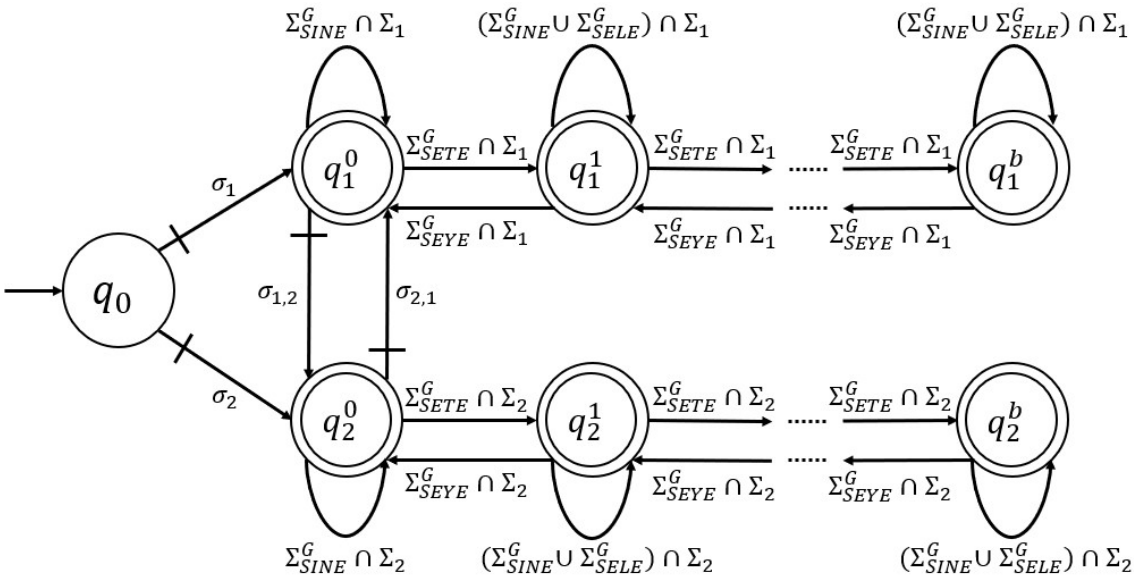

Figure 3.16: Typical BRS

Given the assumptions (iii), (iv) and (v) stated in Definition 10, a SETE (or SEYE) in one component is distinct from any SETE (or SEYE) in other components. Thus, a SEYE in a plant component corresponds to at least one SETE in the same component. However,

the reverse direction is not always true. Consider the example in Figure 3.17, none of SETE in this example has a corresponding SEYE, but the system still satisfies all the six assumptions. In this case, if one of the events in Σ^G_{SETE} occurs, the reconfiguration from $\mathbf{Mode}_1$ to $\mathbf{Mode}_2$ or from $\mathbf{Mode}_2$ to $\mathbf{Mode}_1$ can never occur, so the reconfiguration occurs only before the SETE occurs. This consequence is due to the structure of the system itself but not the proposed approach.

In a plant component, if one SETE corresponds to more than one, for example, two SEYE, the two SEYE cannot both occur in the component unless the SETE occurs twice, since the plant component cannot enter the inner part twice with only one exit. Similarly, the reverse direction is also true. As a result, if in the system there is only one SETE (SEYE) and two SEYE (SETE), there will be only one extra state (q^1_i) needed in the Q^i_E $(i = 1, 2)$. Therefore, for each of the plant components whose event alphabets include at least one SETE, one extra state is needed for a mode in the BRS. Namely, $|Q^1_E| = |Q^2_E| = |\{r|\Sigma^r_{SETE} \neq \emptyset\}| = b$ [5].

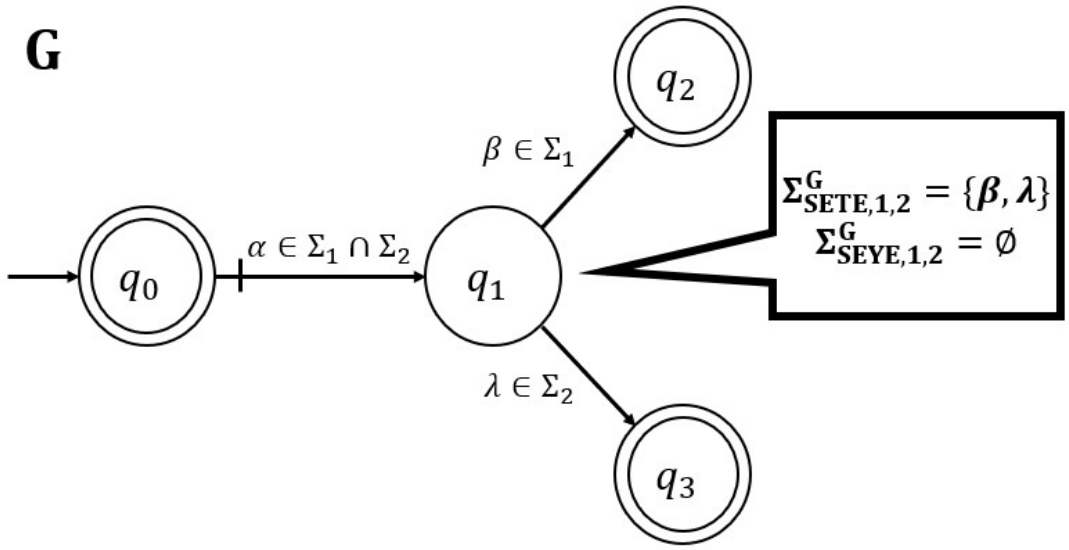

Figure 3.17: Toy example with no SEYE

It turns out that the reconfiguration specification constructed according to Definition 11 solves Problem 2 through the synchronization with the plant DES $\mathbf{G}$, i.e.,

$$\mathbf{RG} = \mathbf{Sync}(\mathbf{G}, \mathbf{R})$$

[5] When there are more than two modes, say n, then $|Q^1_E| = ... = |Q^n_E| = |\{r|\Sigma^r_{SETE} \neq \emptyset\}| = b$

The resulting synchronous product is the reconfiguration plant that incorporates the bidirectional dynamic reconfiguration mechanism.

The main procedure of the bidirectional reconfiguration approach is illustrated in Figure 3.18.

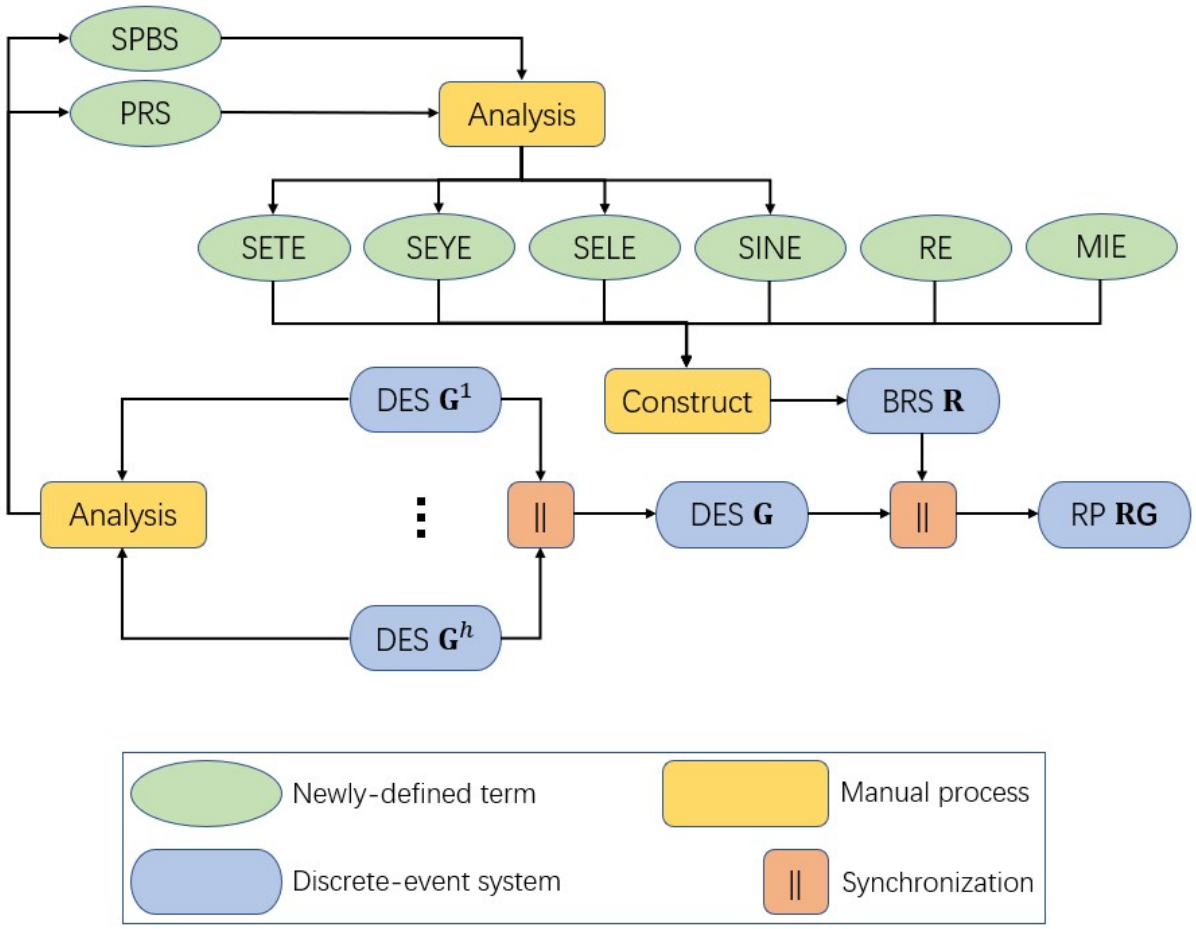

Figure 3.18: Bidirectional reconfiguration of discrete-event systems

Informally, we discuss why the reconfiguration plant constructed from the proposed BRS satisfies the five requirements in Problem 2.

(i) The RP can start in either of the two modes.

With the mode initialization events, the BRS can start at any mode. Since no MIE is in Σ and events in Σ are defined at other states in BRS, the RP can start at any mode through a transition labeled by a MIE from the initial state.

(ii) The reconfiguration can be done when all plant components are at strictly public

states. From one mode, the reconfiguration to another mode is always possible when the source state is in the source mode and the target state is in the target mode.

Note that in the proposed BRS, RE is not allowed to occur after the occurrence of SETE, unless the system returns to the inner part of the two modes via a SEYE. Intuitively, a reconfiguration is always expected to occur at a strictly public state that is shared by the two modes. Otherwise, after the reconfiguration, some events of the target mode would be blocked.

(iii) The reconfiguration can be operated back-and-forth without blocking.

It is true. According to BRS, the backward reconfiguration can even be operated at the target state of the forward reconfiguration after a few transitions in the target mode.

(iv) Nonblocking is guaranteed for either of the two modes separately.

In the plant DES, nonblocking is guaranteed for either of the two modes separately. Since the BRS includes all events in the plant DES and results in no new blocking issues, it is true that nonblocking is guaranteed for either mode separately in the RP.

(v) Every state except for the initial state in the reconfiguration specification satisfies exactly one SMP.

In the reconfiguration plant, events in different modes are defined at different states. In the BRS, at the states where outer events are defined, transitions labeled by RE are eligible to occur. Thus, any state that satisfies two AMP in the plant will be divided into two states in the reconfiguration plant. Either of them satisfies only one SMP. For a state satisfying less than two AMP in the plant, the relevant events are defined at states in Q_E^R. If the state satisfies exactly one AMP, it won't be split. Hence, that state will satisfy exactly one SMP.

All formal results and proofs of correctness are provided in Appendix B.

3.3.2 Restrictions

Since the provided proof is for systems with multiple (≥ 2) modes, it seems as if the proposed approach is able to handle reconfiguration problems in systems with multiple modes, while the Problem 2 is defined in a system with two modes. However, though the approach is able to deal with reconfiguration problems in a system with multiple modes, it works in a restrictive way.

The restriction traces back to the definition of strictly public state. Only the state satisfying all aggregated mode predicates of the system can be recognized as a SPBS. This definition is so restrictive that in many cases only the initial state survives. Moreover, since the definition of strictly exit (entry) event is based on the definition of SPBS, the SETE (SEYE) is defined for all modes rather than for two modes. According to the definition of bidirectional reconfiguration specification, after a number of occurrences of SETE and before an equal number of occurrences of SEYE, the reconfiguration event cannot occur. In other words, when some plant component is not at a SPBS, a reconfiguration with respect to any two modes is not permitted to occur. Under this circumstance, technically, any reconfiguration event is still possible to occur, since $Q^G_{SPBS} \neq \emptyset$ according to Lemma 1. However, this approach does ignore some situations. In a system with more than two modes, if a state satisfies two or more mode predicates but not all of them, there will be no reconfiguration event eligible to occur at the state . But this is too restrictive.

Figure 3.19 shows a DES with three modes. In this DES, state q_0 is the only SPBS at which all reconfiguration events can be defined. But if we only focus on the reconfiguration from $\mathbf{Mode}_1$ to $\mathbf{Mode}_2$ where $\sigma_{1,2}$ is the reconfiguration event, the $\sigma_{1,2}$ is also expected to be eligible to occur at state q_1, since $q_1 \models (P_1 \wedge P_2)$. Of course, any reconfiguration relating to $\mathbf{Mode}_3$ cannot be enabled at q_1, but the proposed approach just eliminates this case along with the possibility of $\sigma_{1,2}$'s occurring at q_1. Therefore, this approach is restrictive when solving reconfiguration problems in a system with more than two modes. Fortunately, the approach is still able to solve the bidirectional reconfiguration problems in a system with just two modes.

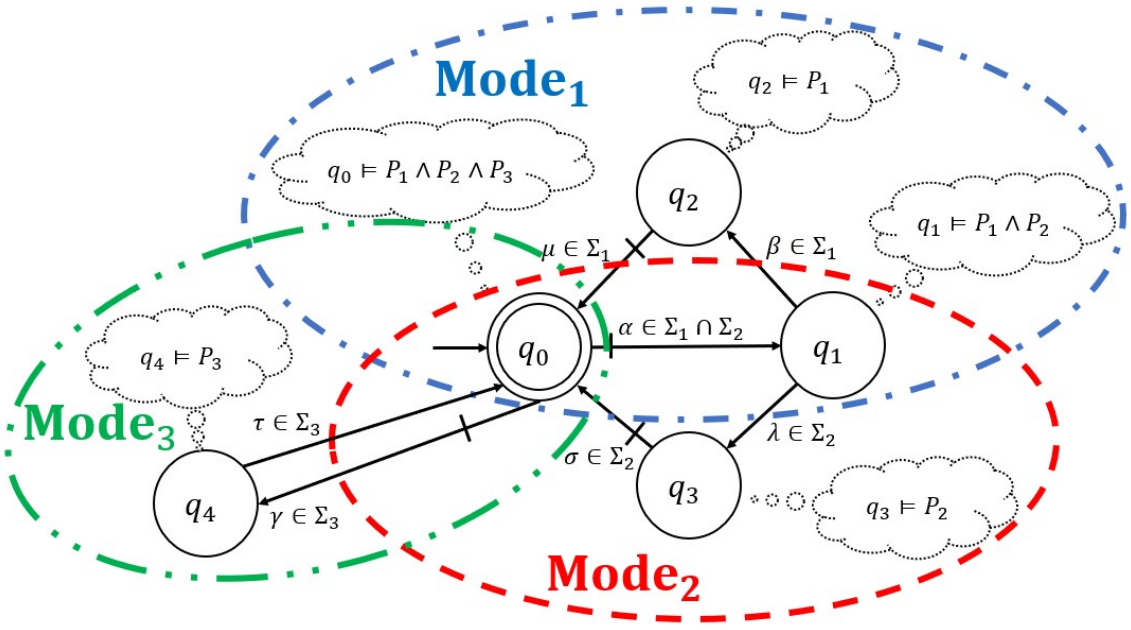

Figure 3.19: Toy example illustrating limitation of proposed approach

3.4 Examples

3.4.1 Toy Example

First of all, the proposed bidirectional reconfiguration approach is applied to an example with two components and two modes.

Consider a small factory with only two machines $\mathbf{M}_1$ and $\mathbf{M}_2$. There are two operating modes of the factory and each mode needs both machines.

From Figure 3.20 we can see that $\Sigma_1 = \{1, 3, 4, 7, 8, 9\}$ and $\Sigma_2 = \{1, 5, 6, 11, 12, 13\}$. Then according to the definition of aggregated mode predicate, in the component $\mathbf{M}_1$, $q_0 \models P_1^{M_1} \wedge P_2^{M_1}, q_1 \models P_1^{M_1} \wedge P_2^{M_1}, q_2 \models P_1^{M_1}$ and $q_3 \models P_2^{M_1}$. Similarly, in $\mathbf{M}_2$, $q_0 \models P_1^{M_2} \wedge P_2^{M_2}, q_1 \models P_1^{M_2}, q_2 \models P_1^{M_2}, q_3 \models P_2^{M_2}$ and $q_4 \models P_2^{M_2}$. Naturally, $Q_{SPBS}^{M_1} = \{q_0, q_1\}, Q_{PRS}^{M_1} = \{q_2, q_3\}$ and $Q_{SPBS}^{M_2} = \{q_0\}, Q_{PRS}^{M_2} = \{q_1, q_2, q_3, q_4\}$.

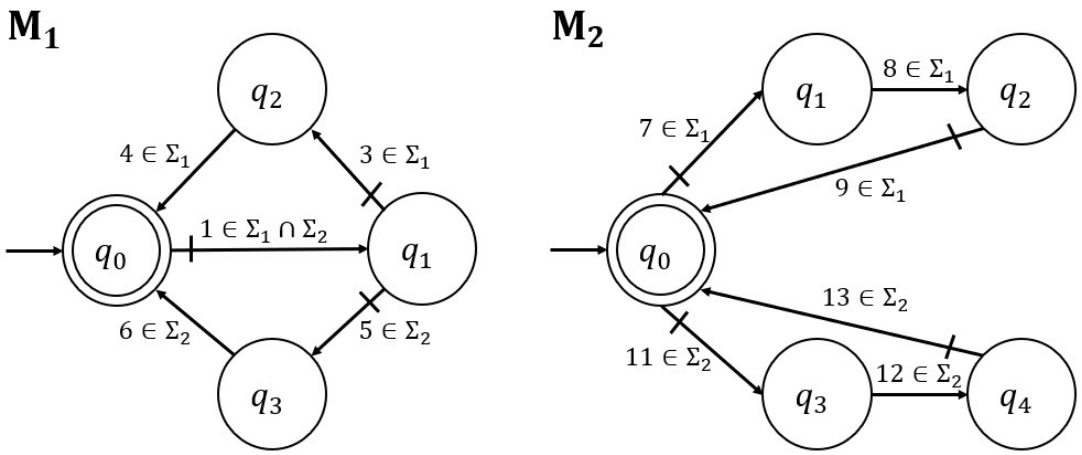

Figure 3.20: System with two plant components and two modes

Thus, in this system, $\Sigma^G_{SETE} = \{3, 5, 7, 11\}, \Sigma^G_{SEYE} = \{4, 6, 9, 13\}, \Sigma^G_{SELE} = \{8, 12\}$ and $\Sigma^G_{SINE} = \{1\}$. Then $b = |\{r|\Sigma^r_{SETE} \neq \emptyset\}| = 2$. According to the definition of BRS, the BRS $\mathbf{R}_B$ for this system is shown in Figure 3.21.

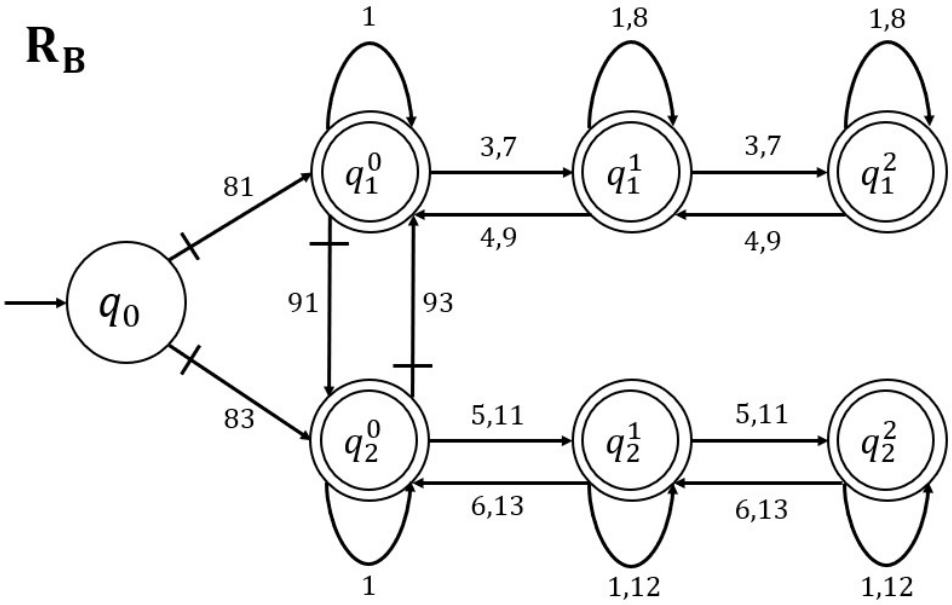

Figure 3.21: Reconfiguration specification $\mathbf{R}_B$

In $\mathbf{R}_B$, apart from the events in Σ, there are also four extra controllable events. Among them, $81 \in \Sigma_{MIE}$ is the mode initialization event for $\mathbf{Mode}_1$, $83 \in \Sigma_{MIE}$ is the mode initialization event for $\mathbf{Mode}_2$, $91 \in \Sigma_{RE}$ is the reconfiguration event from $\mathbf{Mode}_1$ to $\mathbf{Mode}_2$, and $93 \in \Sigma_{RE}$ is the reconfiguration event from $\mathbf{Mode}_2$ to $\mathbf{Mode}_1$.

We then compute

$$\mathbf{RG}_B = \mathbf{Sync}(\mathbf{M}_1, \mathbf{M}_2, \mathbf{R}_B).$$

The resulting reconfiguration plant $\mathbf{RG}_B$ can be obtained by the supervisory controller synthesis software TCT. The $\mathbf{RG}_B$ obtained from TCT is shown in Figure 3.22.

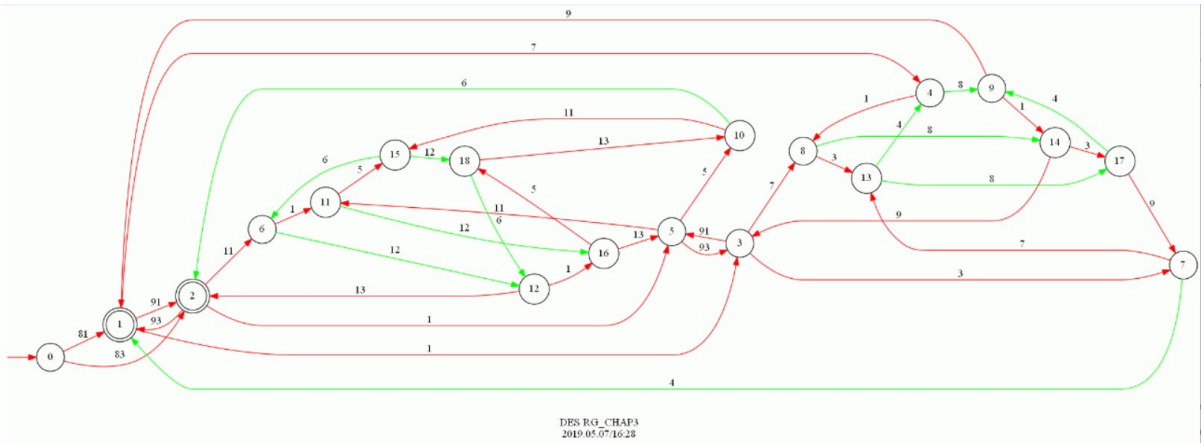

Figure 3.22: Reconfiguration plant $\mathbf{RG}_B$

We then check the five requirements described in Problem 2.

The first requirement states that $\mathbf{RG}_B$ can start in either of the two modes via a transition labeled by a mode initialization event. From the state 0 of $\mathbf{RG}_B$, there is a transition labeled by the MIE 81 for $\mathbf{Mode}_1$ to state 1, meaning that $\mathbf{RG}_B$ can start in $\mathbf{Mode}_1$. There is also a transition labeled by the MIE 83 for $\mathbf{Mode}_2$ from state 0 to state 1, meaning that $\mathbf{RG}_B$ can start in $\mathbf{Mode}_2$.

Part (a) of the second requirement states that the reconfiguration can be done when all plant components are at strictly public states. In $\mathbf{RG}_B$, the RE 91 is defined at state 1 and state 3. State 1 is reached from state 0 via the event 81, which means that $\mathbf{M}_1$ is at state 0 and $\mathbf{M}_2$ is at state 0. These two states are both SPBS. For the state 3 of $\mathbf{RG}_B$, it is reached from state 0 via a string "$< 81 \cdot 1 >$", which means that $\mathbf{M}_1$ is at state 1 and $\mathbf{M}_2$ is at state 0. These two states are both SPBS. In $\mathbf{RG}_B$, the RE 93 is defined at state 2 and state 5. The state 2 is reached from state 0 via the event 83, which means that $\mathbf{M}_1$ is at state 0 and $\mathbf{M}_2$ is at state 0. These two states are both SPBS. For state 5 of $\mathbf{RG}_B$, it is reached from state 0 via a string "$< 83 \cdot 1 >$", which means that $\mathbf{M}_1$ is at state 1 and $\mathbf{M}_2$ is at state 0. These two states are both SPBS. Thus, part (a) of the

second requirement is met.

Part (b) of the second requirement states that each reconfiguration event exists in $\mathbf{RG}_B$. Indeed, in $\mathbf{RG}_B$, the transition from state 1 to state 2 (or from state 3 to state 5) is labeled by the RE 91 and the transition from state 5 to state 3 (or from state 2 to state 1) is labeled by the RE 93.

Part (c) of the second requirement states that from one mode, the reconfiguration to the other mode is always possible when the source state is in the source mode and the target state is in the target mode. According to the definition of SMP, state 1 and state 3 satisfy $P_1^{RG_B}$, state 2 and state 5 satisfy $P_2^{RG_B}$. Thus, part (c) of the second requirement is met.

The third requirement states that the reconfiguration can be operated back-and-forth without blocking. We can see the back-and-forth reconfiguration between state 1 and state 2, or between state 3 and state 5. Thus, the third requirement is met.

The fourth requirement states that every reachable state of $\mathbf{Mode}_1$ or $\mathbf{Mode}_2$ in $\mathbf{RG}_B$ is also coreachable separately, which means that each mode is nonblocking in $\mathbf{RG}_B$. In $\mathbf{Mode}_1$, the only marked state is the state 1, and all states in $Q_1^{RG_B}$ can reach state 1 via some events. Similarly in $\mathbf{Mode}_2$, the only marked state is the state 2, and all states in $Q_2^{RG_B}$ can reach state 2 via some events. Thus, the fourth requirement is met.

The fifth requirement states that every state except for the initial state in $\mathbf{RG}_B$ is required to satisfy exactly one SMP. According to the definition of SMP, $Q_1^{RG_B} = \{1, 3, 4, 7, 8, 9, 13, 14, 17\}, Q_2^{RG_B} = \{2, 5, 6, 10, 11, 12, 15, 16, 18\}$. It is evident that $Q_1^{RG_B} \cap Q_2^{RG_B} = \emptyset$ and $Q_1^{RG_B} \dot{\cup} Q_2^{RG_B} \dot{\cup} \{0\} = Q^{RG_B}$. Thus, the fifth requirement is met.

Therefore, the five requirements are met by using the proposed bidirectional reconfiguration approach, which is as our expectation. In addition to the manual checking procedure, the author has also developed a software program in C++ to check the five requirements. The result obtained from the program coincides with the result from manual checking. The source code of the program has been uploaded to https://github.com/JasonZhangjc/.

3.4.2 Examples from the Literature

We apply the proposed bidirectional reconfiguration approach to a reconfiguration problem from the literature.

According to Example 3.9.1 in [1], the following Figure 3.23 shows a system with three plant components and two modes.

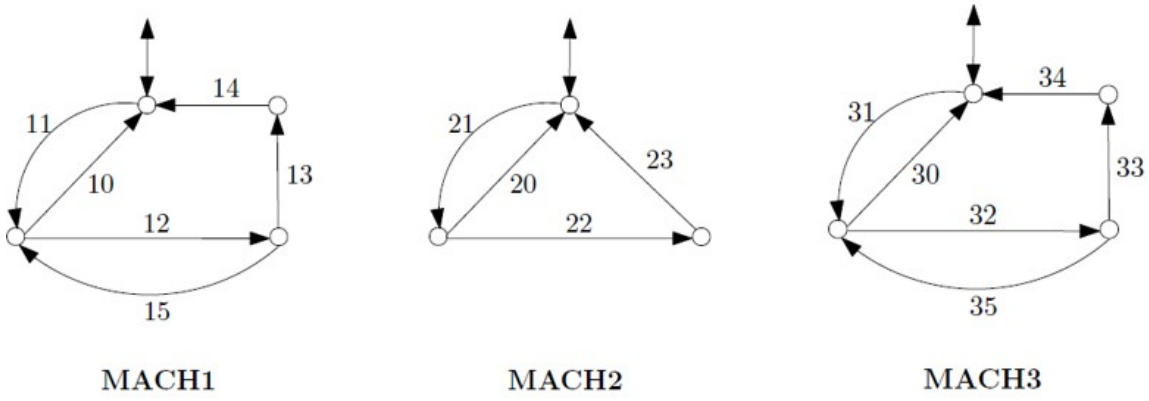

Figure 3.23: Example in literature with three plant components and two modes [1]

Figure 3.23 shows a system with two versions of Small Factory (see Example 3.3.19 in [1]), namely the system has two modes. $\mathbf{Mode}_1$ is **FACT1**, consisting of an "advanced" machine **MACH1** and a "primitive" machine **MACH2**. $\mathbf{Mode}_2$ is **FACT2**, consisting of **MACH1** and an advanced machine **MACH3** similar to **MACH1**. The plant $\mathbf{G}$ = **MACH1**||**MACH2**||**MACH3**.

On the one hand, the two modes are different in components. On the other hand, the two modes are distinguished by two event alphabets, i.e. $\Sigma_1 = \{10, 11, 12, 13, 14, 15, 20, 21, 22, 23\}$ and $\Sigma_2 = \{10, 11, 12, 13, 14, 15, 30, 31, 32, 33, 34, 35\}$. Note that all events in Σ^{MACH2} are in $\mathbf{Mode}_1$ and all events in Σ^{MACH3} are in $\mathbf{Mode}_2$.

According to the two event alphabets, we can see that in **MACH1**, all states are SPBS; while in **MACH2** and **MACH3**, only the two initial states are SPBS. Thus, $\Sigma^G_{SETE} = \{21, 31\}$, $\Sigma^G_{SEYE} = \{20, 23, 30, 34\}$, $\Sigma^G_{SELE} = \{22, 32, 33, 35\}$, and $\Sigma^G_{SINE} = \{10, 11, 12, 13, 14, 15\}$. Since only the event alphabets of **MACH2** and **MACH3** include SETE, $b = 2$.

Let the mode initialization events for $\mathbf{Mode}_1$ and $\mathbf{Mode}_2$ be 81 and 83. Let the

reconfiguration event for $\mathbf{Mode}_1$ to $\mathbf{Mode}_2$ be 91 and the RE for $\mathbf{Mode}_2$ to $\mathbf{Mode}_1$ be 93. Then the bidirectional reconfiguration specification **RB391** is constructed manually as Figure 3.24.

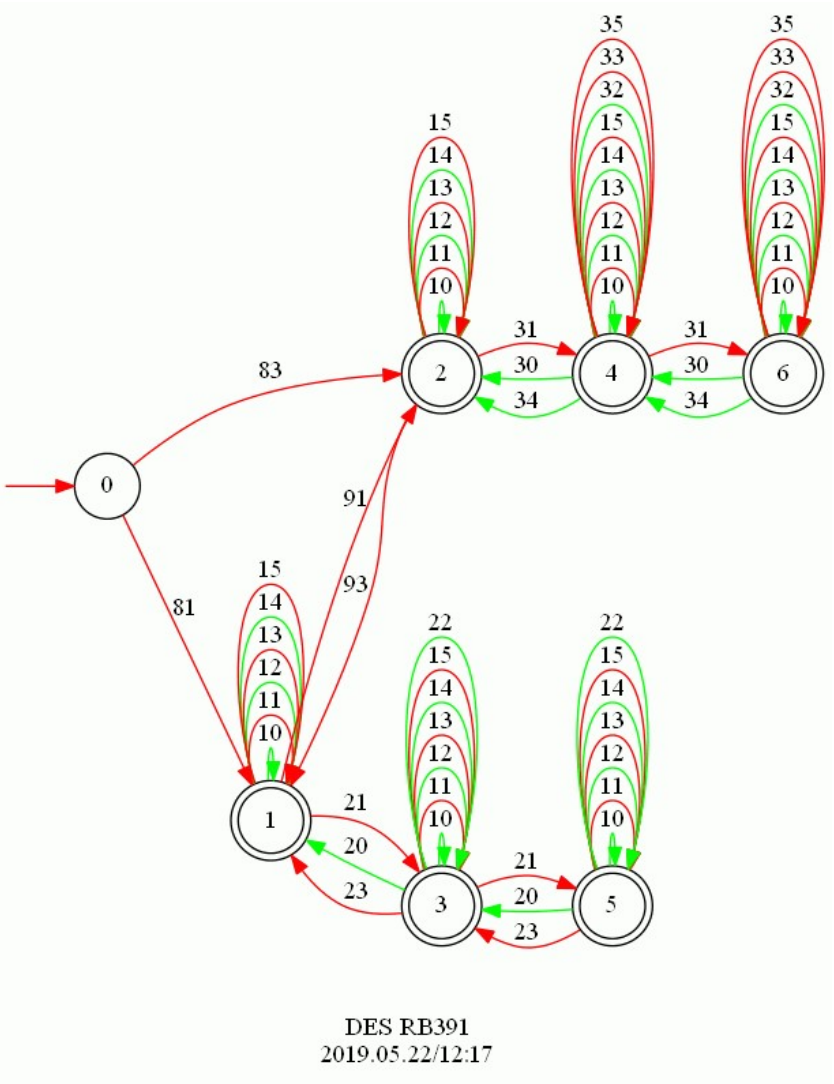

Figure 3.24: Bidirectional reconfiguration specification **R391**

We then compute

$$\mathbf{RG391} = \mathbf{Sync}(\mathbf{MACH1}, \mathbf{MACH2}, \mathbf{MACH3}, \mathbf{RB391}) \quad (29{,}92)$$

The resulting reconfiguration plant is shown in Figure 3.25.

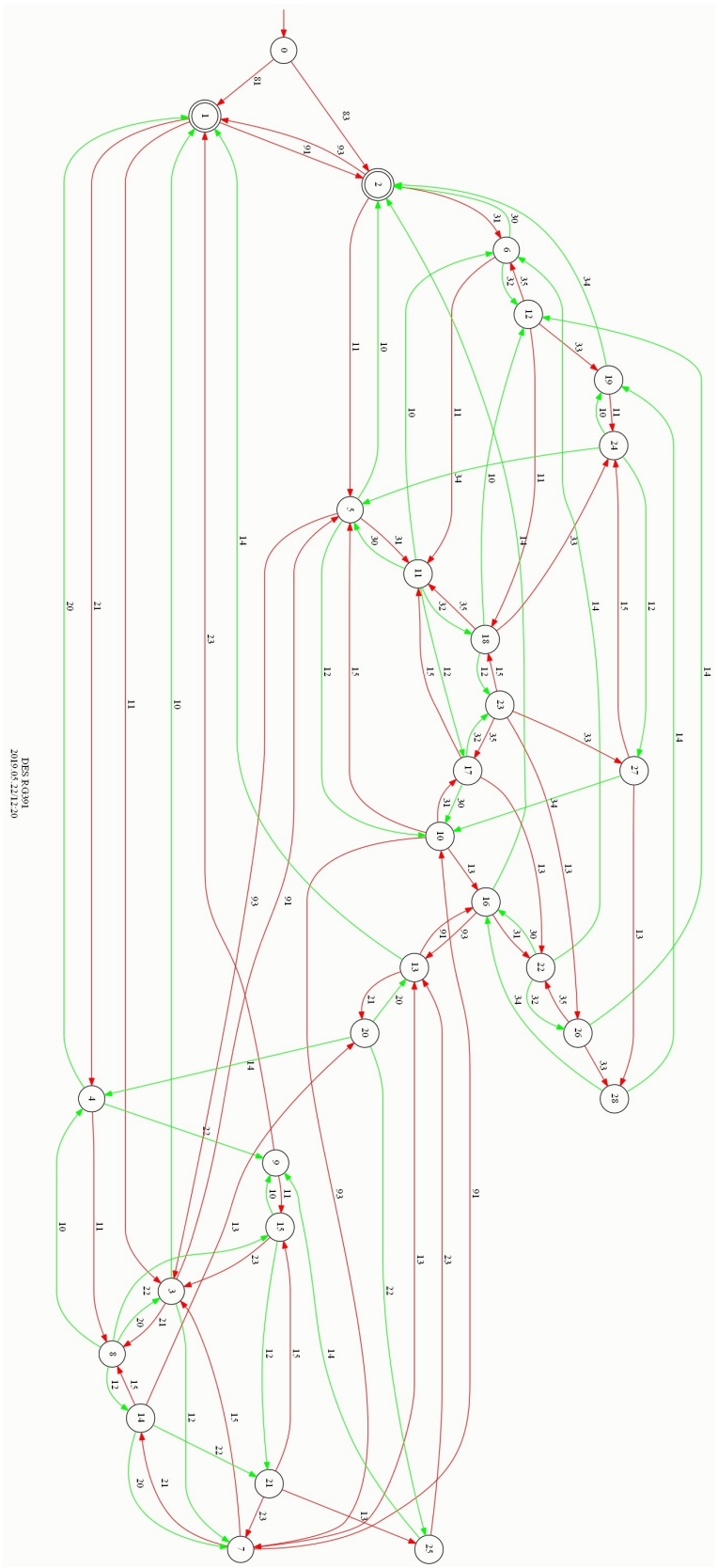

Figure 3.25: Reconfiguration plant **RG391**

The resulting **RG391** has no blocking states. The two modes are completely decoupled, with RE 91 and 93 between them. Note that for each pair of states which are the source and the target states of RE 91 (93), there is also a RE 93 (91) defined from the target state to the source state. According to the software program in C++, the **RG391** meets the five requirements. The reader interested in this aspect may also check them manually. The original example also has two behavioral specifications [1]. We will deal with them in Chapter 4.

The example 3.9.2 in [1] is also about reconfiguration. Again we consider a version of Small Factory, with standard machines **MACH1**, **MACH2**, adapted to work with either mode specification as needed. $\mathbf{Mode}_1, \mathbf{Mode}_2$ are distinguished by buffer specifications according to the buffer capacities 3 (for **BUF1**, in $\mathbf{Mode}_1$), or 1 (**BUF2**, in $\mathbf{Mode}_2$). Thus the bidirectional reconfiguration must prevent both blocking (owing to a work-piece left behind in **BUF1** or **BUF2**) and buffer overflow. The plant and specification components are shown below.

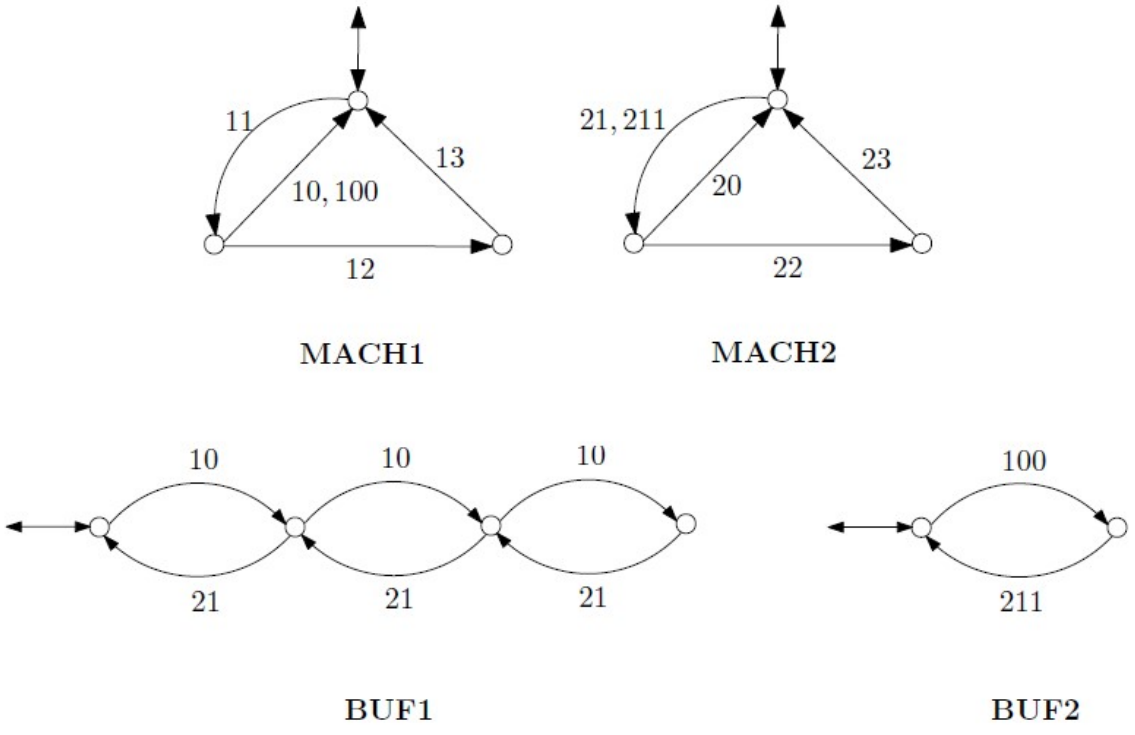

Figure 3.26: Example in literature with two plant components and two modes [**?**]

In this example, although the two modes are distinguished by specifications, the essence

of the difference is the different event alphabets. According to Figure 3.26, the two event alphabets are $\Sigma_1 = \{11, 10, 12, 13, 21, 20, 22, 23\}$ and $\Sigma_2 = \{11, 100, 12, 13, 211, 20, 22, 23\}$. Thus, in both **MACH1** and **MACH2**, all states are SPBS, hence there is no SETE, SEYE, or SELE. Then according to Definition 11, $b = 0$. Therefore, this example is a relatively special one, in which the naive way of combining two unidirectional reconfiguration specifications (as shown in Figure 3.10) would also work. Accordingly, we can construct the bidirectional reconfiguration specification **RB392**.

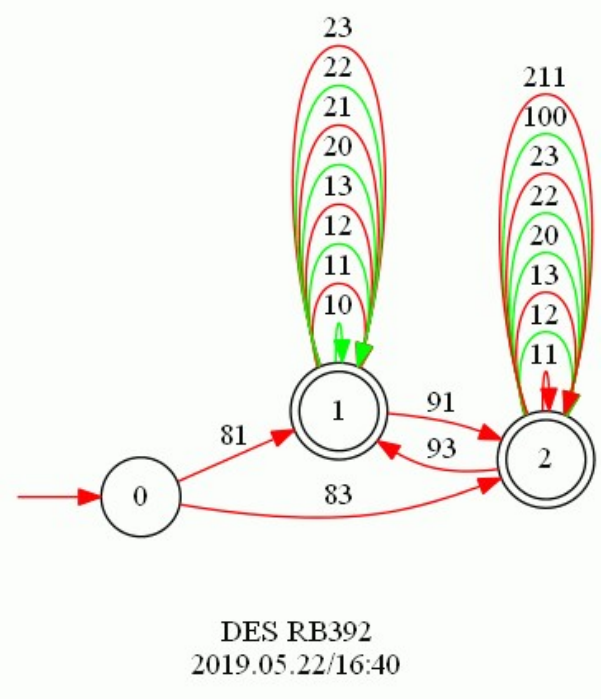

Figure 3.27: Reconfiguration specification **RB392**

We then compute

$$\mathbf{RG392} = \mathbf{Sync}(\mathbf{MACH1}, \mathbf{MACH2}, \mathbf{MACH3}, \mathbf{RB392}) \quad (19{,}68)$$

The resulting reconfiguration plant **RG392** is shown in Figure 3.28.

The resulting **RG392** can start in either $\mathbf{Mode}_1$ or $\mathbf{Mode}_2$. Each state except for the initial state in **RG392** satisfies exactly one SMP. According to the software program in C++, **RG392** meets the five requirements. The reader interested in this aspect may also check them manually. The two behavioral specifications will be dealt with in Chapter 4.

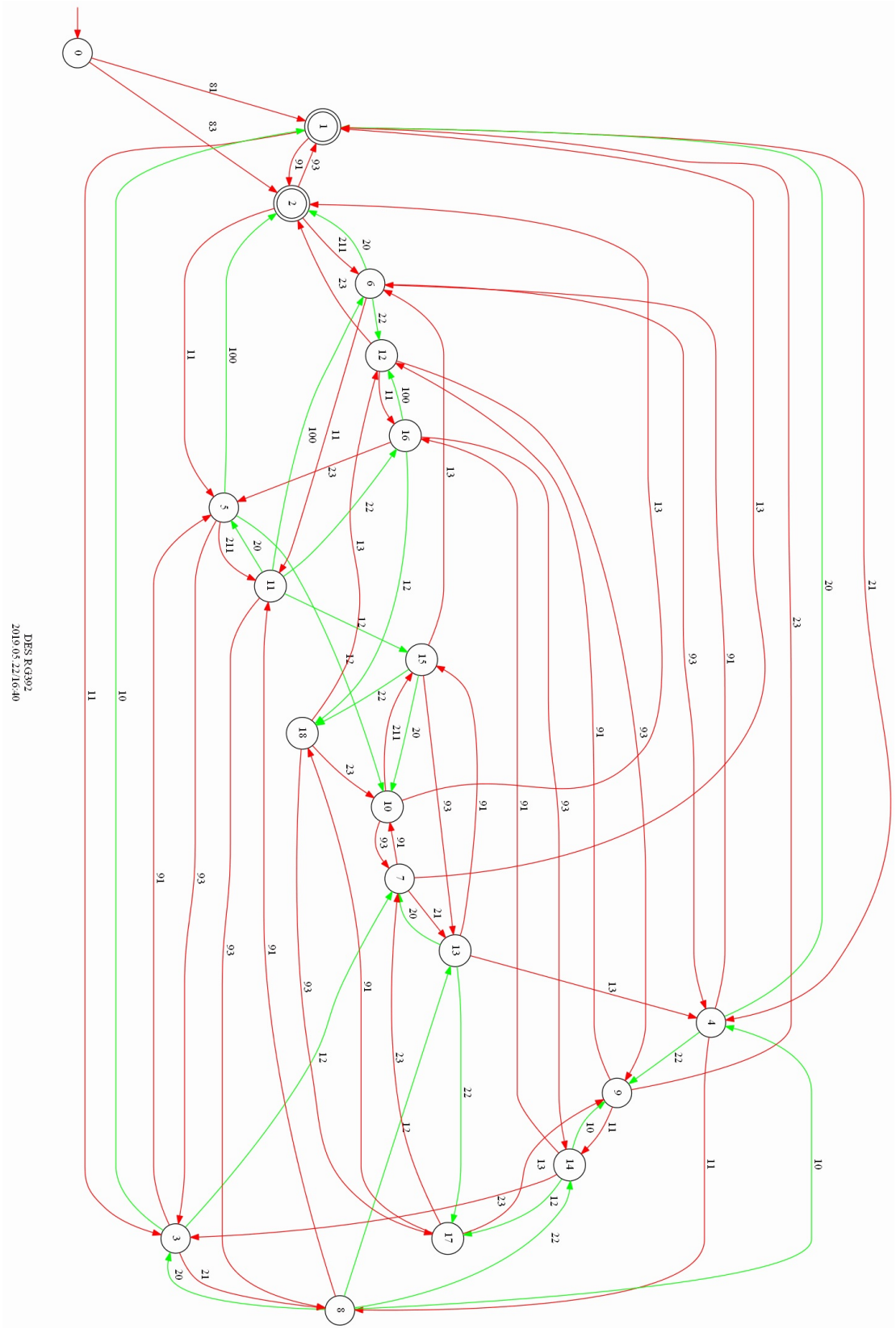

Figure 3.28: Reconfiguration plant **RG392**

3.5 Chapter Summary and Discussion

First of all, this chapter has selected the reconfiguration problem on parallel-mode systems, which is convenient in notation and commonly used in daily life. Then this chapter has formally defined the bidirectional reconfiguration problem on PMS in the SCT framework of DES and analyzed in detail the inadequacy of unidirectional reconfiguration approach. In order to solve the problem, the bidirectional reconfiguration specification along with related notions has been defined to be synchronized with the plant. The proposed bidirectional reconfiguration approach is convenient to apply since it only needs to construct one reconfiguration specification. Thus, when there are only two modes in the plant, this approach should always be considered first.

However, a question that might come up concerning the problem formulation and the proposed approach is: Suppose the system is currently operating in $\mathbf{Mode}_1$ (say), and the user wants to keep it there. What prevents it from uncontrollably slipping into $\mathbf{Mode}_2$ (say), i.e. executing some uncontrollable event that causes it to switch modes unintentionally? We discuss this aspect along with the solvability of Problem 2 in Appendix C.

Finally, the inadequacy of the bidirectional reconfiguration approach in a system with more than two modes (multiple reconfiguration problems) has been reported. The inadequacy is due to the restrictive definition of public states and exit (entry) events. In the next chapter, this restriction will be removed to solve multiple reconfiguration problems.

Chapter 4

Monolithic Multiple Reconfiguration of DES

4.1 Introduction

The approach to bidirectional reconfiguration of discrete-event systems has been proved effective to automatically, dynamically and bidirectionally reconfigure a discrete-event system from one configuration to another. However, to solve a multiple reconfiguration problem (a reconfiguration problem in a system with more than two modes) by this approach, a naive construction of the bidirectional reconfiguration specification would result in a relatively restrictive reconfiguration plant.

In fact, the bidirectional reconfiguration approach is only able to solve the bidirectional reconfiguration problem in a system with two modes. Namely, it fails to realize the mechanism of multiple reconfiguration. Fortunately, the bidirectional reconfiguration approach provides the foundation for multiple reconfiguration. By refining the bidirectional reconfiguration approach presented in the last chapter, the multiple reconfiguration mechanism can also be achieved.

This chapter presents a monolithic architecture to dynamically and bidirectionally reconfigure discrete-event systems with multiple (more than two) modes. To this end, we first review why the bidirectional reconfiguration approach in chapter 3 doesn't work. Then we reformulate the restrictive definitions in Chapter 3, followed by the procedure to construct a new reconfiguration specification. With this slightly complicated reconfig-

uration specification, the multiple reconfiguration of DES can be managed.

This chapter is organized as follows. Section 4.2 informally defines the multiple reconfiguration problem, reviews the inadequacy of the bidirectional reconfiguration approach for this problem and provides a formal definition of multiple reconfiguration problems. Section 4.3 elaborates on the construction of the new reconfiguration specification inspired by the bidirectional reconfiguration approach. Section 4.4 investigates the validity of the reconfiguration after a supervisory controller is applied. Section 4.5 introduces the specification to trigger the reconfiguration and Section 4.6 focuses on checking the guaranteed reachability from a source state to a target state. Section 4.7 illustrates the proposed method with an example, and conclusions are drawn in Section 4.8.

4.2 Multiple Reconfiguration Problem

4.2.1 Informal Problem Definition

With the definition of mode in Chapter 3 and the informal definition of the bidirectional reconfiguration problem, an informal definition of the multiple reconfiguration problem is provided below.

Problem 3. [Multiple Reconfiguration of DES (informal)]. For a DES **G** with n modes, i.e. $\mathbf{Mode}_1, ..., \mathbf{Mode}_n$, synthesize a reconfiguration plant represented by the DES **RG** such that in **RG**:

(i) **RG** can start in any mode;

(ii) From one mode, reconfiguration to another mode is possible only when the plant is at a state shared by the two modes;

(iii) Reconfiguration can be operated back-and-forth without blocking;

(iv) Nonblocking is guaranteed for each mode separately.

◇

As we have mentioned in Chapter 3, the bidirectional approach is not able to solve this problem mainly because of the restrictive definitions of public state, exit event and entry event.

4.2.2 Redefined Notions

In fact, as a public state, a state doesn't have to satisfy all AMP. It's better to analyze the public state locally. For example, when we are considering reconfiguration from $\mathbf{Mode}_i$ to $\mathbf{Mode}_j$, we just need to find out all the states that satisfy both P_i^k and P_j^k for each plant component $\mathbf{G}^k$. We then let the reconfiguration event $\sigma_{i,j}$ be eligible at these states. Thus, the formal definition of public state is as follows.

Definition 12. [Public State (PBS)]. Denote by $\mathbf{G} = \mathbf{G}^1||...||\mathbf{G}^h$ the plant DES formed by synchronization, and $\mathbf{Mode}_1, ..., \mathbf{Mode}_n$ the n modes. Also denote by $P_i^k : Q^k \to \{0, 1\}$ the AMP of $\mathbf{Mode}_i$ on Q^k. For a state $q \in Q^k$, q is a *public state* with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ iff

$$q \models (P_i^k \wedge P_j^k) \tag{4.1}$$

Let $Q^k_{PBS,i,j}$ be the set of all PBS with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ in $\mathbf{G}^k$, and $Q^G_{PBS,i,j}$ be the set of all PBS with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ in $\mathbf{G}$.

◇

Remark. The initial state of each plant component $\mathbf{G}^k$ is a PBS with respect to any pair of modes in the system, since it satisfies all aggregated mode predicates, i.e.

$$(\forall k = 1, ..., h)\ q_o^k \models (P_1^k \wedge ... \wedge P_n^k) \tag{4.2}$$

Note that the public states with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ are the same as the public states with respect to $\mathbf{Mode}_j$ and $\mathbf{Mode}_i$, i.e. $Q^k_{PBS,i,j} = Q^k_{PBS,j,i}$ and $Q^G_{PBS,i,j} = Q^G_{PBS,j,i}$. In the rest of this report, we use $Q^k_{PBS,i,j}$ for $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ when $i < j$.

The example shown in Figure 4.1 illustrates the notion of PBS in the plant DES. In the example, there are two modes distinguished by $\Sigma_1 = \{\alpha, \beta, \mu\}, \Sigma_2 = \{\alpha, \lambda, \sigma\}, \Sigma_3 = \{\gamma, \tau\}$. Evidently, states q_0 and q_1 are two PBS with respect to $\mathbf{Mode}_1$ and $\mathbf{Mode}_2$. At the same time, q_0 is also a PBS with respect to $\mathbf{Mode}_1$ and $\mathbf{Mode}_3$ or $\mathbf{Mode}_2$ and $\mathbf{Mode}_3$.

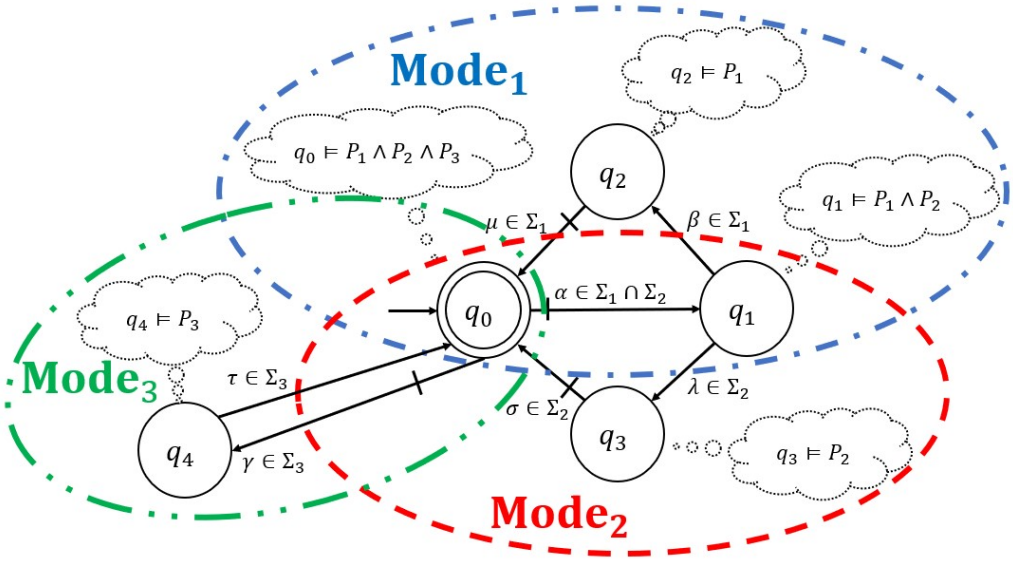

Figure 4.1: Toy example illustrating PBS

The importance of PBS is that the reconfiguration from $\mathbf{Mode}_i$ to $\mathbf{Mode}_j$ and the reconfiguration from $\mathbf{Mode}_j$ to $\mathbf{Mode}_i$ are always expected to be done at PBS with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$.

With the definition of PBS, it is not sufficient to construct a reconfiguration specification, since it's better to characterize an event or a string instead of a state in the SCT framework of DES. Thus, the definitions for the exit event (ETE) and the entry event (EYE) are also reformulated.

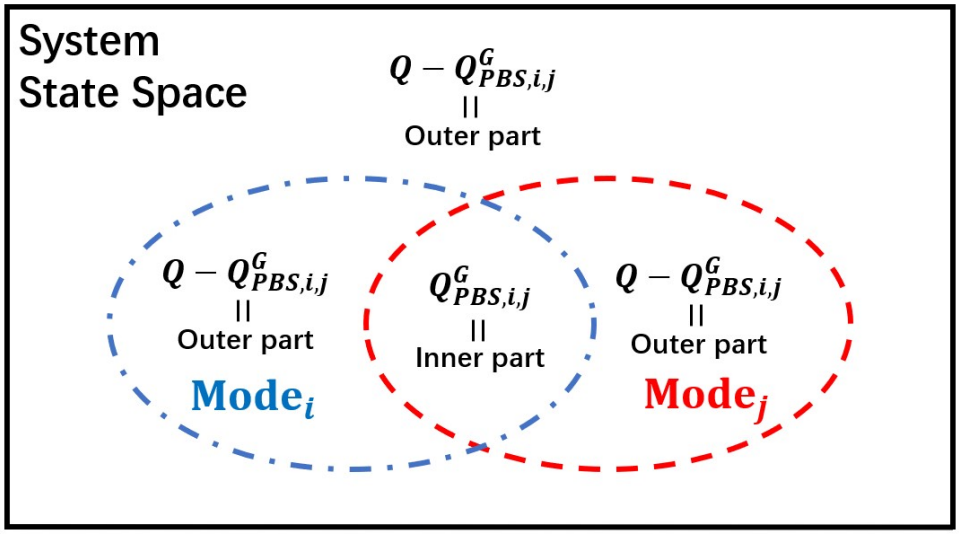

Figure 4.2: Illustration of the inner and outer parts of a system with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$

For two modes $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ of the system (see Figure 4.2) let the inner part be the intersection of the two modes, namely the PBS with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ in $\mathbf{G}$. Then the outer part is naturally any part of the system that doesn't satisfy P_i^G and P_j^G at the same time.

To generalize this setting, consider an arbitrary plant component $\mathbf{G}^k$. For $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$, the inner part is the PBS with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ in $\mathbf{G}^k$; and the outer part is any part of $\mathbf{G}^k$ that doesn't satisfy P_i^k and P_j^k at the same time.

An exit event with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ is an event that leads a plant component $\mathbf{G}^k$ from the inner part to the outer part of the two modes. By contrast, an entry event with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ is an event that leads a plant component $\mathbf{G}^k$ from the outer part to the inner part of the two modes.

Definition 13. [Exit Event (ETE)]. An event $\sigma \in \Sigma$ is an *exit event* with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ iff

$$(\exists k = 1, ..., h)(\exists q, q^{'} \in Q^k)\ [\delta^k(q, \sigma) = q^{'} \wedge q \in Q^k_{PBS,i,j} \wedge q^{'} \notin Q^k_{PBS,i,j}] \tag{4.3}$$

Let $\Sigma^k_{ETE,i,j}$ be the set of exit events with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ in $\mathbf{G}^k$, and $\Sigma^G_{ETE,i,j}$ the set of exit events with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ in $\mathbf{G}$. Let $\Sigma^G_{ETE,}$ be the set of all exit events in $\mathbf{G}$.

$\diamond$

Remark. Evidently, $\Sigma^k_{ETE,i,j} = \Sigma^G_{ETE,i,j} \cap \Sigma^k$ and $\Sigma^G_{ETE} = \bigcup_{i,j \in 1,...,n} \Sigma^G_{ETE,i,j}$. Also note that $\Sigma^k_{ETE,i,j} \subseteq \Sigma^k$.

An ETE with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ is the same as an ETE with respect to $\mathbf{Mode}_j$ and $\mathbf{Mode}_i$, namely $\Sigma^k_{ETE,i,j} = \Sigma^k_{ETE,j,i}$.

In practice, if the plant component (or the plant itself) leaves a public state with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ via an event, then either reconfiguration from $\mathbf{Mode}_i$ to $\mathbf{Mode}_j$ or reconfiguration from $\mathbf{Mode}_j$ to $\mathbf{Mode}_i$ is not eligible. Thus, an exit event with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ is also an exit event with respect to $\mathbf{Mode}_j$ and $\mathbf{Mode}_i$.

The $\Sigma^G_{ETE,i,j}$ will be used in the formal definition of the reconfiguration specification, which specifies each pair of reconfiguration events for the bidirectional reconfiguration.

a RE from $\mathbf{Mode}_i$ to $\mathbf{Mode}_j$ also has a corresponding RE from $\mathbf{Mode}_j$ to $\mathbf{Mode}_i$, and they will be treated together later.

Definition 14. [Entry Event (EYE)]. An event $\sigma \in \Sigma$ is an *entry event* with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ iff

$$(\exists k = 1, ..., h)(\exists q, q^{'} \in Q^k)\ [\delta^k(q, \sigma) = q^{'} \wedge q \notin Q^k_{PBS,i,j} \wedge q^{'} \in Q^k_{PBS,i,j}] \tag{4.4}$$

Let $\Sigma^k_{EYE,i,j}$ be the set of entry events with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ in $\mathbf{G}^k$, and $\Sigma^G_{EYE,i,j}$ the set of entry events with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ in $\mathbf{G}$. Let $\Sigma^G_{EYE,}$ be the set of all entry events in $\mathbf{G}$.

$\diamond$

Remark. Evidently, $\Sigma^k_{EYE,i,j} = \Sigma^G_{EYE,i,j} \cap \Sigma^k$ and $\Sigma^G_{EYE} = \bigcup_{i,j \in 1,...,n} \Sigma^G_{EYE,i,j}$. Also note that $\Sigma^k_{EYE,i,j} \subseteq \Sigma^k$.

An EYE with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ is the same as an EYE with respect to $\mathbf{Mode}_j$ and $\mathbf{Mode}_i$, namely $\Sigma^k_{EYE,i,j} = \Sigma^k_{EYE,j,i}$.

In practice, if the plant component (or the plant itself) enters a public state with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ via an event, then either reconfiguration from $\mathbf{Mode}_i$ to $\mathbf{Mode}_j$ or reconfiguration from $\mathbf{Mode}_j$ to $\mathbf{Mode}_i$ is eligible. Thus, an entry event with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ is also an entry event with respect to $\mathbf{Mode}_j$ and $\mathbf{Mode}_i$.

The $\Sigma^G_{EYE,i,j}$ will be used in the formal definition of the reconfiguration specification, which specifies each pair of reconfiguration events for the bidirectional reconfiguration..

The example shown in Figure 4.1 also illustrates the notion of ETE and EYE in a plant DES. In the example, there are three modes distinguished by $\Sigma_1 = \{\alpha, \beta, \mu\}, \Sigma_2 = \{\alpha, \lambda, \sigma\}, \Sigma_3 = \{\gamma, \tau\}$. For $\mathbf{Mode}_1$ and $\mathbf{Mode}_2$, the PBS are q_0 and q_1. Then the ETE with respect to $\mathbf{Mode}_1$ and $\mathbf{Mode}_2$ are β, λ, γ. Note that the event γ leads the plant to a state in $\mathbf{Mode}_3$, which seems irrelevant to $\mathbf{Mode}_1$ and $\mathbf{Mode}_2$. But, γ is also an ETE with respect to $\mathbf{Mode}_1$ and $\mathbf{Mode}_2$ ($\mathbf{Mode}_2$ and $\mathbf{Mode}_1$).

In contrast, the EYE with respect to $\mathbf{Mode}_1$ and $\mathbf{Mode}_2$ are μ, σ, τ. Note that the event τ leads the plant from a state in $\mathbf{Mode}_3$ back to a PBS with respect to $\mathbf{Mode}_1$ and $\mathbf{Mode}_2$, which seems irrelevant to $\mathbf{Mode}_1$ and $\mathbf{Mode}_2$, but it is also an EYE with respect to $\mathbf{Mode}_1$ and $\mathbf{Mode}_2$ ($\mathbf{Mode}_2$ and $\mathbf{Mode}_1$).

Similarly, since $Q^G_{PBS,1,3} = \{q_0\}$, $\Sigma_{ETE,1,3} = \{\alpha, \gamma\}$ and $\Sigma_{EYE,1,3} = \{\mu, \sigma, \tau\}$. Since $Q^G_{PBS,2,3} = \{q_0\}$, $\Sigma_{ETE,2,3} = \{\alpha, \gamma\}$ and $\Sigma_{EYE,2,3} = \{\mu, \sigma, \tau\}$.

In the later procedure of constructing the reconfiguration specification, two assumptions on exit events and entry events are required. The first states that for two arbitrary modes $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$, the exit events for distinct plant components must be distinct, i.e.

$$(\forall i, j \in 1...n)(\forall k, r = 1, ..., h, k \neq r)\ \Sigma^k_{ETE,i,j} \cap \Sigma^r_{ETE,i,j} = \emptyset. \tag{4.5}$$

Similarly, the second states that for two arbitrary modes $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$, the entry events for distinct plant components must be distinct, i.e.

$$(\forall i, j \in 1...n)(\forall k, r = 1, ..., h, k \neq r)\ \Sigma^k_{EYE,i,j} \cap \Sigma^r_{EYE,i,j} = \emptyset. \tag{4.6}$$

In case there is some shared event, a relabeling could be applied. For example, consider an event σ in the event alphabets of several plant components. This case reflects an unsuitable labeling of events since it may result in confusion of event meanings. Then relabeling can be done to distinguish them. Therefore, a new assumption is required that

$$\Sigma^G_{ETE,i,j} \cap \Sigma^G_{EYE,i,j} = \emptyset. \tag{4.7}$$

To fully classify the events with respect to two modes, the definitions of inner event (INE) and external event (ELE) are also required. These two classes of events will be used in the proof work later. Intuitively, an inner event with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ is an event that leads a plant component $\mathbf{G}^k$ from the inner part to the inner part of the two modes. By contrast, an external event with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ is an event that leads a plant component $\mathbf{G}^k$ from the outer part to the outer part of the two modes.

Definition 15. [Inner Event (INE)]. An event $\sigma \in \Sigma$ is an *inner event* with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ iff

$$(\exists k = 1, ..., h)(\exists q, q^{'} \in Q^k)\ [\delta^k(q, \sigma) = q^{'} \wedge q \in Q^k_{PBS,i,j} \wedge q^{'} \in Q^k_{PBS,i,j}] \tag{4.8}$$

Let $\Sigma^k_{INE,i,j}$ be the set of inner events with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ in $\mathbf{M}_k$, and

$\Sigma^G_{INE,i,j}$ the set of inner events with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ in $\mathbf{G}$. Let $\Sigma^G_{INE,}$ be the set of all inner events in $\mathbf{G}$.

$\diamond$

Remark. Evidently, $\Sigma^k_{INE,i,j} = \Sigma^G_{INE,i,j} \cap \Sigma^k$ and $\Sigma^G_{INE} = \bigcup_{i,j\in 1,...,n} \Sigma^G_{INE,i,j}$. Also note that $\Sigma^k_{INE,i,j} \subseteq \Sigma^k$.

An INE with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ is the same as an INE with respect to $\mathbf{Mode}_j$ and $\mathbf{Mode}_i$, namely $\Sigma^k_{ETE,i,j} = \Sigma^k_{ETE,j,i}$.

Definition 16. [External Event (ELE)]. An event $\sigma \in \Sigma$ is an *external event* with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ iff

$$(\exists k = 1, ..., h)(\exists q, q^{'} \in Q^k)\ [\delta^k(q, \sigma) = q^{'} \wedge q \notin Q^k_{PBS,i,j} \wedge q^{'} \notin Q^k_{PBS,i,j}] \tag{4.9}$$

Let $\Sigma^k_{ELE,i,j}$ be the set of external events with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ in $\mathbf{M}_k$, and $\Sigma^G_{ELE,i,j}$ the set of external events with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ in $\mathbf{G}$. Let $\Sigma^G_{ELE,}$ be the set of all external events in $\mathbf{G}$.

$\diamond$

Remark. Evidently, $\Sigma^k_{ELE,i,j} = \Sigma^G_{ELE,i,j} \cap \Sigma^k$ and $\Sigma^G_{ETE} = \bigcup_{i,j\in 1,...,n} \Sigma^G_{ELE,i,j}$. Also note that $\Sigma^k_{ELE,i,j} \subseteq \Sigma^k$.

An ELE with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ is the same as an ELE with respect to $\mathbf{Mode}_j$ and $\mathbf{Mode}_i$, namely $\Sigma^k_{ELE,i,j} = \Sigma^k_{ELE,j,i}$. For $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$, if the plant component (or the plant itself) is not at public states with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$, where external events are eligible, either the RE from $\mathbf{Mode}_i$ to $\mathbf{Mode}_j$ or the RE from $\mathbf{Mode}_j$ to $\mathbf{Mode}_i$ is not eligible to occur.

The following Figure 4.3 illustrates the notion of ETE, EYE, ELE and INE.

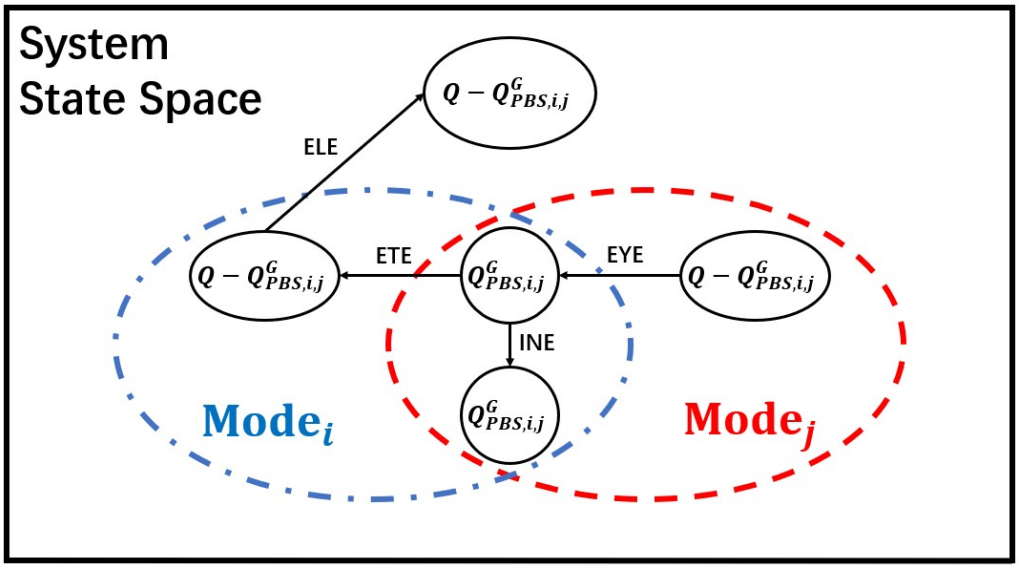

Figure 4.3: Illustration of ETE, EYE, ELE and INE with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$

4.2.3 Formal Problem Definition

With the definitions above, the formal definition of multiple reconfiguration is as follows. In addition to the four requirements mentioned in Problem 1, one extra requirement needs to be formalized. It is that every state except for the initial state in the reconfiguration plant is expected to satisfy only one SMP. The underlying reason is the same as that in Chapter 3.

Problem 4. [Multiple Reconfiguration of DES (Formal)]. For a DES **G** with n modes $\mathbf{Mode}_1, ..., \mathbf{Mode}_n$ that are distinguished by alphabets $\Sigma_1, ..., \Sigma_n$, synthesize a reconfiguration plant $\mathbf{RG} = (Q^{RG}, \Sigma^{RG}, \delta^{RG}, q_o^{RG}, Q_m^{RG})$ such that in **RG**:

(i) **RG** can start in any mode via a transition labeled by a mode initialization event, i.e.

$$(\forall \sigma_i \in \Sigma_{MIE})(\exists q \in Q^{RG})\ \delta^{RG}(q_o^{RG}, \sigma_i) = q; \tag{4.10}$$

(ii) (a) Reconfiguration from $\mathbf{Mode}_i$ to $\mathbf{Mode}_j$ can be done when all plant components are at public states with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$, i.e.

$$\begin{aligned}&(\forall i, j \in 1, ..., n, i \neq j)(\forall s \in L(\mathbf{RG}))\\ &[\delta^{RG}(q_o^{RG}, s\sigma_{i,j})! \Rightarrow (\forall k = 1, ..., h)\ \delta^k(q_o^k, P_{G^k}(s)) \in Q^k_{PBS,i,j}];\end{aligned} \tag{4.11}$$

In this case $P_{G^k} : \Sigma^{RG*} \to \Sigma^{k*}$ is a natural projection [1].

(b) Each reconfiguration event appears in **RG**, i.e.

$$(\forall \sigma_{i,j} \in \Sigma_{RE})(\exists q_s, q_d \in Q^{RG}, q_s \neq q_d)\ \delta^{RG}(q_s, \sigma_{i,j}) = q_d; \tag{4.12}$$

(c) From one mode, reconfiguration to another mode is possible only when the source state is in the source mode and the target state is in the target mode, i.e.

$$\begin{aligned}&(\forall \sigma_{i,j} \in \Sigma_{RE})(\forall q, q^{'} \in Q^{RG}, q \neq q^{'})\\ &[\delta^{RG}(q, \sigma_{i.j}) = q^{'} \Rightarrow (q \models P_i^{RG} \wedge q^{'} \models P_j^{RG})];\end{aligned} \tag{4.13}$$

(iii) Reconfiguration can be operated back-and-forth without blocking, i.e.

$$(\forall \sigma_{i,j}, \sigma_{j,i} \in \Sigma_{RE})\ [\delta^{RG}(q, \sigma_{i,j})! \Rightarrow (\exists s \in L(RG))\delta^{RG}(q, \sigma_{i,j} s \sigma_{j,i})!]; \tag{4.14}$$

(iv) Every reachable state of $\mathbf{Mode}_i$ in **RG** is also coreachable, which means that each mode is nonblocking in **RG**, i.e.

$$\begin{aligned}&(\forall i \in 1, ..., n)(\forall q \models P_i^{RG})(\exists \sigma_i \in \Sigma_{MIE})(\exists s \in \Sigma_i^*)\\ &[\delta^{RG}(q_o^{RG}, \sigma_i s) = q \Rightarrow (\exists q_m \in Q_m^{RG})(\exists s^{'} \in \Sigma_i^*)(\delta^{RG}(q, s^{'}) = q_m \wedge q_m \models P_i^{RG})];\end{aligned} \tag{4.15}$$

(v) Every state except for the initial state in the reconfiguration plant satisfies exactly one SMP, i.e.

$$(\forall q \in Q^{RG}, q \neq q_o^{RG})(\exists i \in 1, ..., n)(\forall j \neq i)\ [q \models P_i^{RG} \wedge q \not\models P_j^{RG}]. \tag{4.16}$$

$\diamond$

Remark. The requirement (v) implies that for the state set corresponding to the SMP $Q_i^{RG} := \{q \in Q^{RG} | P_i^{RG}(q) = 1\} \subseteq Q^{RG}$, it is true that $Q^{RG} = Q_1^{RG} \dot{\cup} ... \dot{\cup} Q_n^{RG} \dot{\cup} \{q_o^{RG}\}$.

All assumptions needed in the proposed approach are summarized below. Not all of them may be necessary in solving practical problems, but they are necessary in establishing the theory.

Definition 17. [Assumptions]. For a multiple reconfiguration problem, there are six assumptions.

(i) The initial state of each plant component is in every mode of the system, i.e.

$$(\forall i = 1, ..., n)(\forall k = 1, ..., h)\ q^k_{o,i} = q^k_o; \tag{4.17}$$

(ii) Every mode in each component DES is both reachable and coreachable by itself, i.e.

$$\begin{aligned}&(\forall i \in 1, ..., n)(\forall k \in 1, ..., h)\ [\Sigma_i \cap \Sigma^k \neq \emptyset \Rightarrow \\ &(\forall q \in Q^k_i)(\exists q^{'} \in Q^k_{m,i})(\exists s, s^{'} \in (\Sigma_i \cap \Sigma^k)^*)\delta^k_i(q^k_o, s) = q \wedge \delta^k_i(q, s^{'}) = q^{'}]\end{aligned} \tag{4.18}$$

and

$$(\forall i \in 1, ..., n)(\forall r \in 1, ..., h)\ [\Sigma_i \cap \Sigma^r = \emptyset \Rightarrow q^r_o \in Q^r_m] \tag{4.19}$$

(iii) For each pair of modes, the exit events for distinct plant components must be distinct, i.e.

$$(\forall i, j \in 1, ..., n, i \neq j)(\forall k, r = 1, ..., h, k \neq r)\ \Sigma^k_{ETE,i,j} \cap \Sigma^r_{ETE,i,j} = \emptyset; \tag{4.20}$$

(iv) For each pair of modes, the entry events for distinct plant components must be distinct, i.e.

$$(\forall i, j \in 1, ..., n, i \neq j)(\forall k, r = 1, ..., h, k \neq r)\ \Sigma^k_{EYE,i,j} \cap \Sigma^r_{EYE,i,j} = \emptyset; \tag{4.21}$$

(v) An event cannot serve as both exit event and entry event with respect to the same pair of modes in different components, i.e.

$$(\forall i, j \in 1, ..., n, i \neq j)\ \Sigma^G_{ETE,i,j} \cap \Sigma^G_{EYE,i,j} = \emptyset. \tag{4.22}$$

(vi) Each event in the plant must belong to at least one of the modes, i.e.

$$\bigcup_{i=1,...,n} \Sigma_i = \Sigma \tag{4.23}$$

◇

4.3 Multiple Reconfiguration Approach

In the last section the multiple reconfiguration problem is formally defined. There are also six important assumptions for the problem. This section aims to solve Problem 4 within the scope of the six assumptions and to prove the correctness of the proposed approach.

4.3.1 Multiple Reconfiguration Specification

A new structured multiple reconfiguration specification (MRS) can be used to represent the multiple reconfiguration behavior and solve Problem 4. The MRS cannot be constructed directly, since the conditions for different pairs of modes are very complicated to model by constructing only one reconfiguration specification. Instead, we decompose the MRS into one core multiple reconfiguration specification and several extra multiple reconfiguration specifications. The formal definition of the core multiple reconfiguration specification (CMRS) is as follows. The CMRS includes all details for the multiple reconfiguration task, such as event alphabets of different modes, and the RE and MIE.

Definition 18. [Core Multiple Reconfiguration Specification (CMRS)]. Denote by $\mathbf{G}$ the plant DES whose event alphabet is Σ. $\mathbf{Mode}_1, ..., \mathbf{Mode}_n$ are n different modes of $\mathbf{G}$ distinguished by event alphabets $\Sigma_1, ..., \Sigma_n$. The *core multiple reconfiguration specification* is defined as the DES

$$\mathbf{R}_c := (Q^{R_c}, \Sigma^{R_c}, \delta^{R_c}, q_o^{R_c}, Q_m^{R_c})$$

where

- $Q^{R_c} := \{q_o^{R_c}\} \dot{\cup} Q_M^{R_c}$, in which $Q_M^{R_c} := \{q_1, ..., q_n\}$ and $|Q_M^{R_c}| = n$;
- $\Sigma^{R_c} := \Sigma \dot{\cup} \Sigma_{RE} \dot{\cup} \Sigma_{MIE}$;
- $\delta^{R_c}(q_o^{R_c}, \sigma) := q_i \quad$ if $\sigma = \sigma_i \in \Sigma_{MIE}$;

  $$\delta^{R_c}(q_i, \sigma) := \begin{cases} q_i & \text{if } \sigma \in \Sigma_i; \\ q_j & \text{if } \sigma = \sigma_{i,j} \in \Sigma_{RE}; \end{cases}$$

- $q_o^{R_c}$ is the initial state of $\mathbf{R}_c$;
- $Q_m^{R_c} = Q_M^{R_c} = Q^{R_c} - \{q_o^{R_c}\}$.

◇

Remark. The structure of this CMRS is the same as the RS in Figure 3.10, so the CMRS has the same shortcoming as the RS (see page 24). The shortcoming is the unsuitable reconfiguration defined at some state belonging to only one mode.

The example shown in Figure 4.4 illustrates the new core multiple reconfiguration specification. In the example, there are three modes distinguished by $\Sigma_1 = \{\alpha, \beta, \mu\}, \Sigma_2 = \{\alpha, \lambda, \sigma\}, \Sigma_3 = \{\gamma, \tau\}$. The MIE σ_i is for $\mathbf{Mode}_i$ where $i = 1, 2, 3$. The RE $\sigma_{j,k}$ represents the reconfiguration from $\mathbf{Mode}_j$ to $\mathbf{Mode}_k$, where $j, k = 1, 2, 3$.

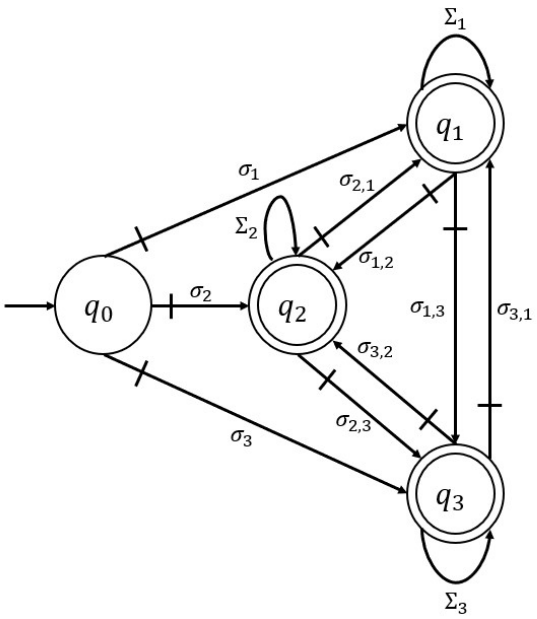

Figure 4.4: Toy example illustrating CMRS

In order to overcome the problem about unsuitable reconfiguration, some extra multiple reconfiguration specifications (EMRS) with simple structures are required.

The new structured EMRS can be used to represent the appropriate bidirectional reconfiguration behavior. The hope is that combining several EMRS for different reconfiguration events is sufficient to deal with every reconfiguration event and solve the multiple reconfiguration problem. The formal definition of EMRS is as follows.

Definition 19. [Extra Multiple Reconfiguration Specification (EMRS)]. Denote by **G** the plant DES formed by synchronization. $\mathbf{Mode}_1, ..., \mathbf{Mode}_n$ are n different modes of **G** distinguished by event alphabets $\Sigma_1, ..., \Sigma_n$. The *extra multiple reconfiguration specification* with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$, $(\forall i, j \in 1, ..., n, i < j)$ is defined, if $\Sigma^G_{ETE,i,j} \neq \emptyset$, as the DES

$$\mathbf{R}_{i,j} := (Q^{R_{i,j}}, \Sigma^{R_{i,j}}, \delta^{R_{i,j}}, q_o^{R_{i,j}}, Q_m^{R_{i,j}})$$

where

- $Q^{R_{i,j}} := \{q_0^{R_{i,j}}, ..., q_{k_{i,j}}^{R_{i,j}}\}$ where $k_{i,j} = |\{r | \Sigma^r_{ETE,i,j} \neq \emptyset\}|$.
- $\Sigma^{R_{i,j}} := \{\sigma_{i,j}, \sigma_{j,i}\} \dot\cup \Sigma^G_{ETE,i,j} \dot\cup \Sigma^G_{EYE,i,j}$;
- $\delta^{R_{i,j}}(q_0^{R_{i,j}}, \sigma) := q_0^{R_{i,j}} \quad$ if $\sigma = \sigma_{i,j}, \sigma_{j,i} \in \Sigma_{RE}$;

  $\delta^{R_{i,j}}(q_r^{R_{i,j}}, \sigma) := \begin{cases} q_{r+1}^{R_{i,j}} & \text{if } \sigma \in \Sigma^G_{ETE,i,j}, 0 \leq r < k_{i,j}; \\ q_{r-1}^{R_{i,j}} & \text{if } \sigma \in \Sigma^G_{EYE,i,j}, 0 < r \leq k_{i,j}; \end{cases}$
- $q_o^{R_{i,j}} := q_0^{R_{i,j}}$ is the initial state;
- $Q_m^{R_{i,j}} := Q^{R_{i,j}}$.

$\diamond$

Remark. If $\Sigma^G_{ETE,i,j} = \emptyset$, then every state of the plant satisfies both P_i^G and P_j^G, so the RE $\sigma_{i,j}$ and $\sigma_{j,i}$ are eligible to occur at every state of the plant. Thus, there is no need to add an extra specification to regulate $\sigma_{i,j}$.

The $q_0^{R_{i,j}}$ is the state at which $\sigma_{i,j}$ and $\sigma_{j,i}$ are defined. If no ETE with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ occurs, then the RE $\sigma_{i,j}$ and $\sigma_{j,i}$ are always eligible to occur. If a number of ETE occur, then after the same number of occurrences of EYE, the EMRS will return to the initial state, where the RE $\sigma_{i,j}$ and $\sigma_{j,i}$ are eligible to occur.

Note that $\Sigma^G_{ETE,i,j}$ may contain events that are not in Σ_i to avoid blocking in the reconfiguration plant. The correctness of the EMRS will be proved later.

The transitions regarding $q_1^{R_{i,j}}, ..., q_{k_{i,j}}^{R_{i,j}}$, $\Sigma^G_{ETE,i,j}$ and $\Sigma^G_{EYE,i,j}$ simply mean that a reconfiguration occurs only when every plant component of the system has returned to the inner part.

The following Figure 4.5 demonstrates the notion of EMRS.

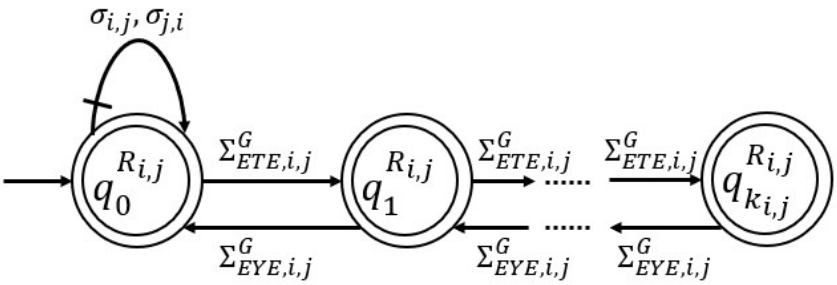

Figure 4.5: Illustration of EMRS with respect to a RE $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$

Given the assumptions (ii), (iii) and (iv) stated in Definition 17, an ETE in a plant component corresponds to at least one EYE in the same component. However, the reverse statement is not always true. Consider the example in Figure 3.17 (see page 40). No ETE in this example has a corresponding EYE, but the system still satisfies all the six assumptions. In this case, if one of the events in $\Sigma^G_{ETE,i,j}$ occurs, then reconfiguration from $\mathbf{Mode}_i$ to $\mathbf{Mode}_j$ (or from $\mathbf{Mode}_j$ to $\mathbf{Mode}_i$) can never occur, so reconfiguration occurs only before the ETE occurs. This consequence is due to the structure of the system itself but not the proposed approach.

If one ETE corresponds to more than one, for example two EYE, the two EYE cannot both occur in the component unless the ETE occurs twice, since the plant component cannot enter the inner part with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ twice with only one exit. Similarly, the reverse direction is also true. As a result, if in the system there is only one ETE and two EYE, there will be only one extra state ($q_1^{R_{i,j}}$) needed in the EMRS. Therefore, for each of the plant components whose event alphabets include at least one ETE with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$, one extra state is needed for the EMRS $\mathbf{R}_{i,j}$. Namely, $|Q^{R_{i,j}}| = |\{r|\Sigma^r_{ETE,i,j} \neq \emptyset\}| = k_{i,j}$.

This EMRS is in a bidirectional fashion since each EMRS only deals with a pair of RE for bidirectional reconfiguration. In total, for a system with n modes, there are at most $n * (n-1)/2$ EMRS required. The EMRS defined above and the CMRS defined before can be synchronized to generate the multiple reconfiguration specification (MRS) to solve Problem 4 under the six assumptions in Definition 17.

Definition 20. [Multiple Reconfiguration Specification (MRS)]. Denote by $\mathbf{G}$ the plant

DES, and $\mathbf{Mode}_1, ..., \mathbf{Mode}_n$ the n modes. The *multiple reconfiguration specification* is defined as the synchronous product

$$\mathbf{R} = \mathbf{R}_c||\mathbf{R}_{1,2}||...||\mathbf{R}_{1,n}||\mathbf{R}_{2,3}||...||\mathbf{R}_{n-1,n} = (Q^R, \Sigma^R, \delta^R, q_o^R, Q_m^R)$$

where $\Sigma^R = \Sigma^{R_c}$.

◇

Remark. There are at most $n * (n-1)/2$ EMRS (i.e. $(\forall i, j \in 1, ..., n, i < j)$ $\mathbf{R}_{i,j}$) that need to be synchronized.

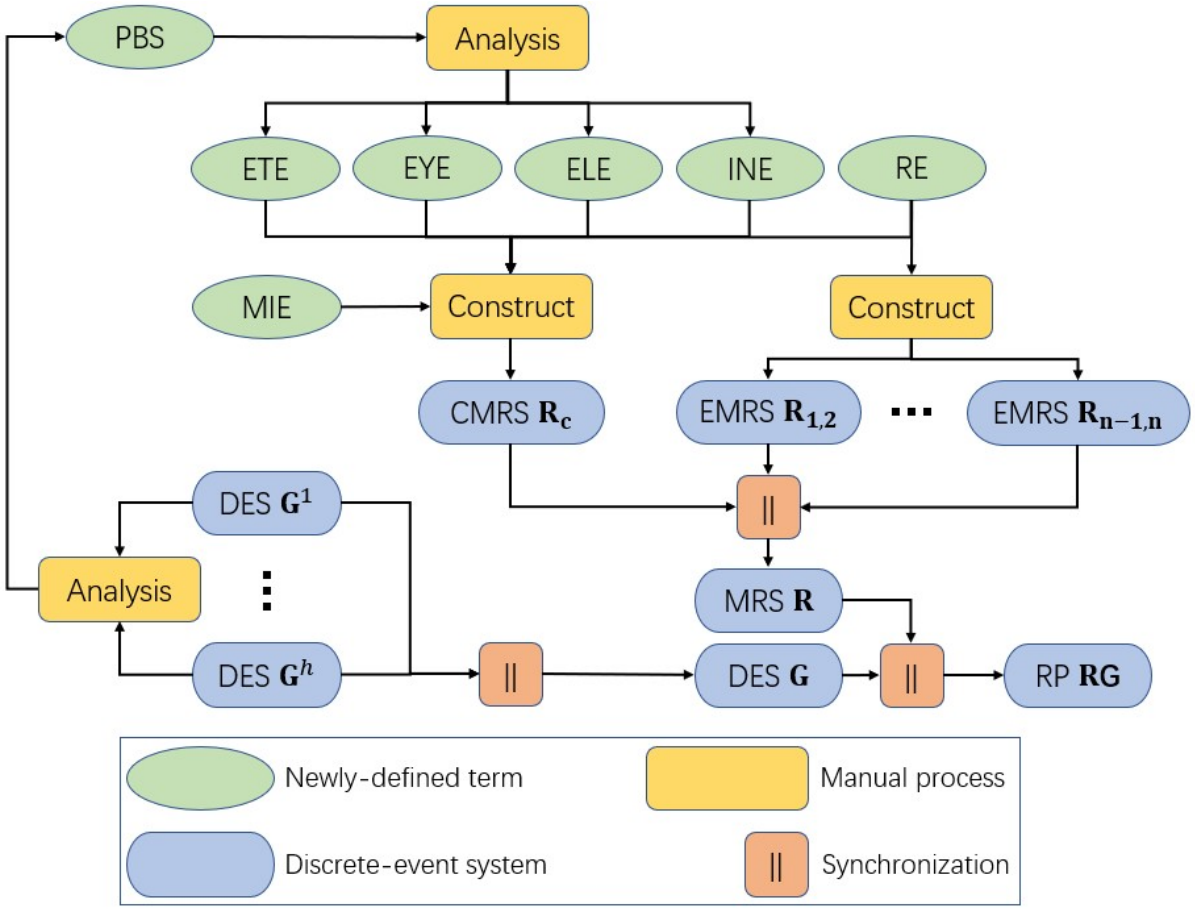

Figure 4.6: Monolithic multiple reconfiguration of discrete-event systems

It turns out that the multiple reconfiguration specification constructed according to Definition 20 solves Problem 4 through the synchronization with the plant DES $\mathbf{G}$. The resulting synchronous product is the reconfiguration plant that incorporates the multiple reconfiguration mechanism.

The main procedure of the monolithic multiple reconfiguration approach is illustrated in Figure 4.6.

Informally, we discuss why the reconfiguration plant constructed from the proposed MRS satisfies the five requirements in Problem 4.

(i) The reconfiguration plant can start in any mode.

With the mode initialization events, the MRS can start in any mode. Since none of MIE is in Σ and events in Σ are defined at other states in MRS, the RP can start in any mode through a transition labeled by a MIE from the initial state.

(ii) Reconfiguration from $\mathbf{Mode}_i$ to $\mathbf{Mode}_j$ can be done when all plant components are at public states with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$. From one mode, the reconfiguration to another mode is possible only when the current state belongs to the two modes.

Note that in the proposed EMRS, a RE from $\mathbf{Mode}_i$ to $\mathbf{Mode}_j$ is not allowed to occur after the occurrence of an ETE, unless the mode comes back to the inner part via an EYE. Intuitively, this statement makes sense since a reconfiguration should always occur at a public state that is shared by the two modes. Otherwise, after the reconfiguration, some events of the destination mode would be blocked.

(iii) Reconfiguration can be operated back-and-forth without blocking.

This is true. According to CMRS, the backward reconfiguration can even be operated at the target state of the forward reconfiguration after a few transitions in the target mode.

(iv) Nonblocking needs to be guaranteed for each mode separately.

In each plant component, nonblocking is guaranteed for each mode separately. Since the MRS includes all events in the plant DES and results in no new blocking issues, it is true that nonblocking is guaranteed for each mode separately in the RP.

(v) Every state except for the initial state in the MRS satisfies only one SMP.

In the reconfiguration plant, events in different modes are defined at different states. Thus, any state that satisfies n AMP in the plant will be divided into n states in

the reconfiguration plant, and each of them satisfies only one SMP. For a state satisfying fewer than n AMP in the plant, the relevant events will be defined at states corresponding to m modes. Hence, that state will be divided into m states in the reconfiguration plant, where each of them satisfies only one SMP.

All formal results and proofs of correctness are provided in Appendix B.

4.3.2 Comparison with the Bidirectional Reconfiguration Approach

The bidirectional reconfiguration approach in Chapter 3 is a special case of the multiple reconfiguration approach in this chapter. When there are only two modes, say $\mathbf{Mode}_1$ and $\mathbf{Mode}_2$, then a public state with respect to $\mathbf{Mode}_1$ and $\mathbf{Mode}_2$ essentially satisfies all AMP in the system, so it can also be considered as a strictly public state. As long as the system is not at a public state with respect to $\mathbf{Mode}_1$ and $\mathbf{Mode}_2$, the reconfiguration event is not eligible to occur; then the exit event (entry event) with respect to $\mathbf{Mode}_1$ and $\mathbf{Mode}_2$ is the same as the strictly exit event (strictly entry event). In fact, for a system with only two modes, the RG in Chapter 3 is the same as the RG in this chapter.

According to Theorem 5 (see page 136 Appendix B), we can see that the bidirectional reconfiguration approach in Chapter 3 is indeed a special case of the multiple reconfiguration approach in Chapter 4. The advantage of the bidirectional reconfiguration approach is that it uses only one reconfiguration specification, but its disadvantage is that it lacks some flexibility and expressiveness. When dealing with systems with only two modes, either of the two approaches works well.

For clarity, the following example illustrates the equivalence of $\mathbf{RG}_B$ and $\mathbf{RG}_M$ when the system has two modes.

Consider the first example in Section 3.4. We have presented the results based on the BRS. For simplicity, in this chapter we only give the results based on the MRS.

In the system, $\Sigma^G_{ETE,1,2} = \{3, 5, 7, 11\}, \Sigma^G_{EYE,1,2} = \{4, 6, 9, 13\}, \Sigma^G_{ELE,1,2} = \{8, 12\}$ and $\Sigma^G_{INE,1,2} = \{1\}$. Then $k_{1,2} = |\{r | \Sigma^r_{SETE} \neq \emptyset\}| = 4$. According to the definition of CMRS and EMRS, the CMRS $\mathbf{R}_C$ for this system is shown in Figure 4.7 and the EMRS $\mathbf{R}_{1,2}$ is shown in Figure 4.8.

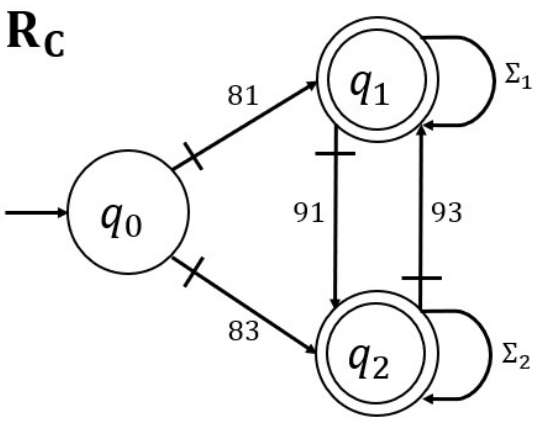

Figure 4.7: CMRS $\mathbf{R}_C$

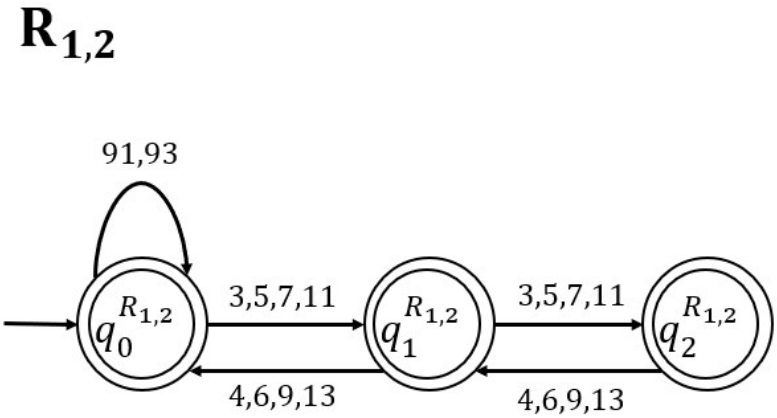

Figure 4.8: EMRS $\mathbf{R}_{1,2}$

In $\mathbf{R}_C$, $\mathbf{R}_{1,2}$, event $81 \in \Sigma_{MIE}$ is the mode initialization event for $\mathbf{Mode}_1$, event $83 \in \Sigma_{MIE}$ is the mode initialization event for $\mathbf{Mode}_2$, event $91 \in \Sigma_{RE}$ is the reconfiguration event from $\mathbf{Mode}_1$ to $\mathbf{Mode}_2$, and event $93 \in \Sigma_{RE}$ is the reconfiguration event from $\mathbf{Mode}_2$ to $\mathbf{Mode}_1$. This is consistent with section 3.4.

We can then compute

$$\mathbf{RG}_M = \mathbf{Sync}(\mathbf{M}_1, \mathbf{M}_2, \mathbf{R}_C, \mathbf{R}_{1,2})$$
$$\mathbf{true} = \mathbf{Isomorph}(\mathbf{RG}_M, \mathbf{RG}_B; \mathbf{identity})$$

The resulting reconfiguration plant $\mathbf{RG}_M$ can be computed by the supervisory controller synthesis software TCT, and is shown in the following Figure 4.9.

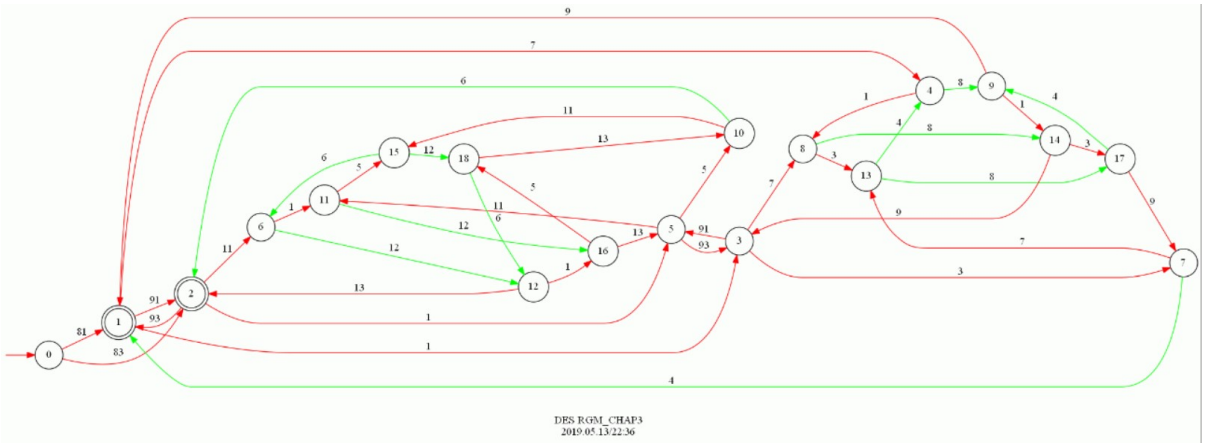

Figure 4.9: Reconfiguration plant $\mathbf{RG}_M$

According to TCT, $\mathbf{RG}_M$ is isomorphic to $\mathbf{RG}_B$ in section 3.4.

Note that though $\mathbf{RG}_M = \mathbf{RG}_B$, $\mathbf{R}_B$ in section 3.4 is different from the $\mathbf{R}_M = \mathbf{R}_C||\mathbf{R}_{1,2}$. The synchronous product $\mathbf{R}_M$ obtained from TCT is shown in Figure 4.10.

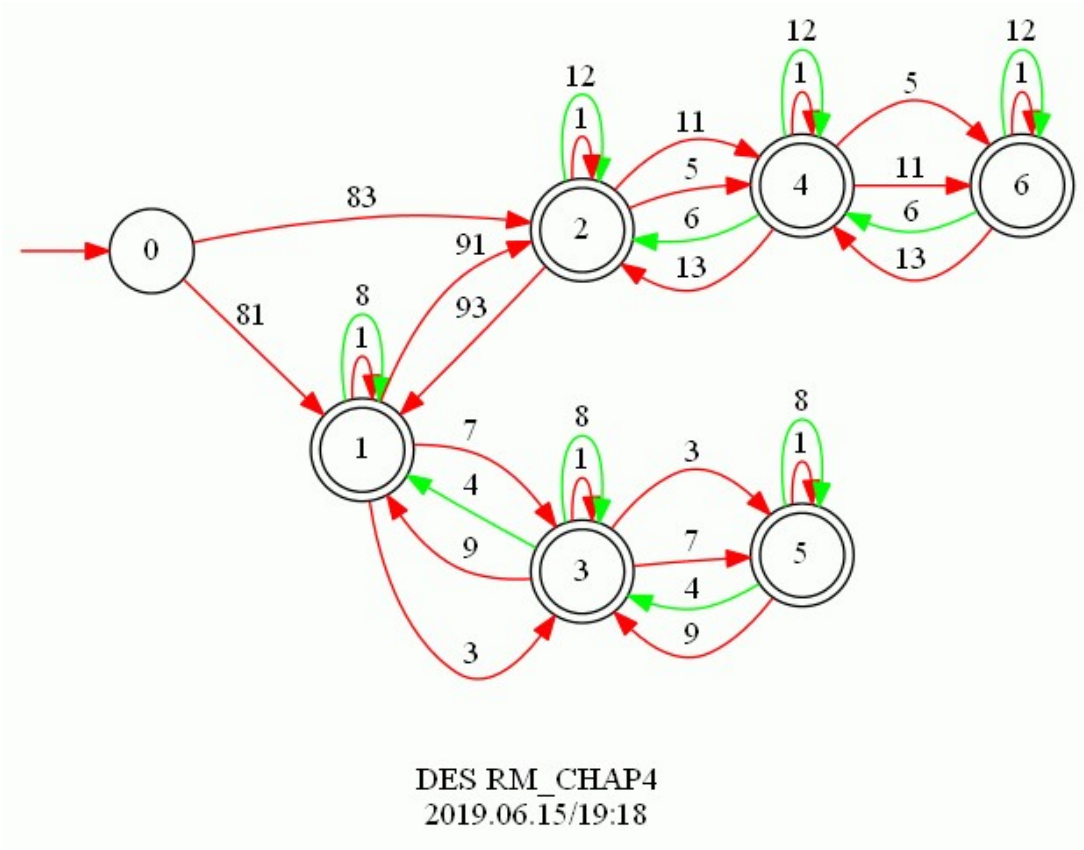

Figure 4.10: Reconfiguration specification $\mathbf{R}_M$

The difference between $\mathbf{R}_M$ and $\mathbf{R}_B$ for this system is that there is some external event defined at state 1 and state 2 of $\mathbf{R}_M$. But in $\mathbf{R}_B$ (see Figure 3.20), there is no strictly external event defined at state q_1^0 and q_2^0. In fact, this difference is eliminated by the synchronization with the plant $\mathbf{G}$, since a (strictly) external event is not eligible to occur before any (strictly) exit event occurs. Therefore, the redundant external events in MRS are harmless.

4.4 Behavioral Specifications

In order to make the proposed approach useful in real systems, reconfiguration events are required to be forcible [1] to preempt any other events in Σ. For this bring in new timeout events. See Chapter 3 of [1] for a guide to how the forcing is set up using timeout events. In this approach, adding timeout events should be done in the reconfiguration plant. Consider each state at which RE is defined. For each uncontrollable event that is defined at the state, timeout events can be added ahead of the uncontrollable event. In fact, we don't need to add timeout events until we implement this reconfiguration approach in real systems. But usually, we don't need to handle them in the early stages of the procedure.

In contrast, attention should often be paid to the behavioral specification $\mathbf{B}$ that only events in Σ are involved in. A behavioral specification (BS) normally represents a logical requirement that has to be met by the plant. Namely, a behavioral specification specifies a particular behavior of the system. Supervisory control theory can synthesize a supervisor to control the plant under the constraints represented by the behavioral specification.

Without considering the RS, we can compute

$$\mathbf{ALLG} = \mathbf{Allevents}(\mathbf{G.DES}) \tag{4.24}$$

$$\mathbf{BSPEC} = \mathbf{Sync}(\mathbf{ALLG}, \mathbf{B}) \tag{4.25}$$

$$\mathbf{SUP} = \mathbf{Supcon}(\mathbf{G}, \mathbf{BSPEC}) \tag{4.26}$$

to obtain the DES **SUP** for the system under supervision. The "**Allevents**" operation is to get the corresponding full alphabet. But when taking RS into consideration, a slightly

different procedure is required. The new procedure is:

$$\mathbf{RG} = \mathbf{Sync}(\mathbf{G}^1, ..., \mathbf{G}^n, \mathbf{R}) \tag{4.27}$$

$$\mathbf{ALLRG} = \mathbf{Allevents}(\mathbf{RG.DES}) \tag{4.28}$$

$$\mathbf{BSPEC} = \mathbf{Sync}(\mathbf{ALLRG}, \mathbf{B}) \tag{4.29}$$

$$\mathbf{RSUP} = \mathbf{Supcon}(\mathbf{RG}, \mathbf{BSPEC}) \tag{4.30}$$

Here **RSUP** represents that the system operates initially in any mode, then optionally switches to any other modes, where the behavioral operation will remain. The states of **RSUP** at which the mode switch is possible, are those at which the controllable event RE is enabled. It is needed that **RSUP** satisfy all the five requirements in the problem definition of multiple reconfiguration. Say **RSUP** is the reconfiguration supervisor.

Theorem 6 in Appendix B (see page 139) indicates that the multiple reconfiguration approach (and the bidirectional reconfiguration approach) are compatible with supervisory control theory, which gives the proposed approach greater practicality. The proof of this theorem is also in Appendix B. An illustrative example will be given in Section 4.7.

4.5 Guaranteed Reachability

In an application, reconfiguration might be demanded at any reachable state q_s in **RG** or **RSUP**, depending on the need of the user. It is then required to construct at least one path in $\mathbf{Mode}_i$ (suppose that $q_s \models P_i^{RG}$ or $q_s \models P_i^{RSUP}$) from q_s to a state q_t (perhaps the "nearest") where a RE $\sigma_{i,j}$ (j is arbitrary) is defined, a path that is feasible in terms of the event disablement and forcible preemption available to the user. This is the *guaranteed reachability* (GR) problem described in [1].

In [1], the GR requirement is formulated in a dynamic programming [53] fashion. In this section, a different backtracking algorithm based on dynamic programming formulation is proposed and implemented in C++. In the dynamic programming formulation, for $k = 0, 1, 2, ...$, define the state subset

$$Q_k := \{q \in Q | q_t \text{ is GR from } q \text{ in } k \text{ or fewer transitions}\} \tag{4.31}$$

By definition $Q_0 := \{q_t\}$. Then also $q_t \in Q_1$. For an arbitrary state $q \neq q_t$, clearly $q \in Q_1$ if and only if there exists $\sigma \in \Sigma$ such that $\delta(q, \sigma) \in Q_0$ and, for all $\sigma' \in \Sigma$ such that $\delta(q, \sigma')!$ and $\delta(q, \sigma') \notin Q_0$, it is required that σ' can be disabled, namely $\sigma' \in \Sigma_c$. Inductively,

$$\begin{aligned} Q_k = Q_{k-1} \cup \{q \in Q | (\exists \sigma \in \Sigma)\ \delta(q, \sigma) \in Q_{k-1} \wedge \\ (\forall \sigma' \in \Sigma)\ [(\delta(q, \sigma')! \wedge \delta(q, \sigma') \notin Q_{k-1}) \Rightarrow \sigma' \in \Sigma_c]\} \end{aligned} \tag{4.32}$$

Define $Q_{GR} := \bigcup\{Q_k | k \geq 0\}$ where the union is finitely convergent at $n = |Q|$. Thus q_t is guaranteed reachable from q_s, i.e. $GR(q_s, q_t)$ iff $q_s \in Q_{GR}$.

A stronger result can be obtained if, in addition to controllable event disablement, event forcing is optionally employed as well. Then the definition becomes

$$\begin{aligned} Q_k = Q_{k-1} \cup \{q \in Q | (\exists \sigma \in \Sigma)\ [\delta(q, \sigma) \in Q_{k-1}) \wedge \\ (\sigma \text{ is locally forcible at } q \vee \\ ((\forall \sigma' \in \Sigma)\ (\delta(q, \sigma')! \wedge \delta(q, \sigma') \notin Q_{k-1}) \Rightarrow \sigma' \in \Sigma_c)]\} \end{aligned} \tag{4.33}$$

For the definition of locally forcible event, please see example 3.8.4 in [1].

This formulation can be directly translated to a backtracking algorithm. The pseudo-code of the algorithm GR-Checking-1 is as follows. This GR-Checking-1 algorithm is used to evaluate the guaranteed reachability from q_s to q_t.

For each state q in the remaining state set Q_{rm}, the GR-Checking-1 algorithm checks all transitions whose source state is q, and evaluates whether to add q to the set Q_{GR}. If q is added to Q_{GR}, it will also be deleted from Q_{rm}. These states and the corresponding transitions8 are structured in a red-black tree [54]. The program will continue running until no more states can be added to Q_{GR} after all remaining states are traversed iteratively. As a result, Q_{GR} will contain all the states that are guaranteed to lead to q_t.

In Figure 4.11, we illustrate how the GR-Checking-1 works. Here Q_{GR} is a state set containing the target state q_t. Any state in Q_{GR} except for q_t is guaranteed to lead to q_t with help of event disablement and event forcing. We then analyze the guaranteed reachability from q_1, q_2, q_3, q_4 and q_s to q_t.

Algorithm 1: GR-Checking-1

Input: q_s, q_t, δ, Q;
Output: true, false;
Initialize the state set $Q_{GR} \leftarrow \{q_t\}$;
Initialize the state set $Q'_{GR} \leftarrow \{q_t\}$;
Initialize the state set $Q_{rm} \leftarrow Q - \{q_t\}$;
while $Q'_{GR} \neq Q_{GR}$ **do**
  $Q'_{GR} \leftarrow Q_{GR}$;
  for *each state* $q \in Q_{rm}$ **do**
    for *all transitions whose source state is* q **do**
      if $(\exists \sigma \in \Sigma)\ \delta(q, \sigma) \in Q_{GR}$ **then**
        if $(\sigma$ *is locally forcible at* $q) \vee$
        $(\forall \sigma' \in \Sigma)\ [(\delta(q, \sigma')! \wedge \delta(q, \sigma') \notin Q_{GR}) \Rightarrow \sigma' \in \Sigma_c]$ **then**
          $Q_{rm} \leftarrow Q_{rm} - \{q\}$;
          $Q_{GR} \leftarrow Q_{GR} \cup \{q\}$;
if $q_s \in Q_{GR}$ **then**
  return *true*;
else
  return *false*;

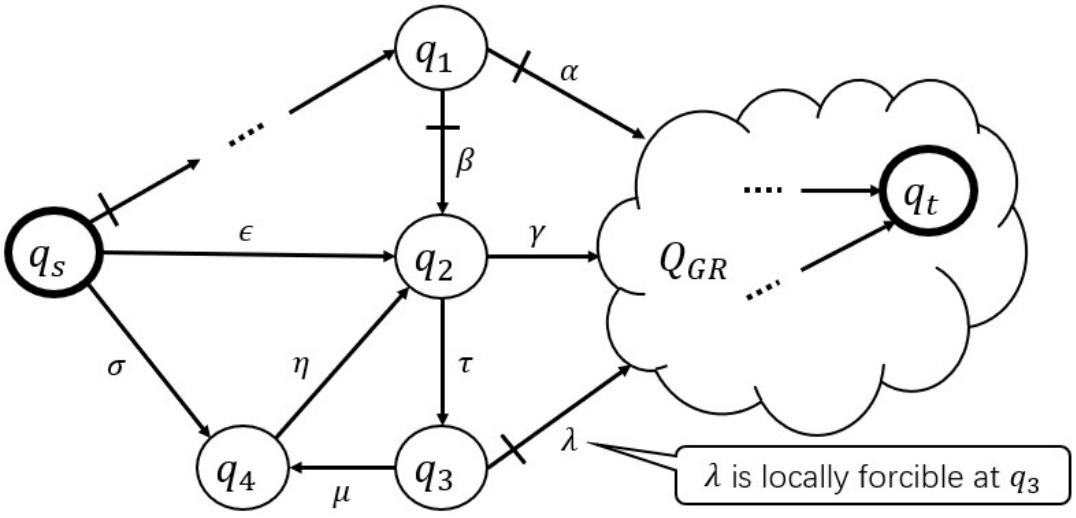

Figure 4.11: Illustration of GR-Checking-1 algorithm

We first analyze each state separately. For q_1, since q_1 can lead to a state in Q_{GR} via the controllable event α and the event β can be disabled, q_1 is guaranteed to lead to q_t. For q_2, since it may lead to q_3 by the uncontrollable event τ, it is not guaranteed to lead to q_t. For q_3, though λ is a controllable event, it is locally forcible at q_3, so q_3 is guaranteed

to lead to q_t. For q_4 and q_s, since they cannot directly lead to any state in Q_{GR} by an event, they are not guaranteed to lead to q_t.

However, if we consider them together, the results are different. Let's say the GR-Checking-1 visits those states in an order $q_1 \rightarrow q_3 \rightarrow q_2 \rightarrow q_4 \rightarrow q_s$. As we have analyzed above, q_1 and q_3 are guaranteed to lead to q_t, thus can be added to Q_{GR}. Then since q_2 can only lead to states in Q_{GR}, it is also guaranteed to lead to q_t. Similarly we can add q_4 and q_s to Q_{GR} in sequence. That's a typical iteration of the while loop of the GR-Checking-1 algorithm. In fact, the real visiting sequence of states might be different from the assumed one, but these five states will eventually be added into Q_{GR}.

Theorem 7 in Appendix B (see page 140) summarizes the time complexity of the GR-Checking-1 algorithm. The proof is also in Appendix B.

After the result of GR-Checking-1 is obtained, a further step could be to generate all the paths from q_s to q_t. Since the DES can also be considered as a directed graph with cycles, many existing algorithms in graph theory can be applied here. Generating all paths can be done by applying the Depth First Search algorithm in $O(L^2)$ time [54], where L is the number of states in Q_{GR}.

Another option is to generate the shortest paths. This can be done by the Dijkstra Algorithm in $O(L^2)$ time as well [54]. Thus, both conditions can be done in polynomial time with respect to the number of states in Q_{GR}.

It is obvious that the number of states in Q_{GR} determines how fast the computation would be. However, the GR-Checking-1 algorithm tends to generate a large state set Q_{GR} that contains all states that are guaranteed to lead to the target state q_t. The computation could be much more efficient if the algorithm generated a smaller number of states that are sufficient to deduce the guaranteed reachability from q_s to q_t.

The difference between the GR-Checking-1 and the GR-Checking-2 algorithm is that the GR-Checking-2 will terminate if the q_s has been added to Q_{GR}, instead of looking for all states that are guaranteed to lead to q_t. Thus, the Q_{GR} will include a smaller number of states that are sufficient to prove the guaranteed reachability from q_s to q_t, if they satisfy the GR, otherwise, the Q_{GR} will contain all states that are guaranteed to lead to q_t. Therefore, if the q_s is guaranteed to lead to q_t, with the Q_{GR} obtained from GR-Checking-2 algorithm, we are able to find the length of the shortest paths, but

Algorithm 2: GR-Checking-2

```
Input: q_s, q_t, δ, Q;
Output: true, false;
Initialize the state set Q_GR ← {q_t};
Initialize the state set Q'_GR ← {q_t};
Initialize the state set Q_rm ← Q − {q_t};
while Q'_GR ≠ Q_GR do
    Q'_GR ← Q_GR;
    for each state q ∈ Q_rm do
        for all transitions whose source state is q do
            if (∃σ ∈ Σ) δ(q, σ) ∈ Q_GR then
                if (σ is locally forcible at q) ∨
                 (∀σ' ∈ Σ) [(δ(q, σ')! ∧ δ(q, σ') ∉ Q_GR) ⇒ σ' ∈ Σ_c] then
                    Q_rm ← Q_rm − {q};
                    Q_GR ← Q_GR ∪ {q};
                    if q = q_s then
                        return true;
return false;
```

may fail to find all shortest paths. To find all shortest paths, the Q_{GR} obtained from GR-Checking-1 is required.

Note that the worst-case time complexity of the GR-Checking-2 is the same as the GR-Checking-1, especially when the q_t is not GR from q_s. However, the average time complexity of the GR-Checking-2 is much higher than the GR-Checking-1, since the GR-Checking-2 may terminate relatively quickly.

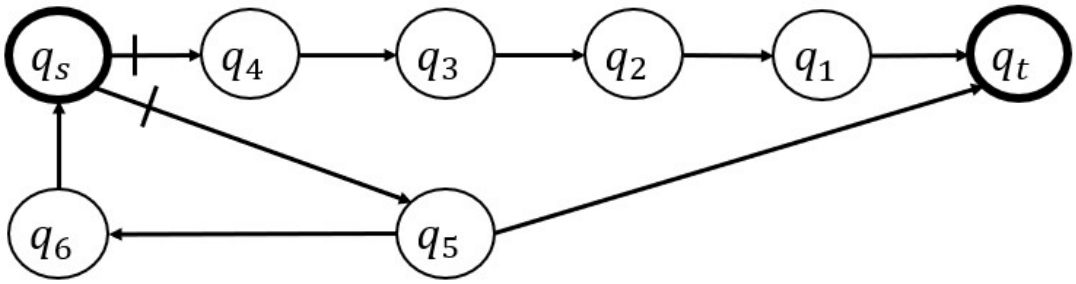

Figure 4.12: Illustration of GR-Checking-2 algorithm

In practice, if the user plans to investigate guaranteed reachability, say finding the shortest paths or finding all paths, the GR-Checking-1 is the algorithm to use. Instead, if the user needs an algorithm to investigate the guaranteed reachability as fast as possible, then

GR-Checking-2 is the best choice. Note that the Q_{GR} obtained from the GR-Checking-2 algorithm cannot be used to generate the shortest paths. For example, in Figure 4.12, all the states except for q_t are guaranteed to lead to q_t. Obviously the shortest path is $q_s \rightarrow q_5 \rightarrow q_t$. However, according to the GR-Checking-2, suppose that the order of visiting states is $q_1, q_2, q_3, q_4, q_s, q_5, q_6$, the state q_1, q_2, q_3, q_4, q_s would be added to Q_{GR}, then the algorithm would terminate. Then the shortest path according to the obtained Q_{GR} is $q_s \rightarrow q_4 \rightarrow q_3 \rightarrow q_2 \rightarrow q_1 \rightarrow q_t$, which has length five.

Therefore, if the user plans to find the shortest paths, the GR-Checking-1 is better. In summary, the GR-Checking algorithms provide the users with a reference for the guaranteed reachability and a series of possible operations to lead to the target state from the source state. The two algorithms are summarized in Table 4.1.

Table 4.1: Guaranteed Reachability Checking Algorithms

Algorithm	**Speed**	**GR**	**Shortest Paths**	**All Paths**
GR-Checking-1	Slow	Yes	Yes	Yes
GR-Checking-2	Fast	Yes	No	No

4.6 Triggering Behavior

The resulting reconfiguration plant and the supervisory controller demonstrate all possible reconfiguration behavior that is legal. However, the possible occurrence of back-and-forth behavior such as $< \sigma_{1,2}, \sigma_{2,1}, \sigma_{1,2}, \sigma_{2,1}, ... >$ is usually not desired. If the user makes a mode change $< \sigma_{1,2} >$ labeling $\mathbf{Mode}_1 \rightarrow \mathbf{Mode}_2$ them, once in $\mathbf{Mode}_2$, the event $< \sigma_{2,1} >$ coding the reverse reconfiguration $\mathbf{Mode}_2 \rightarrow \mathbf{Mode}_1$ is expected to be disabled unless or until it is "wanted" by the user, typically at some later stage. Thus, the sequence $< \sigma_{1,2}, \sigma_{2,1} >$ should be replaced by $< \sigma_{1,2}, s, \sigma_{2,1} >$ where $s \in \Sigma_2^*$ is a trigger string meaning that "If this sequence occurs, then the reconfiguration from $\mathbf{Mode}_2$ back to $\mathbf{Mode}_1$ is in demand". This special trigger sequence can be represented in a specification DES to help coordinate the system.

This new specification is called *trigger specification* (TS). The TS comprises both RE and events in Σ, so it will affect the reconfiguration behavior according to the user's

demand. By applying supervisory control theory, "**Supcon**" can generate a nonblocking, controllable, and maximally permissive close-loop behavior under supervision. Given the TS **T**, usually the procedure is as follows.

$$\mathbf{RG} = \mathbf{Sync}(\mathbf{G}^1, ..., \mathbf{G}^n, \mathbf{R}) \tag{4.34}$$

$$\mathbf{ALLRG} = \mathbf{Allevents}(\mathbf{RG.DES}) \tag{4.35}$$

$$\mathbf{BSPEC} = \mathbf{Sync}(\mathbf{ALLRG}, \mathbf{B}) \tag{4.36}$$

$$\mathbf{RSUP} = \mathbf{Supcon}(\mathbf{RG}, \mathbf{BSPEC}) \tag{4.37}$$

$$\mathbf{TSPEC} = \mathbf{Sync}(\mathbf{ALLRG}, \mathbf{T}) \tag{4.38}$$

$$\mathbf{TRSUP} = \mathbf{Supcon}(\mathbf{RSUP}, \mathbf{TSPEC}) \tag{4.39}$$

where **B** is a behavioral specification. The **TRSUP** represents the controlled behavior with respect to the behavioral specification and the trigger specification. Say **TRSUP** is the reconfiguration supervisor with trigger.

In fact, the foregoing behavioral specifications, trigger specifications, and guaranteed reachability checking can all be dealt with in both the bidirectional and multiple reconfiguration approaches. The following Figure 4.13 illustrates this fact.

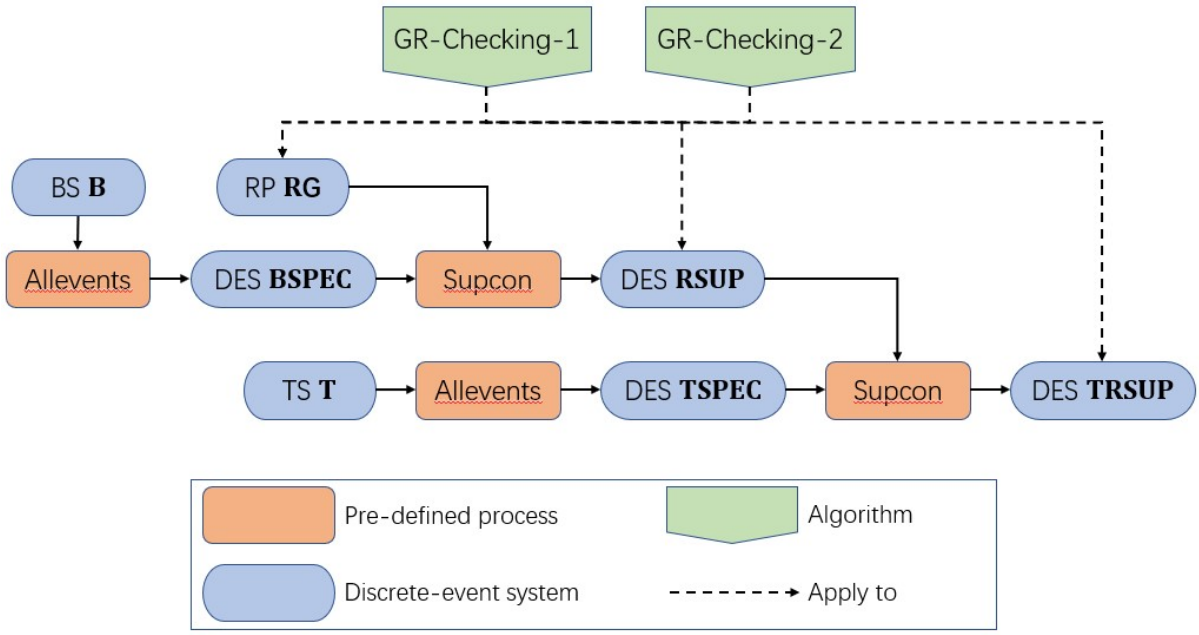

Figure 4.13: Behavioral specification, guaranteed reachability, and trigger

4.7 Example

4.7.1 Toy Example

The objective of this example is to explore change of mode. As the Figure 4.14, 4.15 shown below, the plant DES is $\mathbf{G} = \mathbf{M1}||\mathbf{M2}$. The two plant components are the DES for two machines. Consider the plant $\mathbf{G}$ as a small factory. There are two machines and three operating modes $\mathbf{Mode}_1, \mathbf{Mode}_2$, and $\mathbf{Mode}_3$ that are distinguished by $\Sigma_1 = \{1, 2, 3, 5, 7, 8, 13, 14\}$, $\Sigma_2 = \{1, 2, 3, 5, 7, 8, 23, 24\}$, and $\Sigma_3 = \{1, 2, 33\}$. Note that the event alphabets of distinct modes are not necessarily pairwise disjoint.

The factory is able to handle two tasks. Task 1 needs to be finished in $\mathbf{Mode}_1$. But when the workload is sufficiently high, $\mathbf{Mode}_2$ with a larger workspace will be activated. $\mathbf{Mode}_3$ is for the Task 2, which is much simpler than Task 1. The factory can only be at one of the three modes at a time. **M1** and **M2** are, according to the rough classification of systems, two parallel-mode systems.

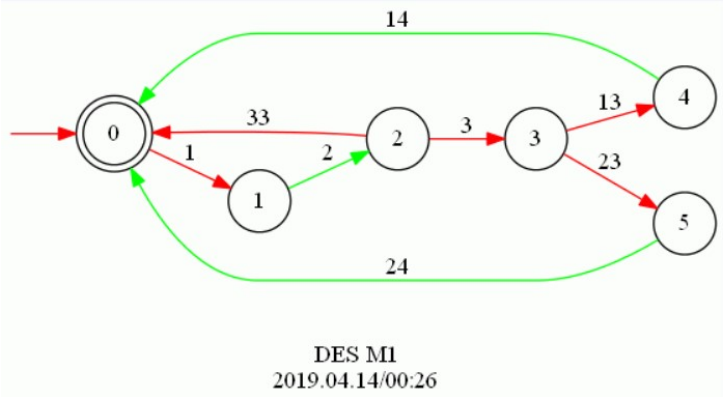

Figure 4.14: Plant component **M1**

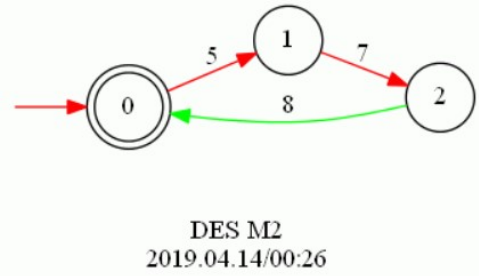

Figure 4.15: Plant component **M2**

According to the event alphabet for the three modes, **M1** involves all the three modes of the system. It is obvious that in **M1**, since state $0, 1, 2$ satisfy $P_1^{M1} \wedge P_2^{M1} \wedge P_3^{M1}$, they are public states with respect to any pair of modes. But since state 3 doesn't satisfy P_3^{M1}, it is only a public state with respect to $\mathbf{Mode}_1$ and $\mathbf{Mode}_2$. Apart from public states, state 4 only satisfies P_1^{M1} and state 5 only satisfies P_2^{M1}.

In **M2**, the initial state 0 is a public state with respect to any pair of modes. Since all events in **M2** are in $\Sigma_1 \cap \Sigma_2$, state 1 and state 2 in **M2** are two public states with respect to $\mathbf{Mode}_1$ and $\mathbf{Mode}_2$.

According to the classified states, $\Sigma_{ETE,1,2} = \{13, 23\}$, $\Sigma_{EYE,1,2} = \{14, 24\}$, $k_{1,2} = 1$, $\Sigma_{ETE,1,3} = \{3, 5\}$, $\Sigma_{EYE,1,3} = \{14, 24, 8\}$, $k_{1,3} = 2$, $\Sigma_{ETE,2,3} = \{3, 5\}$, $\Sigma_{EYE,2,3} = \{14, 24, 8\}$, $k_{2,3} = 2$. Thus, there will be three extra multiple reconfiguration specifications for each pair of modes of the three modes.

Define the mode initialization event and the reconfiguration event as $\sigma_1 = 81$, $\sigma_2 = 83$, $\sigma_3 = 85$, $\sigma_{1,2} = 91$, $\sigma_{2,1} = 93$, $\sigma_{1,3} = 95$, $\sigma_{3,1} = 97$, $\sigma_{2,3} = 99$, $\sigma_{3,2} = 89$. According to the classified events, the CMRS and the EMRS are shown below.

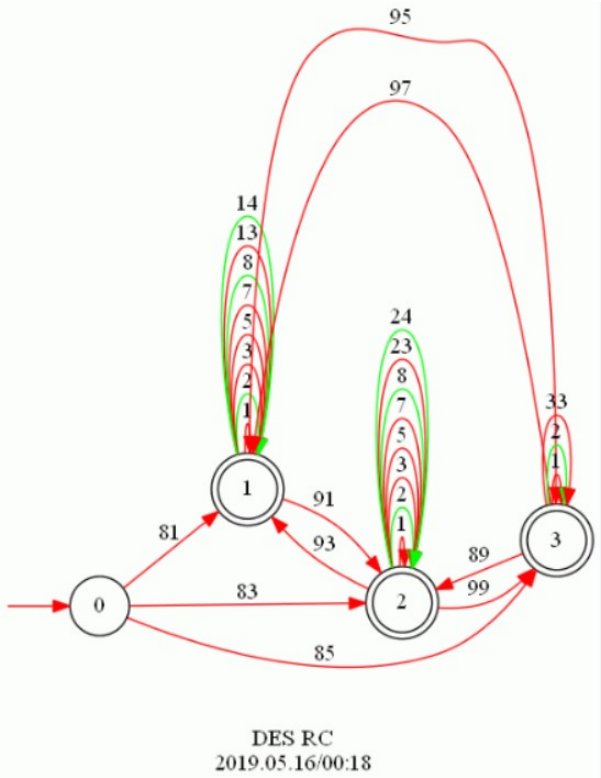

Figure 4.16: CMRS **RC**

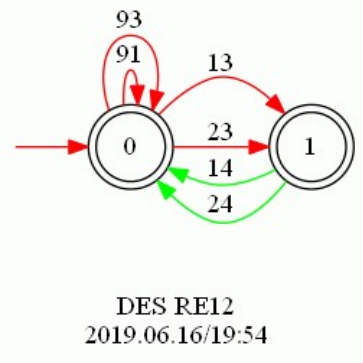

Figure 4.17: EMRS **RE12**

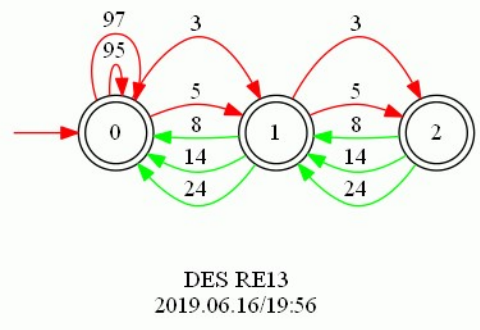

Figure 4.18: EMRS **RE13**

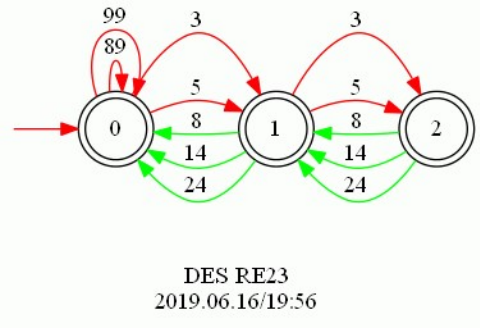

Figure 4.19: EMRS **RE23**

We then compute

$$\mathbf{RG} = \mathbf{Sync}(\mathbf{M1}, \mathbf{M2}, \mathbf{RC}, \mathbf{RE12}, \mathbf{RE13}, \mathbf{RE23}) \text{ (34,102)}$$

The reconfiguration plant **RG** is shown as follows. According to our requirement checking program in C++, the five requirements in Problem 4 are all met. The reader interested in this aspect may check it manually on the following diagram.

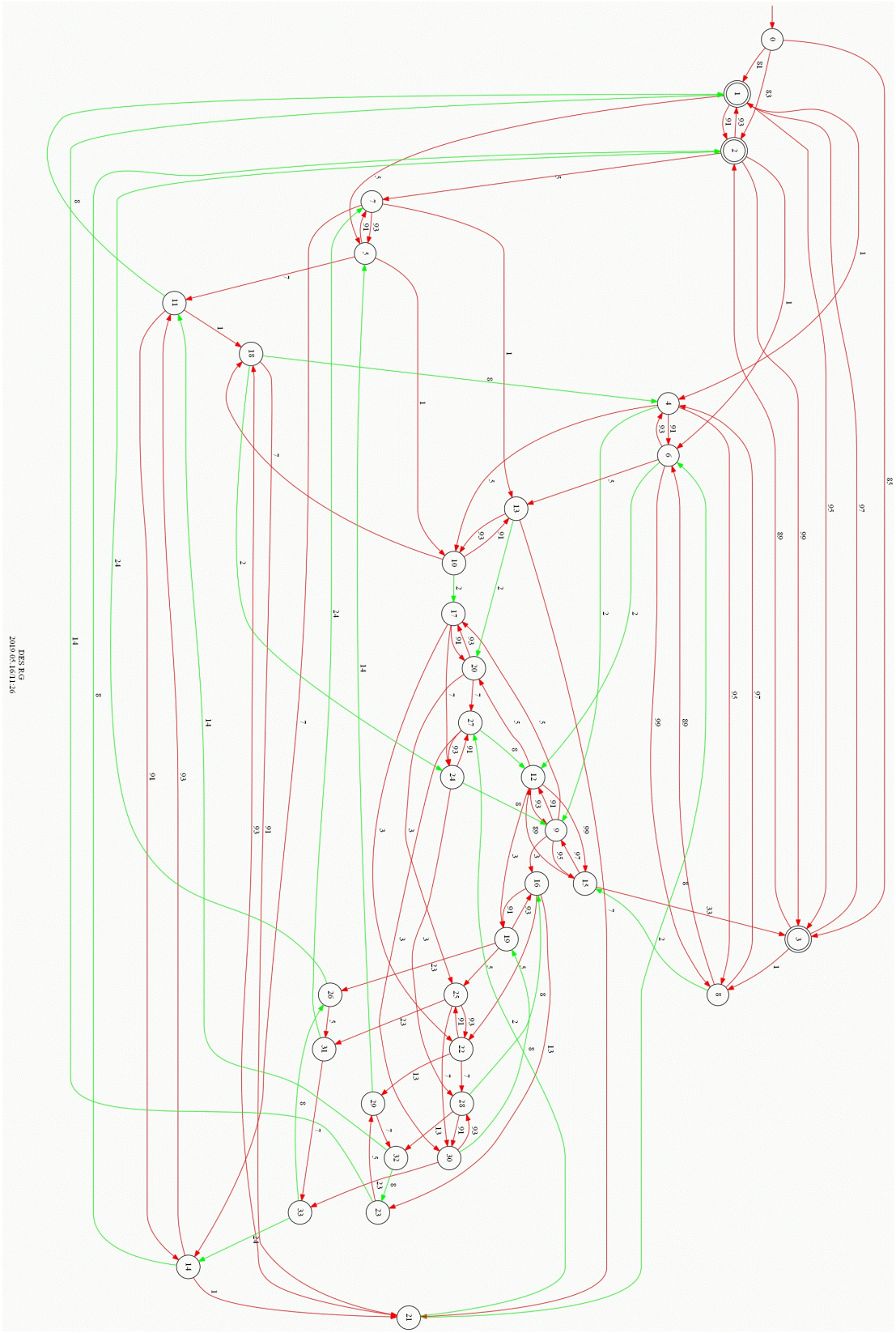

Figure 4.20: Reconfiguration plant **RG**

Next we add a logical requirement between the two machines, which is represented by a behavioral specification **B**. After the machine **M1**, there is a buffer with two slots that can store workpieces sent from **M1**. The workpieces can be picked by **M2** later. The **B** mainly states that the buffer can store at most two workpieces from **M1**, and **M2** cannot pick any workpiece if **M1** hasn't sent one. Since **M2** relates only to $\mathbf{Mode}_1$ and $\mathbf{Mode}_1$, we use event 14 and 24 to report that a workpiece has been sent to the buffer. Naturally, we use event 5 to report that a workpiece has been picked by **M2**. The behavioral specification. **B** is as follows.

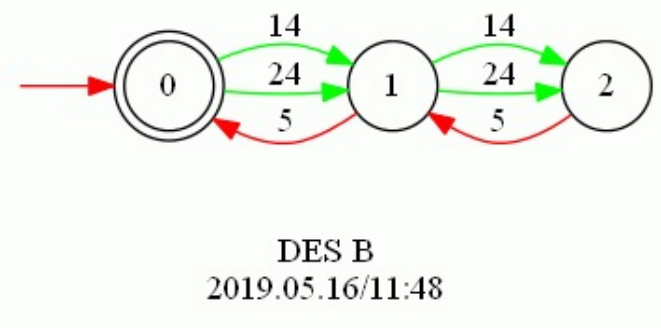

Figure 4.21: Behavioral specification **B**

We then compute

$$\mathbf{ALLRG} = \mathbf{Allevents}(\mathbf{RG.DES}) \text{ (1,20)}$$
$$\mathbf{BSPEC} = \mathbf{Sync}(\mathbf{ALLRG}, \mathbf{B}) \text{ (3,57)}$$
$$\mathbf{RSUP} = \mathbf{Supcon}(\mathbf{RG}, \mathbf{BSPEC}) \text{ (94,272)}$$
$$\mathbf{DRSUP} = \mathbf{Condat}(\mathbf{RG}, \mathbf{RSUP}) \text{ Controllable.}$$
$$\mathbf{RERSUP} = \mathbf{Supreduce}(\mathbf{RG}, \mathbf{RSUP}, \mathbf{DRSUP}) \text{ (3,10;slb=3)}$$

The resulting supervisor **RSUP** is too large to be shown here. Thus, we apply "Supreduce" function [1] to the resulting **RSUP** and obtain the reduced supervisor **RERSUP**. The **RERSUP** is shown in Figure 4.22.

According to the requirement checking program in C++, the **RSUP** satisfies all the five requirements in Problem 4. The reader interested in this aspect may repeat the foregoing procedure in TCT and check the result in the requirement checking program on the author's website (https://github.com/JasonZhangjc/).

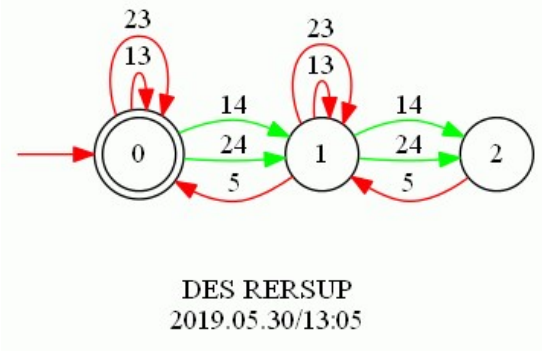

Figure 4.22: Reduced supervisor **RERSUP**

Moreover, the trigger behavior can also be added. Consider the back-and-forth reconfiguration behavior between $\mathbf{Mode}_1$ and $\mathbf{Mode}_2$. The trigger specification shown below can be introduced.

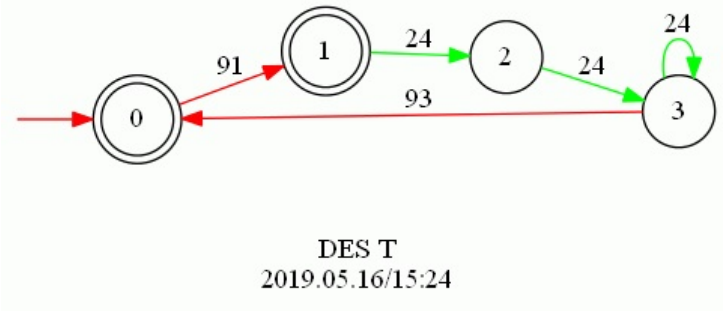

Figure 4.23: Trigger specification **T**

The TS **T** means that every time after the system switches from $\mathbf{Mode}_1$ to $\mathbf{Mode}_2$, it cannot switch back to $\mathbf{Mode}_1$ until finishing sending at least two workpieces to $\mathbf{M}_2$.

We then compute

$$\mathbf{TSPEC} = \mathbf{Sync}(\mathbf{ALLRG}, \mathbf{T}) \text{ (4,73)}$$
$$\mathbf{TRSUP} = \mathbf{Supcon}(\mathbf{RSUP}, \mathbf{TSPEC}) \text{ (352,819)}$$
$$\mathbf{DTRSUP} = \mathbf{Condat}(\mathbf{RSUP}, \mathbf{TRSUP}) \text{ Controllable.}$$
$$\mathbf{RETRSUP} = \mathbf{Supreduce}(\mathbf{RSUP}, \mathbf{TRSUP}, \mathbf{DTRSUP}) \text{ (7,53;slb=4)}$$

The resulting supervisor **TRSUP** is, according to the author's checking, consistent with the TS **T**, but is also too large to be shown. The reader interested in this aspect may repeat the foregoing procedure in TCT and check the result manually. Instead, we show the reduced supervisor **RETRSUP** in Figure 4.24.

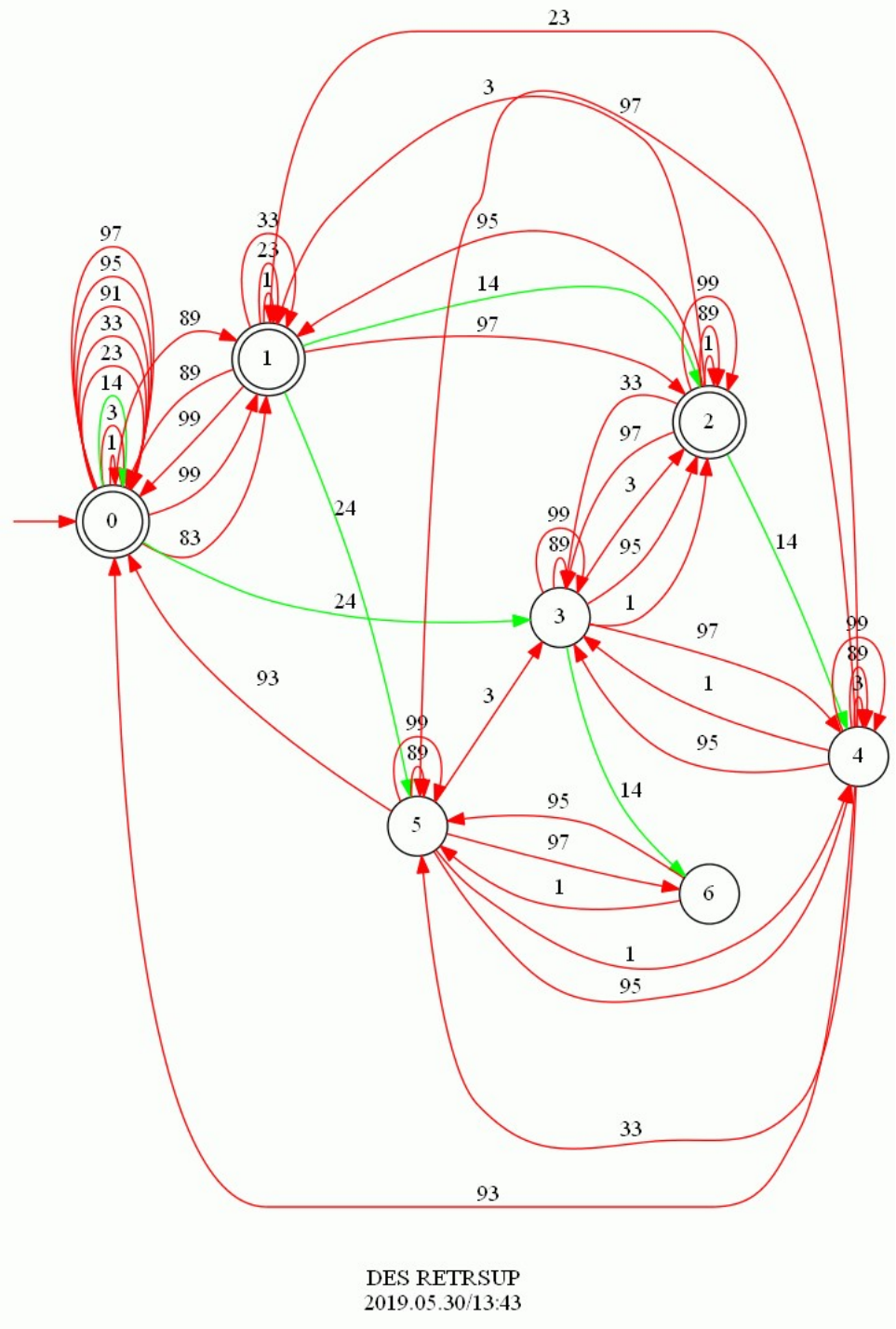

Figure 4.24: Reduced supervisor **RETRSUP**

In **RETRSUP** we can see that the RE 91 and 93 are correctly triggered according to the TS **T**.

Finally, in **RG**, the guaranteed reachability can be investigated with respect to a

source state and a target state. We arbitrarily choose state 5 as the source state and state 6 as the target state. According to our C++ implementation of the GR-Checking-1 algorithm and the GR-Checking-2 algorithm, state 6 is guaranteed reachable from state 5. The length of the shortest paths is 4. For example, some shortest paths are [5] → [11] → [1] → [2] → [6], [5] → [11] → [14] → [2] → [6] and [5] → [7] → [14] → [2] → [6].

The computation time is almost the same for the two algorithms, since this example is relatively small. With a larger example, the GR-Checking-2 tends to compute faster than the GR-Checking-1.

4.7.2 Examples from the Literature

Here we apply the monolithic multiple reconfiguration approach to examples 3.9.1 and 3.9.2 in [1].

For example 3.9.1 in [1], all states of **MACH1** and the initial state of **MACH2** and **MACH3** are public states with respect to $\mathbf{Mode}_1$ and $\mathbf{Mode}_2$. Thus, $\Sigma^G_{ETE,1,2} = \{21, 31\}$, $\Sigma^G_{EYE,1,2} = \{20, 23, 30, 34\}$, and $k_{1,2} = 2$. We still use events 81, 83 and 91, 93 as MIE and RE. Accordingly, the CMRS **RC391** and the EMRS **RE391** are as follows.

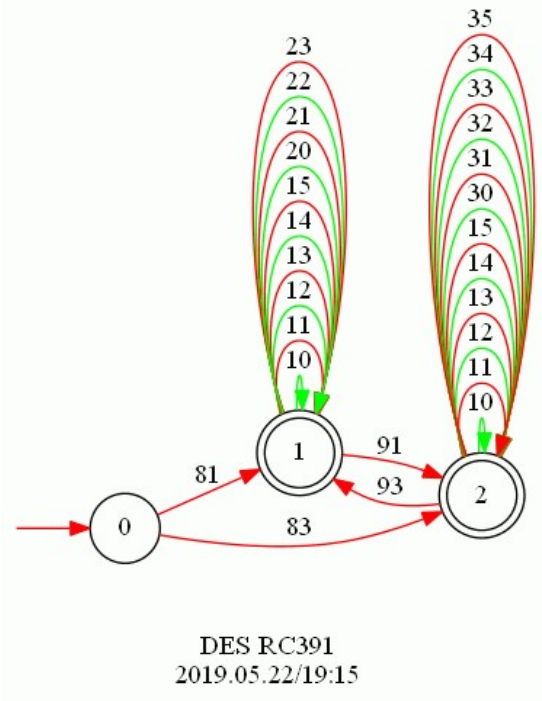

Figure 4.25: CMRS **RC391** for example 3.9.1 in [1]

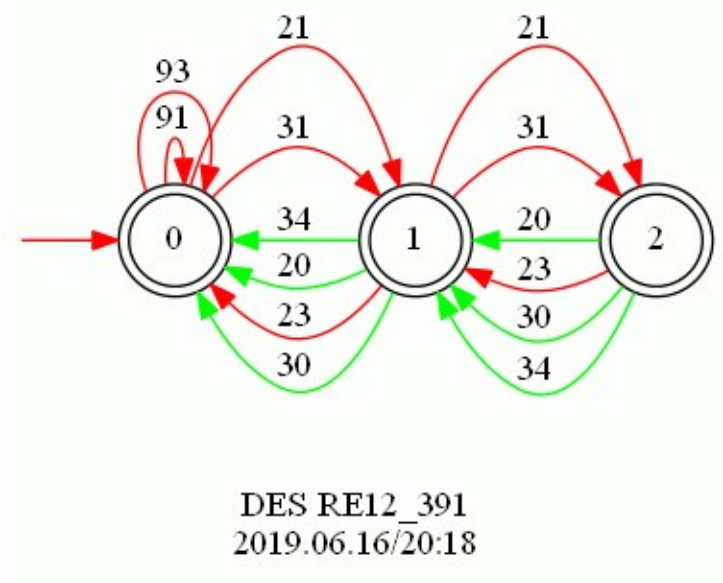

Figure 4.26: EMRS **RE391** for example 3.9.1 in [1]

We then compute

$$\mathbf{RGM391} = \mathbf{Sync}(\mathbf{MACH1}, \mathbf{MACH2}, \mathbf{MACH3}, \mathbf{RC391}, \mathbf{RE391}) \quad (29{,}92)$$
$$\mathbf{true} = \mathbf{Isomorph(RG391,RGM391;identity)}$$

This means that the obtained reconfiguration plant **RGM391** is isomorphic to the **RG391** obtained in Section 3.4.

We next consider the behavioral specifications associated with example 3.9.1. The two specifications are as follows.

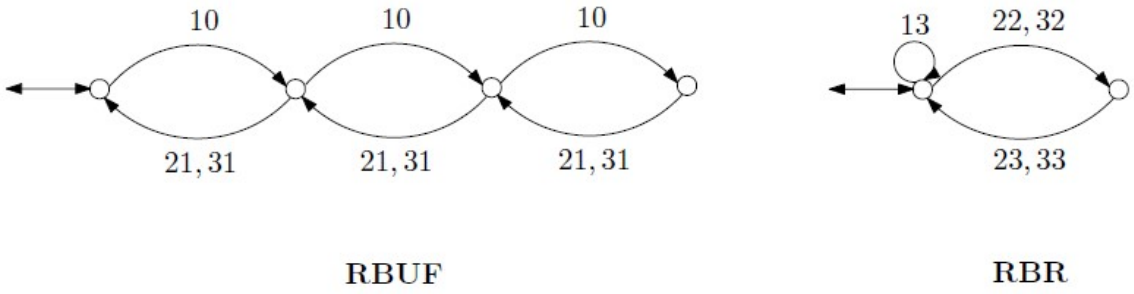

Figure 4.27: Two behavioral specifications for example 3.9.1 in [1]

The first specification is a buffer with 3 slots (**RBUF**), and the second is a priority logic (**RBR**) for breakdown and repair as displayed above. Note that these must be compatible with both modes. We then compute

$$\mathbf{ALLRGM391} = \mathbf{Allevents}(\mathbf{RGM391})\ (1{,}20)$$
$$\mathbf{RSPEC391} = \mathbf{Sync}(\mathbf{ALLRGM391}, \mathbf{RBUF}, \mathbf{RBR})\ (8{,}134)$$
$$\mathbf{RSUP391} = \mathbf{Supcon}(\mathbf{RGM391}, \mathbf{RSPEC391})\ (92{,}257)$$
$$\mathbf{DRSUP391} = \mathbf{Condat}(\mathbf{RGM391}, \mathbf{RSUP391})\ \text{Controllable.}$$
$$\mathbf{RERSUP391} = \mathbf{Supreduce}(\mathbf{RGM391}, \mathbf{RSUP391}, \mathbf{DRSUP391})\ (7{,}77;\text{slb}{=}7)$$

According to the C++ program for requirement checking, the resulting **RSUP391** satisfies the five requirements. The reduced supervisor **RERSUP391** is shown below.

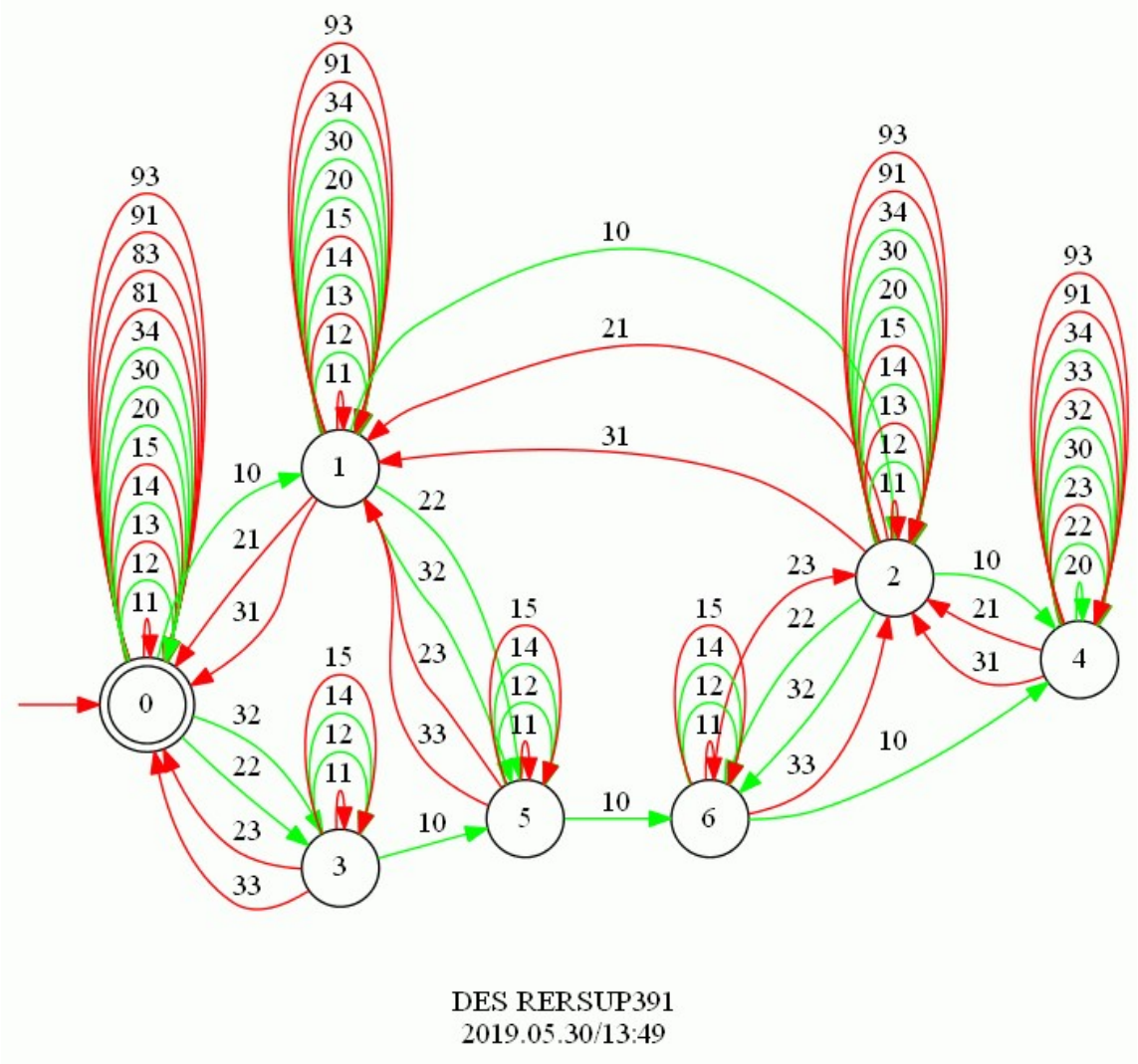

Figure 4.28: Reduced supervisor **RERSUP391**

Evidently, **RERSUP391** coincides with the BS **RBUF** and **RBR**.

Similarly, for example 3.9.2 in [1], all states of **MACH1** and **MACH2** are public

states with respect to $\mathbf{Mode}_1$ and $\mathbf{Mode}_2$. Thus, $\Sigma^G_{ETE,1,2} = \Sigma^G_{EYE,1,2} = \emptyset$. Since $\Sigma^G_{ETE,1,2} = \emptyset$, the EMRS doesn't exist. Here we still use events 81, 83 and 91, 93 as MIE and RE. Accordingly, the CMRS **RC392** is as follows.

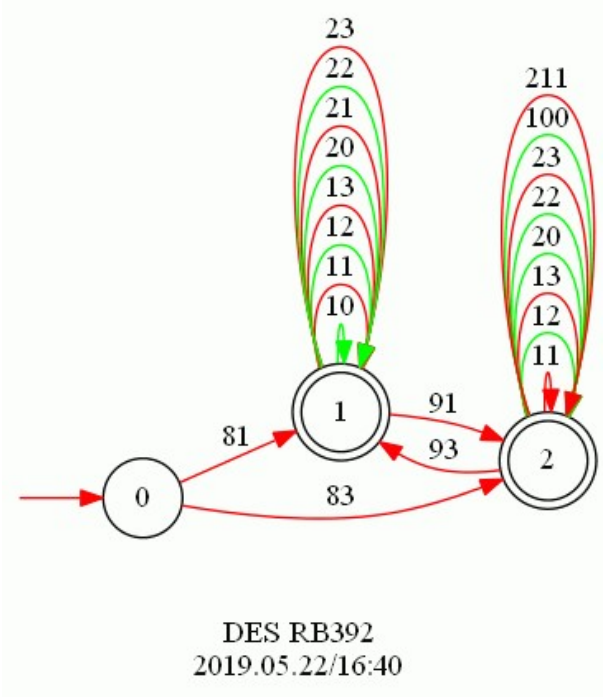

Figure 4.29: CMRS **RC392** for example 3.9.2 in [1]

For this special case, since **RC392** is the same as **RB392** in Section 3.4, the resulting reconfiguration specifications are the same.

For the two behavioral specifications in Section 3.4, we then compute

$$\mathbf{ALLRGM392} = \mathbf{Allevents}(\mathbf{RGM392}) \text{ (1,14)}$$

$$\mathbf{RSPEC392} = \mathbf{Sync}(\mathbf{ALLRGM392}, \mathbf{BUF1}, \mathbf{BUF2}) \text{ (8,100)}$$

$$\mathbf{RSUP392} = \mathbf{Supcon}(\mathbf{RGM392}, \mathbf{RSPEC392}) \text{ (121,384)}$$

$$\mathbf{DRSUP392} = \mathbf{Condat}(\mathbf{RGM392}, \mathbf{RSUP392}) \text{ Controllable.}$$

$$\mathbf{RERSUP392} = \mathbf{Supreduce}(\mathbf{RGM392}, \mathbf{RSUP392}, \mathbf{DRSUP392}) \text{ (12,55;slb=11)}$$

According to the C++ program for requirement checking, the resulting **RSUP391** satisfies the five requirements. The reduced supervisor **RERSUP392** is shown below.

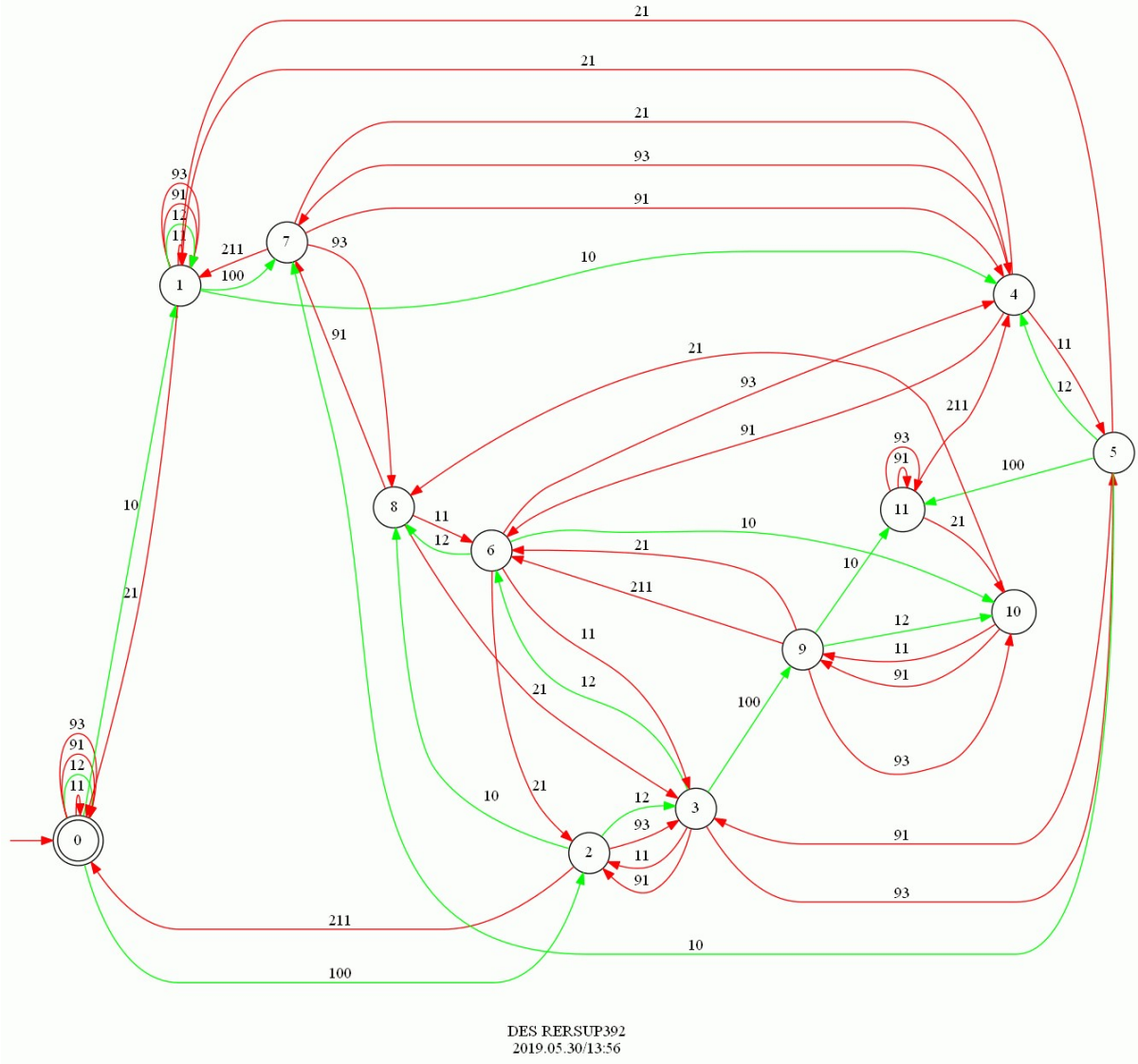

Figure 4.30: Reduced supervisor **RERSUP392**

Evidently, **RERSUP392** coincides with the BS **BUF1** and **BUF2**.

4.8 Chapter Summary and Discussion

First of all, this chapter has extended the bidirectional reconfiguration problems to multiple reconfiguration problems, which cannot be completely solved by the bidirectional reconfiguration approach. Therefore, this chapter has formally redefined some key notions such as public states, exit events, and entry events. With these new definitions, the extra multiple reconfiguration specifications are introduced to compensate for the

lack of information in the core multiple reconfiguration specification. This approach is also bidirectional. Besides, it is proved to be a general version for the bidirectional reconfiguration approach and is shown compatible with supervisory controller synthesis. Trigger behavior is then added in to regulate reconfiguration behavior. Finally, guaranteed reachability in the reconfiguration problem is studied as an important issue in applications.

This multiple reconfiguration approach is monolithic in the sense that the reconfiguration specification is about the entire plant, not each plant component. However, in order to realize the monolithic feature, the assumptions (iii) and (iv), which state that the exit (entry) events for distinct plant component are distinct in Definition 17, are essential. To relax these two assumptions, a localized version of multiple reconfiguration approach is required. In Appendix A, we will briefly introduce the localized multiple reconfiguration approach. Here we generally summarize the advantages and disadvantages of the bidirectional, monolithic multiple and localized multiple reconfiguration approaches in the following table.

Table 4.2: Comparison of reconfiguration approaches in this report

Reconfiguration Approach	**Number of Modes**	**Number of Assumptions**	**Number of RS**	**Structure of RS**
Bidirectional	2	6	1	Complex
Monolithic Multiple	≥ 2	6	$\leq 1 + C_2^n$	Simple
Localized Multiple	≥ 2	4	$\leq 1 + h * C_2^n$	Simple

Chapter 5

Conclusions and Future Work

In conclusion, we briefly summarize the key results of Chapter 3, Chapter 4, and Appendix A of this report.

In Chapter 3, we extended the unidirectional reconfiguration approach to bidirectional and formally proved its correctness under certain assumptions. The approach can dynamically and bidirectionally coordinate reconfiguration behavior, effectively avoid unsuitable reconfiguration, and completely preserve the dynamics of the plant. Besides, this bidirectional approach is able to solve many existing reconfiguration problems in the literature.

In Chapter 4, we constructed the core multiple reconfiguration specification and the extra multiple reconfiguration specifications to deal with reconfiguration problems in a system with more than two modes. A formal proof was provided to show the approach's correctness and another proof was presented to show that the multiple reconfiguration approach is a general version of the bidirectional reconfiguration approach. Moreover, we proved the compatibility of this monolithic multiple reconfiguration approach with supervisory control synthesis. The trigger requirement for reconfiguration events was successfully dealt with in the course of supervisory control synthesis as well. Finally, we designed two backtracking algorithms for different demands regarding guaranteed reachability and path finding and compared their advantages and disadvantages.

In Appendix A, we efficiently localized the multiple reconfiguration approach on each plant component and succeeded in solving multiple reconfiguration problems when assumption (iii) and (iv) are relaxed. Finally, a proof of correctness was provided and the

comparison between the monolithic and localized approaches was briefly discussed.

In future work, the following issues should be addressed.

(i) In order to solve a wider range of reconfiguration problems, a bidirectional reconfiguration approach and a multiple reconfiguration approach should be developed for timed discrete-event systems [55].

(ii) For scalability, the proposed reconfiguration approaches should be adapted to the state tree structures (STS) [10] framework.

(iii) For hierarchy [56], the proposed reconfiguration approaches for hierarchical discrete event systems should be studied.

(iv) To relax the implicit assumption that the event occurrencess are fully observable, the bidirectional reconfiguration and the multiple reconfiguration approaches need to take partial observation [57] into consideration.

Bibliography

[1] W. M. Wonham and K. Cai, *Supervisory Control of Discrete-Event Systems*. Springer, 2018.

[2] R. Oueslati, O. Mosbahi, M. Khalgui, Z. Li, and T. Qu, "Combining semi-formal and formal methods for the development of distributed reconfigurable control systems," *IEEE Access*, vol. 6, pp. 70426–70443, 2018.

[3] C. Bertelle, G. H. Duchamp, and H. Kadri-Dahmani, *Complex systems and self-organization modelling*. Springer Science & Business Media, 2008.

[4] M. Macktoobian and W. Wonham, "Automatic reconfiguration of untimed discrete-event systems," in *2017 14th International Conference on Electrical Engineering, Computing Science and Automatic Control (CCE)*, pp. 1–6, IEEE, 2017.

[5] J. Daintith and E. Wright, *A dictionary of computing*. Oxford University Press, Inc., 2008.

[6] J. Zhang, G. Frey, A. Al-Ahmari, T. Qu, N. Wu, and Z. Li, "Analysis and control of dynamic reconfiguration processes of manufacturing systems," *IEEE Access*, vol. 6, pp. 28028–28040, 2017.

[7] R. Balani, C.-C. Han, R. K. Rengaswamy, I. Tsigkogiannis, and M. Srivastava, "Multi-level software reconfiguration for sensor networks," in *Proceedings of the 6th ACM & IEEE International conference on Embedded software*, pp. 112–121, ACM, 2006.

[8] C. G. Cassandras, *Discrete event systems: modeling and performance analysis*. CRC, 1993.

[9] C. G. Cassandras and S. Lafortune, *Introduction to discrete event systems*. Springer Science & Business Media, 2009.

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

[11] P. A. Maltseff and R. Byford, "Secure multi-mode communication between agents," May 15 2014. US Patent App. 13/677,246.

[12] B. Stern, X. Zhu, C. P. Chen, L. D. Tzuang, J. Cardenas, K. Bergman, and M. Lipson, "On-chip mode-division multiplexing switch," *Optica*, vol. 2, no. 6, pp. 530–535, 2015.

[13] T. Sun, B. Xu, B. Chen, X. Chen, M. Li, P. Shi, and F. Wang, "Anti-counterfeiting patterns encrypted with multi-mode luminescent nanotaggants," *Nanoscale*, vol. 9, no. 8, pp. 2701–2705, 2017.

[14] S. J. Bosman, M. F. Gely, V. Singh, A. Bruno, D. Bothner, and G. A. Steele, "Multi-mode ultra-strong coupling in circuit quantum electrodynamics," *npj Quantum Information*, vol. 3, no. 1, p. 46, 2017.

[15] P. M. Wiegmann, H. J. de Vries, and K. Blind, "Multi-mode standardisation: A critical review and a research agenda," *Research Policy*, vol. 46, no. 8, pp. 1370–1386, 2017.

[16] P. Barriuso, J. Dixon, P. Flores, and L. Morán, "Fault-tolerant reconfiguration system for asymmetric multilevel converters using bidirectional power switches," *IEEE Transactions on Industrial Electronics*, vol. 56, no. 4, pp. 1300–1306, 2008.

[17] P. Pitchappa, M. Manjappa, H. N. Krishnamoorthy, Y. Chang, C. Lee, and R. Singh, "Bidirectional reconfiguration and thermal tuning of microcantilever metamaterial device operating from 77 k to 400 k," *Applied Physics Letters*, vol. 111, no. 26, p. 261101, 2017.

[18] D. Lüdtke, D. Tutsch, A. Walter, and G. Hommel, "Improved performance of bidirectional multistage interconnection networks by reconfiguration," in *Proceedings of*, pp. 21–27, 2005.

[19] Y.-C. Lan, H.-A. Lin, S.-H. Lo, Y. H. Hu, and S.-J. Chen, "A bidirectional noc (binoc) architecture with dynamic self-reconfigurable channel," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 30, no. 3, pp. 427–440, 2011.

[20] W. Wang, P. Mishra, and S. Ranka, *Dynamic Reconfiguration in Real-Time Systems*. Springer, 2012.

[21] G. W. Stewart, *Introduction to matrix computations*. Elsevier, 1973.

[22] C. Wang, X. Yang, Z. Wu, Y. Che, L. Guo, S. Zhang, and Y. Liu, "A highly integrated and reconfigurable microgrid testbed with hybrid distributed energy sources," *IEEE Transactions on Smart Grid*, vol. 7, no. 1, pp. 451–459, 2014.

[23] C. van Leeuwen, Y. Rieter-Barrell, Z. Papp, A. Pruteanu, and T. Vogel, "Model-based engineering of runtime reconfigurable networked embedded systems," in *Runtime Reconfiguration in Networked Embedded Systems*, pp. 1–28, Springer, 2016.

[24] A. Teixeira, J. Araújo, H. Sandberg, and K. H. Johansson, "Distributed actuator reconfiguration in networked control systems," *IFAC Proceedings Volumes*, vol. 46, no. 27, pp. 61–68, 2013.

[25] C. A. Sanchez, O. Mokrenko, L. Zaccarian, and S. Lesecq, "A hybrid control law for energy-oriented tasks scheduling in wireless sensor networks," *IEEE Transactions on Control Systems Technology*, no. 99, pp. 1–13, 2017.

[26] U. D. Atmojo, Z. Salcic, I. Kevin, and K. Wang, "Dynamic reconfiguration and adaptation of manufacturing systems using SOSJ framework," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 6, pp. 2353–2363, 2018.

[27] M. Khalgui, O. Mosbahi, and Z. Li, "On reconfiguration theory of discrete-event systems: From initial specification until final deployment," *IEEE Access*, vol. 7, pp. 18219–18233, 2019.

[28] H. Gharsellaoui and M. Khalgui, "Dynamic reconfiguration of intelligence for high behaviour adaptability of autonomous distributed discrete-event systems," *IEEE Access*, vol. 7, pp. 35487–35498, 2019.

[29] T. Jiao, Y. Gan, G. Xiao, and W. Wonham, "Exploiting symmetry of discrete-event systems by relabeling and reconfiguration," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, no. 99, pp. 1–12, 2018.

[30] J. Li, X. Dai, Z. Meng, J. Dou, and X. Guan, "Rapid design and reconfiguration of petri net models for reconfigurable manufacturing cells with improved net rewriting systems and activity diagrams," *Computers & Industrial Engineering*, vol. 57, no. 4, pp. 1431–1451, 2009.

[31] J. Li, X. Dai, and Z. Meng, "Automatic reconfiguration of petri net controllers for reconfigurable manufacturing systems with an improved net rewriting system-based approach," *IEEE transactions on automation science and engineering*, vol. 6, no. 1, pp. 156–167, 2008.

[32] A. Bukowiec and M. Doligalski, "Petri net dynamic partial reconfiguration in fpga," in *International Conference on Computer Aided Systems Theory*, pp. 436–443, Springer, 2013.

[33] M. Tadao, "Petri nets: properties, analysis and applications," *Proceedings of the IEEE*, vol. 77, no. 4, 1990.

[34] R. Sampath, H. Darabi, U. Buy, and J. Liu, "Control reconfiguration of discrete event systems with dynamic control specifications," *IEEE Transactions on Automation Science and Engineering*, vol. 5, no. 1, pp. 84–100, 2008.

[35] J. F. Zhang, O. Mosbahi, M. Khalgui, and A. Gharbi, "Feasible dynamic reconfigurations of petri nets," in *Formal Methods in Manufacturing Systems: Recent Advances*, pp. 247–267, IGI Global, 2013.

[36] C. Z. Y. Xinmin, "Case study on reconfiguration algorithm for reconfigurable manufacturing system [j]," *Journal of Computer Aided Design & Computer Graphics*, vol. 2, p. 004, 2003.

[37] J. Li, X. Dai, and Z. Meng, "Improved net rewriting systems-based rapid reconfiguration of petri net logic controllers," in *31st Annual Conference of IEEE Industrial Electronics Society, 2005. IECON 2005.*, pp. 6–pp, IEEE, 2005.

[38] T. Bourdeaud'huy and A. Toguyeni, "A petri-net based approach for the reconfiguration of flexible manufacturing systems using optimization techniques," *IFAC Proceedings Volumes*, vol. 39, no. 3, pp. 367–372, 2006.

[39] J. Chen, L.-W. Zhang, and J.-Q. Luo, "Reconfiguration cost analysis based on petrinet for manufacturing system," *Journal of Software Engineering and Applications*, vol. 2, no. 5, pp. 361–369, 2009.

[40] M. Doligalski and A. Bukowiec, "Partial reconfiguration in the field of logic controllers design," *International Journal of Electronics and Telecommunications*, vol. 59, no. 4, pp. 351–356, 2013.

[41] L. Kahloul, K. Djouani, and A. Chaoui, "Formal study of reconfigurable manufacturing systems: A high level petri nets based approach," in *Industrial Applications of Holonic and Multi-Agent Systems*, pp. 106–117, Springer, 2013.

[42] A. Kheldoun, K. Barkaoui, J. Zhang, and M. Ioualalen, "A high level net for modeling and analysis reconfigurable discrete event control systems," in *IFIP International Conference on Computer Science and its Applications*, pp. 551–562, Springer, 2015.

[43] S. Haddad and D. Poitrenaud, "Recursive petri nets," *Acta Informatica*, vol. 44, no. 7-8, pp. 463–508, 2007.

[44] J. Li, X. Guan, and J. Dou, "Petri net algebras for des model transformation," in *2010 International Conference on Computational Intelligence and Software Engineering*, pp. 1–4, IEEE, 2010.

[45] J. Zhang, H. Li, G. Frey, and Z. Li, "Reconfiguration control of dynamic reconfigurable discrete event systems based on NCESs," *IEEE Transactions on Control Systems Technology*, pp. 1–12, 2019.

[46] L. O. Matos and J. W. G. Sanchez, "Reconfiguration strategy for fault tolerance of power distribution systems using petri net," in *2016 IEEE Ecuador Technical Chapters Meeting (ETCM)*, pp. 1–6, IEEE, 2016.

[47] H. E. Garcia and A. Ray, "State-space supervisory control of reconfigurable discrete event systems," *International Journal of Control*, vol. 63, no. 4, pp. 767–797, 1996.

[48] A. Gehin and M. Staroswiecki, "A formal approach to reconfigurability analysis application to the three tank benchmark," in *1999 European Control Conference (ECC)*, pp. 4041–4046, IEEE, 1999.

[49] J. Zhang, M. Khalgui, Z. Li, G. Frey, O. Mosbahi, and H. B. Salah, "Reconfigurable coordination of distributed discrete event control systems," *IEEE Transactions on Control Systems Technology*, vol. 23, no. 1, pp. 323–330, 2014.

[50] R. Kumar and S. Takai, "A framework for control-reconfiguration following fault-detection in discrete event systems," *IFAC Proceedings Volumes*, vol. 45, no. 20, pp. 848–853, 2012.

[51] G. Faraut, L. PIetrac, and E. Niel, "Identification of incompatible states in mode switching," in *2008 IEEE International Conference on Emerging Technologies and Factory Automation*, pp. 121–128, IEEE, 2008.

[52] O. Kamach, L. Pietrac, and E. Niel, "Forbidden and preforbidden states in the multi-model approach," in *The Proceedings of the Multiconference on" Computational Engineering in Systems Applications"*, vol. 2, pp. 1550–1557, IEEE, 2006.

[53] R. Bellman, "Dynamic programming," *Science*, vol. 153, no. 3731, pp. 34–37, 1966.

[54] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to algorithms*. MIT press, 2009.

[55] B. A. Brandin and W. M. Wonham, "Supervisory control of timed discrete-event systems," *IEEE Transactions on Automatic Control*, vol. 39, no. 2, pp. 329–342, 1994.

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

[57] F. Lin and W. Wonham, "Supervisory control of timed discrete-event systems under partial observation," *IEEE Transactions on Automatic Control*, vol. 40, no. 3, pp. 558–562, 1995.

[58] M. L. Pinedo, *Planning and scheduling in manufacturing and services*. Springer, 2005.

Appendix A

Localized Multiple Reconfiguration of DES

A..1 Introduction

The monolithic approach to multiple reconfiguration of discrete-event systems has been proved successful to dynamically, automatically, and bidirectionally reconfigure a DES with more than two modes. However, the approach has to rely on six assumptions in Definition 17. Among them, assumptions (i) and (ii) are inevitable since the reconfiguration problems are studied on parallel-mode systems, assumptions (v) and (vi) are indispensable to avoid confusion in the SCT framework. In contrast, the assumptions (iii) and (iv) are relatively loose and can be relaxed.

In fact, if the approach is not monolithic, namely localized, the assumptions (iii) and (iv) are not required any longer.

This Appendix presents a localized procedure to dynamically and bidirectionally reconfigure discrete-event systems with multiple modes. To this end, we first review why the monolithic multiple reconfiguration approach in Chapter 4 is inconvenient. Then we construct localized reconfiguration specifications. With these reconfiguration specifications, the multiple reconfiguration of DES can be managed in a localized way.

This Appendix is organized as follows. Section A..2 analyzes the inconvenience of the monolithic multiple reconfiguration approach and provides intuition about the new approach. Section A..3 elaborates on the construction of the localized reconfiguration specifications. Section A..4 illustrates the proposed method with a running example. Conclusions are drawn in Section A..5.

A..2 Assumption Relaxation

In Chapter 4, in order to make the monolithic multiple reconfiguration approach valid, six assumptions in Definition 17 are required to hold. Among them, assumptions (iii) and (iv) introduce some restriction to the approach, hence need to be relaxed.

Consider the following example where the plant has two modes and two plant components.

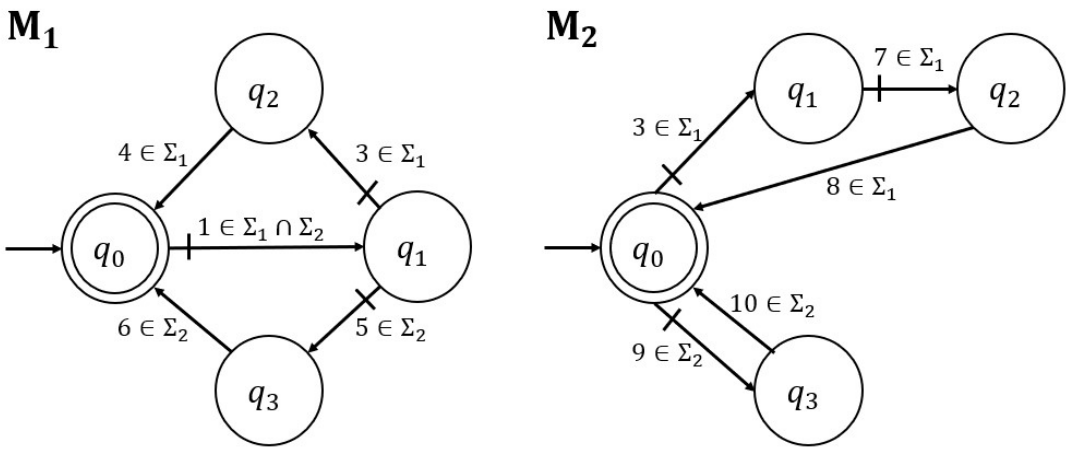

Figure A..1: Plant with two modes and two plant components

In this example, the two modes are distinguished by $\Sigma_1 = \{1, 3, 4, 7, 8\}$ and $\Sigma_2 = \{1, 5, 6, 9, 10\}$. Note that event 3 is shared in $\mathbf{M}_1$ and $\mathbf{M}_2$. This issue will lead to a violation of assumption (iii) and hence neutralize the monolithic multiple reconfiguration approach proposed in Chapter 4. Here we are going to present the problem and analyze the reason in detail.

If we attempt to solve this reconfiguration problem by the monolithic multiple reconfiguration approach, we need to summarize the public state, exit event and the entry event with respect to $\mathbf{Mode}_1$ and $\mathbf{Mode}_2$ first. As a result, $Q^1_{PBS,1,2} = \{q_0, q_1\}$, $Q^2_{PBS,1,2} = \{q_0\}$, $\Sigma^G_{ETE,1,2} = \{3, 5, 9\}$ and $\Sigma^G_{EYE,1,2} = \{4, 6, 8, 10\}$. Thus, $k_{1,2} = 2$. According to the foregoing event sets, the CMRS $\mathbf{R}_C$ and the EMRS $\mathbf{R}_{1,2}$ are as follows.

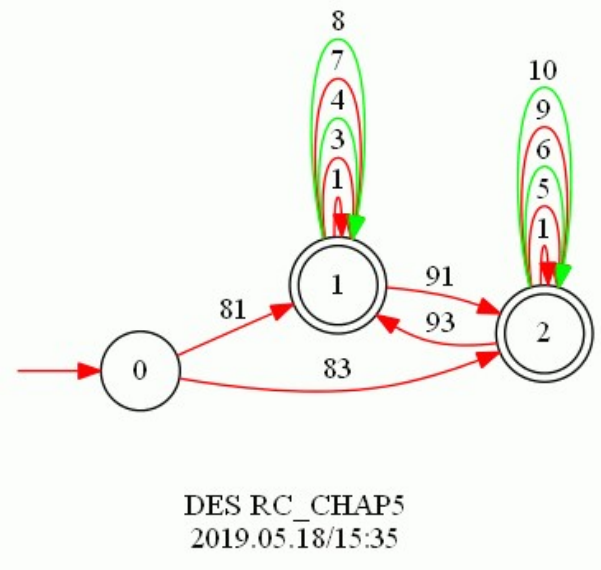

Figure A..2: CMRS **RC**

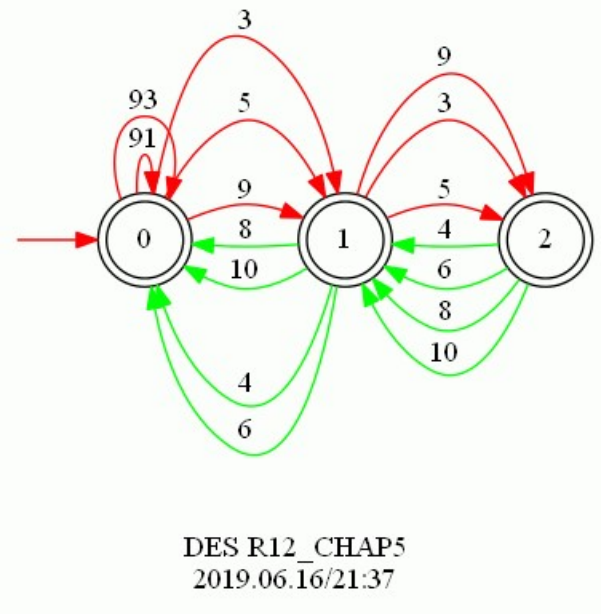

Figure A..3: EMRS **R12**

We then compute

$$\mathbf{RG} = \mathbf{Sync}(\mathbf{M}_1, \mathbf{M}_2, \mathbf{R}_C, \mathbf{R}_{12}) \quad (24{,}46)$$

The resulting reconfiguration plant **RG** is shown as follows.

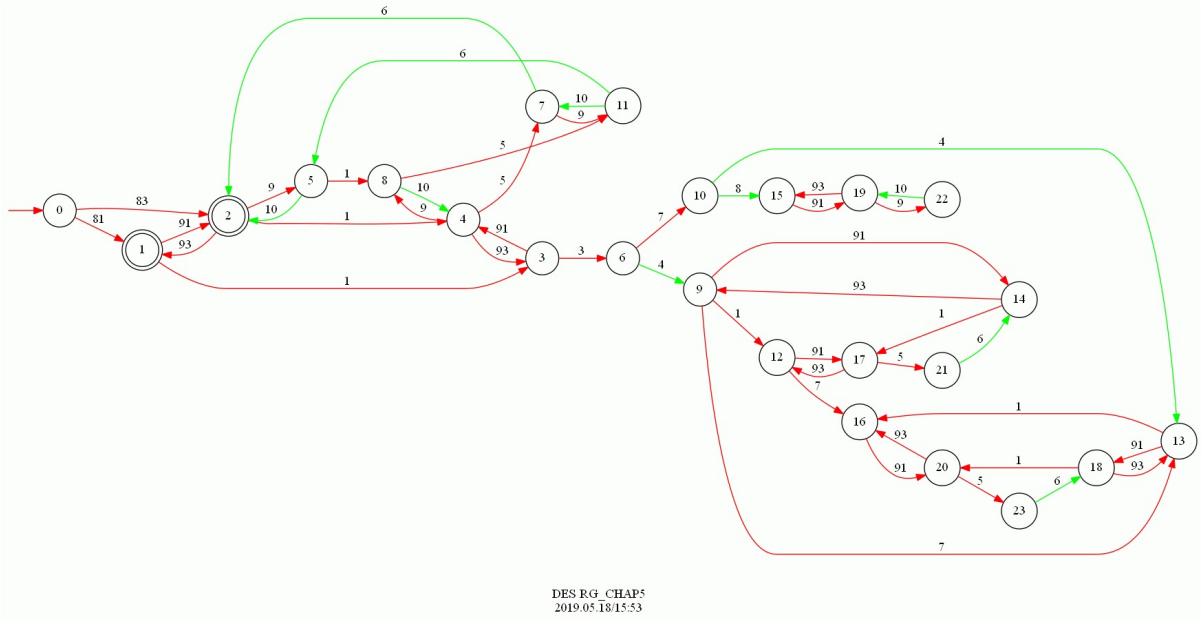

Figure A..4: Reconfiguration plant **RG**

Note that the **RG** fails to be nonblocking. Consider state 3 and any state that is reachable from state 3. They are not coreachable, namely, they cannot reach marked states eventually. The reason here is that event 3 occurs in $\mathbf{R}_{12}$ and leads to state 1 from state 0. Then in the synchronization, since event 3 is shared by $\mathbf{M}_1$ and $\mathbf{M}_2$, event 3 occurs in both $\mathbf{M}_1$ and $\mathbf{M}_2$ at the same time. Ideally, when $\mathbf{R}_{12}$ returns to state 0, the reconfiguration event 91 or 93 are expected to be eligible to occur. After an occurrence of event 4, $\mathbf{R}_{12}$ returns to state 0 and $\mathbf{M}_1$ also returns to state q_0. According to $\mathbf{R}_{12}$, RE 91 and 93 are eligible to occur. However, $\mathbf{M}_2$ is still at state q_1, which is not a PBS with respect to $\mathbf{Mode}_1$ and $\mathbf{Mode}_2$. Thus, the occurrence of RE here will result in blocking.

In general, in a system with more than two modes, the blocking issue still exists. The issue also arises when an exit event is shared by more than two plant components. Similarly, if an entry event with respect to any two modes of the system is shared by more than one plant component, there will also be unsuitable reconfiguration and a blocking issue in the reconfiguration approach. Since there is no violation of the definition of CMRS or EMRS, we can conclude that the monolithic multiple reconfiguration approach doesn't work for a reconfiguration problem without obeying assumptions (iii) and (iv).

We can see that the fundamental problem here lies in the inadequacy of the monolithic

multiple reconfiguration approach when some exit (entry) event is shared by more than one plant component. The simplest way to solve it is to relabel the shared events for distinct plant components. However, relabelling is not always feasible in practice. Therefore, in a practical sense, in order to solve a wider range of reconfiguration problems, a more adaptable multiple reconfiguration approach is required.

Inspired by "Supervisor Localization" [1], we developed a technique to deal with this fundamental problem in a localized architecture, namely by constructing reconfiguration specifications for each plant component separately.

A..3 Localized Multiple Reconfiguration Approach

A..3.1 Localized Multiple Reconfiguration Specification

In the sense of localized or distributed approaches, it is natural to construct a reconfiguration specification for each plant component. However, it's not necessary to construct several CMRS since a CMRS provides the core framework and the simplest structure for the reconfiguration problem of the current plant. Constructing more than one CMRS would only complicate the approach and won't be helpful to solve the critical problem. Thus, we only localize the EMRS for each plant component.

A localized extra multiple reconfiguration specification (LEMRS) for the plant component $\mathbf{G}^k$ is an EMRS within the scope of events in Σ^k instead of Σ. The formal definition of localized extra multiple reconfiguration specification is provided below.

Definition 21. [Localized Extra Multiple Reconfiguration Specification (LEMRS)]. Denote by $\mathbf{G} = \mathbf{G}^1||...||\mathbf{G}^h$ the plant DES formed by synchronization. $\mathbf{Mode}_1, ..., \mathbf{Mode}_n$ are n different modes of $\mathbf{G}$ distinguished by event alphabets $\Sigma_1, ..., \Sigma_n$. The *localized extra multiple reconfiguration specification* with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ ($\forall i, j \in 1, ..., n, i < j$) for the component $\mathbf{G}^k$ is defined, if $\Sigma^k_{ETE,i,j} \neq \emptyset$, as the DES

$$\mathbf{R}^k_{i,j} := (Q^{R^k_{i,j}}, \Sigma^{R^k_{i,j}}, \delta^{R^k_{i,j}}, q_o^{R^k_{i,j}}, Q_m^{R^k_{i,j}})$$

where

- $Q^{R^k_{i,j}} := \{q_0^{R^k_{i,j}}, q_1^{R^k_{i,j}}\}$;

- $\Sigma^{R^k_{i,j}} := \{\sigma_{i,j}, \sigma_{j,i}\} \dot{\cup} \Sigma^k_{ETE,i,j} \dot{\cup} \Sigma^k_{EYE,i,j}$;
- $\delta^{R^k_{i,j}}(q_0^{R^k_{i,j}}, \sigma) := q_0^{R^k_{i,j}}$ if $\sigma = \sigma_{i,j}, \sigma_{j,i} \in \Sigma_{RE}$;
  $\delta^{R^k_{i,j}}(q_0^{R^k_{i,j}}, \sigma) := q_1^{R^k_{i,j}}$ if $\sigma \in \Sigma^k_{ETE,i,j}$;
  $\delta^{R^k_{i,j}}(q_1^{R^k_{i,j}}, \sigma) := q_0^{R^k_{i,j}}$ if $\sigma \in \Sigma^k_{EYE,i,j}$;
- $q_o^{R^k_{i,j}} := q_0^{R^k_{i,j}}$ is the initial state;
- $Q_m^{R^k_{i,j}} := Q^{R^k_{i,j}}$.

$\diamond$

Remark. If $\Sigma^k_{ETE,i,j} = \emptyset$, then every state of the plant component satisfies both P^k_i and P^k_j, so the RE $\sigma_{i,j}$ and $\sigma_{j,i}$ are eligible to occur at every state of the plant component. Thus, the eligibility of $\sigma_{i,j}$ is irrelevant to $\mathbf{G}^k$.

The $q_0^{R^k_{i,j}}$ $\forall k = 1, ..., h$ is the state at which $\sigma_{i,j}$ and $\sigma_{j,i}$ is defined. If no ETE with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ occurs, then the RE $\sigma_{i,j}$ and $\sigma_{j,i}$ are always eligible to occur. If an ETE occurs, then after an occurrence of EYE, the EMRS will return to the initial state, where the RE $\sigma_{i,j}$ and $\sigma_{j,i}$ are eligible to occur.

There is only one state except for the initial state in an LEMRS, instead of $k_{i,j}$ states as in an EMRS, while there might be more than one plant component. The logic here is that once the plant component exits the PBS with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$, it cannot exit them again, so there is no need to add one more state. Similarly, once the plant component enters the PBS with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$, it cannot enter them again. Usually, an EMRS relates to more than one plant components, so we have to consider adding more states. But in a LEMRS, two states in total are sufficient. Even in a situation where no entry event with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ exists as illustrated in Figure 3.17, two states are still sufficient.

Note that $\Sigma^k_{ETE,i,j}$ may contain events that are not in Σ_i or Σ_j to avoid blocking and in the reconfiguration plant. The correctness of the LEMRS will be provided later.

For example, a LEMRS $\mathbf{R}^k_{i,j}$ with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ for component $\mathbf{G}^k$ is shown below.

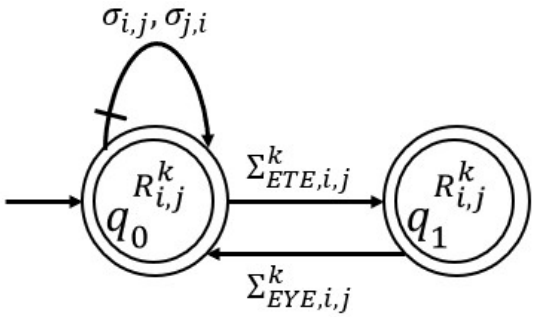

Figure A..5: LEMRS $\mathbf{R}^k_{i,j}$ with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ for $\mathbf{G}^k$

This LEMRS is also bidirectional since each LEMRS only deals with a pair of RE for bidirectional reconfiguration. In total, for a system with n modes and h components, there are at most $h * n * (n-1)/2$ LEMRS required. The LEMRS defined above and the CMRS defined in Chapter 4 can be synchronized to generate the localized multiple reconfiguration specification (LMRS) to solve Problem 4 without respecting assumption (iii) and (iv) in Definition 17.

Definition 22. [Localized Multiple Reconfiguration Specification (LMRS)]. Denote by $\mathbf{G} = \mathbf{G}^1||...||\mathbf{G}^h$ the plant DES, and $\mathbf{Mode}_1, ..., \mathbf{Mode}_n$ the n modes. The *localized multiple reconfiguration specification* is defined as the synchronous product

$$\mathbf{R} = \mathbf{R}_c||\mathbf{R}^1_{1,2}||...||\mathbf{R}^1_{n-1,n}||...||\mathbf{R}^h_{1,2}||...||\mathbf{R}^h_{n-1,n} = (Q^R, \Sigma^R, \delta^R, q^R_o, Q^R_m)$$

where $\Sigma^R = \Sigma^{R_c}$.

$\diamond$

Remark. There are at most $h * n * (n-1)/2$ LEMRS (i.e. $(\forall i, j \in 1, ..., n, i < j)(\forall k \in 1, ..., h)$ $\mathbf{R}^k_{i,j}$) that need to be synchronized.

It turns out that the multiple reconfiguration specification constructed according to Definition 22 solves Problem 4 through the synchronization with the plant DES $\mathbf{G}$ without complying with assumptions (iii) and (iv) in Definition 17. The resulting synchronous product is the reconfiguration plant that incorporates the multiple reconfiguration mechanism.

Here we summarize the assumptions required when using the localized multiple reconfiguration approach.

Definition 23. [Assumptions]. For a multiple reconfiguration problem, there are six assumptions.

(i) The initial state of each plant component belongs to every mode of the system, i.e.

$$(\forall i = 1, ..., n)(\forall k = 1, ..., h)\ q^k_{o,i} = q^k_o; \tag{A..1}$$

(ii) Every mode in each component DES is both reachable and coreachable by itself, i.e.

$$\begin{aligned}&(\forall i \in 1, ..., n)(\forall k \in 1, ..., h)\ [\Sigma_i \cap \Sigma^k \neq \emptyset \Rightarrow \\ &(\forall q \in Q^k_i)(\exists q^{'} \in Q^k_{m,i})(\exists s, s^{'} \in (\Sigma_i \cap \Sigma^k)^*)\delta^k_i(q^k_o, s) = q \wedge \delta^k_i(q, s^{'}) = q^{'}]\end{aligned} \tag{A..2}$$

and

$$(\forall i \in 1, ..., n)(\forall r \in 1, ..., h)\ [\Sigma_i \cap \Sigma^r = \emptyset \Rightarrow q^r_o \in Q^r_m] \tag{A..3}$$

(iii) An event cannot serve as both exit event and entry event with respect to the same pair of modes in different components, i.e.

$$(\forall i, j \in 1, ..., n, i \neq j)\ \Sigma^G_{ETE,i,j} \cap \Sigma^G_{EYE,i,j} = \emptyset. \tag{A..4}$$

(iv) Each event in the plant must belong to at least one of the modes, i.e.

$$\bigcup_{i=1,...,n} \Sigma_i = \Sigma \tag{A..5}$$

◇

Informally, we discuss why the reconfiguration plant constructed from the proposed LMRS can solve the unsuitable reconfiguration and blocking issue when some exit (entry) event is shared in more than one plant component. In the reconfiguration plant, which is the synchronous product of plant components, CMRS, and several LEMRS, if one shared exit event with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ occurs, then the plant components whose alphabet contains the event will have a state transition, so does the LEMRS with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ for these plant components. Then unless all the entry events with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ in all of these plant components have occurred, each plant

component will not return PBS with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$. At the same time, each corresponding LEMRS will not return the initial state either, where the RE between $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ are eligible to occur. Thus, only when all those plant components are at PBS with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ can the RE between $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ occur, which will not result in unsuitable reconfiguration or blocking issue.

All formal results and proofs of correctness are provided in Appendix B.

Similar to Chapter 4, the dynamic feature is not discussed above and it is automatically satisfied by the proposed approach. Moreover, this approach realizes the multiple reconfiguration mechanism in a localized fashion and can solve reconfiguration problems when assumption (iii) and (iv) in Definition 17 don't hold.

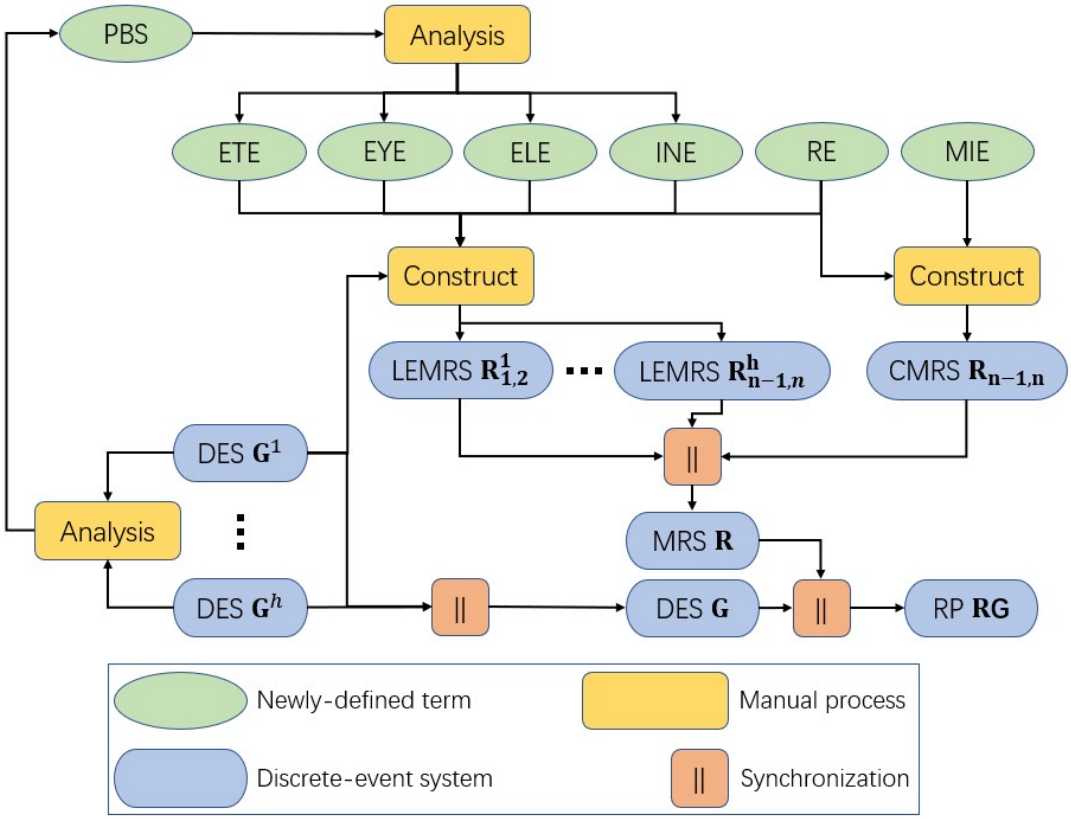

Figure A..6: Localized multiple reconfiguration of discrete-event systems

Evidently, the localized multiple reconfiguration approach can solve reconfiguration problems under only three assumptions in Definition 23, which means that the localized approach is a generalized version of the monolithic approach. The main procedure of the localized multiple reconfiguration approach is illustrated in Figure A..6.

A..3.2 Localized Bidirectional Reconfiguration Approach

In addition, this localization idea can also be applied to the bidirectional reconfiguration approach in Chapter 3 since the monolithic multiple reconfiguration approach is a generalized version of the bidirectional reconfiguration approach. Here we present the localized bidirectional reconfiguration specification (LBRS) but omit the proof of correctness.

Definition 24. [Localized Bidirectional Reconfiguration Specification (LBRS)]. Denote by $\mathbf{G} = \mathbf{G}^1||...||\mathbf{G}^h$ the plant DES formed by synchronization. $\mathbf{Mode}_1$ and $\mathbf{Mode}_2$ are 2 different modes of $\mathbf{G}$ distinguished by event alphabets Σ_1 and Σ_n. Given that $\Sigma_{MIE} = \{\sigma_1, \sigma_2\}$ and $\Sigma_{RE} = \{\sigma_{1,2}, \sigma_{2,1}\}$. The *localized bidirectional reconfiguration specification* for $\mathbf{G}^r$ is defined as the DES

$$\mathbf{R}^r := (Q^{R^r}, \Sigma^{R^r}, \delta^{R^r}, q_o^{R^r}, Q_m^{R^r})$$

where

- $Q^{R^r} := \{q_o^{R^r}\} \dot{\cup} Q_M^{R^r} \dot{\cup} Q_E^{R^r}$, in which $Q_M^{R^r} := \{q_1^0, q_2^0\}$ and for each of the two modes $\mathbf{Mode}_i$, if $\Sigma_{SETE}^r \cap \Sigma_i \neq \emptyset$, then there will be one extra state $q_i^1 \in Q_E^i \subseteq Q_E^{R^r}$. $Q_E^1 \dot{\cup} Q_E^2 = Q_E^{R^r}$ and $Q_E^1 \cap Q_E^2 = \emptyset$;
- $\Sigma^{R^r} := \Sigma^r \cup \Sigma_{RE} \cup \Sigma_{MIE}$;
- $\delta^{R^r}(q_o^{R^r}, \sigma) := q_i^0 \quad$ if $\sigma = \sigma_i \in \Sigma_{MIE}$;

  $\delta^{R^r}(q_i^0, \sigma) := q_j^0 \quad$ if $\sigma = \sigma_{i,j} \in \Sigma_{RE}$;

  $$\delta^{R^r}(q_i^t, \sigma) := \begin{cases} q_i^t & \text{if } \sigma \in \Sigma_i \cap \Sigma_{SINE}^r, 0 \leq t \leq 1; \\ q_i^{t+1} & \text{if } \sigma \in \Sigma_i \cap \Sigma_{SETE}^r, 0 \leq t < 1; \\ q_i^t & \text{if } \sigma \in \Sigma_i \cap \Sigma_{SELE}^r, 0 < t \leq 1; \\ q_i^{t-1} & \text{if } \sigma \in \Sigma_i \cap \Sigma_{SEYE}^r, 0 < t \leq 1; \end{cases}$$
- $q_o^{R^r}$ is the initial state of $\mathbf{R}^r$;
- $Q_m^{R^r} = Q^{R^r} - \{q_o^{R^r}\}$.

$\diamond$

The following Figure A..7 illustrates a typical LBRS $\mathbf{R}^r$ for $\mathbf{G}^r$.

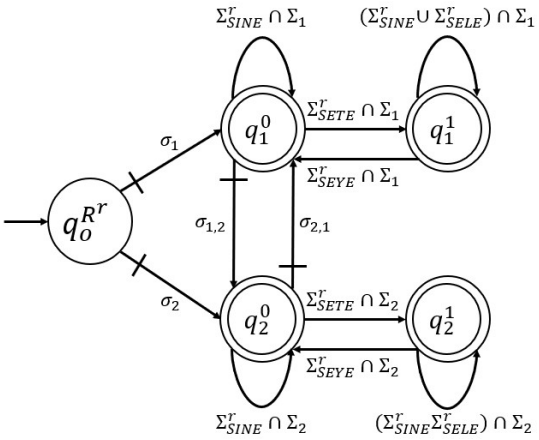

Figure A..7: Typical LBRS $\mathbf{R}^r$ for $\mathbf{G}^r$

We can then compute a bidirectional reconfiguration specification according to LBRS,i.e.,

$$\mathbf{R} = \mathbf{Sync}(\mathbf{R}^1, ..., \mathbf{R}^h)$$

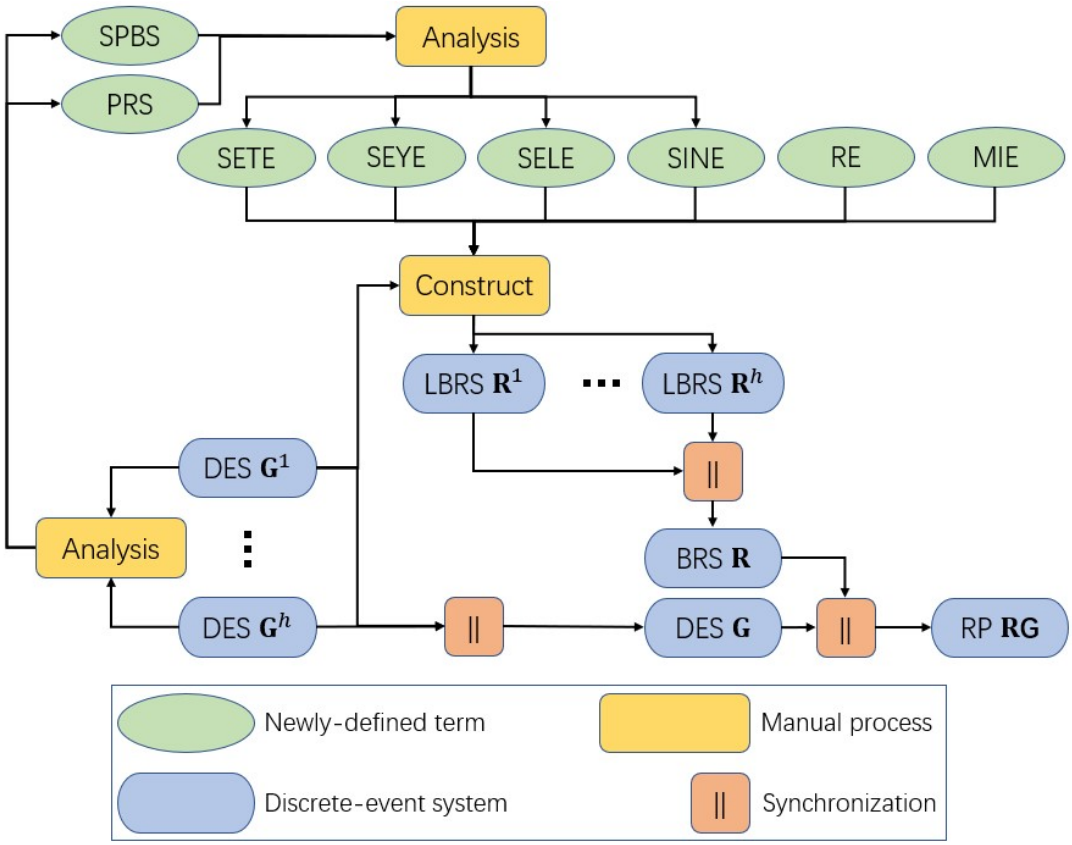

Figure A..8: Localized bidirectional reconfiguration of discrete-event systems

The main procedure of the localized bidirectional reconfiguration approach is illustrated in Figure A..8.

A..4 Examples

For simplicity, we apply the proposed localized approach to the example in section 5.1. For the component $\mathbf{M}_1$ in Figure A..1, $Q^{M_1}_{PBS,1,2} = \{q_0, q_1\}$, $\Sigma^{M_1}_{ETE,1,2} = \{3, 5\}$ and $\Sigma^{M_1}_{EYE,1,2} = \{4, 6\}$. For the component $\mathbf{M}_2$ in Figure 5.1, $Q^{M_2}_{PBS,1,2} = \{q_0\}$, $\Sigma^{M_2}_{ETE,1,2} = \{3, 9\}$ and $\Sigma^{M_2}_{EYE,1,2} = \{8, 10\}$. The CMRS remains the same as shown in Figure A..2. The two LEMRS are shown below.

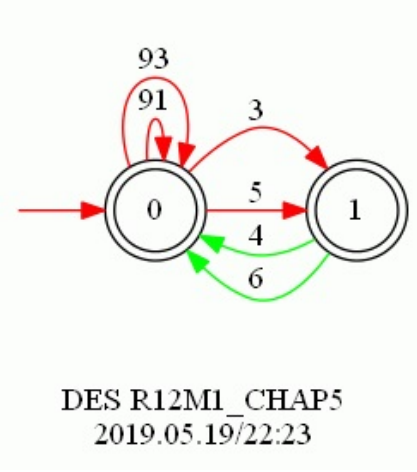

Figure A..9: LEMRS $\mathbf{R}^{M_1}_{1,2}$

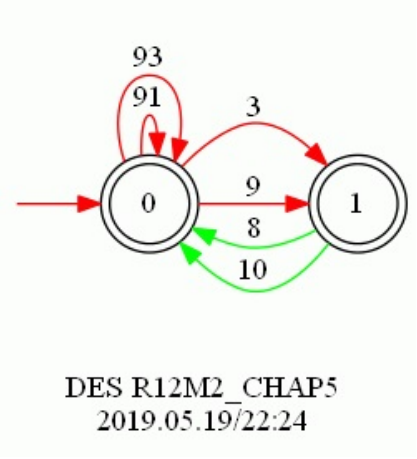

Figure A..10: LEMRS $\mathbf{R}^{M_2}_{1,2}$

We then compute

$$\mathbf{RG} = \mathbf{Sync}(\mathbf{M}_1, \mathbf{M}_2, \mathbf{R}_C, \mathbf{R}^{M_1}_{1,2}, \mathbf{R}^{M_2}_{1,2}) \quad (16{,}31)$$

The resulting reconfiguration plant **RG** is shown in Figure A..11.

It is evident that **RG** is nonblocking. After an occurrence of event 3, the RE 91 cannot occur until event 4 and event 8 occur in no particular order. Thus, the proposed localized multiple reconfiguration approach works. In fact, the resulting **RG** satisfies all the five requirements in Problem 4. The reader who interested in this aspect can manually check **RG**, as **RG** is not very complicated.

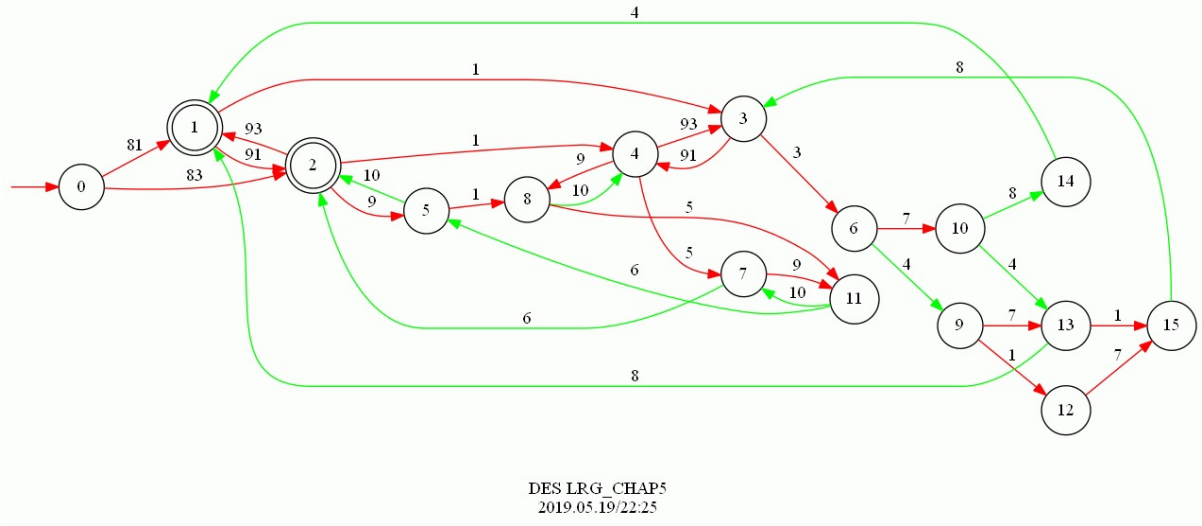

Figure A..11: Reconfiguration plant **RG**

Although the system in this example only has two modes and two plant components, we can easily extend it to a system with more than two modes and more than two plant components.

Moreover, we can test the proposed localized bidirectional reconfiguration approach on this example.

According to Figure 5.1, for the component $\mathbf{M}_1$, $Q^{M_1}_{SPBS} = \{q_0, q_1\}$, $\Sigma^{M_1}_{SETE} = \{3, 5\}$ and $\Sigma^{M_1}_{SEYE} = \{4, 6\}$. For the component $\mathbf{M}_2$, $Q^{M_2}_{SPBS} = \{q_0\}$, $\Sigma^{M_2}_{SETE} = \{3, 9\}$ and $\Sigma^{M_2}_{SEYE} = \{8, 10\}$. Also, $\Sigma^{M_1}_{SINE} = \{1\}$, $\Sigma^{M_1}_{SELE} = \emptyset$, $\Sigma^{M_2}_{SINE} = \emptyset$, $\Sigma^{M_1}_{SELE} = \{7\}$. The two LBRS $\mathbf{R}^{M_1}_B$ and $\mathbf{R}^{M_2}_B$ are shown below.

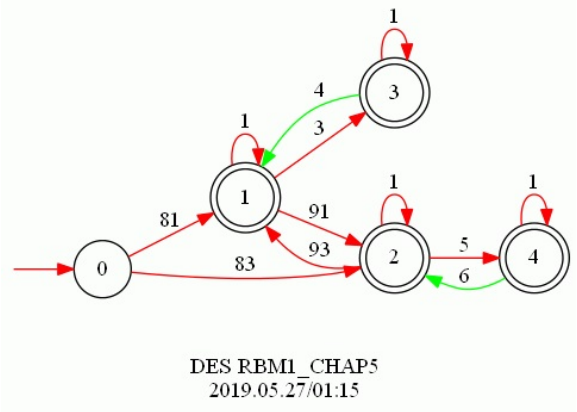

Figure A..12: LBRS $\mathbf{R}^{M_1}_B$

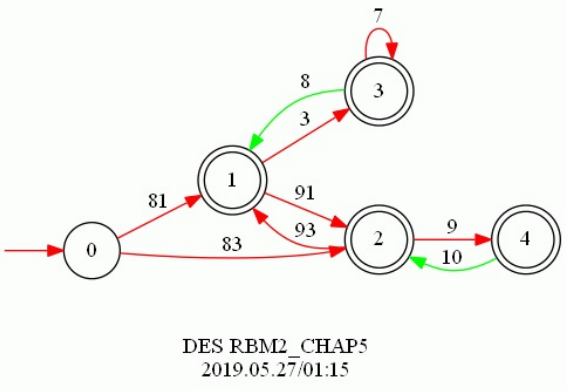

Figure A..13: LBRS $\mathbf{R}^{M_2}_B$

We then compute

$$\mathbf{RG}_B = \mathbf{Sync}(\mathbf{M}_1, \mathbf{M}_2, \mathbf{R}_B^{M_1}, \mathbf{R}_B^{M_2}) \text{ (16,31)}$$
$$\mathbf{true} = \mathbf{Isomorph}(\mathbf{RG}_B, \mathbf{RG}; \mathbf{identity})$$

According to the results from TCT, the reconfiguration plant obtained from the localized bidirectional reconfiguration approach is the same as that obtained from the localized multiple reconfiguration approach.

A..5 Summary and Discussion

First, this appendix has extended the monolithic multiple reconfiguration specification to the localized multiple reconfiguration specification, which can solve reconfiguration problems when exit (entry) events are shared in different plant components. In this approach, the core multiple reconfiguration specification doesn't change while the extra multiple reconfiguration specification has been redesigned for each plant component. This approach is proved to be a general version of the monolithic multiple reconfiguration approach. The localized bidirectional reconfiguration approach is also introduced briefly.

The newly defined localized extra multiple (bidirectional) reconfiguration specification is much simpler in structure; and in return the number of LEMRS is larger. This is one trade-off when applying the localized approach.

Appendix B

Proofs

B..1 Proofs in Chapter 3

In order to formally prove the correctness of the proposed BRS, some observations are needed first. For all the lemmas, propositions and theorems, we prove a stronger version when the system has more than two modes, say n modes. We emphasize it here to avoid confusion in proofs.

Lemma 1. $(\forall k = 1, ..., h)$ $[\Sigma^k_{SETE} = \emptyset \Rightarrow \Sigma^k_{SEYE} = \emptyset]$.

Proof. If for the plant component $\mathbf{G}^k$, $\Sigma^k_{SETE} = \emptyset$, every state in $\mathbf{G}^k$ is a strictly public state. According to the definition of SEYE, $\Sigma^k_{SEYE} = \emptyset$. □

Proposition 1. For the plant $\mathbf{G}$, $\Sigma^G_{SETE} = \emptyset \Rightarrow \Sigma^G_{SEYE} = \emptyset$.

Proof. If for the plant $\mathbf{G}$, $\Sigma^G_{SETE} = \emptyset$, every state in $\mathbf{G}$ is a strictly public state. According to the definition of SEYE, $\Sigma^G_{SEYE} = \emptyset$. □

Lemma 2. $(\forall k = 1, ..., h)$ $Q^k_{SPBS} \neq \emptyset$.

Proof. Since $(\forall k = 1, ..., h)$ $q^k_o \models (P^k_1 \wedge ... \wedge P^k_n)$, the initial state of each plant component is a strictly public state, hence $q^k_o \in Q^k_{SPBS}$. Therefore $Q^k_{SPBS} \neq \emptyset$. □

Proposition 2. For the plant $\mathbf{G}$, $Q^G_{SPBS} \neq \emptyset$.

Proof. On the one hand, when there is only one plant component, it is the plant $\mathbf{G}$, then $q^G_o \in Q^G_{SPBS}$.

On the other hand, since $\forall k = 1, ..., h, q_o^k \models (P_1^k \wedge ... \wedge P_n^k)$ and $\mathbf{G}$ is the synchronous product of all the plant components, then the initial state of $\mathbf{G}$ also satisfies the two AMP, hence $q_o^G \in Q_{SPBS}^G$. Therefore, $Q_{SPBS}^G \neq \emptyset$. □

Lemma 3. Under the assumptions in Definition 10, $(\forall k = 1, ..., h)(\forall s \in L(\mathbf{RG}))$ $\delta^k(q_o^k, P_{G^k}(s)) \in Q_{SPBS}^k$ iff $|s|_{SETE}^k = |s|_{SEYE}^k$ and $\delta^k(q_o^k, P_{G^k}(s)) \in Q_{PRS}^k$ iff $|s|_{SETE}^k = |s|_{SEYE}^k + 1$, where $P_{G^k} : \Sigma^{RG*} \to \Sigma^k$, and $|s|_{SETE}^k (|s|_{SETE}^k)$ is the number of occurrences of SETE (SEYE) defined in $\mathbf{G}^k$ in the string s.

Proof. Consider an arbitrary plant component $\mathbf{G}^k$ and an arbitrary string $s \in L(\mathbf{RG})$. Prove by induction on the length of string s.

(base case).

If $s = \epsilon$, then $\delta^k(q_o^k, P_{G^k}(s)) = q_o^k \in Q_{SPBS}^k$ and $|s|_{SETE}^k = |s|_{SEYE}^k = 0$.

If $s = \sigma \in \Sigma \cup \Sigma_{MIE} \cup \Sigma_{RE}$, when $\sigma \notin \Sigma^k$, σ cannot affect $\delta^k(q_o^k, P_{G^k}(s))$, $|s|_{SETE}^k$ or $|s|_{SEYE}^k$, so only $\sigma \in \Sigma^k$ is further considered here. When $\sigma \in \Sigma^k$, since $(\forall \sigma')$ $[\delta^{RG}(q_o^{RG}, \sigma')! \Rightarrow \sigma' \in \Sigma_{MIE} \subseteq \Sigma^{RG} - \Sigma^k]$, so $\delta^{RG}(q_o^{RG}, \sigma)\not!$, so the lemma holds trivially.

(inductive case).

Suppose that $\delta^k(q_o^k, P_{G^k}(s)) \in Q_{SPBS}^k$ and $|s|_{SETE}^k = |s|_{SEYE}^k = r$. Then consider an event $\sigma \in \Sigma^k \subseteq \Sigma$.

(i) If $\sigma \in \Sigma^k \cap \Sigma_{SETE}^G$ and $\delta^k(q_o^k, P_{G^k}(s\sigma))!$, then $\delta^k(q_o^k, P_{G^k}(s\sigma)) \in Q_{PRS}^k$ since $(\forall \sigma' \in \Sigma_{SETE}^G)$ $[\delta^k(q, \sigma')! \Rightarrow (q \in Q_{SPBS}^k \wedge \delta^k(q, \sigma') \in Q_{PRS}^k)]$. At the same time, $|s\sigma|_{SETE}^k = r + 1 = |s\sigma|_{SEYE}^k + 1$.

(ii) If $\sigma \in \Sigma^k \cap (\Sigma_{SELE}^G \cup \Sigma_{SEYE}^G)$, then $\delta^k(q_o^k, P_{G^k}(s\sigma))\not!$, since $(\forall \sigma' \in \Sigma_{SELE}^G \cup \Sigma_{SEYE}^G)$ $[\delta^k(q, \sigma')! \Rightarrow q \in Q_{PRS}^k]$ but $\delta^k(q_o^k, P_{G^k}(s)) \in Q_{SPBS}^k$.

(iii) If $\sigma \in \Sigma^k \cap \Sigma_{SINE}^G$ and $\delta^k(q_o^k, P_{G^k}(s\sigma))!$, then $\delta^k(q_o^k, P_{G^k}(s\sigma)) \in Q_{SPBS}^k$ since $(\forall \sigma' \in \Sigma_{SINE}^G)$ $[\delta^k(q, \sigma')! \Rightarrow (q \in Q_{SPBS}^k \wedge \delta^k(q, \sigma') \in Q_{SPBS}^k)]$. At the same time, $|s\sigma|_{SETE}^k = |s\sigma|_{SEYE}^k = r$ since a strictly inner event cannot affect $|s\sigma|_{SETE}^k$ or $|s\sigma|_{SEYE}^k$.

Suppose that $\delta^k(q_o^k, P_{G^k}(s)) \in Q_{SPBS}^k$ and $|s\sigma|_{SETE}^k = |s\sigma|_{SEYE}^k + 1 = r + 1$. Then consider an event $\sigma \in \Sigma^k \subseteq \Sigma$.

(i) If $\sigma \in \Sigma^k \cap (\Sigma^G_{SETE} \cup \Sigma^G_{SINE})$, then $\delta^k(q^k_o, P_{G^k}(s\sigma))\not!$, since $(\forall \sigma' \in \Sigma^G_{SETE} \cup \Sigma^G_{SINE})\ [\delta^k(q,\sigma')! \Rightarrow q \in Q^k_{SPBS}]$ but $\delta^k(q^k_o, P_{G^k}(s)) \in Q^k_{PRS}$.

(ii) If $\sigma \in \Sigma^k \cap \Sigma^G_{SELE}$ and $\delta^k(q^k_o, P_{G^k}(s\sigma))!$, then $\delta^k(q^k_o, P_{G^k}(s\sigma)) \in Q^k_{PRS}$ since $(\forall \sigma' \in \Sigma^G_{SELE})\ [\delta^k(q,\sigma')! \Rightarrow ((q \in Q^k_{PRS}) \wedge (\delta^k(q,\sigma') \in Q^k_{PRS}))]$. At the same time, $|s\sigma|^k_{SETE} = |s\sigma|^k_{SEYE} + 1 = r + 1$ since a strictly external event cannot affect $|s\sigma|^k_{SETE}$ or $|s\sigma|^k_{SEYE}$.

(iii) If $\sigma \in \Sigma^k \cap \Sigma^G_{SEYE}$ and $\delta^k(q^k_o, P_{G^k}(s\sigma))!$, then $\delta^k(q^k_o, P_{G^k}(s\sigma)) \in Q^k_{SPBS}$ since $(\forall \sigma' \in \Sigma^G_{SEYE})\ [\delta^k(q,\sigma')! \Rightarrow (q \in Q^k_{PRS} \wedge \delta^k(q,\sigma') \in Q^k_{SPBS})]$. At the same time, $|s\sigma|^k_{SETE} = |s\sigma|^k_{SEYE} = r + 1$.

Since the plant component and the string are arbitrary, the lemma holds. □

Remark. It is obvious that $(\forall k = 1, ..., h)(\forall s \in L(\mathbf{RG}))\ |s|^k_{SETE} \geq |s|^k_{SEYE}$.

Proposition 3. $(\forall s \in L(\mathbf{RG}))\ \delta(q_o, P_G(s)) \in Q^G_{SPBS}$ iff $|s|^G_{SETE} = |s|^G_{SEYE}$ and $\delta(q_o, P_G(s)) \in Q^G_{PRS}$ iff $|s|^G_{SETE} > |s|^G_{SEYE}$, where $P_G : \Sigma^{RG*} \to \Sigma^*$, and $|s|^G_{SETE}(|s|^G_{SEYE})$ is the number of occurrences of SETE (SEYE) defined in $\mathbf{G}$ in the string s.

Proof. On the one hand, when there is only one plant component, it is the plant $\mathbf{G}$. Then, $(\forall s \in L(\mathbf{RG}))\ \delta(q_o, P_G(s)) \in Q^G_{SPBS}$ iff $|s|^G_{SETE} = |s|^G_{SEYE}$ and $\delta(q_o, P_G(s)) \in Q^G_{PRS}$ iff $|s|^G_{SETE} = |s|^G_{SEYE} + 1$. Thus, Lemma 3 and this proposition are naturally true.

On the other hand, given the assumptions in Definition 10, since Lemma 3 is true, the synchronous product of all plant components also guarantees that $(\forall s \in L(\mathbf{RG}))$ $\delta(q_o, P_G(s)) \in Q^G_{SPBS}$ iff $|s|^G_{SETE} = |s|^G_{SEYE}$ and $\delta(q_o, P_G(s)) \in Q^G_{PRS}$ iff $|s|^G_{SETE} > |s|^G_{SEYE}$. Note that when $\delta(q_o, P_G(s)) \in Q^G_{PRS}$, it might be true that $|s|^G_{SETE} = |s|^G_{SEYE}+1$, but it is more common that $|s|^G_{SETE} = |s|^G_{SEYE} + t$, where t is the number of plant components that are not at strictly public states. □

Remark. It is obvious that $(\forall s \in L(\mathbf{RG}))\ |s|^G_{SETE} \geq |s|^G_{SEYE}$.

Lemma 4. Under the assumptions in Definition 10, $(\forall s \in L(\mathbf{RG}))\ |s|^G_{SETE} = |s|^G_{SEYE}$ iff $\delta^R(q^R_o, P_R(s)) \in (Q^R_M \cup q^R_o)$, where $P_R : \Sigma^{RG*} \to \Sigma^{R*}$.

Proof. Since all the strictly exit/entry events are included in the BRS $\mathbf{R}$, namely Σ^G_{SETE} $\dot{\cup}\Sigma^G_{SEYE} \subseteq \Sigma^R$, if $|s|^G_{SETE} = |s|^G_{SEYE}$, then according to the structure of $\mathbf{R}$, $\delta^R(q^R_o, P_R(s))$

$= (Q_M^R \cup q_o^R)$. Similarly, according to the structure of $\mathbf{R}$, $|s|_{SETE}^G > |s|_{SEYE}^G$ iff $\delta^R(q_o^R, P_R(s)) \in Q_E^R$. □

Proposition 4. $(\forall s \in L(\mathbf{RG}))\ \delta(q_o, P_G(s)) \in Q_{SPBS}^G$ iff $\delta^R(q_o^R, P_R(s)) \in (Q_M^R \cup q_o^R)$.

Proof. This statement is a logical combination of Proposition 3 and Lemma 4. Then it is automatically true according to Proposition 3 and Lemma 4. □

Lemma 5. Under the assumptions in Definition 10, $L(\mathbf{G}) \subseteq P_{R\to G}L(\mathbf{R})$, where $P_{R\to G} : \Sigma^{R*} \to \Sigma^{G*}$.

Proof. Prove by induction on the length of the string s.
(base case).

Consider a string $s \in L(\mathbf{RG})$. When $s = \epsilon$, the lemma trivially holds. When $s = \sigma \in \Sigma$, and $\delta(q_o, \sigma)!$, then the event σ can only be a SINE or a SETE. The event σ must be in one of the event alphabets for the modes involved in $\mathbf{G}$. If $\sigma \in \Sigma_i$, then $\sigma_i \in \Sigma_{MIE} \subseteq (\Sigma^R - \Sigma), \sigma \in P_{R\to G}L(\mathbf{R})$.
(inductive case).

Given a string $s \in L(\mathbf{G}) \cap P_{R\to G}L(\mathbf{R})$, then $(\exists s' \in L(\mathbf{R}), P_{R\to G}(s') = s)\ \delta^R(q_o^R, s')!$. Consider an event $\sigma \in \Sigma$.

(i) If $\sigma \in \Sigma_{SETE}^G$ and $\delta(q_o, s\sigma)!$, then $\delta^R(q_o^R, s') \in \{q_1^0, ..., q_n^0\}$. Suppose that $\delta^R(q_o^R, s') = q_i^0$ and $\sigma \in \Sigma_j$. If $i = j$, then $\delta^R(q_o^R, s'\sigma)! = q_i^1$, hence $s'\sigma \in L(\mathbf{R})$ and $s\sigma \in P_{R\to G}L(\mathbf{R})$, since $\sigma \in \Sigma^R \cap \Sigma$. If $i \neq j$, then $(\exists \sigma_{i,j} \in \Sigma_{RE})\ [\delta^R(q_o^R, s')! = q_i^0 \wedge \delta^R(q_o^R, s'\sigma_{i,j})! = q_j^0 \wedge \delta^R(q_o^R, s'\sigma_{i,j}\sigma)! = q_j^1]$, hence $s'\sigma_{i,j}\sigma \in L(\mathbf{R})$ and $s\sigma \in P_{R\to G}L(\mathbf{R})$, since $\sigma \in \Sigma^R \cap \Sigma$ and $\sigma_{i,j} \in \Sigma_{RE} \subseteq (\Sigma^R - \Sigma)$.

(ii) If $\sigma \in \Sigma_{SINE}^G$ and $\delta(q_o^R, s\sigma)!$, then $\delta^R(q_o^R, s') \in \{q_1^0, ..., q_n^0\}$. Suppose that $\delta^R(q_o^R, s') = q_i^0$ and $\sigma \in \Sigma_j$. If $i = j$, then $\delta^R(q_o^R, s'\sigma)! = q_i^0$, hence $s'\sigma \in L(\mathbf{R})$ and $s\sigma \in P_{R\to G}L(\mathbf{R})$, since $\sigma \in \Sigma^R \cap \Sigma$. If $i \neq j$, then $(\exists \sigma_{i,j} \in \Sigma_{RE})\ [\delta^R(q_o^R, s')! = q_i^0 \wedge \delta^R(q_o^R, s'\sigma_{i,j})! = q_j^0 \wedge \delta^R(q_o^R, s'\sigma_{i,j}\sigma)! = q_j^0]$, hence $s'\sigma_{i,j}\sigma \in L(\mathbf{R})$ and $s\sigma \in P_{R\to G}L(\mathbf{R})$, since $\sigma \in \Sigma^R \cap \Sigma$ and $\sigma_{i,j} \in \Sigma_{RE} \subseteq (\Sigma^R - \Sigma)$.

(iii) If $\sigma \in \Sigma_{SEYE}^G$ and $\delta(q_o^R, s\sigma)!$, then $\delta^R(q_o^R, s') \in Q_E^R$. Suppose that $\delta^R(q_o^R, s') = q_i^r$ $(0 < r \leq b)$ and $\sigma \in \Sigma_j$. If $i = j$, then $\delta^R(q_o^R, s'\sigma)! \in \{q_i^{r-1}, q_i^0\}$, hence $s'\sigma \in L(\mathbf{R})$

and $s\sigma \in P_{R\to G}L(\mathbf{R})$, since $\sigma \in \Sigma^R \cap \Sigma$. If $i \neq j$, then $\delta(q_o, s\sigma)!$, since the system is at the inner part and the SEYE $\sigma \in \Sigma_j$ cannot occur for sure.

(iv) If $\sigma \in \Sigma^G_{SELE}$ and $\delta(q_o, s\sigma)!$, then $\delta^R(q^R_o, s') \in Q^R_E$. Suppose that $\delta^R(q^R_o, s') = q^r_i$ $(0 < r \leq b)$ and $\sigma \in \Sigma_j$. If $i = j$, then $\delta^R(q^R_o, s'\sigma)! = q^r_i$, hence $s'\sigma \in L(\mathbf{R})$ and $s\sigma \in P_{R\to G}L(\mathbf{R})$, since $\sigma \in \Sigma^R \cap \Sigma$. If $i \neq j$, then $\delta(q_o, s\sigma)!$, since the system is at the inner part and the SELE $\sigma \in \Sigma_j$ cannot occur for sure.

Since $\Sigma = \Sigma_{SINE} \cup \Sigma_{SETE} \cup \Sigma_{SELE} \cup \Sigma_{SEYE}$, it can be concluded that $L(\mathbf{G}) \subseteq P_{R\to G}L(\mathbf{R})$.

□

Remark. It can be deduced that $L(\mathbf{G}) \subseteq P_{RG\to G}L(\mathbf{R})$, where $P_{RG\to G} : \Sigma^{RG*} \to \Sigma^{G*}$ since $\Sigma^{RG} = \Sigma^R$.

Proposition 5. Under the assumptions in Definition 10, $L_m(\mathbf{G}) \subseteq P_{R\to G}L_m(\mathbf{R})$.

Proof. In a BRS, every state except for the initial state is a marked state. Since the only events defined at the initial state of a BRS are the events in $\Sigma_{MIE} \subseteq (\Sigma^R - \Sigma)$, and all events in Σ are defined at marked states of BRS, then $P_{R\to G}L_m(\mathbf{R}) = P_{R\to G}L(\mathbf{R})$. According to Lemma 5, $L(\mathbf{G}) \subseteq P_{R\to G}L(\mathbf{R})$, and since $L_m(\mathbf{G}) \subseteq L(\mathbf{G})$, it is true that $L_m(\mathbf{G}) \subseteq L(\mathbf{G}) \subseteq P_{R\to G}L(\mathbf{R}) = P_{R\to G}L_m(\mathbf{R})$, namely $L_m(\mathbf{G}) \subseteq P_{R\to G}L_m(\mathbf{R})$. □

The following theorem formally demonstrates that the reconfiguration plant generated by the synchronization of a plant and its BRS preserves the dynamics of the plant.

Theorem 1. Denote by $\mathbf{G}$ the plant DES, and $\mathbf{R}$ the BRS constructed according to Definition 11. The resulting reconfiguration plant $\mathbf{RG} = \mathbf{G}||\mathbf{R}$ guarantees that $L(\mathbf{G}) = P_G(L(\mathbf{G})||L(\mathbf{R}))$ and $L_m(\mathbf{G}) = P_G(L_m(\mathbf{G})||L_m(\mathbf{R}))$, where $P_i : \Sigma^{RG*} \to \Sigma^{i*}, i = \mathbf{G}, \mathbf{R}$.

Proof. $L(\mathbf{RG}) = L(\mathbf{G})||L(\mathbf{R})$. According to Chapter 2, $P_G(L(\mathbf{G})||L(\mathbf{R})) = L(\mathbf{G}) \cap P^{-1}_{G0}(P_{R0}L(\mathbf{R}))$ and $P_R(L(\mathbf{G})||L(\mathbf{R})) = L(\mathbf{R}) \cap P^{-1}_{R0}(P_{G0}L(\mathbf{G}))$, where $P_{i0} : \Sigma^*_i \to (\Sigma^G \cap \Sigma^R)^*, i = \mathbf{G}, \mathbf{R}$. Since $\Sigma^{RG} = \Sigma^R$ and $\Sigma^G \cap \Sigma^R = \Sigma^G$, then $P_G(L(\mathbf{G})||L(\mathbf{R})) = L(\mathbf{G}) \cap P_G L(\mathbf{R})$. Since $P_G L(\mathbf{R}) = P_{R\to G}L(\mathbf{R})$ and $L(\mathbf{G}) \subseteq P_{R\to G}L(\mathbf{R})$, then $P_G(L(\mathbf{G})||L(\mathbf{R})) = L(\mathbf{G})$.

Similarly, $L_m(\mathbf{RG}) = L_m(\mathbf{G})||L_m(\mathbf{R})$. According to Chapter 2, $P_G(L_m(\mathbf{G})||L_m(\mathbf{R})) = L_m(\mathbf{G}) \cap P^{-1}_{G0}(P_{R0}L_m(\mathbf{R}))$ and $P_R(L_m(\mathbf{G})||L_m(\mathbf{R})) = L_m(\mathbf{R}) \cap P^{-1}_{R0}(P_{G0}L_m(\mathbf{G}))$. Since

$\Sigma^{RG} = \Sigma^R$ and $\Sigma^G \cap \Sigma^R = \Sigma^G$, then $P_G(L_m(\mathbf{G})||L_m(\mathbf{R})) = L_m(\mathbf{G}) \cap P_G L_m(\mathbf{R})$. Since $P_G L_m(\mathbf{R}) = P_{R\to G} L_m(\mathbf{R})$ and $L_m(\mathbf{G}) \subseteq P_{R\to G} L_m(\mathbf{R})$, then $P_G(L_m(\mathbf{G})||L_m(\mathbf{R})) = L_m(\mathbf{G})$. □

Theorem 1 shows that the reconfiguration plant will not lose any information about the plant, namely the reconfiguration approach preserves the dynamics of the plant. If it was not the case, the reconfiguration approach would be invalid automatically. Now we can go one step further to prove the correctness of the approach.

Formally, the following theorem is presented to show that the BRS constructed according to Definition 11 solves Problem 2 through the synchronization with the plant DES $\mathbf{G}$.

Theorem 2. [Correctness of BRS]. Denote by $\mathbf{G} = \mathbf{G}^1||...||\mathbf{G}^h$ the plant DES formed by synchronization, $\mathbf{Mode}_1$ and $\mathbf{Mode}_2$ are two different modes of $\mathbf{G}$ distinguished by event alphabets Σ_1 and Σ_2. Also denote by $\mathbf{R}$ the BRS constructed according to Definition 11. The resulting reconfiguration plant $\mathbf{RG} = \mathbf{G}||\mathbf{R}$. The MIE and RE are $\Sigma_{MIE} = \{\sigma_1, \sigma_2\}$ and $\Sigma_{RE} = \{\sigma_{1,2}, \sigma_{2,1}\}$. Given the assumptions in Definition 10, then $\mathbf{RG}$ satisfies all five requirements described in Problem 2.

Proof. It is evident that $\Sigma^R = \Sigma \dot\cup \Sigma_{MIE} \dot\cup \Sigma_{RE} = \Sigma^1 \cup ... \cup \Sigma^h \dot\cup \Sigma_{MIE} \dot\cup \Sigma_{RE}$. A natural projection is defined as $P_j : \Sigma^{RG*} \to \Sigma^{j*}, (j = \mathbf{G}, \mathbf{R}, \mathbf{G}^1, ..., \mathbf{G}^h)$, where $\Sigma^{RG} = \Sigma \cup \Sigma^R = \Sigma^R$. Thus, $P_G : \Sigma^{RG*} \to \Sigma^*$ and $P_R : \Sigma^{RG*} \to \Sigma^{R*}$, namely $P_R : \Sigma^{R*} \to \Sigma^{R*}$. At the same time, $p_G^{-1} : Pwr(\Sigma^*) \to Pwr(\Sigma^{R*})$ and $p_R^{-1} : Pwr(\Sigma^{R*}) \to Pwr(\Sigma^{R*})$ are two inverse projections. Since $\mathbf{RG} = \mathbf{G}||\mathbf{R}, L(\mathbf{RG}) = L(\mathbf{G})||L(\mathbf{R}) = P_G^{-1}L(\mathbf{G}) \cap P_R^{-1}L(\mathbf{R}) = P_G^{-1}L(\mathbf{G}) \cap L(\mathbf{R})$.

Besides, according to Chapter 3 of [1], it is true that $P_G L(\mathbf{RG}) = P_G(L(\mathbf{G})||L(\mathbf{R})) \subseteq L(\mathbf{G})$ and $P_R L(\mathbf{RG}) = P_R(L(\mathbf{G})||L(\mathbf{R})) \subseteq L(\mathbf{R})$. Similarly, $P_{G^k} L(\mathbf{RG}) \subseteq L(\mathbf{G}^k)$ for each plant component $\mathbf{G}^k$. Therefore, if a string $s \in L(\mathbf{RG})$, then $P_{G^k}(s) \in L(\mathbf{G}^k)$ and $P_R(s) \in L(\mathbf{R})$. The requirements are proved in the order: (i), (ii), (iii), (iv), (v).

(i) $(\forall \sigma_i = \sigma_1, \sigma_2)(\exists q \in Q^{RG})\ \delta^{RG}(q_o^{RG}, \sigma_i) = q$.

Here we prove a stronger statement in a system with multiple (≥ 2) modes, i.e.,

$$(\forall \sigma_i \in \Sigma_{MIE})(\exists q \in Q^{RG})\ \delta^{RG}(q_o^{RG}, \sigma_i) = q. \tag{B..1}$$

Consider an arbitrary $\sigma_i \in \Sigma_{MIE}$. Since $\Sigma_{MIE} \subseteq \Sigma^R - \Sigma$, if σ_i can occur in $\mathbf{R}$, it can also occur in $\mathbf{RG}$ (it cannot be blocked by transitions in $\mathbf{G}$). In $\mathbf{R}$, $\delta^R(q_o^R, \sigma) = q_i^0 \in Q_M^R$, if $\sigma = \sigma_i \in \Sigma_{MIE}$. At the same time, in $\mathbf{R}$, all events in $\Sigma^R - \Sigma_{MIE}$ are disabled to occur at the initial state. Therefore, after synchronization, $\delta^{RG}(q_o^{RG}, \sigma_i)!$, i.e. $(\exists q \in Q^{RG})\ \delta^{RG}(q_o^{RG}, \sigma_i) = q$.

(ii) (a) $(\forall \sigma_{i,j} = \sigma_{1,2}, \sigma_{2,1})(\forall s \in L(\mathbf{RG}))\ [\delta^{RG}(q_o^{RG}, s\sigma_{i,j})! \Rightarrow (\forall k = 1, ..., h)\delta^k(q_o^k, P_{G^k}(s)) \in Q_{SPBS}^k]$

Here we prove a stronger statement in a system with multiple (≥ 2) modes, i.e.,

$$\begin{aligned}&(\forall \sigma_{i,j} \in \Sigma_{RE})(\forall s \in L(\mathbf{RG}))\\ &[\delta^{RG}(q_o^{RG}, s\sigma_{i,j})! \Rightarrow (\forall k = 1, ..., h)\delta^k(q_o^k, P_{G^k}(s)) \in Q_{SPBS}^k].\end{aligned} \tag{B..2}$$

Note that $(\exists \sigma^{'} \in \Sigma_{RE})\ \delta^{RG}(q_o^{RG}, s\sigma^{'})!$ means that $\delta^R(q_o^R, s) = q_i^0$ for mode $\mathbf{Mode}_i$.

When s doesn't contain any reconfiguration event, i.e. $(\forall \sigma^{''} \in \Sigma^{RG})\ [s^{'}\sigma^{''}s^{''} = s \Rightarrow \sigma^{''} \in \Sigma_{RE}]$. If q_i^0 is directly reached from q_o^R, i.e. $\delta^R(q_o^R, \sigma_i) = q_i^0$, then $(\forall k = 1, ..., h)\ \delta^k(q_o^k, P_{G^k}(s)) = q_o^k \in Q_{SPBS}^k$ since $s = \sigma_i$. If q_i^0 is led to via s and $|s| \neq 1$, then there must be an equal number of occurrences of SETE and SEYE in Σ_i in s since $(\forall \sigma_1 \in \Sigma_i \cap \Sigma_{SETE})(\forall r \in 0, ..., b-1)\ \delta^R(q_i^r, \sigma_1) = q_i^{r+1}$ and $(\forall \sigma_2 \in \Sigma_i \cap \Sigma_{SEYE})(\forall r \in 0, ..., b-1)\ \delta^R(q_i^{r+1}, \sigma_2) = q_i^r$. Since there is an equal number of occurrences of SETE and SEYE, and SEYE is eligible only after a SETE has occurred in each component, for each SETE in s and also in Σ^k for an arbitrary $\mathbf{G}^k$, there must be a SEYE in Σ^k also in s, namely $|s|_{SETE}^k = |s|_{SEYE}^k$. According to Lemma 3, for all those components $\mathbf{G}^k$, $\delta^k(q_o^k, P_{G^k}(s)) \in Q_{SPBS}^k$. For each of the other components $\mathbf{G}^t$, it is also true that $\delta^t(q_o^t, P_{G^t}(s)) \in Q_{SPBS}^t$ since no SETE in $\mathbf{G}^t$ has occurred yet. Thus, $(\forall k = 1, ..., h)\ \delta^k(q_o^k, P_{G^k}(s)) \in Q_{SPBS}^k$ holds.

When s contains a reconfiguration event, say $\sigma_{j,i}$, where $j \neq i$, then s can be expressed as $s = s_1\sigma_{j,i}s_2$. Obviously $\delta^R(q_o^R, s_1) = q_j^0$, then $(\forall k = 1, ..., h)\ |s_1|_{SETE}^k = |s_1|_{SEYE}^k$ and $\delta^k(q_o^k, P_{G^k}(s_1)) \in Q_{SPBS}^k$. After the occurrence of $\sigma_{j,i}$, $\delta^R(q_o^R, s_1\sigma_{j,i}) = q_i^0$ and $(\forall k = 1, ..., h)\ \delta^k(q_o^k, P_{G^k}(s_1\sigma_{j,i})) \in Q_{SPBS}^k$ since $\sigma_{j,i}$ doesn't af-

fect any plant component. Similarly, if $(\delta^R(q_o^R, s_1\sigma_{j,i}s_2) = q_i^0) \wedge (\delta^R(q_o^R, s_1\sigma_{j,i}) = q_i^0)$, then $(\forall k = 1, ..., h)\ |s_2|^k_{SETE} = |s_2|^k_{SEYE}$. Thus, $(\forall k = 1, ..., h)\ |s_1\sigma_{j,i}s_2|^k_{SETE} = |s_1\sigma_{j,i}s_2|^k_{SEYE}$ and hence $(\forall k = 1, ..., h)\ \delta^k(q_o^k, P_{G^k}(s_1\sigma_{j,i}s_2)) \in Q^k_{SPBS}$. Furthermore, if s contains more than one reconfiguration event, it can be easily deduced that $(\forall k = 1, ..., h)\ \delta^k(q_o^k,\ P_{G^k}(s)) \in Q^k_{SPBS}$.

(b) $(\forall \sigma_{i,j} = \sigma_{1,2}, \sigma_{2,1})(\exists q_s, q_d \in Q^{RG}, q_s \neq q_d)\ \delta^{RG}(q_s, \sigma_{i,j}) = q_d$

Here we prove a stronger statement in a system with multiple (≥ 2) modes, i.e.,

$$(\forall \sigma_{i,j} \in \Sigma_{RE})(\exists q_s, q_d \in Q^{RG}, q_s \neq q_d)\ \delta^{RG}(q_s, \sigma_{i,j}) = q_d. \tag{B..3}$$

Consider an arbitrary $\sigma_{i,j} \in \Sigma_{RE}$. Since $\Sigma_{RE} \subseteq \Sigma^R - \Sigma$, if $\sigma_{i,j}$ can occur in **R**, it can also occur in **RG** (it cannot be blocked by transitions in **G**). In **R**, $\delta^R(q_i^0, \sigma_{i,j}) = q_j^0 \neq q_i^0$. Therefore, after synchronization, $(\forall \sigma_{i,j} \in \Sigma_{RE})(\exists q_s, q_d \in Q^{RG})\ \delta^{RG}(q_s, \sigma_{i,j})! = q_d$, and q_s, q_d can be distinguished by q_i^0 and q_j^0 in **R**.

(c) $(\forall \sigma_{i,j} = \sigma_{1,2}, \sigma_{2,1})(\forall q, q^{'} \in Q^{RG}, q \neq q^{'})\ [\delta^{RG}(q, \sigma_{i.j}) = q^{'} \Rightarrow (q \models P_i^{RG} \wedge q^{'} \models P_j^{RG})]$

Here we prove a stronger statement in a system with multiple (≥ 2) modes, i.e.,

$$\begin{aligned}&(\forall \sigma_{i,j} \in \Sigma_{RE})(\forall q, q^{'} \in Q^{RG}, q \neq q^{'})\\ &[\delta^{RG}(q, \sigma_{i,j}) = q^{'} \Rightarrow (q \models P_i^{RG} \wedge q^{'} \models P_j^{RG})].\end{aligned} \tag{B..4}$$

Consider $\delta^{RG}(q_o^{RG}, s) = q$ and $\delta^{RG}(q_o^{RG}, s\sigma_{i,j}) = q^{'}$. Suppose that $s = \sigma_k s_1 \sigma_{k,i} s_2$ and $s_1 \in \Sigma_k^*, s_2 \in \Sigma_i^*$. Since $\delta^{RG}(q_o^{RG}, \sigma_k s_1 \sigma_{k,i})!, \delta^R(q_o^R, \sigma_k s_1) = q_k^0$ and $(\forall r = 1, ..., h)\ \delta^r(q_o^r, P_{G^r}(\sigma_k s_1)) \in Q^r_{SPBS}$ according to (ii)(a). Thus, $(\forall r = 1, ..., h)\ \delta^r(q_o^r, P_{G^r}(\sigma_k s_1)) \models P_i^r$. According to assumption (ii) in Definition 10, each mode in each plant component is reachable by itself, so $(\exists s_3 \in \Sigma_i^*)$ such that $\delta(q_o, s_1) = \delta(q_o, s_3)$. It is natural that $\sigma_i s_3 s_2 \in P_G^{-1}L(\mathbf{G})$. Besides, since $s_3 s_2 \in \Sigma_i^* \cap L(\mathbf{G})$ and $L(\mathbf{G}) \subseteq P_G L(\mathbf{R})$ according to Lemma 4, then $\delta^R(q_i^0, s_3 s_2)!$, and hence $\delta^R(q_o^R, \sigma_i s_3 s_2)!$ i.e. $\sigma_i s_3 s_2 \in L(\mathbf{R})$. Since $L(\mathbf{RG}) = P_G^{-1}L(\mathbf{G}) \cap L(\mathbf{R}), \sigma_i s_3 s_2 \in L(\mathbf{RG})$. Since $\delta(q_o, s_1) = \delta(q_o, s_3), (\forall r = 1, ..., h)\ \delta^r(q_o^r, P_{G^r}(\sigma_i s_3)) \in Q^r_{SPBS}$. Since $s_3 \in \Sigma_i^*$, under this simple case in each component that involves $\mathbf{Mode}_i$, there must be an equal number of occurrences of SETE and SEYE in s_3, so

$\delta^R(q_o^R, \sigma_i s_3) = q_i^0$ according to the definition of BRS. Thus, $\delta^R(q_o^R, \sigma_i s_3) = \delta^R(q_o^R, \sigma_k s_1 \sigma_{k,i})$ and $\delta^R(q_o^R, \sigma_i s_3 s_2) = \delta^R(q_o^R, \sigma_k s_1 \sigma_{k,i} s_2)$. Since $\delta^R(q_o^R, \sigma_i s_3 s_2) = \delta^R(q_o^R, s)$ and $\delta^R(q_o^R, P_G(\sigma_i s_3 s_2)) = \delta^R(q_o^R, P_G(s)), \delta^{RG}(q_o^{RG}, s) = q = \delta^{RG}(q_o^{RG}, \sigma_i s_3 s_2)$. Since σ_i is the MIE for $\mathbf{Mode}_i$ and $s_3 s_2 \in \Sigma_i^*, q \models P_i^{RG}$. For a more complicated string such as $s = \sigma_p s_1 \sigma_{p,k} s_2 ... s_3 \sigma_{r,i} s_4$ where $s_4 \in \Sigma_i^*$, by a similar replacement procedure, it is also true that $q \models P_i^{RG}$. Similarly, $q^{'} \models P_j^{RG}$.

(iii) $(\forall \sigma_{i,j}, \sigma_{j,i} = \sigma_{1,2}, \sigma_{2,1})\ [\delta^{RG}(q, \sigma_{i,j})! \Rightarrow (\exists s \in L(RG))\delta^{RG}(q, \sigma_{i,j} s \sigma_{j,i})!]$

Here we prove a stronger statement in a system with multiple (≥ 2) modes, i.e.,

$$(\forall \sigma_{i,j}, \sigma_{j,i} \in \Sigma_{RE})(\forall q \in Q^{RG})\ [\delta^{RG}(q, \sigma_{i,j})! \Rightarrow (\exists s \in L(\mathbf{RG}))\delta^{RG}(q, \sigma_{i,j} s \sigma_{j,i})]. \tag{B..5}$$

Consider the string $s^{'} \in L(\mathbf{RG})$ such that $\delta^{RG}(q_o^{RG}, s^{'}) = q$, where the event $\sigma_{i,j}$ is eligible to occur, then $\delta^R(q_o^R, s^{'}) = q_i^0$. After the event $\sigma_{i,j}$ occurs, $\delta^R(q_o^R, s^{'}\sigma_{i,j}) = q_j^0$, where the event $\sigma_{j,i}$ is immediately eligible to occur. At that time, $s = \epsilon$. Consider $s \neq \epsilon$. If at q_j^0, there are some SINE defined, then after a string $s \in (\Sigma_{SINE} \cap \Sigma_j)^*, \sigma_{j,i}$ is still eligible to occur since $(\forall \sigma \in \Sigma_{SINE}^G)\ \delta^R(q, \sigma) = q$. On the other hand, if the string $s \in \Sigma_j^*$ also contains SETE and SEYE, if the numbers of SETE and SEYE are equal, then according to the structure of BRS, $\delta^R(q_k^0, s) = q_j^0$, hence $\delta^{RG}(q, \sigma_{i,j} s \sigma_{j,i})!$.

(iv) $(\forall i = 1, 2)(\forall q \models P_i^{RG})(\exists \sigma_i \in \Sigma_{MIE})(\exists s \in \Sigma_i^*)\ [\delta^{RG}(q_o^{RG}, \sigma_i s) = q \Rightarrow (\exists q_m \in Q_m^{RG})(\exists s^{'} \in \Sigma_i^*)(\delta^{RG}(q, s^{'}) = q_m \wedge q_m \models P_i^{RG})]$

Here we prove a stronger statement in a system with multiple (≥ 2) modes, i.e.,

$$\begin{aligned}&(\forall i = 1, ..., n)(\forall q \models P_i^{RG})(\exists \sigma_i \in \Sigma_{MIE})(\exists s \in \Sigma_i^*)\\ &[\delta^{RG}(q_o^{RG}, \sigma_i s) = q \Rightarrow (\exists q_m \in Q_m^{RG})(\exists s^{'} \in \Sigma_i^*)(\delta^{RG}(q, s^{'}) = q_m \wedge q_m \models P_i^{RG})].\end{aligned} \tag{B..6}$$

Since $\delta^{RG}(q_o^{RG}, \sigma_i s)!$ and $s \in \Sigma_i^*$, $\delta(q_o, s)!$. Given the assumption (ii) that each mode is coreachable in each plant component by itself, then $(\forall k = 1, ..., h)\ \delta^k(q_o, P_{G^k}(s))$ can lead to a marked state, i.e. $(\exists s^{k'} \in \Sigma_i^* \cap \Sigma^{k*})\ \delta^k(q_o^k, P_{G^k}(s) s^{k'}) \in Q_m^k$. Then it is natural that $(\exists s^{'} \in \Sigma_i^*)\ \delta(q_o, s s^{'})! \in Q_m$, where $(\forall k = 1, ..., h)\ P_{G^k}(s^{'}) = s^{k'}$. Since $s s^{'} \in L(\mathbf{G}) \subseteq P_G L(\mathbf{R})$ according to Lemma 5 and $s s^{'} \in \Sigma_i^*$, then $\delta^R(q_i^0, s s^{'})!$, hence $(\exists \sigma_i \in \Sigma_{MIE})\ \delta^R(q_o^R, \sigma_i s s^{'})!$. Furthermore, since $(\forall q \in Q^R, q \neq q_o^R)\ q \in Q_m^R$, and

$\delta(q_o, ss') \in Q_m, \delta^{RG}(q_o^{RG}, \sigma_i ss') \in Q_m^{RG}$, namely $(\exists q_m \in Q_m^{RG})(\exists s' \in \Sigma_i^*)\ \delta^{RG}(q, s') = q_m$. Obviously, since $\delta^{RG}(q, s') = \delta^{RG}(q_o^{RG}, ss') = q_m$ and $ss' \in \Sigma_i^*$, according to the definition of SMP, $q_m \models P_i^{RG}$.

(v) $(\forall q \in Q^{RG}, q \neq q_o^{RG})\ [(q \models P_1^{RG} \wedge q \not\models P_2^{RG}) \vee (q \models P_2^{RG} \wedge q \not\models P_1^{RG})]$

Here we prove a stronger statement in a system with multiple (≥ 2) modes, i.e.,

$$(\forall q \in Q^{RG}, q \neq q_o^{RG})(\exists i \in 1, ..., n)(\forall j \in 1, ..., n, i \neq j)\ [q \models P_i^{RG} \wedge q \not\models P_j^{RG}]. \tag{B..7}$$

(a) $(\forall q \in Q^{RG}, q \neq q_o^{RG})(\exists i \in 1, ..., n)\ q \models P_i^{RG}$.

Suppose that $\delta^{RG}(q_o^{RG}, s) = q$. If $(\exists \sigma_{i,j} \in \Sigma_{RE})\ \delta^{RG}(q, \sigma_{i,j})!$, then $(\forall k = 1, ..., h)\ \delta^r(q_o^k, P_{G^k}(s)) \in Q_{SPBS}^k$ according to (ii)(a). Thus, $(\forall k = 1, ..., h)\delta^k(q_o^k, P_{G^k}(s)) \models P_i^k$. According to assumption (ii), each mode in each plant component is reachable by itself, so $(\exists s_1 \in \Sigma_i^*)\ \delta(q_o, s_1) = \delta(q_o, P_G(s))$. It is natural that $\sigma_i s_1 \in P_G^{-1}L(\mathbf{G})$. Besides, since $s_1 \in \Sigma_i^* \cap L(\mathbf{G})$ and $L(\mathbf{G}) \subseteq P_G L(\mathbf{R})$ according to Lemma 5, then $\delta^R(q_i^0, s_1)!$, and hence $\delta^R(q_o^R, \sigma_i s_1)!$ i.e. $\sigma_i s_1 \in L(\mathbf{R})$. Since $L(\mathbf{RG}) = P_G^{-1}L(\mathbf{G}) \cap L(\mathbf{R})$, $\sigma_i s_1 \in L(\mathbf{RG})$. Since $\delta(q_o, s_1) = \delta(q_o, P_G(s)), (\forall k = 1, ..., h)\ \delta^k(q_o^k, P_{G^k}(\sigma_i s_1)) \in Q_{SPBS}^k$. Since $s_1 \in \Sigma_i^*$, under this simple case in each component that involves $\mathbf{Mode}_i$, there must be an equal number of occurrences of SETE and SEYE in s_1, so $\delta^R(q_o^R, \sigma_i s_1) = q_i^0$ according to the definition of BRS. Thus, $\delta^R(q_o^R, \sigma_i s_1) = \delta^R(q_o^R, s)$. Since $\delta^R(q_o^R, \sigma_i s_1) = \delta^R(q_o^R, s)$ and $\delta(q_o, P_G(\sigma_i s_1)) = \delta(q_o, P_G(s)), \delta^{RG}(q_o^{RG}, s) = q = \delta^{RG}(q_o^{RG}, \sigma_i s_1)$. Since σ_i is the MIE for $\mathbf{Mode}_i$ and $s_1 \in \Sigma_i^*, q \models P_i^{RG}$.

On the other hand, if $(\nexists\ \sigma_{i,j} \in \Sigma_{RE})\ \delta^{RG}(q, \sigma_{i,j})!$, then s can always be expressed as $s = s_1 s_2$ where $(\exists \sigma_{i,j} \in \Sigma_{RE})\ \delta^{RG}(q_o^{RG}, s_1 \sigma_{i,j})!$ and $s_2 \in \Sigma_i^*$. Then $\delta^{RG}(q_o^{RG}, s_1) \models P_i^{RG}$ based on the proof above. According to the definition of segregated mode predicate, since $s_2 \in \Sigma_i^*, \delta^{RG}(q_o^{RG}, s_1 s_2) \models P_i^{RG}$.

(b) $(\forall q \in Q^{RG}, q \neq q_o^{RG})(\exists i \in 1, ..., n)(\forall j \in 1, ..., n, i \neq j)\ [q \models P_i^{RG} \wedge q \not\models P_j^{RG}]$.

Prove by contradiction. Assume that $q \models P_i^{RG}$ and $q \models P_j^{RG}, i \neq j$. Then $s_1 \in \Sigma_i^*, \delta^{RG}(q_o^{RG}, \sigma_i s_1) = q$ and $(\exists s_2 \in \Sigma_j^*)\ \delta^{RG}(q_o^{RG}, \sigma_j s_2) = q$. Since events in both s_1 and s_2 need to occur in $\mathbf{R}$ according to synchronization, then nat-

urally $\delta^R(q_o^R, \sigma_i s_1) \in \{q_i^0, q_i^1, ..., q_i^{k_i}\}$ and $\delta^R(q_o^R, \sigma_j s_2) \in \{q_j^0, q_j^1, ..., q_j^{k_j}\}$. However, $\{q_i^0, q_i^1, ..., q_i^{k_i}\} \cap \{q_j^0, q_j^1, ..., q_j^{k_j}\} = \emptyset$, which contradicts $\delta^R(q_o^R, \sigma_i s_1) = \delta^{RG}(q_o^{RG}, \sigma_j s_2)$ according to the assumption.

□

Note that the reconfiguration approach we have proved is for dynamic reconfiguration of DES. The dynamic feature is not discussed in the proof. In fact, it is automatically satisfied by the proposed approach. All the system dynamics are included in the plant hence in the reconfiguration plant; and the reconfiguration events are defined at the states which map back to the shared states in the plant. Therefore, the reconfiguration is eligible as long as the system is at a strictly public state after several operations (events) without shutting the whole system down.

On the other hand, the system can still run after the occurrence of a reconfiguration event, since the system is at a strictly public state after the reconfiguration and each mode is nonblocking separately. Namely, the proposed bidirectional reconfiguration approach realizes the dynamic reconfiguration mechanism.

B..2 Proofs in Chapter 4

In order to formally prove the correctness of the proposed MRS, some observations are needed first.

Lemma 6. $(\forall k = 1, ..., h)(\forall i, j = 1, ..., n, i < j)\ [\Sigma^k_{ETE,i,j} = \emptyset \Rightarrow \Sigma^k_{EYE,i,j} = \emptyset]$.

Proof. If for the plant component $\mathbf{G}^k$, $\Sigma^k_{ETE,i,j} = \emptyset$, every state in $\mathbf{G}^k$ is a public state with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$. According to the definition of EYE, $\Sigma^k_{EYE,i,j} = \emptyset$. □

Proposition 6. For the plant $\mathbf{G}$, $(\forall i, j = 1, ..., n, i < j)\ [\Sigma^G_{ETE,i,j} = \emptyset \Rightarrow \Sigma^G_{EYE,i,j} = \emptyset]$.

Proof. If for the plant $\mathbf{G}$, $\Sigma^G_{ETE,i,j} = \emptyset$, every state in $\mathbf{G}$ is a public state with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$. According to the definition of EYE, $\Sigma^G_{EYE,i,j} = \emptyset$. □

Lemma 7. $(\forall k = 1, ..., h)(\forall i, j \in 1, ..., n)\ Q^k_{PBS,i,j} \neq \emptyset$.

Proof. Since $(\forall k = 1, ..., h)\ q_o^k \models (P_1^k \wedge ... \wedge P_n^k)$, the initial state of each plant component is a public state with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$, hence $q_o^k \in Q_{PBS,i,j}^k$. Therefore $Q_{PBS,i,j}^k \neq \emptyset$. □

Proposition 7. For the plant $\mathbf{G}$, $(\forall i, j \in 1, ..., n)\ Q_{PBS,i,j}^G \neq \emptyset$.

Proof. On the one hand, when there is only one plant component, it is the plant $\mathbf{G}$, then $q_o^G \in Q_{PBS,i,j}^G$. Therefore $Q_{PBS,i,j}^G \neq \emptyset$.

On the other hand, since $(\forall k = 1, ..., h)\ q_o^k \models (P_1^k \wedge ... \wedge P_n^k)$ and $\mathbf{G}$ is the synchronous product of all the plant components, then the initial state of $\mathbf{G}$ also satisfies all AMP, hence $q_o^G \in Q_{PBS,i,j}^{MG}$. Therefore, $Q_{SPBS}^G \neq \emptyset$. □

Lemma 8. Under the assumptions in definition 17, $(\forall k = 1, ..., h)(\forall i, j \in 1, ..., n)(\forall s \in L(\mathbf{RG}))\ \delta^k(q_o^k, P_{G^k}(s)) \in Q_{PBS,i,j}^k$ iff $|s|_{ETE,i,j}^k = |s|_{EYE,i,j}^k$ and $\delta^k(q_o^k, P_{G^k}(s)) \notin Q_{PBS,i,j}^k$ iff $|s|_{ETE,i,j}^k = |s|_{EYE,i,j}^k + 1$, where $P_{G^k} : \Sigma^{RG*} \to \Sigma^{k*}$, and $|s|_{ETE,i,j}^k(|s|_{EYE,i,j}^k)$ is the number of occurrences of ETE (EYE) defined in $\mathbf{G}^k$ in the string s.

Proof. Consider an arbitrary plant component $\mathbf{G}^k$, two arbitrary modes $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ and an arbitrary string $s \in L(\mathbf{RG})$. Prove by induction on the length of string s.

(base case).

If $s = \epsilon$, then $\delta^k(q_o^k, P_{G^k}(s)) = q_o^k \in Q_{PBS,i,j}^k$ and $|s|_{ETE,i,j}^k = |s|_{EYE,i,j}^k = 0$.

If $s = \sigma \in \Sigma \cup \Sigma_{MIE} \cup \Sigma_{RE}$, when $\sigma \notin \Sigma^k$, σ cannot affect $\delta^k(q_o^k, P_{G^k}(s))$, $|s|_{ETE,i,j}^k$ or $|s|_{EYE,i,j}^k$, so only $\sigma \in \Sigma^k$ should be further considered here. When $\sigma \in \Sigma^k$, since $(\forall \sigma')\ \delta^{RG}(q_o^{RG}, \sigma')! \Rightarrow \sigma' \in \Sigma_{MIE} \subseteq \Sigma^{RG} - \Sigma^k$, so $\delta^{RG}(q_o^{RG}, \sigma)\not!$, so the lemma holds trivially.

(inductive case).

Suppose that $\delta^k(q_o^k, P_{G^k}(s)) \in Q_{PBS,i,j}^k$ and $|s|_{ETE,i,j}^k = |s|_{EYE,i,j}^k = r$. Then consider an event $\sigma \in \Sigma^k \subseteq \Sigma$.

(i) If $\sigma \in \Sigma^k \cap \Sigma_{ETE,i,j}^G$ and $\delta^k(q_o^k, P_{G^k}(s\sigma))!$, then $\delta^k(q_o^k, P_{G^k}(s\sigma)) \notin Q_{PBS,i,j}^k$ since $(\forall \sigma' \in \Sigma_{ETE,i,j}^G)\ [\delta^k(q, \sigma')! \Rightarrow (q \in Q_{PBS,i,j}^k \wedge \delta^k(q, \sigma') \notin Q_{PBS,i,j}^k)]$. At the same time, $|s\sigma|_{ETE,i,j}^k = r + 1 = |s\sigma|_{EYE,i,j}^k + 1$.

(ii) If $\sigma \in \Sigma^k \cap (\Sigma^G_{ELE,i,j} \cup \Sigma^G_{EYE,i,j})$, then $\delta^k(q^k_o, P_{G^k}(s\sigma))\not!$, since $(\forall \sigma' \in \Sigma^G_{ELE,i,j} \cup \Sigma^G_{EYE,i,j})\ [\delta^k(q,\sigma')! \Rightarrow q \notin Q^k_{PBS,i,j}]$ but $\delta^k(q^k_o, P_{G^k}(s)) \in Q^k_{PBS,i,j}$.

(iii) If $\sigma \in \Sigma^k \cap \Sigma^G_{INE,i,j}$ and $\delta^k(q^k_o, P_{G^k}(s\sigma))!$, then $\delta^k(q^k_o, P_{G^k}(s\sigma)) \in Q^k_{PBS,i,j}$ since $(\forall \sigma' \in \Sigma^G_{INE,i,j})\ [\delta^k(q,\sigma')! \Rightarrow (q \in Q^k_{PBS,i,j} \wedge \delta^k(q,\sigma') \in Q^k_{PBS,i,j})]$. At the same time, $|s\sigma|^k_{ETE,i,j} = |s\sigma|^k_{EYE,i,j} = r$ since an inner event cannot affect $|s\sigma|^k_{ETE,i,j}$ or $|s\sigma|^k_{EYE,i,j}$.

Suppose that $\delta^k(q^k_o, P_{G^k}(s)) \in Q^k_{PBS,i,j}$ and $|s\sigma|^k_{ETE,i,j} = |s\sigma|^k_{EYE,i,j} + 1 = r + 1$. Then consider an event $\sigma \in \Sigma^k \subseteq \Sigma$.

(i) If $\sigma \in \Sigma^k \cap (\Sigma^G_{ETE,i,j} \cup \Sigma^G_{INE,i,j})$, then $\delta^k(q^k_o, P_{G^k}(s\sigma))\not!$, since $(\forall \sigma' \in \Sigma^G_{ETE,i,j} \cup \Sigma^G_{INE,i,j})\ [\delta^k(q,\sigma')! \Rightarrow q \in Q^k_{PBS,i,j}]$ but $\delta^k(q^k_o, P_{G^k}(s)) \notin Q^k_{PBS,i,j}$.

(ii) If $\sigma \in \Sigma^k \cap \Sigma^G_{ELE,i,j}$ and $\delta^k(q^k_o, P_{G^k}(s\sigma))!$, then $\delta^k(q^k_o, P_{G^k}(s\sigma)) \notin Q^k_{PBS,i,j}$ since $(\forall \sigma' \in \Sigma^G_{ELE,i,j})\ [\delta^k(q,\sigma')! \Rightarrow (q \notin Q^k_{PBS,i,j} \wedge \delta^k(q,\sigma') \notin Q^k_{PBS,i,j})]$. At the same time, $|s\sigma|^k_{ETE,i,j} = |s\sigma|^k_{EYE,i,j} + 1 = r + 1$ since an external event cannot affect $|s\sigma|^k_{ETE,i,j}$ or $|s\sigma|^k_{EYE,i,j}$.

(iii) If $\sigma \in \Sigma^k \cap \Sigma^G_{EYE,i,j}$ and $\delta^k(q^k_o, P_{G^k}(s\sigma))!$, then $\delta^k(q^k_o, P_{G^k}(s\sigma)) \in Q^k_{PBS,i,j}$ since $(\forall \sigma' \in \Sigma^G_{EYE,i,j})\ [\delta^k(q,\sigma')! \Rightarrow (q \notin Q^k_{PBS,i,j} \wedge \delta^k(q,\sigma') \in Q^k_{PBS,i,j})]$. At the same time, $|s\sigma|^k_{ETE,i,j} = |s\sigma|^k_{EYE,i,j} = r + 1$.

Since the plant component and the string are arbitrary, the lemma holds. □

Remark. It is obvious that $(\forall k = 1, ..., h)(\forall i, j \in 1, ..., n)(\forall s \in L(\mathbf{RG}))\ |s|^k_{ETE,i,j} \geq |s|^k_{EYE,i,j}$.

Proposition 8. $(\forall i, j \in 1, ..., n)(\forall s \in L(\mathbf{RG}))\ \delta(q_o, P_G(s)) \in Q^G_{PBS,i,j}$ iff $|s|^G_{ETE,i,j} = |s|^G_{EYE,i,j}$ and $\delta(q_o, P_G(s)) \notin Q^G_{PBS,i,j}$ iff $|s|^G_{ETE,i,j} > |s|^G_{EYE,i,j}$, where $P_G : \Sigma^{RG*} \to \Sigma^*$, and $|s|^G_{ETE,i,j}(|s|^G_{EYE,i,j})$ is the number of occurrences of ETE (EYE) defined in **G** in the string s.

Proof. On the one hand, when there is only one plant component, it is the plant **G**. Then, $(\forall i, j \in 1, ..., n)(\forall s \in L(\mathbf{RG}))\ \delta(q_o, P_G(s)) \in Q^G_{PBS,i,j}$ iff $|s|^G_{ETE,i,j} = |s|^G_{EYE,i,j}$ and $\delta(q_o, P_G(s)) \notin Q^G_{PBS,i,j}$ iff $|s|^G_{ETE,i,j} = |s|^G_{EYE,i,j} + 1$. Thus, it should satisfy Lemma 8 and this proposition naturally.

On the other hand, given the assumptions that for each pair of modes $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$, the exit events for distinct plant components should be distinct, the entry events for distinct plant components should be distinct, and an event cannot serve as both exit event and entry event with respect to the same pair of modes in different components, then since Lemma 8 is true, the synchronous product of all plant components can also guarantee that $(\forall i,j \in 1,...,n)(\forall s \in L(\mathbf{RG}))\ \delta(q_o, P_G(s)) \in Q^G_{PBS,i,j}$ iff $|s|^G_{ETE,i,j} = |s|^G_{EYE,i,j}$ and $\delta(q_o, P_G(s)) \notin Q^G_{PBS,i,j}$ iff $|s|^G_{ETE,i,j} > |s|^G_{EYE,i,j}$ [1]. □

Remark. It is obvious that $(\forall i,j \in 1,...,n)(\forall s \in L(\mathbf{RG}))\ |s|^G_{ETE,i,j} \geq |s|^G_{EYE,i,j}$.

Lemma 9. Under the assumptions in definition 17, $(\forall i,j \in 1,...,n)(\forall s \in L(\mathbf{RG}))$, $|s|^G_{ETE,i,j} = |s|^G_{EYE,i,j}$ iff $\delta^{R_{i,j}}(q_o^{R_{i,j}}, P_{R_{i,j}}(s)) = q_o^{R_{i,j}}$, where $P_{R_{i,j}} : \Sigma^{RG*} \to \Sigma^{R_{i,j}*}$.

Proof. Since all the exit/entry events with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ are included in the EMRS $\mathbf{R}_{i,j}$, namely $\Sigma^G_{ETE,i,j} \dot\cup \Sigma^G_{EYE,i,j} \subseteq \Sigma^{R_{i,j}}$, if $|s|^G_{ETE,i,j} = |s|^G_{EYE,i,j}$, then according to the structure of $\mathbf{R}_{i,j}$, $\delta^{R_{i,j}}(q_o^{R_{i,j}}, P_{R_{i,j}}(s)) = q_o^{R_{i,j}}$. Similarly, according to the structure of $\mathbf{R}_{i,j}$, $|s|^G_{ETE,i,j} > |s|^G_{EYE,i,j}$ iff $\delta^{R_{i,j}}(q_o^{R_{i,j}}, P_{R_{i,j}}(s)) = \{q_1^{R_{i,j}}, ..., q_{k_{i,j}}^{R_{i,j}}\}$. □

Proposition 9. $(\forall i,j \in 1,...,n)(\forall s \in L(\mathbf{RG}))\ \delta(q_o, P_G(s)) \in Q^G_{PBS,i,j}$ iff $\delta^{R_{i,j}}(q_o^{R_{i,j}}, P_{R_{i,j}}(s)) = q_o^{R_{i,j}}$.

Proof. It is a logical combination of Proposition 8 and Lemma 9, then it is automatically true according to Proposition 8 and Lemma 9. □

Lemma 10. Under the assumptions in Definition 17, $\forall s = s_1\sigma_1 \in L(\mathbf{G})$ if $\delta(q_0, s) \models P_{t_1} \wedge ... \wedge P_{t_s}$, then $(\forall s_1')\ [(P_G(s_1') = s_1 \wedge \delta^{R_c}(q_o^{R_c}, s_1'\sigma_1)!) \Rightarrow \delta^{R_c}(q_o^{R_c}, s_1'\sigma_1) \in \{q_{t_1}, ..., q_{t_s}\}]$.

Proof. Since $\delta(q_0, s) \models (P_{t_1} \wedge ... \wedge P_{t_s})$, according to the definition of AMP, $\sigma_1 \in \Sigma_{t_1} \cap ... \cap \Sigma_{t_s}$. Then according to the structure of $\mathbf{R}_c$, σ_1 is only defined at $\{q_{t_1}, ..., q_{t_s}\}$ in $\mathbf{R}_c$, then s_1' can be any sequence of events in Σ^{Rc} that can reach $\{q_{t_1}, ..., q_{t_s}\}$ since $\delta^{R_c}(q_o^{R_c}, s_1'\sigma_1)!$. Thus, $\delta^{R_c}(q_o^{R_c}, s_1'\sigma_1) \in \{q_{t_1}, ..., q_{t_s}\}$.

□

Lemma 11. Under the assumptions in Definition 17, $L(\mathbf{G}) \subseteq P_G L(\mathbf{R})$, where $P_G : \Sigma^{R*} \to \Sigma^{G*}$.

[1] when $\delta(q_o, P_G(s)) \notin Q^G_{PBS,i,j}$, it might be true that $|s|^G_{ETE,i,j} = |s|^G_{EYE,i,j} + 1$, but it is more common that $|s|^G_{ETE,i,j} = |s|^G_{EYE,i,j} + t$, where t is the number of plant components that are not at public states with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$.

Proof. Prove by induction on the length of the string s.
(base case).

Consider a string $s \in L(\mathbf{RG})$. When $s = \epsilon$, the lemma holds trivially. When $s = \sigma \in \Sigma$, and $\delta(q_o, \sigma)!$, since q_o is a PBS with respect to arbitrary two modes, then the event σ can only be an INE or an ETE with respect to arbitrary two modes, and it can never be an ELE or EYE. The event σ must be in one of the event alphabets for the modes involved in $\mathbf{G}$. Consider $\sigma \in \Sigma_i$, if σ is an INE with respect to two modes, then it is either defined at the initial state of some EMRS as an ETE, or is completely not defined in some EMRS. And since $\sigma_i \in \Sigma_{MIE} \subseteq (\Sigma^{R_c} - \Sigma_{i,j}), \forall i, j \in 1, ..., n$, $\sigma_i\sigma \in L(\mathbf{R}_{1,2}||...||\mathbf{R}_{1,n}||\mathbf{R}_{2,3}||...||\mathbf{R}_{n-1,n})$. Since $\sigma_i\sigma \in L(\mathbf{R}_c)$, and $\sigma_i \in \Sigma_{MIE} \subseteq (\Sigma^R - \Sigma)$, then $\sigma \in P_G L(\mathbf{R})$. The situation when σ is an ETE is the same.
(inductive case).

Given a string $s \in L(\mathbf{G}) \cap P_G L(\mathbf{R}), \exists s^{'} \in L(\mathbf{R}), P_G(s^{'}) = s, \delta^R(q_o^R, s^{'})!$, consider an event $\sigma \in \Sigma$. Suppose that the event σ serves as different class of events with respect to different pair of modes.

(i) If $\sigma \in \Sigma^G_{ETE,i,j}$ and $\delta(q_o, s\sigma)!$, then $\delta(q_o, s)! \in Q^G_{PBS,i,j}$. According to Proposition 8, $\delta(q_o, P_G(s^{'})) = \delta(q_o, s) \in Q^G_{PBS,i,j}$ iff $|s^{'}|^G_{ETE,i,j} = |s^{'}|^G_{EYE,i,j}$. According to Lemma 8, $|s^{'}|^G_{ETE,i,j} = |s^{'}|^G_{EYE,i,j}$ iff $\delta^{R_{i,j}}(q_o^{R_{i,j}}, P_{R_{i,j}}(s')) = q_o^{R_{i,j}}$. Then $\delta^{R_{i,j}}(q_o^{R_{i,j}}, P_{R_{i,j}}(s')) = q_o^{R_{i,j}}$. Therefore, $\delta^{R_{i,j}}(q_o^{R_{i,j}}, P_{R_{i,j}}(s')\sigma)! = \delta^{R_{i,j}}(q_o^{R_{i,j}}, P_{R_{i,j}}(s'\sigma))!$.

(ii) If $\sigma \in \Sigma^G_{INE,i,j}$ and $\delta(q_o, s\sigma)!$, then $\delta(q_o^R, s)! \in Q^G_{PBS,i,j}$. According to Proposition 8, $\delta(q_o, P_G(s^{'})) = \delta(q_o, s) \in Q^G_{PBS,i,j}$ iff $|s^{'}|^G_{ETE,i,j} = |s^{'}|^G_{EYE,i,j}$, and since $\Sigma^G_{INE,i,j}$ is not defined in $\mathbf{R}_{i,j}$, then σ is defined after synchronization, if it is defined in other components of the synchronous product.

(iii) If $\sigma \in \Sigma^G_{EYE,i,j}$ and $\delta(q_o, s\sigma)!$, then $\delta(q_o^R, s)! \notin Q^G_{PBS,i,j}$. According to Proposition 8, $\delta(q_o, P_G(s^{'})) = \delta(q_o, s) \in Q^G_{PBS,i,j}$ iff $|s^{'}|^G_{ETE,i,j} = |s^{'}|^G_{EYE,i,j}$, and according to Lemma 8, $|s^{'}|^G_{ETE,i,j} = |s^{'}|^G_{EYE,i,j}$ iff $\delta^{R_{i,j}}(q_o^{R_{i,j}}, P_{R_{i,j}}(s')) = q_o^{R_{i,j}}$, then $\delta^{R_{i,j}}(q_o^{R_{i,j}}, P_{R_{i,j}}(s')) = \{q_1^{R_{i,j}}, ..., q_{k_{i,j}}^{R_{i,j}}\}$. Therefore, $\delta^{R_{i,j}}(q_o^{R_{i,j}}, P_{R_{i,j}}(s')\sigma)! = \delta^{R_{i,j}}(q_o^{R_{i,j}}, P_{R_{i,j}}(s'\sigma))!$.

(iv) If $\sigma \in \Sigma^G_{ELE,i,j}$ and $\delta(q_o, s\sigma)!$, then $\delta(q_o^R, s)! \notin Q^G_{PBS,i,j}$. According to Proposition 8, $\delta(q_o, P_G(s^{'})) = \delta(q_o, s) \in Q^G_{PBS,i,j}$ iff $|s^{'}|^G_{ETE,i,j} = |s^{'}|^G_{EYE,i,j}$. Since $\Sigma^G_{ELE,i,j}$ is

not defined in $\mathbf{R}_{i,j}$, then σ is defined after synchronization, if it is defined in other components of the synchronous product.

For the CMRS, consider $s = s_1\sigma_1$ and assume that $\delta(q_0, s) \models P_{t_1} \wedge ... \wedge P_{t_s}$. It is true that $(\exists s_1^{'})\ P_G(s_1^{'}) = s_1$ and $\delta^{R_c}(q_o^{R_c}, s_1^{'}\sigma_1)!$. According to Lemma 10, $\delta^{R_c}(q_o^{R_c}, s_1^{'}\sigma_1) \in \{q_{t_1}, ..., q_{t_s}\}$. Then if $\sigma \in \Sigma_{t_1} \cup ... \cup \Sigma_{t_s}$, suppose that $\delta^{R_c}(q_o^{R_c}, s_1^{'}\sigma_1) = q_{t_i}$ and $\sigma \in \Sigma_{t_j}$, for the EMRS $\mathbf{R}_{t_i,t_j}$, $\delta^{R_{t_i,t_j}}(q_o^{R_{t_i,t_j}}, P_{R_{t_i,t_j}}(s_1^{'}\sigma_1)) = q_o^{R_{t_i,t_j}}$, at which σ_{t_i,t_j} is enabled. Thus, $s_1\sigma_1\sigma_{t_i,t_j}\sigma$ is enabled in $\mathbf{R}_c$. If $\sigma \notin \Sigma_{t_1} \cup ... \cup \Sigma_{t_s}$, then since $\delta^{R_c}(q_o^{R_c}, s_1^{'}\sigma_1) \in \{q_{t_1}, ..., q_{t_s}\}$, the assumption that each mode is reachable and coreachable by itself is violated. Thus, $s\sigma \in P_G L(\mathbf{R}_c)$.

Since $\Sigma = \Sigma^G_{ETE,i,j} \cup \Sigma^G_{INE,i,j} \cup \Sigma^G_{EYE,i,j} \cup \Sigma^G_{ELE,i,j}$, even if the event σ may serve as different types of events with respect to different pairs of modes, it is still eligible to occur in $\mathbf{R}$ if $\delta(q_o, s\sigma)!$, so $s^{'}\sigma \in L(\mathbf{R})$ and $P_G(s^{'}\sigma) \in P_G L(\mathbf{R})$. We can concluded that $L(\mathbf{G}) \subseteq P_G L(\mathbf{R})$.

□

Remark. It can be deduced that $L(\mathbf{G}) \subseteq P_{RG\to G} L(\mathbf{R})$, where $P_{RG\to G} : \Sigma^{RG*} \to \Sigma^{G*}$ since $\Sigma^{RG} = \Sigma^R$.

Proposition 10. Under the assumptions in Definition 17, $L_m(\mathbf{G}) \subseteq P_G L_m(\mathbf{R})$.

Proof. In a CMRS, every state except for the initial state is a marked state. In an EMRS, every state is a marked state. Since the only events defined at the initial state of a CMRS are the events in $\Sigma_{MIE} \subseteq (\Sigma^R - \Sigma)$, and all events in Σ are defined at marked states of CMRS, then we can get that $P_G L_m(\mathbf{R}) = P_G L(\mathbf{R})$.

According to Lemma 11, $L(\mathbf{G}) \subseteq P_G L(\mathbf{R})$, and since $L_m(\mathbf{G}) \subseteq L(\mathbf{G})$, it is true that $L_m(\mathbf{G}) \subseteq L(\mathbf{G}) \subseteq P_G L(\mathbf{R}) = P_G L_m(\mathbf{R})$, namely $L_m(\mathbf{G}) \subseteq P_G L_m(\mathbf{R})$. □

Theorem 3. Denote by $\mathbf{G}$ the plant DES, and $\mathbf{R}$ the MRS constructed according to Definition 20. The resulting reconfiguration plant $\mathbf{RG} = \mathbf{G}||\mathbf{R}$ guarantees that $L(\mathbf{G}) = P_G(L(\mathbf{G})||L(\mathbf{R}))$ and $L_m(\mathbf{G}) = P_G(L_m(\mathbf{G})||L_m(\mathbf{R}))$, where $P_i : \Sigma^{RG*} \to \Sigma^{i*}, i = \mathbf{G}, \mathbf{R}$.

Proof. $L(\mathbf{RG}) = L(\mathbf{G})||L(\mathbf{R})$. According to Chapter 2, $P_G(L(\mathbf{G})||L(\mathbf{R})) = L(\mathbf{G}) \cap P_{G0}^{-1}(P_{R0}L(\mathbf{R}))$ and $P_R(L(\mathbf{G})||L(\mathbf{R})) = L(\mathbf{R}) \cap P_{R0}^{-1}(P_{G0}L(\mathbf{G}))$, where $P_{i0} : \Sigma_i^* \to (\Sigma^G \cap \Sigma^R)^*, i = \mathbf{G}, \mathbf{R}$. Since $\Sigma^{RG} = \Sigma^R$ and $\Sigma^G \cap \Sigma^R = \Sigma^G$, then $P_G(L(\mathbf{G})||L(\mathbf{R})) = L(\mathbf{G}) \cap$

$P_G L(\mathbf{R})$. Since $P_G L(\mathbf{R}) = P_{R\to G} L(\mathbf{R})$ and $L(\mathbf{G}) \subseteq P_{R\to G} L(\mathbf{R})$, then $P_G(L(\mathbf{G})||L(\mathbf{R})) = L(\mathbf{G})$.

Similarly, $L_m(\mathbf{RG}) = L_m(\mathbf{G})||L_m(\mathbf{R})$. According to Chapter 2, $P_G(L_m(\mathbf{G})||L_m(\mathbf{R})) = L_m(\mathbf{G}) \cap P_{G0}^{-1}(P_{R0} L_m(\mathbf{R}))$ and $P_R(L_m(\mathbf{G})||L_m(\mathbf{R})) = L_m(\mathbf{R}) \cap P_{R0}^{-1}(P_{G0} L_m(\mathbf{G}))$. Since $\Sigma^{RG} = \Sigma^R$ and $\Sigma^G \cap \Sigma^R = \Sigma^G$, then $P_G(L_m(\mathbf{G})||L_m(\mathbf{R})) = L_m(\mathbf{G}) \cap P_G L_m(\mathbf{R})$. Since $P_G L_m(\mathbf{R}) = P_{R\to G} L_m(\mathbf{R})$ and $L_m(\mathbf{G}) \subseteq P_{R\to G} L_m(\mathbf{R})$, then $P_G(L_m(\mathbf{G})||L_m(\mathbf{R})) = L_m(\mathbf{G})$. □

Theorem 3 shows that the reconfiguration plant will not discard any information about the plant, namely the reconfiguration approach preserves the dynamics of the plant. If that were not the case, the reconfiguration approach would be invalid automatically. Now we go one step further to prove the correctness of the approach.

Theorem 4. [Correctness of MRS]. Denote by $\mathbf{G} = (Q, \Sigma, \delta, q_o, Q_m) = \mathbf{G}^1||...||\mathbf{G}^h$ the plant DES formed by synchronization, where $\mathbf{G}^k$ is the k^{th} component. $\mathbf{Mode}_1, ..., \mathbf{Mode}_n$ are n different modes of $\mathbf{G}$ distinguished by event alphabets $\Sigma_1, ..., \Sigma_n$. Also denote by $\mathbf{R}$ the MRS constructed according to Definition 20. The resulting reconfiguration plant $\mathbf{RG} = \mathbf{G}||\mathbf{R}$. Given the assumptions in definition 17, then $\mathbf{RG}$ satisfies all five requirements described in Problem 4.

Proof. Some necessary preliminaries need to be stated to help the proof. It is obvious that $\Sigma^R = \Sigma \cup \Sigma_{MIE} \cup \Sigma_{RE} = \Sigma^1 \cup ... \cup \Sigma^h \cup \Sigma_{MIE} \cup \Sigma_{RE} = \Sigma_1 \cup ... \cup \Sigma_n \cup \Sigma_{MIE} \cup \Sigma_{RE}$ The notion of natural projection is also needed. A natural projection is defined as $P_{G^k} : \Sigma^{RG*} \Rightarrow \Sigma_k^*, (k = \mathbf{G}, \mathbf{R}, \mathbf{G}^1, ..., \mathbf{G}^h, \mathbf{R}_{1,2}, ..., \mathbf{R}_{n,n-1})$, where $\Sigma^{RG} = \Sigma \cup \Sigma^R = \Sigma^R$. Thus, $P_G : \Sigma^{RG*} \to \Sigma^*$ and $P_R : \Sigma^{RG*} \Rightarrow \Sigma^{R*}$, namely $P_R : \Sigma^{R*} \to \Sigma^{R*}$. At the same time, $p_G^{-1} : Pwr(\Sigma^*) \to Pwr(\Sigma^{R*})$ and $p_R^{-1} : Pwr(\Sigma^{R*}) \to Pwr(\Sigma^{R*})$ are two inverse projections. Since $\mathbf{RG} = \mathbf{G}||\mathbf{R}, L(\mathbf{RG}) = L(\mathbf{G})||L(\mathbf{R}) = P_G^{-1}(L(\mathbf{G})) \cap P_R^{-1}(L(\mathbf{R})) = P_G^{-1}(L(\mathbf{G})) \cap L(\mathbf{R})$.

Besides, according to Chapter 3 of [1], it is true that $P_G(L(\mathbf{RG})) = P_G(L(\mathbf{G})||L(\mathbf{R})) \subseteq L(\mathbf{G})$ and $P_R(L(\mathbf{RG})) = P_R(L(\mathbf{G})||L(\mathbf{R})) \subseteq L(\mathbf{R})$. Similarly, $P_{G^i}(L(\mathbf{RG})) \subseteq L(\mathbf{G}^i)$ for each plant component $\mathbf{G}^i$. Therefore, if a string $s \in L(\mathbf{RG})$, then $P_{G^i}(s) \in L(\mathbf{G}^i)$ and $P_R(s) \in L(\mathbf{R})$. The requirements are proved in the order: (i), (ii), (iii), (iv), (v).

(i) $(\forall \sigma_i \in \Sigma_{MIE})(\exists q \in Q^{RG})\ \delta^{RG}(q_o^{RG}, \sigma_i) = q$.

Consider an arbitrary $\sigma_i \in \Sigma_{MIE}$. Since $\Sigma_{MIE} \subseteq \Sigma^R - \Sigma$, if σ_i can occur in $\mathbf{R}$, it can also occur in $\mathbf{RG}$ (it cannot be blocked by transitions in $\mathbf{G}$). In $\mathbf{R}$, σ_i is only defined in $\mathbf{R}_c$ but not in any EMRS. In $\mathbf{R}_c$, $\delta^{R_c}(q_o^{R_c}, \sigma_i) = q_i$. At the same time, in $\mathbf{R}_c$, all events in $\Sigma^R - \Sigma_{MIE}$ are disabled to occur at the initial state, and they will also be disabled at the initial state in $\mathbf{RG}$ after synchronization. Therefore, after synchronization, $\delta^{RG}(q_o^{RG}, \sigma_i)!$, i.e. $(\exists q \in Q^{RG})\ \delta^{RG}(q_o^{RG}, \sigma_i) = q$.

(ii) (a) $(\forall i, j \in 1, ..., n, i \neq j)(\forall s \in L(\mathbf{RG}))\ [\delta^{RG}(q_o^{RG}, s\sigma_{i,j})! \Rightarrow (\forall k = 1, ..., h)$ $\delta^k(q_o^k, P_{G^k}(s)) \in Q^k_{PBS,i,j}]$.

$\delta^{RG}(q_o^{RG}, s\sigma_{i,j})!$ means that $\delta^{R_c}(q_o^{R_c}, s) = q_i$ and $\delta^{R_{i,j}}(q_o^{R_{i,j}}, P_{R_{i,j}}(s)) = q_o^{R_{i,j}}$.

If q_i is directly reached by $q_o^{R_c}$ in $\mathbf{R}_c$, i.e. $\delta^{R_c}(q_o^{R_c}, \sigma_i) = q_i$, then $(\forall k = 1, ..., h)\ \delta^k(q_o^k, P_{G^k}(s)) = q_o^k \in Q^k_{PBS,i,j}$ since $s = \sigma_i$.

If q_i is reached by s and $|s| \neq 1$, then there must be an equal number of occurrences of ETE and EYE with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ in s since $(\forall \sigma_1 \in \Sigma^G_{ETE,i,j})(\forall t \in 0, ..., k_{i,j} - 1)\ \delta^{R_{i,j}}(q_t^{R_{i,j}}, \sigma_1) = q_{r+1}^{R_{i,j}}$ and $(\forall \sigma_2 \in \Sigma^G_{EYE,i,j})(\forall t \in 0, ..., k_{i,j} - 1)\ \delta^{R_{i,j}}(q_{r+1}^{R_{i,j}}, \sigma_2) = q_r^{R_{i,j}}$ and $\delta^{R_{i,j}}(q_o^{R_{i,j}}, P_{R_{i,j}}(s)) = q_0^{R_{i,j}}$. Since there is an equal number of occurrences of ETE and EYE with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$, and EYE is eligible only after an ETE has occurred in each component. For each ETE in s and also in Σ^k for an arbitrary $\mathbf{G}^k$, there must be an EYE in Σ^k also in s, namely $|s|^k_{EYE,i,j} = |s|^k_{ETE,i,j}$. According to Lemma 7, for all those component $\mathbf{G}^k$, $\delta^k(q_o^k, P_{G^k}(s)) \in Q^k_{PBS,i,j}$. For each of other components $\mathbf{G}^e$, it is also true that $\delta^e(q_o^e, P_{G^e}(s)) \in Q^e_{PBS,i,j}$ since no ETE in $\mathbf{G}^e$ has occurred yet. Thus, $(\forall i, j \in 1, ..., n, i \neq j)(\forall k = 1, ..., h)\ \delta^k(q_o^k, P_{G^k}(s)) \in Q^k_{PBS,i,j}$ holds.

(b) $(\forall \sigma_{i,j} \in \Sigma_{RE})(\exists q_s, q_d \in Q^{RG}, q_s \neq q_d)\ \delta^{RG}(q_s, \sigma_{i,j}) = q_d$.

Consider an arbitrary $\sigma_{i,j} \in \Sigma_{RE}$. Since $\Sigma_{RE} \subseteq \Sigma^R - \Sigma$, if $\sigma_{i,j}$ can occur in $\mathbf{R}$, it can also occur in $\mathbf{RG}$ (it cannot be blocked by transitions in $\mathbf{G}$). In $\mathbf{R}_c$, $\delta^{R_c}(q_i, \sigma_{i,j}) = q_j \neq q_i$. In $\mathbf{R}_{i,j}$, as long as there is an equal number of occurrences of ETE and EYE with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$, $\delta^{R_{i,j}}(q_o^{R_{i,j}}, \sigma_{i,j}) = q_o^{R_{i,j}}$. Therefore, after synchronization, $(\forall \sigma_{i,j} \in \Sigma_{RE})(\exists q_s, q_d \in Q^{RG})\ \delta^{RG}(q_s, \sigma_{i,j})! = q_d$, and q_s, q_d can be distinguished by q_i and q_j in $\mathbf{R}_c$.

(c) $(\forall \sigma_{i,j} \in \Sigma_{RE})(\forall q, q' \in Q^{RG}, q \neq q')\ [\delta^{RG}(q, \sigma_{i,j}) = q' \Rightarrow (q \models P_i^{RG} \wedge q' \models P_j^{RG})]$.

Consider $\delta^{RG}(q_o^{RG}, s) = q$ and $\delta^{RG}(q_o^{RG}, s\sigma_{i,j}) = q^{'}$. Suppose that $s = \sigma_k s_1 \sigma_{k,i} s_2$ and $s_1 \in \Sigma_k^*, s_2 \in \Sigma_i^*$. Since $\delta^{RG}(q_o^{RG}, \sigma_k s_1 \sigma_{k,i})!, \delta^{R_c}(q_o^{R_c}, \sigma_k s_1) = q_k$ and $(\forall r = 1, ..., h)\ \delta^r(q_o^r, P_{G^r}(\sigma_k s_1)) \in Q^r_{PBS,k,i}$ according to (ii)(a). Thus, $(\forall r = 1, ..., h)\ \delta^r(q_o^r, P_{G^r}(\sigma_k s_1)) \models P_i^r$.

According to assumption (ii) in Definition 17, each mode in each plant component is reachable by itself, so $\exists s_3 \in \Sigma_i^*$ such that $\delta(q_o, s_1) = \delta(q_o, s_3)$. It is natural that $\sigma_i s_3 s_2 \in P_G^{-1}(L(\mathbf{G}))$. Besides, since $s_3 s_2 \in \Sigma_i^* \cap L(\mathbf{G})$ and $L(\mathbf{G}) \subseteq P_G(L(\mathbf{R}))$ according to Lemma 11, then $\delta^R(q_o^R, \sigma_i s_3 s_2)!$ i.e. $\sigma_i s_3 s_2 \in L(\mathbf{R})$. Since $L(\mathbf{RG}) = P_G^{-1}(L(\mathbf{G})) \cap L(\mathbf{R}), \sigma_i s_3 s_2 \in L(\mathbf{RG})$. Since $\delta(q_o, s_1) = \delta(q_o, s_3)$, then $(\forall r = 1, ..., h)\ \delta^r(q_o^r, P_{G^r}(\sigma_i s_3)) \in Q^r_{PBS,k,i}$. Since $s_3 \in \Sigma_i^*$, $\delta^{R_c}(q_o^{R_c}, \sigma_i s_3) = q_i$ according to the definition of CMRS. Thus, $\delta^R(q_o^R, \sigma_i s_3) = \delta^R(q_o^R, \sigma_k s_1 \sigma_{k,i})$ and $\delta^R(q_o^R, \sigma_i s_3 s_2) = \delta^R(q_o^R, \sigma_k s_1 \sigma_{k,i} s_2)$. Since $\delta^R(q_o^R, \sigma_i s_3 s_2) = \delta^R(q_o^R, s)$ and $\delta^R(q_o^R, P_G(\sigma_i s_3 s_2)) = \delta^R(q_o^R, P_G(s)), \delta^{RG}(q_o^{RG}, s) = q = \delta^{RG}(q_o^{RG}, \sigma_i s_3 s_2)$. Since σ_i is the MIE for $\mathbf{Mode}_i$ and $s_3 s_2 \in \Sigma_i^*, q \models P_i^{RG}$. For a more complicated string such as $s = \sigma_p s_1 \sigma_{p,k} s_2 ... s_3 \sigma_{r,i} s_4$ where $s_4 \in \Sigma_i^*$, by a similar replacement procedure, it is still true that $q \models P_i^{RG}$. Similarly, $q^{'} \models P_j^{RG}$.

(iii) $(\forall \sigma_{i,j}, \sigma_{j,i} \in \Sigma_{RE})(\forall q \in Q^{RG})\ [\delta^{RG}(q, \sigma_{i,j})! \Rightarrow (\exists s \in L(\mathbf{RG}))\delta^{RG}(q, \sigma_{i,j} s \sigma_{j,i})!]$.

Consider the string $s^{'} \in L(\mathbf{RG})$ such that $\delta^{RG}(q_o^{RG}, s^{'}) = q$, where the event $\sigma_{i,j}$ is eligible to occur, then $\delta^{R_c}(q_o^{R_c}, s^{'}) = q_i$ and $\delta^{R_{i,j}}(q_o^{R_{i,j}}, s^{'}) = q_o^{R_{i,j}}$. According to Proposition 9, $\delta(q_o, P_G(s^{'})) \in Q^G_{PBS,i,j}$ since $\delta^{R_{i,j}}(q_o^{R_{i,j}}, P_{R_{i,j}}(s)) = q_o^{R_{i,j}}$. After the event $\sigma_{i,j}$ occurs, $\delta^{R_c}(q_o^{R_c}, s^{'}\sigma_{i,j}) = q_j$, where the event $\sigma_{j,i}$ is immediately eligible to occur. Besides, since $\delta^{R_{i,j}}(q_o^{R_{i,j}}, s^{'}\sigma_{i,j}) = q_o^{R_{i,j}}$, the RE $\sigma_{j,i}$ is also eligible to occur in $\mathbf{R}_{i,j}$ according to the structure of EMRS. Since $\sigma_{j,i}$ is not defined in any other EMRS, $\sigma_{j,i}$ is eligible to occur. At that time, $s = \epsilon$.

Consider $s \neq \epsilon$. If at q_j of $\mathbf{R}_c$, there are some INE with respect to $\mathbf{Mode}_j$ and $\mathbf{Mode}_i$ defined, then after a string $s \in (\Sigma_{INE,j,i} \cap \Sigma_j)^*, \sigma_{j,i}$ is still eligible to occur since $(\forall \sigma \in \Sigma_{INE,j,i})\ \delta^{R_{i,j}}(q_o^{R_{i,j}}, \sigma)\not!$. On the other hand, if the string $s \in \Sigma_j^*$ also contains ETE and EYE with respect to $\mathbf{Mode}_j$ and $\mathbf{Mode}_i$, if the numbers of ETE and EYE are equal, then according to the structure of EMRS, $\delta^{R_{i,j}}(q_o^{R_{i,j}}, P_{R_{i,j}}(s)) = q_o^{R_{i,j}}$, hence $\delta^{RG}(q, \sigma_{i,j} s \sigma_{j,i})!$.

(iv) $(\forall i \in 1,...,n)(\forall q \models P_i^{RG})(\exists \sigma_i \in \Sigma_{MIE})(\exists s \in \Sigma_i^*)\ [\delta^{RG}(q_o^{RG}, \sigma_i s) = q \Rightarrow (\exists q_m \in Q_m^{RG})(\exists s^{'} \in \Sigma_i^*)(\delta^{RG}(q, s^{'}) = q_m \wedge q_m \models P_i^{RG})]$.

Since $\delta^{RG}(q_o^{RG}, \sigma_i s)!$ and $s \in \Sigma_i^*$, $\delta(q_o, s)!$. Given the assumption (ii) that each mode is coreachable in each plant component by itself, then $(\forall k = 1,...,h)\ \delta^k(q_o, P_{G^k}(s))$ can lead to a marked state, i.e. $(\exists s^{k'} \in \Sigma_i^* \cap \Sigma^{k*})\ \delta^k(q_o^k, P_{G^k}(s)s^{k'}) \in Q_m^k$. Then it is natural that $(\exists s^{'} \in \Sigma_i^*)\ \delta(q_o, ss^{'})! \in Q_m$, where $(\forall k = 1,...,h)\ P_{G^k}(s^{'}) = s^{k'}$. Since $ss^{'} \in L(\mathbf{G}) \subseteq P_G(L(\mathbf{R}))$ according to Lemma 11 and $ss^{'} \in \Sigma_i^*$, then $\delta^{R_c}(q_o^{R_c}, ss^{'})!$, hence $(\exists \sigma_i \in \Sigma_{MIE})\ \delta^{R_c}(q_o^{R_c}, \sigma_i ss^{'})!$. Furthermore, since $(\forall q \in Q^R, q \neq q_o^R)\ q \in Q_m^R$, and $\delta(q_o, ss^{'}) \in Q_m, \delta^{RG}(q_o^{RG}, \sigma_i ss^{'}) \in Q_m^{RG}$, namely $(\exists q_m \in Q_m^{RG})(\exists s^{'} \in \Sigma_i^*)\ \delta^{RG}(q, s^{'}) = q_m$. Obviously, since $\delta^{RG}(q, s^{'}) = \delta^{RG}(q_o^{RG}, ss^{'}) = q_m$ and $ss^{'} \in \Sigma_i^*$, according to the definition of SMP, $q_m \models P_i^{RG}$.

(v) $(\forall q \in Q^{RG}, q \neq q_o^{RG})(\exists i \in 1,...,n)(\forall j \in 1,...,n, i \neq j)\ [q \models P_i^{RG} \wedge q \not\models P_j^{RG}]$.

(a) $(\forall q \in Q^{RG}, q \neq q_o^{RG})(\exists i \in 1,...,n)\ q \models P_i^{RG}$.

Suppose that $\delta^{RG}(q_o^{RG}, s) = q$. If $(\exists \sigma_{i,j} \in \Sigma_{RE})\ \delta^{RG}(q, \sigma_{i,j})!$, then $(\forall k = 1,...,h)\ \delta^k(q_o^k, P_{G^r}(s)) \in Q_{PBS,i,j}^k$ according to (ii)(a). Thus, $(\forall k = 1,...,h)$ $\delta^k(q_o^k, P_{G_k}(s)) \models P_i^k$. It is also true that $\delta^G(q_o^G, P_G(s)) \in Q_{PBS,i,j}^G$, and $\delta^{R_{i,j}}(q_o^{R_{i,j}}, P_{R_{i,j}}(s)) = q_o^{R_{i,j}}$ according to Proposition 9. Besides, $\delta^{R_c}(q_o^{R_c}, s) = q_i$. According to assumption (ii), each mode in each plant component is reachable by itself, so $\exists s_1 \in \Sigma_i^*$ such that $\delta(q_o, s_1) = \delta(q_o, P_G(s))$. It is natural that $\sigma_i s_1 \in P_G^{-1}(L(\mathbf{G}))$. Besides, since $s_1 \in \Sigma_i^* \cap L(\mathbf{G})$ and $L(\mathbf{G}) \subseteq P_G(L(\mathbf{R}))$ according to Lemma 11, then $\delta^{R_c}(q_o^{R_c}, \sigma_i s_1)!$ i.e. $\sigma_i s_1 \in L(\mathbf{R})$. Since $L(\mathbf{RG}) = P_G^{-1}(L(\mathbf{G})) \cap L(\mathbf{R})$, $\sigma_i s_1 \in L(\mathbf{RG})$. Since $\delta(q_o, s_1) = \delta(q_o, P_G(s))$, then $(\forall k = 1,...,h)\ \delta^k(q_o^k, P_{G^r}(\sigma_i s_1)) \in Q_{PBS,i,j}^k$. Since $s_1 \in \Sigma_i^*$, $\delta^{R_c}(q_o^{R_c}, \sigma_i s_1) = q_i$. In this simple case in each component that involves $\mathbf{Mode}_i$, there must be an equal number of occurrences of ETE and EYE with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ in s_1, so $\delta^{R_{i,j}}(q_o^{R_{i,j}}, \sigma_i s_1) = q_o^{R_{i,j}}$ according to the definition of EMRS. Thus, $\delta^R(q_o^R, \sigma_i s_1) = \delta^R(q_o^R, s)$. Since $\delta^R(Q_O^r, \sigma_i s_1) = \delta^R(q_o^R, s)$ and $\delta(q_o, P_G(\sigma_i s_1)) = \delta(q_o, P_G(s))$, $\delta^{RG}(q_o^{RG}, s) = q = \delta^{RG}(q_o^{RG}, \sigma_i s_1)$. Since σ_i is the MIE for $\mathbf{Mode}_i$ and $s_1 \in \Sigma_i^*, q \models P_i^{RG}$.

On the other hand, if $(\nexists\ \sigma_{i,j} \in \Sigma_{RE})\ \delta^{RG}(q, \sigma_{i,j})!$, then s can always be expressed as $s = s_1 s_2$ where $(\exists \sigma_{i,j} \in \Sigma_{RE})\ \delta^{RG}(q_o^{RG}, s_1 \sigma_{i,j})!$ and $s_2 \in \Sigma_i^*$. Then

$\delta^{RG}(q_o^{RG}, s_1) \models P_i^{RG}$ based on the proof above. According to the definition of segregated mode predicate, since $s_2 \in \Sigma_i^*, \delta^{RG}(q_o^{RG}, s_1 s_2) \models P_i^{RG}$.

(b) $(\forall q \in Q^{RG}, q \neq q_o^{RG})(\exists i \in 1, ..., n)(\forall j \in 1, ..., n, i \neq j)\ [q \models P_i^{RG} \wedge q \not\models P_j^{RG}]$.

Prove by contradiction. Assume that $q \models P_i^{RG}$ and $q \models P_j^{RG}, i \neq j$. Then $s_1 \in \Sigma_i^*, \delta^{RG}(q_o^{RG}, \sigma_i s_1) = q$ and $(\exists s_2 \in \Sigma_j^*)\ \delta^{RG}(q_o^{RG}, \sigma_j s_2) = q$. Since events in both s_1 and s_2 need to occur in $\mathbf{R}_c$ according to synchronization, then naturally $\delta^{R_c}(q_o^{R_c}, \sigma_i s_1) = q_i$ and $\delta^{R_c}(q_o^{R_c}, \sigma_j s_2) = q_j$. However, $\{q_i\} \cap \{q_j\} = \emptyset$, which contradicts $\delta^{R_c}(q_o^{R_c}, \sigma_i s_1) = \delta^{R_c}(q_o^{R_c}, \sigma_j s_2)$ according to the assumption.

□

Similar to Chapter 3, the dynamic feature is not discussed above and it is automatically satisfied by the proposed approach. Moreover, this approach realizes the multiple reconfiguration mechanism.

In fact, for a system with only two modes, the RG in Chapter 3 is the same as the RG in Chapter 5. Here we are going to present a theorem to show this fact and provide a proof. Before the theorem, two lemmas need to be given to reveal the equivalence between SPBS and PBS, and between SETE(EYE) and ETE(EYE) when the system only has two modes.

Lemma 12. Denote by $\mathbf{G}$ a plant DES with two modes $\mathbf{Mode}_1$ and $\mathbf{Mode}_2$ distinguished by event alphabets Σ_1 and Σ_2. Then $(\forall k = 1, ..., h)\ Q_{SPBS}^k = Q_{PBS,1,2}^k$.

Proof. In $\mathbf{G}^k$, for an arbitrary state $q \in Q_{PBS,1,2}^k$, it is true that $q \models (P_1^k \wedge P_2^k)$. Since there are only two modes in $\mathbf{G}^k$, it is also true that $q \in Q_{SPBS}^k$. □

With Lemma 12, Lemma 13 can be simply derived.

Lemma 13. Denote by $\mathbf{G}$ a plant DES with two modes $\mathbf{Mode}_1$ and $\mathbf{Mode}_2$ distinguished by event alphabets Σ_1 and Σ_2. Then $\forall k = 1, ..., h$

(i) $\Sigma_{SETE}^k = \Sigma_{ETE,1,2}^k$;

(ii) $\Sigma_{SEYE}^k = \Sigma_{EYE,1,2}^k$;

(iii) $\Sigma_{SELE}^k = \Sigma_{ELE,1,2}^k$;

(iv) $\Sigma^k_{SINE} = \Sigma^k_{INE,1,2}$.

Proof. (i) $\Sigma^k_{SETE} = \Sigma^k_{ETE,1,2}$

According to Definition 6, the source state of a SETE is a SPBS, so the source state is also a PBS with respect to $\mathbf{Mode}_1$ and $\mathbf{Mode}_2$ according to Lemma 12. Besides, the target state of a SETE doesn't satisfy $P^k_1 \wedge P^k_2$. Since there are just two AMP in the system, the target state of a SETE satisfies only one AMP. According to the definition of ETE, the target state of an ETE with respect to $\mathbf{Mode}_1$ and $\mathbf{Mode}_2$ doesn't satisfy $P^k_1 \wedge P^k_2$ either, hence satisfies only one AMP. Since the source state and the target state of events in Σ^k_{SETE} and $\Sigma^k_{ETE,1,2}$ belong to the same class, it is true that $\Sigma^k_{SETE} = \Sigma^k_{ETE,1,2}$.

(ii) $\Sigma^k_{SEYE} = \Sigma^k_{EYE,1,2}$

According to Definition 7, the target state of a SEYE is a SPBS, so the target state is also a PBS with respect to $\mathbf{Mode}_1$ and $\mathbf{Mode}_2$ according to Lemma 12. Besides, the source state of a SEYE doesn't satisfy $P^k_1 \wedge P^k_2$. Since there are just two AMP in the system, the source state of a SEYE satisfies only one AMP. According to the definition of EYE, the source state of an EYE with respect to $\mathbf{Mode}_1$ and $\mathbf{Mode}_2$ doesn't satisfy $P^k_1 \wedge P^k_2$ either, hence satisfies only one AMP. Since the source state and the target state of events in Σ^k_{SEYE} and $\Sigma^k_{EYE,1,2}$ belong to the same class, it is true that $\Sigma^k_{SEYE} = \Sigma^k_{EYE,1,2}$.

(iii) $\Sigma^k_{SELE} = \Sigma^k_{ELE,1,2}$

According to Definition 8, either the source state or the target state of a SELE is an SPRS, which doesn't satisfy both P^k_1 and P^k_2. Besides, the source/target state of a SELE satisfies only one AMP, since there are just two AMP in the system. Moreover, either the source state or the target state of an ELE with respect to $\mathbf{Mode}_1$ and $\mathbf{Mode}_2$ doesn't satisfy $P^k_1 \wedge P^k_2$ according to Definition 16, so it satisfies only one AMP on $\mathbf{G}^k$. Since the source state and the target state of events in Σ^k_{SELE} and $\Sigma^k_{ELE,1,2}$ belong to the same class, it is true that $\Sigma^k_{SELE} = \Sigma^k_{ELE,1,2}$.

(iv) $\Sigma^k_{SINE} = \Sigma^k_{INE,1,2}$

Since a SPBS is equivalent to a PBS with respect to $\mathbf{Mode}_1$ and $\mathbf{Mode}_2$, then ac-

cording to Definition 9 and Definition 15, $(\forall i, j = 1, 2, i \neq j)(\forall k = 1, ..., h)\Sigma^k_{SINE} = \Sigma^k_{INE,1,2}$.

□

Lemma 13 builds the bridge between notions in Chapter 3 and Chapter 4 when the system only has two modes. The following proposition is a natural extension of Lemma 13.

Proposition 11. Denote by $\mathbf{G}$ a plant DES with two modes $\mathbf{Mode}_1$ and $\mathbf{Mode}_2$ distinguished by event alphabets Σ_1 and Σ_2. Then

(i) $\Sigma^G_{SETE} = \Sigma^G_{ETE,1,2}$;

(ii) $\Sigma^G_{SEYE} = \Sigma^G_{EYE,1,2}$;

(iii) $\Sigma^G_{SELE} = \Sigma^G_{ELE,1,2}$;

(iv) $\Sigma^G_{SINE} = \Sigma^G_{INE,1,2}$.

Proof. Suppose that there are h plant components. Take statement (i) as an example, since $\Sigma^1_{SETE} \cup ... \cup \Sigma^h_{SETE} = \Sigma^G_{SETE}$. Since $\Sigma^1_{ETE,1,2} \cup ... \cup \Sigma^h_{ETE,1,2} = \Sigma^G_{ETE,1,2}$, and $(\forall k = 1, ..., h)\ \Sigma^k_{SETE} = \Sigma^k_{ETE,1,2}$ according to Lemma 13, it's evident that $\Sigma^1_{SETE} \cup ... \cup \Sigma^h_{SETE} = \Sigma^1_{ETE,1,2} \cup ... \cup \Sigma^h_{ETE,1,2} = \Sigma^G_{SETE} = \Sigma^G_{ETE,1,2}$. The proofs for statement (ii), (iii), (iv) are similar to the proof for statement (i). □

Apart from Lemma 12 and Lemma 13, Lemma 14 establishes a connection between the BRS and the CMRS for a system with only two modes.

Lemma 14. Denote by $\mathbf{G}$ a plant DES with two modes $\mathbf{Mode}_1$ and $\mathbf{Mode}_2$ distinguished by event alphabets Σ_1 and Σ_2. Also denote by $\mathbf{R}_B$ the BRS for $\mathbf{G}$, and $\mathbf{R}_C$ the CMRS for $\mathbf{G}$. Then $(\forall s)(\forall i = 1, 2)\ \delta^{R_B}(q^{R_B}_o, s) \in \{q^0_i, ..., q^b_i\}$ iff $\delta^{R_C}(q^{R_C}_o, s) = q_i$.

Proof. Prove by induction on the length of string s.

(base case).

When $s = \epsilon$, the statement holds trivially. When $s = \sigma$, according to the definition of BRS and CMRS, σ can only be one of the MIE, i.e. $\sigma = \sigma_i \in \Sigma_{MIE}$ and $i = 1, 2$. Then $\delta^{R_B}(q^{R_B}_o, s) = q^0_i$ and $\delta^{R_C}(q^{R_C}_o, s) = q_i$.

(inductive case).

Suppose that there is a string s such that $\delta^{R_B}(q_o^{R_B}, s) \in \{q_i^0, ..., q_i^b\}$ and $\delta^{R_C}(q_o^{R_C}, s) = q_i$. Without loss of generality, assume $i = 1$. We then investigate whether $s\sigma$ also holds. If $\sigma \notin \Sigma_1$, it is not defined at either $\{q_1^0, ..., q_1^b\}$ of BRS or q_1 of CMRS, so $\sigma \in \Sigma_1$.

(i) When $\sigma \in \Sigma_{SETE}^G$, consider BRS first, $\delta^{R_B}(q_o^{R_B}, s) = q_1^r \in \{q_1^0, ..., q_1^{b-1}\}$. The event σ will lead the BRS to $q_1^{r+1} \in \{q_1^1, ..., q_1^b\}$. For CMRS, $\delta^{R_C}(q_o^{R_C}, s) = q_1$, and since σ is defined at q_1, $\delta^{R_C}(q_o^{R_C}, s\sigma)! = q_1$ according to the Definition 18.

(ii) When $\sigma \in \Sigma_{SEYE}^G$, consider BRS first, $\delta^{R_B}(q_o^{R_B}, s) = q_1^r \in \{q_1^1, ..., q_1^b\}$. The event σ will lead the BRS to $q_1^{r-1} \in \{q_1^0, ..., q_1^{b-1}\}$. For CMRS, $\delta^{R_C}(q_o^{R_C}, s) = q_1$, and since σ is defined at q_1, $\delta^{R_C}(q_o^{R_C}, s\sigma)! = q_1$.

(iii) When $\sigma \in \Sigma_{SELE}^G$, consider BRS first, $\delta^{R_B}(q_o^{R_B}, s) = q_1^r \in \{q_1^1, ..., q_1^b\}$. The event σ will lead the BRS to $q_1^r \in \{q_1^1, ..., q_1^b\}$. For CMRS, $\delta^{R_C}(q_o^{R_C}, s) = q_1$, and since σ is defined at q_1, $\delta^{R_C}(q_o^{R_C}, s\sigma)! = q_1$.

(iv) When $\sigma \in \Sigma_{SINE}^G$, consider BRS first, $\delta^{R_B}(q_o^{R_B}, s) = q_1^r \in \{q_1^0, ..., q_1^b\}$. The event σ will lead the BRS to $q_1^r \in \{q_1^0, ..., q_1^b\}$. For CMRS, $\delta^{R_C}(q_o^{R_C}, s) = q_1$, and since σ is defined at q_1, $\delta^{R_C}(q_o^{R_C}, s\sigma)! = q_1$.

(v) When $\sigma \in \Sigma_{RE}$, consider BRS first, $\delta^{R_B}(q_o^{R_B}, s) = q_1^0$. The event σ will lead the BRS to $q_2^0 \in \{q_2^0, ..., q_2^b\}$. For CMRS, $\delta^{R_C}(q_o^{R_C}, s) = q_1$, and since σ is defined at q_1, $\delta^{R_C}(q_o^{R_C}, s\sigma)! = q_2$.

Thus, $(\forall s)(\forall i = 1, 2)\ \delta^{R_B}(q_o^{R_B}, s) \in \{q_i^0, ..., q_i^b\}$ iff $\delta^{R_C}(q_o^{R_C}, s) = q_i$.

□

Theorem 5. Denote by $\mathbf{G}$ a plant DES with two modes $\mathbf{Mode}_1$ and $\mathbf{Mode}_2$ distinguished by event alphabets Σ_1 and Σ_2. Also denote by $\mathbf{R}_B$ the BRS for $\mathbf{G}$, and $\mathbf{R}_M = \mathbf{R}_C||\mathbf{R}_{1,2}$ the MRS for $\mathbf{G}$. Let $\mathbf{RG}_B = \mathbf{G}||\mathbf{R}_B$ and $\mathbf{RG}_M = \mathbf{G}||\mathbf{R}_M$. Then it is always true that $\mathbf{RG}_B = \mathbf{RG}_M$.

Proof. To prove $\mathbf{RG}_B = \mathbf{RG}_M$, it is equivalent to prove $L(\mathbf{RG}_B) = L(\mathbf{RG}_M)$ and $L_m(\mathbf{RG}_B) = L_m(\mathbf{RG}_M)$. We first prove $L(\mathbf{RG}_B) = L(\mathbf{RG}_M)$, i.e. $(\forall s)\ s \in L(\mathbf{RG}_B) \Leftrightarrow s \in L(\mathbf{RG}_M)$.

We have already presented that $L(\mathbf{RG}_B) = P_G^{-1}L(\mathbf{G}) \cap L(\mathbf{R}_B)$ and $L(\mathbf{RG}_M) = P_G^{-1}L(\mathbf{G}) \cap L(\mathbf{R}_M)$, where $P_G : \Sigma^R \to \Sigma$ and $\Sigma^R = \Sigma^{R_B} = \Sigma^{R_M}$.

(a) $(\forall s)\ s \in L(\mathbf{RG}_B) \Rightarrow s \in L(\mathbf{RG}_M)$

Prove by induction on the length of the string s.

(base case).

When $s = \epsilon$, it is true that $s \in P_G^{-1}L(\mathbf{G})$, $s \in L(\mathbf{R}_B)$ and $s \in L(\mathbf{R}_M)$. When $s \neq \epsilon$, consider $s = \sigma$, then according to the structure of BRS, $\sigma = \sigma_1, \sigma_2 \in \Sigma_{MIE}$. Since $\Sigma_{MIE} \subseteq \Sigma^R - \Sigma$, $\sigma \in P_G^{-1}L(\mathbf{G})$. Besides, $s = \sigma$ is in $L(\mathbf{R}_M)$ according to the structure of CMRS and EMRS, so $s = \sigma$ is in $L(\mathbf{RG}_M)$.

(inductive case).

Suppose that $s \in L(\mathbf{RG}_B)$ and $s \in L(\mathbf{RG}_M)$, then consider $s\sigma \in L(\mathbf{RG}_B)$.

(i) When $\sigma \in \Sigma_{SETE}^G$, without loss of generality, assume $\sigma \in \Sigma_1$ and $\sigma \notin \Sigma_2$ (otherwise σ will be a SINE). Then $\delta^{R_B}(q_o^{R_B}, s) = q_1^r$ and $r \in \{0, ..., k_1 - 1\}$. Then CMRS is at q_1 according to Lemma 14. Then in s. the number of occurrences of event in $\Sigma_{SETE}^G \cap \Sigma_1$ is greater than the number of occurrences of event in $\Sigma_{SEYE}^G \cap \Sigma_1$ by r. Besides, in s, the occurrences of event in $\Sigma_{SETE}^G \cap \Sigma_2$ is equal to the occurrences of event in $\Sigma_{SEYE}^G \cap \Sigma_2$. Thus in s, according to Proposition 11, the occurrences of event in $\Sigma_{ETE,1,2}^G$ is greater than the occurrences of event in $\Sigma_{EYE,1,2}^G$ by r, which means that $\mathbf{R}_{1,2}$ is at state $q_r^{R_{i,j}}$. According to Proposition 11, since $\Sigma_{SETE}^G = \Sigma_{ETE,1,2}^G$, event σ is defined at $q_r^{R_{i,j}}$ of $\mathbf{R}_{1,2}$. Since σ is also defined at q_1 of CMRS, it is defined in $\mathbf{R}_M$, namely $\delta^{R_M}(q_o^{R_M}, s\sigma)!$. Thus, $s\sigma \in L(\mathbf{R}_M)$. Since $s\sigma \in L(\mathbf{RG}_B)$, we can have that $s\sigma \in P_G^{-1}L(\mathbf{G})$, so $s\sigma$ is in $L(\mathbf{RG}_M)$.

(ii) When $\sigma \in \Sigma_{SEYE}^G$, the proof is very similar to the proof when $\sigma \in \Sigma_{SETE}^G$.

(iii) When $\sigma \in \Sigma_{SINE}^G$, the BRS and the CMRS are not at their initial states. According to proposition 11, $\Sigma_{SINE}^G = \Sigma_{INE,1,2}^G$. Since any event in $\Sigma_{INE,1,2}^G$ is defined at both q_1 and q_2 of CMRS and is not defined in $\mathbf{R}_{1,2}$, then the event σ is defined in $\mathbf{R}_M$, namely $\delta^{R_M}(q_o^{R_M}, s\sigma)!$. Thus, $s\sigma \in L(\mathbf{R}_M)$. Since $s\sigma \in L(\mathbf{RG}_B)$, we can have that $s\sigma \in P_G^{-1}L(\mathbf{G})$, so $s\sigma$ is in $L(\mathbf{RG}_M)$.

(iv) When $\sigma \in \Sigma_{SELE}^G$, without loss of generality, assume $\sigma \in \Sigma_1$ and $\sigma \notin \Sigma_2$ (otherwise σ will be a SINE). Then CMRS is at q_1 according to Lemma 14.

Since any external event will not be defined in $\mathbf{R}_{1,2}$, the event σ is defined in $\mathbf{R}_M$, namely $\delta^{R_M}(q_o^{R_M}, s\sigma)!$. Thus, $s\sigma \in L(\mathbf{R}_M)$. Since $s\sigma \in L(\mathbf{RG}_B)$, we can have that $s\sigma \in P_G^{-1}L(\mathbf{G})$, so $s\sigma$ is in $L(\mathbf{RG}_M)$.

(v) When $\sigma \in \Sigma_{RE}$, without loss of generality, assume $\sigma = \sigma_{1,2}$. Then CMRS is at q_1 according to Lemma 14. At the same time, in s, there is an equal number of occurrences of event in Σ^G_{SETE} and Σ^G_{SEYE}, which means that there is an equal number of occurrences of event in $\Sigma^G_{ETE,1,2}$ and $\Sigma^G_{EYE,1,2}$. Thus, $\mathbf{R}_{1,2}$ is at $q_o^{R_{1,2}}$, where $\sigma = \sigma_{1,2}$ is defined. Since $\sigma = \sigma_{1,2}$ is also defined at q_1 of CMRS, the event σ is defined in $\mathbf{R}_M$, namely $\delta^{R_M}(q_o^{R_M}, s\sigma)!$. Thus, $s\sigma \in L(\mathbf{R}_M)$. Since $s\sigma \in L(\mathbf{RG}_B)$, we can have that $s\sigma \in P_G^{-1}L(\mathbf{G})$, so $s\sigma$ is in $L(\mathbf{RG}_M)$.

(b) $(\forall s)\ [s \in L(\mathbf{RG}_M) \Rightarrow s \in L(\mathbf{RG}_B)]$

Prove by induction on the length of the string s.

(base case).

When $s = \epsilon$, it is true that $s \in P_G^{-1}L(\mathbf{G})$, $s \in L(\mathbf{R}_M)$ and $s \in L(\mathbf{R}_B)$. When $s \neq \epsilon$, consider $s = \sigma$, then according to the structure of MRS, $\sigma = \sigma_1, \sigma_2 \in \Sigma_{MIE}$. Since $\Sigma_{MIE} \subseteq \Sigma^R - \Sigma$, $\sigma \in P_G^{-1}L(\mathbf{G})$. Besides, $s = \sigma$ is in $L(\mathbf{R}_B)$ according to the structure of BRS, so $s = \sigma$ is in $L(\mathbf{RG}_B)$.

(inductive case).

Suppose that $s \in L(\mathbf{RG}_M)$ and $s \in L(\mathbf{RG}_B)$, then consider $s\sigma \in L(\mathbf{RG}_M)$.

(i) When $\sigma \in \Sigma^G_{ETE,1,2}$. According to Proposition 11, $\Sigma^G_{SETE} = \Sigma^G_{ETE,1,2}$. Without loss of generality, assume $\sigma \in \Sigma_1$, which means that CMRS is at q_1. Then the BRS is at a state in $\{q_1^0, ...q_1^b\}$. At the same time, $\mathbf{R}_{1,2}$ is at $q_r^{R_{1,2}} \in \{q_0^{R_{1,2}}, ..., q_{k_{1,2}-1}^{R_{1,2}}\}$. In s, the occurrences of event in $\Sigma^G_{ETE,1,2}$ is greater than the occurrences of event in $\Sigma^G_{EYE,1,2}$ by r, which means that the $\mathbf{R}_B$ is at q_1^r according to Lemma 14. Thus in s, event σ is defined at q_1^r of $\mathbf{R}_B$, namely $\delta^{R_B}(q_o^{R_B}, s\sigma)!$. Thus, $s\sigma \in L(\mathbf{R}_B)$. Since $s\sigma \in L(\mathbf{RG}_M)$, we can have that $s\sigma \in P_G^{-1}L(\mathbf{G})$, so $s\sigma$ is in $L(\mathbf{RG}_B)$.

(ii) When $\sigma \in \Sigma^G_{EYE,1,2}$, the proof is very similar to the proof when $\sigma \in \Sigma^G_{ETE,1,2}$.

(iii) When $\sigma \in \Sigma^G_{INE,1,2}$, the BRS and the CMRS are not at their initial states. According to Proposition 11, $\Sigma^G_{SINE} = \Sigma^G_{INE,1,2}$. Since any event in Σ^G_{SINE} is defined at any state except for the initial state of BRS, the event σ is defined in $\mathbf{R}_B$, namely $\delta^{R_B}(q_o^{R_B}, s\sigma)!$. Thus, $s\sigma \in L(\mathbf{R}_B)$. Since $s\sigma \in L(\mathbf{RG}_M)$, we can have that $s\sigma \in P_G^{-1}L(\mathbf{G})$, so $s\sigma$ is in $L(\mathbf{RG}_B)$.

(iv) When $\sigma \in \Sigma^G_{ELE,1,2}$. According to proposition 9, $\Sigma^G_{SELE} = \Sigma_{ELE,1,2}$. Without loss of generality, assume $\sigma \in \Sigma_1$, which means that CMRS is at q_1. Then the BRS is at a state in $\{q_1^0, ...q_1^b\}$. At the same time, since $s\sigma \in L(\mathbf{RG}_M)$, we can have that $s\sigma \in P_G^{-1}L(\mathbf{G})$. Thus, $\mathbf{R}_{1,2}$ is at $q_r^{R_{1,2}} \in \{q_1^{R_{1,2}}, ..., q_{k_{1,2}-1}^{R_{1,2}}\}$ since there is at least one plant component not at a public state to let an external event eligible to occur. In s, the occurrences of event in $\Sigma^G_{ETE,1,2}$ is greater than the occurrences of event in $\Sigma^G_{EYE,1,2}$ by r, which means that the $\mathbf{R}_B$ is at q_1^r according to Lemma 14. Thus in s, event σ is defined at q_1^r of $\mathbf{R}_B$, namely $\delta^{R_B}(q_o^{R_B}, s\sigma)!$. Thus, $s\sigma \in L(\mathbf{R}_B)$ and $s\sigma$ is in $L(\mathbf{RG}_B)$.

(v) When $\sigma \in \Sigma_{RE}$, without loss of generality, assume $\sigma = \sigma_{1,2}$. Then CMRS is at q_1 according to Lemma 14. At the same time, in s, there is an equal number of occurrences of event in $\Sigma^G_{ETE,1,2}$ and $\Sigma^G_{EYE,1,2}$, which means that there is an equal number of occurrences of event in Σ^G_{SETE} and Σ^G_{SEYE}. Thus, $\mathbf{R}_B$ is at q_1^0 according to Lemma 14, where $\sigma = \sigma_{1,2}$ is defined, namely $\delta^{R_B}(q_o^{R_B}, s\sigma)!$. Thus, $s\sigma \in L(\mathbf{R}_B)$. Since $s\sigma \in L(\mathbf{RG}_M)$, we can have that $s\sigma \in P_G^{-1}L(\mathbf{G})$, so $s\sigma$ is in $L(\mathbf{RG}_B)$.

(c) $(\forall s)\ [s \in L_m(\mathbf{RG}_B) \Leftrightarrow s \in L_m(\mathbf{RG}_M)]$

In $\mathbf{R}_B$, the only state that is not marked is the initial state, where any event defined is a MIE. Similarly, in $\mathbf{R}_M$, the only state that is not marked is the initial state according to the structure of CMRS and two EMRS. Since $L(\mathbf{RG}_B) = L(\mathbf{RG}_M)$, $\mathbf{RG}_B = \mathbf{G}||\mathbf{R}_B$, $\mathbf{RG}_M = \mathbf{G}||\mathbf{R}_M$, $L(\mathbf{RG}_B) = P_G^{-1}L(\mathbf{G}) \cap L(\mathbf{R}_B)$ and $L(\mathbf{RG}_M) = P_G^{-1}L(\mathbf{G}) \cap L(\mathbf{R}_M)$, we can conclude that $L_m(\mathbf{RG}_B) = L_m(\mathbf{RG}_M)$.

Therefore, $L(\mathbf{RG}_B) = L(\mathbf{RG}_M)$ and $L_m(\mathbf{RG}_B) = L_m(\mathbf{RG}_M)$, namely $\mathbf{RG}_B = \mathbf{RG}_M$.

□

Theorem 6. Denote by $\mathbf{G}$ the plant DES with n modes $\mathbf{Mode}_1, ..., \mathbf{Mode}_n$ distinguished

by event alphabets $\Sigma_1, ..., \Sigma_n$. Also denote by $\mathbf{R}$ the MRS for $\mathbf{G}$. Let $\mathbf{RG} = \mathbf{G}||\mathbf{R}$ be the reconfiguration plant that satisfies all five requirements described in Problem 1. Given that $\mathbf{B}$ is the behavioral specification, the supervisory controller $\mathbf{RSUP}$ computed by $\mathbf{RSUP} = \mathbf{Supcon}(\mathbf{RG}, \mathbf{B})$ also satisfies all the five requirements.

Proof. The requirements (i), (ii) and (iii) are met trivially, since $\mathbf{B}$ doesn't affect REs and MIEs. The requirement (iv) is also met since the "**Supcon**" function guarantees the nonblockingness of the resulting supervisory controller [1]. The requirement (v) is met according to a similar proof for (v) in Theorem 1. □

Theorem 7. [Time Complexity of GR-Checking-1]. When all the state sets are implemented in linked lists and the transitions with the corresponding state are structured in a red-black tree, the worst-case time complexity of GR-Checking-1 is $O(VE+V^2log(V))$, where V is the number of states, and E is the number of transitions in the DES to be checked.

Proof. In each iteration of the loop from line 6 to line 15, since each transition will be visited at most once, then $O(E)$ operations are needed given that each step from line 8 to line 13 needs only $O(1)$ operations with the linked list data structure. Since the states and the corresponding transitions are structured in a red-black tree, to locate the right transitions for each state needs $O(log(V))$ operations. There are at most $V-1$ states need to be considered in an iteration of the loop from line 6 to line 15, hence $O(E+Vlog(V))$ operations are needed inside the while loop. Since in each iteration of the while loop, there is at least one state being added to the Q_{GR}, otherwise the algorithm would terminate, then there are at most $V-1$ iterations of the while loop. The final checking part from line 17 to line 21 needs $O(V)$ operations. But it is obvious that the while loop dominates. Therefore, the worst-case time complexity of the GR-Checking-1 is $O(VE+V^2log(V))$. □

B..3 Proofs in Appendix A

In order to formally prove the correctness of the proposed LMRS, some observations are needed first.

Lemma 15. Under the assumptions in Definition 23, $(\forall i,j \in 1,...,n)(\forall k \in 1,...,h)(\forall s \in L(\mathbf{RG}))\ |s|^k_{ETE,i,j} = |s|^k_{EYE,i,j}$ iff $\delta^{R^k_{i,j}}(q_o^{R^k_{i,j}}, P_{R^k_{i,j}}(s)) = q_o^{R^k_{i,j}}$ and $|s|^k_{ETE,i,j} = |s|^k_{EYE,i,j} + 1$ iff $\delta^{R^k_{i,j}}(q_o^{R^k_{i,j}}, P_{R^k_{i,j}}(s)) = q_1^{R^k_{i,j}}$, where $P_{R^k_{i,j}} : \Sigma^{RG*} \to \Sigma^{R^k_{i,j}*}$.

Proof. Since all the exit(entry) events with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ in $\mathbf{G}^k$ are included in the LEMRS $\mathbf{R}^k_{i,j}$, namely $\Sigma^k_{ETE,i,j} \dot\cup \Sigma^k_{EYE,i,j} \subseteq \Sigma^{R^k_{i,j}}$, if $|s|^k_{ETE,i,j} = |s|^k_{EYE,i,j}$, then according to the structure of $\mathbf{R}^k_{i,j}$, $\delta^{R^k_{i,j}}(q_o^{R^k_{i,j}}, P_{R^k_{i,j}}(s)) = q_o^{R^k_{i,j}}$. Similarly, according to the structure of $\mathbf{R}^k_{i,j}$, $|s|^k_{ETE,i,j} = |s|^k_{EYE,i,j} + 1$ iff $\delta^{R^k_{i,j}}(q_o^{R^k_{i,j}}, P_{R^k_{i,j}}(s)) = q_1^{R^k_{i,j}}$. □

Lemma 16. Under the assumptions in definition 17, $(\forall i,j \in 1,...,n)(\forall k = 1,...,h)(\forall s \in L(\mathbf{RG}))\ |s|^k_{ETE,i,j} = |s|^k_{EYE,i,j}$ iff $\delta^{R^k_{i,j}}(q_o^{R^k_{i,j}}, P_{R^k_{i,j}}(s)) = q_o^{R^k_{i,j}}$ and $|s|^k_{ETE,i,j} = |s|^k_{EYE,i,j} + 1$ iff $\delta^{R^k_{i,j}}(q_o^{R^k_{i,j}}, P_{R^k_{i,j}}(s)) = q_1^{R^k_{i,j}}$, where $P_{R^k_{i,j}} : \Sigma^{RG*} \to \Sigma^{R^k_{i,j}*}$.

Proof. According to the structure of LEMRS, this lemma is true. □

Proposition 12. $(\forall i,j \in 1,...,n)(\forall k = 1,...,h)(\forall s \in L(\mathbf{RG}))\ \delta^k(q^k_o, P_{G^k}(s)) \in Q^k_{PBS,i,j}$ iff $\delta^{R^k_{i,j}}(q_o^{R^k_{i,j}}, P_{R^k_{i,j}}(s)) = q_o^{R^k_{i,j}}$.

Proof. This statement is a logical combination of Lemma 15 and Lemma 16. Then it is automatically true according to Proposition 15 and Lemma 16. □

Lemma 17. Under the assumptions in Definition 23, $L(\mathbf{G}) \subseteq P_G L(\mathbf{R})$, where $P_G : \Sigma^{R*} \to \Sigma^{G*}$.

Proof. Prove by induction on the length of the string s.
(base case).

Consider a string $s \in L(\mathbf{RG})$. When $s = \epsilon$, the lemma holds trivially. When $s = \sigma \in \Sigma$, and $\delta(q_o, \sigma)!$, since q_o is a PBS with respect to arbitrary two modes, then the event σ can only be an INE or an ETE with respect to arbitrary two modes, and it can never be an ELE or EYE. The event σ must be in one of the event alphabets for the modes involved in $\mathbf{G}$. Consider $\sigma \in \Sigma_i$, if σ is an INE with respect to two modes, then it is either defined at the initial state of some EMRS as an ETE, or is completely not defined in some EMRS. And since $\sigma_i \in \Sigma_{MIE} \subseteq (\Sigma^{R_c} - \Sigma_{i,j})$, $(\forall i,j \in 1,...,n)\ \sigma_i\sigma \in L(\mathbf{R} = \mathbf{R}_c||\mathbf{R}^1_{1,2}||...||\mathbf{R}^1_{n-1,n}||...||\mathbf{R}^h_{1,2}||...||\mathbf{R}^h_{n-1,n})$. Since $\sigma_i\sigma \in L(\mathbf{R}_c)$, and $\sigma_i \in \Sigma_{MIE} \subseteq (\Sigma^R - \Sigma)$, then $\sigma \in P_G L(\mathbf{R})$. The situation when σ is an ETE is the same.

(inductive case).

Given a string $s \in L(\mathbf{G}) \cap P_G L(\mathbf{R}), (\exists s^{'} \in L(\mathbf{R}))\ P_G(s^{'}) = s \Rightarrow \delta^R(q_o^R, s^{'})!$. Consider an event $\sigma \in \Sigma$. Suppose that the event σ serves as different class of events with respect to different pair of modes.

(i) If $\sigma \in \Sigma^G_{ETE,i,j}$ and $\delta(q_o, s\sigma)!$, then consider every plant component $\mathbf{G}^k$ such that $\sigma \in \Sigma^k_{ETE,i,j}$, we have $\delta^k(q_o^k, P_k(s)) \in Q^k_{PBS,i,j}$. Then according to Lemma 8, $|s^{'}|^k_{ETE,i,j} = |s^{'}|^k_{EYE,i,j}$, hence $\delta^{R^k_{i,j}}(q_o^{R^k_{i,j}}, P_{R^k_{i,j}}(s')) = q_o^{R^k_{i,j}}$ according to Lemma 15. Therefore, for each $\mathbf{G}^k$ such that $\sigma \in \Sigma^k_{ETE,i,j}$, $\delta^{R^k_{i,j}}(q_o^{R^k_{i,j}}, P_{R^k_{i,j}}(s')\sigma)! = \delta^{R^k_{i,j}}(q_o^{R^k_{i,j}}, P_{R^k_{i,j}}(s'\sigma))!$. According to assumption (iii) in Definition 23, σ is not defined in any other plant component or the corresponding LEMRS.

(ii) If $\sigma \in \Sigma^G_{INE,i,j}$ and $\delta(q_o, s\sigma)!$, then $\delta(q_o^R, s)! \in Q^G_{PBS,i,j}$. According to Proposition 8, $\delta(q_o, P_G(s^{'})) = \delta(q_o, s) \in Q^G_{PBS,i,j}$ iff $|s^{'}|^G_{ETE,i,j} = |s^{'}|^G_{EYE,i,j}$, and since $\Sigma^G_{INE,i,j}$ is not defined in $\mathbf{R}^k_{i,j}$ for every plant component $\mathbf{G}^k$, then σ is defined after synchronization, if it is defined in other components of the synchronous product.

(iii) If $\sigma \in \Sigma^G_{EYE,i,j}$ and $\delta(q_o, s\sigma)!$, then consider every plant component $\mathbf{G}^k$ such that $\sigma \in \Sigma^k_{EYE,i,j}$, we have $\delta^k(q_o^k, P_k(s)) \notin Q^k_{PBS,i,j}$. Then according to Lemma 8, $|s^{'}|^k_{ETE,i,j} = |s^{'}|^k_{EYE,i,j} + 1$, hence $\delta^{R^k_{i,j}}(q_o^{R^k_{i,j}}, P_{R^k_{i,j}}(s')) = q_1^{R^k_{i,j}}$ according to Lemma 15. Therefore, for each $\mathbf{G}^k$ such that $\sigma \in \Sigma^k_{EYE,i,j}$, $\delta^{R^k_{i,j}}(q_o^{R^k_{i,j}}, P_{R^k_{i,j}}(s')\sigma)! = \delta^{R^k_{i,j}}(q_o^{R^k_{i,j}}, P_{R^k_{i,j}}(s'\sigma))!$. According to assumption (iii) in Definition 23, σ is not defined in any other plant component or the corresponding LEMRS.

(iv) If $\sigma \in \Sigma^G_{ELE,i,j}$ and $\delta(q_o, s\sigma)!$, then $\delta(q_o^R, s)! \notin Q^G_{PBS,i,j}$. According to Proposition 8, $\delta(q_o, P_G(s^{'})) = \delta(q_o, s) \in Q^G_{PBS,i,j}$ iff $|s^{'}|^G_{ETE,i,j} = |s^{'}|^G_{EYE,i,j}$, and since $\Sigma^G_{ELE,i,j}$ is not defined in $\mathbf{R}^k_{i,j}$ for every plant component $\mathbf{G}^k$, then σ is defined after synchronization, if it is defined in other components of the synchronous product.

For the CMRS, consider $s = s_1\sigma_1$ and assume that $\delta(q_0, s) \models P_{t_1} \wedge ... \wedge P_{t_s}$. It is true that $(\exists s_1^{'})\ P_G(s_1^{'}) = s_1$ and $\delta^{R_c}(q_o^{R_c}, s_1^{'}\sigma_1)!$. According to Lemma 10, $\delta^{R_c}(q_o^{R_c}, s_1^{'}\sigma_1) \in \{q_{t_1}, ..., q_{t_s}\}$. Then if $\sigma \in \Sigma_{t_1} \cup ... \cup \Sigma_{t_s}$, suppose that $\delta^{R_c}(q_o^{R_c}, s_1^{'}\sigma_1) = q_{t_i}$ and $\sigma \in \Sigma_{t_j}$, for each LEMRS $\mathbf{R}^k_{t_i,t_j}$, $\delta^{R^k_{t_i,t_j}}(q_o^{R^k_{t_i,t_j}}, P_{R^k_{t_i,t_j}}(s_1^{'}\sigma_1)) = q_o^{R^k_{t_i,t_j}}$, at which σ_{t_i,t_j} is enabled. Thus, $s_1\sigma_1\sigma_{t_i,t_j}\sigma$ is enabled in $\mathbf{R}_c$. If $\sigma \notin \Sigma_{t_1} \cup ... \cup \Sigma_{t_s}$, then since $\delta^{R_c}(q_o^{R_c}, s_1^{'}\sigma_1) \in \{q_{t_1}, ..., q_{t_s}\}$,

it violate the assumption that each mode is reachable and coreachable by itself. Thus, $s\sigma \in P_G L(\mathbf{R}_c)$.

Since $\Sigma = \Sigma^G_{ETE,i,j} \cup \Sigma^G_{INE,i,j} \cup \Sigma^G_{EYE,i,j} \cup \Sigma^G_{ELE,i,j}$, even if the event σ may serve as different types of events with respect to different pairs of modes, it is still eligible to occur in $\mathbf{R}$ if $\delta(q_o, s\sigma)!$, so $s^{'}\sigma \in L(\mathbf{R})$ and $P_G(s^{'}\sigma) \in P_G L(\mathbf{R})$. It can be concluded that $L(\mathbf{G}) \subseteq P_G L(\mathbf{R})$.

□

Remark. It can be deduced that $L(\mathbf{G}) \subseteq P_{RG\to G} L(\mathbf{R})$, where $P_{RG\to G} : \Sigma^{RG*} \to \Sigma^{G*}$ since $\Sigma^{RG} = \Sigma^R$.

Proposition 13. Under the assumptions in Definition 23, $L_m(\mathbf{G}) \subseteq P_G L_m(\mathbf{R})$.

Proof. In a CMRS, every state except for the initial state is a marked state. In an LEMRS, every state is a marked state. Since the only events defined at the initial state of a CMRS are the events in $\Sigma_{MIE} \subseteq (\Sigma^R - \Sigma)$, and all events in Σ are defined at marked states of CMRS, then we can get that $P_G L_m(\mathbf{R}) = P_G L(\mathbf{R})$.

According to Lemma 16, $L(\mathbf{G}) \subseteq P_G L(\mathbf{R})$, and since $L_m(\mathbf{G}) \subseteq L(\mathbf{G})$, it is true that $L_m(\mathbf{G}) \subseteq L(\mathbf{G}) \subseteq P_G L(\mathbf{R}) = P_G L_m(\mathbf{R})$, namely $L_m(\mathbf{G}) \subseteq P_G L_m(\mathbf{R})$. □

The following theorem formally demonstrates that the reconfiguration plant generated by the synchronization of a plant and its LMRS preserves the dynamics of the plant.

Theorem 8. Denote by $\mathbf{G}$ the plant DES, and $\mathbf{R}$ the LMRS constructed according to Definition 22. The resulting reconfiguration plant $\mathbf{RG} = \mathbf{G}||\mathbf{R}$ guarantees that $L(\mathbf{G}) = P_G(L(\mathbf{G})||L(\mathbf{R}))$ and $L_m(\mathbf{G}) = P_G(L_m(\mathbf{G})||L_m(\mathbf{R}))$, where $P_i : \Sigma^{RG*} \to \Sigma^{i*}, i = \mathbf{G}, \mathbf{R}$.

Proof. The proof is exactly the same as the proof for Theorem 3 in Chapter 4. □

Theorem 8 shows that the reconfiguration plant will not discardd any information of the plant, namely the localized multiple reconfiguration approach preserves the dynamics of the plant. If that were not the case, the approach would be invalid automatically. Now we can go one step further to prove the correctness of the approach.

Formally, the following theorem is presented to show that the LMRS constructed according to Definition 22 can solve Problem 4 under assumptions in Definition 23 through the synchronization with the plant DES $\mathbf{G}$.

Theorem 9. [Correctness of LMRS]. Denote by $\mathbf{G} = (Q, \Sigma, \delta, q_o, Q_m) = \mathbf{G}^1||...||\mathbf{G}^h$ the plant DES formed by synchronization, where $\mathbf{G}^k$ is the k^{th} component. $\mathbf{Mode}_1, ..., \mathbf{Mode}_n$ are n different modes of $\mathbf{G}$ distinguished by event alphabets $\Sigma_1, ..., \Sigma_n$. Also denote by $\mathbf{R}$ the LMRS constructed according to Definition 22. The resulting reconfiguration plant $\mathbf{RG} = \mathbf{G}||\mathbf{R}$. Given the assumptions in Definition 23, then $\mathbf{RG}$ satisfies all five requirements described in Problem 4.

Proof. It is obvious that $\Sigma^R = \Sigma \cup \Sigma_{MIE} \cup \Sigma_{RE} = \Sigma^1 \cup ... \cup \Sigma^h \cup \Sigma_{MIE} \cup \Sigma_{RE} = \Sigma_1 \cup ... \cup \Sigma_n \cup \Sigma_{MIE} \cup \Sigma_{RE}$ The notion of natural projection is also needed. A natural projection is defined as $P_k : \Sigma^{RG*} \to \Sigma_k^*, (k = \mathbf{G}, \mathbf{R}, \mathbf{G}^1, ..., \mathbf{G}^h, \mathbf{R}^1_{1,2}, ..., \mathbf{R}^h_{n,n-1})$, where $\Sigma^{RG} = \Sigma \cup \Sigma^R = \Sigma^R$. Thus, $P_G : \Sigma^{RG*} \to \Sigma^*$ and $P_R : \Sigma^{RG*} \to \Sigma^{R*}$, namely $P_R : \Sigma^{R*} \to \Sigma^{R*}$. At the same time, $p_G^{-1} : Pwr(\Sigma^*) \to Pwr(\Sigma^{R*})$ and $p_R^{-1} : Pwr(\Sigma^{R*}) \to Pwr(\Sigma^{R*})$ are two inverse projections. Since $\mathbf{RG} = \mathbf{G}||\mathbf{R}, L(\mathbf{RG}) = L(\mathbf{G})||L(\mathbf{R}) = P_G^{-1}(L(\mathbf{G})) \cap P_R^{-1}(L(\mathbf{R})) = P_G^{-1}(L(\mathbf{G})) \cap L(\mathbf{R})$.

Besides, according to Chapter 3 of [1], it is true that $P_G(L(\mathbf{RG})) = P_G(L(\mathbf{G})||L(\mathbf{R})) \subseteq L(\mathbf{G})$ and $P_R(L(\mathbf{RG})) = P_R(L(\mathbf{G})||L(\mathbf{R})) \subseteq L(\mathbf{R})$. Similarly, $P_{G^i}(L(\mathbf{RG})) \subseteq L(\mathbf{G}^i)$ for each plant component $\mathbf{G}^i$. Therefore, if a string $s \in L(\mathbf{RG})$, then $P_{G_i}(s) \in L(\mathbf{G}^i)$ and $P_R(s) \in L(\mathbf{R})$. The requirements are proved in the order: (i), (ii), (iii), (iv), (v).

(i) $(\forall \sigma_i \in \Sigma_{MIE})(\exists q \in Q^{RG})\ \delta^{RG}(q_o^{RG}, \sigma_i) = q$.

Consider an arbitrary $\sigma_i \in \Sigma_{MIE}$. Since $\Sigma_{MIE} \subseteq \Sigma^R - \Sigma$, if σ_i can occur in $\mathbf{R}$, it can also occur in $\mathbf{RG}$ (it cannot be blocked by transitions in $\mathbf{G}$). In $\mathbf{R}$, σ_i is only defined in $\mathbf{R}_c$ but not in any LEMRS. In $\mathbf{R}_c$, $\delta^{R_c}(q_o^{R_c}, \sigma_i) = q_i$. At the same time, in $\mathbf{R}_c$, all events in $\Sigma^R - \Sigma_{MIE}$ are disabled to occur at the initial state, and they will also be disabled at the initial state in $\mathbf{RG}$ after synchronization. Therefore, after synchronization, $\delta^{RG}(q_o^{RG}, \sigma_i)!$, i.e. $(\exists q \in Q^{RG})\ \delta^{RG}(q_o^{RG}, \sigma_i) = q$.

(ii) (a) $(\forall i, j \in 1, ..., n, i \neq j)(\forall s \in L(\mathbf{RG}))\ [\delta^{RG}(q_o^{RG}, s\sigma_{i,j})! \Rightarrow (\forall k = 1, ..., h)\delta^k(q_o^k, P_{G^k}(s)) \in Q^k_{PBS,i,j}]$.
$\delta^{RG}(q_o^{RG}, s\sigma_{i,j})!$ means that $\delta^{R_c}(q_o^{R_c}, s) = q_i$ and $(\forall k \in 1, ..., h)\ \delta^{R^k_{i,j}}(q_o^{R^k_{i,j}}, P_{R^k_{i,j}}(s)) = q_o^{R^k_{i,j}}$. If q_i is directly reached by $q_o^{R_c}$ in $\mathbf{R}_c$, i.e. $\delta^{R_c}(q_o^{R_c}, \sigma_i) = q_i$, then $(\forall k = 1, ..., h)\ \delta^k(q_o^k, P_{G^k}(s)) = q_o^k \in Q^k_{PBS,i,j}$ since $s = \sigma_i$.
If q_i is reached by s and $|s| \neq 1$, then for each plant component $\mathbf{G}^k$, there

must be an equal number of occurrences of ETE and EYE with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ in s since $(\forall \sigma_1 \in \Sigma^k_{ETE,i,j})\ \delta^{R^k_{i,j}}(q_o^{R^k_{i,j}}, \sigma_1) = q_1^{R^k_{i,j}}$ and $(\forall \sigma_2 \in \Sigma^k_{EYE,i,j})\ \delta^{R^k_{i,j}}(q_1^{R^k_{i,j}}, \sigma_2) = q_o^{R^k_{i,j}}$ and $\delta^{R^k_{i,j}}(q_o^{R^k_{i,j}}, P_{R^k_{i,j}}(s)) = q_0^{R^k_{i,j}}$. Since there is an equal number of occurrences of ETE and EYE with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$, and EYE is eligible only after an ETE has occurred in each component. For each ETE in s and also in Σ^k for an arbitrary $\mathbf{G}^r$, there must be an EYE in Σ^k also in s, namely $|s|^k_{EYE,i,j} = |s|^k_{ETE,i,j}$. According to Lemma 8, for all those component $\mathbf{G}^k$, $\delta^k(q^k_o, P_{G^k}(s)) \in Q^k_{PBS,i,j}$. For each of other components $\mathbf{G}^e$, it is also true that $\delta^e(q^e_o, P_{G^e}(s)) \in Q^e_{PBS,i,j}$ since no ETE in $\mathbf{G}^e$ has occurred yet. Thus, $(\forall i,j \in 1,...,n, i \neq j)(\forall k = 1,...,h)\ \delta^k(q^k_o, P_{G^k}(s)) \in Q^k_{PBS,i,j}$ holds.

(b) $(\forall \sigma_{i,j} \in \Sigma_{RE})(\exists q_s, q_d \in Q^{RG}, q_s \neq q_d)\ \delta^{RG}(q_s, \sigma_{i,j}) = q_d$.

Consider an arbitrary $\sigma_{i,j} \in \Sigma_{RE}$. Since $\Sigma_{RE} \subseteq \Sigma^R - \Sigma$, if $\sigma_{i,j}$ can occur in $\mathbf{R}$, it can also occur in $\mathbf{RG}$ (it cannot be blocked by transitions in $\mathbf{G}$). In $\mathbf{R}_c$, $\delta^{R_c}(q_i, \sigma_{i,j}) = q_j \neq q_i$. In $\mathbf{R}^k_{i,j}$ for each plant component $\mathbf{G}^k$, as long as there is an equal number of occurrences of ETE and EYE with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$, $\delta^{R^k_{i,j}}(q_o^{R^k_{i,j}}, \sigma_{i,j}) = q_o^{R^k_{i,j}}$. Therefore, after synchronization, $(\forall \sigma_{i,j} \in \Sigma_{RE})(\exists q_s, q_d \in Q^{RG})\ \delta^{RG}(q_s, \sigma_{i,j})! = q_d$, and q_s, q_d can be distinguished by q_i and q_j in $\mathbf{R}_c$.

(c) $(\forall \sigma_{i,j} \in \Sigma_{RE})(\forall q, q' \in Q^{RG}, q \neq q')\ [\delta^{RG}(q, \sigma_{i,j}) = q' \Rightarrow (q \models P^{RG}_i \wedge q' \models P^{RG}_j)]$.

Consider $\delta^{RG}(q^{RG}_o, s) = q$ and $\delta^{RG}(q^{RG}_o, s\sigma_{i,j}) = q'$. Suppose that $s = \sigma_k s_1 \sigma_{k,i} s_2$ and $s_1 \in \Sigma^*_k, s_2 \in \Sigma^*_i$. Since $\delta^{RG}(q^{RG}_o, \sigma_k s_1 \sigma_{k,i})!, \delta^{R_c}(q_o^{R_c}, \sigma_k s_1) = q_k$ and $(\forall r = 1,...,h)\ \delta^r(q^r_o, P_{G^r}(\sigma_k s_1)) \in Q^r_{PBS,k,i}$ according to (ii)(a). Thus, $(\forall r = 1,...,h)\ \delta^r(q^r_o, P_{G^r}(\sigma_k s_1)) \models P^r_i$.

According to assumption (ii) in Definition 23, each mode in each plant component is reachable by itself, so $\exists s_3 \in \Sigma^*_i$ such that $\delta(q_o, s_1) = \delta(q_o, s_3)$. It is natural that $\sigma_i s_3 s_2 \in P^{-1}_G(L(\mathbf{G}))$. Besides, since $s_3 s_2 \in \Sigma^*_i \cap L(\mathbf{G})$ and $L(\mathbf{G}) \subseteq P_G(L(\mathbf{R}))$ according to Lemma 11, then $\delta^R(q^R_o, \sigma_i s_3 s_2)!$ i.e. $\sigma_i s_3 s_2 \in L(\mathbf{R})$. Since $L(\mathbf{RG}) = P^{-1}_G(L(\mathbf{G})) \cap L(\mathbf{R}), \sigma_i s_3 s_2 \in L(\mathbf{RG})$. Since $\delta(q_o, s_1) = \delta(q_o, s_3)$, then $(\forall r = 1,...,h)\ \delta^r(q^r_o, P_{G^r}(\sigma_i s_3)) \in Q^r_{PBS,k,i}$. Since $s_3 \in \Sigma^*_i$, $\delta^{R_c}(q_o^{R_c}, \sigma_i s_3) = q_i$ according to the definition of CMRS. Thus, $\delta^R(q^R_o, \sigma_i s_3) = \delta^R(q^R_o, \sigma_k s_1 \sigma_{k,i})$ and $\delta^R(q^R_o, \sigma_i s_3 s_2) = \delta^R(q^R_o, \sigma_k s_1 \sigma_{k,i} s_2)$. Since $\delta^R(q^R_o, \sigma_i s_3 s_2) = \delta^R(q^R_o, s)$

and $\delta^R(q_o^R, P_G(\sigma_i s_3 s_2)) = \delta^R(q_o^R, P_G(s)), \delta^{RG}(q_o^{RG}, s) = q = \delta^{RG}(q_o^{RG}, \sigma_i s_3 s_2)$. Since σ_i is the MIE for $\mathbf{Mode}_i$ and $s_3 s_2 \in \Sigma_i^*, q \models P_i^{RG}$. For a more complicated string such as $s = \sigma_p s_1 \sigma_{p,k} s_2 ... s_3 \sigma_{r,i} s_4$ where $s_4 \in \Sigma_i^*$, by a similar replacement procedure, it is still true that $q \models P_i^{RG}$. Similarly, $q^{'} \models P_j^{RG}$.

(iii) $(\forall \sigma_{i,j}, \sigma_{j,i} \in \Sigma_{RE})(\forall q \in Q^{RG})(\exists s \in L(\mathbf{RG}))\ [\delta^{RG}(q, \sigma_{i,j})! \Rightarrow \delta^{RG}(q, \sigma_{i,j} s \sigma_{j,i})!]$.

Consider the string $s^{'} \in L(\mathbf{RG})$ such that $\delta^{RG}(q_o^{RG}, s^{'}) = q$, where the event $\sigma_{i,j}$ is eligible to occur, then $\delta^{R_c}(q_o^{R_c}, s^{'}) = q_i$ and $(\forall k = 1, ..., h)\ \delta^{R_{i,j}^k}(q_o^{R_{i,j}^k}, s^{'}) = q_o^{R_{i,j}^k}$. According to Proposition 12, $\delta^k(q_o^k, P_{G^k}(s^{'})) \in Q_{PBS,i,j}^k$ since $\delta^{R_{i,j}^k}(q_o^{R_{i,j}^k}, P_{R_{i,j}^k}(s')) = q_o^{R_{i,j}^k}$. After the event $\sigma_{i,j}$ occurs, $\delta^{R_c}(q_o^{R_c}, s^{'}\sigma_{i,j}) = q_j$, where the event $\sigma_{j,i}$ is immediately eligible to occur. Besides, since $(\forall k = 1, ..., h)\ \delta^{R_{i,j}^k}(q_o^{R_{i,j}^k}, s^{'}\sigma_{i,j}) = q_o^{R_{i,j}^k}$, the RE $\sigma_{j,i}$ is also eligible to occur in $\mathbf{R}_{i,j}$ according to the structure of LEMRS. Since $\sigma_{j,i}$ is not defined in any other LEMRS, $\sigma_{j,i}$ is eligible to occur. At that time, $s = \epsilon$.

Consider $s \neq \epsilon$. If at q_j of $\mathbf{R}_c$, there are some INE with respect to $\mathbf{Mode}_j$ and $\mathbf{Mode}_i$ defined, then after a string $s \in (\Sigma_{INE,j,i}^k \cap \Sigma_j)^*, \sigma_{j,i}$ is still eligible to occur since $(\forall \sigma \in \Sigma_{INE,j,i}^k)(\forall k = 1, ..., h)\ \delta^{R_{i,j}^k}(q_o^{R_{i,j}^k}, \sigma)\not!$. On the other hand, if the string $s \in \Sigma_j^*$ also contains ETE and EYE with respect to $\mathbf{Mode}_j$ and $\mathbf{Mode}_i$, if the numbers of ETE and EYE are equal within the scope of each plant component, then according to the structure of LEMRS, $(\forall k = 1, ..., h)\ \delta^{R_{i,j}^k}(q_o^{R_{i,j}^k}, P_{R_{i,j}^k}(s)) = q_o^{R_{i,j}^k}$, hence $\delta^{RG}(q, \sigma_{i,j} s \sigma_{j,i})!$.

(iv) $(\forall i \in 1, ..., n)(\forall q \models P_i^{RG})(\exists \sigma_i \in \Sigma_{MIE})(\exists s \in \Sigma_i^*)\ [\delta^{RG}(q_o^{RG}, \sigma_i s) = q \Rightarrow (\exists q_m \in Q_m^{RG})(\exists s^{'} \in \Sigma_i^*)(\delta^{RG}(q, s^{'}) = q_m \wedge q_m \models P_i^{RG})]$.

Since $\delta^{RG}(q_o^{RG}, \sigma_i s)!$ and $s \in \Sigma_i^*$, $\delta(q_o, s)!$. Given the assumption (ii) that each mode is coreachable in each plant component by itself, then $(\forall k = 1, ..., h)\ \delta^k(q_o, P_{G^k}(s))$ can lead to a marked state, i.e. $(\exists s^{k'} \in \Sigma_i^* \cap \Sigma^{k*})\ \delta^k(q_o^k, P_{G^k}(s) s^{k'}) \in Q_m^k$. Then it is natural that $(\exists s^{'} \in \Sigma_i^*)\ \delta(q_o, s s^{'})! \in Q_m$, where $(\forall k = 1, ..., h)\ P_{G^k}(s^{'}) = s^{k'}$. Since $s s^{'} \in L(\mathbf{G}) \subseteq P_G(L(\mathbf{R}))$ according to Lemma 11 and $s s^{'} \in \Sigma_i^*$, then $\delta^{R_c}(q_o^{R_c}, s s^{'})!$, hence $(\exists \sigma_i \in \Sigma_{MIE})\ \delta^{R_c}(q_o^{R_c}, \sigma_i s s^{'})!$. Furthermore, since $(\forall q \in Q^R, q \neq q_o^R)\ q \in Q_m^R$, and $\delta(q_o, s s^{'}) \in Q_m, \delta^{RG}(q_o^{RG}, \sigma_i s s^{'}) \in Q_m^{RG}$, namely $(\exists q_m \in Q_m^{RG})(\exists s^{'} \in \Sigma_i^*)\ \delta^{RG}(q, s^{'}) = q_m$. Obviously, since $\delta^{RG}(q, s^{'}) = \delta^{RG}(q_o^{RG}, s s^{'}) = q_m$ and $s s^{'} \in \Sigma_i^*$,

according to the definition of SMP, $q_m \models P_i^{RG}$.

(v) $(\forall q \in Q^{RG}, q \neq q_o^{RG})(\exists i \in 1,...,n)(\forall j \in 1,...,n, i \neq j)\ [q \models P_i^{RG} \wedge q \not\models P_j^{RG}]$.

(a) $(\forall q \in Q^{RG}, q \neq q_o^{RG})(\exists i \in 1,...,n)\ q \models P_i^{RG}$.

Suppose that $\delta^{RG}(q_o^{RG}, s) = q$. If $(\exists \sigma_{i,j} \in \Sigma_{RE})\ \delta^{RG}(q, \sigma_{i,j})!$, then $(\forall r = 1,...,h)\ \delta^r(q_o^r, P_{G^r}(s)) \in Q^r_{PBS,i,j}$ according to (ii)(a). Thus, $(\forall r = 1,...,h)\ \delta^r(q_o^r, P_{G^r}(s)) \models P_i^r$. It is also true that $\delta^G(q_o^G, P_G(s)) \in Q^G_{PBS,i,j}$, and $(\forall r = 1,...,h)\ \delta^{R^r_{i,j}}(q_o^{R^r_{i,j}}, P_{R^{G^r}_{i,j}}(s)) = q_o^{R^r_{i,j}}$ according to Proposition 9. Besides, $\delta^{R_c}(q_o^{R_c}, s) = q_i$. According to assumption (ii), each mode in each plant component is reachable by itself, so $\exists s_1 \in \Sigma_i^*$ such that $\delta(q_o, s_1) = \delta(q_o, P_G(s))$. It is natural that $\sigma_i s_1 \in P_G^{-1}(L(\mathbf{G}))$. Besides, since $s_1 \in \Sigma_i^* \cap L(\mathbf{G})$ and $L(\mathbf{G}) \subseteq P_G(L(\mathbf{R}))$ according to Lemma 11, then $\delta^{R_c}(q_o^{R_c}, \sigma_i s_1)!$ i.e. $\sigma_i s_1 \in L(\mathbf{R})$. Since $L(\mathbf{RG}) = P_G^{-1}(L(\mathbf{G})) \cap L(\mathbf{R})$, $\sigma_i s_1 \in L(\mathbf{RG})$. Since $\delta(q_o, s_1) = \delta(q_o, P_G(s))$, then $(\forall r = 1,...,h)\ \delta^r(q_o^r, P_{G^r}(\sigma_i s_1)) \in Q^r_{PBS,i,j}$. Since $s_1 \in \Sigma_i^*$, $\delta^{R_c}(q_o^{R_c}, \sigma_i s_1) = q_i$. In this simple case in each component that involves $\mathbf{Mode}_i$, there must be an equal number of occurrences of ETE and EYE with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ within the scope of $\mathbf{G}^r$ $(r = 1,...,h)$ in s_1, so $\delta^{R^r_{i,j}}(q_o^{R^r_{i,j}}, \sigma_i s_1) = q_o^{R^r_{i,j}}$ according to the definition of LEMRS. Thus, $\delta^R(q_o^R, \sigma_i s_1) = \delta^R(q_o^R, s)$. Since $\delta^R(Q_O^r, \sigma_i s_1) = \delta^R(q_o^R, s)$ and $\delta(q_o, P_G(\sigma_i s_1)) = \delta(q_o, P_G(s))$, $\delta^{RG}(q_o^{RG}, s) = q = \delta^{RG}(q_o^{RG}, \sigma_i s_1)$. Since σ_i is the MIE for $\mathbf{Mode}_i$ and $s_1 \in \Sigma_i^*, q \models P_i^{RG}$.

On the other hand, if $(\nexists\ \sigma_{i,j} \in \Sigma_{RE})\ \delta^{RG}(q, \sigma_{i,j})!$, then s can always be expressed as $s = s_1 s_2$ where $(\exists \sigma_{i,j} \in \Sigma_{RE})\ \delta^{RG}(q_o^{RG}, s_1 \sigma_{i,j})!$ and $s_2 \in \Sigma_i^*$. Then $\delta^{RG}(q_o^{RG}, s_1) \models P_i^{RG}$ based on the proof above. According to the definition of segregated mode predicate, since $s_2 \in \Sigma_i^*, \delta^{RG}(q_o^{RG}, s_1 s_2) \models P_i^{RG}$.

(b) $(\forall q \in Q^{RG}, q \neq q_o^{RG})(\exists i \in 1,...,n)(\forall j \in 1,...,n, i \neq j)\ [q \models P_i^{RG} \wedge q \not\models P_j^{RG}]$.

Prove by contradiction. Assume that $q \models P_i^{RG}$ and $q \models P_j^{RG}, i \neq j$. Then $s_1 \in \Sigma_i^*, \delta^{RG}(q_o^{RG}, \sigma_i s_1) = q$ and $(\exists s_2 \in \Sigma_j^*)\ \delta^{RG}(q_o^{RG}, \sigma_j s_2) = q$. Since events in both s_1 and s_2 need to occur in $\mathbf{R}_c$ according to synchronization, then naturally $\delta^{R_c}(q_o^{R_c}, \sigma_i s_1) = q_i$ and $\delta^{R_c}(q_o^{R_c}, \sigma_j s_2) = q_j$. However, $\{q_i\} \cap \{q_j\} = \emptyset$, which contradicts $\delta^{R_c}(q_o^{R_c}, \sigma_i s_1) = \delta^{R_c}(q_o^{R_c}, \sigma_j s_2)$ according to the assumption.

□

Appendix C

Problem Solvability

In Chapter 3 we raised the question: Suppose the system is currently operating in $\mathbf{Mode}_1$ (say), and the user wants to keep it there. What guarantees that the proposed reconfiguration approaches keep the system in a given mode when there is no demand to leave it? This question relates to the solvability of reconfiguration problems for DES.

First, let's study an illustrative example. Let's consider the following DES **GAPPC** with two modes. For $\mathbf{Mode}_1$, $\Sigma_1 = \{1, 2, 3\}$. For $\mathbf{Mode}_2$, $\Sigma_1 = \{1, 4, 5\}$.

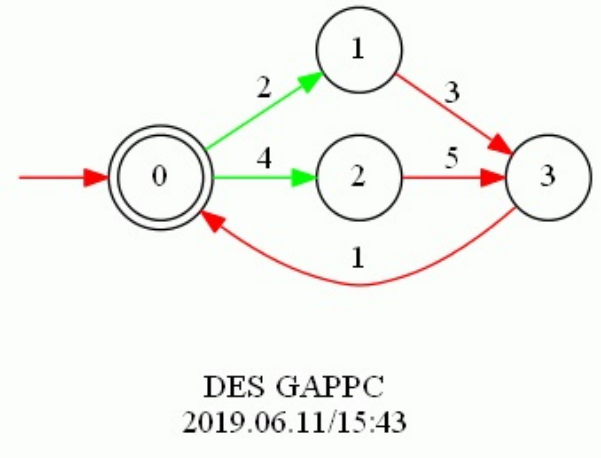

Figure C..1: Plant **GAPPC** with two modes

Suppose the system is currently operating in $\mathbf{Mode}_1$ (say at state 0), and the user wants to keep it there. Evidently, without the help of the proposed bidirectional reconfiguration approach, we cannot prevent it from uncontrollably slipping into $\mathbf{Mode}_2$, i.e. we cannot disable event 4 at state 0.

However, if we add a bidirectional reconfiguration specification and generate the syn-

chronous product of the plant and the reconfiguration specification, this problem will be solved. Please see the following BRS **RAPPC**.

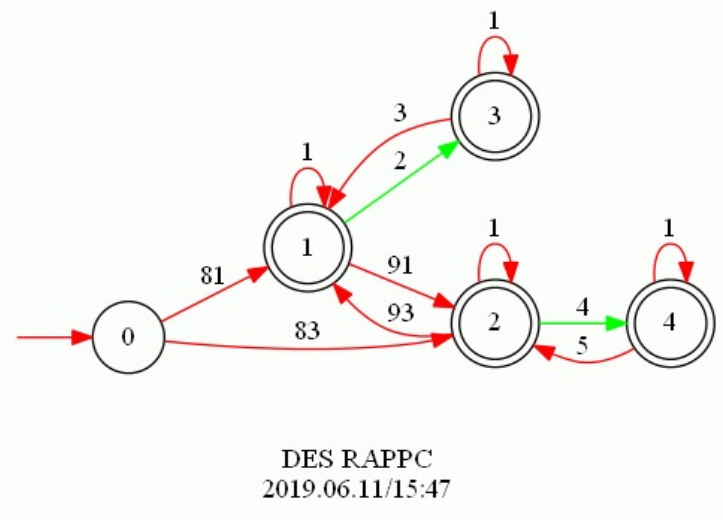

Figure C..2: BRS **RAPPC**

In **RAPPC**, events 81 (83) is a mode initialization event for $\mathbf{Mode}_1$ ($\mathbf{Mode}_2$). Event 91 (93) is a reconfiguration event from $\mathbf{Mode}_1$ to $\mathbf{Mode}_2$ (from $\mathbf{Mode}_2$ to $\mathbf{Mode}_1$). All the MIE and RE are controllable.

We then compute

$$\mathbf{RGAPPC} = \mathbf{Sync}(\mathbf{GAPPC},\mathbf{RAPPC})\ (7{,}12)\ \text{Blocked events} = \text{None}$$

Please see the following reconfiguration plant **RGAPPC**.

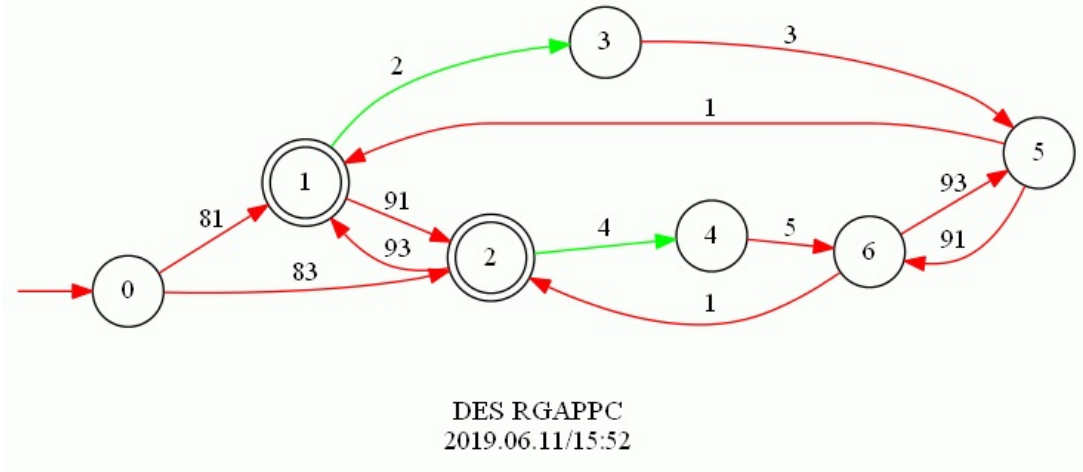

Figure C..3: Reconfiguration plant **RGAPPC**

From **RGAPPC** we can see that when the system is currently operating in $\mathbf{Mode}_1$, i.e., at state 1, 3, and 5, the system cannot slip into $\mathbf{Mode}_2$ uncontrollably. The only way to enter $\mathbf{Mode}_2$ is the occurrence of the reconfiguration event 91, but every reconfiguration event is controllable. That's why a reconfiguration event needs to be controllable.

Furthermore, if we apply the proposed multiple reconfiguration approach, the explanations will be clearer. Please see the following core multiple reconfiguration specification **RCAPPC**.

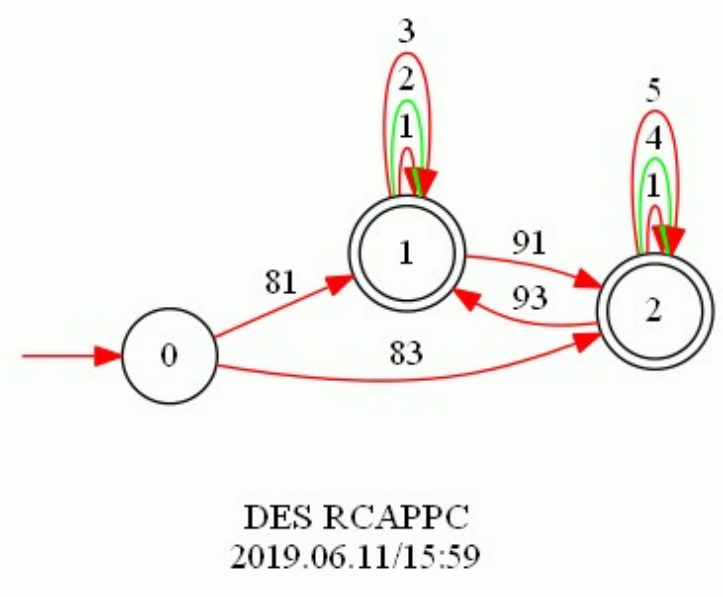

Figure C..4: CMRS **RCAPPC**

The CMRS provides a core structure for the multiple reconfiguration. All events in Σ_2 are self-looped at state 1 and all events in Σ_2 are self-looped at state 2. Between these two states, there are reconfiguration events 91 and 93. Namely, unless a reconfiguration event occurs, the system cannot slip into another mode from the current mode. Since every reconfiguration event is controllable according to our definition, we can certainly prevent a system from uncontrollably slipping into another mode from the current mode.

As for the extra multiple reconfiguration specification, the objective is to disable reconfiguration events when the system is at one of some states (e.g., states 1 and 2 of **APPC**). Please see the following EMRS **R12APPC**.

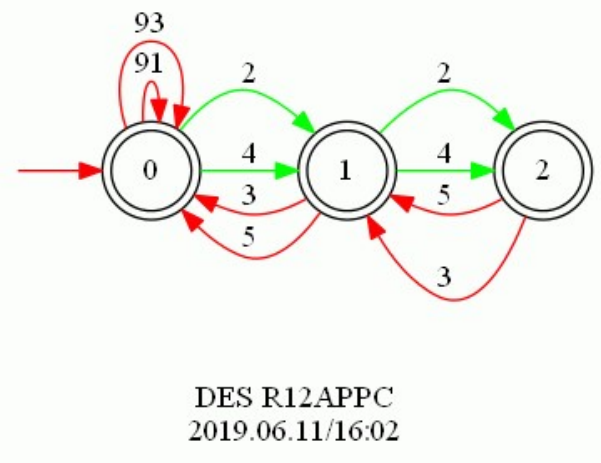

Figure C..5: EMRS **R12APPC**

We then compute

RGMAPPC = **Sync**(**GAPPC**,**RCAPPC**,**R12APPC**) (7,12) Blocked events = None

true = **Isomorph**(**RGAPPC**,**RGMAPPC**;identity)

We can see that the reconfiguration plant obtained from the multiple reconfiguration approach is the same as the RP obtained from the bidirectional reconfiguration approach.

In fact, the purpose of the proposed reconfiguration approaches is to generate a reconfiguration plant with the help of synchronization. A reconfiguration plant is still a plant without control. In the original plant, different modes are intertwined. Thus, even with the help of the "**Supcon**" function, we still cannot prevent the system from uncontrollably slipping into another mode from the current mode. However, in the reconfiguration plant, different modes are decoupled sufficiently and are connected by reconfiguration events, without discarding any information about the original plant. Namely, we can prevent the system from uncontrollably slipping into another mode from the current mode. So in the reconfiguration plant, we can manage reconfiguration behaviors and preserve system dynamics.

Moreover, we provide the following propositions with their proofs to show this fact.

Proposition 14 is for the bidirectional reconfiguration approach. It states that for a system with two modes, if in its reconfiguration plant there are two states satisfying different SMP that are connected by a transition, then the event labeling this transition must be a reconfiguration event. Proposition 14 implies that the reconfiguration plant

cannot slip into another mode from the current mode without the help of reconfiguration.

Proposition 14. In a bidirectional reconfiguration problem, $(\forall i, j = 1, 2, i \neq j)(\forall q \models P_i^{RG})(\forall q^{'} \models P_j^{RG})$ $[(\exists \sigma \in \Sigma^{RG})\delta^{RG}(q, \sigma) = q^{'} \Rightarrow \sigma \in \Sigma_{RE}]$.

Proof. Here we prove a stronger statement that $(\forall i, j \in 1, ..., n, i \neq j)(\forall q \models P_i^{RG})(\forall q^{'} \models P_j^{RG})$ $[(\exists \sigma \in \Sigma^{RG})\delta^{RG}(q, \sigma) = q^{'} \Rightarrow \sigma \in \Sigma_{RE}]$.

Prove by contradiction.

Suppose that $(\exists i, j \in 1, ..., n, i \neq j)(\exists q \models P_i^{RG})(\exists q^{'} \models P_j^{RG})$ $[(\exists \sigma \in \Sigma^{RG})\delta^{RG}(q, \sigma) = q^{'} \Rightarrow \sigma \notin \Sigma_{RE}]$. Then $\sigma \in \Sigma_{MIE} \cup \Sigma$.

When $\sigma \in \Sigma_{MIE}$, according to the structure of the BRS, every MIE is defined at the initial state of **RG**. However, according to the definition of SMP, the initial state of **RG** doesn't satisfy any of the n SMP, which contradicts to $q \models P_i^{RG}$.

When $\sigma \in \Sigma$, according to the definition of SMP, $\sigma \notin \Sigma_i$. Thus $\sigma \in \Sigma - \Sigma_i$, say $\sigma \in \Sigma_k$. According to the definition of SMP, there exists a string $s \in \Sigma_i^*$ such that $\delta^{RG}(q_o^{RG}, \sigma_i s) = q$, where $\sigma_i \in \Sigma_{MIE}$ is the mode initialization event of $\mathbf{Mode}_i$. Since $\Sigma^{RG} = \Sigma^R = \Sigma \dot{\cup} \Sigma_{MIE} \dot{\cup} \Sigma_{RE}$ and $s \in L(\mathbf{RG})$, $s \in L(\mathbf{R})$. Then $\delta^R(q_o^R, \sigma_i s) \in \{q_i^0, ..., q_i^b\}$. Since $\delta^{RG}(q, \sigma) = q^{'}$, then $\delta^R(q_o^R, \sigma_i s \sigma)!$. However, since $\sigma \in \Sigma - \Sigma_i$, it is not defined in any state in $\{q_i^0, ..., q_i^b\}$, which contradicts to $\delta^R(q_o^R, \sigma_i s) \in \{q_i^0, ..., q_i^b\}$. □

Proposition 15 is for the (both monolithic and localized) multiple reconfiguration approach. It states that for a system with n modes, if in its reconfiguration plant there are two states satisfying different SMP that are connected by a transition, then the event labeling this transition must be a reconfiguration event. Proposition 15 implies that the reconfiguration plant cannot slip into another mode from the current mode without the help of reconfiguration.

Proposition 15. In a (both monolithic and localized) multiple reconfiguration problem, $(\forall i, j = 1, ..., n, i \neq j)(\forall q \models P_i^{RG})(\forall q^{'} \models P_j^{RG})$ $[(\exists \sigma \in \Sigma^{RG})\delta^{RG}(q, \sigma) = q^{'} \Rightarrow \sigma \in \Sigma_{RE}]$.

Proof. Prove by contradiction.

Suppose that $(\exists i, j \in 1, ..., n, i \neq j)(\exists q \models P_i^{RG})(\exists q^{'} \models P_j^{RG})$ $[(\exists \sigma \in \Sigma^{RG})\delta^{RG}(q, \sigma) = q^{'} \Rightarrow \sigma \notin \Sigma_{RE}]$. Then $\sigma \in \Sigma_{MIE} \cup \Sigma$.

When $\sigma \in \Sigma_{MIE}$, according to the structure of the CMRS, every MIE is defined at

the initial state of **RG**. However, according to the definition of SMP, the initial state of **RG** doesn't satisfy any of the n SMP, which contradicts to $q \models P_i^{RG}$.

When $\sigma \in \Sigma$, according to the definition of SMP, $\sigma \notin \Sigma_i$. Thus $\sigma \in \Sigma - \Sigma_i$, say $\sigma \in \Sigma_k$. According to the definition of SMP, there exists a string $s \in \Sigma_i^*$ such that $\delta^{RG}(q_o^{RG}, \sigma_i s) = q$, where $\sigma_i \in \Sigma_{MIE}$ is the mode initialization event of $\mathbf{Mode}_i$. Since $\Sigma^{RG} = \Sigma^{R_c} = \Sigma \dot{\cup} \Sigma_{MIE} \dot{\cup} \Sigma_{RE}$ and $s \in L(\mathbf{RG})$, $s \in L(\mathbf{R}_c)$. Then $\delta^{R_c}(q_o^{R_c}, \sigma_i s) = q_i$. Since $\delta^{RG}(q, \sigma) = q^{'}$, then $\delta^{R_c}(q_o^{R_c}, \sigma_i s \sigma)!$. However, since $\sigma \in \Sigma - \Sigma_i$, it is not defined at q_i, which contradicts to $\delta^{R_c}(q_o^{R_c}, \sigma_i s) = q_i$. □

According to the two propositions, we can see that the reconfiguration events prevent the reconfiguration plant from uncontrollably slipping into another mode from the current mode. The reconfiguration events can be incorporated in the reconfiguration plant by the "**Sync**" operation.

As for the "**Supcon**" function, we need to use it when we need control, i.e., when there are behavioral specifications. Besides, if it is required to trigger reconfiguration under a specific logic, "**Supcon**" is also in demand.

Although this approach simply uses "**Sync**" as the main operation, it is not trivial. Though less fancy, we use a simpler tool rather than a more complex tool to solve a problem, which is satisfactory at this stage.

However, though the proposed approaches are able to prevent the reconfiguration plant from uncontrollably slipping into another mode, this situation violates the physical interpretation of the plant DES. Still consider the example in Figure D..1, events 2 and 4 are uncontrollable events hence are never under control. Then, no matter what mode the system is currently in, events 2 and 4 are both possible to occur as long as the system is at state 0. Thus, in practice event 4 can never be prevented from occurring when the system is in $\mathbf{Mode}_1$. We attribute this phenomenon to the solvability of the problem and the system modeling.

On the one hand, in the plant DES, when there exists some uncontrollable event that serves as an exit event with respect to some pair of modes, the corresponding reconfiguration problem is unsolvable. The reason lies in the physical interpretation of the system. We summarize this statement as the following theorem in the context of multiple reconfiguration. The statement for bidirectional reconfiguration can be similarly

addressed.

Theorem 10. [Solvability of Problem 4]. Denote by $\mathbf{G} = (Q, \Sigma, \delta, q_o, Q_m) = \mathbf{G}^1||...||\mathbf{G}^h$ the plant DES formed by synchronization, where $\mathbf{G}^k$ is the k^{th} component. $\mathbf{Mode}_1, ..., \mathbf{Mode}_n$ are n different modes of $\mathbf{G}$ distinguished by event alphabets $\Sigma_1, ..., \Sigma_n$. Also denote by $\Sigma^k_{ETE,i,j}$ the set of exit events with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ in $\mathbf{G}^k$. The multiple reconfiguration problem is solvable by the proposed reconfiguration approaches iff

$$(\forall i, j \in 1, ..., n)(\forall k \in 1, ..., h)(\forall \sigma \in \Sigma^k_{ETE,i,j})\ \sigma \in \Sigma_c \tag{C..1}$$

The multiple reconfiguration problem is partly solvable with respect to $\mathbf{Mode}_i$ and $\mathbf{Mode}_j$ by the proposed reconfiguration approaches iff

$$(\forall k \in 1, ..., h)(\forall \sigma \in \Sigma^k_{ETE,i,j})\ \sigma \in \Sigma_c \tag{C..2}$$

This theorem states that for each pair of modes, if for each plant component, the exit events with respect to this pair of modes are all controllable, then the multiple reconfiguration problem is partly solvable with respect to these two modes. Naturally, if the multiple reconfiguration problem is partly solvable with respect to all pairs of modes in the system, then this problem is solvable.

On the other hand, sometimes the multiple reconfiguration problem is not solvable since the modes are not defined properly in DES. If users can avoid uncontrollable exit events in system modeling, then the multiple reconfiguration problem is always solvable by the proposed reconfiguration approaches.

Therefore, we conclude that the proposed reconfiguration approaches are competent to prevent the reconfiguration plant from uncontrollably slipping into another mode; this situation should always be avoided by suitable system modeling. Otherwise, the reconfiguration problem is completely unsolvable or partly unsolvable.

Appendix D

Case Study

We apply the proposed reconfiguration approaches to two benchmark production systems FESTO and EnAS that are used by researchers in different universities for study.

D..1 FESTO

FESTO is a reconfigurable control system [2]. The main function of FESTO is to drill holes in workpieces. It has three units:

(i) Distribution transmits cylindrical workpieces to the next unit from stock. It consists of a pneumatic feeder and a converter;

(ii) Test executes type, height and color tests on workpieces. If a workpiece satisfies these three tests, it will be transmitted to the next unit. It consists of a detector, a tester, and an elevator;

(iii) Processing is composed of a rotating disk, a drill machine, and a control machine. The rotating disk transports workpieces between the input, drilling, control and output positions.

Suppose that the processing unit can operate with two drilling machines (Drill1 and Drill2) and FESTO has three production modes according to the number of workpieces.

The FESTO system is represented by eight functional behaviors as shown in Figure D..1 where FBf is the f^{th} FESTO behavior in FESTO system, i.e.,

- FB1 = op1; op2; op3; op4 (Error in Test unit.);

- FB2 = op1; op2; op3; op5; op61; op7; op62; op11; op63; op12 (Light1);
- FB3 = op1; op2; op3; op5; op61; op7 (Error in Drill1.);
- FB4 = op1; op2; op3; op5; op61; op8; op62; op11; op63; op12 (Light2);
- FB5 = op1; op2; op3; op5; op61; op9; op62; op11; op63; op12 (Medium);
- FB6 = op1; op2; op3; op5; op61; op9 (Error in Drill1 or in Drill2.);
- FB7 = op1; op2; op3; op5; op61; op10; op62; op11; op63; op12 (High);
- FB8 = op1; op2; op3; op5; op61; op10 (Error in Drill1 or in Drill2.).

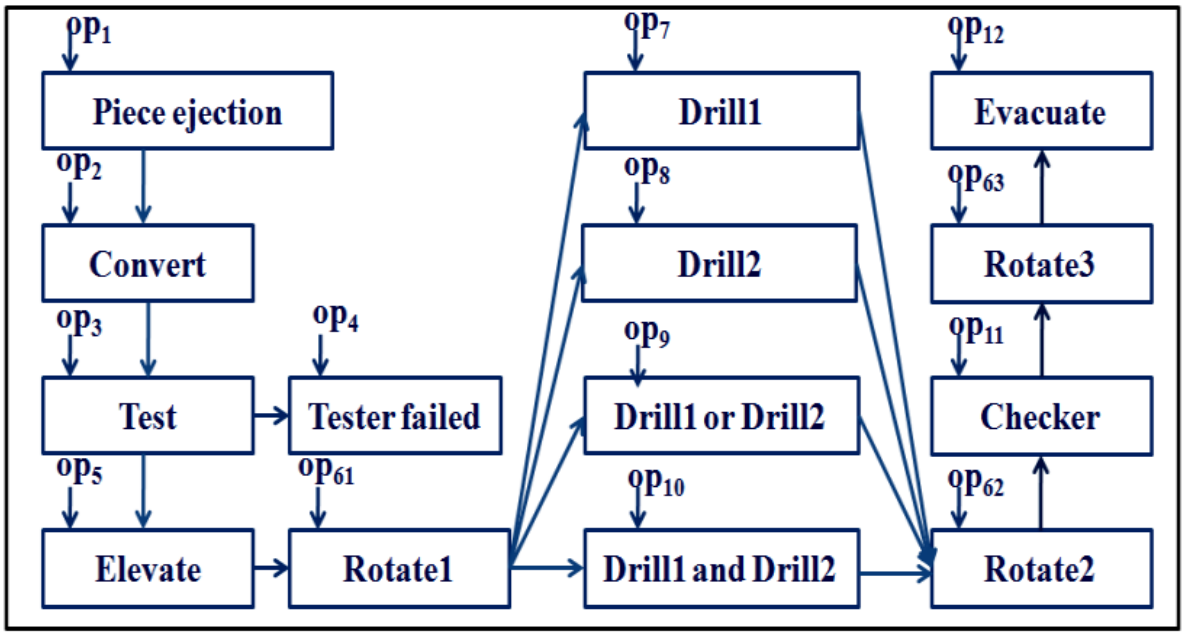

Figure D..1: Operating process of FESTO [2]

The four production modes are modeled by these functional behaviors as follows:

- **Light1** is represented by FB1, FB2 and FB3;
- **Light2** is represented by FB4;
- **Medium** is represented by FB5, FB6;
- **High** is represented by FB7, FB8.

If the two drilling machines are broken down, then the system will be stopped. In case of errors or user requirements, FESTO should be able to switch production modes automatically.

The reconfiguration behavior of FESTO is depicted in a state diagram in Figure D..2. The state diagram is composed of four states L1, L2, M, and H, which represent the four production modes of FESTO system. The transitions describe the switching between modes when FESTO applies reconfiguration owing to drill machine errors or user requirements.

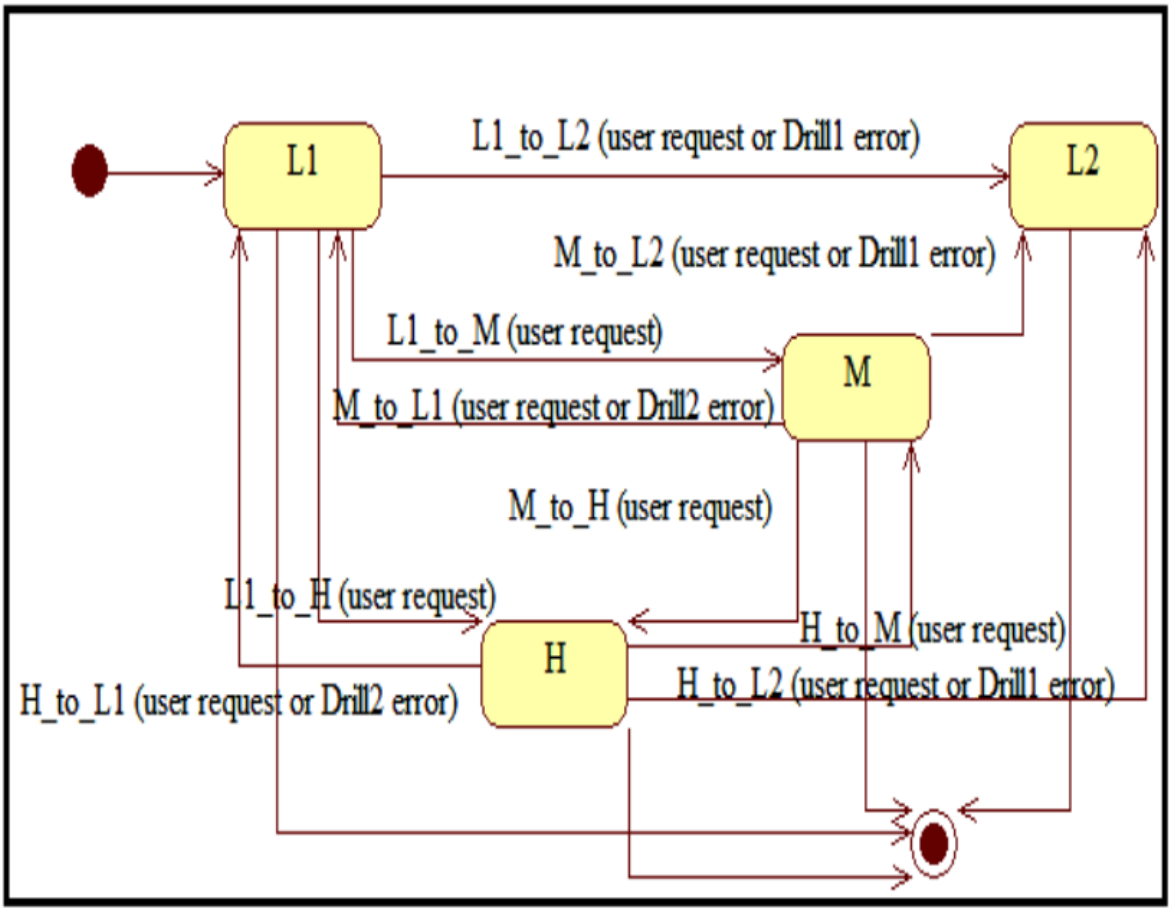

Figure D..2: State diagram of reconfiguration behaviors in FESTO [2]

In Figure D..1, FESTO is modeled according to operations at each state, instead of transitions between states. Similarly, the modes are distinguished according to operations at each state, namely state-based. However, our reconfiguration approach is event-based. Thus, we cannot directly apply our reconfiguration approach to the FESTO system.

Therefore, we need to convert the state-based FESTO system to an equivalent event-based system FESTO'. According to the conversion procedure from a job-on-node diagram to a job-on-arc diagram described in [58], the following precedence graph is provided as shown in Figure D..3.

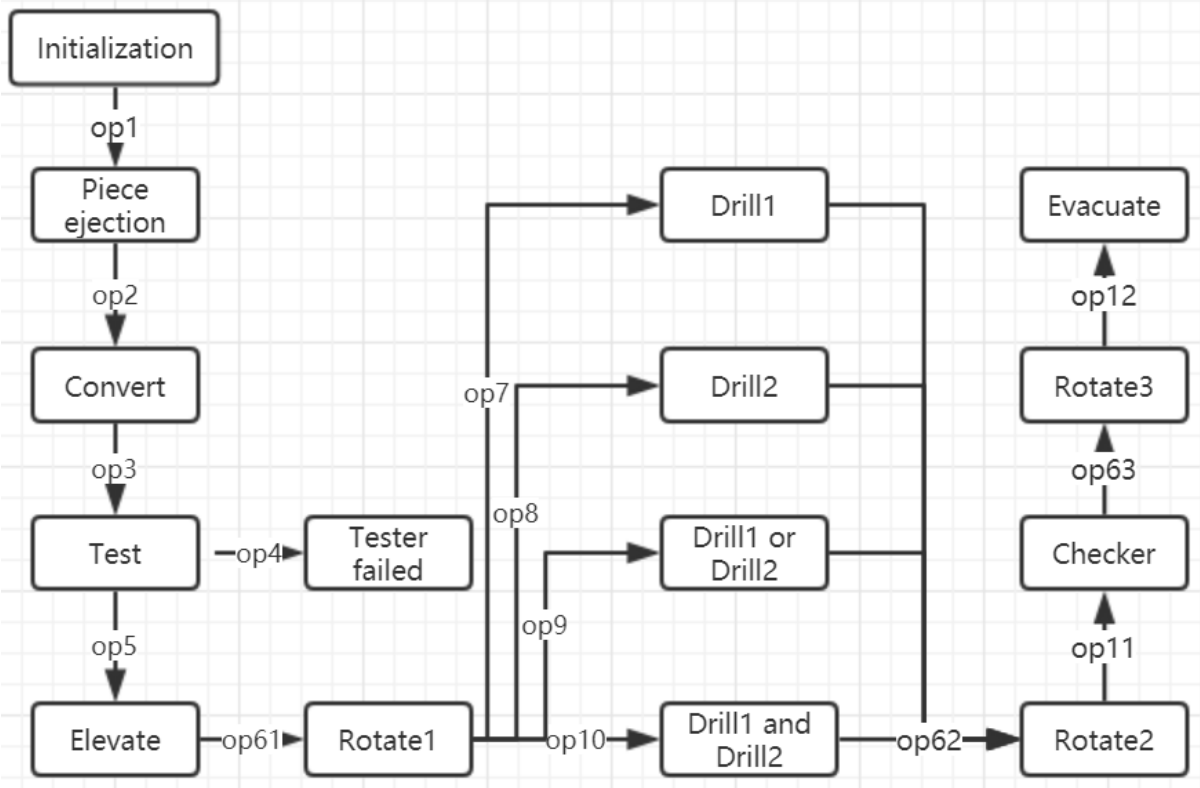

Figure D..3: Job-on-arc precedence graph of FESTO'

According to the job-on-arc precedence diagram of FESTO', the proposed **monolithic** multiple reconfiguration approach can be applied to the system. Before that, the FESTO' system is required to be converted to a DES model. Figure D..4 shows the DES model of FESTO'. The correspondence of states and events in FESTO' and its DES model is shown in Table D..1.

Note that the initialization state doesn't exist in FESTO. But after the conversion from job-on-node to job-on-arc, in order to be consistent with its DES model, FESTO' has an extra state "initialization".

States 4, 7, 9, 10 and 12 are the final states of FB1 to FB8. Thus, we mark them to show that the functional behaviors end. And we also add five transitions labeled by

uncontrollable events 40, 70, 90, 100 and 120 from these five states back to state 0 to reset the system.

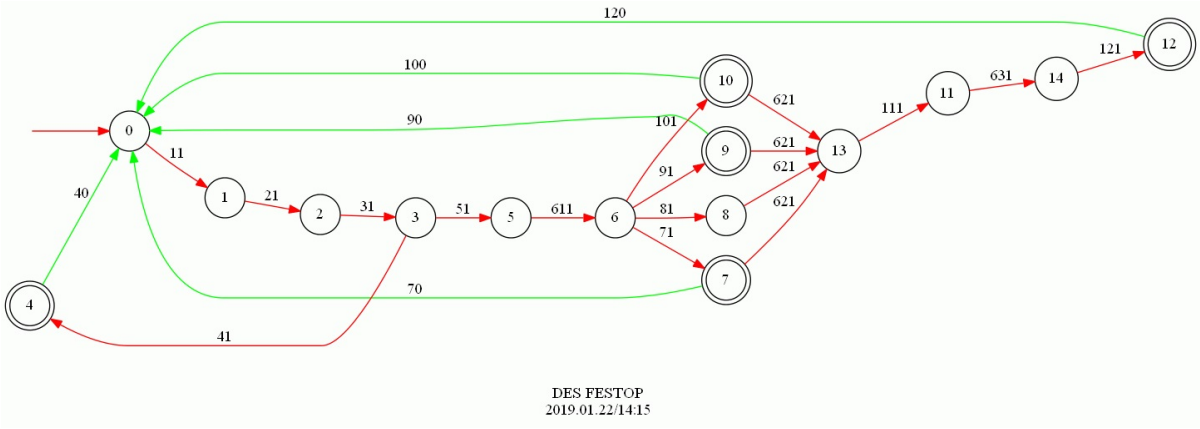

Figure D..4: DES model of FESTO'

Table D..1: Correspondence of states and events in FESTO' and its DES model

State in FESTO'	**State in the DES model**	**Event in FESTO'**	**Event in the DES model**
Piece ejection	1	op1	11
Convert	2	op2	21
Test	3	op3	31
Elevate	5	op4	41
Tester failed	4	op5	51
Rotate1	6	op61	611
Drill1	7	op7	71
Drill2	8	op8	81
Drill1 or Drill2	9	op9	91
Drill1 and Drill2	10	op10	101
Rotate2	13	op62	621
Checker	11	op11	111
Rotate3	14	op63	631
Evacuate	12	op12	121
Initialization	0		

For the four modes, say $\mathbf{Mode}_1$ is **Light1**, $\mathbf{Mode}_2$ is **Light2**, $\mathbf{Mode}_3$ is **Medium** and $\mathbf{Mode}_4$ is **High**. Then, $\Sigma_1 = \{11, 21, 31, 40, 41, 51, 70, 71, 111, 120, 121, 611, 621, 631\}$, $\Sigma_2 = \{11, 21, 31, 51, 81, 111, 120, 121, 611, 621, 631\}$, $\Sigma_3 = \{11, 21, 31, 51, 90, 91, 111, 120, 121, 611, 621, 631\}$ and $\Sigma_4 = \{11, 21, 31, 51, 100, 101, 111, 120, 121, 611, 621, 631\}$. According to the event alphabets and the DES model in Figure D..4, we can see that $Q^G_{PBS,1,2} =$

$Q^G_{PBS,1,3} = Q^G_{PBS,1,4} = Q^G_{PBS,2,3} = Q^G_{PBS,2,4} = Q^G_{PBS,3,4} = \{0, 1, 2, 3, 5, 6, 11, 12, 13, 14\}$. Thus, $\Sigma^G_{ETE,1,2} = \Sigma^G_{ETE,1,3} = \Sigma^G_{ETE,1,4} = \Sigma^G_{ETE,2,3} = \Sigma^G_{ETE,2,4} = \Sigma^G_{ETE,3,4} = \{41, 71, 81, 91, 101\}$ and $\Sigma^G_{EYE,1,2} = \Sigma^G_{EYE,1,3} = \Sigma^G_{EYE,1,4} = \Sigma^G_{EYE,2,3} = \Sigma^G_{EYE,2,4} = \Sigma^G_{EYE,3,4} = \{40, 70, 90, 100, 621\}$. Besides, since there is only one plant component, $k_{1,2} = k_{1,3} = k_{1,4} = k_{2,3} = k_{2,4} = k_{3,4} = 1$.

According to Figure D..2, FESTO can only start in mode **Light1**, so there is only one mode initialization event 19 for $\mathbf{Mode}_1$. Besides, there is no reconfiguration from mode **Light2** to other modes, so the reconfiguration events are $\sigma_{1,2} = 129$, $\sigma_{1,3} = 139$, $\sigma_{1,4} = 149$, $\sigma_{3,1} = 319$, $\sigma_{3,2} = 329$, $\sigma_{3,4} = 349$, $\sigma_{4,1} = 419$, $\sigma_{4,2} = 429$ and $\sigma_{4,3} = 439$. Then the core multiple reconfiguration specification **FESTORC** and the extra multiple reconfiguration specifications **FESTOR12**, **FESTOR13**, **FESTOR14**, **FESTOR23**, **FESTOR24**, **FESTOR34** are provided below.

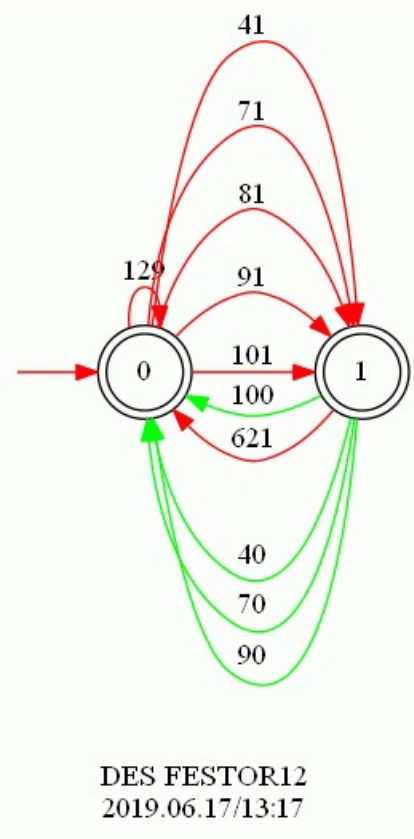

Figure D..5: EMRS **FESTOR12**

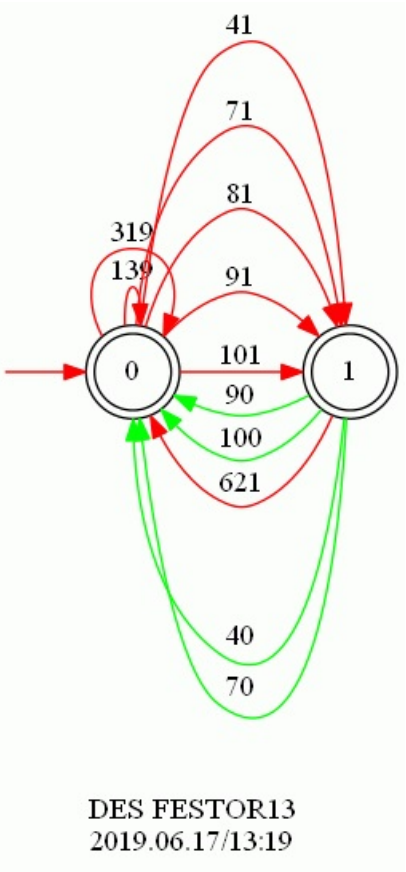

Figure D..6: EMRS **FESTOR13**

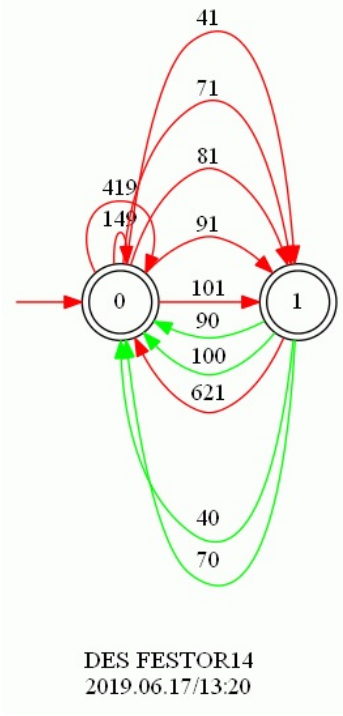

Figure D..7: EMRS **FESTOR14**

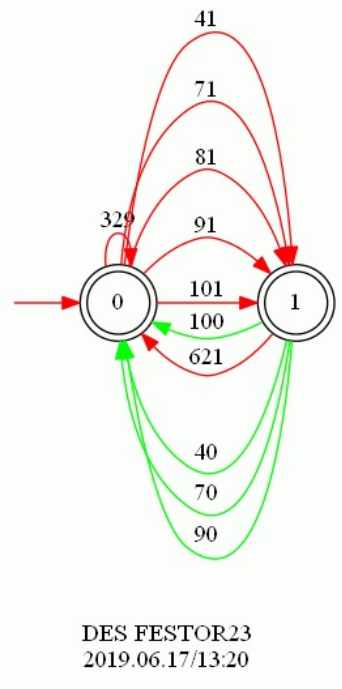

Figure D..8: EMRS **FESTOR23**

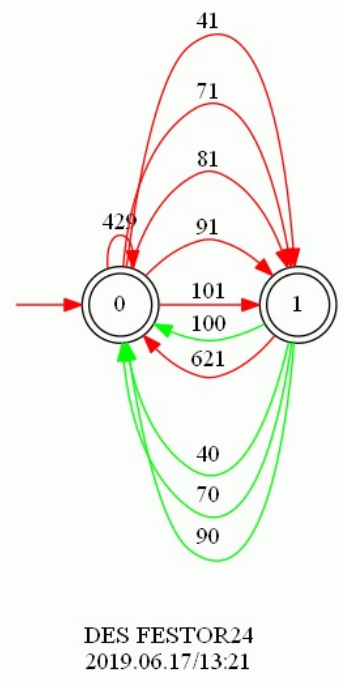

Figure D..9: EMRS **FESTOR24**

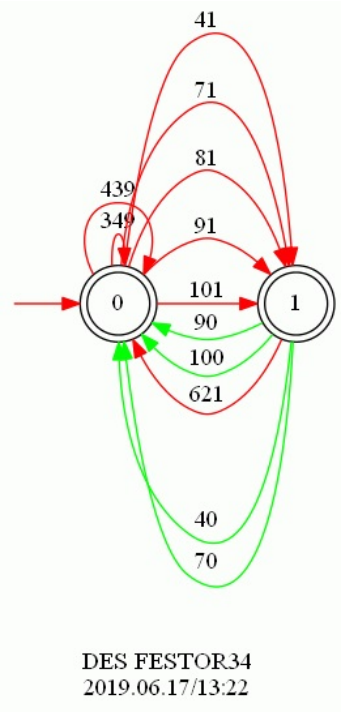

Figure D..10: EMRS **FESTOR34**

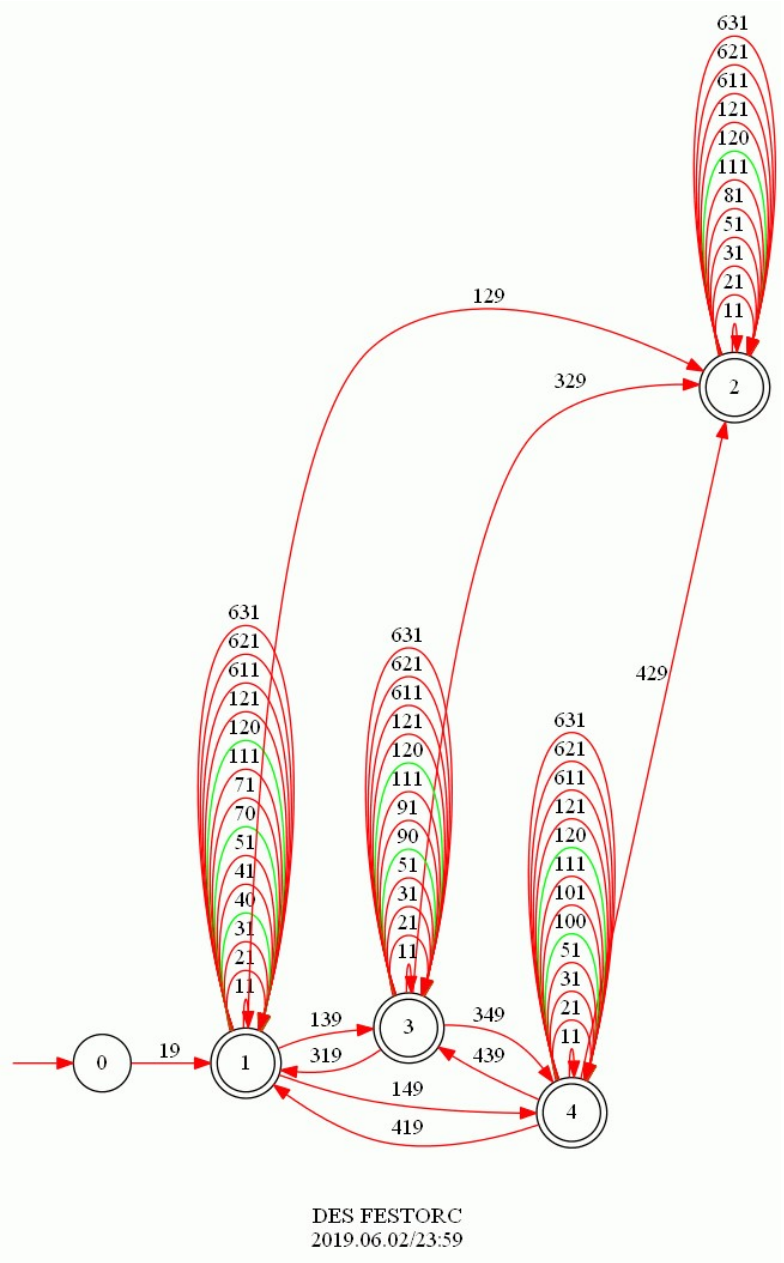

Figure D..11: CMRS **FESTORC**

We then compute

FESTORG = **Sync**(**FESTO'**, **FESTORC**, **FESTOR12**, **FESTOR13**, **FESTOR14**, **FESTO23**, **FESTO24**, **FESTO34**) (46,140) Blocked events = None

The resulting reconfiguration plant **FESTORG** is shown as follows. However, it is too large to be shown clearly. The reader interested in this aspect may obtain it in TCT.

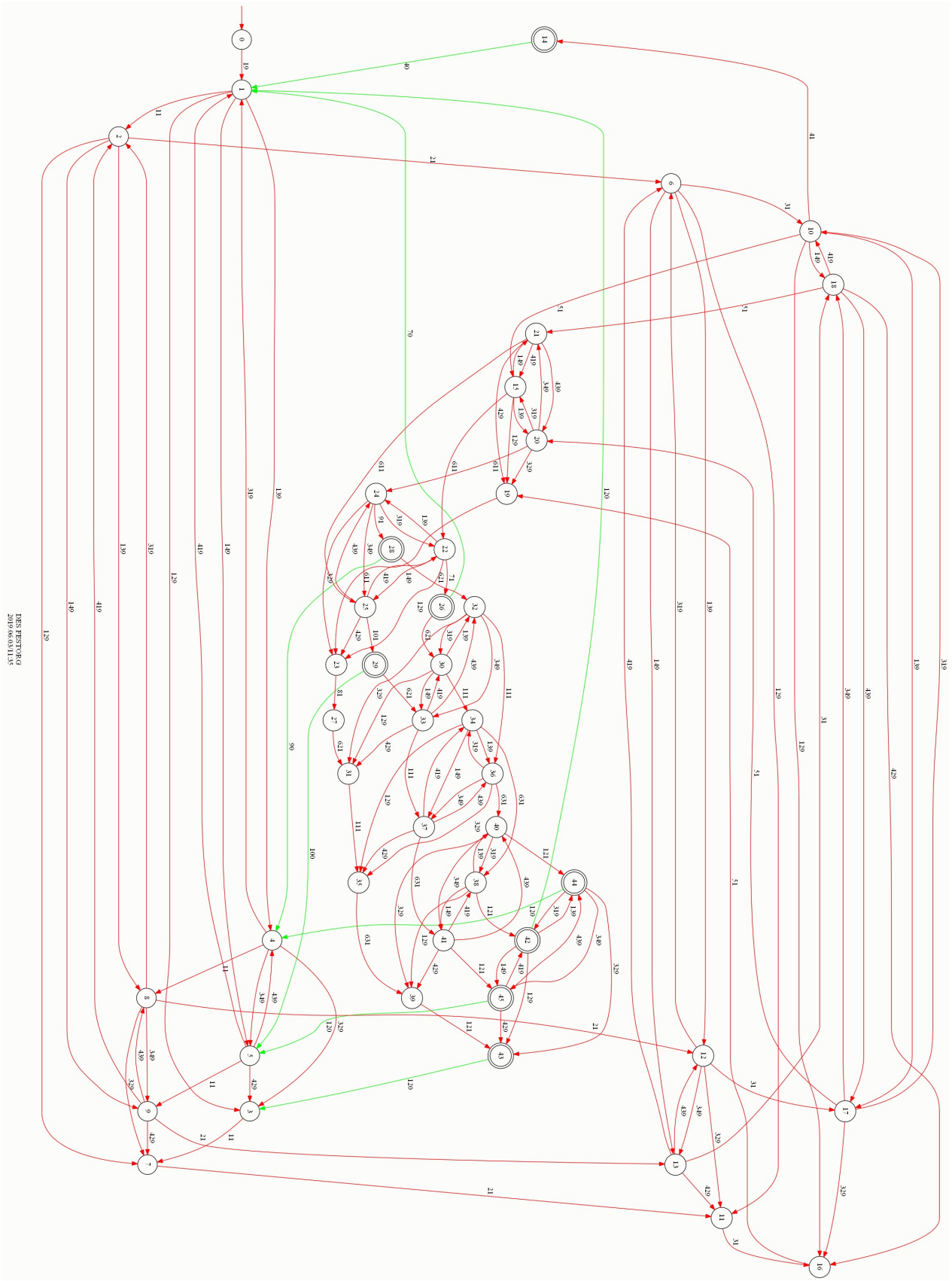

Figure D..12: Reconfiguration plant **FESTORG**

Since FESTO can only start in $\mathbf{Mode}_1$, **FESTORG** can only start in $\mathbf{Mode}_1$ via event 19. Each reconfiguration event occurs only when FESTO is at a public state with respect to the two modes corresponding to the reconfiguration event. Apart from the reconfiguration from $\mathbf{Mode}_2$, each pair of reconfiguration events can occur back-and-forth. Moreover, each mode is nonblocking separately in **FESTORG** and each state except for the initial state satisfies exactly one SMP. The reader interested in this aspect may check the five requirements by using the requirement checking program on the author's website (https://github.com/JasonZhangjc/). Note that the first and the third requirement are not met owing to the characteristics of FESTO.

D..2 EnAS

EnAS is also a reconfigurable control system [2]. The main function of EnAS system is to place workpieces inside tins subsequently to be closed with caps. EnAS is composed of the following components.

(i) A belt moves a particular pallet containing a cap and a tin;

(ii) Two jack stations (J1 and J2) place newly drilled workpieces from FESTO and close tins with caps;

(iii) Two gripper stations (G1 and G2) remove charged tins into storing stations (ST1 and ST2) from the belt.

EnAS has three production modes, according to the number of drilled workpieces, tins and caps. As shown in Figure D..13, EnAS system is represented by twelve functional behaviors where EBe is the e^{th} EnAS behavior in EnAS system.

- EB1= op1'; op2'; op3'; op4' (Policy1);
- EB2= op1'; op2' (error in J1.);
- EB3= op1'; op2'; op3' (error in G1.);
- EB4= op1'; op6'; op8'; op10'; op11' (Policy2);
- EB5= op1'; op6'; op8'; op10' (error in G2.);

- EB6= op1'; op2'; op5'; op11' (Policy3);
- EB7= op1'; op2'; op5' (error in G2.);
- EB8= op1'; op6'; op8'; op9'; op4' (Policy3);
- EB9= op1'; op6'; op8'; op9' (error in G1.);
- EB10= op1'; op6'; op7'; op8'; op10'; op11' (Policy4);
- EB11= op1'; op6'; op7'; op8' (error in J2.);
- EB12= op1'; op6'; op7'; op8'; op10' (error in G2.).

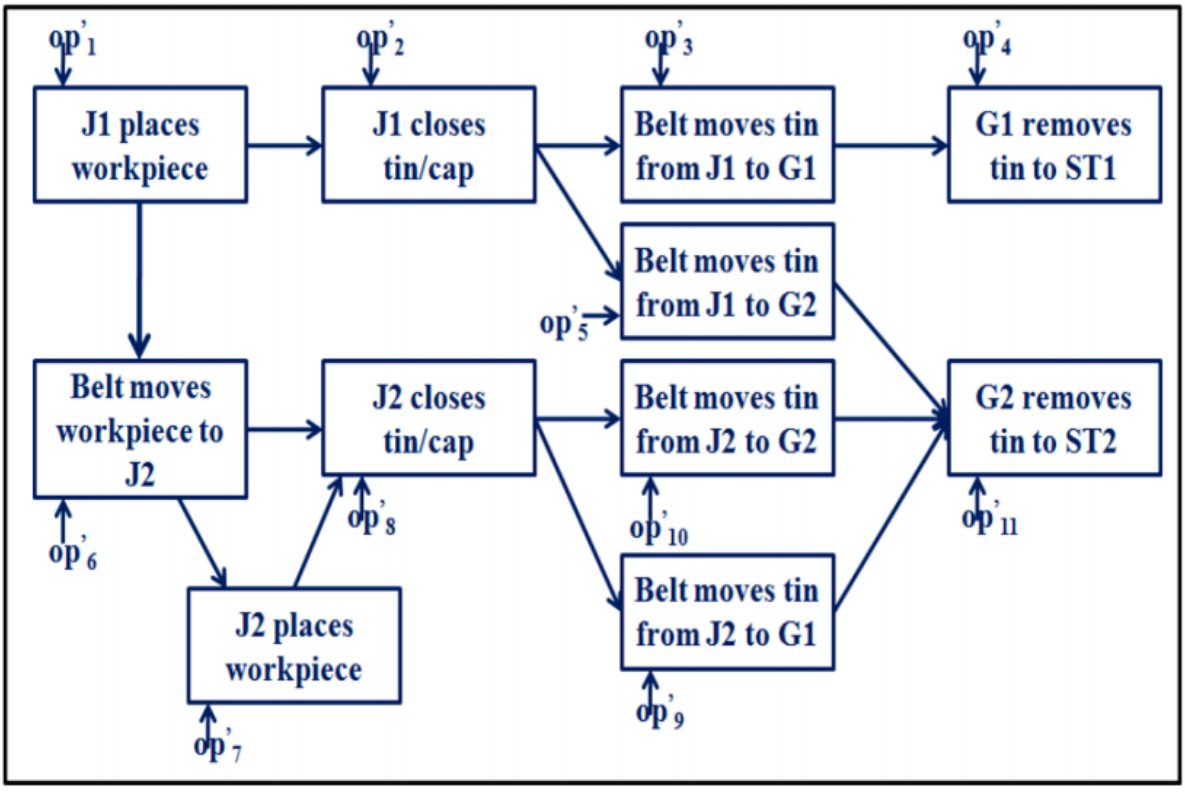

Figure D..13: Operating process of EnAS [2]

The four production modes are modeled by these functional behaviors as follows:

- **Policy1** is represented by EB1, EB2 and EB3;
- **Policy2** is described by EB4 and EB5;

- **Policy3** is represented by EB6, EB7, EB8 and EB9;
- **Policy4** is described by EB10, EB11 and EB1.

If the two jack stations are broken then the system will be stopped. Depending on any change in working environment caused by errors (i.e., error in J1/G1 or error in J2/G2) or user requirements, EnAS is required to be able to switch policies (modes) automatically without a halt.

As explained in Figure D..14, the state diagram is composed of four states P1, P2, P3 and P4 corresponding to the four possible behaviors of EnAS. The transitions represent the possible reconfiguration of this reconfigurable control system caused for instance by jack or gripper stations errors or user requirements.

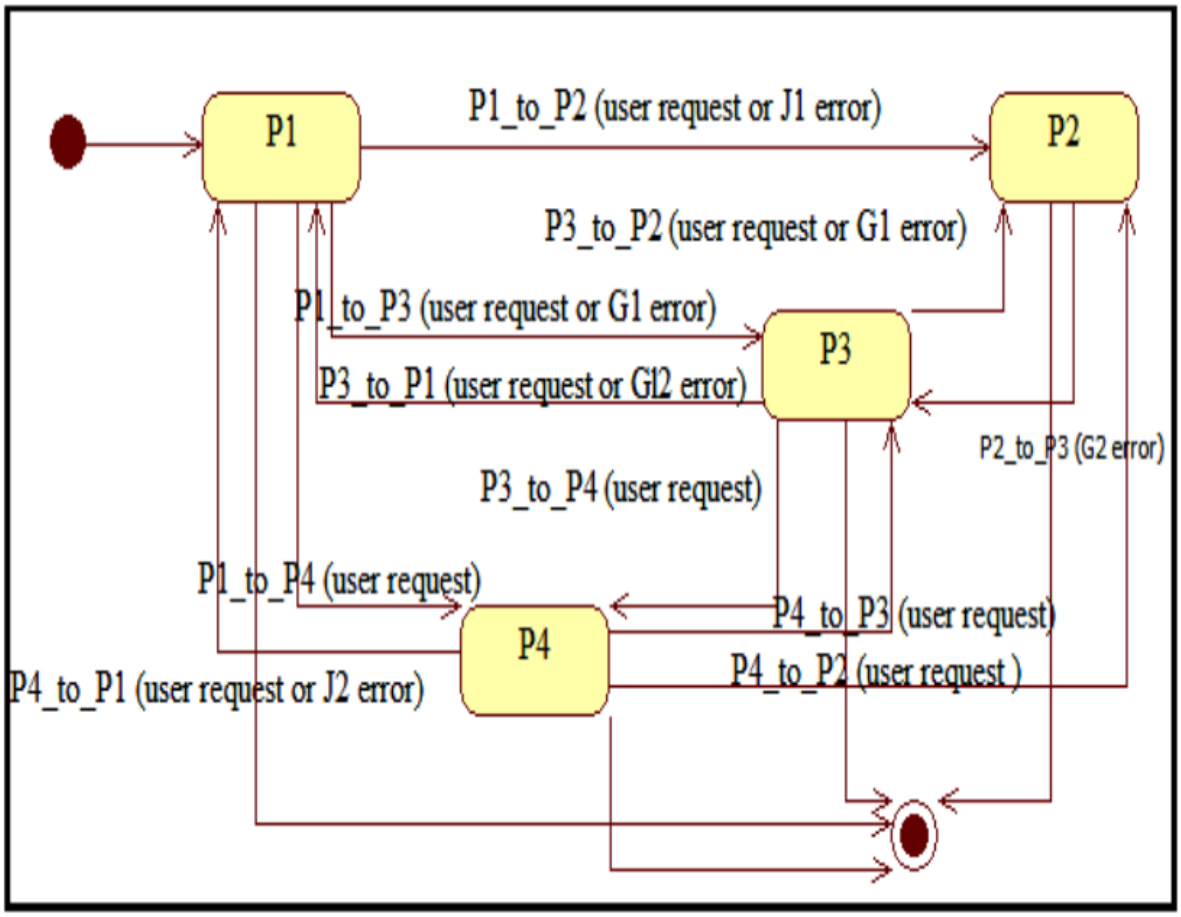

Figure D..14: State diagram of reconfiguration behaviors in EnAS [2]

Similarly, we convert this diagram to a job-on-arc precedence diagram as shown in Figure

D..15.

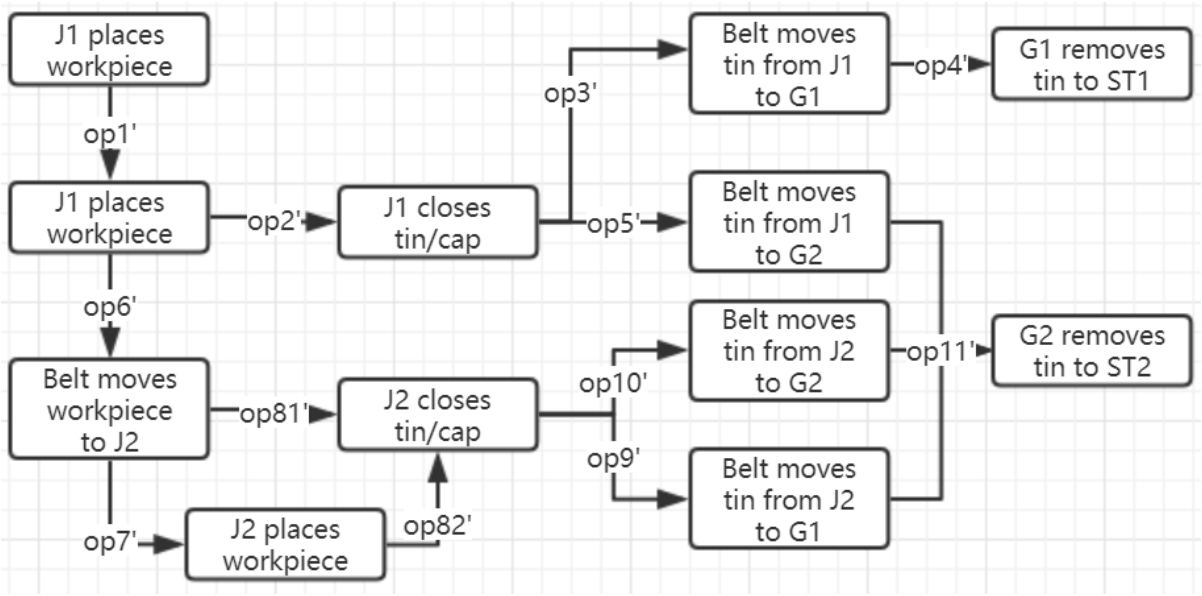

Figure D..15: Job-on-arc precedence graph of EnAS'

According to the job-on-arc precedence diagram of EnAS', the proposed **localized** multiple reconfiguration approach can be applied to the system. (The localized multiple reconfiguration approach can be applied to the FESTO' as well.) Figure D..16 shows the DES model of EnAS' system.

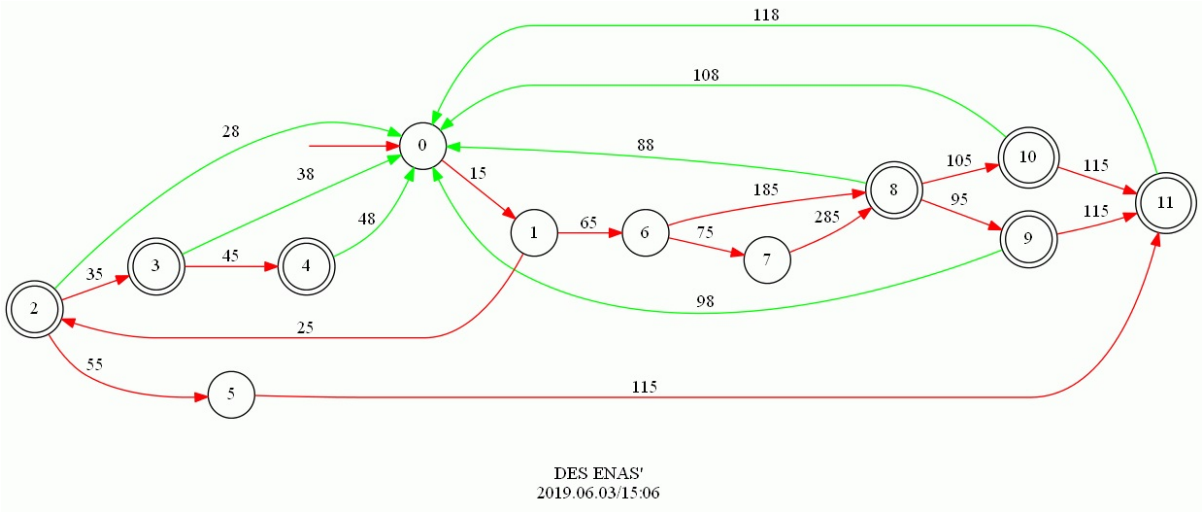

Figure D..16: DES model of EnAS'

The following table presents the correspondence of states and events in EnAS' and its

DES model.

Table D..2: Correspondence of states and events in EnAS' and its DES model

State in EnAS'	State in the DES model	Event in EnAS'	Event in the DES model
J1 places workpiece	1	op1'	15
J1 closes tin/cap	2	op2'	25
Belt moves tin from J1 to G1	3	op3'	35
G1 removes tin to ST1	4	op4'	45
Belt moves tin from J1 to G2	5	op5'	55
Belt moves workpiece to J2	6	op6'	65
J2 places workpiece	7	op7'	75
J2 closes tin/cap	8	op8'	185
Belt moves tin from J2 to G1	9	op9'	95
Belt moves tin from J2 to G2	10	op10'	105
G2 removes tin to ST2	11	op11'	115
Initialization	0	op8'	285

States 2, 3, 4, 8, 9, 10, 11 are the final states of EB1 to EB12. Thus, we mark them to show that the functional behaviors end. We also add five transitions labeled by uncontrollable events 28, 38, 48, 88, 98, 108 and 118 from these five states back to state 0 to reset the system.

For the four modes, say $\mathbf{Mode}_1$ is **Policy1**, $\mathbf{Mode}_2$ is **Policy2**, $\mathbf{Mode}_3$ is **Policy3** and $\mathbf{Mode}_4$ is **Policy4**. Then, $\Sigma_1 = \{15, 25, 28, 35, 38, 45, 48\}$, $\Sigma_2 = \{15, 65, 88, 105, 108, 115, 118, 185\}$, $\Sigma_3 = \{15, 25, 28, 55, 65, 88, 95, 98, 115, 118, 185\}$ and $\Sigma_4 = \{15, 65, 75, 88, 105, 108, 115, 118, 285\}$. According to the event alphabets and the DES model in Figure D..4, we can see that $Q^G_{PBS,1,2} = \{0, 1\}$, $Q^G_{PBS,1,3} = \{0, 1, 2\}$, $Q^G_{PBS,1,4} = \{0, 1\}$, $Q^G_{PBS,2,3} = \{0, 1, 6, 8, 11\}$, $Q^G_{PBS,2,4} = \{0, 1, 6, 8, 10, 11\}$ and $Q^G_{PBS,3,4} = \{0, 1, 6, 8, 11\}$. Thus, $\Sigma^G_{ETE,1,2} = \{25, 65\}$, $\Sigma^G_{ETE,1,3} = \{35, 55, 65\}$, $\Sigma^G_{ETE,1,4} = \{25, 65\}$, $\Sigma^G_{ETE,2,3} = \{25, 75, 95, 105\}$, $\Sigma^G_{ETE,2,4} = \{25, 75, 95\}$ and $\Sigma^G_{ETE,3,4} = \{25, 75, 95, 105\}$. Besides, $\Sigma^G_{EYE,1,2} = \{28, 38, 48, 88, 98, 108, 118\}$, $\Sigma^G_{EYE,1,3} = \{38, 48, 88, 98, 108, 118\}$, $\Sigma^G_{EYE,1,4} = \{28, 38, 48, 88, 98, 108, 118\}$, $\Sigma^G_{EYE,2,3} = \{28, 38, 48, 98, 108, 115, 285\}$, $\Sigma^G_{EYE,2,4} = \{28, 38, 48, 98, 115, 285\}$ and $\Sigma^G_{EYE,3,4} = \{28, 38, 48, 98, 108, 115, 285\}$.

According to Figure C..14, FESTO can only start in mode *Policy*1, so there is only one mode initialization event 17 for $\mathbf{Mode}_1$. Besides, there is no reconfiguration from mode **Light2** to mode **Policy1** or **Policy4**, so the reconfiguration events are $\sigma_{1,2} = 127$, $\sigma_{1,3} = 137$, $\sigma_{1,4} = 147$, $\sigma_{2,3} = 237$ $\sigma_{3,1} = 317$, $\sigma_{3,2} = 327$, $\sigma_{3,4} = 347$, $\sigma_{4,1} = 417$, $\sigma_{4,2} = 427$

and $\sigma_{4,3} = 437$. Then the core multiple reconfiguration specification **ENASRC** and the extra multiple reconfiguration specifications **ENASR12**, **ENASR13**, **ENASR14**, **ENASR23**, **ENASR24**, **ENASR34** are provided below.

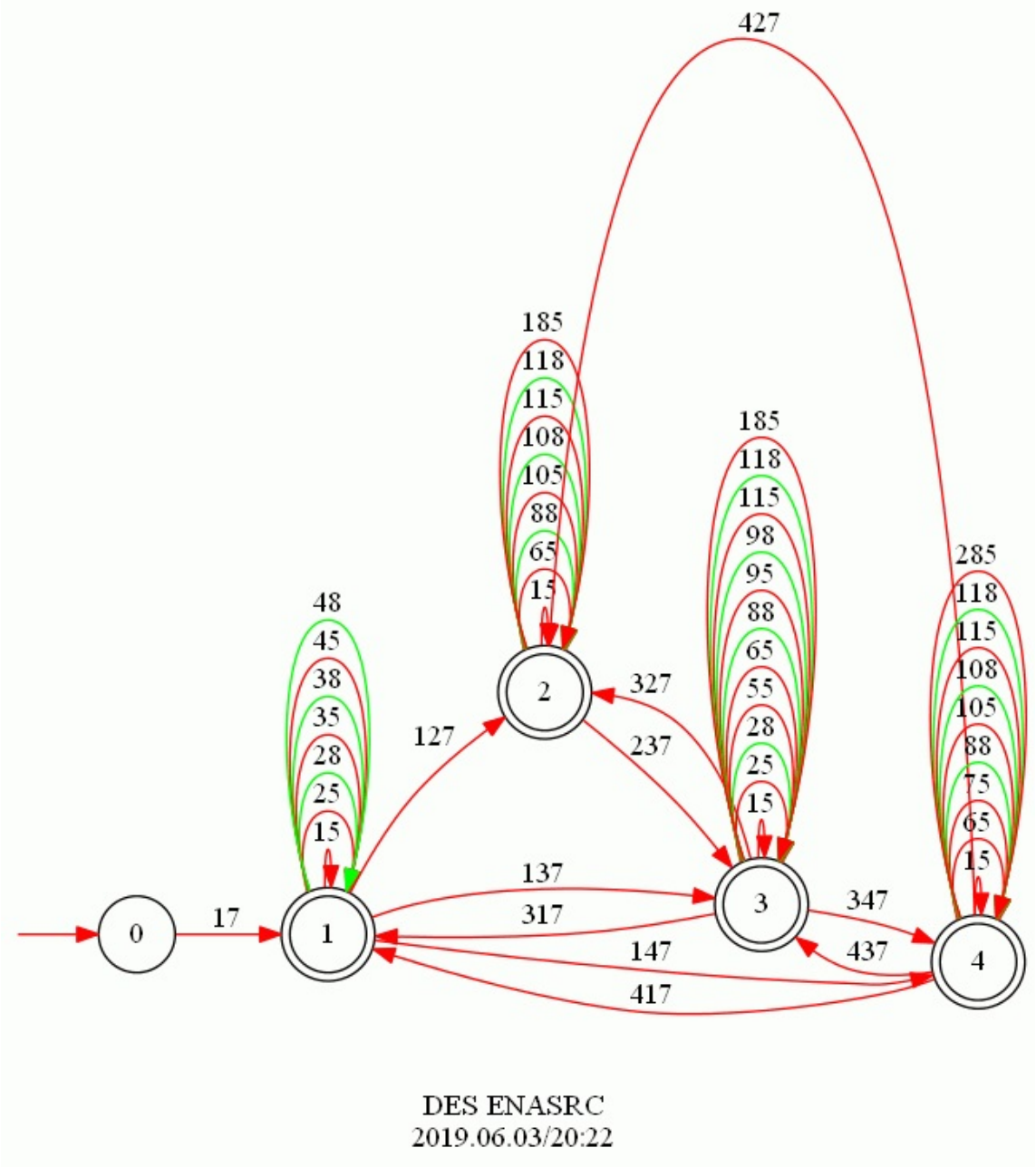

Figure D..17: LCMRS **ENASRC**

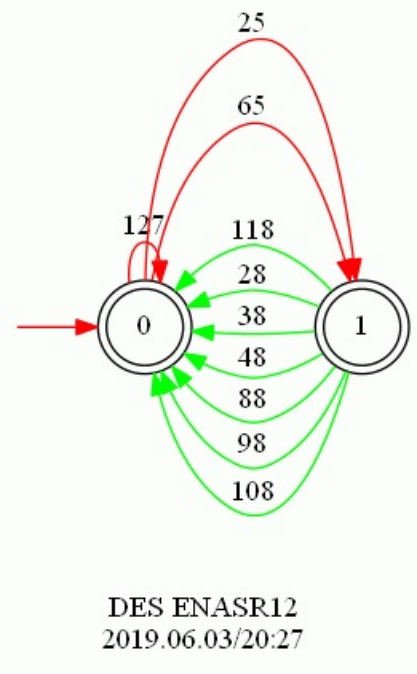

Figure D..18: LEMRS **ENASR12**

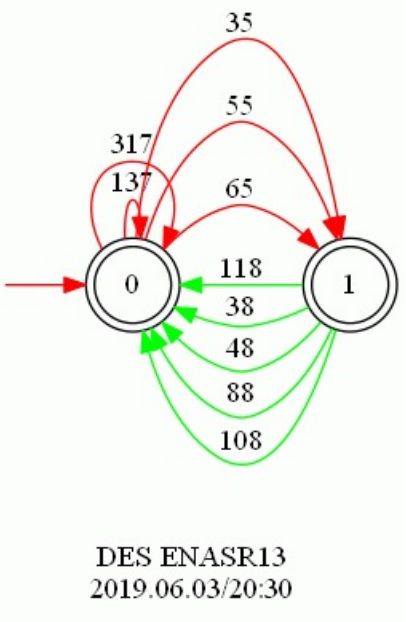

Figure D..19: LEMRS **ENASR13**

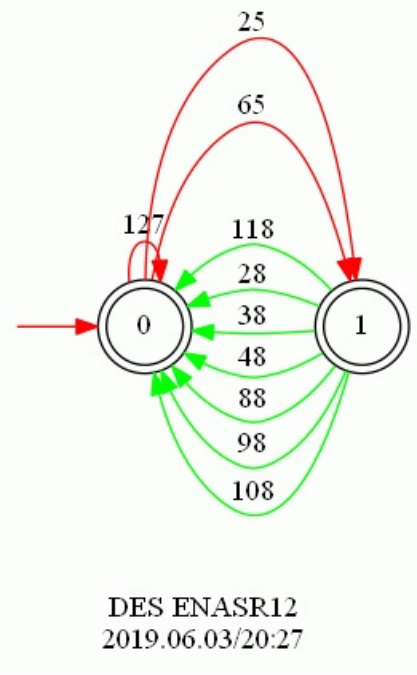

Figure D..20: LEMRS **ENASR14**

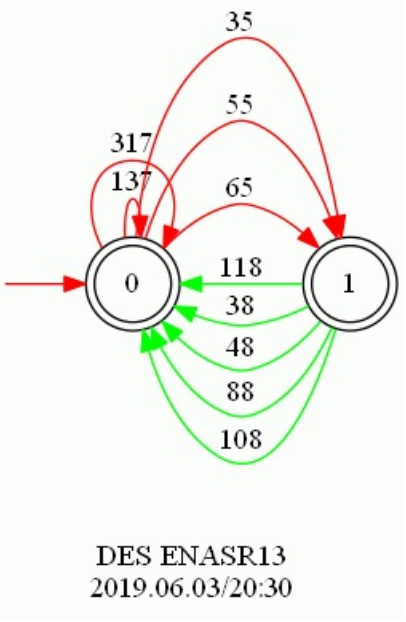

Figure D..21: LEMRS **ENASR23**

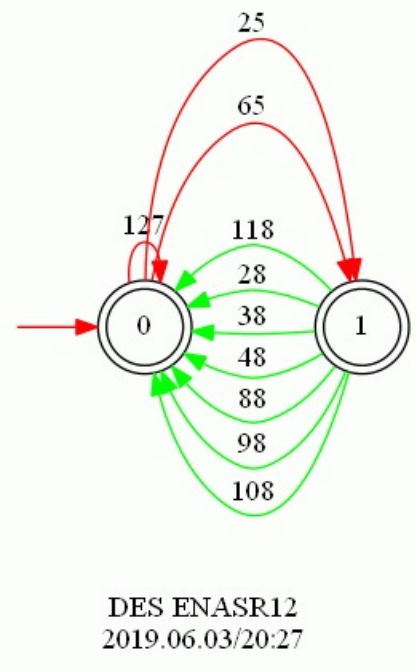

Figure D..22: LEMRS **ENASR24**

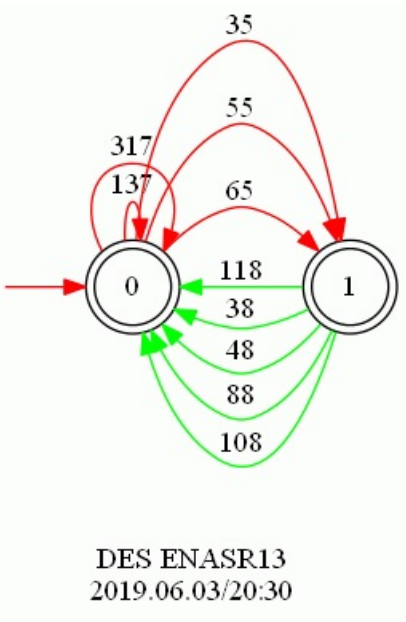

Figure D..23: LEMRS **ENASR34**

We then compute

ENASRG = **Sync**(**ENAS'**, **ENASRC**, **ENASR12**, **ENASR13**, **ENASR14**, **ENAS23**, **ENAS24**, **ENAS34**) (37,97) Blocked events = None

The resulting reconfiguration plant **ENASRG** is shown in Figure C..24. However, it is too large to be shown clearly. The reader interested in this aspect may obtain it in TCT.

Since ENAS can only start in $\mathbf{Mode}_1$, **ENASRG** can only start in $\mathbf{Mode}_1$ via the event 17. Each reconfiguration event occurs only when ENAS is at a public state with respect to the two modes corresponding to the reconfiguration event. Apart from the reconfiguration from $\mathbf{Mode}_2$ to $\mathbf{Mode}_1$ ($\mathbf{Mode}_4$), each pair of reconfiguration events can occur back-and-forth. Moreover, each mode is nonblocking separately in **ENASRG** and each state except for the initial state satisfies exactly one SMP. The reader interested in this aspect may check the five requirements by using the requirement checking program on the author's website (https://github.com/JasonZhangjc/). Note that the first and the third requirement are not met owing to the characteristics of ENAS.

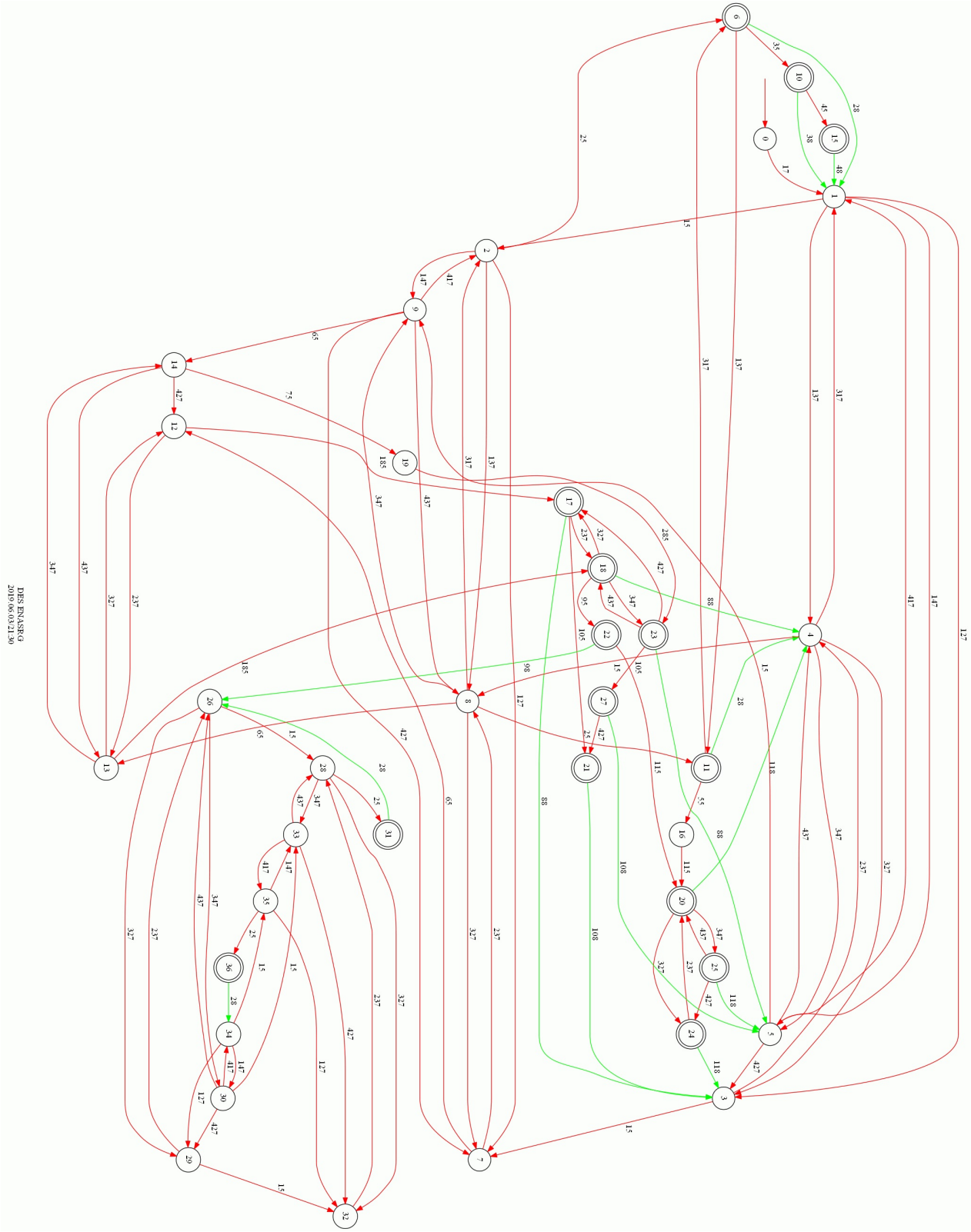

Figure D..24: Reconfiguration plant **ENASRG**

The next step is to add a behavioral specification. Recall that the two jack stations (J1 and J2) place newly drilled workpieces from FESTO and close tins with caps. Then consider a buffer **FESTOENASBUF** with two slots between FESTO and EnAS. Then when the buffer is empty, neither J1 nor J2 is allowed to pick workpieces from the buffer. Also, when the buffer is full, namely has two workpieces inside, FESTO is not permitted to send workpieces to the buffer. The operation of sending a workpiece from FESTO is represented by op12 (event 120 in the DES model of FESTO') and the operations of picking workpiece from FESTO are op1' and op7'. Thus, the buffer is represented by a behavioral specification and is depicted as follows.

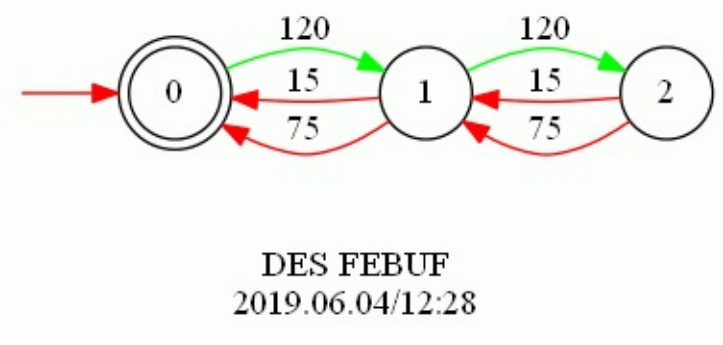

Figure D..25: Behavioral specification **FESTOENASBUF**

We then compute

$$\textbf{FERG} = \textbf{Sync}(\textbf{FESTORG}, \textbf{ENASRG})\ (1702{,}9642)\ \text{Blocked events} = \text{None}$$
$$\textbf{ALLFERG} = \textbf{Allevents}(\textbf{FERG.DES})\ (1{,}59)$$
$$\textbf{FEBUFSPEC} = \textbf{Sync}(\textbf{ALLFERG}, \textbf{FEBUF})\ (3{,}174)\ \text{Blocked events} = \text{None}$$
$$\textbf{FEBUFSUP} = \textbf{Supcon}(\textbf{FERG}, \textbf{FEBUFSPEC})\ (4852{,}27118)$$
$$\textbf{FEBUFD} = \textbf{Condat}(\textbf{FERG}, \textbf{FEBUFSUP})\ \text{Controllable.}$$
$$\textbf{FEBUFRESUP} = \textbf{Supreduce}(\textbf{FERG}, \textbf{FEBUFSUP}, \textbf{FEBUFD})\ (3{,}8;\text{slb}{=}3)$$

The supervisor **FEBUFSUP** is too large to be shown here, so we present the reduced supervisor **FEBUFRESUP** in Figure D..26.

This reduced supervisor is as we expected. In order to disable event 120, we have to disable event 121 since 120 is an uncontrollable event. This implicit information has been expressed by the reduced supervisor **FEBUFRESUP** as there is no solf-loop of event 121 at state 2 of **FEBUFRESUP**.

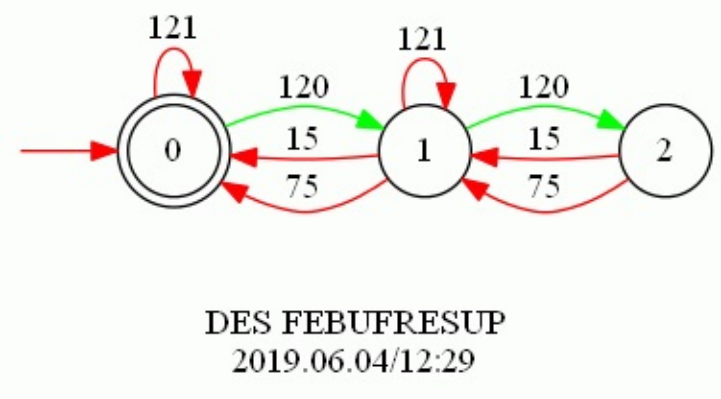

Figure D..26: Reduced supervisor **FEBUFRESUP**