HANS BHERER

Controller Synthesis for Parameterized Discrete Event Systems

Thèse présentée
à la Faculté des études supérieures de l'Université Laval
dans le cadre du programme de doctorat en informatique
pour l'obtention du grade de Philosophiae Doctor (Ph.D.)

FACULTÉ DES SCIENCES ET DE GÉNIE
UNIVERSITÉ LAVAL
QUÉBEC

2009

Résumé

Les systèmes à événements discrets sont des systèmes dynamiques particuliers. Ils changent d'état de façon discrète et le terme *événement* est utilisé afin de représenter l'occurrence de changements discontinus. Ces systèmes sont principalement construits par l'homme et on les retrouve surtout dans les secteurs manufacturier, de la circulation automobile, des bases de données et des protocoles de communication. Cette thèse s'intéresse au contrôle des systèmes paramétrés à événements discrets où les spécifications sont exprimées à l'aide de prédicats et satisfont une condition de similarité. Des conditions sont données afin de déduire des propriétés, en observation partielle ou totale, pour un système composé de n processus similaires à partir d'un système composé de n_0 processus, avec $n \geq n_0$. De plus, il est montré comment inférer des politiques de contrôle en présence de relations d'interconnexion entre les processus. Cette étude est principalement motivée par la faiblesse des méthodes actuelles de synthèse pour le traitement des problèmes industriels de taille réelle.

Abstract

Discrete event systems are a special type of dynamic systems. The *state* of these systems changes only at discrete instants of time and the term *event* is used to represent the occurrence of discontinuous changes. These systems are mostly man-made and arise in the domains of manufacturing systems, traffic systems, database management systems and communication protocols. This thesis investigates the control of parameterized discrete event systems when specifications are given in terms of predicates and satisfy a similarity assumption. For systems consisting of similar processes under total or partial observation, conditions are given to deduce properties of a system of n processes from properties of a system of n_0 processes, with $n \geq n_0$. Furthermore, it is shown how to infer a control policy for the former from the latter's, while taking into account interconnections between processes. This study is motivated by a weakness in current synthesis methods that do not scale well to huge systems.

Avant-propos

En premier lieu, je tiens à remercier mon directeur de recherche, le professeur Jules Desharnais, et mon codirecteur, le professeur Richard St-Denis, pour leur patience et leurs encouragements. Outre les aspects scientifiques, ils m'ont communiqué des valeurs d'intégrité et d'honnêteté intellectuelle. Ils ont su me faire apprécier la rigueur de même que l'importance d'une vision globale en science et particulièrement en informatique.

Je tiens à remercier mes parents qui ont toujours valorisé le savoir et l'éducation. Enfin, je remercie Ann qui m'a appris à vivre, à m'amuser et à faire l'équilibre entre le présent et le futur.

To Ann

Who controls the past controls the future.
Who controls the present controls the past.
George Orwell

Contents

List of Figures

Chapter 1

Introduction

1.1 Context

This thesis is about *system theory* and the supervisory control of discrete event systems (DESs). The supervisory control problem considered is to find (synthesize), if possible, a supervisor (controller) under whose control the system will behave according to a given specification. The situation under consideration is DESs modeled at the logical level by automata with specifications expressed in predicate logic.

It is well known that the state-space explosion problem constitutes a barrier to the modeling and control of DESs in the framework of the supervisory control theory (SCT). This renders automatic synthesis methods unworkable for many realistic applications, since the state space to be considered is so huge as to be intractable, even if ad hoc implementations of supervisors are relatively small in terms of lines of code. A potential solution consists in representing a system by a parameterized model, synthesizing a control policy with size independent of parameter values and determining properties about the closed-loop system behavior for arbitrary (sometimes bounded) parameter values. Control policies obtained in this way are, in essence, scalable.

While modular systems are, in general, heterogeneous, some have constituent elements with the same structure. Processes in such systems can be partitioned into classes defined using parameters. For instance, a parameter symbolizes the number of processes in a class or an internal dimension of a data structure (which is often represented by an automaton in the SCT framework). Adding parameters to a model entails adding corresponding parameters to the specifications. Addition of parameters that can be replaced by arbitrary natural numbers constitutes a major obstacle in synthesizing supervisors, because such systems may have infinitely many reachable states. Since there exists no algorithm for deciding any relevant property formulated in SCT (e.g., controllability) for recursively enumerable languages [Kumar and Garg 1995], several researchers assume that the languages involved in the computation of supervisors are regular. This is equivalent to computing a new supervisor for each instance of the parameters. This solution is not in the spirit of the method proposed in this thesis, because it is not scalable. When the languages are not regular, Petri nets are often used, but they must satisfy strict structural conditions so that procedures for verifying properties of interest can become decidable. For instance, *Elementary Composed State Machines*, which appear to be a restrictive class of Petri nets, can be used to model realistic systems and synthesize supervisors [Giua and DiCesare 1994]. Petri nets are not used in this thesis, but comparable restrictions must be made to obtain a class of parameterized discrete event systems (PDESs) for which control problems are decidable. A good starting point is to study the case in which all processes belong to a unique group with a single parameter, which represents the number of similar processes. To achieve a capability comparable to existing synthesis procedures for modular systems, families of similar processes should be combined. A reasonable approach would consist in dealing with them case by case, from the simplest (e.g., connection of a pair of replicated structures through a shared variable) to the hardest (e.g., split, merge, parallel connections between a multitude of replicated structures). The study of such compositions is beyond the scope of this thesis.

Throughout this thesis, soundness is referred to in many ways. The most general case (as in a *sound method*) refers to a method that is based on valid reasoning, free from logical flaws, and trustworthy. In the context of PDES, the soundness of a synthesis procedure (or method) refers to its ability to deduce valid properties (or SFBC functions) of a system of n processes from valid properties (or SFBC functions) of a system of n_0 processes, with $n \geq n_0$. In order to be more precise about the soundness of a given synthesis method, two new definitions (or concepts) are introduced. A synthesis method is said to be *strongly sound* if the supervisor calculated from the simplified model is behaviorally equivalent to the one corresponding to the concrete model (the system with n processes). It is said to be *weakly sound* if the system of n processes under control never violates the specification. Those notions of weak and strong soundness are mainly used in Chapter 7 to characterize the results of the synthesis method in the different contexts (partial observation and total observation). Finally, another type of soundness for section 6.2 could have been introduced because of a result (in total observation) that is stronger than strong soundness (not only is it behaviorally equivalent but as Theorem 6.6 shows, it is the same SFBC function) but was not. Nevertheless, the text before Theorem 6.6 explains the soundness for this particular result.

1.2 Objective

Essentially, the study of PDESs includes two main issues. The first consists in determining if properties, such as *controllability*, *observability* and *nonblockingness*, are preserved when the state space is expanded from dimension n_0 to dimension n whatever the value of $n \geq n_0$. The second issue concerns conditions to be satisfied in order to ensure that synthesis methods intended to deal with parameters are sound. A synthesis method is said to be *strongly sound* if the supervisor calculated from the simplified model is behaviorally equivalent to the one corresponding to the concrete model. It is said to be *weakly sound* if the system of n processes under control never violates the specification, but such a control may be unduly restrictive.

The case of partial observation raises some difficulties. On the one hand, even if the supremal controllable and normal subpredicate always exists [Li 1991], the *normality* property is generally too restrictive for real systems. On the other hand, the notion of *strong M–controllability* [Takai and Kodama 1997]—a strong version of *M–controllability* [Takai et al. 1995]—ensures the existence of a supremal element. Both notions depend on the concept of *bad event set*, which merges states that are observed in the same way, but, unlike the latter, the states are merged whether they satisfy the specification or not in the case of strong M–controllability. Notwithstanding these differences, all these notions hide some pitfalls that significantly impact the goal of achieving strong soundness. First, supremal elements are only expressed in their simplest form as an iterative computational schema, which rather limits the scope of theoretical results in modular control. Second, the notion of M–controllability includes a reachability property, similar to the one for the notion of controllability, which cannot be preserved [Bherer et al. 2006b]. Since problems for PDESs are, in general, undecidable, one of the ambitions with this parameterized approach is to develop synthesis methods that are sound, but necessarily incomplete, or to consider some restricted supervisory control problems that are decidable.

The main objective of this thesis is to consider a class of decidable control problems, namely that of the state feedback control (SFBC) of PDESs, consisting of similar processes, under total and under partial observation, and to develop a sound synthesis method that does not need any heuristic to synthesize supervisors.

1.3 Methodology

The methodology proposed in this thesis combines the modular control paradigm with an abstraction technique. First, it relies on three main concepts developed in the verification domain, but exploited here in the context of SCT: *reduction*, *parameterization* and *symmetry*. By analogy with the synthesis of concurrent programs with many similar processes [Attie and Emerson 1998], supervisor synthesis for a concrete system of n processes is reduced to the synthesis of a supervisor for a simplified system of n_0 processes, with $n_0 \leq n$. This is possible if both the system and specifications are parameterized and if symmetries emerge from their modeling. Second, based on some similarity assumptions, it considers the supervisor as a modular supervisor formed from m individual supervisors, each derived from an instance of the parameterized system and specifications, except that the synthesis of m individual supervisors is replaced by the off-line synthesis of only one small supervisor with m on-line syntactic renaming transformations, where $m \leq \binom{n}{n_0}$.

1.4 New Results

In general terms, this thesis presents a new method for the controller synthesis problem of a special class of PDESs. The main results establish the soundness of the method, both in total and partial observation. Most of these results have been recently published in [Bherer et al. 2008].

More precisely, the principal results of this thesis are the following.

1. Assumptions that capture the homogeneity of a PDES are given. These assumptions ensure the consistency of the results between the higher and the lower dimension state spaces. The *process similarity assumption* (PSA) formalizes the concept of replicated structure of [Attie and Emerson 1998]. The *mask similarity assumption* (MSA) ensures that the mask is the same for every system process (up to index substitution) and is introduced to deal with partial observation scenarios. These assumptions are presented in Chapter 4 and were first introduced in [Bherer et al. 2004] and [Bherer et al. 2005], respectively. The *specification similarity assumption* (SSA) imposes restrictions on the instances of a parameterized predicate representing the specification. SSA relates a state to all its projections. It is presented in Chapter 5 and was first published in [Bherer et al. 2004].

2. Chapter 5 presents two results regarding closure properties for the SSA. It is shown that SSA is closed under arbitrary conjunctions and disjunctions. While the latter is more of theoretical interest, the former presents an interest in modular control approaches where one may have to determine a centralized representation of decentralized supervisors. These properties were first published in [Bherer et al. 2008].

3. In order to provide broader results than [Bherer et al. 2004, Bherer et al. 2005], a predicate restriction is defined and formalized in Chapter 5. It permits the modeling of interconnection relations between processes. While a parameterized predicate captures constraints on the state of processes, an interconnection relation imposes additional constraints based on their identity. This interconnection relation first appeared in [Bherer et al. 2008] and is inspired by the interconnection relation of [Attie and Emerson 1998].

4. A method for the simplest case (total observation and no interconnection) is presented in Chapter 6. Is is based on [Bherer et al. 2004]. The main result, besides the on-line synthesis method, is the soundness of the method. It is shown that the method leads to an optimal solution by SFBC functions equality. This main result not only establishes that the method is strongly sound [Bherer et al. 2008] but also implies a kind of syntactic soundness between SFBC functions, which is a stronger result.

5. Chapter 7 addresses the case of partial observation and introduces an interconnection relation which was not present in [Bherer et al. 2005]. It also establishes that the method, based on strong M-controllability, is strongly sound (Section 7.5). This result improves the soundness result of [Bherer et al. 2005]. The soundness result, based on M-controllability, is also presented. In that latter case, the method is shown to be weakly sound. This result was first presented in [Bherer et al. 2006a] where the case of interconnection relations was not handled.

6. Section 7.5 considers the case of total observation, under interconnection relations, as a special case of partial observation. It is established that the method is strongly sound [Bherer et al. 2008].

7. Section 7.4 presents computational complexity results and Section 7.3 proposes an algorithm for on-line synthesis that brings a linear gain on complexity. This algorithm first appeared in [Bherer et al. 2008].

1.5 Structure of the Thesis

This thesis is structured as follows. Chapter 2 presents a survey of methods and techniques to tackle the state-space explosion problem that arises in basic control problems. Chapter 3 provides a concise review of concepts and results developed in the context of SFBC when DESs are totally or partially observed. Chapter 4 and Chapter 5 introduce the notation, basic definitions and properties required to consider subjects related to PDESs and parameterized specifications, respectively. Chapter 6 and Chapter 7 present the synthesis methods, as well as soundness results, for SFBC functions for the cases of total and partial observation, respectively. Finally, Chapter 8 situates this work from a more technical perspective and ends with a few concluding remarks.

Chapter 2

State of the art

Since the elaboration of SCT by Ramadge and Wonham, computational complexity has been a major concern resulting in a constant stream of research. A number of formal treatments have been proposed and solutions to this issue can be classified according to the following criteria: *control paradigm*, *semantic model*, *data structure*, *algorithmic technique*, *abstraction technique*, *problem reduction* and *integration of verification techniques*.

2.1 Control Paradigms

Control paradigms to lower the computational complexity are based on modularity, hierarchical structure and on-line control. These paradigms solve basic control problems for totally or partially observed DESs. Formulated in its most conventional form, a basic control problem consists in synthesizing a supervisor to restrain the uncontrolled behavior of a DES, represented by an automaton G, in order to achieve a given specification, represented by a language K.

The *modular control* paradigm is based on an horizontal decomposition of G and K. A specification K is written as an intersection of specifications, $K = K_1 \cap \cdots \cap K_m$, and the control policy is established from the conjunction of m supervisors, each synthesized from G and K_i to avoid the generation of a huge state space that stems from the calculation of an intersection [Ramadge and Wonham 1987, Wonham and Ramadge 1988]. The computational complexity can be reduced much more if a DES is modeled as a composition of asynchronous subsystems, $G = \|_{i=1}^{n} G_i$. A local specification K_i is applied to a subset of subsystems directly restricted by K_i and represented by a set of indices I_i. A supervisor is synthesized from each local specification K_i and $G^i = \|_{j \in I_i} G_j$ [Queiroz and Cury 2000]. This is particularly useful when a specification is not applied to the whole system and when synchronous subsystems share the same local specifications. Recently, other variants, in which languages are prefix-closed, have been proposed by considering indecomposable specifications and only one element corresponding to G_i for the computation of a local supervisor [Komenda and van Schuppen 2005, Komenda et al. 2005]. In general, the realization of a control policy in a modular fashion results in memory savings, but the supervisors may be blocking and checking this property is intrinsically a global problem [Cassandras and Lafortune 1999].

However, several approaches have been proposed to achieve better experimental and computational outcomes than the worst case. Some of these approaches have been presented in [Pena et al. 2006]. Another approach consists in combining supervisor reduction with modular and decentralized methodologies [Whittaker and Rudie 2008]. Supervisor state-space reduction involves the amalgamation of states that perform identical control or no control at all. The property of *comparability*, which can be verified in polynomial time and that encapsulates most of the concepts found in reduction-based approaches, was introduced and shown to be preserved under conjunction.

In the setting of the modular and centralized approach, a language property called *partial controllability condition*, which depends on the specification, has been introduced [Gaudin and Marchand 2007]. Based on this language property, control is performed on some approximations of the system. Under some new conditions, the behavior of the obtained controlled system corresponds to the supremal controllable language. One condition concerns the system itself and requires the shared events to be controllable, the other one relates to the specification and is called *local consistency*. The latter is strictly less restrictive than the separability condition. Unfortunately, the nonblocking aspect has not been studied under the partial controllability condition.

The *hierarchical control* paradigm is based on a vertical decomposition of systems and supervisors. They are exemplified by aggregate models, aggregate (bottom-up) multilevel hierarchies and structural (top-down) multilevel hierarchies. An *aggregate model* is obtained from a low-level model by refining the information sent up from the low-level model to the next one in order to ensure that the high-level supervisor can be implemented in the low-level model. This property is called *hierarchical consistency* [Zhong and Wonham 1990] and its fulfillment results in a hierarchy with tightly coupled levels. The primary purpose of this approach is the concrete expression of a report-command strategy by considering more abstract information at a given level; the higher the supervisor level, the fewer the computational resources used by synthesis algorithms. In the *aggregate multilevel hierarchy* approach, a master-slave or client-server mode is established through an interface that prescribes the interaction between the high-level and low-level models [Leduc et al. 2005]. Engineers must initially provide the interface and supervisors. Then, controllability and nonblockingness properties must be independently verified for each level. Therefore, engineers must repeatedly modify the models by hand, including the interface and supervisors, until they satisfy the properties. In this verification process, there is no global model. Recently, a synthesis procedure was designed to automatically derive locally maximally permissive supervisors from separate specifications [Song and Leduc 2006]. Computational savings are possible as long as the client and server have roughly the same size and the interface is relatively small compared with their size. Furthermore, verification is more appropriate for larger systems, because the verification procedure requires fewer resources than the synthesis procedure. In the *structural multilevel hierarchy* approach, DESs are modeled by using state tree structures (STSs), a kind of hierarchical state machine [Ma and Wonham 2005]. Connections between levels must satisfy boundary consistency and local coupling. Contrary to the previous approaches, only one nonblocking supervisor is synthesized for a given system modeled by an STS. Appropriate techniques that take advantage of this representation must be used to deal with larger systems. Generally speaking, hierarchical approaches are not sufficient in themselves to solve the state-space explosion problem because nothing ensures that the cost of verifying the underlying properties and synthesizing all the supervisors is less than that of deriving a global supervisor.

In the *on-line control* paradigm, the off-line synthesis of a complete control policy for all possible behaviors of the DES (which has exponential complexity in the number of its components) is replaced by a multitude of polynomial complexity calculations along the specific trace of events generated by the DES at run-time. Thus, the supervisor prescribes the next control action after each step of the closed-loop system based on an N–step forward projection of the DES behavior and a limited lookahead control policy [Chung et al. 1992]. The broader the available information about the DES the supervisor has, the lower the computational complexity. Several algorithms using this schema have been proposed with significant computational advantages [Heymann and Lin 1994, Ben Hadj-Alouane et al. 1994, Ben Hadj-Alouane et al. 1996]. This paradigm is, however, most relevant when DES behavior is modeled by recursive functions. In addition, the polynomial computational complexity is achieved to the detriment of a weaker validation procedure, since faults may be discovered at run-time due to the limited lookahead.

2.2 Semantic Models

Formal notations used to represent various aspects that are needed in the modeling of DESs are generally assessed with respect to their power of expressivity. Their semantics must be sufficiently rich to specify concurrency, synchronization, hierarchy, timing information, infinite behaviors, safety properties or liveness properties. For instance, automata can only express the order in which events occur in a system and Petri nets are particularly useful to describe interacting concurrent components. Both formalisms have been extended to satisfy other specific needs [Cassandras and Lafortune 1999]. In order to consider systems with huge state spaces, it is also important to have compact representations for preserving memory space in the computer and to take advantage of algebraic regularity of their internal structure to develop more efficient, more powerful synthesis algorithms that operate on them in comparison with those that work on an unstructured state set. Assorted Petri net models with various design approaches have been extensively exploited for these purposes in the context of SCT [Holloway et al. 1997]. To be efficient, however, these approaches must avoid the explicit construction of the reachability tree. This is particularly the case of *vector* DESs [Li and Wonham 1993] with linear predicates on the set of n-dimensional integer vectors as specifications. Based on a characterization of the reachable set from a given state by a system of linear inequalities, the calculation of an optimal policy is reduced to solving linear integer programming problems, one per pair consisting of a reachable state and a controllable event such that there exists at least one uncontrollable path beyond the transition defined by the pair. However, if strict structural conditions associated with the uncontrollable part of the system (e.g., mutual independence between some uncontrollable events and some conditions on the trees of the forest representing the uncontrollable part of the system) are satisfied, then the construction of an optimal policy is reduced to solving smaller linear integer programming problems in an appropriate form. This requires solving only one per tree of the forest in order to algorithmically express the control policy in a disjunction of linear inequalities, which can be evaluated for any reachable state at run-time [Li and Wonham 1994]. A more recent formalism, adapted from statecharts [Harel 1987], STSs [Ma and Wonham 2005], is especially effective when a DES, expressed in terms of coordinating components, has a high degree of concurrency, synchronization and hierarchy. An STS is composed of a state tree and *holons* that describe the local dynamics. Models are manipulated in a fashion which is logarithmically concise compared with the size of the underlying state spaces. This formalism impacts on the way supervisors are synthesized. The ultimate goal is exploring a set of objects significantly smaller than the overall state set.

2.3 Data Structures

In the areas of model checking and VLSI computer-aided verification, sizable progress has been achieved through an intensive use of BDDs, a data structure for compact representations of Boolean functions [Dreschsler and Sieling 2001]. Such representations do not eliminate the state-space explosion problem, but allow verification of larger systems. Their application in the SCT framework, particularly for the derivation of optimal supervisors that result from calculation of fixed points, is more modest. Fixed point procedures implemented with BDDs have been developed both in the SCT *language-based formulation* and SCT *state-based formulation*. In the former, the fixed point procedure is expressed in terms of Boolean functions describing the DES and specification automata [Balemi et al. 1993]. In the latter, it is formulated in terms of predicates characterizing the hierarchical state space, transition structures and forbidden state specifications [Ma and Wonham 2005]. Even though they may be of substantial help in many control problems, BDDs are not a panacea since the theoretical computational complexity remains beyond existing computational resources.

2.4 Algorithmic Techniques

A synthesis procedure can be implemented in many ways. Major improvements to conventional synthesis algorithms can be carried out by considering algorithmic techniques. One of them is *compositional synthesis*. It is based on compositional minimization, using concepts of process equivalence. The monolithic representation of the state-space is avoided by the use of simplified automata at the intermediate stages of the algorithm [Flordal et al. 2007]. Another one consists in postponing very expensive processing until the construction of the supervisor by performing some computations on the fly. For instance, instead of explicitly calculating the product transition structure of components and specifications from which the supervisor is extracted (*the extensional approach*), an efficient implementation expands such structures on the fly from the transition functions of components and specifications (*the intensional approach*) simultaneously with, and guided by, supervisor construction. Such a synthesis algorithm does not require the generation of any global behavioral model for the whole system or explicit storage of the entire workspace. This is efficient when the specifications severely constrain system behavior [Barbeau et al. 1997]. This technique is particularly useful when the system is modeled by an STS [Ma and Wonham 2005]. Since this kind of structure is more complex than an automaton, the intensional definition of the global transition function must be sound in the sense that it must be equivalent to that defined over a flat state set. Other algorithmic techniques are based on search mechanisms with heuristics and control-directed backtracking [Ben Hadj-Alouane et al. 1996, Barbeau et al. 1997]. Exploring implementation details is important, but complete comparison studies must be conducted [Kerjean et al. 2006].

2.5 Abstraction Techniques

Abstraction techniques lead to simplification because they discard irrelevant details for the problem at hand. They are especially relevant when both the DES and specifications exhibit symmetry. Instead of working with the automaton-based representations of the DES and specifications, a smaller supervisor can be derived from their quotient structures [Eyzell and Cury 2001] using techniques originally developed in model checking [Emerson and Sistla 1997]. Another possibility is to take advantage of colored Petri nets with symmetry specifications to solve a forbidden state avoidance problem [Makungu et al. 1999]. Colored Petri nets with a finite color set have the same expressive power as ordinary place/transition nets, but they offer a more compact representation of large systems consisting of many similar interacting components. The former approach is less restrictive than the latter, because it does not limit a specification to that of a specific forbidden state type. It requires, however, the use of a permutation index table that occupies an exponential space in the general case. Nevertheless, the computational complexity of the synthesis algorithm is reduced by a factor of N^2 when the DES consists of N similar components. Generally, this is clearly insufficient for conventional synthesis algorithms with an exponential growth rate in terms of N. Finally, the use of PDESs, as proposed in this thesis, constitutes an approach in which abstraction techniques are dominant.

2.6 Problem Reduction

One way to reduce the running time of synthesis procedures is to transpose SCT control problems into equivalent but easier problems in another theoretical framework. Under the assumption that $L(H) \subseteq L(G)$, where $L(H) = K$, and that all states of G and H are marked, the problem of computing the supremal controllable sublanguage of K with respect to $L(G)$ and Σ_u (the set of uncontrollable events) is equivalent to finding the greatest bisimulation relation between H and G with respect to Σ_u [Barrett and Lafortune 1998]. The computational complexity of the latter is smaller than the former. Exploiting this solution in synthesis procedures can be advantageous, particularly in the construction of on-line supervisors in which reachability and blockingness are not of interest.

2.7 Integration of Verification Techniques

The controller synthesis problem for PDESs has been the subject of some investigations, but the soundness (deduce properties of a system of n processes from properties of a system of n_0 processes, with $n \geq n_0$) still remains an issue [Frappier and St-Denis 2001, St-Denis 2002].

A sound synthesis method has been suggested for bounded-data parameterized discrete event systems (BDPDESs) under total observation [Bherer et al. 2003]. It integrates an automatic verification technique [Pnueli et al. 2001] into a synthesis method [Frappier and St-Denis 2001, St-Denis 2002]. The verification technique is based on a heuristic for an algorithmic construction of an inductive assertion, but it is incomplete because the algorithm may fail after two trials. The synthesis method is founded on attributed control (AC) and integrates abstraction techniques. While more general than the one described in [Bherer et al. 2008], it is incomplete since it requires human intervention.

The method described in [Frappier and St-Denis 2001, St-Denis 2002] is an iterative method where the behavior of the DES, composed of active components, is supervised by an attributed controller which encompasses passive components, denoted $\mathcal{O}$. The passive components are uncontrollable objects defined from algebraic specifications of abstract data types. The active components are grouped into classes whose cardinality is represented by a vector $\mathbf{M}$ of parameters. The parameters for the passive components are represented by a vector $\mathbf{N}$. In that context, the problem of synthesizing a controller $\mathcal{P}(\mathbf{M}, \mathbf{N})$ from a model $\mathcal{M}(\mathbf{M})$ of a DES and a specification $p(\mathbf{M}, \mathbf{N})$ can be formulated in the following way.

$$\begin{aligned}&\text{Given } \mathcal{M}(\mathbf{M}),\ \mathcal{O}(\mathbf{N}) \text{ and } p(\mathbf{M}, \mathbf{N}),\\ &\text{synthesize } \mathcal{P}(\mathbf{M}, \mathbf{N}) \text{ such that } (\mathcal{M}(\mathbf{M}) \parallel \mathcal{P}(\mathbf{M}, \mathbf{N})) \models p(\mathbf{M}, \mathbf{N}).\end{aligned} \tag{2.1}$$

In Equation 2.1, the term $\mathcal{M}(\mathbf{M}) \parallel \mathcal{P}(\mathbf{M}, \mathbf{N})$ represents the interaction between $\mathcal{M}(\mathbf{M})$ and $\mathcal{P}(\mathbf{M}, \mathbf{N})$, and $\models$ is the satisfaction relation between a model and a specification. It should be noted that the term $\mathcal{O}(\mathbf{N})$, which represents the passive components, is absent from (2.1) because the passive components are integrated into the solution represented by the term $\mathcal{P}(\mathbf{M}, \mathbf{N})$. The iterative method of [Frappier and St-Denis 2001] is composed of five steps: modeling, parameterization, reduction, synthesis and merging. Once the problem has been modeled and parameterized, only the last three steps of the method remain. They are presented in Figure 2.1.

$$\begin{tikzcd}
\mathcal{M}(\mathbf{M}),\ \mathcal{O}(\mathbf{N}) \text{ and } p(\mathbf{M},\mathbf{N}) \arrow[d, "\text{reduction}"] \\
(\mathcal{M}_\mathcal{A}, \mathcal{O}_\mathcal{A}, p_\mathcal{A}) \arrow[d, "\text{synthesis}"] \\
\mathcal{P}_I \text{ such that } (\mathcal{M}_\mathcal{A} \parallel \mathcal{P}_I) \models p_\mathcal{A} \arrow[d, "\text{merging}"] \\
\mathcal{P}_A \text{ such that } (\mathcal{M}_\mathcal{A} \parallel \mathcal{P}_A) \models p_\mathcal{A}
\end{tikzcd}$$

Figure 2.1: Attributed controller synthesis procedure

First, the synthesis of an attributed controller consists of assigning small values to all the parameters of Equation 2.1 in order to obtain an abstract model $(\mathcal{M}_\mathcal{A}, \mathcal{O}_\mathcal{A}, p_\mathcal{A})$ (it is the *reduction* step). Then, a standard controller synthesis algorithm is used to synthesize an intermediate controller $\mathcal{P}_I$ (it is the *synthesis* step). Since all parameters have small values, the state explosion problem is avoided. The intermediate controller is guarantied to be correct by a correctness proof of the general synthesis algorithm, hence $(\mathcal{M}_\mathcal{A} \parallel \mathcal{P}_I) \models p_\mathcal{A}$ holds. Finally, the intermediate controller is transformed into an attributed controller $\mathcal{P}_A$ (it is the *merging* step). If the transformation rules used in the merging step preserve the correctness, then $(\mathcal{M}_\mathcal{A} \parallel \mathcal{P}_A) \models p_\mathcal{A}$ holds. The attributed controller that takes into account the parameters and the different classes of active and passive components is noted $\mathcal{P}(\mathbf{M}, \mathbf{N})$. The transformation rules for merging in [Frappier and St-Denis 2001] were shown to preserve correctness. That is to say that the following holds:

$$(\mathcal{M}_\mathcal{A} \parallel \mathcal{P}_I) \models p_\mathcal{A} \Rightarrow (\mathcal{M}_\mathcal{A} \parallel \mathcal{P}_A) \models p_\mathcal{A}.$$

The problem of soundness is more difficult. Soundness is said to be preserved if the following holds:

$$(\mathcal{M}_\mathcal{A} \parallel \mathcal{P}_A) \models p_\mathcal{A} \Rightarrow (\mathcal{M}(\mathbf{M}) \parallel \mathcal{P}(\mathbf{M}, \mathbf{N})) \models p(\mathbf{M}, \mathbf{N}).$$

The method in [Frappier and St-Denis 2001, St-Denis 2002] falls short in term of soundness and [Bherer et al. 2003] proposes a partial solution to this problem by integrating a verification technique [Pnueli et al. 2001] into an attributed controller synthesis procedure for BDPDESs.

The verification method described in [Pnueli et al. 2001] applies to BDPDESs and properties of the form $\Box p$. A BDPDES is defined from a set of variables V, an initial condition $\Theta(V)$ and a transition relation $\rho(V, V')$. Some restrictions apply to V, Θ and ρ (see [Arons et al. 2001, Kesten et al. 2002, Pnueli et al. 2001] for details). The method consists in finding an assertion φ that satisfies premises I1 to I3 of the following deductive rule, called INV [Manna and Pnueli 1995]:

$$\begin{array}{ll} \text{I1.} & \Theta \rightarrow \varphi \\ \text{I2.} & \varphi \wedge \rho \rightarrow \varphi' \\ \text{I3.} & \varphi \rightarrow p \\ \hline & \Box p \end{array}$$

In order to automatically verify I1 to I3, a theoretical result concerning a particular form of assertions, called R-assertions, is used. If an R-assertion holds for all n such that $1 < n \leq N_0$, where N_0 is a constant that depends only on V and φ, then it holds for all $n > 1$. Moreover, when φ is an R-assertion, then conditions I1 to I3 are R-assertions. Finally, the verification method is based on a heuristic where two forms of φ are proposed and a value for N_0 is calculated for each φ. If the verification algorithm fails for both forms, no conclusion is reached. Hence, the method is incomplete. The following example, taken from [Pnueli et al. 2001], is also used in [Bherer et al. 2003] to illustrate the new synthesis method based on a verification technique.

Example 2.1 Consider the mutual-exclusion problem where N processes share a single resource. Figure 2.2 shows a transition diagram that represents the behavior of each process P_i $(1 \leq i \leq N)$. There are four states. I_i is the state where process i is not in a critical state and does not have the resource. State T_i is the state in which process i waits for the resource. C_i is the critical state and finally E_i is the state where process i has the resource. As shown, there are four events and only all_i is controllable (the small bar on an arrow indicates that a transition is controllable). Hence, $\mathcal{M}(\mathbf{M}) = \mathcal{M}(\langle N \rangle) = \|_{i=1}^{N} P_i$.

In the context of [Bherer et al. 2003], the problem of Example 2.1 is solved with an attributed controller that is synthesized for a small value of the parameter N (the reduction step). The idea is to then use N_0, from the verification algorithm, as this small value for N. Then, an attributed controller for $N = N_0$ can be synthesized, with the hope that the resulting closed-loop system is sound for all $N > 1$. In [Bherer et al. 2003], an attributed controller was synthesized for $N = 3$ since $N_0 = 3$ gave positive verification results in [Pnueli et al. 2001]. Figure 2.3 shows the feedback function for the intermediate controller. Only the legal states are represented along with the respective disabled events. Figure 2.4 shows the final attributed controller where a passive object

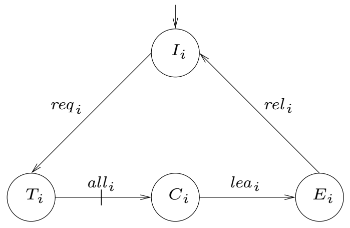

Figure 2.2: Behavior of a process using a resource

$(I_1\ I_2\ I_3\ \mathit{True})$	$\{\}$	$(C_1\ T_2\ I_3\ \mathit{False})$	$\{all_2\}$	$(C_1\ T_2\ T_3\ \mathit{False})$	$\{all_2\ all_3\}$
$(T_1\ I_2\ I_3\ \mathit{True})$	$\{\}$	$(C_1\ I_2\ T_3\ \mathit{False})$	$\{all_3\}$	$(T_1\ E_2\ I_3\ \mathit{False})$	$\{all_1\}$
$(I_1\ T_2\ I_3\ \mathit{True})$	$\{\}$	$(T_1\ C_2\ I_3\ \mathit{False})$	$\{all_1\}$	$(T_1\ C_2\ T_3\ \mathit{False})$	$\{all_1\ all_3\}$
$(I_1\ I_2\ T_3\ \mathit{True})$	$\{\}$	$(T_1\ T_2\ T_3\ \mathit{True})$	$\{\}$	$(T_1\ T_2\ C_3\ \mathit{False})$	$\{all_1\ all_2\}$
$(C_1\ I_2\ I_3\ \mathit{False})$	$\{\}$	$(T_1\ I_2\ C_3\ \mathit{False})$	$\{all_1\}$	$(T_1\ I_2\ E_3\ \mathit{False})$	$\{all_1\}$
$(T_1\ T_2\ I_3\ \mathit{True})$	$\{\}$	$(I_1\ E_2\ I_3\ \mathit{False})$	$\{\}$	$(I_1\ E_2\ T_3\ \mathit{False})$	$\{all_3\}$
$(T_1\ I_2\ T_3\ \mathit{True})$	$\{\}$	$(I_1\ C_2\ T_3\ \mathit{False})$	$\{all_3\}$	$(I_1\ T_2\ E_3\ \mathit{False})$	$\{all_2\}$
$(I_1\ C_2\ I_3\ \mathit{False})$	$\{\}$	$(I_1\ T_2\ C_3\ \mathit{False})$	$\{all_2\}$	$(E_1\ T_2\ T_3\ \mathit{False})$	$\{all_2\ all_3\}$
$(I_1\ T_2\ T_3\ \mathit{True})$	$\{\}$	$(I_1\ I_2\ E_3\ \mathit{False})$	$\{\}$	$(T_1\ E_2\ T_3\ \mathit{False})$	$\{all_1\ all_3\}$
$(I_1\ I_2\ C_3\ \mathit{False})$	$\{\}$	$(E_1\ T_2\ I_3\ \mathit{False})$	$\{all_2\}$	$(T_1\ T_2\ E_3\ \mathit{False})$	$\{all_1\ all_2\}$
$(E_1\ I_2\ I_3\ \mathit{False})$	$\{\}$	$(E_1\ I_2\ T_3\ \mathit{False})$	$\{all_3\}$		

Figure 2.3: Feedback function

(the fourth component of the state tuple) of type Boolean (see Figure 2.5) is used to represent the state of the resource.

In order to apply the verification technique described in [Pnueli et al. 2001], the retroaction loop $\mathcal{M}(\langle N\rangle) \parallel \mathcal{P}(\langle N\rangle)$ must be translated into the formalism used by the verification technique (the SPL language). Moreover, the attributed controller must be integrated into the SPL program. Figure 2.6 represents the final SPL program that was synthesized.

The program of Figure 2.6 is exactly the MUX-SEM program of [Pnueli et al. 2001]

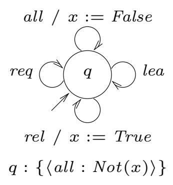

Figure 2.4: Attributed controller

$Boolean() :=$
  hidden sorts: $boolean$
operations:
  $False : \rightarrow boolean$
  $True : \rightarrow boolean$
  $Not : boolean \rightarrow boolean$
  $\ldots$
equations: $x \in boolean$
  $Not(True) = False \qquad Not(False) = True$
  $Not(Not(x)) = x$
  $\ldots$

Figure 2.5: *Boolean* abstract data type

$$\begin{array}{lll} \texttt{in} & N: & \texttt{natural where } N > 1 \\ \texttt{local} & x: & \texttt{boolean where } x = True \\ \multicolumn{2}{r}{\Big\|_{h=1}^{N} P[h] ::} & \left[\begin{array}{l} \texttt{loop forever do} \\ \left[\begin{array}{l} I : \texttt{Non-Critical} \\ T : \texttt{when } x = True \texttt{ do } x := False \\ C : \texttt{Critical} \\ E : x := True \end{array}\right] \end{array}\right] \end{array}$$

Figure 2.6: Closed-loop control SPL program

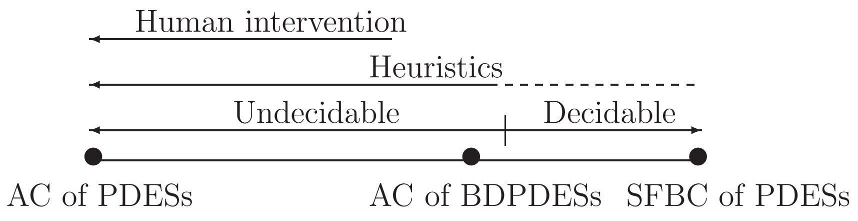

Figure 2.7: Classes of supervisory control problems for PDESs

that was positively verified for $N_0 = 3$. Contrary to [Pnueli et al. 2001] where the program was constructed empirically, it was systematically derived from a controller synthesis procedure. Hence, the approach proposed in [Bherer et al. 2003] constitutes a first step for the integration of a verification technique into a synthesis procedure. The verification procedure provides heuristics for determining the small values for the parameters in the reduction step. An attributed controller can then be systematically synthesized and its soundness can be proved by the same verification procedure. $\square$

In conclusion, none of these paradigms and techniques offers universal solutions, since they all have strengths and weaknesses compared with the others. Some of them may be particularly effective for a family of applications, while others may be inappropriate. Finally, Figure 2.7 summarizes what is considered in this thesis: a class of decidable control problems, namely that of SFBC of PDESs under total and under partial observation, for which the synthesis method is sound and does not need any heuristic to synthesize supervisors.

Chapter 3

Preliminaries on State Feedback Control

The concepts introduced in this chapter are part of the pioneering work originally developed by Ramadge and Wonham [Ramadge and Wonham 1987] and Li and Wonham [Li and Wonham 1988, Li 1991]. It was later extended by many others, including Kumar et al. [Kumar et al. 1993] and Takai, Ushio and Kodoma [Takai et al. 1995, Takai and Kodama 1997].

A DES is modeled by an automaton $G := (X, \Sigma, \delta, x_0, X_m)$, where X is a set of states; Σ is a finite set of events divided into two disjoint subsets Σ_c and Σ_u of controllable and uncontrollable events, respectively; $\delta : X \times \Sigma \to X$ is the partial transition function; x_0 is the initial state; and X_m is the subset of marked states, which represents the completed tasks. It is assumed that G is accessible; that is, all states are reachable from x_0 [Takai and Kodama 1997].

An SFBC function for G is a total function $f : X \to \Gamma$, where $\Gamma := \{\Sigma' \mid \Sigma_u \subseteq \Sigma' \subseteq \Sigma\}$. If $\sigma \in f(x)$, then σ is enabled at x; otherwise, it is disabled. An element of Γ is called a *control action*. For $\sigma \in \Sigma$, the predicate f_σ on X is defined by $f_\sigma(x) :\Leftrightarrow \sigma \in f(x)$ (where $:\Leftrightarrow$ means equivalent by definition). Thus, f may be described by a family of predicates $\{f_\sigma \mid \sigma \in \Sigma\}$.

Let $\delta(x, \sigma)!$ mean that $\delta(x, \sigma)$ is defined (for $s \in \Sigma^*$, $\delta(x, s)$ and $\delta(x, s)!$ are defined in the usual way and in particular $\delta(x, \epsilon)!$ always holds). The supervisor, represented by f, and the DES, represented by G, are embodied in a closed loop defined by $G^f := (X, \Sigma, \delta^f, x_0, X_m)$, where $\delta^f(x, \sigma) := \delta(x, \sigma)$ if $\delta(x, \sigma)!$ and $f_\sigma(x)$, and is undefined otherwise.

When the states of the DES are partially observed, X is partitioned into a set Y of equivalence classes, called observability classes. The membership map $M : X \to Y$, called the *mask*, is defined as a mapping from the state space X to the observation space Y. At the current state $x \in X$, the supervisor observes the value $M(x) \in Y$. Let F_o be the set of SFBC functions that satisfy the following assumption [Li 1991].

Assumption 3.1 *Restriction of an SFBC function f to the observability classes—For any $x, x' \in X$, $M(x) = M(x') \Rightarrow f(x) = f(x')$.*

An SFBC function $f \in F_o$ selects a control action $f(x)$ based on $M(x)$. The pair $(F_o, \leq)$ is a partially ordered set, with $f \leq g$ if $f(x) \subseteq g(x)$ for all $x \in X$. It is sometimes useful to denote the observability class of x by its representative element $x' \in X$ and simply write $M(x) = x'$.

3.1 Predicates and Predicate Transformers

Let $\mathrm{Pred}(X) := \{\mathsf{true}, \mathsf{false}\}^X$ be the set of all predicates on the state space X. A predicate $Q \in \mathrm{Pred}(X)$ generally represents the specification to be fulfilled. A partial order on $\mathrm{Pred}(X)$ is defined[1] as:

$$Q_1 \leq Q_2 \quad :\Leftrightarrow \quad (\forall x \mid x \in X : Q_1(x) \Rightarrow Q_2(x)).$$

The symbols true and false are overloaded to also denote the predicates that are true and false everywhere; that is, $\mathsf{true}(x) = \mathsf{true}$ and $\mathsf{false}(x) = \mathsf{false}$ for all x.

The predicate $Re(G|f) \in \mathrm{Pred}(X)$ holds exactly at the reachable states in G^f. It is defined inductively as:

1. $Re(G|f)(x_0)$ holds;
2. $Re(G|f)(x) \wedge \delta^f(x, \sigma)! \Rightarrow Re(G|f)(\delta(x, \sigma))$;
3. no other states satisfy $Re(G|f)$.

[1] Quantifications have the form ($\mathit{quantifier\ bound\ variable} \mid \mathit{range\ restriction} : \mathit{quantified\ expression}$) (see, *e.g.*, [Gries and Schneider 1995]); an empty range in a quantification means that the bound variable ranges over all possible values. $(\exists x \mid P : Q)$ is read as "there exists x such that P and Q". $(\forall x \mid P : Q)$ is read as "for all x such that P, Q holds" or as "for all x, P implies Q".

The predicate transformers M, $M^{-1}M$, wp_σ and wlp_σ (for a fixed $\sigma \in \Sigma$) on $\mathrm{Pred}(X)$ are defined as:

$$\begin{array}{rcl} M(Q)(y) & :\Leftrightarrow & (\exists x \mid x \in X : y = M(x) \wedge Q(x)); \\ M^{-1}(M(Q))(x) & :\Leftrightarrow & (\exists x' \mid x' \in X : M(x) = M(x') \wedge Q(x')); \\ \mathrm{wp}_\sigma(Q)(x) & :\Leftrightarrow & \delta(x,\sigma)! \wedge Q(\delta(x,\sigma)); \\ \mathrm{wlp}_\sigma(Q)(x) & :\Leftrightarrow & \neg\delta(x,\sigma)! \vee Q(\delta(x,\sigma)). \end{array}$$

Predicate transformers were introduced by E.W. Dijkstra for program specification and derivation in [Dijkstra 1975, Dijkstra 1976] and first applied to DES control theory in [Ramadge and Wonham 1987, Wonham and Ramadge 1988]. The predicate transformers M and $M^{-1}M$ are used in the context of partial observation while wp_σ and wlp_σ play a central role for controllability and reachability properties. In order to prevent the violation of a specification Q by disabling controllable events at a state x or a state observed as y, various definitions of bad event set have been introduced in the literature:

$$\begin{array}{rcl} A(Q,x) & := & \{\sigma \in \Sigma_c \mid \neg\mathrm{wlp}_\sigma(Q)(x)\}; \\ \hat{A}(Q,y) & := & \{\sigma \in \Sigma_c \mid (\exists x \mid x \in X : y = M(x) \wedge \neg\mathrm{wlp}_\sigma(Q)(x))\}; \\ \breve{A}(Q,y) & := & \{\sigma \in \Sigma_c \mid (\exists x \mid x \in X : y = M(x) \wedge Q(x) \wedge \neg\mathrm{wlp}_\sigma(Q)(x))\}. \end{array}$$

The set $\breve{A}(Q,y)$ is used in the case of partial observation and its definition imposes that $Q(x)$ holds if x is observed as y [Takai et al. 1995]. This condition is removed in the definition of $\hat{A}(Q,y)$ [Takai and Kodama 1997]. Finally, the set $A(Q,x)$ is used in the case of total observation, for which M is the identity function.

Reachability predicates can be defined from the above definitions of bad event set. For instance, $R(G,Q)$ is defined in the usual way. Let $Q \in \mathrm{Pred}(X)$. If $Q(x_0)$ does not hold, then $R(G,Q) := \mathsf{false}$; otherwise, $R(G,Q)$ is defined by induction as:

1. $R(G,Q)(x_0)$ holds;
2. $R(G,Q)(x) \wedge \sigma \notin A(Q,x) \wedge \mathrm{wp}_\sigma(Q)(x) \Rightarrow R(G,Q)(\delta(x,\sigma))$;
3. no other states satisfy $R(G,Q)$.

The reachability predicate $\hat{R}(G,Q)$ (resp. $\breve{R}(G,Q)$) is defined in the same manner, except that $A(Q,x)$ is replaced by $\hat{A}(Q,M(x))$ (resp. $\breve{A}(Q,M(x))$) in the inductive case.

Remark 3.2 If Q is Σ_u-invariant (see the definition in Section 3.2), then the inductive case (case 2) of the definition of $R(G, Q)$ can be replaced by

$$R(G,Q)(x) \wedge \sigma \notin A(Q,x) \wedge \delta(x,\sigma)! \Rightarrow R(G,Q)(\delta(x,\sigma))$$

because of the following property:

$$\begin{aligned} & R(G,Q)(x) \wedge \sigma \notin A(Q,x) \wedge \delta(x,\sigma)! \\ \Leftrightarrow\ & R(G,Q)(x) \wedge \sigma \notin A(Q,x) \wedge \mathrm{wp}_\sigma(Q)(x). \end{aligned} \tag{3.1}$$

The remark also holds for $\hat{R}(G, Q)$ and $\breve{R}(G, Q)$.

Finally, the predicate transformer $\langle\cdot\rangle : \mathrm{Pred}(X) \to \mathrm{Pred}(X)$ is defined by

$$\langle Q\rangle(x) :\Leftrightarrow (\forall s \mid s \in \Sigma_u^* : \neg\delta(x,s)! \vee Q(\delta(x,s))).$$

Intuitively, the predicate transformer $\langle\cdot\rangle$ characterizes the states from which no string of uncontrollable events can lead to an illegal state. The next proposition shows that $\langle\cdot\rangle$ is idempotent.

Proposition 3.3 *Let* $Q \in \mathrm{Pred}(X)$. *Then* $\langle\langle Q\rangle\rangle = \langle Q\rangle$.

Proof. From $\epsilon \in \Sigma_u^*$ and $\delta(x, \epsilon) = x$, it is immediate that $\langle\langle Q\rangle\rangle \le \langle Q\rangle$. Next, for $x \in X$, suppose that $\langle Q\rangle(x)$ holds but $\langle\langle Q\rangle\rangle(x)$ does not. Hence, there must exist $s \in \Sigma_u^*$ such that $\delta(x, s)!$ holds but $\langle Q\rangle(\delta(x, s))$ does not. This implies that there exists $t \in \Sigma_u^*$ such that $\delta(\delta(x, s), t)!$ and $Q(\delta(\delta(x, s), t))$ does not hold. So, $\delta(x, st)!$ and $\neg Q(\delta(x, st))$ both hold with $st \in \Sigma_u^*$, implying that $\langle Q\rangle(x)$ does not hold. This is a contradiction and completes the proof. □

3.2 Various Definitions of Controllability

Let $Q \in \mathrm{Pred}(X)$. The predicate Q is Σ_u-*invariant* with respect to G if $Q \le \mathrm{wlp}_\sigma(Q)$ for all $\sigma \in \Sigma_u$. It is *normal* if $M^{-1}(M(Q)) \le Q$. It is *controllable* with respect to G if Q is Σ_u-invariant with respect to G and satisfies a reachability condition that depends on the underlying context:

$$(\forall\sigma \mid \sigma \in \Sigma_u : Q \le \mathrm{wlp}_\sigma(Q)) \wedge \begin{cases} Q \le R(G,Q) \text{ if controllability;} \\ Q \le \breve{R}(G,Q) \text{ if M-controllability;} \\ Q \le \hat{R}(G,Q) \text{ if strong M-controllability.} \end{cases}$$

Intuitively, Q is controllable if, for any x that satisfies Q, x is reachable from x_0 via a sequence of states satisfying Q and Q is invariant under a sequence of uncontrollable events. The following theorem states that a nontrivial predicate Q is controllable when it can be inferred from an SFBC f.

Theorem 3.4 *Let $Q \in \mathrm{Pred}(X)$, $Q \neq \mathsf{false}$. Then Q is controllable if and only if there exists an SFBC function $f \in F_o$ such that $Re(G|f) = Q$.*

This theorem is valid whatever the reachability condition considered and its proofs, depending on the reachability condition, can be found in [Wonham 2008] for controllability and [Takai et al. 1995] for the M-controllability. For the case of strong M-controllability the proof can be deduced from [Takai and Kodama 1997, Takai and Kodama 1998]. Moreover, these proofs give a way to construct f. For each $\sigma \in \Sigma$:

$$f_\sigma(x) :\Leftrightarrow \begin{cases} \sigma \notin A(Q,x) & \text{if } Q \text{ is controllable;} \\ \sigma \notin \breve{A}(Q,M(x)) & \text{if } Q \text{ is M-controllable;} \\ \sigma \notin \hat{A}(Q,M(x)) & \text{if } Q \text{ is strongly M-controllable.} \end{cases}$$

The condition $\sigma \notin A(Q,x)$ is equivalent to $\sigma \in \Sigma_c \Rightarrow \mathrm{wlp}_\sigma(Q)(x)$.

Theorem 3.4 raises the natural question of what kind of control can be exercised when Q fails to be controllable. Following the conventional procedure, define the following families of predicates:

$$\begin{aligned} \mathcal{CP}(Q) &:= \{Q' \in \mathrm{Pred}(X) \mid Q' \leq Q \text{ and } Q' \text{ is controllable}\}; \\ \mathcal{C}(Q) &:= \{Q' \in \mathrm{Pred}(X) \mid Q' \leq Q \text{ and } Q' \text{ is M-controllable}\}; \\ \mathcal{SC}(Q) &:= \{Q' \in \mathrm{Pred}(X) \mid Q' \leq Q \text{ and } Q' \text{ is strongly M-controllable}\}; \\ \mathcal{CN}(Q) &:= \{Q' \in \mathrm{Pred}(X) \mid Q' \leq Q \text{ and } Q' \text{ is controllable and normal}\}. \end{aligned}$$

The supremal element $\sup \mathcal{CP}(Q)$ exists in $\mathcal{CP}(Q)$ and is equal to $R(G, \langle Q \rangle)$. The supremal elements $\sup \mathcal{SC}(Q)$ and $\sup \mathcal{CN}(Q)$ exist, but they are obtained from an iterative computational procedure rather than being given by a compact expression as for $\sup \mathcal{CP}(Q)$ [Takai and Kodama 1997, Li 1991]. The supremal element $\sup \mathcal{C}(Q)$ does not always exist, because, contrary to $\hat{A}$, $\breve{A}$ fails to be antimonotone with respect to its first argument. Finally, $\mathcal{CN}(Q) \subseteq \mathcal{SC}(Q) \subseteq \mathcal{C}(Q) \subseteq \mathcal{CP}(Q)$, where the first inclusion is valid under a certain condition on the mask [Takai and Kodama 1997].

3.3 State Feedback Supervisors

The Σ_u-invariance property plays a key role in the derivation of SFBC functions, particularly when reachability is not a concern. If Q fails to be Σ_u-invariant, the predicate $\sup \mathcal{CI}(Q)$ is then targeted, where $\mathcal{CI}(Q)$ is the set of all Σ_u-invariant predicates stronger than Q. Let the function $H : \text{Pred}(X) \to \text{Pred}(X)$ be defined by [Ramadge and Wonham 1987]

$$H(T) := Q \wedge \bigwedge_{\sigma \in \Sigma_u} \text{wlp}_\sigma(T).$$

Then, $\sup \mathcal{CI}(Q)$ is the greatest fixed point of H, which is equal to $\langle Q \rangle$ as shown by the following proposition.

Proposition 3.5 $\nu H = \langle Q \rangle$.

Proof. By a standard result of lattice theory [Davey and Priestley 1990], it is sufficient to show (i) $\langle Q \rangle \leq H(\langle Q \rangle)$ and (ii) for any $U \in \text{Pred}(X)$, $U \leq H(U)$ implies $U \leq \langle Q \rangle$.

(i) Let $x \in X$ and suppose that $\langle Q \rangle(x)$ holds. Then $Q(x)$ must hold. By Proposition 3.3, $\langle \cdot \rangle$ is idempotent. Also, $\Sigma_u \subseteq \Sigma_u^*$. Thus:

$$\begin{aligned}
\mathsf{true} &\Leftrightarrow \langle Q \rangle(x) \Leftrightarrow \langle\langle Q \rangle\rangle(x) \Leftrightarrow (\forall s \mid s \in \Sigma_u^* : \neg\delta(x,s)! \vee \langle Q \rangle(\delta(x,s))) \\
&\Rightarrow (\forall \sigma \mid \sigma \in \Sigma_u : \neg\delta(x,\sigma)! \vee \langle Q \rangle(\delta(x,\sigma))) \\
&\Leftrightarrow (\forall \sigma \mid \sigma \in \Sigma_u : \text{wlp}_\sigma(\langle Q \rangle)(x)) \Leftrightarrow \left(\textstyle\bigwedge_{\sigma \in \Sigma_u} \text{wlp}_\sigma(\langle Q \rangle)\right)(x).
\end{aligned}$$

This shows that

$$\langle Q \rangle \leq \left(Q \wedge \bigwedge_{\sigma \in \Sigma_u} \text{wlp}_\sigma(\langle Q \rangle) \right) = H(\langle Q \rangle).$$

(ii) Suppose $U \leq H(U)$. The goal is to show that $U \leq \langle Q \rangle$. So, assume $U(x)$. Let us show that $\langle Q \rangle(x)$ holds by proving that if $\delta(x,s)!$, then $Q(\delta(x,s))$, for any $s \in \Sigma_u^*$. Because $U \leq H(U) \leq Q$, it is sufficient to prove that if $\delta(x,s)!$, then $U(\delta(x,s))$, for any $s \in \Sigma_u^*$. The proof is by induction on the length of s.

- Base case, $s = \epsilon$: This is direct by $\delta(x,\epsilon)!$ and $U(x) \Leftrightarrow U(\delta(x,\epsilon))$.
- Induction step: Let $s = t\sigma$, for some $t \in \Sigma_u^*$ and $\sigma \in \Sigma_u$. Assume $\delta(x,s)!$. Then, $\delta(x,t)!$, so that, by the induction hypothesis, $U(\delta(x,t))$. Because $U \leq H(U) \leq \text{wlp}_\sigma(U)$, then $U(\delta(\delta(x,t),\sigma))$; that is, $U(\delta(x,s))$. □

Remark 3.6 Based on this result, the Σ_u-invariance property for a given predicate Q, which is defined above as $Q \leq \mathrm{wlp}_\sigma(Q)$ for all $\sigma \in \Sigma_u$, is equivalent to $Q \leq \langle Q \rangle$. Both conditions are used in this thesis.

Proposition 3.7 *Let $Q \in \mathrm{Pred}(X)$ be such that Q is Σ_u-invariant and $Q(x_0)$ holds, and let f be the SFBC function that corresponds to Q.*

1. *If $\delta^f(x,\sigma)! \Leftrightarrow \sigma \notin A(Q,x) \wedge \delta(x,\sigma)!$ for all $x \in X$ and $\sigma \in \Sigma$, then $Re(G|f) = R(G,Q)$.*
2. *If $\delta^f(x,\sigma)! \Rightarrow \sigma \notin A(Q,x)$ for all $x \in X$ and $\sigma \in \Sigma$, then $Re(G|f) \leq R(G,Q)$.*

The same properties hold if A and R are replaced by $\breve{A}$ (with $M(x)$ instead of x) and $\breve{R}$, respectively, or by $\hat{A}$ (with $M(x)$ instead of x) and $\hat{R}$, respectively.

Proof.

1. When $Q(x_0)$ holds, there is only one difference in the formal structure of the definition of $Re(G|f)$ and that of $R(G,Q)$: the antecedent of the implication in the inductive case (case 2) of the definitions. Because Q is Σ_u-invariant, (3.1) holds, and thus the antecedent in the definition of $R(G,Q)$ is equivalent to $R(G,Q)(x) \wedge \sigma \notin A(Q,x) \wedge \delta(x,\sigma)!$. Thus the definitions of $Re(G|f)$ and $R(G,Q)$ have the same structure when $\delta^f(x,\sigma)! \Leftrightarrow \sigma \notin A(Q,x) \wedge \delta(x,\sigma)!$.
2. The argument is similar, using the hypothesis and $\delta^f(x,\sigma)! \Rightarrow \delta(x,\sigma)!$.

The proof is the same for $\breve{A}$, $\breve{R}$ and for $\hat{A}$, $\hat{R}$. □

Let the SFBC functions f^*, $\breve{f}$ and $\hat{f}$ be defined as follows for all $\sigma \in \Sigma_c$ and $x \in X$:

$$f^*_\sigma(x) \;:\Leftrightarrow\; \sigma \notin A(\langle Q \rangle, x); \tag{3.2}$$
$$\breve{f}_\sigma(x) \;:\Leftrightarrow\; \sigma \notin \breve{A}(\langle Q \rangle, M(x)); \tag{3.3}$$
$$\hat{f}_\sigma(x) \;:\Leftrightarrow\; \sigma \notin \hat{A}(\langle Q \rangle, M(x)). \tag{3.4}$$

Let Q be such that $\langle Q \rangle(x_0)$ holds. In the case of total observation, f^* is optimal (in the sense that it is the behaviorally least restrictive SFBC function) and $Re(G|f^*) = R(G, \langle Q \rangle)$ (by Proposition 3.7(1)). This SFBC function is slightly different from the one given in [Wonham 2008], but it should be noted that $f^*_\sigma(x)$ may be evaluated arbitrarily when $\delta(x,\sigma)$ is undefined. In the case of partial observation, the SFBC $\breve{f}$ is such that $\sup \mathcal{SC}(Q) \leq \breve{R}(G, \langle Q \rangle) = Re(G|\breve{f})$. Thus,

$$\hat{f}^* \leq \breve{f},$$

where $\hat{f}^*$ is the optimal SFBC function that corresponds to the supremal element $\sup \mathcal{SC}(Q)$ [Takai and Kodama 1998].

The following proposition gives a means to compute $\breve{f}$ or $\hat{f}$ from f^*.

Proposition 3.8

$$\breve{f}(x) \quad = \quad \left(\bigcap x' \mid M(x) = M(x') \wedge \langle Q \rangle(x') : f^*(x')\right); \tag{3.5}$$

$$\hat{f}(x) \quad = \quad \left(\bigcap x' \mid M(x) = M(x') : f^*(x')\right). \tag{3.6}$$

Proof.

$$\begin{aligned}
&\sigma \notin \breve{f}(x)\\
\Leftrightarrow\ &\sigma \in \breve{A}(\langle Q \rangle, M(x))\\
\Leftrightarrow\ &\sigma \in \Sigma_c \wedge (\exists x' \mid x' \in X : M(x) = M(x') \wedge \langle Q \rangle(x') \wedge \neg \mathrm{wlp}_\sigma(\langle Q \rangle)(x'))\\
\Leftrightarrow\ &(\exists x' \mid x' \in X : M(x) = M(x') \wedge \langle Q \rangle(x') \wedge \sigma \in A(\langle Q \rangle, x'))\\
\Leftrightarrow\ &(\exists x' \mid x' \in X : M(x) = M(x') \wedge \langle Q \rangle(x') \wedge \sigma \notin f^*(x'))\\
\Leftrightarrow\ &\sigma \in \left(\bigcup x' \mid x' \in X \wedge M(x) = M(x') \wedge \langle Q \rangle(x') : \overline{f}^*(x')\right),
\end{aligned}$$

where $\overline{f}^*(x') := \Sigma - f^*(x')$.

The other result is proved in a similar manner. □

The reasons behind the selection of these SFBC functions are based on the following observations. Recently, the notion of *weak controllability* has been introduced and defined by dropping the reachability condition $Q \leq R(G, Q)$ in the definition of controllability [Ma and Wonham 2005]. This condition is computationally expensive and unnecessary for the synthesis of an SFBC function. The main argument is that, if Q is weakly controllable, then $R(G, Q)$ is controllable. Unfortunately, this result cannot be extended to the case of partial observation when $\hat{R}$ is used instead of R [Bherer et al. 2006a] and its impact is discussed in Section 7.5. Nevertheless, if Q is weakly controllable, then

$\breve{R}(G, Q)$ is M-controllable [Takai and Kodama 1998]. It follows that $\breve{R}(G, \langle Q \rangle)$ is a better approximation for Q than $\sup \mathcal{SC}(Q)$. Furthermore, $\breve{f}$ as defined by (3.3) is maximal in the sense that there is no f such that $Re(G|f) = \breve{R}(G, \langle Q \rangle)$ and $\breve{f} < f$ [Takai et al. 1995].

Chapter 4

Parameterized Discrete Event Systems

Many modular systems have constituent elements with the same structure and processes in such systems can be partitioned into classes defined using parameters. For instance, a parameter can represent the number of processes in a class. This thesis considers the case where all processes belong to a unique class and hence have a single parameter: the number of processes.

Let us consider a PDES G^N, where N is a parameter that denotes the number of processes, defined from the finite composition of a replicated structure

$$P_i \quad := \quad (X_i, \Sigma_s \cup \Sigma_i, \delta_i),$$

where X_i is a finite set of states indexed by i; Σ_s is a finite set of non-indexed, controllable events; Σ_i is a finite set of events indexed by i and partitioned into two subsets $\Sigma_{c,i}$ and $\Sigma_{u,i}$ of controllable and uncontrollable events, respectively; and $\delta_i : X_i \times (\Sigma_s \cup \Sigma_i) \rightarrow X_i$ is the partial transition function. The replicated structure represents the behavior of similar processes. The parameter N can be replaced by any number $n \in \mathbb{N}$. The events that belong to Σ_s are shared by all processes and allow for synchronization.

The concept of replicated structure is translated into a process similarity assumption [Attie and Emerson 1998]. Formally, let $\theta := \{j/i\}$ be a substitution such that $\theta i = j$ $(1 \leq i, j \leq N)$.

Assumption 4.1 *Process Similarity Assumption (PSA)*—$(\forall i, j \mid 1 \leq i, j \leq N : P_j = \theta P_i)$, *where*

$$\begin{array}{rcl} \theta P_i & := & (\theta X_i, \Sigma_s \cup \theta\Sigma_{c,i} \cup \theta\Sigma_{u,i}, \theta\delta_i); \\ \theta X_i & := & X_{\theta i} := \{x_{\theta i} \mid x_i \in X_i\}; \\ \theta\Sigma_{c,i} & := & \Sigma_{c,\theta i} := \{\sigma_{\theta i} \mid \sigma_i \in \Sigma_{c,i}\}; \\ \theta\Sigma_{u,i} & := & \Sigma_{u,\theta i} := \{\sigma_{\theta i} \mid \sigma_i \in \Sigma_{u,i}\}; \\ \theta\delta_i(x_i, \sigma) & := & \delta_{\theta i}(x_{\theta i}, \sigma) \text{ if } \sigma \in \Sigma_s; \\ \theta\delta_i(x_i, \sigma_i) & := & \delta_{\theta i}(x_{\theta i}, \sigma_{\theta i}) \text{ if } \sigma_i \in \Sigma_i. \end{array}$$

Therefore, a process can be derived from any other process by index substitution. A global state $x \in X^N$ is represented by a tuple of N local states. Let $x[i]$ denote the i-th component of x. The transition structure G^N is defined from a synchronous composition for events in Σ_s and an interleaving composition for events in each Σ_i. Thus, $G^N := (X^N, \Sigma^N, \delta^N)$, where $\Sigma^N = \Sigma_s \cup \Sigma_1 \cup \cdots \cup \Sigma_N$ and $(\delta^N(x, \sigma))[i] = \delta_i(x[i], \sigma)$ if $\sigma \in \Sigma_s \cup \Sigma_i$ and $(\delta^N(x, \sigma))[i] = x[i]$ otherwise. An instance of a PDES G^N is denoted by (G^n, x_0^n), where $x_0^n \in X^n$ is the initial state.

To illustrate the previous definitions, let us consider the running example of N users under control trying to acquire a single resource while satisfying various constraints based on their identity.

Example 4.2 Figure 4.1a shows a transition diagram that represents the behavior of user i $(1 \leq i \leq N)$. It includes three states: I_i (Idle), R_i (Requesting) and U_i (Using). For instance, the user can move from state I_i to state R_i on event α_i (request the resource), then from state R_i to state U_i on event β_i (allocate the resource) and, finally, from state U_i to state I_i on event γ_i (release the resource). There are two additional controllable transitions, labeled r, to reset all users in the initial configuration in which all users are idle, one from state R_i to state I_i and a self-loop on state I_i. Events α_i and β_i are controllable. In Figure 4.1, the small bar on an arrow indicates that a transition is controllable. □

In order to deal with PDESs, many definitions need to be introduced. In fact, most of these definitions introduce some functions (or operators) related to projections (from the state space of dimension n to the state space of dimension n_0) and the renaming (substitution) of indices. The basic thing one needs to understand is the meaning of $x = \langle U_1, I_2, R_3, U_4, R_5 \rangle$, for example. Here, it means that x is a state in the state space of dimension 5 and that process 1 is in state U, process 2 is in state I, and so on.

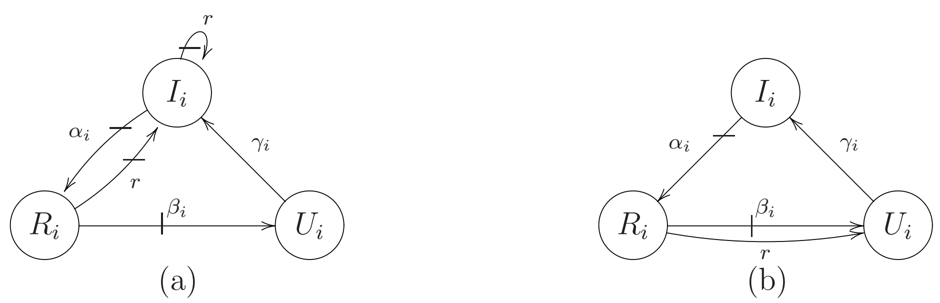

Figure 4.1: Replicated structures for the users

The reason why the states of the processes are indexed is the need to keep track of process identities when projections are applied to global states of dimension n. For example, the projection of processes 2, 3 and 5 from x to a state x', of dimension 3, results in $x' = \langle I_2, R_3, R_5 \rangle$. But here, there is a problem with the indices because in dimension 3, the index 5, for example, has no meaning. And in fact, the projection transforms process 5 into process 3, process 3 into process 2 and process 2 into process 1. Therefore, there is a need for an index substitution operator to rename the indices after the projection, hence the state x' now becomes $x' = \langle I_1, R_2, R_3 \rangle$, thus making a coherent state of dimension 3 that is the result of a projection and a renaming, where process 1 of x' is in fact process 2 of x (and similarly for the other two processes that were projected).

Definition 4.3 Let $x := \langle x[1], x[2], \ldots, x[n] \rangle \in X^n$. Then

$$M^n(x) \quad := \quad \langle M_1(x[1]), M_2(x[2]), \ldots, M_n(x[n]) \rangle,$$

where $M_i : X_i \rightarrow Y_i$ is the mask for process i.

The next definitions introduce the projection and substitution operators on global states, events, sets of events and strings of events. They are useful to establish relationships between a system consisting of n processes and a system consisting of n_0 processes, where $n_0 \leq n$. Moreover, it should be noted that these operators have lower precedence than function application. Hence, for an operator Θ and a function f, one should read $\Theta f(x)$ as $\Theta(f(x))$.

Definition 4.4 Let $n_0, n \in \mathbb{N}$, where $1 \leq n_0 \leq n$. Let $\mathcal{J}^n_{n_0}$ be the set of subsets of indices defined by $\mathcal{J}^n_{n_0} := \{J \mid J \subseteq \{i \mid 1 \leq i \leq n\} \wedge |J| = n_0\}$.

In the sequel, the expression "Let $J \in \mathcal{J}^n_{n_0}$" means "Let $J = \{j_1, \ldots, j_{n_0}\}$ and $1 \leq j_1 < \cdots < j_{n_0} \leq n$".

Definition 4.5 Let $J \in \mathcal{J}^n_{n_0}$. The projection operator $\uparrow_J$ on a global state $x \in X^n$ is a function $\uparrow_J : X^n \rightarrow X_{j_1} \times \cdots \times X_{j_{n_0}}$ that is defined as:

$$\uparrow_J x := \langle x[j_1], \ldots, x[j_{n_0}] \rangle.$$

The next definition introduces the substitution operator that ensures that the states of processes in the state space of dimension n_0 are indexed from 1 to n_0.

Definition 4.6 Let $J \in \mathcal{J}^n_{n_0}$. The substitution operator θ_J on a global state $x \in X_{j_1} \times \cdots \times X_{j_{n_0}}$ is a function $\theta_J : X_{j_1} \times \cdots \times X_{j_{n_0}} \rightarrow X^{n_0}$ that expresses the simultaneous replacement of process indices $j_1, \ldots, j_{n_0}$ by process indices $1, \ldots, n_0$, respectively. It is defined as:

$$\theta_J x \quad := \quad \langle \{1/j_1\}(x[1]), \ldots, \{n_0/j_{n_0}\}(x[n_0]) \rangle.$$

The next two definitions define the projection and substitution operators on events, based on a set of indices J, while Definition 4.9 and Definition 4.10 generalize in a natural way those operators to sets and strings of events, respectively.

Definition 4.7 Let $J \in \mathcal{J}^n_{n_0}$. The projection operator $\uparrow_J$ on an event $\sigma \in \Sigma^n$ is a function $\uparrow_J : \Sigma^n \rightarrow \Sigma_s \cup \Sigma_{j_1} \cup \cdots \cup \Sigma_{j_{n_0}} \cup \{\epsilon\}$ that is defined as: $\uparrow_J \sigma := \sigma$ if $\sigma \in \Sigma_s$ or $\sigma \in \Sigma_i$ and $i \in J$; and $\uparrow_J \sigma := \epsilon$ if $\sigma \in \Sigma_i$ and $i \notin J$.

Definition 4.8 Let $J \in \mathcal{J}^n_{n_0}$. The substitution operator θ_J on an event $\sigma \in \Sigma_s \cup \Sigma_{j_1} \cup \cdots \cup \Sigma_{j_{n_0}} \cup \{\epsilon\}$ is a function $\theta_J : \Sigma_s \cup \Sigma_{j_1} \cup \cdots \cup \Sigma_{j_{n_0}} \cup \{\epsilon\} \rightarrow \Sigma^{n_0} \cup \{\epsilon\}$ that is defined as: $\theta_J \sigma := \sigma$ if $\sigma \in \Sigma_s$; $\theta_J \sigma := \{k/j_k\}\sigma$ if $\sigma \in \Sigma_{j_k}$ and $j_k \in J$; and $\theta_J \epsilon := \epsilon$.

Definition 4.9 Let $\Omega \subseteq \Sigma_s \cup \Sigma_{j_1} \cup \cdots \cup \Sigma_{j_{n_0}} \cup \{\epsilon\}$ and $J \in \mathcal{J}^n_{n_0}$. The operator θ_J on a set of events is a function $\theta_J : 2^{\Sigma_s \cup \Sigma_{j_1} \cup \cdots \cup \Sigma_{j_{n_0}} \cup \{\epsilon\}} \rightarrow 2^{\Sigma^{n_0} \cup \{\epsilon\}}$ that is defined as: $\theta_J \Omega := \{\theta_J \sigma \mid \sigma \in \Omega\}$.

Let $\Theta_J := \theta_J \circ \uparrow_J$. If $x \in X^n$, $\Theta_J x$ is well defined and $\Theta_J : X^n \rightarrow X^{n_0}$. Furthermore, if $\sigma \in \Sigma^n$, $\Theta_J \sigma$ is well defined and $\Theta_J : \Sigma^n \rightarrow \Sigma^{n_0} \cup \{\epsilon\}$.

Definition 4.10 Let $J \in \mathcal{J}^n_{n_0}$. The operator Θ_J on a string of events is a function $\Theta_J : (\Sigma^n)^* \rightarrow (\Sigma^{n_0})^*$ that is recursively defined as: $\Theta_J \epsilon := \epsilon$ and $\Theta_J s\sigma := (\Theta_J s)(\Theta_J \sigma)$, where $\sigma \in \Sigma^n$ and $s \in (\Sigma^n)^*$.

The next example illustrates the effect of the previously introduced operators. In that example, the higher state space is of dimension 5 and the lower one is of dimension 3.

Example 4.11 Let $n_0 = 3$, $n = 5$ and consider the system introduced in Example 4.2. Let $x = \langle U_1, I_2, R_3, U_4, R_5\rangle$ and $s = \alpha_2\gamma_4\gamma_1 r\alpha_3$. If $J = \{2, 3, 4\}$, then $\Theta_J x = \langle I_1, R_2, U_3\rangle$ and $\Theta_J s = \alpha_1\gamma_3 r\alpha_2$. □

Example 4.11, with $J = \{2, 3, 4\}$, illustrates the projection of processes 2, 3 and 4 to the space of dimension 3. Hence, process 2 becomes process 1, process 3 becomes process 2 and process 4 becomes process 3. So $\uparrow_J x = \langle I_2, R_3, U_4\rangle$ and $\theta_J(\uparrow_J x) = \langle I_1, R_2, U_3\rangle$. Then, the string of events $s = \alpha_2\gamma_4\gamma_1 r\alpha_3$ becomes $\uparrow_J s = \alpha_2\gamma_4 r\alpha_3$. Finally, the renaming of indices of each event produces $\theta_J(\uparrow_J s) = \alpha_1\gamma_3 r\alpha_2$.

Remark 4.12 Let $s \in (\Sigma^{n_0})^*$, $J \in \mathcal{J}^n_{n_0}$ and $\theta_J = \{1/j_1, \dots, n_0/j_{n_0}\}$. Then $\theta_J^{-1}s$ exists, since $\theta_J^{-1} = \{j_1/1, \dots, j_{n_0}/n_0\}$. Also, $\Theta_J(\theta_J^{-1}s) = \theta_J(\theta_J^{-1}s) = s$ and $s = \theta_J t \Leftrightarrow t = \theta_J^{-1}s$. It should be noted that an element of $(\Sigma^{n_0})^*$ is also an element of $(\Sigma^n)^*$.

Remark 4.13 Let $x \in X^{n_0}$ and $J \in \mathcal{J}^n_{n_0}$. Then $\Theta_J(\theta_J^{-1}x) = \theta_J(\theta_J^{-1}x) = x$ and $x = \theta_J y \Leftrightarrow y = \theta_J^{-1}x$. The last equivalence also holds if $k/j_k \in \theta_J$, $x \in X_k$ and $y \in X_{j_k}$.

Remark 4.14 Let $x \in X^n$, $J \in \mathcal{J}^n_{n_0}$ and $s \in (\Sigma^n - \Sigma_s)^*$. Then $\delta^n(x, s)! \Rightarrow \delta^n(x, \uparrow_J s)!$. This is easy to see by noting that a transition with event σ_i does not affect the definedness of transitions with event σ_j if $i \neq j$, because no synchronization occurs.

Besides PSA as a condition on the processes, a system under partial observation must satisfy another similarity assumption imposed on the mask. Intuitively, it ensures that the mask is the same for every system process up to index substitution.

Assumption 4.15 *Mask Similarity Assumption (MSA)*—$(\forall i \mid 1 \leq i \leq N : \theta M_i(x_i) = M_{\theta i}(x_{\theta i}))$.

Several relationships may be established between a system composed of n processes and a system of n_0 processes under the assumptions PSA and MSA. Some of them are presented here. The following lemmas show that each diagram in Figure 4.2 commutes and give necessary and sufficient conditions for $\delta^n(x, s)$ to be defined with respect to equivalent information in the state space of dimension n_0.

$$\begin{tikzcd}
X^n \arrow[r, "\Theta_J"'] \arrow[d, "M^n"'] & X^{n_0} \arrow[d, "M^{n_0}"] \\
Y^n \arrow[r, "\Theta_J"'] & Y^{n_0}
\end{tikzcd}
\qquad
\begin{tikzcd}
X^n \times (\Sigma^n)^* \arrow[r, "\Theta_J"'] \arrow[d, "\delta^n"'] & X^{n_0} \times (\Sigma^{n_0})^* \arrow[d, "\delta^{n_0}"] \\
X^n \arrow[r, "\Theta_J"'] & X^{n_0}
\end{tikzcd}$$

Figure 4.2: Commutative diagrams

Lemma 4.16 *Let $x \in X^n$ and $J \in \mathcal{J}^n_{n_0}$. Then $M^{n_0}(\Theta_J x) = \Theta_J M^n(x)$.*

Proof.

$M^{n_0}(\Theta_J x)$

$=$ $\langle$ Typing of Θ_J $\rangle$

$M^{n_0}(\langle(\Theta_J x)[1], \ldots, (\Theta_J x)[n_0]\rangle)$

$=$ $\langle$ Definition 4.3 $\rangle$

$\langle M_1((\Theta_J x)[1]), \ldots, M_{n_0}((\Theta_J x)[n_0])\rangle$

$=$ $\langle$ Definitions 4.5 and 4.6 $\rangle$

$\langle M_1(\{1/j_1\}(x[j_1])), \ldots, M_{n_0}(\{n_0/j_{n_0}\}(x[j_{n_0}]))\rangle$

$=$ $\langle$ MSA (Assumption 4.15) $\rangle$

$\langle \{1/j_1\}M_{j_1}(x[j_1]), \ldots, \{n_0/j_{n_0}\}M_{j_{n_0}}(x[j_{n_0}])\rangle$

$=$ $\langle$ Definition 4.6 $\rangle$

$\theta_J\langle M_{j_1}(x[j_1]), \ldots, M_{j_{n_0}}(x[j_{n_0}])\rangle$

$=$ $\langle$ Definition 4.3 $\rangle$

$\theta_J\langle (M^n(x))[j_1], \ldots, (M^n(x))[j_{n_0}]\rangle$

$=$ $\langle$ Definition 4.5 and definition of Θ_J (Page 31) $\rangle$

$\Theta_J M^n(x)$ □

Lemma 4.17 *Let $x \in X^n$, $\sigma \in \Sigma^n$ and $J \in \mathcal{J}^n_{n_0}$. If $\delta^n(x, \sigma)!$, then*

$$\delta^{n_0}(\Theta_J x, \Theta_J \sigma) \quad = \quad \Theta_J \delta^n(x, \sigma).$$

If $\sigma \in \Sigma_i$ with $i \in J$, then $\delta^{n_0}(\Theta_J x, \Theta_J \sigma)! \Leftrightarrow \delta^n(x, \sigma)!$.

PROOF. There are three cases to consider.

1. First case: σ is an indexed event, say $\sigma_i \in \Sigma_i$ and $i \notin J$.

$$\begin{aligned}
& \delta^{n_0}(\Theta_J x, \Theta_J \sigma_i) \\
= \quad & \quad \langle \text{ Definitions 4.7 and 4.8 } \rangle \\
& \delta^{n_0}(\Theta_J x, \epsilon) \\
= \quad & \quad \langle\ \delta(x,\epsilon) = x\ \rangle \\
& \Theta_J x \\
= \quad & \quad \langle\ i \notin J \text{ and hence, for } j \in J,\ (\delta^n(x,\sigma_i))[j] = x[j] \ \ \&\ \ \delta^n(x,\sigma)!\ \rangle \\
& \Theta_J \delta^n(x, \sigma_i)
\end{aligned}$$

2. Second case: σ is an indexed event, say $\sigma_{j_k} \in \Sigma_{j_k}$ and $j_k \in J$.

$$\begin{aligned}
& \delta^{n_0}(\Theta_J x, \Theta_J \sigma_{j_k}) \\
= \quad & \quad \langle \text{ Typing of } \Theta_J \ \ \& \ \text{ Definitions 4.7 and 4.8 } \rangle \\
& \delta^{n_0}(\langle(\Theta_J x)[1], \ldots, (\Theta_J x)[n_0]\rangle, \sigma_k) \\
= \quad & \quad \langle \text{ Definition of } \delta^{n_0} \text{ (Page 29) } \rangle \\
& \langle(\Theta_J x)[1], \ldots, \delta_k((\Theta_J x)[k], \sigma_k), \ldots, (\Theta_J x)[n_0]\rangle \\
= \quad & \quad \langle \text{ Definitions 4.5 and 4.6 } \rangle \\
& \langle\{1/j_1\}(x[j_1]), \ldots, \delta_k(\{k/j_k\}(x[j_k]), \sigma_k), \ldots, \{n_0/j_{n_0}\}(x[j_{n_0}])\rangle \\
= \quad & \quad \langle \text{ PSA (Assumption 4.1) } \rangle \\
& \langle\{1/j_1\}(x[j_1]), \ldots, \{k/j_k\}\delta_{j_k}(x[j_k], \sigma_{j_k}), \ldots, \{n_0/j_{n_0}\}(x[j_{n_0}])\rangle \\
= \quad & \quad \langle \text{ Definition 4.6 } \rangle \\
& \theta_J\langle x[j_1], \ldots, \delta_{j_k}(x[j_k], \sigma_{j_k}), \ldots, x[j_{n_0}]\rangle \\
= \quad & \quad \langle \text{ Definition of } \delta^n \text{ (Page 29) } \rangle \\
& \theta_J\langle(\delta^n(x,\sigma_{j_k}))[j_1], \ldots, (\delta^n(x,\sigma_{j_k}))[j_k], \ldots, (\delta^n(x,\sigma_{j_k}))[j_{n_0}]\rangle \\
= \quad & \quad \langle \text{ Definition 4.5 and definition of } \Theta_J \text{ (Page 31) } \rangle \\
& \Theta_J \delta^n(x, \sigma_{j_k})
\end{aligned}$$

Since the hypothesis $\delta^n(x,\sigma)!$ is not used in the proof, each term of the equality is defined precisely when the other is. Because the operator Θ_J is total, this means that $\delta^{n_0}(\Theta_J x, \Theta_J \sigma)! \Leftrightarrow \delta^n(x,\sigma)!$. This also implies that if $\delta^n(x,\sigma)!$, then the equality holds.

3. Third case: σ is a common event, $\sigma \in \Sigma_s$.

$$
\begin{aligned}
& \delta^{n_0}(\Theta_J x, \Theta_J \sigma) \\
= & \qquad \langle \text{ Typing of } \Theta_J \;\; \& \;\; \text{Definitions 4.7 and 4.8 } \rangle \\
& \delta^{n_0}(\langle(\Theta_J x)[1], \ldots, (\Theta_J x)[n_0]\rangle, \sigma) \\
= & \qquad \langle \text{ Definition of } \delta^{n_0} \text{ (Page 29) } \rangle \\
& \langle \delta_1((\Theta_J x)[1], \sigma), \ldots, \delta_{n_0}((\Theta_J x)[n_0], \sigma)\rangle \\
= & \qquad \langle \text{ Definitions 4.5 and 4.6 } \rangle \\
& \langle \delta_1(\{1/j_1\}(x[j_1]), \sigma), \ldots, \delta_{n_0}(\{n_0/j_{n_0}\}(x[j_{n_0}]), \sigma)\rangle \\
= & \qquad \langle \text{ PSA (Assumption 4.1) } \rangle \\
& \langle \{1/j_1\}\delta_{j_1}(x[j_1], \sigma), \ldots, \{n_0/j_{n_0}\}\delta_{j_{n_0}}(x[j_{n_0}], \sigma)\rangle \\
= & \qquad \langle \text{ Definition 4.6 } \rangle \\
& \theta_J \langle \delta_{j_1}(x[j_1], \sigma), \ldots, \delta_{j_{n_0}}(x[j_{n_0}], \sigma)\rangle \\
= & \qquad \langle \text{ Definition of } \delta^n \text{ (Page 29) } \;\; \& \;\; \delta^n(x,\sigma)! \ \rangle \\
& \theta_J \langle (\delta^n(x,\sigma))[j_1], \ldots, (\delta^n(x,\sigma))[j_{n_0}]\rangle \\
= & \qquad \langle \text{ Definition 4.5 and definition of } \Theta_J \text{ (Page 31) } \rangle \\
& \Theta_J \delta^n(x, \sigma)
\end{aligned}
$$

□

Lemma 4.18 *Let $x \in X^n$ and $J \in \mathcal{J}^n_{n_0}$.*

1. *If $s \in (\Sigma^n)^*$ and $\delta^n(x,s)!$, then $\delta^{n_0}(\Theta_J x, \Theta_J s) = \Theta_J \delta^n(x,s)$.*
2. *If $s \in (\Sigma^n - \Sigma_s)^*$ and $s = \uparrow_J s$ (this condition ensures that for all event σ in s, $\sigma \in \Sigma_i$ with $i \in J$), then $\delta^{n_0}(\Theta_J x, \Theta_J s)! \Leftrightarrow \delta^n(x,s)!$.*

Proof.

1. The proof is by induction. The base case is $s = \epsilon$. The result follows by using $\Theta_J \epsilon = \epsilon$ and the fact that, for all x, $\delta(x, \epsilon) = x$:

$$\delta^{n_0}(\Theta_J x, \Theta_J s) = \delta^{n_0}(\Theta_J x, \epsilon) = \Theta_J x = \Theta_J \delta^n(x, \epsilon) = \Theta_J \delta^n(x, s).$$

The induction case is $s = t\sigma$, for some $t \in (\Sigma^n)^*$ and $\sigma \in \Sigma^n$. Assume that $\delta^{n_0}(\Theta_J x, \Theta_J t) = \Theta_J \delta^n(x,t)$ if $\delta^n(x,t)!$. Since $\delta^n(x,s)!$ implies $\delta^n(x,t)!$, this is equivalent to assuming $\delta^{n_0}(\Theta_J x, \Theta_J t) = \Theta_J \delta^n(x,t)$. The result follows by using Definition 4.10, the fact that $\delta(x, ab) = \delta(\delta(x,a), b)$ for all x, a, b, the induction hypothesis and Lemma 4.17 (noting that $\delta^n(x,s)!$ implies $\delta^n(\delta^n(x,t),\sigma)!$):

$$\begin{aligned}
\delta^{n_0}(\Theta_J x, \Theta_J s) &= \delta^{n_0}(\Theta_J x, \Theta_J(t\sigma)) = \delta^{n_0}(\Theta_J x, (\Theta_J t)(\Theta_J \sigma)) \\
&= \delta^{n_0}(\delta^{n_0}(\Theta_J x, \Theta_J t), \Theta_J \sigma) = \delta^{n_0}(\Theta_J \delta^n(x,t), \Theta_J \sigma) \\
&= \Theta_J \delta^n(\delta^n(x,t),\sigma) = \Theta_J \delta^n(x, t\sigma) = \Theta_J \delta^n(x,s).
\end{aligned}$$

2. The proof by induction is similar to the preceding one. For the base case $s = \epsilon$, the result follows from $\delta^{n_0}(\Theta_J x, \epsilon)!$ and $\delta^n(x,\epsilon)!$. For the induction case $s = t\sigma$, assume that $\delta^{n_0}(\Theta_J x, \Theta_J t)! \Leftrightarrow \delta^n(x,t)!$ if $t \in (\Sigma^n - \Sigma_s)^*$ and $t = \uparrow_J t$. Since the hypotheses on s imply $t \in (\Sigma^n - \Sigma_s)^*$ and $t = \uparrow_J t$, this is equivalent to assuming $\delta^{n_0}(\Theta_J x, \Theta_J t)! \Leftrightarrow \delta^n(x,t)!$.

$\delta^{n_0}(\Theta_J x, \Theta_J s)!$

$\Leftrightarrow$ ⟨ Detailed steps are as in the proof of the first item ⟩

$\delta^{n_0}(\delta^{n_0}(\Theta_J x, \Theta_J t), \Theta_J \sigma)!$

$\Leftrightarrow$ ⟨ For all x, a, b, $\delta(\delta(x,a),b)! \Rightarrow \delta(x,a)!$ ⟩

$\delta^{n_0}(\Theta_J x, \Theta_J t)! \wedge \delta^{n_0}(\delta^{n_0}(\Theta_J x, \Theta_J t), \Theta_J \sigma)!$

$\Leftrightarrow$ ⟨ Induction hypothesis ⟩

$\delta^n(x,t)! \wedge \delta^{n_0}(\delta^{n_0}(\Theta_J x, \Theta_J t), \Theta_J \sigma)!$

$\Leftrightarrow$ ⟨ Part 1 of this lemma ⟩

$\delta^n(x,t)! \wedge \delta^{n_0}(\Theta_J \delta^n(x,t), \Theta_J \sigma)!$

$\Leftrightarrow$ ⟨ $s \in (\Sigma^n - \Sigma_s)^* \wedge s = \uparrow_J s \;\Rightarrow\; \sigma \in \Sigma^n - \Sigma_s \wedge \sigma = \uparrow_J \sigma$
$\Rightarrow \sigma \in \Sigma_i$ with $i \in J$ & Lemma 4.17 ⟩

$\delta^n(x,t)! \wedge \delta^n(\delta^n(x,t),\sigma)!$

$\Leftrightarrow$ ⟨ $s = t\sigma$ & Definition of ! for δ (Page 19) ⟩

$\delta^n(x,s)!$ □

Lemma 4.19 *Let $x \in X^n$ and $\sigma \in \Sigma^n$. Then*

$$\delta^n(x,\sigma)! \;\Leftrightarrow\; (\forall J \mid J \in \mathcal{J}^n_{n_0} : \delta^{n_0}(\Theta_J x, \Theta_J \sigma)!).$$

PROOF. The right implication ($\Rightarrow$) is a direct consequence of Lemma 4.17.

The proof of ($\Leftarrow$) is by contraposition. Suppose that $\delta^n(x,\sigma)$ is undefined. Then, there exists i ($1 \le i \le n$) such that $\delta_i(x[i],\sigma)$ is undefined and either $\sigma = \sigma_i$ or

$\sigma \in \Sigma_s$. Let $J \in \mathcal{J}^n_{n_0}$, with $i = j_k \in J$. If $\sigma = \sigma_i$, then, by PSA (Assumption 4.1), $\delta_k(\{k/j_k\}(x[j_k]), \sigma_k)$ is undefined; it follows that $\delta^{n_0}(\Theta_J x, \Theta_J \sigma)$ is undefined, because $\delta_k(\{k/j_k\}(x[j_k]), \sigma_k) = \delta_k((\Theta_J x)[k], \Theta_J \sigma_{j_k})$. If $\sigma \in \Sigma_s$, then, by PSA, $\delta_k(\{k/j_k\}(x[j_k]), \sigma)$ is undefined; it follows that $\delta^{n_0}(\Theta_J x, \Theta_J \sigma)$ is undefined, because $\delta_k(\{k/j_k\}(x[j_k]), \sigma) = \delta_k((\Theta_J x)[k], \Theta_J \sigma)$. □

Lemma 4.20 *Let $x \in X^n$ and $s \in (\Sigma^n)^*$. Then*

$$\delta^n(x, s)! \quad \Leftrightarrow \quad (\forall J \mid J \in \mathcal{J}^n_{n_0} : \delta^{n_0}(\Theta_J x, \Theta_J s)!).$$

Proof. The right implication ($\Rightarrow$) is a direct consequence of Lemma 4.18.

The proof of ($\Leftarrow$) is by contraposition. Suppose that $\delta^n(x, s)$ is undefined. Then $s = t\sigma u$ for some $t, u \in (\Sigma^n)^*$ and $\sigma \in \Sigma^n$ such that $\delta^n(x, t)!$ and $\delta^n(\delta^n(x, t), \sigma)$ is undefined. By Lemma 4.19, there exists $J \in \mathcal{J}^n_{n_0}$ such that $\delta^{n_0}(\Theta_J \delta^n(x, t), \Theta_J \sigma)$ is undefined. But

$\delta^{n_0}(\Theta_J x, \Theta_J s)!$

$\Leftrightarrow$ $\quad\langle\ s = t\sigma u\ \rangle$

$\delta^{n_0}(\Theta_J x, \Theta_J (t\sigma u))!$

$\Leftrightarrow$ $\quad\langle$ Definition 4.10 $\rangle$

$\delta^{n_0}(\Theta_J x, (\Theta_J t)(\Theta_J \sigma)(\Theta_J u))!$

$\Leftrightarrow$ $\quad\langle\ \delta(x, ab) = \delta(\delta(x, a), b)$ for all $x, a, b\ \rangle$

$\delta^{n_0}(\delta^{n_0}(\delta^{n_0}(\Theta_J x, \Theta_J t), \Theta_J \sigma), \Theta_J u)!$

$\Rightarrow$ $\quad\langle$ Since the outer δ^{n_0} is defined, its left argument is defined $\rangle$

$\delta^{n_0}(\delta^{n_0}(\Theta_J x, \Theta_J t), \Theta_J \sigma)!$

$\Leftrightarrow$ $\quad\langle\ \delta^n(x, t)!\ \ \&\ $ Lemma 4.18 $\rangle$

$\delta^{n_0}(\Theta_J \delta^n(x, t), \Theta_J \sigma)!$

so that $\delta^{n_0}(\Theta_J x, \Theta_J s)$ is undefined. □

As seen, PDESs exhibit symmetries that record an invariance property with respect to a change of process identity. This property constitutes the essence of PDESs and is expressed by two similarity assumptions throughout this thesis:

- Process Similarity Assumption (Assumption 4.1, PSA),
- Mask Similarity Assumption (Assumption 4.15, MSA).

PSA and MSA limit processes to be defined from a replicated structure. These assumptions appear very restrictive, but they are necessary to ensure that the different objects (e.g., processes, masks) manipulated in the higher dimension (n) are always consistent with the corresponding objects in the lower dimension (n_0). Overall, they capture homogeneity in a system. In addition to these assumptions, a condition is imposed on the events shared by the processes. They must be controllable. This condition is required to establish a fundamental result (Proposition 5.7) that is used to prove soundness of the synthesis method.

How far is it possible to relax some of these assumptions with respect to achieving soundness remains an open question that is discussed in Chapter 8.

Chapter 5

Parameterized Specifications

Adding parameters to a model, as presented in Chapter 4, entails adding corresponding parameters to the specifications. In order to draw conclusions about a system of arbitrary size from a system of bounded size with properties of interest (e.g., Σ_u-invariance, normality), specifications must exhibit symmetries. The method proposed in this thesis relies on no particular specification language. The specification must, however, be given by a parameterized predicate $Q^N \in \text{Pred}(X^N)$, which expresses conditions on indexed states. The predicates Q^{n_0} and Q^n, with $n_0 \leq n$, are instances of Q^N and represent the specifications for the system of bounded size (with n_0 processes) and a system of arbitrary size (with n processes), respectively.

Example 5.1 Let us consider the PDES described in Example 4.2. The following parameterized predicates are possible specifications for this system:

$$
\begin{aligned}
Q_1^N(x) &:\Leftrightarrow (\forall i, j \mid 1 \leq i, j \leq N \wedge i \neq j : \neg(x[i] = U_i \wedge x[j] = U_j));\\
Q_2^N(x) &:\Leftrightarrow (\forall i, j \mid 1 \leq i, j \leq N \wedge i < j : \neg(x[i] = R_i \wedge x[j] = U_j));\\
Q_3^N(x) &:\Leftrightarrow (\forall i, j, k, l \mid 1 \leq i, j, k, l \leq N \wedge \text{distinct}(i, j, k, l) :\\
&\qquad\qquad \neg(x[i] = U_i \wedge x[j] = U_j \wedge x[k] = U_k \wedge x[l] = U_l)).
\end{aligned}
$$

The first predicate forbids two users from sharing the resource. The second predicate is equivalent to giving priority to the user with the lowest number when the resource is free and simultaneously requested by some users or preventing a user from requesting the resource when it is already used by a user with a higher number. Finally, the last predicate permits at most three users to share the resource. □

Assumption 5.2 *Specification Similarity Assumption (SSA)—The assumption is*

$$(\exists n_0 \mid: (\forall n \mid n \geq n_0 : (\forall x \mid x \in X^n : Q^n(x) \Leftrightarrow (\forall J \mid J \in \mathcal{J}^n_{n_0} : Q^{n_0}(\Theta_J x))))).$$

Intuitively, SSA imposes the following restriction on instances of Q^N: a state $x \in X^n$ satisfies Q^n if and only if all the projections of x on the state space of dimension n_0 satisfy Q^{n_0}. When all instances Q^n of Q^N satisfy SSA, Q^N is said to satisfy SSA. In the sequel, when it is said that a predicate Q^N satisfies SSA for a given n_0, it means that

$$(\forall n \mid n \geq n_0 : (\forall x \mid x \in X^n : Q^n(x) \Leftrightarrow (\forall J \mid J \in \mathcal{J}^n_{n_0} : Q^{n_0}(\Theta_J x))))$$

holds for that n_0. SSA is closed under arbitrary conjunctions and disjunctions as shown by the next two propositions and illustrated by the companion examples.

Proposition 5.3 *Let Q^N be a parameterized predicate that satisfies SSA for a given n_0. Then Q^N satisfies SSA for any $m \geq n_0$.*

Proof. The proof is by induction on the value of m.

- Base case, $m = n_0$: This is direct, since Q^N satisfies SSA with n_0.
- Induction step: Assume that Q^N satisfies SSA for a given $k \geq n_0$. Then

$$\begin{array}{ll}
 & (\forall J \mid J \in \mathcal{J}^n_{k+1} : Q^{k+1}(\Theta_J x)) \\
\Leftrightarrow & \quad \langle \text{ Induction hypothesis with the specific instance } Q^{k+1} \rangle \\
 & (\forall J \mid J \in \mathcal{J}^n_{k+1} : (\forall J' \mid J' \in \mathcal{J}^{k+1}_k : Q^k(\Theta_{J'}(\Theta_J x)))) \\
\Leftrightarrow & \quad \langle \{\Theta_{J'}(\Theta_J x) \mid J \in \mathcal{J}^n_{k+1} \wedge J' \in \mathcal{J}^{k+1}_k\} = \{\Theta_J x \mid J \in \mathcal{J}^n_k\} \rangle \\
 & (\forall J \mid J \in \mathcal{J}^n_k : Q^k(\Theta_J x)) \\
\Leftrightarrow & \quad \langle \text{ Induction hypothesis } \rangle \\
 & Q^n(x).
\end{array}$$

□

For a given n_0, if Q^N and Q'^N satisfy SSA, then $Q^N \wedge Q'^N$ satisfies SSA (by distributivity of $\forall$ over $\wedge$). According to Proposition 5.3, if Q^N and Q'^N satisfy SSA for given n_0 and n'_0, respectively, then $Q^N \wedge Q'^N$ satisfies SSA with $\max(n_0, n'_0)$.

Example 5.4 Let us consider the parameterized predicates Q^N_1 and Q^N_2 in Example 5.1. The following proof shows that Q^N_2 satisfies SSA with $n_0 = 2$.

$$
\begin{aligned}
& (\forall J \mid J \in \mathcal{J}_2^n : Q_2^2(\Theta_J x)) \\
\Leftrightarrow \quad & \langle \text{ Definitions of } Q_2^n \text{ and } \mathcal{J}_{n_0}^n \ \ \& \ \text{ De Morgan } \rangle \\
& (\forall j_1, j_2 \mid 1 \le j_1 < j_2 \le n : (\forall i, j \mid 1 \le i, j \le 2 \wedge i < j : \\
& \qquad\qquad (\Theta_{\{j_1,j_2\}}x)[i] \ne R_i \vee (\Theta_{\{j_1,j_2\}}x)[j] \ne U_j)) \\
\Leftrightarrow \quad & \langle \text{ The constraints on } i \text{ and } j \text{ yield } i = 1 \text{ and } j = 2 \rangle \\
& (\forall j_1, j_2 \mid 1 \le j_1 < j_2 \le n : (\Theta_{\{j_1,j_2\}}x)[1] \ne R_1 \vee (\Theta_{\{j_1,j_2\}}x)[2] \ne U_2) \\
\Leftrightarrow \quad & \langle \ \Theta_J := \theta_J \circ \uparrow_J \ \ \& \ \text{ Applying } \uparrow_{\{j_i, j_2\}} \ \rangle \\
& (\forall j_1, j_2 \mid 1 \le j_1 < j_2 \le n : (\theta_{\{j_1,j_2\}}\langle x[j_1], x[j_2]\rangle)[1] \ne R_1 \vee \\
& \qquad\qquad (\theta_{\{j_1,j_2\}}\langle x[j_1], x[j_2]\rangle)[2] \ne U_2) \\
\Leftrightarrow \quad & \langle \text{ Definition 4.6 } \rangle \\
& (\forall j_1, j_2 \mid 1 \le j_1 < j_2 \le n : \langle \{1/j_1\}(x[j_1]), \{2/j_2\}(x[j_2])\rangle[1] \ne R_1 \vee \\
& \qquad\qquad \langle \{1/j_1\}(x[j_1]), \{2/j_2\}(x[j_2])\rangle[2] \ne U_2) \\
\Leftrightarrow \quad & \langle \text{ Component selection } \rangle \\
& (\forall j_1, j_2 \mid 1 \le j_1 < j_2 \le n : \{1/j_1\}(x[j_1]) \ne R_1 \vee \{2/j_2\}(x[j_2]) \ne U_2) \\
\Leftrightarrow \quad & \langle \text{ Remark 4.13 } \rangle \\
& (\forall j_1, j_2 \mid 1 \le j_1 < j_2 \le n : x[j_1] \ne \{j_1/1\}(R_1) \vee x[j_2] \ne \{j_2/2\}(U_2)) \\
\Leftrightarrow \quad & \langle \text{ Index substitution } \rangle \\
& (\forall j_1, j_2 \mid 1 \le j_1 < j_2 \le n : x[j_1] \ne R_{j_1} \vee x[j_2] \ne U_{j_2}) \\
\Leftrightarrow \quad & \langle \text{ Renaming the bound variables } \rangle \\
& (\forall i, j \mid 1 \le i < j \le n : x[i] \ne R_i \vee x[j] \ne U_j) \\
\Leftrightarrow \quad & \langle \text{ Definition of } Q_2^n \ \ \& \ \text{ De Morgan } \rangle \\
& Q_2^n(x)
\end{aligned}
$$

It can similarly be shown that Q_1^N also satisfies SSA with $n_0 = 2$. Therefore, $Q_1^N \wedge Q_2^N$ satisfies SSA with $n_0 = 2$. It should be noted that SSA is not closed under negation, since the predicate $\neg Q_1^N$ does not satisfy SSA. □

Proposition 5.5 *Let Q^N and Q'^N be two parameterized predicates that satisfy SSA for given n_0 and n'_0, respectively. Then $Q^N \vee Q'^N$ satisfies SSA with $n_0 + n'_0$; that is, SSA is closed under arbitrary disjunctions.*

Proof. The equivalent formula

$$\neg(Q^n \vee Q'^n)(x) \Leftrightarrow (\exists J \mid J \in \mathcal{J}^n_{n_0+n'_0} : \neg(Q^{n_0+n'_0} \vee Q'^{n_0+n'_0})(\Theta_J x))$$

is proved instead.

$\neg(Q^n \vee Q'^n)(x)$

$\Leftrightarrow$ $\langle$ De Morgan $\rangle$

$\neg Q^n(x) \wedge \neg Q'^n(x)$

$\Leftrightarrow$ $\langle$ SSA (Assumption 5.2) $\rangle$

$(\exists J \mid J \in \mathcal{J}^n_{n_0} : \neg Q^{n_0}(\Theta_J x)) \wedge (\exists J' \mid J' \in \mathcal{J}^n_{n'_0} : \neg Q'^{n'_0}(\Theta_{J'} x))$

$\Leftrightarrow$ $\langle$ For $\Rightarrow$, choose $J'' \in \mathcal{J}^n_{n_0+n'_0}$ such that $J \subseteq J'' \wedge J' \subseteq J''$ and use Proposition 5.3 & For $\Leftarrow$, use SSA (Assumption 5.2) $\rangle$

$(\exists J'' \mid J'' \in \mathcal{J}^n_{n_0+n'_0} : \neg Q^{n_0+n'_0}(\Theta_{J''} x) \wedge \neg Q'^{n_0+n'_0}(\Theta_{J''} x))$

$\Leftrightarrow$ $\langle$ De Morgan $\rangle$

$(\exists J'' \mid J'' \in \mathcal{J}^n_{n_0+n'_0} : \neg(Q^{n_0+n'_0} \vee Q'^{n_0+n'_0})(\Theta_{J''} x))$ □

Example 5.6 Let us consider the parameterized predicate Q_1^N in Example 5.1 and the following parameterized predicate:

$$Q_4^N(x) \quad :\Leftrightarrow \quad (\forall i, j \mid 1 \leq i, j \leq N \wedge i \neq j : \neg(x[i] = R_i \wedge x[j] = R_j)).$$

These predicates satisfy SSA with $n_0 = 2$. Let $x = \langle R_1, R_2, U_3, U_4 \rangle$. $(Q_1^4 \vee Q_4^4)(x)$ does not hold even if $Q_1^2 \vee Q_4^2$ holds for all the projections of x. However, according to Proposition 5.5, $Q_1^N \vee Q_4^N$ satisfies SSA with $n_0 = 4$. □

The following proposition establishes that, if Q^N satisfies SSA, then so does $\langle Q^N \rangle$. It should be noted that the strings of uncontrollable events s and t used in the proof of this proposition do not contain shared events because $\Sigma_s \cap \Sigma_u^n = \emptyset$ and $\Sigma_s \cap \Sigma_u^{n_0} = \emptyset$, respectively, by definition of Σ_s (Page 28).

Proposition 5.7 *Let Q^N be a parameterized predicate that satisfies SSA for a given n_0. Then $\langle Q^N \rangle$ satisfies SSA with n_0; that is, for all $n \geq n_0$ and for all $x \in X^n$, $\langle Q^n \rangle(x) \Leftrightarrow (\forall J \mid J \in \mathcal{J}^n_{n_0} : \langle Q^{n_0} \rangle(\Theta_J x))$.*

Proof. Suppose $Q^n(x) \Leftrightarrow (\forall J \mid J \in \mathcal{J}^n_{n_0} : Q^{n_0}(\Theta_J x))$. Proving the formula $\langle Q^n \rangle(x) \Leftrightarrow (\forall J \mid J \in \mathcal{J}^n_{n_0} : \langle Q^{n_0} \rangle(\Theta_J x))$ amounts to the same thing as proving the equivalent formula $\neg\langle Q^n \rangle(x) \Leftrightarrow (\exists J \mid J \in \mathcal{J}^n_{n_0} : \neg\langle Q^{n_0} \rangle(\Theta_J x))$.

$$
\begin{array}{ll}
 & \neg\langle Q^n \rangle(x) \\
\Leftrightarrow & \qquad \langle \text{ Definition of } \langle\cdot\rangle \text{ (Page 22) } \rangle \\
 & (\exists s \mid s \in (\Sigma^n_u)^* : \delta^n(x,s)! \wedge \neg Q^n(\delta^n(x,s))) \\
\Leftrightarrow & \qquad \langle \text{ SSA (Asumption 5.2) } \rangle \\
 & (\exists s \mid s \in (\Sigma^n_u)^* : \delta^n(x,s)! \wedge (\exists J \mid J \in \mathcal{J}^n_{n_0} : \neg Q^{n_0}(\Theta_J \delta^n(x,s)))) \\
\Leftrightarrow & \qquad \langle \text{ Distributivity of } \wedge \text{ over } \exists \ \ \& \ \ J \text{ not free in } \delta^n(x,s)! \rangle \\
 & (\exists s \mid s \in (\Sigma^n_u)^* : (\exists J \mid J \in \mathcal{J}^n_{n_0} : \delta^n(x,s)! \wedge \neg Q^{n_0}(\Theta_J \delta^n(x,s)))) \\
\Leftrightarrow & \qquad \langle \text{ Lemma 4.18 } \ \& \ \text{ Interchange of dummies } \rangle \\
 & (\exists J \mid J \in \mathcal{J}^n_{n_0} : (\exists s \mid s \in (\Sigma^n_u)^* : \delta^n(x,s)! \wedge \neg Q^{n_0}(\delta^{n_0}(\Theta_J x, \Theta_J s)))) \\
\Leftrightarrow & \qquad \langle \ (\exists t \mid t \in (\Sigma^{n_0}_u)^* : t = \Theta_J s) \text{ is } \mathsf{true} \ \rangle \\
 & (\exists J \mid J \in \mathcal{J}^n_{n_0} : (\exists s \mid s \in (\Sigma^n_u)^* : (\exists t \mid t \in (\Sigma^{n_0}_u)^* : t = \Theta_J s) \wedge \\
 & \qquad\qquad\qquad\qquad \delta^n(x,s)! \wedge \neg Q^{n_0}(\delta^{n_0}(\Theta_J x, \Theta_J s)))) \\
\Leftrightarrow & \qquad \langle \text{ Distributivity of } \wedge \text{ over } \exists \ \ \& \\
 & \qquad\ \ t \text{ not free in } \delta^n(x,s)! \wedge \neg Q^{n_0}(\delta^{n_0}(\Theta_J x, \Theta_J s)) \ \rangle \\
 & (\exists J \mid J \in \mathcal{J}^n_{n_0} : (\exists s \mid s \in (\Sigma^n_u)^* : (\exists t \mid t \in (\Sigma^{n_0}_u)^* : \\
 & \qquad\qquad t = \Theta_J s \wedge \delta^n(x,s)! \wedge \neg Q^{n_0}(\delta^{n_0}(\Theta_J x, \Theta_J s))))) \\
\Leftrightarrow & \qquad \langle \text{ Interchange of dummies } \ \& \ \text{ Using } t = \Theta_J s \ \rangle \\
 & (\exists J \mid J \in \mathcal{J}^n_{n_0} : (\exists t \mid t \in (\Sigma^{n_0}_u)^* : (\exists s \mid s \in (\Sigma^n_u)^* : \\
 & \qquad\qquad t = \Theta_J s \wedge \delta^n(x,s)! \wedge \neg Q^{n_0}(\delta^{n_0}(\Theta_J x, t))))) \\
\Leftrightarrow & \qquad \langle \text{ Distributivity of } \wedge \text{ over } \exists \ \ \& \ \ s \text{ not free in } \neg Q^{n_0}(\delta^{n_0}(\Theta_J x, t)) \ \rangle \\
 & (\exists J \mid J \in \mathcal{J}^n_{n_0} : (\exists t \mid t \in (\Sigma^{n_0}_u)^* : (\exists s \mid s \in (\Sigma^n_u)^* : t = \Theta_J s \wedge \delta^n(x,s)!) \wedge \\
 & \qquad\qquad\qquad\qquad \neg Q^{n_0}(\delta^{n_0}(\Theta_J x, t)))) \\
\Leftrightarrow & \qquad \langle \text{ For } \Leftarrow \text{, choose } s := \uparrow_J s \ \ \& \\
 & \qquad\ \ \text{For } \Rightarrow \text{, use } s \in (\Sigma^n_u)^* \Rightarrow \uparrow_J s \in (\Sigma^n_u)^*,\ \Theta_J s = \Theta_J \uparrow_J s \text{ and} \\
 & \qquad\ \ \delta^n(x,s)! \Rightarrow \delta^n(x, \uparrow_J s)! \text{ (by Remark 4.14) } \rangle \\
 & (\exists J \mid J \in \mathcal{J}^n_{n_0} : (\exists t \mid t \in (\Sigma^{n_0}_u)^* : (\exists s \mid \uparrow_J s \in (\Sigma^n_u)^* : t = \Theta_J \uparrow_J s \wedge \\
 & \qquad\qquad\qquad\qquad\qquad\qquad \delta^n(x, \uparrow_J s)!) \wedge \\
 & \qquad\qquad\qquad\qquad \neg Q^{n_0}(\delta^{n_0}(\Theta_J x, t)))) \\
\Leftrightarrow & \qquad \langle \ \Theta_J \uparrow_J s = \theta_J \uparrow_J \uparrow_J s = \theta_J \uparrow_J s \ \ \& \\
 & \qquad\ \ t = \theta_J \uparrow_J s \Leftrightarrow \theta_J^{-1} t = \uparrow_J s \text{ (by Remark 4.12) } \rangle
\end{array}
$$

$$(\exists J \mid J \in \mathcal{J}^n_{n_0} : (\exists t \mid t \in (\Sigma^{n_0}_u)^* : (\exists s \mid {\uparrow_J} s \in (\Sigma^n_u)^* : \theta_J^{-1} t = {\uparrow_J} s \wedge \delta^n(x, \theta_J^{-1}t)!) \wedge \neg Q^{n_0}(\delta^{n_0}(\Theta_J x, t))))$$

$\Leftrightarrow$ $\langle$ Distributivity of $\wedge$ over $\exists$ & s not free in $\delta^n(x, \theta_J^{-1}t)!$ $\rangle$

$$(\exists J \mid J \in \mathcal{J}^n_{n_0} : (\exists t \mid t \in (\Sigma^{n_0}_u)^* : (\exists s \mid {\uparrow_J} s \in (\Sigma^n_u)^* : \theta_J^{-1} t = {\uparrow_J} s) \wedge \delta^n(x, \theta_J^{-1}t)! \wedge \neg Q^{n_0}(\delta^{n_0}(\Theta_J x, t))))$$

$\Leftrightarrow$ $\langle$ Since $t \in (\Sigma^{n_0}_u)^*$, there exists a string of events s such that ${\uparrow_J} s \in (\Sigma^n_u)^*$ and $\theta_J^{-1} t = {\uparrow_J} s$, namely, $s := \theta_J^{-1} t$ $\rangle$

$$(\exists J \mid J \in \mathcal{J}^n_{n_0} : (\exists t \mid t \in (\Sigma^{n_0}_u)^* : \delta^n(x, \theta_J^{-1}t)! \wedge \neg Q^{n_0}(\delta^{n_0}(\Theta_J x, t))))$$

$\Leftrightarrow$ $\langle$ Lemma 4.18(2) & Remark 4.12 $\rangle$

$$(\exists J \mid J \in \mathcal{J}^n_{n_0} : (\exists t \mid t \in (\Sigma^{n_0}_u)^* : \delta^{n_0}(\Theta_J x, t)! \wedge \neg Q^{n_0}(\delta^{n_0}(\Theta_J x, t))))$$

$\Leftrightarrow$ $\langle$ Definition of $\langle\cdot\rangle$ (Page 22) $\rangle$

$$(\exists J \mid J \in \mathcal{J}^n_{n_0} : \neg \langle Q^{n_0} \rangle (\Theta_J x))$$ □

Example 5.8 This example shows that Proposition 5.7 would not stand in the presence of uncontrollable events in Σ_s. Consider the replicated structure in Figure 4.1b, in which event r is uncontrollable, and the predicate Q^N_1 in Example 5.1. It is easy to observe that $\langle Q^2_1 \rangle(\langle R_1, R_2 \rangle)$ does not hold (with the string $s = r$), but $\langle Q^3_1 \rangle(\langle R_1, R_2, U_3 \rangle)$ holds, since event r cannot occur for user 3.

Therefore, $\mathrm{wlp}_{\alpha_1}(\langle Q^3_1 \rangle)(\langle I_1, R_2, U_3 \rangle) \not\Rightarrow \mathrm{wlp}_{\alpha_1}(\langle Q^2_1 \rangle)(\langle I_1, R_2 \rangle)$, which means that disabling an event $\sigma \notin \Sigma_s$, such as α_1, in the lower dimension may be too restrictive in the higher dimension. This is not the case for an event $\sigma \in \Sigma_s$, because disabling such an event has no impact if the users cannot synchronize in the higher dimension. □

SSA relates Q^n and Q^{n_0}. In order to provide broader results, the restriction of Q^n with respect to a subset of $\mathcal{J}^n_{n_0}$ is introduced.

Definition 5.9 Let Q^N be a parameterized predicate that satisfies SSA for a given n_0 and let $\mathcal{I} \subseteq \mathcal{J}^n_{n_0}$. The restriction of Q^n with respect to $\mathcal{I}$, denoted $\lfloor Q^n \rfloor_{\mathcal{I}}$, is defined as: $\lfloor Q^n \rfloor_{\mathcal{I}}(x) :\Leftrightarrow (\forall J \mid J \in \mathcal{I} : Q^{n_0}(\Theta_J x))$, where it is implicitly assumed that if $J \in \mathcal{I}$, $J = \{j_1, \ldots, j_{n_0}\}$ and $1 \leq j_1 < \cdots < j_{n_0} \leq n$.

The definition of $\lfloor Q^n \rfloor_{\mathcal{I}}$ is consistent with SSA, because having $\lfloor Q^n \rfloor_{\mathcal{J}^n_{n_0}} = Q^n$ for all $n \geq n_0$ is equivalent to Q^N satisfying SSA (with n_0). Generally, $\lfloor Q^N \rfloor_{\mathcal{I}}$ does not satisfy SSA even if Q^N does (see Example 7.1). In Chapter 7, a set of subsets of indices $\mathcal{I}$ represents an interconnection relation between processes.

The following two propositions reveal the preservation, under the similarity assumptions, of Σ_u-*invariance* and *normality* properties when the state space is expanded from dimension n_0 to dimension n.

Proposition 5.10 *Let Q^N be a parameterized predicate that satisfies SSA for a given n_0. For all $n \geq n_0$, predicate $\lfloor Q^n \rfloor_{\mathcal{I}}$ is Σ_u^n-invariant if Q^{n_0} is $\Sigma_u^{n_0}$-invariant.*

Proof. By definition of the Σ_u^n-invariance property, the goal is to show that

$$(\forall \sigma \mid \sigma \in \Sigma_u^n : \lfloor Q^n \rfloor_{\mathcal{I}} \leq \mathrm{wlp}_\sigma(\lfloor Q^n \rfloor_{\mathcal{I}})),$$

which is equivalent to

$$(\forall \sigma \mid \sigma \in \Sigma_u^n : (\forall x \mid x \in X^n : \lfloor Q^n \rfloor_{\mathcal{I}}(x) \wedge \delta^n(x,\sigma)! \Rightarrow \lfloor Q^n \rfloor_{\mathcal{I}}(\delta^n(x,\sigma)))).$$

Suppose that $\sigma \in \Sigma_u^n$ and $\delta^n(x,\sigma)!$. Let us show that

$$\lfloor Q^n \rfloor_{\mathcal{I}}(x) \Rightarrow \lfloor Q^n \rfloor_{\mathcal{I}}(\delta^n(x,\sigma)).$$

$$
\begin{aligned}
& \lfloor Q^n \rfloor_{\mathcal{I}}(x) \\
\Leftrightarrow \quad & \langle \text{ Definition 5.9 } \ \& \ \delta^n(x,\sigma)! \ \& \text{ Lemma 4.19 } \rangle \\
& (\forall J \mid J \in \mathcal{I} : Q^{n_0}(\Theta_J x)) \wedge (\forall J \mid J \in \mathcal{J}^n_{n_0} : \delta^{n_0}(\Theta_J x, \Theta_J \sigma)!) \\
\Rightarrow \quad & \langle\ J \in \mathcal{I} \Rightarrow J \in \mathcal{J}^n_{n_0} \ \& \text{ Range strengthening } \rangle \\
& (\forall J \mid J \in \mathcal{I} : Q^{n_0}(\Theta_J x)) \wedge (\forall J \mid J \in \mathcal{I} : \delta^{n_0}(\Theta_J x, \Theta_J \sigma)!) \\
\Leftrightarrow \quad & \langle \text{ Distributivity } \rangle \\
& (\forall J \mid J \in \mathcal{I} : Q^{n_0}(\Theta_J x) \wedge \delta^{n_0}(\Theta_J x, \Theta_J \sigma)!) \\
\Leftrightarrow \quad & \langle\ \Theta_J\sigma = \epsilon \vee \Theta_J \sigma \neq \epsilon \ \& \text{ Distributivity } \rangle \\
& (\forall J \mid J \in \mathcal{I} : (\Theta_J\sigma = \epsilon \wedge Q^{n_0}(\Theta_J x) \wedge \delta^{n_0}(\Theta_J x, \Theta_J \sigma)!) \vee \\
& \qquad (\Theta_J\sigma \neq \epsilon \wedge Q^{n_0}(\Theta_J x) \wedge \delta^{n_0}(\Theta_J x, \Theta_J \sigma)!)) \\
\Rightarrow \quad & \langle\ \delta(x,\epsilon)! \ \& \ \delta(x,\epsilon) = x \ \& \\
& \quad \sigma \in \Sigma_u^n \wedge \Theta_J\sigma \neq \epsilon \Rightarrow \Theta_J\sigma \in \Sigma_u^{n_0} \ \& \ Q^{n_0} \text{ is } \Sigma_u^{n_0}\text{-invariant } \rangle \\
& (\forall J \mid J \in \mathcal{I} : (\Theta_J\sigma = \epsilon \wedge Q^{n_0}(\delta^{n_0}(\Theta_J x, \Theta_J \sigma))) \vee
\end{aligned}
$$

$$(\Theta_J\sigma \neq \epsilon \wedge Q^{n_0}(\delta^{n_0}(\Theta_J x, \Theta_J\sigma))))$$

$\Leftrightarrow$ $\langle$ Distributivity & $\Theta_J\sigma = \epsilon \vee \Theta_J\sigma \neq \epsilon$ $\rangle$

$(\forall J \mid J \in \mathcal{I} : Q^{n_0}(\delta^{n_0}(\Theta_J x, \Theta_J\sigma)))$

$\Leftrightarrow$ $\langle$ Assumption $\delta^n(x,\sigma)!$ & Lemma 4.17 $\rangle$

$(\forall J \mid J \in \mathcal{I} : Q^{n_0}(\Theta_J(\delta^n(x,\sigma))))$

$\Leftrightarrow$ $\langle$ Definition 5.9 $\rangle$

$\lfloor Q^n \rfloor_{\mathcal{I}}(\delta^n(x,\sigma))$ □

Example 5.11 The following counterexample shows that, in Proposition 5.10, the reverse implication does not hold, in particular when $\mathcal{I} = \mathcal{J}^n_{n_0}$ (hence $\lfloor Q^n \rfloor_{\mathcal{I}} = Q^n$, see Definition 5.9).

Consider a replicated structure close to the one in Figure 4.1a, but with events α_i and γ_i as controllable events and without event r. The parameterized predicate[1]

$$Q^N(x) :\Leftrightarrow (\forall i,j \mid 1 \leq i,j \leq N \wedge i \neq j : \neg(I_i \wedge I_j) \wedge \neg(R_i \wedge R_j) \wedge \neg(U_i \wedge U_j))$$

is such that, for $n \geq 4$, $Q^n = \mathsf{false}$. Thus $Q^n \leq \langle Q^n \rangle$ for $n \geq 4$ (see Remark 3.6).

In this example, $n_0 = 2$. The states $\langle I_1, R_2\rangle$, $\langle I_1, U_2\rangle$, $\langle R_1, I_2\rangle$, $\langle R_1, U_2\rangle$, $\langle U_1, I_2\rangle$ and $\langle U_1, R_2\rangle$ satisfy Q^2, but only the states $\langle I_1, R_2\rangle$, $\langle I_1, U_2\rangle$, $\langle R_1, I_2\rangle$ and $\langle U_1, I_2\rangle$ satisfy $\langle Q^2\rangle$. Therefore, $Q^2 \not\leq \langle Q^2\rangle$. □

Proposition 5.12 *Let Q^N be a parameterized predicate that satisfies SSA for a given n_0. For all $n \geq n_0$, predicate $\lfloor Q^n \rfloor_{\mathcal{I}}$ is normal if Q^{n_0} is normal.*

Proof. By definition of the normality property, the goal is to show that

$$(M^n)^{-1}(M^n(\lfloor Q^n \rfloor_{\mathcal{I}})) \leq \lfloor Q^n \rfloor_{\mathcal{I}}$$

when assuming $(M^{n_0})^{-1}(M^{n_0}(Q^{n_0})) \leq Q^{n_0}$. This is equivalent to showing

$$(\forall x \mid x \in X^n : (M^n)^{-1}(M^n(\lfloor Q^n \rfloor_{\mathcal{I}}))(x) \Rightarrow \lfloor Q^n \rfloor_{\mathcal{I}}(x)).$$

$(M^n)^{-1}(M^n(\lfloor Q^n \rfloor_{\mathcal{I}}))(x)$

[1]In several examples, the abbreviation A_i is used for $x[i] = A_i$, where $A_i \in X_i$.

$\Leftrightarrow$ $\langle$ See the definition of $M^{-1}M$ in Section 3.1 $\rangle$

$(\exists x' \mid x' \in X^n : M^n(x) = M^n(x') \wedge \lfloor Q^n \rfloor_{\mathcal{I}}(x'))$

$\Leftrightarrow$ $\langle$ Definition 4.3 & Definition 5.9 $\rangle$

$(\exists x' \mid x' \in X^n : (\forall J \mid J \in \mathcal{J}^n_{n_0} : \Theta_J M^n(x) = \Theta_J M^n(x')) \wedge$
$(\forall J \mid J \in \mathcal{I} : Q^{n_0}(\Theta_J x')))$

$\Rightarrow$ $\langle$ $J \in \mathcal{I} \Rightarrow J \in \mathcal{J}^n_{n_0}$ & Range strengthening & Lemma 4.16 $\rangle$

$(\exists x' \mid x' \in X^n : (\forall J \mid J \in \mathcal{I} : M^{n_0}(\Theta_J x) = M^{n_0}(\Theta_J x')) \wedge$
$(\forall J \mid J \in \mathcal{I} : Q^{n_0}(\Theta_J x')))$

$\Leftrightarrow$ $\langle$ Distributivity $\rangle$

$(\exists x' \mid x' \in X^n : (\forall J \mid J \in \mathcal{I} : M^{n_0}(\Theta_J x) = M^{n_0}(\Theta_J x') \wedge Q^{n_0}(\Theta_J x')))$

$\Rightarrow$ $\langle$ Interchange of dummies $\rangle$

$(\forall J \mid J \in \mathcal{I} : (\exists x' \mid x' \in X^n : M^{n_0}(\Theta_J x) = M^{n_0}(\Theta_J x') \wedge Q^{n_0}(\Theta_J x')))$

$\Rightarrow$ $\langle$ Taking $x'' = \Theta_J x'$ $\rangle$

$(\forall J \mid J \in \mathcal{I} : (\exists x'' \mid x'' \in X^{n_0} : M^{n_0}(\Theta_J x) = M^{n_0}(x'') \wedge Q^{n_0}(x'')))$

$\Leftrightarrow$ $\langle$ See the definition of $M^{-1}M$ in Section 3.1 $\rangle$

$(\forall J \mid J \in \mathcal{I} : (M^{n_0})^{-1}(M^{n_0}(Q^{n_0}))(\Theta_J x))$

$\Rightarrow$ $\langle$ Q^{n_0} is normal $\rangle$

$(\forall J \mid J \in \mathcal{I} : Q^{n_0}(\Theta_J x))$

$\Leftrightarrow$ $\langle$ Definition 5.9 $\rangle$

$\lfloor Q^n \rfloor_{\mathcal{I}}(x)$ □

Example 5.13 The following counterexample shows that, in Proposition 5.12, the reverse implication does not hold, in particular when $\mathcal{I} = \mathcal{J}^n_{n_0}$.

Consider the replicated structure in Figure 4.1a, the parameterized predicate in Example 5.11 and the mask M defined as: $M_i(I_i) = M_i(R_i) = S_i$ and $M_i(U_i) = T_i$. For $n \geq 4$, $Q^n = \mathsf{false}$ and thus $(M^n)^{-1}(M^n(Q^n)) = \mathsf{false}$. Therefore, $(M^n)^{-1}(M^n(Q^n)) \leq Q^n$.

As in Example 5.11, $n_0 = 2$ and the states $\langle I_1, R_2\rangle$, $\langle I_1, U_2\rangle$, $\langle R_1, I_2\rangle$, $\langle R_1, U_2\rangle$, $\langle U_1, I_2\rangle$ and $\langle U_1, R_2\rangle$ satisfy Q^2. Since the observable states $\langle S_1, S_2\rangle$, $\langle S_1, T_2\rangle$ and $\langle T_1, S_2\rangle$ satisfy $M^2(Q^2)$, $(M^2)^{-1}(M^2(Q^2))(x)$ holds for any state x that belongs to $X^2 - \{\langle U_1, U_2\rangle\}$. In particular, $\langle I_1, I_2\rangle$ satisfies $(M^2)^{-1}(M^2(Q^2))$, but not Q^2. □

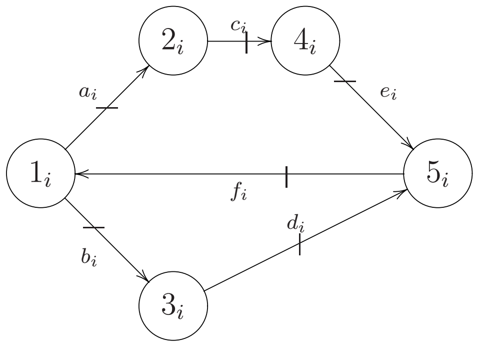

Figure 5.1: Replicated structure for Example 5.14

Controllability, *M-controllability* and *strong M-controllability* cannot generally be preserved, since they all contain a reachability condition in their definition. Let a state $x \in X^n$ be such that $Q^n(x)$ holds. Even if all the projections of x are reachable in the state space of dimension n_0, x may not be reachable. Generally, $Q^{n_0} \leq R(G^{n_0}, Q^{n_0}) \not\Rightarrow Q^n \leq R(G^n, Q^n)$. The next example illustrates this fact.

Example 5.14 Consider the replicated structure in Figure 5.1 and the following parameterized predicate:

$$\begin{aligned} Q^N(x) :\Leftrightarrow (\forall i, j \mid 1 \leq i, j \leq N \wedge i \neq j :\\ &\neg(1_i \wedge 3_j) \wedge \neg(1_i \wedge 4_j) \wedge \neg(2_i \wedge 2_j) \wedge \neg(2_i \wedge 4_j) \wedge \\ &\neg(3_i \wedge 3_j) \wedge \neg(3_i \wedge 4_j) \wedge \neg(3_i \wedge 5_j) \wedge \neg(4_i \wedge 4_j)). \end{aligned}$$

The predicate Q^N satisfies SSA with $n_0 = 2$. If the initial state of each instance of the PDES is derived from the parameterized state $x_0^N = \langle 1_1, \ldots, 1_N \rangle$, which is automorphic, it is a simple matter to verify that $Q^2 \leq R(G^2, Q^2)$ and $Q^3(\langle 1_1, 1_2, 5_3 \rangle)$ holds, but that $R(G^3, Q^3)(\langle 1_1, 1_2, 5_3 \rangle)$ does not hold. Hence, the specification similarity assumption does not preserve the reachability property when the state space is expanded from dimension n_0 to dimension n, even in the absence of synchronization. □

Similar to PDES, specifications must exhibit symmetries. Even if SSA narrows the form of those predicates representing constraints to be satisfied, it is necessary in order to ensure consistency between different corresponding objects from higher and lower dimensions. The possibility to restrict Q^n with respect to a subset of $\mathcal{J}^n_{n_0}$ permits the modeling of interconnection relations between processes and hence contributes to broader results. These interconnection relations are taken into account in Chapter 7.

Chapter 6

Control of Parameterized DES Under Total Observation

Since the state space grows exponentially with respect to n, it is unrealistic to compute an SFBC function for an arbitrarily large value of n. Therefore, the synthesis method proposed in this thesis includes two phases: an off-line synthesis and an on-line synthesis in which n_0 and n are involved, respectively. As mentioned in [Prosser et al. 1998], the only assumption needed is that the elapsed time period between event occurrences be longer than the on-line computation time. These limitations are reasonable in systems whose events do not occur very frequently or when computational resources are plentiful.

In this chapter, the SFBC functions on state spaces of dimension n_0 and n contain the set of prohibited controllable events and are denoted by $\overline{f}^{n_0}(\cdot)$ and $\overline{f}^{n}(\cdot)$, respectively. The control policy can then be presented in a more concise form. This modification with respect to the definitions given in Chapter 3 is taken into consideration in the proof of Theorem 6.6. Furthermore, the software environment used to compute $\overline{f}^{n_0}$ identifies the states that are unreachable under supervision by $\overline{f}^{n_0}$. They are indicated by empty entries (which are different from the empty set). Moreover, only the case where $\mathcal{I} = \mathcal{J}_{n_0}^{n}$ (no restriction on interconnections) is considered. The more general case where $\mathcal{I} \subseteq \mathcal{J}_{n_0}^{n}$ is developed in the next chapter, where partial observation is considered and where total observation is shown to be a special case.

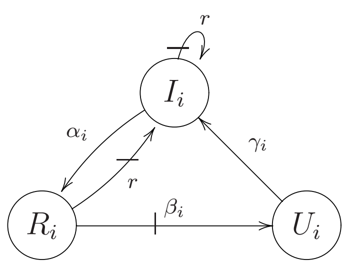

Figure 6.1: Replicated structures for the users

6.1 The Synthesis Method

An SFBC function is synthesized from a particular instance of G^N, say G^n, and a control specification. The latter can be given in two ways: i) by a parameterized predicate $Q^N \in \text{Pred}(X^N)$ or ii) by a predicate $Q^{n_0} \in \text{Pred}(X^{n_0})$ with $n_0 \leq n$. In the first case, Q^{n_0} and Q^n are instances of Q^N. In the second case Q^n is deduced from Q^{n_0} by similarity. In both cases, Q^{n_0} and Q^n must satisfy SSA.

To illustrate the definitions and synthesis procedure, let us again consider the running example of N independent users sharing a single resource (a simplified version of Example 4.2).

Example 6.1 Figure 6.1, which shows a transition structure that represents the behavior of user number i ($1 \leq i \leq N$), includes three states: I_i (Idle), R_i (Requesting), and U_i (Using). For instance, the user can move from state I_i to state R_i on event α_i (request the resource), then from state R_i to state U_i on event β_i (allocate the resource), and finally, from state U_i to state I_i on event γ_i (release the resource). There are two additional controllable transitions labeled r to reset all users in the initial configuration, one from state R_i to state I_i and a self-loop on state I_i. Event β_i is controllable. An SFBC function must be derived in order to satisfy the following constraint: only one user can own the resource at one time. This mutual exclusion property is specified by the following parameterized predicate:

$$Q^N(x) :\Leftrightarrow (\forall i, j \mid 1 \leq i, j \leq N \wedge i \neq j : \neg(x[i] = U_i \wedge x[j] = U_j)).$$

The predicate Q^N satisfies SSA with $n_0 = 2$. □

The off-line synthesis consists in calculating an SFBC function on X^{n_0} with respect to $(G^{n_0}, x_0^{n_0})$, Q^{n_0}, such that $Re(G|f^{n_0}) = \sup \mathcal{CP}(Q^{n_0})$, where n_0 usually denotes a small value. This problem is, in general, undecidable (see Page 200 of [Wonham 2008]), but since X_i is finite, a correct solution can be mechanically constructed by using a suitable synthesis algorithm for total observation.

Example 6.2 The following SFBC function $\overline{f}^2$ has been synthesized using our software environment.

$\langle I_1, I_2\rangle$: $\{\ \}$	$\langle R_1, I_2\rangle$: $\{\ \}$	$\langle U_1, I_2\rangle$: $\{\ \}$
$\langle I_1, R_2\rangle$: $\{\ \}$	$\langle R_1, R_2\rangle$: $\{\ \}$	$\langle U_1, R_2\rangle$: $\{\beta_2\}$
$\langle I_1, U_2\rangle$: $\{\ \}$	$\langle R_1, U_2\rangle$: $\{\beta_1\}$	$\langle U_1, U_2\rangle$:

□

Finally, the on-line synthesis computes $\overline{f}^n$ from $\overline{f}^{n_0}$, where $n_0 \leq n$, in the following way:

$$\overline{f}^n(x) := \bigcup_{J \in \mathcal{J}^n_{n_0}} \theta_J^{-1}\overline{f}^{n_0}(\Theta_J x), \tag{6.1}$$

where the term $\theta_J^{-1}\overline{f}^{n_0}(\Theta_J x)$ yields events that are prohibited because their projection, with respect to a given J, may lead from $\Theta_J x$ to a state in which the corresponding n_0 processes violate Q^{n_0}, either directly or after transitions with uncontrollable events.

Example 6.3 This example shows how $\overline{f}^3(\langle U_1, R_2, R_3\rangle)$ is calculated.

$$\begin{aligned}
\overline{f}^3(\langle U_1, R_2, R_3\rangle) &= \theta^{-1}_{\{1,2\}}\overline{f}^2(\Theta_{\{1,2\}}\langle U_1, R_2, R_3\rangle) \cup \theta^{-1}_{\{1,3\}}\overline{f}^2(\Theta_{\{1,3\}}\langle U_1, R_2, R_3\rangle) \\
&\quad \cup \theta^{-1}_{\{2,3\}}\overline{f}^2(\Theta_{\{2,3\}}\langle U_1, R_2, R_3\rangle) \\
&= \theta^{-1}_{\{1,2\}}\overline{f}^2(\langle U_1, R_2\rangle) \cup \theta^{-1}_{\{1,3\}}\overline{f}^2(\langle U_1, R_2\rangle) \cup \theta^{-1}_{\{2,3\}}\overline{f}^2(\langle R_1, R_2\rangle) \\
&= \theta^{-1}_{\{1,2\}}\{\beta_2\} \cup \theta^{-1}_{\{1,3\}}\{\beta_2\} \cup \theta^{-1}_{\{2,3\}}\{\ \} \\
&= \{\beta_2\} \cup \{\beta_3\}.
\end{aligned}$$

Hence, at state $\langle U_1, R_2, R_3\rangle$ (user 1 is using the resource), the events β_2 and β_3 are disabled because they would lead to states that violate $Q^3(x)$ like $\langle U_1, U_2, R_3\rangle$ (users 1 and 2 are using the resource) for example. □

The worst-case computational complexity for $\overline{f}^{n_0}$ is still exponential with respect to n_0, but as n_0 is usually small, this step becomes tractable. The computation of $\overline{f}^n(\cdot)$ relies on the number of elements in $\mathcal{J}^n_{n_0}$, which is $\binom{n}{n_0}$, with n_0 now being a constant.

Therefore, the worst-case computational complexity is in $O(n^{n_0})$, which is the same complexity class as $O((n-n_0+1)^{n_0})$, where the latter form better highlights the fact that when $n = n_0$, the computation of $\overline{f}^n(\cdot)$ is done in constant time. Of course, in this last scenario, the method presents no gain in computational complexity.

Example 6.4 The trains problem —Let N trains run on a unidirectional circular railway. The track is divided into ten different sections. The train i in section k, $0 \leq k \leq 9$, is represented by the state $x_{k,i}$ and the passage of train i from section k to the adjacent section $k \oplus 1$ by the event $x_kTOx_{k\oplus 1}\_t_i$, where $k \oplus 1 = (k+1) \bmod 10$. Formally, $\delta_i(x_{k,i}, x_kTOx_{k\oplus 1}\_t_i) = x_{k\oplus 1,i}$. The events $x_kTOx_{k\oplus 1}\_t_i$, with k odd, are controllable.

The behavior of the trains must be restrained to prevent the trains from colliding. Therefore, any two trains must be separated by at least one section to ensure that an incoming train can stop at a proper distance. This constraint is formally defined by the following parameterized predicate:

$$\begin{aligned} Q^N(x) :\Leftrightarrow (\forall i, j \mid 1 \leq i, j \leq N \wedge i \neq j \quad : \quad & \neg((x[i] = x_{k,i} \wedge x[j] = x_{k,j}) \vee \\ & (x[i] = x_{k,i} \wedge x[j] = x_{k\oplus 1,j}))). \end{aligned}$$

The predicate Q^N satisfies SSA with $n_0 = 2$. The SFBC function for $n_0 = 2$ and the initial state $\langle x_{1,1}, x_{3,2}\rangle$, calculated by a synthesis algorithm, is given in Fig. 6.2. Only the significant entries are listed. In a system with three trains, for which trains 1, 2, and 3 are in sections 5, 7 and 3, respectively, that is, $x = \langle x_{5,1}, x_{7,2}, x_{3,3}\rangle$, $\overline{f}^3(x)$ is equal to:

$$\begin{aligned} \overline{f}^3(x) &= \theta^{-1}_{\{1,2\}}\overline{f}^2(\Theta_{\{1,2\}}x) \cup \theta^{-1}_{\{1,3\}}\overline{f}^2(\Theta_{\{1,3\}}x) \cup \theta^{-1}_{\{2,3\}}\overline{f}^2(\Theta_{\{2,3\}}x) \\ &= \theta^{-1}_{\{1,2\}}\overline{f}^2(\langle x_{5,1}, x_{7,2}\rangle) \cup \theta^{-1}_{\{1,3\}}\overline{f}^2(\langle x_{5,1}x_{3,2}\rangle) \cup \theta^{-1}_{\{2,3\}}\overline{f}^2(\langle x_{7,1}, x_{3,2}\rangle) \\ &= \theta^{-1}_{\{1,2\}}\{x_5TOx_6\_t_1\} \cup \theta^{-1}_{\{1,3\}}\{x_3TOx_4\_t_2\} \cup \theta^{-1}_{\{2,3\}}\{\ \} \\ &= \{x_5TOx_6\_t_1\} \cup \{x_3TOx_4\_t_3\}. \end{aligned}$$

Hence, at state $\langle x_{5,1}, x_{7,2}, x_{3,3}\rangle$ events $x_5TOx_6\_t_1$ and $x_3TOx_4\_t_3$ must be disabled since they could lead to states, like $\langle x_{5,1}, x_{7,2}, x_{4,3}\rangle$, where two trains (trains 1 and 3) would not be separated by at least one section. In a system with five trains and for which $x_0 = \langle x_{1,1}, x_{3,2}, x_{5,3}, x_{7,4}, x_{9,5}\rangle$,

$$\overline{f}^5(x_0) = \{x_1TOx_2\_t_1, x_3TOx_4\_t_2, x_5TOx_6\_t_3, x_7TOx_8\_t_4, x_9TOx_0\_t_5\}.$$

In this case, it can be verified that the system is blocking. Finally, in a system with six trains, $\overline{f}^6(\langle x_{1,1}, x_{3,2}, x_{5,3}, x_{7,4}, s_{9,5}, x_{4,6}\rangle)$ is considered undefined, which means that Q^6 is false, because $\overline{f}^2(\langle x_{3,1}, x_{4,2}\rangle)$ and $\overline{f}^2(\langle x_{5,1}, x_{4,2}\rangle)$ are considered undefined, since the states $\langle x_{3,1}, x_{4,2}\rangle$ and $\langle x_{5,1}, x_{4,2}\rangle$ are unreachable in the system with two trains. $\square$

$\langle x_{0,1}, x_{7,2}\rangle : \{x_7TOx_8\_t_2\}$	$\langle x_{3,1}, x_{6,2}\rangle : \{x_3TOx_4\_t_1\}$	$\langle x_{7,1}, x_{5,2}\rangle : \{x_5TOx_6\_t_2\}$
$\langle x_{1,1}, x_{3,2}\rangle : \{x_1TOx_2\_t_1\}$	$\langle x_{4,1}, x_{1,2}\rangle : \{x_1TOx_2\_t_2\}$	$\langle x_{7,1}, x_{9,2}\rangle : \{x_7TOx_8\_t_1\}$
$\langle x_{1,1}, x_{4,2}\rangle : \{x_1TOx_2\_t_1\}$	$\langle x_{5,1}, x_{3,2}\rangle : \{x_3TOx_4\_t_2\}$	$\langle x_{8,1}, x_{5,2}\rangle : \{x_5TOx_6\_t_2\}$
$\langle x_{1,1}, x_{9,2}\rangle : \{x_9TOx_0\_t_2\}$	$\langle x_{5,1}, x_{7,2}\rangle : \{x_5TOx_6\_t_1\}$	$\langle x_{9,1}, x_{1,2}\rangle : \{x_9TOx_0\_t_1\}$
$\langle x_{2,1}, x_{9,2}\rangle : \{x_9TOx_0\_t_2\}$	$\langle x_{5,1}, x_{8,2}\rangle : \{x_5TOx_6\_t_1\}$	$\langle x_{9,1}, x_{2,2}\rangle : \{x_9TOx_0\_t_1\}$
$\langle x_{3,1}, x_{1,2}\rangle : \{x_1TOx_2\_t_2\}$	$\langle x_{6,1}, x_{3,2}\rangle : \{x_3TOx_4\_t_2\}$	$\langle x_{9,1}, x_{7,2}\rangle : \{x_7TOx_8\_t_2\}$
$\langle x_{3,1}, x_{5,2}\rangle : \{x_3TOx_4\_t_1\}$	$\langle x_{7,1}, x_{0,2}\rangle : \{x_7TOx_8\_t_1\}$	

Figure 6.2: The SFBC function for the trains problem ($n_0 = 2$)

6.2 Soundness

The following proposition is of great importance to prove the soundness of the synthesis method in the case of total observation. It states that an SFBC function achieving $\sup \mathcal{CP}(Q^n)$ enables an event σ at a state x if and only if, for every defined transformation Θ_J on σ and x, the SFBC function achieving $\sup \mathcal{CP}(Q^{n_0})$ permits the event $\Theta_J\sigma$ at the state $\Theta_J x$. As a transformation of an event could return ϵ, the convention that $f_\epsilon(x)$ holds is used.

Proposition 6.5 *Let $x \in X^n$ and $\sigma \in \Sigma_c^n$. Then*

$$f_\sigma^{n^*}(x) \Leftrightarrow (\forall J \mid J \in \mathcal{J}_{n_0}^n : f_{\Theta_J\sigma}^{n_0{}^*}(\Theta_J x)).$$

Proof.

$$f_\sigma^{n^*}(x)$$

$\Leftrightarrow$ $\langle$ Definition of $f_\sigma^{n^*}$ (Equation 3.2) $\rangle$

$$\delta^n(x,\sigma)! \wedge \langle Q^n\rangle(\delta^n(x,\sigma))$$

$\Leftrightarrow$ $\langle$ Lemma 4.20 $\rangle$

$$(\forall J \mid J \in \mathcal{J}_{n_0}^n : \delta^{n_0}(\Theta_J x, \Theta_J\sigma)!) \wedge \langle Q^n\rangle(\delta^n(x,\sigma))$$

$\Leftrightarrow$ $\langle$ Proposition 5.7 $\rangle$

$$(\forall J \mid J \in \mathcal{J}_{n_0}^n : \delta^{n_0}(\Theta_J x, \Theta_J\sigma)!) \wedge (\forall J \mid J \in \mathcal{J}_{n_0}^n : \langle Q^{n_0}\rangle(\Theta_J\delta^n(x,\sigma)))$$

$\Leftrightarrow$ $\langle$ Lemmas 4.17 and 4.19 $\rangle$

$$(\forall J \mid J \in \mathcal{J}_{n_0}^n : \delta^{n_0}(\Theta_J x, \Theta_J\sigma)!) \wedge (\forall J \mid J \in \mathcal{J}_{n_0}^n : \langle Q^{n_0}\rangle(\delta^{n_0}(\Theta_J x, \Theta_J\sigma)))$$

$\Leftrightarrow$ $\langle$ Distributivity $\rangle$

$(\forall J \mid J \in \mathcal{J}^n_{n_0} : \delta^{n_0}(\Theta_J x, \Theta_J \sigma)! \wedge \langle Q^{n_0}\rangle(\delta^{n_0}(\Theta_J x, \Theta_J \sigma)))$

$\Leftrightarrow$ $\langle$ Definition of $f^{n_0{}^*}_\sigma$ (Equation 3.2) $\rangle$

$(\forall J \mid J \in \mathcal{J}^n_{n_0} : f^{n_0{}^*}_{\Theta_J \sigma}(\Theta_J x))$ $\square$

The following theorem constitutes the main result of this chapter. It establishes the soundness of the synthesis method in the following sense. The SFBC function $\overline{f}^n$ calculated from $\overline{f}^{n_0}$ by the on-line synthesis procedure (6.1) is correct if $\overline{f}^{n_0}$ is correct, whenever $n \geq n_0$, where an SFBC function is said to be *correct* if it allows one to achieve $\sup\mathcal{CP}(Q)$. Intuitively, Theorem 6.6 says that if the SFBC function in the lower space allows one to achieve $\sup\mathcal{CP}(Q^{n_0})$ (the antecedent of Theorem 6.6) then the SFBC function calculated from Equation 6.1 (Page 51) is exactly the SFBC function that would have been synthesized from the concrete model (system with n processes) as defined by Equation 3.2 (Page 25) that allows one to achieve $\sup\mathcal{CP}(Q^n)$ (the consequent of Theorem 6.6).

Theorem 6.6 *Let* $x \in X^n$*,*

$$(\forall J \mid J \in \mathcal{J}^n_{n_0} : \overline{f}^{n_0}(\Theta_J x) = \{\sigma' \mid \sigma' \in \Sigma^{n_0}_c \wedge \neg f^{n_0{}^*}_{\sigma'}(\Theta_J x)\}) \Rightarrow$$
$$\overline{f}^n(x) = \{\sigma \mid \sigma \in \Sigma^n_c \wedge \neg f^{n^*}_\sigma(x)\}.$$

Proof.

$\overline{f}^n(x)$

$=$ $\langle$ Definition of $\overline{f}^n$ (Equation 6.1) $\rangle$

$\bigcup_{J\in\mathcal{J}^n_{n_0}} \theta_J^{-1}\overline{f}^{n_0}(\Theta_J x)$

$=$ $\langle$ Hypothesis $\rangle$

$\bigcup_{J\in\mathcal{J}^n_{n_0}} \theta_J^{-1}\{\sigma' \mid \sigma' \in \Sigma^{n_0}_c \wedge \neg f^{n_0{}^*}_{\sigma'}(\Theta_J x)\}$

$=$ $\langle$ Definition 4.9 $\rangle$

$\bigcup_{J\in\mathcal{J}^n_{n_0}} \{\theta_J^{-1}\sigma' \mid \sigma' \in \Sigma^{n_0}_c \wedge \neg f^{n_0{}^*}_{\sigma'}(\Theta_J x)\}$

$=$ $\langle$ Changing dummy & Remark 4.12 & $\sigma = \theta_J^{-1}\sigma' \Leftrightarrow \sigma' = \Theta_J\sigma)$ $\rangle$

$\bigcup_{J\in\mathcal{J}^n_{n_0}} \{\sigma \mid \theta_J\sigma \in \Sigma^{n_0}_c \wedge \neg f^{n_0{}^*}_{\theta_J\sigma}(\Theta_J x)\}$

$= \{\sigma \mid \sigma \in \Sigma^n_c \wedge (\exists J \mid J \in \mathcal{J}^n_{n_0} : \neg f^{n_0{}^*}_{\Theta_J\sigma}(\Theta_J x))\}$

$=$ $\langle$ Proposition 6.5 $\rangle$

$\{\sigma \mid \sigma \in \Sigma_c^n \wedge \neg f_\sigma^{n^*}(x)\}$ $\square$

Theorem 6.6 establishes a soundness that is stronger than strong soundness. The former soundness can be seen as a kind of syntactic soundness between f^n and f^{n*} where the latter, as in Theorem 7.18, can be seen as a kind of behavioural soundness between $Re(G^n|f^n)$ and $Re(G^n|f^{n*})$. This stronger version of soundness is particular to the case of total observation with the restriction that $\mathcal{I} = \mathcal{J}_{n_0}^n$.

Chapter 7

Control of Parameterized DES Under Partial Observation

In some real applications, states of a PDES are not completely observed. Those situations can be modeled by introducing a mask which is a mapping from the state space to the observation space. The case of partial observation raises some difficulties in term of controllability. The study of PDES in the context of partial observation requires new properties such as *normality* [Lin and Wonham 1988, Li 1991], *strong M–controllability* [Takai and Kodama 1997] and *M–controllability* [Takai et al. 1995]. Depending on the underlying controllability definition, different results in terms of supremal sublanguages are obtained and all those properties hide some pitfalls that significantly impact the goal of achieving strong soundness.

The synthesis method proposed in this chapter also includes two phases: an off-line synthesis and an on-line synthesis in which n_0 and n are involved, respectively. The off-line synthesis consists in calculating an SFBC function on X^{n_0} as permissive as possible, with respect to $(G^{n_0}, x_0^{n_0})$, Q^{n_0} and M^{n_0}, where n_0 usually denotes a small value. This problem is, in general, undecidable (see Page 200 of [Wonham 2008]), but since X_i is finite, a correct solution can be mechanically constructed by using a suitable synthesis algorithm where $\breve{f}^{n_0}$ and $\hat{f}^{n_0}$ can be computed from f^{n_0*} by using Equations 3.5 and 3.6, respectively.

The on-line synthesis includes the use of a symmetric interconnection relation $\mathcal{I} \subseteq \mathbb{N}^{n_0}$, which is part of the specification process. An n_0-ary relation is symmetric in the sense that if $(k_1, \ldots, k_{n_0}) \in \mathcal{I}$, then so is any permutation of $(k_1, \ldots, k_{n_0})$. Without loss of generality, these tuples are considered as indistinguishable and $(k_1, \ldots, k_{n_0})$ and $\{k_1, \ldots, k_{n_0}\}$ are used interchangeably. When using the latter form, $\mathcal{I}$ is handled as a subset of $\mathcal{J}^n_{n_0}$. The goal of an interconnection relation is to indicate the processes subjected to the specification. While a parameterized predicate captures constraints on the states of processes, an interconnection relation imposes additional constraints based on their identity.

Example 7.1 In addition to the predicates of Example 5.1, the following interconnection relations, which define classes of users, could be part of the specification of a control problem.[1]

$$\begin{aligned}
\mathcal{I}_1 &= \text{symmetric-closure}(\{(k_1, k_2) \mid 1 \leq k_1, k_2 \leq n \wedge k_2 = k_1 \oplus 1\});\\
\mathcal{I}_2 &= \text{symmetric-closure}(\{(k_1, 10) \mid k_1 \in \mathbb{N} \wedge k_1 \neq 10\});\\
\mathcal{I}_3 &= \{(k_1, k_2) \mid k_1, k_2 \in \mathbb{N} \wedge k_1 \neq k_2 \wedge k_1 \equiv k_2 \pmod 3\};\\
\mathcal{I}_4 &= \{(k_1, k_2) \mid k_1, k_2 \in \mathbb{N} \wedge k_1 \not\equiv k_2 \pmod 3\}.
\end{aligned}$$

For instance, Q^n_1 used in conjunction with $\mathcal{I}_1$ (which represents a ring) forbids two adjacent users from sharing the resource (like in the dining philosophers problem) and

$$\begin{aligned}
\lfloor Q^n_1 \rfloor_{\mathcal{I}_1}(x) &\Leftrightarrow Q^2_1(\Theta_{\{1,n\}}x) \wedge (\forall i \mid 1 \leq i \leq n-1 : Q^2_1(\Theta_{\{i,i\oplus 1\}}x))\\
&\Leftrightarrow Q^2_1(\theta_{\{1,n\}}\langle x[1], x[n]\rangle) \wedge\\
&\qquad (\forall i \mid 1 \leq i \leq n-1 : Q^2_1(\theta_{\{i,i\oplus 1\}}\langle x[i], x[i \oplus 1]\rangle))\\
&\Leftrightarrow (\forall i \mid 1 \leq i \leq n : \neg(x[i] = U_i \wedge x[i \oplus 1] = U_{i\oplus 1})).
\end{aligned}$$

The predicate $\lfloor Q^N_1 \rfloor_{\mathcal{I}_1}$ is an example of a parameterized predicate that does not satisfy SSA even if Q^N_1 does, because changing (through Θ_J) the identity of users that satisfy $\lfloor Q^n_1 \rfloor_{\mathcal{I}_1}$ can lead to users that do not satisfy $\lfloor Q^{n_0}_1 \rfloor_{\mathcal{I}_1}$ since, for example, two nonadjacent users could become adjacent after a projection.

The relation $\mathcal{I}_2$ (which represents a star) focuses on a specific user. The last two relations enable users i and j to share the resource depending on whether $i \equiv j \pmod 3$ or not. □

The arity of $\mathcal{I}$ must be equal to n_0 for two reasons. On the one hand, if the arity of $\mathcal{I}$ were less than n_0, some limitations would appear. For instance, the irreflexive and symmetric binary relation $\mathcal{I}_1$ used with Q^n_3 (an instance of Q^N_3 defined in Example 5.1) represents a mutual exclusion problem on pairs of adjacent users. In that particular case, limiting the interconnection relation to a binary relation reduces expressiveness. It prevents one from forbidding the use of the resource by a group of more than three consecutive users. On the other hand, if the arity of $\mathcal{I}$ were greater than n_0, some

[1] $i \oplus 1$ equals 1 if $i = n$, and $i + 1$ otherwise.

misinterpretations would be ineluctable. Computing an SFBC function on X^n from an SFBC function on a state space in a lower dimension would be dealt with case by case. For instance, what is the meaning of the following relation

$$\mathcal{I} = \{(i, j, k) \mid i, j, k \in \mathbb{N} \wedge \text{distinct}(i, j, k) \wedge (i = 5 \vee j = 5 \vee k = 5)\}$$

with respect to a state space of dimension two? However, based on Proposition 5.3, the aforementioned computation could be done from an SFBC function on X^m, where m is equal to the arity of $\mathcal{I}$.

7.1 Parameterized Specifications Under Partial Observation

Despite the fact that SSA does not preserve controllability, M-controllability and strong M-controllability (see Example 5.14), the next propositions and corollaries establish relationships between bad event sets in the state spaces of dimension n_0 and n. Knowing that an SFBC function can be expressed in terms of a bad event set (see 3.2 to 3.4 on page 25), these results are fundamental because they suggest a means for computing an SFBC function on X^n from an SFBC function on X^{n_0}. In the case of strong M–controllability, this association is not straightforward, because a discordant condition appears (see Condition 7.2 of Proposition 7.2).

As usual, the occurrence of an event that belongs to a bad event set associated with an observability class included in X^{n_0} leads to a state that violates Q^{n_0}. The bad event sets in dimension n are, however, calculated from the restriction of Q^n.

Propositions 7.2 and 7.5 deserve more attention. The former relates to *strong M-controllability* and the latter to *M-controllability*. They both establish relationships between bad event sets in the state spaces of dimension n_0 (Equations 7.1 and 7.2 in Proposition 7.2 for example) and n (Equation 7.3 in Proposition 7.2 for example). The main question behind those results is: how can we recover the bad event set in dimension n from projections to dimension n_0? That is the idea of the whole method. Proposition 7.2 and Proposition 7.5 bring two different answers (or characterizations). The result of Proposition 7.2 is stronger as it establishes an equivalence, hence fully characterizing the bad event set in dimension n from projections. Nevertheless, this equivalence is not straightforward. The result of Proposition 7.2 is directly reflected in Theorem 7.16 and leads to a strongly sound synthesis method (mainly because of the equivalence in Proposition 7.2).

Proposition 7.2 *Let Q^N be a parameterized predicate that satisfies SSA for a given n_0, and let $\mathcal{I} \subseteq \mathcal{J}^n_{n_0}$, $x \in X^n$ and $\sigma \in \Sigma^n_c$. If $\delta^n(x, \sigma)!$, then*

$$(\exists J \mid J \in \mathcal{I} : \Theta_J\sigma \in \hat{A}(Q^{n_0}, \Theta_J M^n(x))) \tag{7.1}$$

$$\vee$$

$$(\exists J \mid J \in \mathcal{I} : \Theta_J\sigma = \epsilon \wedge (\exists x' \mid x' \in X^{n_0} : \Theta_J M^n(x) = M^{n_0}(x') \wedge \neg Q^{n_0}(x'))) \tag{7.2}$$

$$\Leftrightarrow$$

$$\sigma \in \hat{A}(\lfloor Q^n \rfloor_{\mathcal{I}}, M^n(x)). \tag{7.3}$$

Proof.

$$
\begin{aligned}
& (\exists J \mid J \in \mathcal{I} : \Theta_J\sigma \in \hat{A}(Q^{n_0}, \Theta_J M^n(x))) \vee \\
& (\exists J \mid J \in \mathcal{I} : \Theta_J\sigma = \epsilon \wedge (\exists x' \mid x' \in X^{n_0} : \Theta_J M^n(x) = M^{n_0}(x') \wedge \neg Q^{n_0}(x'))) \\
\Leftrightarrow \quad & \langle \text{ Definition of } \hat{A} \text{ (Page 21) and } \mathrm{wlp}_\sigma \text{ (Page 21)} \;\;\&\;\; \delta(x,\epsilon)! \;\;\&\;\; \delta(x,\epsilon) = x \;\rangle \\
& (\exists J \mid J \in \mathcal{I} : \Theta_J\sigma \in \Sigma^{n_0}_c \wedge (\exists x' \mid x' \in X^{n_0} : \Theta_J M^n(x) = M^{n_0}(x') \wedge \\
& \qquad\qquad \delta^{n_0}(x', \Theta_J\sigma)! \wedge \neg Q^{n_0}(\delta^{n_0}(x', \Theta_J\sigma)))) \vee \\
& (\exists J \mid J \in \mathcal{I} : \Theta_J\sigma = \epsilon \wedge (\exists x' \mid x' \in X^{n_0} : \Theta_J M^n(x) = M^{n_0}(x') \wedge \\
& \qquad\qquad \delta^{n_0}(x', \Theta_J\sigma)! \wedge \neg Q^{n_0}(\delta^{n_0}(x', \Theta_J\sigma)))) \\
\Leftrightarrow \quad & \langle\; \sigma \in \Sigma^n_c \Rightarrow (\Theta_J\sigma \in \Sigma^{n_0}_c \Leftrightarrow \Theta_J\sigma \neq \epsilon) \;\;\&\;\; \text{Distributivity} \;\rangle \\
& (\exists J \mid J \in \mathcal{I} : (\Theta_J\sigma \neq \epsilon \vee \Theta_J\sigma = \epsilon) \wedge \\
& \qquad (\exists x' \mid x' \in X^{n_0} : \Theta_J M^n(x) = M^{n_0}(x') \wedge \\
& \qquad\qquad \delta^{n_0}(x', \Theta_J\sigma)! \wedge \neg Q^{n_0}(\delta^{n_0}(x', \Theta_J\sigma)))) \\
\Leftrightarrow \quad & \langle \text{ Excluded middle } \;\&\; \text{ Identity of } \wedge \;\rangle \\
& (\exists J \mid J \in \mathcal{I} : (\exists x' \mid x' \in X^{n_0} : \Theta_J M^n(x) = M^{n_0}(x') \wedge \\
& \qquad\qquad \delta^{n_0}(x', \Theta_J\sigma)! \wedge \neg Q^{n_0}(\delta^{n_0}(x', \Theta_J\sigma)))) \\
\Leftrightarrow \quad & \langle \text{ Use } x' = \Theta_J x'' \text{ with } x' \in X^{n_0} \text{ and } x'' \in X^n \;\rangle \\
& (\exists J \mid J \in \mathcal{I} : (\exists x'' \mid x'' \in X^n : \Theta_J M^n(x) = M^{n_0}(\Theta_J x'') \wedge \\
& \qquad\qquad \delta^{n_0}(\Theta_J x'', \Theta_J\sigma)! \wedge \neg Q^{n_0}(\delta^{n_0}(\Theta_J x'', \Theta_J\sigma))))
\end{aligned}
$$

$\Leftrightarrow$ $\langle$ $\overline{J}$ is the complement of J & There exists a state $x' \in X^n$ such that $\delta^n(x',\sigma)! \wedge \upharpoonright_J x' = \upharpoonright_J x'' \wedge \upharpoonright_{\overline{J}} M^n(x') = \upharpoonright_{\overline{J}} M^n(x)$, namely the state x' defined by $\upharpoonright_J x' = \upharpoonright_J x'' \wedge \upharpoonright_{\overline{J}} x' = \upharpoonright_{\overline{J}} x$. Indeed,

- if $i \in \overline{J}$, then $(\delta^n(x',\sigma))[i] = (\delta^n(x,\sigma))[i]$, since $\delta^n(x,\sigma)!$ by hypothesis;
- if $i \in J$ and $\sigma \in \Sigma_s \cup \Sigma_i$, then $(\delta^n(x',\sigma))[i] = (\delta^n(x'',\sigma))[i]$ by PSA and because $\delta^{n_0}(\Theta_J x'', \Theta_J\sigma)!$;
- if $i \in J$ and $\sigma \in \Sigma_j$, with $i \neq j$, then $(\delta^n(x',\sigma))[i] = x'[i]$ by definition of δ^n.

$\rangle$

$$(\exists J \mid J \in \mathcal{I} : (\exists x'' \mid x'' \in X^n : \Theta_J M^n(x) = M^{n_0}(\Theta_J x'') \wedge \delta^{n_0}(\Theta_J x'', \Theta_J\sigma)! \wedge \neg Q^{n_0}(\delta^{n_0}(\Theta_J x'', \Theta_J\sigma)) \wedge (\exists x' \mid x' \in X^n : \delta^n(x',\sigma)! \wedge \upharpoonright_J x' = \upharpoonright_J x'' \wedge \upharpoonright_{\overline{J}} M^n(x') = \upharpoonright_{\overline{J}} M^n(x))))$$

$\Leftrightarrow$ $\langle$ Nesting & Distributivity of $\wedge$ over $\exists$ & x' not free in $\Theta_J M^n(x) = M^{n_0}(\Theta_J x'') \wedge \delta^{n_0}(\Theta_J x'', \Theta_J\sigma)! \wedge \neg Q^{n_0}(\delta^{n_0}(\Theta_J x'', \Theta_J\sigma))$ $\rangle$

$$(\exists J \mid J \in \mathcal{I} : (\exists x', x'' \mid x', x'' \in X^n : \Theta_J M^n(x) = M^{n_0}(\Theta_J x'') \wedge \delta^{n_0}(\Theta_J x'', \Theta_J\sigma)! \wedge \neg Q^{n_0}(\delta^{n_0}(\Theta_J x'', \Theta_J\sigma)) \wedge \delta^n(x',\sigma)! \wedge \upharpoonright_J x' = \upharpoonright_J x'' \wedge \upharpoonright_{\overline{J}} M^n(x') = \upharpoonright_{\overline{J}} M^n(x)))$$

$\Leftrightarrow$ $\langle$ $\Theta_J x' = \Theta_J x''$, because $\upharpoonright_J x' = \upharpoonright_J x''$ $\rangle$

$$(\exists J \mid J \in \mathcal{I} : (\exists x', x'' \mid x', x'' \in X^n : \Theta_J M^n(x) = M^{n_0}(\Theta_J x') \wedge \delta^{n_0}(\Theta_J x', \Theta_J\sigma)! \wedge \neg Q^{n_0}(\delta^{n_0}(\Theta_J x', \Theta_J\sigma)) \wedge \delta^n(x',\sigma)! \wedge \upharpoonright_J x' = \upharpoonright_J x'' \wedge \upharpoonright_{\overline{J}} M^n(x') = \upharpoonright_{\overline{J}} M^n(x)))$$

$\Leftrightarrow$ $\langle$ Lemma 4.16 & Nesting & Distributivity of $\wedge$ over $\exists$ & x'' not free in $\Theta_J M^n(x) = \Theta_J M^n(x') \wedge \delta^n(x',\sigma)! \wedge \neg Q^{n_0}(\delta^{n_0}(\Theta_J x', \Theta_J\sigma)) \wedge \upharpoonright_{\overline{J}} M^n(x) = \upharpoonright_{\overline{J}} M^n(x')$ & Lemma 4.19 $\rangle$

$$(\exists J \mid J \in \mathcal{I} : (\exists x' \mid x' \in X^n : \Theta_J M^n(x) = \Theta_J M^n(x') \wedge \delta^n(x',\sigma)! \wedge \neg Q^{n_0}(\delta^{n_0}(\Theta_J x', \Theta_J\sigma)) \wedge \upharpoonright_{\overline{J}} M^n(x) = \upharpoonright_{\overline{J}} M^n(x') \wedge (\exists x'' \mid x'' \in X^n : \upharpoonright_J x' = \upharpoonright_J x'')))$$

$\Leftrightarrow$ $\langle$ $\Theta_J M^n(x) = \Theta_J M^n(x') \Leftrightarrow \upharpoonright_J M^n(x) = \upharpoonright_J M^n(x')$ (apply θ_J^{-1} to the left equality and θ_J to the right one to get the other) & There exists a state $x'' \in X^n$ such that $\upharpoonright_J x' = \upharpoonright_J x''$ $\rangle$

$$(\exists J \mid J \in \mathcal{I} : (\exists x' \mid x' \in X^n : \upharpoonright_J M^n(x) = \upharpoonright_J M^n(x') \wedge \delta^n(x',\sigma)! \wedge \neg Q^{n_0}(\delta^{n_0}(\Theta_J x', \Theta_J\sigma)) \wedge \upharpoonright_{\overline{J}} M^n(x) = \upharpoonright_{\overline{J}} M^n(x')))$$

$\Leftrightarrow$ $\langle\ v = w \Leftrightarrow \upharpoonright_J v = \upharpoonright_J w \wedge \upharpoonright_{\overline{J}} v = \upharpoonright_{\overline{J}} w$ & Interchange of dummies $\rangle$

$$(\exists x' \mid x' \in X^n : (\exists J \mid J \in \mathcal{I} : M^n(x) = M^n(x') \wedge \delta^n(x', \sigma)! \wedge \neg Q^{n_0}(\delta^{n_0}(\Theta_J x', \Theta_J \sigma))))$$

$\Leftrightarrow$ $\langle$ Lemma 4.17 & Distributivity of $\wedge$ over $\exists$ & J not free in $M^n(x) = M^n(x') \wedge \delta^n(x', \sigma)!$ $\rangle$

$$(\exists x' \mid x' \in X^n : M^n(x) = M^n(x') \wedge \delta^n(x', \sigma)! \wedge (\exists J \mid J \in \mathcal{I} : \neg Q^{n_0}(\Theta_J \delta^n(x', \sigma))))$$

$\Leftrightarrow$ $\langle\ Q^N$ satisfies SSA (Assumption 5.2) & Definition 5.9 $\rangle$

$$(\exists x' \mid x' \in X^n : M^n(x) = M^n(x') \wedge \delta^n(x', \sigma)! \wedge \neg \lfloor Q^n \rfloor_{\mathcal{I}}(\delta^n(x', \sigma)))$$

$\Leftrightarrow$ $\langle$ Definition of $\hat{A}$ (Page 21) and wlp_σ (Page 21) & Hypothesis $\sigma \in \Sigma_c^n$ $\rangle$

$$\sigma \in \hat{A}(\lfloor Q^n \rfloor_{\mathcal{I}}, M^n(x))$$ □

Since Proposition 7.2 is fundamental for understanding almost all the other results, here are some details about the three equations that comprise it.

First, Equation 7.3 simply says that σ is prohibited in the state space of dimension n for any state in the observability class of the state x and where the processes are subject to the interconnection relation $\mathcal{I}$.

Second, Equation 7.1 says that the projection/renaming of this σ (that is, $\Theta_J\sigma$) is prohibited in the state space of dimension n_0 for some projection $J \in \mathcal{I}$. By definition of $\hat{A}$, this projection/renaming ($\Theta_J\sigma$) cannot be ϵ. Hence, we see that Equation 7.1 considers the case where $\Theta_J\sigma \neq \epsilon$ while Equation 7.2 considers the case $\Theta_J\sigma = \epsilon$.

Finally, Equation 7.2 considers the case $\Theta_J\sigma = \epsilon$ and is less intuitive. Let us see why it is nevertheless needed. It is best explained through an example. Let us consider the system of Example 4.2 (Page 29). For simplicity, assume that $n = 3$, $\mathcal{I} = \mathcal{J}^n_{n_0}$, and that the specification prohibits two users from using the resource at the same time (mutual exclusion problem with $n_0 = 2$). Let the mask be $M(I_i) = I_i$, $M(R_i) = R_i$, $M(U_i) = R_i$. Now, consider the state $x = \langle U_1, U_2, I_3\rangle$. Note that x does not satisfy the specification and is observed as $M^3(x)$ which is equal to $\langle R_1, R_2, I_3\rangle$. Now, it is easy to verify that $\sigma = \alpha_3$ satisfies Equation 7.3 (by definition of $\hat{A}$ (Page 21)) and hence must be prohibited. Can this α_3 be recovered from Equation 7.1? No, because it cannot be recovered from $J = \{1, 3\}$ or $J = \{2, 3\}$ (the only two possibilities where $\Theta_J\alpha_3 \neq \epsilon$) since both of these projections allow α_2 (the renaming of α_3 after the projection/renaming). Intuitively, this situation happens because in the definition of $\hat{A}$, the state x needs not satisfy the specification and when this is the case, the triggering of an event, from a process that was not responsible for the nonsatisfaction of the specification in the state x (process 3 in the above example) still leads to a state that does not satisfy the specification and hence must be prohibited. In our example, for α_3 to be erased ($\Theta_J\alpha_3 = \epsilon$), we must take $J = \{1, 2\}$, and obtain the projected state $\langle U_1, U_2\rangle$ which is seen as $\langle R_1, R_2\rangle$ which is equal to $M^2(x')$ with $x' = \langle U_1, U_2\rangle$ and where this state x' does not satisfy the specification. This is exactly the kind of event that Equation 7.2 recuperates.

Corollary 7.3 *Let Q^N be a parameterized predicate that satisfies SSA for a given n_0, and let $\mathcal{I} \subseteq \mathcal{J}^n_{n_0}$, $x \in X^n$ and $\sigma \in \Sigma^n_c$. If $\delta^n(x, \sigma)!$, $\lfloor\langle Q^n\rangle\rfloor_{\mathcal{I}}(x)$ holds and the mask is the identity function, then*

$$(\forall J \mid J \in \mathcal{I} \wedge \Theta_J\sigma \neq \epsilon : \Theta_J\sigma \notin A(\langle Q^{n_0}\rangle, \Theta_J x)) \Leftrightarrow \sigma \notin A(\lfloor\langle Q^n\rangle\rfloor_{\mathcal{I}}, x).$$

Proof. $M(x) = x$ and $\hat{A}(Q, M(x)) = A(Q, x)$ when the mask is the identity function. Since $\langle Q^N\rangle$ satisfies SSA with n_0 by Proposition 5.7, Q^{n_0} and Q^n can be replaced in Proposition 7.2 by $\langle Q^{n_0}\rangle$ and $\langle Q^n\rangle$, respectively, and Condition 7.2 is false because $\lfloor\langle Q^n\rangle\rfloor_{\mathcal{I}}(x) \Rightarrow \langle Q^{n_0}\rangle(\Theta_J x)$ for any $J \in \mathcal{I}$. Conclusively, $\Theta_J\sigma \in A(\langle Q^{n_0}\rangle, \Theta_J x) \Rightarrow \Theta_J\sigma \neq \epsilon$. □

Corollary 7.3 shows that, under total observation, σ is not a bad event for the system with n processes if and only if $\Theta_J\sigma$ is not a bad event for the system with n_0 processes for any projection $J \in \mathcal{I}$ such that $\Theta_J\sigma \neq \epsilon$. This result makes it possible to conceive a strongly sound synthesis method (see Theorem 7.16).

The next corollary is the contrapositive of Proposition 7.2 and considers $\langle Q^n\rangle$ instead of Q^n.

Corollary 7.4 *Let Q^N be a parameterized predicate that satisfies SSA for a given n_0, and let $\mathcal{I} \subseteq \mathcal{J}^n_{n_0}$, $x \in X^n$ and $\sigma \in \Sigma^n_c$. If $\delta^n(x, \sigma)!$, then*

$$\begin{array}{l}(\forall J \mid J \in \mathcal{I} \wedge \Theta_J\sigma \neq \epsilon : \Theta_J\sigma \notin \hat{A}(\langle Q^{n_0}\rangle, \Theta_J M^n(x))) \wedge \\ (\forall J \mid J \in \mathcal{I} \wedge \Theta_J\sigma = \epsilon : (\forall x' \mid x' \in X^{n_0} \wedge \Theta_J M^n(x) = M^{n_0}(x') : \langle Q^{n_0}\rangle(x'))) \\ \Leftrightarrow \\ \sigma \notin \hat{A}(\lfloor\langle Q^n\rangle\rfloor_{\mathcal{I}}, M^n(x)).\end{array}$$

The next proposition shows that, compared with Proposition 7.2, only a weaker result can be established for $\breve{A}$ because of a further condition in its definition with respect to that of $\hat{A}$. Unfortunately, this leads to a weakly sound synthesis method (see Theorem 7.17).

Proposition 7.5 *Let Q^N be a parameterized predicate that satisfies SSA for a given n_0, and let $\mathcal{I} \subseteq \mathcal{J}^n_{n_0}$, $x \in X^n$ and $\sigma \in \Sigma^n_c$. Then*

$$\sigma \in \breve{A}(\lfloor Q^n \rfloor_{\mathcal{I}}, M^n(x)) \Rightarrow (\exists J \mid J \in \mathcal{I} : \Theta_J \sigma \in \breve{A}(Q^{n_0}, \Theta_J M^n(x))).$$

PROOF.

$\sigma \in \breve{A}(\lfloor Q^n \rfloor_{\mathcal{I}}, M^n(x))$

$\Leftrightarrow$ $\langle$ Definition of $\breve{A}$ (Page 21) & Hypothesis $\sigma \in \Sigma^n_c$ $\rangle$

$(\exists x' \mid x' \in X^n : M^n(x) = M^n(x') \wedge \lfloor Q^n \rfloor_{\mathcal{I}}(x') \wedge \delta^n(x', \sigma)! \wedge$
$\neg \lfloor Q^n \rfloor_{\mathcal{I}}(\delta^n(x', \sigma)))$

$\Leftrightarrow$ $\langle$ Q^N satisfies SSA (Assumption 5.2) & Definition 5.9 $\rangle$

$(\exists x' \mid x' \in X^n : M^n(x) = M^n(x') \wedge \lfloor Q^n \rfloor_{\mathcal{I}}(x') \wedge \delta^n(x', \sigma)! \wedge$
$\neg(\forall J \mid J \in \mathcal{I} : Q^{n_0}(\Theta_J \delta^n(x', \sigma))))$

$\Leftrightarrow$ $\langle$ De Morgan & Lemma 4.17 $\rangle$

$(\exists x' \mid x' \in X^n : M^n(x) = M^n(x') \wedge \lfloor Q^n \rfloor_{\mathcal{I}}(x') \wedge \delta^n(x', \sigma)! \wedge$
$(\exists J \mid J \in \mathcal{I} : \neg Q^{n_0}(\delta^{n_0}(\Theta_J x', \Theta_J \sigma))))$

$\Leftrightarrow$ $\langle$ J not free in $M^n(x) = M^n(x') \wedge \lfloor Q^n \rfloor_{\mathcal{I}}(x') \wedge \delta^n(x', \sigma)!$ &
Distributivity of $\wedge$ over $\exists$ $\rangle$

$(\exists x' \mid x' \in X^n : (\exists J \mid J \in \mathcal{I} : M^n(x) = M^n(x') \wedge \lfloor Q^n \rfloor_{\mathcal{I}}(x') \wedge \delta^n(x', \sigma)! \wedge$
$\neg Q^{n_0}(\delta^{n_0}(\Theta_J x', \Theta_J \sigma))))$

$\Rightarrow$ $\langle$ Q^N satisfies SSA & $\lfloor Q^n \rfloor_{\mathcal{I}}(x') \Rightarrow Q^{n_0}(\Theta_J x')$ by Definition 5.9 &
Monotonicity of $\exists$ $\rangle$

$(\exists x' \mid x' \in X^n : (\exists J \mid J \in \mathcal{I} : M^n(x) = M^n(x') \wedge Q^{n_0}(\Theta_J x') \wedge \delta^n(x', \sigma)! \wedge$
$\neg Q^{n_0}(\delta^{n_0}(\Theta_J x', \Theta_J \sigma))))$

$\Rightarrow$ $\langle$ $\Theta_J \sigma = \epsilon \wedge Q^{n_0}(\Theta_J x') \wedge \neg Q^{n_0}(\delta^{n_0}(\Theta_J x', \Theta_J \sigma))$
$\Rightarrow Q^{n_0}(\Theta_J x') \wedge \neg Q^{n_0}(\delta^{n_0}(\Theta_J x', \epsilon)) \Rightarrow Q^{n_0}(\Theta_J x') \wedge \neg Q^{n_0}(\Theta_J x')$
$\Rightarrow \mathsf{false}$ & Monotonicity of $\exists$ $\rangle$

$(\exists x' \mid x' \in X^n : (\exists J \mid J \in \mathcal{I} : \Theta_J M^n(x) = \Theta_J M^n(x') \wedge Q^{n_0}(\Theta_J x') \wedge$
$\delta^n(x', \sigma)! \wedge \Theta_J \sigma \neq \epsilon \wedge$
$\neg Q^{n_0}(\delta^{n_0}(\Theta_J x', \Theta_J \sigma))))$

$\Leftrightarrow \qquad \langle$ Lemma 4.16 & $\sigma \in \Sigma_c^n \Rightarrow (\Theta_J\sigma \in \Sigma_c^{n_0} \Leftrightarrow \Theta_J\sigma \neq \epsilon) \rangle$

$$\begin{aligned}(\exists x' \mid x' \in X^n : (\exists J \mid J \in \mathcal{I} : \Theta_J M^n(x) = M^{n_0}(\Theta_J x') \wedge Q^{n_0}(\Theta_J x') \wedge \\ \delta^n(x',\sigma)! \wedge \Theta_J\sigma \in \Sigma_c^{n_0} \wedge \\ \neg Q^{n_0}(\delta^{n_0}(\Theta_J x', \Theta_J\sigma))))\end{aligned}$$

$\Rightarrow \qquad \langle\ \delta^n(x',\sigma)! \Rightarrow \delta^{n_0}(\Theta_J x', \Theta_J\sigma)!$ by Lemma 4.19 & Monotonicity of $\exists\ \rangle$

$$\begin{aligned}(\exists x' \mid x' \in X^n : (\exists J \mid J \in \mathcal{I} : \Theta_J M^n(x) = M^{n_0}(\Theta_J x') \wedge Q^{n_0}(\Theta_J x') \wedge \\ \delta^{n_0}(\Theta_J x', \Theta_J\sigma)! \wedge \Theta_J\sigma \in \Sigma_c^{n_0} \wedge \\ \neg Q^{n_0}(\delta^{n_0}(\Theta_J x', \Theta_J\sigma))))\end{aligned}$$

$\Leftrightarrow \qquad \langle$ Interchange of dummies & x' not free in $\Theta_J\sigma \in \Sigma_c^{n_0}$ & Distributivity of $\wedge$ over $\exists\ \rangle$

$$\begin{aligned}(\exists J \mid J \in \mathcal{I} : \Theta_J\sigma \in \Sigma_c^{n_0} \wedge \\ (\exists x' \mid x' \in X^n : \Theta_J M^n(x) = M^{n_0}(\Theta_J x') \wedge Q^{n_0}(\Theta_J x') \wedge \\ \delta^{n_0}(\Theta_J x', \Theta_J\sigma)! \wedge \neg Q^{n_0}(\delta^{n_0}(\Theta_J x', \Theta_J\sigma))))\end{aligned}$$

$\Leftrightarrow \qquad \langle$ Use $x'' = \Theta_J x'$ with $x'' \in X^{n_0}$ and $x' \in X^n\ \rangle$

$$\begin{aligned}(\exists J \mid J \in \mathcal{I} : \Theta_J\sigma \in \Sigma_c^{n_0} \wedge \\ (\exists x'' \mid x'' \in X^{n_0} : \Theta_J M^n(x) = M^{n_0}(x'') \wedge Q^{n_0}(x'') \wedge \\ \delta^{n_0}(x'', \Theta_J\sigma)! \wedge \neg Q^{n_0}(\delta^{n_0}(x'', \Theta_J\sigma))))\end{aligned}$$

$\Leftrightarrow \qquad \langle$ Definition of $\breve{A}$ (Page 21) $\rangle$

$$(\exists J \mid J \in \mathcal{I} : \Theta_J\sigma \in \breve{A}(Q^{n_0}, \Theta_J M^n(x))) \qquad \square$$

Proposition 7.5 is understood in the same way as Proposition 7.2 but in the context of $\breve{A}$ instead of $\hat{A}$. In the former context, there is no need for an equation like Equation 7.2 since the definition of $\breve{A}$ has the condition that x must satisfy the specification. Intuitively, Proposition 7.5 shows that bad events can be recovered from projections but that in addition events that do not belong to the bad event set in dimension n could in fact be recovered too. So, a synthesis method based on Proposition 7.5 could lead to a controller that is unduly restrictive. Such a method would only be weakly sound and that is exactly what Theorem 7.17 shows.

The next corollary is the contrapositive of Proposition 7.5 and considers $\langle Q^n \rangle$ instead of Q^n.

Corollary 7.6 *Let Q^N be a parameterized predicate that satisfies SSA for a given n_0, and let $\mathcal{I} \subseteq \mathcal{J}_{n_0}^n$, $x \in X^n$ and $\sigma \in \Sigma_c^n$. Then*

$$(\forall J \mid J \in \mathcal{I} \wedge \Theta_J\sigma \neq \epsilon : \Theta_J\sigma \notin \breve{A}(\langle Q^{n_0}\rangle, \Theta_J M^n(x)))$$

$$\Rightarrow$$

$$\sigma \notin \breve{A}(\lfloor\langle Q^n\rangle\rfloor_{\mathcal{I}}, M^n(x)).$$

Example 7.7 This example shows that the reverse implication does not hold in Proposition 7.5, even if the state x is legal and $\delta^n(x,\sigma)!$ as in Proposition 7.2. Consider the replicated structure $P_i := (\{A_i, B_i, C_i, D_i, E_i\}, \{a_i\}, \delta_i)$, where the states A_i, C_i and D_i are in the same observability class, the event a_i is controllable, and $\delta_i(A_i, a_i) = B_i$, $\delta_i(B_i, a_i) = C_i$, $\delta_i(C_i, a_i) = D_i$ and $\delta_i(D_i, a_i) = E_i$.

Consider the case in which $\mathcal{I} = \mathcal{J}^n_{n_0}$ and the specification Q^N forbids any processes i and j to be simultaneously in states C_i and E_j. Therefore, $n_0 = 2$.

It can be seen that $\langle A_1, A_2, E_3\rangle$ is legal and that its observability class contains the following nine states: $\langle A_1, A_2, E_3\rangle$, $\langle C_1, A_2, E_3\rangle$, $\langle D_1, A_2, E_3\rangle$, $\langle A_1, C_2, E_3\rangle$, $\langle C_1, C_2, E_3\rangle$, $\langle D_1, C_2, E_3\rangle$, $\langle A_1, D_2, E_3\rangle$, $\langle C_1, D_2, E_3\rangle$ and $\langle D_1, D_2, E_3\rangle$. Out of these nine states, only the following four states satisfy the specification: $\langle A_1, A_2, E_3\rangle$, $\langle D_1, A_2, E_3\rangle$, $\langle A_1, D_2, E_3\rangle$ and $\langle D_1, D_2, E_3\rangle$. Now, $a_1 \notin \breve{A}(Q^3, M^3(\langle A_1, A_2, E_3\rangle))$ because out of the latter four states, the event a_1 is only defined for $\langle A_1, A_2, E_3\rangle$ and $\langle A_1, D_2, E_3\rangle$ and in both cases leads to a state that satisfies the specification. However, for $J = \{1, 2\}$, $a_1 \in \breve{A}(Q^2, M^2(\langle A_1, A_2\rangle))$ because the legal state $\langle D_1, C_2\rangle$, which is in the same observability class as the state $\langle A_1, A_2\rangle$, is such that $\delta^2(\langle D_1, C_2\rangle, a_1)$ is an illegal state. □

Remark 7.8 Suppose that Q^{n_0} is normal. Then $\lfloor Q^n\rfloor_{\mathcal{I}}$ is normal by Proposition 5.12, which means that $\lfloor Q^n\rfloor_{\mathcal{I}}(x) \Leftrightarrow \lfloor Q^n\rfloor_{\mathcal{I}}(x')$ for all x and x' in the same observability class (the evaluation of a normal predicate gives the same value for all states in the same observability class). If $\lfloor Q^n\rfloor_{\mathcal{I}}(x)$ holds, then $\breve{A}(\lfloor Q^n\rfloor_{\mathcal{I}}, M^n(x)) = \hat{A}(\lfloor Q^n\rfloor_{\mathcal{I}}, M^n(x))$; otherwise $\breve{A}(\lfloor Q^n\rfloor_{\mathcal{I}}, M^n(x)) = \emptyset$. Furthermore, if $\lfloor Q^n\rfloor_{\mathcal{I}}(x)$ holds, Condition 7.2 is always false because $Q^{n_0}(x')$ holds for any $x' \in X^{n_0}$ observed as $\Theta_J x$ whatever the projection $J \in \mathcal{I}$ (Lemma 4.16, Definition 5.9 and normality of Q^{n_0}). Therefore, under the hypothesis that $\delta^n(x,\sigma)!$ and $\lfloor Q^n\rfloor_{\mathcal{I}}(x)$ holds, it can be shown that

$$(\forall J \mid J \in \mathcal{I} \wedge \Theta_J\sigma \neq \epsilon : \Theta_J\sigma \notin \mathcal{A}(Q^{n_0}, \Theta_J M^n(x)))$$

$$\Leftrightarrow$$

$$\sigma \notin \mathcal{A}(\lfloor Q^n\rfloor_{\mathcal{I}}, M^n(x)),$$

where $\breve{A}$ or $\hat{A}$ can substitute for $\mathcal{A}$.

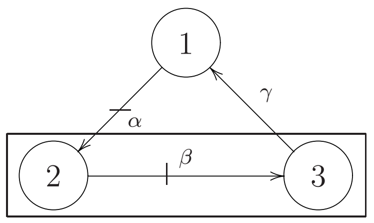

Figure 7.1: A three-state DES

Unfortunately, as Example 7.9 shows, if a predicate Q is normal, but not Σ_u-invariant, then $\langle Q \rangle$ is not necessarily normal and the previous result cannot be extended to $\langle Q^{n_0} \rangle$ and $\lfloor \langle Q^n \rangle \rfloor_{\mathcal{I}}$.

Example 7.9 Consider the DES depicted in Figure 7.1 where $\Sigma_u = \{\gamma\}$, $\Sigma_c = \{\alpha, \beta\}$ and where the states 2 and 3 belong to the same observability class. If $Q = \{2, 3\}$ then Q is normal but not Σ_u-invariant since $\langle Q \rangle = \{2\}$. Finally, $\langle Q \rangle$ is not normal since there is an observability class that contains both a legal and an illegal state (state 2 satisfies $\langle Q \rangle$ but state 3 does not). $\square$

7.2 The Synthesis Method

The on-line synthesis method consists in calculating an SFBC f^n in the following way for a given $x \in X^n$:

$$f^n(x) := \Sigma^n - \bigcup_{J \in \mathcal{I}} \left(\theta_J^{-1}(\Sigma^{n_0} - \mathbf{f}^{n_0}(\Theta_J x)) \cup \boldsymbol{\xi}\right), \tag{7.4}$$

where the term $\theta_J^{-1}(\Sigma^{n_0} - \mathbf{f}^{n_0}(\Theta_J x))$ yields events that are prohibited because their projection, with respect to a given J, may lead from $\Theta_J x$ (or possibly another state observed as $\Theta_J x$ under the mask) to a state in which the corresponding n_0 interconnected processes violate Q^{n_0}, either directly or after transitions with uncontrollable events. The other term, $\boldsymbol{\xi}$, represents the set of controllable events erased by J ($\Theta_J \sigma = \epsilon$), but that must nevertheless be prohibited because there are unsafe states in the observability class of $\Theta_J x$. Recall that $J \in \mathcal{I}$ implies $J = \{j_1, \ldots, j_{n_0}\}$ and $1 \leq j_1 < \cdots < j_{n_0} \leq n$, for some $j_1, \ldots, j_{n_0}$. The terms $\mathbf{f}^{n_0}$ and $\boldsymbol{\xi}$ are written in bold because they are the parameters of the synthesis procedure and the substitution of specific objects for $\mathbf{f}^{n_0}$ and $\boldsymbol{\xi}$ fixes the context: total observation or partial observation founded on M-controllability or strong M-controllability.

7.2.1 The Case of Total Observation

In the case of total observation, the mask is the identity function and $\boldsymbol{\xi}$ is replaced by $\emptyset$ in Equation 7.4. Furthermore, it is shown in Section 7.5 that the synthesis method is strongly sound; that is,

$$\text{if } Re(G^{n_0}|\mathbf{f}^{n_0}) = \sup \mathcal{CP}(Q^{n_0}), \text{ then } Re(G^n|f^n) = R(G^n, \lfloor\langle Q^n\rangle\rfloor_{\mathcal{I}}).$$

For instance, if $\mathbf{f}^{n_0}$ is replaced by f^{n_0*}, which is defined by (3.2) on page 25, f^n is behaviorally equivalent to the SFBC function derived from the same procedure as that used to synthesize f^{n_0*}, but by considering the predicate $\lfloor Q^n\rfloor_{\mathcal{I}}$.

Example 7.10 For the system of Example 4.2 with $Q_1^N \wedge Q_2^N$ as specification (Q_1^N and Q_2^N are defined in Example 5.1) and $\mathcal{I}_1$ as interconnection relation ($\mathcal{I}_1$ is defined in Example 7.1), the optimal SFBC is expressed as follows for $n_0 = 2$:

$$\overline{f}^{2*}(\langle I_1, U_2\rangle) = \{\alpha_1\},\ \overline{f}^{2*}(\langle R_1, R_2\rangle) = \{\beta_2\},\ \overline{f}^{2*}(\langle U_1, R_2\rangle) = \{\beta_2\}$$

and $\overline{f}^{2*}(\langle x_1, x_2\rangle) = \emptyset$ for all other states, where, as in Chapter 6, $\overline{f}^{2*}(\cdot) := \Sigma^2 - f^{2*}(\cdot)$ is the set of prohibited controllable events. By using Equation 7.4:

$$\begin{aligned}
\overline{f}^4(\langle R_1, I_2, U_3, R_4\rangle) &= \theta^{-1}_{\{1,2\}}\overline{f}^{2*}(\Theta_{\{1,2\}}\langle R_1, I_2, U_3, R_4\rangle) \cup \\
&\quad\ \theta^{-1}_{\{1,4\}}\overline{f}^{2*}(\Theta_{\{1,4\}}\langle R_1, I_2, U_3, R_4\rangle) \cup \\
&\quad\ \theta^{-1}_{\{2,3\}}\overline{f}^{2*}(\Theta_{\{2,3\}}\langle R_1, I_2, U_3, R_4\rangle) \cup \\
&\quad\ \theta^{-1}_{\{3,4\}}\overline{f}^{2*}(\Theta_{\{3,4\}}\langle R_1, I_2, U_3, R_4\rangle) \cup \\
&= \theta^{-1}_{\{1,2\}}\overline{f}^{2*}(\langle R_1, I_2\rangle) \cup \theta^{-1}_{\{1,4\}}\overline{f}^{2*}(\langle R_1, R_2\rangle) \cup \\
&\quad\ \theta^{-1}_{\{2,3\}}\overline{f}^{2*}(\langle I_1, U_2\rangle) \cup \theta^{-1}_{\{3,4\}}\overline{f}^{2*}(\langle U_1, R_2\rangle) \\
&= \theta^{-1}_{\{1,2\}}\{\ \} \cup \theta^{-1}_{\{1,4\}}\{\beta_2\} \cup \theta^{-1}_{\{2,3\}}\{\alpha_1\} \cup \theta^{-1}_{\{3,4\}}\{\beta_2\} \\
&= \{\beta_4\} \cup \{\alpha_2\} \cup \{\beta_4\} \\
&= \{\alpha_2, \beta_4\}.
\end{aligned}$$

Even if user 3 holds the resource, β_1 is not forbidden because users 1 and 3 are not connected ($(1,3) \notin \mathcal{I}_1$). Event β_4 is prohibited because it would lead to the state $\langle R_1, I_2, U_3, U_4\rangle$ which violates both Q_1^4 and Q_2^4 since users 1 and 4 are connected. Finally, event α_2 is prohibited because it leads to the state $\langle R_1, R_2, U_3, R_4\rangle$ which violates Q_2^4 since users 2 and 3 are connected. □

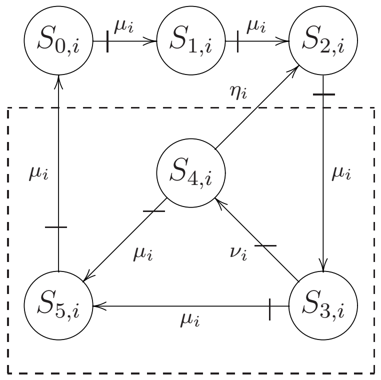

Figure 7.2: Replicated structure for the carts

7.2.2 The Case of Partial Observation

In the case of partial observation, the expression for $\boldsymbol{\xi}$ depends on the underlying property. For strong M-controllability, the term $\theta_J^{-1}(\Sigma^{n_0} - \mathbf{f}^{n_0}(\Theta_J x))$ in Equation 7.4 corresponds to Condition 7.1 in Proposition 7.2 and Condition 7.2 indicates that $\boldsymbol{\xi}$ must be replaced by

$$\{\sigma \in \Sigma_c^n \mid \Theta_J \sigma = \epsilon \wedge (\exists x' \mid x' \in X^{n_0} : M^{n_0}(\Theta_J x) = M^{n_0}(x') \wedge \neg \langle Q^{n_0} \rangle (x'))\}. \quad (7.5)$$

Indeed, $\Theta_J M^n(x) = M^{n_0}(\Theta_J x)$ by Lemma 4.16 and $\langle Q^{n_0} \rangle$ is used instead of Q^{n_0}, because $\mathbf{f}^{n_0}$ is replaced by $\hat{f}^{n_0}$, which is defined by Equation 3.4 on page 25.

In this setting, the set $\boldsymbol{\xi}$ contains events erased by the projection J that is considered, but that must be disabled because the projection of the state x on J is in an observability class in which there is an unsafe state (for instance, x' does not satisfy $\langle Q^{n_0} \rangle$). If $\delta^n(x, \sigma)!$, this implies that σ must be forbidden by definition of $\hat{A}$. In fact, let $x'' \in X^n$ be such that $x''[i] = x[i]$ if $i \notin J$, and $x''[j_k] = \{j_k/k\}(x'[k])$ if $j_k \in J$. It can be checked that $M^n(x) = M^n(x'')$ and $\delta^n(x'', \sigma)!$ ($\sigma \notin \Sigma_s$ because $\Theta_J \sigma = \epsilon$). Furthermore, $\Theta_J \delta^n(x'', \sigma) = \delta^{n_0}(\Theta_J x'', \Theta_J \sigma) = \delta^{n_0}(x', \epsilon) = x'$. By Proposition 5.7 and Definition 5.9, $\delta^n(x'', \sigma)$ cannot satisfy $\lfloor \langle Q^n \rangle \rfloor_{\mathcal{I}}$.

When the synthesis method is founded on M-controllability, more states are reachable under control while maintaining a predicate invariant and $\boldsymbol{\xi}$ is replaced by $\emptyset$ as indicated by Corollary 7.6. The following example illustrates the variation between these two cases.

Example 7.11 Consider a cart-traffic control system over a floor-running carrier divided into six sections. The replicated structure for the carts is depicted in Figure 7.2.

The fact that cart i is in section k, $0 \leq k \leq 5$, is represented by the state $S_{k,i}$. The unidirectional movements of cart i from a given section are indicated by the controllable events μ_i and ν_i, and the uncontrollable event η_i. The states $S_{3,i}$, $S_{4,i}$ and $S_{5,i}$ are in the same observability class; that is, $M_i(S_{3,i}) = M_i(S_{4,i}) = M_i(S_{5,i})$. Each section has a capacity of one, except sections 0 and 1, which have unlimited capacity. This constraint is formulated by the following parameterized predicate:

$$Q^N(x) \Leftrightarrow (\forall i, j, k \mid 1 \leq i, j \leq N \wedge i \neq j \wedge 2 \leq k \leq 5 : \neg(x[i] = S_{k,i} \wedge x[j] = S_{k,j})).$$

The system must be controlled in order to provide a safe automatic transportation of materials for all carts ($\mathcal{I} = \mathcal{J}^n_{n_0}$). By definition of $\hat{A}$,

$$\hat{A}(\langle Q^3 \rangle, M^3(\langle S_{0,1}, S_{2,2}, S_{3,3} \rangle)) = \{\mu_1, \mu_2, \nu_3\}.$$

For instance, since the states $S_{4,i}$ and $S_{3,i}$ are in the same observability class, the state $\langle S_{0,1}, S_{2,2}, S_{4,3} \rangle$ is observed as the state $\langle S_{0,1}, S_{2,2}, S_{3,3} \rangle$ and the transition with μ_1 from the former to $\langle S_{1,1}, S_{2,2}, S_{4,3} \rangle$ is defined, but $\langle Q^3 \rangle(\langle S_{1,1}, S_{2,2}, S_{4,3} \rangle)$ does not hold because the uncontrollable transition with η_3 from $\langle S_{1,1}, S_{2,2}, S_{4,3} \rangle$ leads to $\langle S_{1,1}, S_{2,2}, S_{2,3} \rangle$, which does not satisfy Q^3.

The evaluation of $\hat{A}$ for the projections of $\langle S_{0,1}, S_{2,2}, S_{3,3} \rangle$ in the state space of dimension 2 yields:

$$\begin{aligned} \hat{A}(\langle Q^2 \rangle, M^2(\langle S_{0,1}, S_{2,2} \rangle)) &= \hat{A}(\langle Q^2 \rangle, M^2(\langle S_{0,1}, S_{3,2} \rangle)) = \emptyset; \\ \hat{A}(\langle Q^2 \rangle, M^2(\langle S_{2,1}, S_{3,2} \rangle)) &= \{\mu_1, \nu_2\}. \end{aligned}$$

From the above bad event sets, the value of $\hat{A}(\langle Q^3 \rangle, M^3(\langle S_{0,1}, S_{2,2}, S_{3,3} \rangle))$ cannot be recovered, in particular, event μ_1, since $\theta^{-1}_{\{2,3\}}\{\mu_1, \nu_2\} = \{\mu_2, \nu_3\}$. However, $\Theta_{\{2,3\}}\mu_1 = \epsilon$,

$$M^2(\Theta_{\{2,3\}}\langle S_{0,1}, S_{2,2}, S_{3,3} \rangle) = M^2(\langle S_{2,1}, S_{3,2} \rangle) = M^2(\langle S_{2,1}, S_{4,2} \rangle)$$

and $\langle Q^2 \rangle(\langle S_{2,1}, S_{4,2} \rangle)$ does not hold. Therefore, the value associated with $\boldsymbol{\xi}$ is $\{\mu_1\}$ according to Equation 7.5. It should be noted that the state $\langle S_{0,1}, S_{2,2}, S_{4,3} \rangle$ is ignored in the calculation of $\breve{A}(\langle Q^3 \rangle, M^3(\langle S_{0,1}, S_{2,2}, S_{3,3} \rangle))$, which is equal to $\{\mu_2, \nu_3\}$, because $\neg\langle Q^3 \rangle(\langle S_{0,1}, S_{2,2}, S_{4,3} \rangle)$. □

It is shown in Section 7.5 that

$$\text{if } Re(G^{n_0}|\mathbf{f}^{n_0}) = \hat{R}(G^{n_0}, \langle Q^{n_0} \rangle), \text{ then } Re(G^n|f^n) = \hat{R}(G^n, \lfloor \langle Q^n \rangle \rfloor_{\mathcal{I}})$$

and

$$\text{if } Re(G^{n_0}|\mathbf{f}^{n_0}) = \breve{R}(G^{n_0}, \langle Q^{n_0} \rangle), \text{ then } Re(G^n|f^n) \leq \breve{R}(G^n, \lfloor \langle Q^n \rangle \rfloor_{\mathcal{I}}).$$

This means that the synthesis method is strongly sound if $\mathbf{f}^{n_0}$ is replaced by $\hat{f}^{n_0}$. Once again, f^n is behaviorally equivalent to the SFBC function derived from the same procedure than the one used to synthesize $\hat{f}^{n_0}$, namely the one that implements (3.4), but by considering the predicate $\lfloor Q^n \rfloor_{\mathcal{I}}$. This is not the case if $\mathbf{f}^{n_0}$ is replaced by $\breve{f}^{n_0}$, where $\breve{f}^{n_0}$ is defined by Equation 3.3, because, in that particular case, the method is only weakly sound.

7.3 Implementation of the On-line Synthesis

Equation 7.4 involves some calculations that are unnecessary when considering the history of the closed-loop system behavior at run-time. On a state change following the occurrence of an event $\sigma \in \Sigma^n$, it is sufficient to consider the projections that contain the identity of at least one process among those that have progressed on σ (this set of processes is denoted by P). The other projections, those for which $J \cap P = \emptyset$, can be ignored, because $\Theta_J \delta^n(x, \sigma) = \Theta_J x$. Indeed, the current control action can be established by using positive counters, one per controllable event that belongs to Σ_c^n.

Let γ_σ be the counter associated with $\sigma \in \Sigma_c^n$. Its value gives the number of projections that prevent the evolution of all processes on σ if $\sigma \in \Sigma_s$, or the evolution of process i on σ if $\sigma \in \Sigma_i$. Therefore, if $\gamma_\sigma = 0$, then σ is enabled; otherwise, it is disabled. The counters, which are a representation of a multiset of prohibited events, are updated according to the algorithm in Figure 7.3. The initial step (lines 1 to 3) considers only the initial state and all its projections of interconnected processes as in Equation 7.4. Line 4 calculates the set P from local state changes, where $M(x')$ is the current observable state that results from an observable state change following the occurrence of an event when the system was in the previous observable state $M(x)$. Lines 5 to 11 increase or decrease some counters based on the information deduced from the previous state. Consider the subset of n_0 processes associated with a given projection $J \in \mathcal{I}$ and an event σ such that $\Theta_J \sigma \neq \epsilon$. The evolution of these processes through a sequence of observable states $x^1, \ldots x^l$, such that $\Theta_J \sigma \in \mathbf{f}^{n_0}(\Theta_J x^k) \Leftrightarrow \Theta_J \sigma \in \mathbf{f}^{n_0}(\Theta_J x^{k+1})$ $(1 \leq k < l)$, never changes the value of γ_σ with respect to J (see the conditions in lines 7–8 and 10–11). If the next state x^{l+1} results from the progression of exactly one (on an asynchronous event) or some (on a synchronous event) of these processes ($J \cap P \neq \emptyset$) and $\neg(\Theta_J \sigma \in \mathbf{f}^{n_0}(\Theta_J x^l) \Leftrightarrow \Theta_J \sigma \in \mathbf{f}^{n_0}(\Theta_J x^{l+1}))$, then γ_σ is increased (respectively, decreased) because this time the condition in lines 7–8 (respectively, lines 10–11) is satisfied. This indicates that the event σ that was enabled (respectively, disabled) is now disabled (respectively, enabled) with respect to J. In the case of partial observation, internal state changes are equivalent to self-loops on a representative state and the algorithm is still correct

Initial step

```
1.  for all $\sigma \in \Sigma_c^n$ do $\gamma_\sigma := 0$;
2.  for all $J \in \mathcal{I}$ do
3.      for all $\sigma \in \theta_J^{-1}(\Sigma^{n_0} - \mathbf{f}^{n_0}(\Theta_J x_0^n)) \cup \boldsymbol{\xi}$ do $\gamma_\sigma := \gamma_\sigma + 1$.
```

Other steps

```
    // $x'$ is the current state
    // $x$ is the previous state
4.  $P := \{i \mid (M^n(x'))[i] \neq (M^n(x))[i]\}$
5.  for all $J \in \mathcal{I}$ such that $J \cap P \neq \emptyset$ do
6.      for all $\sigma \in \Sigma_c^n$ do
7.          if $\sigma \notin \theta_J^{-1}(\Sigma^{n_0} - \mathbf{f}^{n_0}(\Theta_J x)) \cup \boldsymbol{\xi}$ and
8.             $\sigma \in \theta_J^{-1}(\Sigma^{n_0} - \mathbf{f}^{n_0}(\Theta_J x')) \cup \boldsymbol{\xi}$ then $\gamma_\sigma := \gamma_\sigma + 1$;
9.          else
10.            if $\sigma \in \theta_J^{-1}(\Sigma^{n_0} - \mathbf{f}^{n_0}(\Theta_J x)) \cup \boldsymbol{\xi}$ and
11.               $\sigma \notin \theta_J^{-1}(\Sigma^{n_0} - \mathbf{f}^{n_0}(\Theta_J x')) \cup \boldsymbol{\xi}$ then $\gamma_\sigma := \gamma_\sigma - 1$.
```

Figure 7.3: Algorithm for the on-line synthesis

because of Assumption 3.1.

Example 7.12 This example shows how the counters are updated by the algorithm in Figure 7.3 when applied to a sequence of states from $\langle I_1, I_2, R_3, I_4\rangle$ to $\langle I_1, I_2, I_3, R_4\rangle$ on the admissible sequence of events $\alpha_4\alpha_1\beta_3\gamma_3\beta_1\gamma_1$, by using the SFBC f^{2*} in Example 7.10 and the interconnection relation $\mathcal{I}_1$ in Example 7.1. The following trace shows the evolution of counters:

				α_1	α_2	α_3	α_4	β_1	β_2	β_3	β_4	r
I_1	I_2	R_3	I_4	0	0	0	0	0	0	0	0	0
			$\downarrow \alpha_4$									
I_1	I_2	R_3	R_4	0	0	0	0	0	0	0	$1\ \{3,4\}$	0
$\downarrow \alpha_1$												
R_1	I_2	R_3	R_4	0	0	0	0	0	0	0	$2\ \{1,4\}$	0
		$\downarrow \beta_3$										
R_1	I_2	U_3	R_4	0	$1\ \{2,3\}$	0	0	0	0	0	2	0
		$\downarrow \gamma_3$										
R_1	I_2	I_3	R_4	0	$0\ \{2,3\}$	0	0	0	0	0	$1\ \{3,4\}$	0
$\downarrow \beta_1$												
U_1	I_2	I_3	R_4	0	0	0	0	0	0	0	1	0
$\downarrow \gamma_1$												
I_1	I_2	I_3	R_4	0	0	0	0	0	0	0	$0\ \{1,4\}$	0

The projection used to update a counter appears to the right of its value in order to emphasize a modification. It can be seen that i belongs to this projection on a local state change of P_i. □

7.4 Computational Complexity

The worst-case computational complexity for $\mathbf{f}^{n_0}$ is still exponential with respect to n_0, but, as n_0 is usually small, this step becomes tractable. Additional information required in the space of dimension n_0, namely, the set of states which are in an observability class that contains a state x such that $\neg\langle Q^{n_0}\rangle(x)$, can also be precomputed before system execution. Thus, the term $\boldsymbol{\xi}$ can be calculated in constant time for a given J.

The computation of $f^n(\cdot)$, by using Equation 7.4, relies on the number of elements in $\mathcal{I} \subseteq \mathcal{J}^n_{n_0}$, which is $\binom{n}{n_0}$ in the worst-case, with n_0 now being a constant. Therefore, as in the total observation case, the worst-case computational complexity is in $O(n^{n_0})$, which is the same complexity class as $O((n - n_0 + 1)^{n_0})$, where the latter form better highlights the fact that when $n = n_0$, the computation of $f^n(\cdot)$ is done in constant time. Of course, in this last scenario, the method presents no gain in computational complexity.

However, the algorithm in Figure 7.3 considers only $\binom{n-1}{n_0-1}$ projections in the case of the occurrence of an asynchronous event (because $|P| = 1$). The computational cost is reduced by a factor n/n_0. This linear gain on complexity is generally important. For example, a quadratic algorithm ($n_0 = 2$) becomes linear. Finally, the algorithm could be adapted to the case where $|P|$ is large, for which it is better to use Equation 7.4 with a memoization technique to record the control actions for later reuse. Furthermore, if none of these control actions disables events, only the initialization of counters to zero is then required.

7.5 Soundness of the Synthesis Method

The proof of the soundness of the synthesis method depends on the SFBC function used in the state space of dimension n_0 and the expression used for $\boldsymbol{\xi}$ when considering Equation 7.4 as (a specification of) the algorithm for computing enabled events.

First, links must be established between the method (Equation 7.4) and the different contexts (Propositions 7.2 and 7.5). This is what Lemmas 7.13 and 7.14 do. For example, Lemma 7.13 characterizes the method in the context of strong M-controllability and intuitively links Equation 7.4 to Corollary 7.4. It then turns out to be the main result used in the proof of Theorem 7.16. In fact, from Lemma 7.13, we see that the inferred SFBC function on the state space of dimension n, obtained through Equation 7.4, is in accordance with the result of Proposition 7.2 (and hence Corollary 7.4) and paves the way for a strongly sound synthesis method, proved by Theorem 7.16. Similar reasoning applies to Proposition 7.5, Corollary 7.6, Lemma 7.14 and Theorem 7.17 (for M-controllability, and a weakly sound synthesis method this time).

The following lemmas characterize f^n given by Equation 7.4 with other expressions according to substitutions for the parameters $\mathbf{f}^{n_0}$ and $\boldsymbol{\xi}$. These preliminary results are mainly used for proving the soundness of the synthesis method, but they also reveal something that is not apparent in Equation 7.4. In the case of partial observation, it seems that the supervisor, represented by Equation 7.4, handles the system state x, which it is not supposed to observe. The next two propositions clearly show that only $M^n(x)$ is used.

Lemma 7.13 *Let Q^{n_0} be an instance of a parameterized predicate Q^N, $\mathcal{I} \subseteq \mathcal{J}^n_{n_0}$ and $x \in X^n$. If, in Equation 7.4, $\hat{f}^{n_0}$ (defined by Equation 3.4) and Equation 7.5 substitute for $\mathbf{f}^{n_0}$ and $\boldsymbol{\xi}$, respectively, then*

$$
\begin{aligned}
f^n(x) = \Sigma^n_u \cup &\\
&\big\{\sigma \mid \sigma \in \Sigma^n_c \wedge \\
&\quad (\forall J \mid J \in \mathcal{I} \wedge \Theta_J\sigma \neq \epsilon : \Theta_J\sigma \notin \hat{A}(\langle Q^{n_0}\rangle, \Theta_J M^n(x))) \wedge \\
&\quad (\forall J \mid J \in \mathcal{I} \wedge \Theta_J\sigma = \epsilon : \\
&\qquad (\forall x' \mid x' \in X^{n_0} \wedge \Theta_J M^n(x) = M^{n_0}(x') : \langle Q^{n_0}\rangle(x')))\big\}.
\end{aligned}
$$

Intuitively, Lemma 7.13 characterizes the events that should be enabled at state x (in the higher dimension space) when the underlying property is strong M-controllability. First, uncontrollable events must be enabled. Second, two conditions need to be satisfied in order for a controllable event to be enabled. The first one is that all non-erased projections of that event must lead to events that are themselves enabled in the lower dimension space. The second condition relates to Equation 7.2 of Proposition 7.2 and states that the projections that erase the controllable event must project the state x (under the mask) to an observability class where all states satisfy $\langle Q^{n_0}\rangle$.

Proof.

$$
\begin{aligned}
& f^n(x) \\
= \quad & \langle \text{ Equation 7.4 and substitution of } \hat{f}^{n_0} \text{ for } \mathbf{f}^{n_0} \rangle \\
& \Sigma^n - \textstyle\bigcup_{J\in\mathcal{I}} \left(\theta_J^{-1}(\Sigma^{n_0} - \hat{f}^{n_0}(\Theta_J x)) \cup \boldsymbol{\xi}\right) \\
= \quad & \langle \text{ Definition of } f_\sigma \text{ (Page 23) } \ \& \ \text{ Definition 4.9 } \rangle \\
& \Sigma^n - \textstyle\bigcup_{J\in\mathcal{I}} \left(\{\theta_J^{-1}\sigma' \mid \sigma' \in \Sigma^{n_0} \wedge \neg \hat{f}^{n_0}_{\sigma'}(\Theta_J x)\} \cup \boldsymbol{\xi}\right) \\
= \quad & \langle \text{ Remark 4.12 } \ \& \ \text{ Changing dummy, } \sigma = \theta_J^{-1}\sigma' \Leftrightarrow \sigma' = \Theta_J\sigma \ \rangle \\
& \Sigma^n - \textstyle\bigcup_{J\in\mathcal{I}} \left(\{\sigma \mid \Theta_J\sigma \in \Sigma^{n_0} \wedge \neg \hat{f}^{n_0}_{\Theta_J\sigma}(\Theta_J x)\} \cup \boldsymbol{\xi}\right) \\
= \quad & \langle \ \Theta_J\sigma \in \Sigma^{n_0} \Leftrightarrow \Theta_J\sigma \neq \epsilon \ \rangle \\
& \Sigma^n - \textstyle\bigcup_{J\in\mathcal{I}} \left(\{\sigma \mid \Theta_J\sigma \neq \epsilon \wedge \neg \hat{f}^{n_0}_{\Theta_J\sigma}(\Theta_J x)\} \cup \boldsymbol{\xi}\right) \\
= \quad & \langle \ \Theta_J\sigma \neq \epsilon \wedge \neg \hat{f}^{n_0}_{\Theta_J\sigma}(\Theta_J x) \Rightarrow \Theta_J\sigma \neq \epsilon \wedge \Theta_J\sigma \notin \Sigma_u^{n_0} \Rightarrow \sigma \in \Sigma_c^n \ \ \& \\
& \quad \text{Replacement of } \boldsymbol{\xi} \text{ by Equation 7.5 } \ \& \ \text{ Distributivity } \rangle \\
& \Sigma^n - \big\{\sigma \mid (\exists J \mid J \in \mathcal{I} : \sigma \in \Sigma_c^n \wedge \Theta_J\sigma \neq \epsilon \wedge \neg \hat{f}^{n_0}_{\Theta_J\sigma}(\Theta_J x)) \vee \\
& \qquad (\exists J \mid J \in \mathcal{I} : \sigma \in \Sigma_c^n \wedge \Theta_J\sigma = \epsilon \wedge \\
& \qquad\qquad (\exists x' \mid x' \in X^{n_0} : M^{n_0}(\Theta_J x) = M^{n_0}(x') \wedge \\
& \qquad\qquad\qquad \neg\langle Q^{n_0}\rangle(x')))\big\} \\
= \quad & \langle \ J \text{ not free in } \sigma \in \Sigma_c^n \ \ \& \ \text{ Distributivity } \ \& \ \text{ De Morgan } \rangle \\
& \big\{\sigma \mid \sigma \notin \Sigma_c^n \vee \\
& \quad \big(\neg(\exists J \mid J \in \mathcal{I} \wedge \Theta_J\sigma \neq \epsilon : \neg \hat{f}^{n_0}_{\Theta_J\sigma}(\Theta_J x)) \wedge \\
& \quad \neg(\exists J \mid J \in \mathcal{I} \wedge \Theta_J\sigma = \epsilon : (\exists x' \mid x' \in X^{n_0} \wedge M^{n_0}(\Theta_J x) = M^{n_0}(x') : \\
& \qquad\qquad \neg\langle Q^{n_0}\rangle(x')))\big)\big\} \\
= \quad & \langle \text{ De Morgan } \ \& \ \Sigma^n = \Sigma_u^n \cup \Sigma_c^n \ \rangle \\
& \Sigma_u^n \cup \big\{\sigma \mid \sigma \in \Sigma_c^n \wedge \\
& \qquad (\forall J \mid J \in \mathcal{I} \wedge \Theta_J\sigma \neq \epsilon : \hat{f}^{n_0}_{\Theta_J\sigma}(\Theta_J x)) \wedge \\
& \qquad (\forall J \mid J \in \mathcal{I} \wedge \Theta_J\sigma = \epsilon : \\
& \qquad\qquad (\forall x' \mid x' \in X^{n_0} \wedge M^{n_0}(\Theta_J x) = M^{n_0}(x') : \langle Q^{n_0}\rangle(x')))\big\} \\
= \quad & \langle \ \hat{f}^{n_0} \text{ defined by (3.4) } \ \& \ \text{ Lemma 4.16 } \rangle \\
& \Sigma_u^n \cup \big\{\sigma \mid \sigma \in \Sigma_c^n \wedge \\
& \qquad (\forall J \mid J \in \mathcal{I} \wedge \Theta_J\sigma \neq \epsilon : \Theta_J\sigma \notin \hat{A}(\langle Q^{n_0}\rangle, \Theta_J M^n(x))) \wedge \\
& \qquad (\forall J \mid J \in \mathcal{I} \wedge \Theta_J\sigma = \epsilon : \\
& \qquad\qquad (\forall x' \mid x' \in X^{n_0} \wedge \Theta_J M^n(x) = M^{n_0}(x') : \langle Q^{n_0}\rangle(x')))\big\} \qquad \square
\end{aligned}
$$

It should be noted that, if $\hat{f}^{n_0 *}$ (the optimal SFBC function that corresponds to

$\sup \mathcal{SC}(Q^{n_0}))$ were substituted for $\mathbf{f}^{n_0}$ in Equation 7.4, then the equality should be replaced by an inclusion. In general, $\hat{f}^* \leq \hat{f}$, since $\sup \mathcal{SC}(Q) \leq \langle Q \rangle$ and $\hat{A}(Q, y)$ is antimonotone in Q.

Lemma 7.14 *Let Q^{n_0} be an instance of a parameterized predicate Q^N, $\mathcal{I} \subseteq \mathcal{J}^n_{n_0}$ and $x \in X^n$. If, in Equation 7.4, $\breve{f}^{n_0}$ (defined by (3.3)) and $\emptyset$ are substituted for $\mathbf{f}^{n_0}$ and $\boldsymbol{\xi}$, respectively, then*

$$f^n(x) = \Sigma^n_u \cup \big\{\sigma \mid \sigma \in \Sigma^n_c \wedge (\forall J \mid J \in \mathcal{I} \wedge \Theta_J\sigma \neq \epsilon : \Theta_J\sigma \notin \breve{A}(\langle Q^{n_0}\rangle, \Theta_J M^n(x)))\big\}.$$

Proof. The proof is similar to that for Lemma 7.13, but $\boldsymbol{\xi}$ is replaced by $\emptyset$ and $\breve{A}$ is used instead of $\hat{A}$. □

The next lemma characterizes the enabled events, in terms of bad event sets and projections, in the context of total observation.

Lemma 7.15 *Let Q^{n_0} be an instance of a parameterized predicate Q^N, $\mathcal{I} \subseteq \mathcal{J}^n_{n_0}$ and $x \in X^n$. If, in Equation 7.4, f^{n_0*} (defined by (3.2)) and $\emptyset$ substitute for $\mathbf{f}^{n_0}$ and $\boldsymbol{\xi}$, respectively, then*

$$f^n(x) = \Sigma^n_u \cup \big\{\sigma \mid \sigma \in \Sigma^n_c \wedge (\forall J \mid J \in \mathcal{I} \wedge \Theta_J\sigma \neq \epsilon : \Theta_J\sigma \notin A(\langle Q^{n_0}\rangle, \Theta_J x))\big\}.$$

Proof. The proof is similar to that for Lemma 7.13, but $\boldsymbol{\xi}$ is replaced by $\emptyset$ and A is used instead of $\hat{A}$. □

The following theorems establish the strong or weak soundness of the synthesis method with respect to various values of its parameters.

Theorem 7.16 *Let Q^N be a parameterized predicate that satisfies SSA for a given n_0 and let $\mathcal{I} \subseteq \mathcal{J}^n_{n_0}$. Let $\hat{f}^{n_0}$ and (7.5) substitute for $\mathbf{f}^{n_0}$ and $\boldsymbol{\xi}$, respectively, in Equation 7.4. If $\lfloor\langle Q^n\rangle\rfloor_{\mathcal{I}}(x^n_0)$ holds, then $Re(G^n|f^n) = \hat{R}(G^n, \lfloor\langle Q^n\rangle\rfloor_{\mathcal{I}})$.*

Proof.

$$\delta^{f^n}(x,\sigma)!$$

$\Leftrightarrow$ $\langle$ Definition of δ^f (Page 19) and ! (Page 19) $\rangle$

$$\sigma \in f^n(x) \wedge \delta^n(x,\sigma)!$$

$\Leftrightarrow$ $\langle$ Lemma 7.13 & Corollary 7.4 $\rangle$

$$(\sigma \in \Sigma_u^n \vee (\sigma \in \Sigma_c^n \wedge \sigma \notin \hat{A}(\lfloor\langle Q^n\rangle\rfloor_{\mathcal{I}}, M^n(x)))) \wedge \delta^n(x,\sigma)!$$

$\Leftrightarrow$ $\langle$ Definition of $\hat{A}$ (Page 21) $\rangle$

$$\sigma \notin \hat{A}(\lfloor\langle Q^n\rangle\rfloor_{\mathcal{I}}, M^n(x)) \wedge \delta^n(x,\sigma)!$$

The result then follows from Proposition 3.7(1) and the facts that $\lfloor\langle Q^n\rangle\rfloor_{\mathcal{I}}(x_0^n)$ holds and $\lfloor\langle Q^n\rangle\rfloor_{\mathcal{I}}$ is Σ_u^n-invariant (by Proposition 5.10). □

It should be noted that if $\mathcal{I} = \mathcal{J}_{n_0}^n$ then $Re(G^n|f^n) = \hat{R}(G^n, \langle Q^n\rangle) = Re(G^n|\hat{f}^n)$.

Theorem 7.17 *Let Q^N be a parameterized predicate that satisfies SSA for a given n_0 and let $\mathcal{I} \subseteq \mathcal{J}_{n_0}^n$. Let $\breve{f}^{n_0}$ and $\emptyset$ substitute for $\mathbf{f}^{n_0}$ and $\boldsymbol{\xi}$, respectively, in Equation 7.4. If $\lfloor\langle Q^n\rangle\rfloor_{\mathcal{I}}(x_0^n)$ holds, then $Re(G^n|f^n) \leq \breve{R}(G^n, \lfloor\langle Q^n\rangle\rfloor_{\mathcal{I}})$.*

Proof. The proof is similar to that for Theorem 7.16, except that Lemma 7.14, Corollary 7.6 and Proposition 3.7(2) are invoked. □

Theorem 7.18 *Let Q^N be a parameterized predicate that satisfies SSA for a given n_0 and let $\mathcal{I} \subseteq \mathcal{J}_{n_0}^n$. Let $f^{n_0 *}$ and $\emptyset$ substitute for $\mathbf{f}^{n_0}$ and $\boldsymbol{\xi}$, respectively, in Equation 7.4. If $\lfloor\langle Q^n\rangle\rfloor_{\mathcal{I}}(x_0^n)$ holds, then $Re(G^n|f^n) = R(G^n, \lfloor\langle Q^n\rangle\rfloor_{\mathcal{I}})$.*

Proof. The proof is similar to that for Theorem 7.16, except that Lemma 7.15 and Corollary 7.3 are invoked. □

The fact that the method founded on M-controllability is not strongly sound can be justified by the presence of the term $\langle Q\rangle(x')$ in Equation 3.5, which is absent in Equation 3.6. For $\breve{f}^{n_0}$, the term $\langle Q^{n_0}\rangle(\Theta_J x')$ is too conservative with respect to the corresponding term $\langle Q^n\rangle(x')$ for $\breve{f}^n$. Indeed, for a given $x' \in X^n$ such that $M^n(x) = M^n(x')$, $\langle Q^n\rangle(x')$ might not hold, while $\langle Q^{n_0}\rangle(\Theta_J x')$ might hold when $M^{n_0}(\Theta_J x) =$

$M^{n_0}(\Theta_J x')$ for a given J, as in Example 7.7 (with $x = \langle A_1, A_2, E_3 \rangle$, $x' = \langle D_1, C_2, E_3 \rangle$ and $\Theta_J x' = \langle D_1, C_2 \rangle$). When the method is founded on strong M-controllability, the deviant cases (additional events that must be prohibited) are treated by replacing $\boldsymbol{\xi}$ by Equation 7.5. The synthesis method for M-controllability could similarly be adapted to take into consideration the deviant cases (in order to remove the events that must not be prohibited). However, contrary to the evaluation of Equation 7.5 that uses only information available in the state space of dimension n_0 independently of knowledge about objects in the state space of dimension n, the identification of a state $x'' \in X^{n_0}$ for which the evaluation of $\langle Q^{n_0} \rangle(x'')$ must be viewed as false requires objects in the state space of dimension n.

So there is a choice for the on-line synthesis of an SFBC function in the case of partial observation: using $\hat{f}^{n_0}$ or $\breve{f}^{n_0}$. To distinguish between these two possibilities for f^n, the following notation is used:

$$f^n_{(7.4)\langle \hat{f}^{n_0}, \xi \rangle} \text{ for the former and } f^n_{(7.4)\langle \breve{f}^{n_0}, \emptyset \rangle} \text{ for the latter.}$$

Since the method is strongly sound for strong M-controllability, if $\mathcal{I} = \mathcal{J}^n_{n_0}$ then

$$Re(G^n | f^n_{(7.4)\langle \hat{f}^{n_0}, \xi \rangle}) = \hat{R}(G^n, \langle Q^n \rangle) = Re(G^n | \hat{f}^n).$$

However, again with $\mathcal{I} = \mathcal{J}^n_{n_0}$,

$$Re(G^n | f^n_{(7.4)\langle \breve{f}^{n_0}, \emptyset \rangle}) \leq \breve{R}(G^n, \langle Q^n \rangle) = Re(G^n | \breve{f}^n),$$

because the method is only weakly sound for M-controllability.

The predicates $Re(G^n | f^n)$ of these two SFBC functions are then incomparable in general, because, by Proposition 3.8, $Re(G^n | \hat{f}^n) \leq Re(G^n | \breve{f}^n)$. In other words, the results of a strongly sound synthesis procedure, like the one described by Equation 7.4 with $\langle \hat{f}^{n_0}, \xi \rangle$, are in accordance with those expected in the state space of dimension n and this choice can be qualified as conservative. In the absence of strong soundness for M-controllability, the use of Equation 7.4 with $\langle \breve{f}^{n_0}, \emptyset \rangle$ constitutes an optimistic choice in the sense that one would expect that the SFBC f^n would be near $\breve{f}^n$, which is more permissive than $\hat{f}^n$. This can be the case if $\langle Q^{n_0} \rangle$ is almost normal because, under the assumption that Q^{n_0} is normal and Σ_u-invariant, it can be shown that the synthesis method is strongly sound ($\lfloor Q^n \rfloor_{\mathcal{I}}$ is Σ_u-invariant by Proposition 5.10 and Remark 7.8). Finally, Theorem 7.18 is compatible with the results of Chapter 6, Section 6.2, and [Bherer et al. 2004] results dealing only with total observation and the particular case $\mathcal{I} = \mathcal{J}^n_{n_0}$; that is, $f^n = f^{n*}$ when $f^{n_0 *}$ substitutes for $\mathbf{f}^{n_0}$.

Chapter 8

Conclusion

The theoretical framework investigated in this thesis was originally stimulated by a lack of scalable synthesis methods, mainly because of the state-space explosion problem that causes considerable difficulties in the calculation of supervisors for realistic systems. It is subsumed under the conventional modular control paradigm, but specialized to systems that exhibit symmetries, for instance, a telephone system with millions of devices that behave in the same way or a reliable system with many redundant components. In this framework, a supervisor may demonstrate a form of robustness because it can dynamically react to some perturbations (addition or deletion of a process) occurring in the controlled system by taking into account the number of processes that are alive during the calculation of control actions by the underlying on-line synthesis algorithm. In the case of total observation and the case of partial observation founded on strong M-controllability, strong soundness of the synthesis method relies on the fulfillment of SSA by $\langle Q^N \rangle$, which is true if i) Q^N satisfies SSA and ii) all the events that belong to Σ_s are controllable (otherwise the method would be weakly sound). Nevertheless, the introduction of interconnection relations provides for considering predicates that do not satisfy SSA, but they must, however, be obtained from those that satisfy SSA. In the case of partial observation founded on M-controllability, it was proved that the method is only weakly sound. Other sorts of soundness could be defined in the cases for which there is a relationship between SFBC functions constructed in different ways. For instance, if $\hat{f}^{n_0*}$ is used in dimension n_0 and f^n is compared with $\hat{f}^n$ in dimension n.

Further remarks on related work

Apart from the few studies on synthesis methods for symmetric systems mentioned in the introduction, much work exploiting symmetry has been done in model checking. Most approaches suggest that a system be represented by a quotient model defined from a state equivalence relation based on symmetry. The method developed in this thesis differs from these as it uses symmetries in order to establish a small cutoff [Emerson and Kahlon 2000] for the purpose of the off-line phase. It was inspired by work on program synthesis, which details a method for constructing a program from a temporal logic specification, for a system consisting of K similar interconnected sequential processes executing in parallel, based on the calculation of a solution to a pair-system [Attie and Emerson 1998]. In this method, the interconnection relation is a symmetric binary relation and the specification language is a subset of an extension of CTL*. In particular, liveness properties cannot be expressed over a pair of processes. In addition to the use of a different paradigm (SCT) in which some events are uncontrollable and some states are unobservable, our method allows one to express safety properties with the aid of general predicates that are not limited to pair-systems (e.g., the mutual exclusion problem in which at most $p > 2$ processes can simultaneously use a resource).

In the case of total observation, a comparison with the conventional modular control approach is direct when the global specification Q is expressed as a conjunction of predicates: $Q = \bigwedge_{i=1}^{m} Q_i$, where $m = \binom{n}{n_0}$ and each Q_i represents the same local constraint, but specific to a given combination of n_0 processes. Formally, for all $x \in X^n$, $Q_i(x) \Leftrightarrow Q^{n_0}(\Theta_J x)$ for a given $J \in \mathcal{J}^n_{n_0}$. On the one hand, if $Re(G|f_i) = \sup \mathcal{CP}(Q_i)$ and the SFBC f is calculated as $f_\sigma := \bigwedge_{i=1}^{m} f_{i,\sigma}$ for all $\sigma \in \Sigma$, then $Re(G|f) = \sup \mathcal{CP}(Q)$ under the assumption that each f_i is balanced [Wonham 2008]. Thus, the method is strongly sound. The synthesis of the f_i cannot distinguish between synchronous and asynchronous events, since it is done with respect to G, in which these distinctions cannot be made. However, it was shown that our method is strongly sound only if all synchronous events are controllable. This difference is understandable by the synthesis of only one local supervisor with respect to G^{n_0} (in which only n_0 processes agree on a synchronous event) and the use of on-line renaming transformations. Within this setting and PSA, the unique supervisor does not need to be balanced for achieving optimality. On the other hand, the modular approaches, which avoid the calculation of the overall system, impose various conditions incompatible or too restrictive compared with our approach. For instance, some turn out badly if an event is shared by all processes [Queiroz and Cury 2000], some use natural projections [Komenda et al. 2005] and some take advantage of a specification defined over a subset of the system alphabet [Schmidt et al. 2006]. In the case of partial observation, unsubstantial results in modular control confine the ways of making comparisons. However, weak soundness established by Theorem 7.17 is compatible with a previous result that could achieve strong soundness to the detriment of the verification of a global condition for each instance (i.e., for each value of $n \geq n_0$) of a parameterized predicate (see Theorem 3 in [Takai et al. 1995]). Nevertheless, this is contrary to the approach developed in this thesis.

Differences between the state-feedback theory of vector DESs (VDESs) and our framework must also be highlighted. The former is useful for solving control problems for systems composed of concurrent processes when the specification is the conjunction of a finite number of linear predicates and all states of the system are observable [Li and Wonham 1993, Li and Wonham 1994]. However, the calculation of an SFBC function is only practicable under several restrictions. First, in general, the uncontrollable part of the system must be loop-free and the number of processes must be fixed in order to solve linear integer programming problems on-line (i.e., to avoid the explicit exploration of the reachability tree off-line). Second, as mentioned in the introduction, an SFBC function can be expressed in closed form (by using variables that represent an arbitrary number of processes in a specific state) under additional conditions. Such structural conditions are unnecessary in our approach. Other points must be emphasized. For VDESs, parameters are not explicitly used in the modeling of the specification, even if this possibility should not be excluded in the computation of an SFBC function in closed form. Processes in a VDES have no identity, limiting the way of considering some classes of processes unless duplicating some parts of the VDES. Finally, any conjunction of a finite number of linear predicates, $a_{i,1}x_1 + a_{i,2}x_2 + \cdots + a_{i,l}x_l \leq b_i$, $i = 1, \ldots m$, satisfies SSA with $n_0 = (\max i \mid: \lfloor \frac{b_i}{(\min j | a_{i,j} \neq 0 : a_{i,j})} \rfloor) + 1$ if $a_{i,j} \geq 0$ and $b_i \geq 0$. However, to achieve a power of expressivity comparable to that of VDESs, the definition of PDES should be modified to cope with multiple classes of similar processes and various ways of connecting them.

Overall, our approach puts together two paradigms and opens multiple research subjects within another perspective, while providing an efficient implementation of a supervisor. There is a linear gain of computational complexity with respect to the naive solution and the use of an interconnection relation is explicitly integrated into the on-line synthesis phase. To the best of our knowledge, such features have not been examined before in conventional modular approaches.

Future directions

Many important issues remain to be addressed within the reduction-parameterization paradigm. First, the scope of this thesis was limited to control problems with safety properties. Enlargement to treat liveness properties, particularly fairness properties, would require a different framework. Since liveness properties cannot be formalized with the aid of a predicate $Q \in \mathrm{Pred}(X)$, a temporal logic should be used to express such properties in conjunction with appropriate algorithms for checking the underlying assumptions [Attie and Emerson 1998]. Dynamic state feedback control, which requires memories to record history information, could be considered if a stronger notion of fairness that avoids the analysis of infinite strings [Li and Wonham 1993] is adopted. The use of abstract data types, such as queues, constitutes a good avenue [Gohari and Wonham 2005]. Second, efficient algorithms for determining if an arbitrary number of similar processes under control may be blocking (with the smallest value of the number of processes in the positive case) could fail in finite time, because of the undecidability of equivalence between a system of arbitrary size and a system of bounded size [Thistle and Nazari 2005]. This issue is presently under investigation [Bherer et al. 2006b]. The idea is to consider a replicated structure as an n–bounded state graph for a PDES with n processes and to construct its reachability graph by using rewriting rules that manipulate symbolic expressions and symbolic constraints. The power of a finite set of rewriting rules is, however, limited, especially if the application of rules is regulated by criteria that ensure that the generation of nodes progresses necessarily to a solution or until no rule can be applied. In the latter case, the algorithm fails to generate a solution. Third, the way to make SSA more flexible was to separate processes into different classes by using an interconnection relation. Relaxing assumptions (e.g., weakening SSA or allowing some shared events to be shared only by a subset of the processes, which conflicts with PSA) would then require finding appropriate types of syntactic renaming transformations. Finally, several studies could be initiated by examining other classes of control problems with various forms of symmetry within the proposed paradigm. The key to the advancement in this area will depend on solutions to the aforementioned interwoven issues.

Bibliography

[Attie and Emerson 1998] Attie, P. C. and Emerson, E. A. 1998. Synthesis of concurrent systems with many similar processes. *ACM Transactions on Programming Languages and Systems*, 20 (1): 1–65.

[Arons et al. 2001] Arons, T., Pnueli, A., Ruah, S., Xu, Y., and Zuck, L. 2001. Parameterized verification with automatically computed inductive assertions. In G. Berry, H. Comon, and A. Finkel (eds.), *Computer aided Verification*, Vol. 2102 of Lecture Notes in Computer Science, 221–234, Springer.

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

[Barbeau et al. 1997] Barbeau, M., Kabanza, F., and St-Denis, R. 1997. An efficient algorithm for controller synthesis under full observation. *Journal of Algorithms*, 25 (1): 144–161.

[Barrett and Lafortune 1998] Barrett, G. and Lafortune, S. 1998. Bisimulation, the supervisory control problem and strong model matching for finite state machines. *Discrete Event Dynamic Systems: Theory and Applications*, 8 (4): 377–429.

[Ben Hadj-Alouane et al. 1994] Ben Hadj-Alouane, N., Lafortune, S., and Lin F. 1994. Variable lookahead supervisory control with state information. *IEEE Transactions on Automatic Control*, 39 (12): 2398–2410.

[Ben Hadj-Alouane et al. 1996] Ben Hadj-Alouane, N., Lafortune, S., and Lin F. 1996. Centralized and distributed algorithms for on-line synthesis of maximal control policies under partial observation. *Discrete Event Dynamic Systems: Theory and Applications*, 6 (4): 379–427.

[Bherer et al. 2003] Bherer, H., Desharnais, J., Frappier, M., and St-Denis, R. 2003. Intégration d'une technique de vérification dans une procédure de synthèse de contrôleurs de systèmes paramétrés. In D. Méry, N. Rezg, and X. Xie (eds.), *Modélisation des systèmes réactifs (MSR 2003)*, pp. 553–566.

[Bherer et al. 2004] Bherer, H., Desharnais, J., Frappier, M., and St-Denis, R. 2004. Synthesis of state feedback controllers for parameterized discrete event systems. In F. Wang (ed.), *Automated Technology for Verification and Analysis (ATVA'2004)*, Vol. 3299 of Lecture Notes in Computer Science, pp. 487–490.

[Bherer et al. 2005] Bherer, H., Desharnais, J., and St-Denis, R. 2005. Synthesis of state feedback controllers for parameterized discrete event systems under partial observation. *Proceedings of the 44th IEEE Conference on Decision and Control and European Control Conference 2005*, Seville, pp. 3499–3506.

[Bherer et al. 2006a] Bherer, H., Desharnais, J., and St-Denis, R. 2006a. Parameterized discrete event systems under partial observation revisited. *Proceedings of the 8th IASTED International Conference on Control and Applications*, Montréal, pp. 273–280.

[Bherer et al. 2006b] Bherer, H., Desharnais, J., and St-Denis, R. 2006b. On the reachability and nonblocking properties for parameterized discrete event systems. *Proceedings of the 8th International Workshop on Discrete Event Systems*, Ann Arbor, MI, pp. 113–118.

[Bherer et al. 2008] Bherer, H., Desharnais, J., and St-Denis, R. 2008. Control of parameterized discrete event systems. *Discrete Event Dynamic Systems: Theory and Applications*, 19 (2): 213–265.

[Cassandras and Lafortune 1999] Cassandras, C. G. and Lafortune, S. 1999. *Introduction to Discrete Event Systems*, Boston, MA: Kluwer.

[Chung et al. 1992] Chung, S.-L., Lafortune, S., and Lin, F. 1992. Limited lookahead policies in supervisory control of discrete event systems. *IEEE Transactions on Automatic Control*, 37 (12): 1921–1935.

[Davey and Priestley 1990] Davey, B. A. and Priestley, H. A. 1990. *Introduction to Lattices and Order*. Cambridge, UK: Cambridge University Press.

[Dijkstra 1975] Dijkstra, E. W. 1975. Guarded commands, nondeterminacy and formal derivation of program. *Communications of the ACM*, 18 (8): 453–457.

[Dijkstra 1976] Dijkstra, E. W. 1976. *A Discipline of Programming*. Prentice-Hall.

[Dreschsler and Sieling 2001] Dreschsler, R. and Sieling, D. 2001. Binary decision diagrams in theory and practice. *International Journal on Software Tools for Technology Transfer*, 3 (2): 112–136.

[Emerson and Kahlon 2000] Emerson, E. A. and Kahlon, V. 2000. Reducing model checking of the many to the few. In D. A. McAllester (ed.), *Automated Deduction (CADE'2000)*, Vol. 1831 of Lecture Notes in Computer Science, pp. 236–354.

[Emerson and Sistla 1997] Emerson, E. A. and Sistla, A. P. 1997. Utilizing symmetry when model-checking under fairness assumptions: an automata-theoretic approach. *ACM Transactions on Programming Languages and Systems*, 19 (4): 617–638.

[Eyzell and Cury 2001] Eyzell, J. M. and Cury, J. E. R. 2001. Exploiting symmetry in the synthesis of supervisors for discrete event systems. *IEEE Transactions on Automatic Control*, 46 (9): 1500–1505.

[Flordal et al. 2007] Flordal, H., Malik R., Fabian M., and Akesson K. 2007. Compositional synthesis of maximally permissive supervisors using supervision equivalence. *Discrete Event Dynamic Systems: Theory and Applications*, 17 (4): 475–504.

[Frappier and St-Denis 2001] Frappier, M. and St-Denis, R. 2001. Towards a computer-aided design of reactive systems. In R. Moreno-Díaz, B. Buchberger, and J.-L. Freire (eds.), *Computer Aided Systems Theory – EUROCAST 2001*, Vol. 2178 of Lecture Notes in Computer Science, pp. 421–436.

[Gaudin and Marchand 2007] Gaudin B. and Marchand H. 2007. An efficient modular method for the control of concurrent discrete event systems: A language-based approach. *Discrete Event Dynamic Systems: Theory and Applications*, 17 (2): 179–209.

[Gohari and Wonham 2005] Gohari, P. and Wonham, W. M. 2005. Efficient implementation of fairness in discrete-event systems using queues. *IEEE Transactions on Automatic Control*, 50 (11): 1845–1849.

[Gries and Schneider 1995] Gries, D. and Schneider, F. B. 1995. *A Logical Approach to Discrete Math*. New-York: Springer-Verlag.

[Giua and DiCesare 1994] Giua, A. and DiCesare F. 1994. Petri net structural analysis for supervisory control. *IEEE Transactions on Robotics and Automation*, 10 (2): 185–195.

[Harel 1987] Harel, D. 1987. Statecharts: a visual formalism for complex systems. *Science of Computer Programming*, 8 (3): 231–274.

[Heymann and Lin 1994] Heymann, M. and Lin, F. 1994. On-line control of partially observed discrete event systems. *Discrete Event Dynamic Systems: Theory and Applications*, 4 (3): 221–236.

[Holloway et al. 1997] Holloway, L. E., Krogh, B. H., and Giua, A. 1997. A survey of Petri net methods for controlled discrete event systems. *Discrete Event Dynamic Systems: Theory and Applications*, 7 (2): 151–190.

[Kerjean et al. 2006] Kerjean, S., Kabanza, F., St-Denis, R., and Thiébaux, S. 2006. Analyzing LTL model checking techniques for plan synthesis and controller synthesis. *Electronic Notes in Theoretical Computer Science*, 149 (2): 91–104.

[Kesten et al. 2002] Kesten, Y., Pnueli, A., Shahar, E., and Zuck, L. 2002. Network invariants in action. In L. Brim, P. Jancar, M. Kretinsky, and A. Kucera (eds.), CONCUR 2002, *Concurrency Theory*, Vol. 2421 of Lecture Notes in Computer Science, pp. 101–115, Springer.

[Komenda and van Schuppen 2005] Komenda, J. and van Schuppen, J. H. 2005. Supremal sublanguages of general specification languages arising in modular control of dicrete-event systems. *Proceedings of the 44th IEEE Conference on Decision and Control and European Control Conference 2005*, Seville, pp. 2275–2780.

[Komenda et al. 2005] Komenda, J., van Schuppen, J. H., Gaudin, B., and Marchand, H. 2005. Modular supervisory control with general indecomposable specification languages. *Proceedings of the 44th IEEE Conference on Decision and Control and European Control Conference 2005*, Seville, pp. 3474–3479.

[Kumar and Garg 1995] Kumar, R. and Garg, V. K. 1995. *Modeling and Control of Logical Discrete Event Systems*, Boston, MA: Kluwer.

[Kumar et al. 1993] Kumar, R., Garg, V., and Marcus, S. I. 1993. Predicates and predicate transformers for supervisory control of discrete event dynamical systems. *IEEE Transactions on Automatic Control*, 38 (2): 232–247.

[Leduc et al. 2005] Leduc, R. J., Brandin, B. A., Lawford, M., and Wonham, W. M. 2005. Hierarchical interface-based supervisory control—part I: serial case. *IEEE Transactions on Automatic Control*, 50 (9): 1322–1335.

[Li 1991] Li, Y. 1991. Control of vector discrete-event systems. Ph.D. thesis, University of Toronto.

[Li and Wonham 1988] Li, Y. and Wonham, W. M. 1988. Controllability and observability in the state-feedback control of discrete-event systems. *Proceedings of 27th IEEE Conference on Decision and Control*, Austin, TX, pp. 203–208.

[Li and Wonham 1993] Li, Y. and Wonham, W. M. 1993. Control of vector discrete-event systems I—the base model. *IEEE Transactions on Automatic Control*, 38 (8): 1214–1227.

[Li and Wonham 1994] Li, Y. and Wonham, W. M. 1994. Control of vector discrete-event systems II—controller synthesis. *IEEE Transactions on Automatic Control*, 39 (3): 512–531.

[Lin and Wonham 1988] Lin, F. and Wonham, W. M. 1988. On observability of discrete event systems. *Information Sciences*, 44 (3): 173–198.

[Ma and Wonham 2005] Ma, C. and Wonham, W. M. 2005. *Nonblocking Supervisory Control of State Tree Structures*. Vol. 317 of Lecture Notes in Control and Information Sciences. Berlin, Germany: Springer.

[Makungu et al. 1999] Makungu, M., Barbeau, M., and St-Denis, R. 1999. Synthesis of controllers of processes modeled as colored Petri nets. *Discrete Event Dynamic Systems: Theory and Applications*, 9 (2): 147–169.

[Manna and Pnueli 1995] Manna, Z. and Pnueli, A. 1995. *Temporal Verification of Reactive Systems : Safety*, New-York: Springer-Verlag.

[Pena et al. 2006] Pena, P. N., Cury, J. E. R., and Lafortune, S. 2006. Testing modularity of local supervisors: an approach based on abstractions. *Proceedings of the 8th International Workshop on Discrete Event Systems*, Ann Arbor, MI, pp. 107–112.

[Pnueli et al. 2001] Pnueli, A., Ruah, S., and Zuck, L. 2001. Automatic deductive verification with invisible invariants. In T. Margaria and W. Yi (eds.), *Tools and Algorithms for the Construction and Analysis of Systems*, Vol. 2031 of Lecture Notes in Computer Science, pp. 82–97.

[Prosser et al. 1998] Prosser, J. H., Kam, M., and Kwatny, H. G. 1998. Online supervisor synthesis for partially observed discrete-event systems. *IEEE Transactions on Automatic Control*, 43 (11): 1630–1634.

[Queiroz and Cury 2000] Queiroz, M. H. de and Cury, J. E. R. 2000. Modular supervisory control of large scale discrete event systems. In R. Boel and G. Stremersch (eds.), *Discrete Event Systems: Analysis and Control*, Vol. 569 of the International Series in Engineering and Computer Science, pp. 103–110.

[Ramadge and Wonham 1987] Ramadge, P. J. G. and Wonham, W. M. 1987. Modular feedback logic for discrete event systems. *SIAM Journal on Control and Optimization*, 25 (5): 1202–1218.

[Schmidt et al. 2006] Schmidt, K., Marchand, H., and Gaudin, B. 2006. Modular and decentralized supervisory control of concurrent discrete event systems using reduced system models. *Proceedings of the 8th International Workshop on Discrete Event Systems*, Ann Arbor, MI, pp. 149–154.

[Song and Leduc 2006] Song, R. and Leduc, R. J. 2006. Symbolic synthesis and verification of hierarchical interface-based supervisory control. *Proceedings of the 8th International Workshop on Discrete Event Systems*, Ann Arbor, MI, pp. 419–426.

[St-Denis 2002] St-Denis, R. 2002. Designing reactive systems: integration of abstraction techniques into a synthesis procedure. *The Journal of Systems and Software*, 60 (2): 103–112.

[Takai and Kodama 1997] Takai, S. and Kodama, S. 1997. M-controllable subpredicates arising in state feedback control of discrete event systems. *International Journal of Control*, 67 (4): 553–566.

[Takai and Kodama 1998] Takai, S. and Kodama, S. 1998. Characterization of all M-controllable subpredicates of a given predicate. *International Journal of Control*, 70 (4): 541–549.

[Takai et al. 1995] Takai, S., Ushio, T., and Kodama, S. 1995. Static-state feedback control of discrete-event systems under partial observation. *IEEE Transactions on Automatic Control*, 40 (11): 1950–1954.

[Thistle and Nazari 2005] Thistle, J. G. and Nazari, S. 2005. Analysis of arbitrarily large networks of discrete-event systems. *Proceedings of the 44th IEEE Conference on Decision and Control and European Control Conference 2005*, Seville, pp. 3468–3473.

[Whittaker and Rudie 2008] Whittaker S. J. and Rudie K. 2008. Lose Fat, Not Muscle: An examination of supervisor reduction in discrete-event systems. *Discrete Event Dynamic Systems: Theory and Applications*, 18 (3): 285–321.

[Wonham 2008] Wonham, W. M. 2008. Supervisory control of discrete-event systems. ECE 1636F/1637S, Systems Control Group, University of Toronto.

[Wonham and Ramadge 1988] Wonham, W. M. and Ramadge, P. J. G. 1988. Modular supervisory control of discrete event systems. *Mathematics of Control, Signals, and Systems*, 1 (1): 13–30.

[Zhong and Wonham 1990] Zhong, H. and Wonham, W. M. 1990. On the consistency of hierarchical supervision in discrete-event systems. *IEEE Transactions on Automatic Control*, 35 (10): 1125–1134.

Index