Discrete-event system abstractions for on-line logical and real-time hierarchical control

Ngo, Quang Ha

2016

Ngo, Q. H. (2016). Discrete-event system abstractions for on-line logical and real-time hierarchical control. Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/69228>

<https://doi.org/10.32657/10356/69228>

NANYANG
TECHNOLOGICAL
UNIVERSITY

**DISCRETE-EVENT SYSTEM ABSTRACTIONS FOR
ON-LINE LOGICAL AND REAL-TIME
HIERARCHICAL CONTROL**

NGO QUANG HA
SCHOOL OF COMPUTER SCIENCE AND ENGINEERING
2016

**DISCRETE-EVENT SYSTEM ABSTRACTIONS FOR
ON-LINE LOGICAL AND REAL-TIME
HIERARCHICAL CONTROL**

NGO QUANG HA

School of Computer Science and Engineering

A thesis submitted to the Nanyang Technological University
in partial fulfilment of the requirement for the degree of
Doctor of Philosophy

2016

Dedication

This thesis is dedicated to my family.

Acknowledgements

First and foremost, I would like to express my deepest gratitude to my research advisor, Professor Seow Kiam Tian, for the continuous support of my Ph.D study and research, and for his patience, motivation, and immense knowledge. His guidance helped me at every stage of the research right up to the writing of this thesis. I would like to thank Professor Seow for his continuous encouragement and for his training that has enabled me to think like a research scientist. His advice on both research as well as my career has been priceless.

I would also like to thank Professor Deepu Rajan for co-supervising me in the last two years. Without his support, I would not have been able to continue my research.

I would like to thank Professor Miao Chunyan, Director of Lily Lab, lab-technicians and lab-mates for the opportunity to carry out my research in Lily. Lily is such a great place where I can always find friendship and support.

I would like to thank Dr. Pham Manh Tung and Dr. Amrith Dhananjayan for discussions during tea-breaks and for their quick responses whenever I raised a question.

I gratefully acknowledge the School of Computer Science and Engineering, Nanyang Technological University for providing me with financial support and all the necessary facilities to work on my research.

Last but not least, I would also like to thank my parents for their love and support, and my wife and daughter for their encouragement all along.

Ngo Quang Ha
Singapore, 2016

Table of Contents

Acknowledgements	iii
List of Figures	ix
List of Tables	x
List of Abbreviations	xi
Summary	xiii
1 Introduction	1
1.1 Overview of Discrete-Event Systems (DES's)	2
1.1.1 DES Model	2
1.1.2 Supervisory Control	3
1.1.3 Hierarchical Control	6
1.2 Problems and Motivation	7
1.2.1 Problem 1: On-line Command and Control of Logical DES's	7
1.2.2 Problem 2: Hierarchical Consistency of Timed DES's (TDES's) . .	8
1.3 Natural Output Control Structure	10
1.4 Main Contributions	11
1.4.1 On-line Command and Control for Logical DES's	11
1.4.2 Hierarchical Consistency with Output-Time Fidelity for TDES's . .	11
1.5 Thesis Organization	12

2	Background	14
2.1	Languages and Automata	14
2.2	Cartesian and Synchronous Products	15
2.3	Preliminaries on Hierarchical Control of DES's	17
2.3.1	Supervisory Control of DES's	17
2.3.2	Moore Automaton for Low-Level DES Modeling	19
2.4	Preliminaries on Hierarchical Control of TDES's	21
2.4.1	TDES Model	21
2.4.2	TDES Composition	23
2.4.3	Control Theoretic Setting for TDES's	24
2.4.4	Time Fidelity of TDES Model	26
2.4.5	Supervisory Control of TDES's	28
2.5	Moore Reachability Tree for Conceptual Computation	31
3	System Abstraction for On-line Logical Hierarchical Control	32
3.1	Hierarchical Control of DES's: A State of the Art	33
3.2	Towards On-line Command and Control: A Motivation	40
3.3	System-Output Control with Command Input	41
3.3.1	Preliminaries	41
3.3.2	Problem Approach: Uncontrollable and Controllable States	42
3.3.3	Output Control Feasibility	45
3.3.4	The Decomposition Theorem	46
3.4	Command and Control Operational Design	49
3.4.1	The Implementability Theorem	49
3.4.2	Algorithmic Procedures	53
3.4.3	Computational Complexity	54
3.5	Conclusion	61

4 System Abstraction for Real-Time Hierarchical Control:

Consistency Formulation	62
4.1 Control of TDES's: A State of the Art	63
4.2 Towards a TDES Hierarchy with Time Fidelity	64
4.2.1 Low-level TDES Modeling for Hierarchical Control	65
4.2.2 System Abstraction: Need for Output-Time Fidelity	70
4.3 Timed Control Structure	73
4.3.1 Activity Output-Control Consistency	75
4.3.2 Output-Force Consistency	76
4.3.3 Output Time-Compliance	78
4.3.4 Timed Output-Control Consistency	79
4.4 Hierarchical Consistency: Theoretical Conditions	79
4.4.1 Consistency for Hierarchical Control	80
4.4.2 Vocal-State Partnership and Strictness of Output-Control Consistency	81
4.4.3 Hierarchical Consistency Theorem	85
4.4.4 An Illustrative Example: Hierarchical Mission Control of a Robotic Camera System	88
4.5 Conclusion	91

5 System Abstraction for Real-Time Hierarchical Control:

Consistency Synthesis	92
5.1 String-wise Control Partitions of Outputs	94
5.2 OCC-System Refinability and Refinement	94
5.3 PF-System Refinement and SOCC-System Refinability	100
5.4 SOCC-System Synthesis Theorems	103
5.5 STOCC-System Synthesis: A Discussion	107
5.6 Hierarchical Consistency for NTU/NTI Systems	107
5.6.1 SOCC-System Synthesis for NTU/NTI Systems	109

5.6.2	STOCC-System Synthesis for Linear NTI Systems	110
5.7	Hierarchical Control of a Photocopying System - a Linear NTI System . . .	122
5.7.1	System Description	122
5.7.2	Initial Hierarchical System Design	123
5.7.3	Linear NTI System - A Verification	126
5.7.4	SOCC-System Synthesis and High-Level Time Fidelity Issues . . .	128
5.7.5	STOCC-System Synthesis	130
5.7.6	Hierarchical Control Specification	132
5.8	Framework Generalization and Scalability	132
5.9	Conclusion	133
6	Conclusion & Future Research	134
6.1	System Abstraction for Logical Hierarchical Control	134
6.2	System Abstraction for Real-Time Hierarchical Control	135
	Author's Publications	138
	Bibliography	139

List of Figures

1.1	A simple DES model of a light bulb system	3
1.2	Supervisory Control of a DES	4
1.3	A DES model of two light bulb subsystems and a supervisor	4
1.4	The command and control concept for hierarchical control [1]	6
3.1	Operational designs for hierarchical control	41
3.2	Feedback control of a DES with command input and system output	42
3.3	Conceptual decomposition of a DES	47
3.4	An example OCC DES G_{lo} and its τ -uncontrollable state sets, $\tau \in T_c$	49
3.5	Procedural realization for high-level controller C_{hi}	53
3.6	Procedural realization for low-level controller C_{lo}	54
3.7	Procedural realization for computing B_u^r for $\tau \in T_c$	55
4.1	Implications on the timing semantics of a high-level specification on system abstractions without and with high-level time fidelity: An illustration	72
4.2	Vocal-state partners q_1 and q_2 of a TDES G_{lo}	82
4.3	Hierarchical system concepts and their constituent relationships for a Moore TDES model G_{lo}	84
4.4	Camera system: Moore ATG model and associated event timings	88
4.5	Camera system: A constructed hierarchy (G_{lo}, G_{hi})	90
5.1	An example showing that partnership removal can introduce new partnership	101

5.2	Photocopying machine models for hierarchical control system design	125
5.3	A hierarchical photocopying system (G_{lo}, G_{hi}) that is HC	129
5.4	A hierarchical photocopying system (G_{lo}, G_{hi}) that is HC-OTF	131
5.5	A (high-level) specification TTG for the model G_{hi} in Figure 5.4(a)	132

List of Tables

3.1	Control operational design of a high-level regular specification language	
	$E \subseteq T^*$ for a HCN pair (G_{lo}, G_{hi}) : A time complexity comparison	59
4.1	Symbol definitions for a robotic camera system hierarchy (G_{lo}, G_{hi})	89
5.1	Symbol definitions of events and outputs for the photocopying system	123

List of Abbreviations

ALF Activity-Loop Freeness

AOCC Activity Output-Control Consistent

ATG Activity Transition Graph

BDD Binary Decision Diagram

CPS Cyber-Physical System

DES Discrete-Event System

HC Hierarchical Consistency / Hierarchically Consistent

HCM Hierarchical Consistency with Marking

HCN Hierarchical Nonblockingness

HC-OTF Hierarchically Consistent with Output-Time Fidelity

NTI Non-terminal Time-Control Invariant

NTU Next-output Terminal-control Unambiguous

OCC Output-Control Consistent

OFC Output-Force Consistent

OTC Output Time-Compliant

PF Partner-Free

PHF Partner-Hiding-State Free

SOCC Strictly Output-Control Consistent

STOCC Strictly Timed Output-Control Consistent

TDES Timed Discrete-Event System

TOCC Timed Output-Control Consistent

TP Time-Progressivity

TTG Timed Transition Graph

Summary

This thesis addresses two abstraction problems for two-level hierarchical control of discrete-event systems (DES's). DES's are dynamic systems driven by asynchronous events evolving through discrete states. The first problem is state abstractions of controllable outputs for realizing hierarchical control of logical DES's as on-line command and control. On-line command and control may be the runtime operational mode for some engineering applications, or required when subjecting a control solution to validation tests by simulation and observation. The problem is addressed by formulating, for a reference output set, a new abstraction concept for output control feasibility of a DES modeled by a Moore automaton, and a system decomposition of a Moore automaton with a natural output control structure into a controllable subsystem and an uncontrollable subsystem. These results are then used to implement the new command and control design, which is shown to offer the intrinsic merit of furnishing causal clarity of command over control during on-line operation, along with off-line complexity reduction and fast on-line computation, and can therefore more efficiently and effectively support the intended application use.

The second problem addressed is the abstraction of system for hierarchical consistency of timed DES's (TDES's) with output-time fidelity. Characterization of system output-time fidelity is essential to furnish a sound real-time high-level specification design foundation for hierarchical control. The problem is addressed by unearthing the fundamental properties of preemptability and fidelity of system atomic time modeled by a special tick event, and formulating the necessary real-time concepts for hierarchical consistency of TDES's - without and with output-time fidelity guarantee. In terms of the concepts developed, results

for the structural existence and synthesis of the sufficiency structure for hierarchical consistency with no output-time fidelity guarantee are developed. It is then shown that a class of Moore TDES's exists whose structure can be refined to achieve hierarchical consistency with output-time fidelity, and is a linear subclass of a mildly restrictive class of TDES's whose structure can be refined to achieve hierarchical consistency with no output-time fidelity guarantee.

Chapter 1

Introduction

Discrete-event systems (DES's) are dynamic systems driven by asynchronous qualitative changes called events evolving through discrete states over time [2]. In developing a control theory for DES's, the core issues have centered around the modeling problem of DES's in either a logical (e.g., [3, 4, 5]), real-time (e.g., [6, 7, 8, 9]) or stochastic (e.g., [10, 11]) framework, and the associated synthesis problem of supervisory controls to comply with specifications of behavioral constraints under various event-control settings and control architectures. These issues have included correct solution existence, optimality or maximal permissiveness, complexity and transparency. The research in this field [4, 12, 13, 14] uses different mathematical formalisms, methodologies and tools from control, computer science and operations research, and is motivated by the needs of many different engineering domains such as cybersecurity [15, 16, 17, 18], cyber-physical systems [19, 20], manufacturing [21, 22, 23, 24], software engineering [25, 26, 27], transportation [28, 29, 30, 31, 32] and power distribution [33, 34]. Although DES's has become a relatively mature field now since its founding in the 1980's [3, 35], new challenging problems are continually being motivated, for which research efforts are needed to further develop and extend the science of supervisory control to address these challenges.

In this thesis, we motivate and address two new problems in hierarchical control. The

first addresses on-line command and control of logical DES's, while the second extends hierarchical control to real-time DES's. The research is anchored in the control framework of formal languages and finite automata [12].

Before stating and motivating the problems, and outlining the contributions and organization of the thesis, we provide an informal background on the basic ideas of supervisory control [35, 3] and hierarchical control [1] for DES's.

1.1 Overview of Discrete-Event Systems (DES's)

1.1.1 DES Model

In [35, 3] and our research, DES's are modeled by finite-state machines also called finite automata [3]. An example of a simple DES model is depicted in Figure 1.1. This DES automaton represents the operational dynamics of switching a light bulb, with two states denoting lights *off* and *on*, and two switching events denoting switch *turnedon* and *turnedoff*. In this example, we assume that the bulb is initially lights *off*. The initial state *off*, identified by an entering arrow, is also graphically bolded or darkened to distinguish it as a marked state, representing the completion of some task in general. In this state, if event *turnedon* occurs, i.e., the switch is turned to 'on'-position, the light bulb transits to state *on* and the light comes on. It transits back to state *off* and the light goes off, if event *turnedoff* occurs, i.e., the switch is turned to 'off'-position. The DES generates sequences of events called strings. A set of strings is called a language. The set of all strings that start from the initial state of the DES is called the prefix-closed language, and those that end in a marked state is called the marked language. In general, a DES can have more than one marked state.

Figure 1.1: A simple DES model of a light bulb system

1.1.2 Supervisory Control

The fundamental problem in DES research is how to control a logical or untimed DES to satisfy some ‘closed’ behavioral or language requirements. This is a well-known supervisory control problem [3]. In [3] and our research, a requirement, usually first expressed in a natural language, is translated and formally prescribed as a specification in a formal language (of events). It is also assumed that the event set of a logical DES contains two exclusive types of events, whose characteristics are known in advance as either controllable or uncontrollable. Controllable events are those that can be prevented from occurring by event disablement, while uncontrollable events are those that cannot be prevented or disabled. In this setting, the key control-theoretic idea is that of a controllable specification for a given DES.

Informally, a specification language is said to be controllable [35] if the DES never slips out of the prefix-closure of the specification language on an uncontrollable event. A maximally permissive supervisor or controller can be constructed to implement a controllable language by disabling appropriate controllable events at each state of the DES. In a feedback configuration, the supervisor and the DES form a closed-loop system as depicted in Figure 1.2.

Consider an example of a logical DES model representing the dynamics of two light bulb subsystems of the kind depicted in Figure 1.1, as shown in Figure 1.3(a). In forming the DES model, the two subsystems are synchronized by composing states and interleaving events between the subsystems, and the (labels of) states and events belonging to each subsystem are subscripted with the respective subsystem identification number $i \in \{1, 2\}$.

Figure 1.2: Supervisory Control of a DES

Suppose $turnedon_i$'s are controllable and $turnedoff_i$'s are uncontrollable; and we consider the following requirement: 'At any point in time, at most one light bulb is turned on'. Then the corresponding supervisor is the automaton model shown in Figure 1.3(b); essentially, this model generates the controllable specification language that formalizes the requirement. Intuitively, the supervisor, following the occurrence of $turnedon_1$ ($turnedon_2$, respectively), disables $turnedon_2$ ($turnedon_1$, respectively) and does so until $turnedoff_1$ ($turnedoff_2$, respectively) occurs.

Figure 1.3: A DES model of two light bulb subsystems and a supervisor

Note that in this example, the controllable specification language is a sublanguage (or a subset) of the DES marked language. Under this condition, the corresponding supervisor is said to be nonblocking [35], in that it always permits the DES to enter any state in its marked state set, of which this example DES has one, thereby completing its task within the prefix-closure of the (controllable) specification.

Evolving from the fundamental control problem discussed, research in supervisory control has formulated and addressed a variety of problems in supervisory control under the

formal languages and automata framework. A literature sampling of major research efforts includes:

- Nonblocking control [36, 37, 38],
- Control under partial observation [39, 40] where some events are not observable,
- Robust and adaptive control [41, 42] for DES's with model uncertainty, without and with uncertainty resolution, respectively,
- Robust nonblocking control under partial observation [43],
- Control of infinite behaviors [44],
- Control with limited look-ahead capability [45, 46, 47] and on-line control [48] that reduce off-line synthesis complexity,
- Robust control with limited look-ahead [49],
- Specification automaton transparency for validation [50] and translation [51] from a class of linear time temporal logic that supports writing more readable control requirements.

In providing structured approaches to improve design clarity and lower the complexity of control, various control architectures have been motivated and studied. They include:

- Modular or decentralized control [52, 53, 54, 55, 56, 57, 58, 59],
- Decentralized control under partial observation [60],
- Multiagent coordination [61, 62, 63, 64] and a closely related idea of localized or distributed control [65],
- Distributed control with communication delay [66], as well as
- Hierarchical control [1, 67] and the related paradigm of organizational control [68].

This thesis is concerned with research in hierarchical control of DES's.

1.1.3 Hierarchical Control

Hierarchical control is one of the major control architectures of interest in supervisory control theory. The hierarchical control setup has been conceptualized as a command and control structure in [1]. In that structure, the fundamental hierarchy proposed consists of the following components: a low-level DES, a low-level supervisor (controller), a high-level DES, and a high-level supervisor (controller), together with necessary control, feedback and communication channels as shown in Figure 1.4.

Figure 1.4: The command and control concept for hierarchical control [1]

In a standard feedback fashion, the low-level DES G_{lo} is a real system to be supervised by the low-level controller C_{lo} , and the high-level DES G_{hi} is an abstract and simplified model to be supervised by the high-level controller C_{hi} . This is effected via the respective control channels, Con_{hi} and Con_{lo} , and the information feedback channels, Inf_{hi} and Inf_{lo} . The two levels are interconnected via the top-down command channel Com_{hilo} and the bottom-up information channel Inf_{lohi} . The controller C_{hi} is said to be virtual since G_{hi} is an abstract model driven by G_{lo} via the channel Inf_{lohi} ; and it summarizes the behavior of G_{lo} that is important. Significant information from G_{lo} is reported to G_{hi} via the channel Inf_{lohi} , and in turn is fed back to C_{hi} via the channel Inf_{hi} . In response, the virtual control of C_{hi} on G_{hi} , via the channel Con_{hi} , is sent to C_{lo} as a command via the channel Com_{hilo} , which in turn controls G_{lo} via the channel Con_{lo} as commanded.

In general, the high-level DES G_{hi} is due to abstraction by a natural [69, 70] or a virtual [1, 12] projection on the prefix-closed language of the low-level DES G_{lo} , which is a Moore automaton [71] constructed to implement the hierarchy, based on a given DES and the projection. The projection is said to model the information channel Inf_{lohi} .

In the setup described, the structural conditions for hierarchical consistency [1] and that with marking [67, 72], to be met by a suitably formulated hierarchical information map for a properly structured DES, have been developed. Intuitively, achieving hierarchical consistency means that a specification task of the high-level controller can be realized supremally through low-level control, whereas achieving hierarchical consistency with marking means that it can be realized supremally with nonblockingness through low-level nonblocking control. Note that if the states of the low-level DES are all marked, then for a controllable specification task of the high-level controller, the two versions of hierarchical consistency are equivalent.

Notable efforts have been made to generalize and extend the basis concept of a two-level hierarchy to support a multi-level hierarchy (e.g., [73, 68]).

1.2 Problems and Motivation

In this thesis, motivated by the current state-of-the-art research and emerging real world applications, we identify and address two new design problems, outlined and motivated in this section. The research problems are investigated within the control-theoretic framework of formal languages and finite automata for both logical DES's [35, 3] and timed DES's (TDES's) [74].

1.2.1 Problem 1: On-line Command and Control of Logical DES's

Based on the general architectural setting reviewed and depicted in Figure 1.4, the first problem addressed is state abstractions of controllable outputs for realizing hierarchical control of logical DES's concretely as on-line command and control. This new problem

is motivated by the fact that on-line command and control may be the runtime operational mode for many engineering control applications, or it is required when subjecting a control solution to validation tests by simulation and observation.

In the current literature, under hierarchical consistency or hierarchical consistency with marking, the existing operational design is a flat low-level controller [12, 1, 67, 75, 72] synthesized from a base-level specification converted from a given high-level specification. Besides incurring high complexity of control synthesis, this resultant flat controller is in essence an off-line consolidation that ‘collapses’ all the regulated commands at the high level into possible controls for the real DES. Using this low-level controller, the command and control structure as conceptualized in Figure 1.4 is lost in implementation.

1.2.2 Problem 2: Hierarchical Consistency of Timed DES’s (TDES’s)

The second problem addressed is the abstraction of system for hierarchical consistency of TDES’s with output-time fidelity. Based on the same two-level setup as the hierarchical control of untimed DES’s [1], the setup for TDES’s has the low-level system modeled by a Moore automaton, but that is constructed from the TDES model proposed in [74], and a timed formulation of a hierarchical reporter map modeling the information channel via which the low level drives the high level. This TDES model is a timed transition graph formulated to possess sound system (event-) timing semantics implicitly founded in [74] but formally and explicitly elucidated in our research. In this system model, time is represented by a special event denoting an atomic tick of the global clock. By sound system timing semantics, we refer to time fidelity of the model, characterizing time progression as unstoppable and never halting an executing activity event.

For control purposes, the events other than the tick event of the adopted TDES model are also known in advance as exclusively forcible or non-forcible, such that an enforced forcible event can preempt the tick event. In this real-time control setting, informally, a controllable specification language extends to one for which, within the prefix-closure of the specification language, the TDES never slips out on an uncontrollable event, and any

tick event it slips out on can be preempted without the TDES slipping out as a result. Using this concept, the synthesis theory of nonblocking control [74] parallel to the untimed version [35] has been developed.

From the design perspective, the usual purpose of the resultant high-level system in a hierarchical control setting is to provide significant and unambiguous control information of an engineering application of interest at a suitable level of abstraction where the designer better understands the application (such as, e.g., robotic object picking and placing), for clearer specification design from above for control synthesis based on the timed controllability concept [74] informally explained above, without the need to delve into detailed information of the low-level system (such as, e.g., robotic arm extension and contraction). This has been a compelling motivation for hierarchical control design [1]. Generalizing to real time, the high-level model abstraction, in our opinion, might aggregate time but must also preserve time fidelity as with the low-level system model. Without high-level or output-time fidelity, designers would need to go beneath the abstraction level, increasing specification design complexity and effort as a result just to ensure that desired timing requirements are correctly specified.

The need for physical time fidelity in system abstraction is motivated in part by real-time design challenges in cyber-physical systems (CPS's) [76, 77]. CPS's are holistic integrations of computation, communication, and physical processes, and are an important source of hierarchical TDES's. In CPS study, it has been argued that system abstractions need to correctly unify the low-level (or digital and physical) timing dynamics and the high-level (or cyber) computations in a high-level model for system control design. Generalizing logical [1] to timed hierarchical control and borrowing the CPS terminology from [77], the proposed system abstraction for hierarchical consistency with output-time fidelity is said to correctly 'physicalize the cyber' by aggregating the physicality of time into the cyber (virtual) system, and 'cyberize the physical' by semantically linking cyber or high-level events and their control properties defined in the cyber system for an application of interest to appropriate physical control behaviors in terms of timed low-level events in the real

(physical) system. Importantly, with or without time fidelity, the high-level TDES model of the hierarchy resulting from the application of either abstraction concept proposed is endowed with a natural timed control structure, which subsumes the tick preemption concept of event forcing [74] and is a generalization of the untimed version [1].

However, despite its importance as a control architecture, the issue of output-time fidelity for hierarchical control of TDES's has hitherto not been addressed in the literature. In fact, there has been relatively little research conducted on hierarchical control in a real-time framework. A related early effort is reported in [78]. Therein, it is observed that the model time tick at both the low level and the high level is akin to timeout^{1.1} and may be treated like any other event. The research [78] on timed hierarchical control therefore does not consider system time fidelity at all, and in this aspect is fundamentally different from this research.

Finally, to the best of our knowledge, there is no reported research in the literature that furnishes system synthesis procedures needed for building a TDES hierarchy, apart from those developed for building a logical one [1].

1.3 Natural Output Control Structure

A common concept threading through the two hierarchical control problems is system output-control consistency - a formulation that assures a natural output control structure. By this structure, high-level events or outputs also possess unambiguous control properties as the low-level events; additionally, when extended to a real-time version proposed in this thesis, the high-level system also possesses time fidelity as the low-level system. Importantly, the timed version lays a sound real-time high-level specification design foundation for hierarchical control, allowing the same real-time control synthesis [74] for the low level

^{1.1} A timeout event can be used in timed models as explained in [2]. It is a marker that specifies the maximal duration that a system can stay in a system state, within which an activity event it is defined for at the state is expected to occur. A timeout occurrence indicates that the activity event has failed to occur within the specified duration. Because the timeout occurrence may model disablement of the activity and other events upon entering a new state, it is different from time ticks that simply model time progression.

to be applicable to the high level,

1.4 Main Contributions

The main contributions of this research are summarized as follows.

1.4.1 On-line Command and Control for Logical DES's

In contrast to the existing consolidated flat structure, in the first part of this thesis, we propose an alternative design implementing the command and control structure, hitherto only a theoretical concept, through a two-level control algorithm for on-line translation of high-level commands for low-level control. In so doing, the control algorithm can facilitate a deeper understanding of control during runtime operation. The novelty of this new design idea lies in basing the on-line command-to-control translation on some reusable ‘control technology’ developed for the low-level DES, and the algorithmic cooperation between the two levels. In addition, compared to existing approaches in the literature [12, 72], the operational design proposed is shown to significantly reduce off-line complexity and achieve fast on-line computation.

1.4.2 Hierarchical Consistency with Output-Time Fidelity for TDES's

In the second part of this thesis, we define output-time fidelity and propose structural conditions for the low-level Moore TDES in the hierarchy to possess output-time fidelity. Then the well-known concept of hierarchical consistency is extended to TDES's. Two versions are developed, without and with output-time fidelity guarantee. Central to timed system abstractions for hierarchical consistency without and with output-time fidelity guarantee, is the strict version of the respective new system abstraction concepts called output-control consistency and timed output-control consistency, extending the untimed version [1] to real time in this thesis. Of critical interest is the latter stronger concept that captures the notion of time fidelity in system abstraction.

In our theoretical development, the concepts of output-control consistency and partner-freeness for hierarchical control are first developed as generalizations, from untimed DES's [1] to TDES's [78] where time fidelity need not be respected; and the foundation is then augmented with the new concept of output-time compliance for a class of Moore TDES's [74] that possesses time fidelity, to develop a new real-time control-theoretic framework for hierarchical control where output-time fidelity is also respected, i.e., the framework of hierarchical consistency with output-time fidelity. Under this framework, supporting system existence and synthesis results for strict output-control consistency are developed, on which the results of system synthesis, for achieving hierarchical consistency and that with output-time fidelity, are established for a mildly restrictive class of Moore TDES's and its 'linear' subclass, respectively.

1.5 Thesis Organization

The rest of this thesis is organized into five chapters and summarized as follows.

In Chapter 2, the basic notation and foundation concepts within the framework of formal languages and finite automata are formally introduced for both logical and timed DES's. These form the basic background for the thesis. Each set of contributions that follows, in Chapter 3 for logical DES's, and Chapters 4 and 5 for timed DES's, is relatively self-contained, in that additional concepts and notation are presented therein as required for a full technical exposition of the contributions.

In Chapter 3, we present our contributions on a new operational design for hierarchical control of logical DES's, bringing the structure of command and control from concept to realization for on-line control operation. We first review existing work on hierarchical control and motivate the need for on-line command and control. For a command reference input, a new concept for output control feasibility of a Moore DES is then characterized, followed by a system decomposition of a suitably structured Moore DES into a controllable subsystem and an uncontrollable subsystem is formulated; and these are illustrated by an

example. Based on these theoretical findings, the implementability of the new command and control design for hierarchical control is investigated, and a resultant control algorithm is developed with its design complexity examined.

In Chapter 4, we present our contributions on a new theoretical framework for hierarchical control of TDES's that supports real-time high-level specification design for control synthesis. We first review existing work on TDES's, followed by a Moore system formulation of the TDES model for two-level hierarchical control. The need for system output-time fidelity to support sound real-time high-level specification design is then motivated and explained using an example. The constituent concepts for the system concept of timed output-control consistency are defined, by which the system abstraction at the high level possesses a natural timed control structure as the system at the low level. Together with these constituent concepts, a timed concept of partner-freeness is added to formulate system sufficiency structures for hierarchical consistency between the two levels, without and with output-time fidelity guarantee. This chapter concludes with an illustrative example of hierarchical consistency with output-time fidelity.

In Chapter 5, we present further contributions addressing the issues of system synthesis under the real-time hierarchical control framework proposed in Chapter 4. The structural existence and procedural synthesis of the sufficiency structure for hierarchical consistency is investigated, based on which it is then shown that hierarchical consistency can be achieved for a class of TDES hierarchies under mild output-system design restrictions, and the version with output-time fidelity can be achieved for a subclass 'linearly' structured under further output-system design restrictions. We also present a discussion on generalizing and scaling to multiple levels the consistency of a two-level hierarchy. In applying the theoretical concepts and procedures developed, and reiterating the necessity for output-time fidelity, we provide a detailed example to illustrate how to sufficiently refine a Moore TDES of the linear subclass studied to achieve hierarchical consistency with output-time fidelity.

In Chapter 6, we conclude with a summary of our main contributions, and discuss potential directions for future research.

Chapter 2

Background

In this chapter, the relevant components of the language and automata framework for supervisory control of DES's [12, 35, 2, 79] and TDES's [74] are reviewed.

2.1 Languages and Automata

Let Σ be a finite set of symbols representing events. A string is a finite sequence of events. Let Σ^* be the set of strings over Σ , including the empty string ε (a sequence with no events); and $\Sigma^+ = \Sigma^* - \{\varepsilon\}$. Given a string $s \in \Sigma^*$, a string s' is a prefix of s , denoted by $s' \leq s$, if $(\exists t \in \Sigma^*) s't = s$; a strict prefix of s , denoted by $s' < s$, if $s' \leq s$ and $s' \neq s$; and a suffix of s if $(\exists t \in \Sigma^*) ts' = s$.

A formal language L is defined over Σ by a subset of Σ^* . For $L_1, L_2 \subseteq \Sigma^*$, L_1 is said to be a sublanguage of L_2 if $L_1 \subseteq L_2$. The prefix closure of L , denoted by $\overline{L}$, is $\overline{L} = \{s' \mid (\exists s \in L) s' \leq s\}$, the set of prefix strings of strings in L . Clearly, $L \subseteq \overline{L}$, and $L \neq \emptyset$ provided $\varepsilon \in \overline{L}$. The language L is said to be prefix-closed if $L = \overline{L}$.

A regular language is a language that can be generated by a finite-state automaton [80]. An automaton G is a 5-tuple $(Q, \Sigma, \delta, q_0, Q_m)$, where

- Q is the set of states,

- Σ is the finite set of events,
- $\delta : \Sigma \times Q \rightarrow Q$ is the (partial, deterministic) transition function,
- q_0 is the initial state,
- $Q_m \subseteq Q$ is the subset of marked states.

Note that the state set Q is finite unless otherwise specified. That an event $\sigma \in \Sigma$ is defined at a state $q \in Q$ is denoted by $\delta(\sigma, q)!$, and $\neg\delta(\sigma, q)!$ otherwise. For an event subset $\Sigma' \subseteq \Sigma$ and a state $q \in Q$, let $\Sigma'(q) = \{\sigma \in \Sigma' \mid \delta(\sigma, q)!\}$, the subset of events in Σ' that are defined at state q . The transition function δ can be extended to Σ^* as follows: $\delta(\varepsilon, q) = q$, and $(\forall \sigma \in \Sigma)(\forall s \in \Sigma^*)\delta(s\sigma, q) = \delta(\sigma, \delta(s, q))$, which is defined if $q' = \delta(s, q)$ and $\delta(\sigma, q')$ are both defined.

The behavior of automaton G may be described by two languages: the prefix-closed language $L(G) = \{s \in \Sigma^* \mid \delta(s, q_0)!\}$ and the marked language $L_m(G) = \{s \in L(G) \mid \delta(s, q_0) \in Q_m\}$. By definition, $L_m(G) \subseteq L(G)$. We write $G = \text{EMPTY}$ (called an empty automaton) provided the state set $Q = \emptyset$; and $L(\text{EMPTY}) = L_m(\text{EMPTY}) = \emptyset$.

A state $q \in Q$ is reachable (from the initial state q_0) if $(\exists s \in \Sigma^*)\delta(s, q_0) = q$, and coreachable if $(\exists s \in \Sigma^*)\delta(s, q) \in Q_m$. Automaton G is reachable if all its states are reachable, and coreachable if all its states are coreachable and so $\overline{L_m(G)} = L(G)$. Finally, automaton G is trim if it is both reachable and coreachable.

Graphically, an automaton model G is represented by an edge-labeled directed graph with a state represented by a node, and a transition $\delta(\sigma, q) = q'$ by a directed edge from state q to q' labeled with the symbol σ of an event whose occurrence it represents. The initial state q_0 is represented by a node with an entering arrow, and a marked state by a darkened node or a double-concentric circle.

2.2 Cartesian and Synchronous Products

The *Cartesian* and *synchronous* products of automata are defined as follows [81].

Given two automata $G_1 = (Q_1, \Sigma_1, \delta_1, q_{1,0}, Q_{1,m})$ and $G_2 = (Q_2, \Sigma_2, \delta_2, q_{2,0}, Q_{2,m})$, the *Cartesian* product of G_1 and G_2 , denoted by $G_1 \sqcap G_2$, is a reachable automaton $G = (Q, \Sigma, \delta, q_0, Q_m)$ with

- $Q = Q_1 \times Q_2$,
- $\Sigma = \Sigma_1 \cup \Sigma_2$,
- $\delta = \delta_1 \times \delta_2$,
- $q_0 = (q_{1,0}, q_{2,0})$,
- $Q_m = Q_{1,m} \times Q_{2,m}$,

and the transition function δ is defined as follows: $(\forall (q_1, q_2) \in Q_1 \times Q_2)$ and $(\forall \sigma \in \Sigma_1 \cup \Sigma_2)$,

$$\delta(\sigma, (q_1, q_2)) = \begin{cases} (\delta_1(\sigma, q_1), \delta_2(\sigma, q_2)), & \text{if } \delta_1(\sigma, q_1)! \text{ and } \delta_2(\sigma, q_2)! \\ \text{undefined,} & \text{otherwise.} \end{cases}$$

In the *Cartesian* product of G_1 and G_2 , an event transition at a composite state (q_1, q_2) is defined on a shared event, i.e., an event in $\Sigma_1 \cap \Sigma_2$, provided the event is defined at both the component states q_1 and q_2 .

The *synchronous* product of G_1 and G_2 , denoted by $G_1 \parallel G_2$, is a reachable automaton $G = (Q, \Sigma, \delta, q_0, Q_m)$ with

- $Q = Q_1 \times Q_2$,
- $\Sigma = \Sigma_1 \cup \Sigma_2$,
- $\delta = \delta_1 \times \delta_2$,
- $q_0 = (q_{1,0}, q_{2,0})$,
- $Q_m = Q_{1,m} \times Q_{2,m}$.

and the transition function δ is defined as follows: $(\forall (q_1, q_2) \in Q_1 \times Q_2)$ and $(\forall \sigma \in \Sigma_1 \cup \Sigma_2)$,

$$\delta(\sigma, (q_1, q_2)) = \begin{cases} (\delta_1(\sigma, q_1), \delta_2(\sigma, q_2)), & \text{if } \delta_1(\sigma, q_1)! \text{ and } \delta_2(\sigma, q_2)! \\ (\delta_1(\sigma, q_1), q_2), & \text{if } \sigma \in \Sigma_1 - \Sigma_2 \text{ and } \delta_1(\sigma, q_1)! \\ (q_1, \delta_2(\sigma, q_2)), & \text{if } \sigma \in \Sigma_2 - \Sigma_1 \text{ and } \delta_2(\sigma, q_2)! \\ \text{undefined,} & \text{otherwise.} \end{cases}$$

In the *synchronous* product of G_1 and G_2 , an event transition at a composite state (q_1, q_2) is either defined on a shared event if the event is defined at both the component states q_1 and q_2 , or defined on an event that is not shared, as long as the event is defined at component state q_1 or q_2 . If $\Sigma_1 = \Sigma_2$, the synchronous product $G_1 \parallel G_2$ reduces to the Cartesian product $G_1 \sqcap G_2$.

In either product, $\Sigma = \Sigma_1 \cup \Sigma_2$ is the combined event set, and the composite state is marked if both its component states are marked. By the associativity of $\sqcap$ and $\parallel$, the *Cartesian* and *synchronous* products of three or more automata can be recursively constructed as defined above.

2.3 Preliminaries on Hierarchical Control of DES's

2.3.1 Supervisory Control of DES's

Let a reachable automaton $G = (Q, \Sigma, \delta, q_0, Q_m)$ model a DES, with the event set Σ partitioned into the controllable event set Σ_c and the uncontrollable event set Σ_u . A specification language $K \subseteq \Sigma^*$ is said to be controllable with respect to (w.r.t) G if $\bar{K}\Sigma_u \cap L(G) \subseteq \bar{K}$. This controllability condition complies with the fact that a supervisor that exists for DES G cannot physically disable an uncontrollable event, and so only the occurrence of any uncontrollable event always not exiting the bounds of $\bar{K}$ can guarantee non-violation of the specification K . If K is not controllable, there exists a supremal (or largest) controllable marked sublanguage of the DES G that lies within the language K . This sublanguage can

be generated by the trim automaton returned by $Supcon(G, K)$ [79], which has worst-case time complexity of $O(lm^2)$, where l and m are the respective cardinality of the event set Σ and the cross-product state set of DES G and the automaton modeling K .

$Supcon(G, K)$ is a supervisor automaton $S = (X, \Sigma, \xi, x_0, X_m)$, and is said to be non-blocking (for DES G) since $\overline{L_m(S)} = L(S)$ for a trim and hence coreachable $S = Supcon(G, K)$. Its associated control data set $Condat$ [12] w.r.t G is given by

$$Condat(G, S) = \{ \Delta(q, x) \subseteq \Sigma_c \mid (\exists s \in \Sigma^*) \delta(s, q_0) = q \in Q \text{ and } \xi(s, x_0) = x \in X \},$$

where $\Delta(q, x) = \Sigma_c(q) - \Sigma_c(x)$. $Condat(G, S)$ has worst-case time complexity of $O(cm)$, where c is the cardinality of the controllable event set Σ_c .

Intuitively, each data element $\Delta(q, x)$ specifies the controllable events to be disabled at a composite state $(q, x) \in Q \times X$ reachable by a common $s \in \Sigma^*$. By default, the events in $\Sigma(q) \cap \Sigma(x)$ are enabled at state (q, x) . With $L(S) \subseteq L(G)$, a data element $\Delta(\delta(s, q_0), \xi(s, x_0))$ is also conveniently denoted by $Condat(S, s)$.

A control law for a DES G specifies a set of controllable events to be disabled following every input history $s \in L(G)$. Formally, a control law f is a function $f : L(G) \rightarrow 2^\Sigma$ with the constraint

$$(\forall s \in L(G)) (\Sigma_u \cap \Sigma(\delta(s, q_0)) \subseteq (\Sigma - f(s))).$$

The prefix-closed language that results from imposing f on G is denoted by $L(f, G)$ and defined as

$$\varepsilon \in L(f, G), (\forall s \in L(f, G)) (\forall s\sigma \in L(G)) s\sigma \notin L(f, G) \Leftrightarrow \sigma \in f(s).$$

It follows that the supervisor S defined above for a DES G can be implemented by a control law f with $f(s) = Condat(S, s)$, such that $L(f, G) = L(S)$.

2.3.2 Moore Automaton for Low-Level DES Modeling

In the study of two-level hierarchical control (e.g., [1]), the real system at the base or low level is a DES that is equipped with an output function to send significant events to the high-level. A Moore automaton [71] is used to model a class of such DES's.

In general, a DES automaton G with event set Σ needs to be re-structured into a Moore automaton - an automaton^{2.1} $G_{lo} = (Q, \Sigma, \delta, q_0, Q_m)$ associated with an information channel defined by a vocalization map $V : Q \rightarrow T \cup \{\tau_o\}$ - such that $L(G_{lo}) = L(G)$ and $L_m(G_{lo}) = L_m(G)$. T denotes the high-level (virtual) event set, and the symbol $\tau_o \notin T$ denotes a 'silent output'. The Moore construction [71] for the DES G is based on a given reporter map - a virtual projection $\theta : L(G) \rightarrow T^*$, defined such that $\theta(\varepsilon) = \varepsilon$ and, for $\sigma \in \Sigma$ and $s\sigma \in L(G)$, $\theta(s\sigma)$ is either $\theta(s)$ or $\theta(s)\tau$ for some $\tau \in T$. For the constructed G_{lo} , the vocalization map V for every $s' \in L(G_{lo})$ is defined by

$$V(\delta(s', q_0)) = \begin{cases} \tau_o, & \text{if } s' = \varepsilon \text{ or } \delta(s', q_0) \notin Q_{voc} \\ \tau \in T, & \text{otherwise,} \end{cases}$$

where the selected subset $Q_{voc} \subseteq Q$, called vocal state set, is defined as follows. For $\sigma \in \Sigma$ and $s' = s\sigma$,

$$\delta(s\sigma, q_0) \begin{cases} \notin Q_{voc}, & \text{if } \theta(s\sigma) = \theta(s) \\ \in Q_{voc}, & \text{if } \theta(s\sigma) = \theta(s)\tau. \end{cases}$$

The reporter map θ can be extended to $\theta(K) \subseteq T^*$ for $K \subseteq L(G_{lo})$ as follows: $\theta(K) = \{\theta(s) \mid s \in K\}$. The inverse reporter map for $t \in T^*$ is then defined as follows: $\theta^{-1}(t) = \{s \in L(G_{lo}) \mid \theta(s) = t\}$. The inverse reporter map θ^{-1} can be extended to $\theta^{-1}(E) \subseteq L(G_{lo})$ for $E \subseteq T^*$ as follows: $\theta^{-1}(E) = \bigcup_{t \in E} \theta^{-1}(t)$.

The Moore automaton (G_{lo}, V) is simply referred to as G_{lo} when V is understood. Under

^{2.1}Although the same 5-tuple notation is used as in Subsection 2.3.1, it should be clear in the context that the structure of G_{lo} is in general not the same as that of a given DES G .

the map V , G_{lo} outputs events in T to drive some high-level θ -image model G_{hi} whenever it reaches a vocal state $q \in Q_{voc}$, and otherwise outputs the silent symbol $\tau_o \notin T$ to signal no ‘significant’ change for the high level. Formally, model G_{hi} , the high-level image of G_{lo} , is an automaton such that $L(G_{hi}) = \{\theta(s) \mid s \in L(G_{lo})\}$ and $L_m(G_{hi}) = \{\theta(s) \mid s \in L_m(G_{lo})\}$. G_{hi} is said to generate events of T under the θ -map on $L(G_{lo})$.

The high-level event set T of G_{hi} is partitioned into the controllable event set T_c and the uncontrollable event set T_u . To ensure that every such high-level event $\tau \in T$ defined and output by (G_{lo}, V) is unambiguously controllable or uncontrollable, the Moore automaton (G_{lo}, V) , or simply G_{lo} when V is understood, needs to be refined so that it becomes output-control consistent. Formally, (the Moore transition structure of) a DES G_{lo} is said to be output-control consistent [12] if, for every string $s \in L(G_{lo})$ of the form

$$s = \sigma_1 \sigma_2 \cdots \sigma_k \text{ or, respectively, } s = s' \sigma_1 \sigma_2 \cdots \sigma_k$$

where $s' \in \Sigma^+$, with

- $V(\delta(\sigma_1 \sigma_2 \cdots \sigma_i, q_0)) = \tau_o \quad (1 \leq i \leq k-1),$
- $V(\delta(s, q_0)) = \tau \in T$

or, respectively,

- $V(\delta(s', q_0)) \in T,$
- $V(\delta(s' \sigma_1 \sigma_2 \cdots \sigma_i, q_0)) = \tau_o \quad (1 \leq i \leq k-1),$
- $V(\delta(s, q_0)) = \tau \in T,$

it is the case that

- if $\tau \in T_c$, then for some $i \ (1 \leq i \leq k), \sigma_i \in \Sigma_c,$
- if $\tau \in T_u$, then for all $i \ (1 \leq i \leq k), \sigma_i \in \Sigma_u.$

2.4 Preliminaries on Hierarchical Control of TDES's

2.4.1 TDES Model

A TDES [74] can be modeled by an automaton called activity transition graph (ATG) and the timing information associated with each system event. Combining the ATG model and timing information furnishes a timed transition graph (TTG), an automaton generating prefixed-closed and marked languages that explicitly model the timed behaviors of the TDES.

Formally, the ATG of a TDES is the automaton

$$G_{act} = (A, \Sigma_{act}, \delta_{act}, a_0, A_m), \quad (2.4.1)$$

where the state set is redesignated as A , the set of activities and is finite, with each activity associated with a time duration, Σ_{act} is the finite set of activity events, $\delta_{act} : \Sigma_{act} \times A \rightarrow A$ is the activity transition function, a_0 is the initial activity and $A_m \subseteq A$ is the subset of marked activities.

Let $\mathbb{N} = \{0, 1, 2, \dots\}$, the set of natural numbers. In associating the ATG G_{act} with timing information, each event $\sigma \in \Sigma_{act}$ is assigned with time bounds, namely, a lower time bound $l_\sigma \in \mathbb{N}$ and an upper time bound $u_\sigma \in \mathbb{N} \cup \{\infty\}$, where $l_\sigma \leq u_\sigma$, and specified as $\sigma[l_\sigma, u_\sigma]$. A time bound is quantified in terms of a number of ticks of the global clock. A time tick is denoted by a special event symbol $tick \notin \Sigma_{act}$, and its occurrence denotes a transition or passage of an atomic unit of time. Under these time bound assignments, Σ_{act} is divided into two disjoint subsets Σ_{spe} and Σ_{rem} , i.e., $\Sigma_{act} = \Sigma_{spe} \cup \Sigma_{rem}$ and $\Sigma_{spe} \cap \Sigma_{rem} = \emptyset$, and this partition is denoted by $\Sigma_{act} = \Sigma_{spe} \dot{\cup} \Sigma_{rem}$. $\Sigma_{rem} = \{\sigma \in \Sigma_{act} \mid u_\sigma = \infty\}$ is called the subset of remote events; and $\Sigma_{spe} = \{\sigma \in \Sigma_{act} \mid u_\sigma < \infty\}$ is called the subset of prospective events. Each event $\sigma \in \Sigma_{act}$ has a local countdown timer t_σ with a default value

$t_{\sigma 0}$ initialized as

$$t_{\sigma 0} = \begin{cases} u_{\sigma}, & \text{if } \sigma \in \Sigma_{spe} \\ l_{\sigma}, & \text{if } \sigma \in \Sigma_{rem}. \end{cases}$$

Intuitively, the existence of a lower time bound means that an event σ is only eligible or *ready to occur* in the TDES after l_{σ} ticks upon entering an activity in G_{act} (2.4.1) where σ is defined, and will never occur before that; and each *tick* occurrence decreases the timer t_{σ} by one tick count, until $t_{\sigma} = 0$. If σ is a remote event and t_{σ} is or has decreased to 0, it becomes eligible but might or might not occur next. If σ is a prospective event, it might occur during $0 \leq t_{\sigma} \leq u_{\sigma} - l_{\sigma}$, and must occur next when $t_{\sigma} = 0$ unless it is preempted by another imminent event. The timer interval or duration D_{σ} is defined for σ as follows:

$$D_{\sigma} = \begin{cases} [0, u_{\sigma}], & \text{if } \sigma \in \Sigma_{spe} \\ [0, l_{\sigma}], & \text{if } \sigma \in \Sigma_{rem}. \end{cases}$$

Therefore, $t_{\sigma} \in D_{\sigma}$. Being instantaneous [74], an event occurrence is modeled as abrupt with no time duration.

Let $\Sigma = \Sigma_{act} \dot{\cup} \{tick\}$. Given the ATG G_{act} (2.4.1) and timer information as defined above for each event $\sigma \in \Sigma_{act}$, the TTG of the TDES is the automaton

$$G = (Q, \Sigma, \delta, q_0, Q_m), \quad (2.4.2)$$

with finite state set $Q = A \times \prod \{D_{\sigma} \mid \sigma \in \Sigma_{act}\}$ and marked state set $Q_m \subseteq A_m \times \prod \{D_{\sigma} \mid \sigma \in \Sigma_{act}\}$. Each state $q \in Q$ is of the form $q = (a, \{t_{\sigma} \mid \sigma \in \Sigma_{act}\})$, and $q_0 = (a_0, \{t_{\sigma 0} \mid \sigma \in \Sigma_{act}\})$ is the initial state.

For an activity event $\sigma \in \Sigma_{act}$ and a state $q = (a, -) \in Q$, σ is eligible at q provided $\delta(\sigma, q)!$, and is said to be enabled at q provided $\delta_{act}(\sigma, a)!$; and $\delta(\sigma, q)!$ iff $\delta_{act}(\sigma, a)!$ and

$$\begin{cases} t_{\sigma} = 0, & \text{if } \sigma \in \Sigma_{rem} \\ 0 \leq t_{\sigma} \leq u_{\sigma} - l_{\sigma}, & \text{if } \sigma \in \Sigma_{spe}. \end{cases} \quad (2.4.3)$$

The TDES G is also subjected to the following conditions: For every $q = (a, -) \in Q$,

$$\delta(\text{tick}, q)! \text{ iff } (\forall \beta \in \Sigma_{spe}, \delta_{act}(\beta, a)!) t_\beta > 0, \quad (2.4.4)$$

and

$$(\forall s \in \Sigma_{act}^+, \delta(s, q)!) \delta(s, q) \neq q. \quad (2.4.5)$$

Condition (2.4.4) - time-progressivity (TP) - characterizes that the time event *tick* is eligible at state q provided no prospective event is due at the state. Condition (2.4.5) - activity-loop freeness (ALF) - asserts that there is no activity loop at a state $q \in Q$ in TDES G . An activity loop is a cycle containing only activity events, and repeated execution of an activity loop is deemed to incur no time duration. As such loops are physically infeasible, this condition is needed to exclude such loops in (the languages of) TDES G .

In meeting TP (2.4.4) and ALF (2.4.5), the persistence of time (evolution) is not violated in TDES model G , characterizing the fact that a TDES can never stop the clock [74].

2.4.2 TDES Composition

We now review TDES composition. A TDES G is usually a modular system of n component TDES's $G_1, G_2, \dots, G_n, n \geq 2$, with their respective component ATG's

$G_{1,act}, G_{2,act}, \dots, G_{n,act}$; and it is herewith denoted as $G = G_1 \parallel G_2 \parallel \dots \parallel G_n$ where $\parallel$ is called the composition operator, and the ATG G_{act} of G is given by $G_{act} = G_{1,act} \parallel G_{2,act} \parallel \dots \parallel G_{n,act}$.

In composing the TDES G from its components, the general approach [74] is to first construct the modular ATG G_{act} by applying the synchronous operator $\parallel$, and then use it to construct the TDES by applying the TDES transition function rules for event eligibility and countdown timer update of each activity event σ , based on its time bounds (l_σ, u_σ) in the TDES G that are determined as follows. For any activity event σ shared by two or more component TDES's, its time bounds in G are defined to be (l_σ, u_σ) , where the lower time bound l_σ is the maximum of the lower time bounds for σ among the component TDES's

(sharing it), and its upper bound u_σ is the minimum of the upper time bounds for σ among these component TDES's, provided $l_\sigma \leq u_\sigma$. If $l_\sigma > u_\sigma$ for any shared activity event σ , the composition G is not defined. For any activity event σ defined in only one component TDES, the time bounds (l_σ, u_σ) are the same as those given for the component TDES.

Finally, if no two arbitrary component TDES's share an activity event, then

$$G = G_1 \parallel G_2 \parallel \dots \parallel G_n = G_1 \parallel G_2 \parallel \dots \parallel G_n \text{ [12].}$$

2.4.3 Control Theoretic Setting for TDES's

The control-theoretic setting [74] for TDES's assumes that the subset of events controllable by an external supervisor is predetermined. In a logical DES, an event is controllable if it is prohibitable, in that it can be prevented from occurring by (control) disablement. Extending to a TDES G (2.4.2), this notion of controllability is subsumed for activity events, and the event *tick* solely denoting an elapsed real time is also considered controllable at where its system transitions can be preempted. In TDES G (2.4.2), it is further postulated that an event in Σ_{spe} is not prohibitable, or uncontrollable, and it must occur next once its upper time bound is reached unless it is preempted by another imminent event, whereas an event in Σ_{rem} may be. With $\Sigma_{act} = \Sigma_{spe} \dot{\cup} \Sigma_{rem}$, it follows that the set of prohibitable events, denoted by Σ_{hib} , is a subset of Σ_{rem} , i.e., $\Sigma_{hib} \subseteq \Sigma_{rem}$. In what follows, the uncontrollable event set is defined as $\Sigma_u = \Sigma_{act} - \Sigma_{hib} = \Sigma_{spe} \dot{\cup} (\Sigma_{rem} - \Sigma_{hib})$. Let $\Sigma_{for} \subseteq \Sigma_{act}$ be the set of forcible events. An enforced forcible event can only preempt *tick*, i.e., only *tick* will not occur next, at a state where both *tick* and the forcible event are eligible. As a forcible event can be either prohibitable or uncontrollable, various cases with regard to the preemptability of *tick* by a forcible event are distinguished in Definition 2.1.

Definition 2.1 (Tick preemptability). *The event $tick \in \Sigma(q)$ with $q = \delta(s, q_0)$ for an arbitrary $s \in L(G)$ is said to be*

- *non-preemptable at q if $\Sigma(q) \cap \Sigma_{for} = \emptyset$,*
- *unambiguously preemptable at q if $\Sigma(q) \cap \Sigma_{for} \cap \Sigma_u \neq \emptyset$,*

- *and ambiguously preemptable at q if*

$$\Sigma(q) \cap \Sigma_{for} \cap \Sigma_u = \emptyset \ \& \ \Sigma(q) \cap \Sigma_{for} \cap \Sigma_{hib} \neq \emptyset.$$

Where the context is understood, ‘at q ’ in Definition 2.1 is dropped when referring to tick preemptability.

The event *tick* is ‘controllable’ by preemption through a forcible event, not by disablement as for an event in Σ_{hib} , since nothing can stop the global clock. Accordingly, the controllable event set is defined as $\Sigma_c = \Sigma - \Sigma_u = \Sigma_{hib} \dot{\cup} \{tick\}$. Therefore, $\Sigma = \Sigma_c \dot{\cup} \Sigma_u$ and this is identical to the control-theoretic setting for logical DES’s [35].

Whenever it is not written as a member of the respective event subsets, a prohibitible and an uncontrollable event may be identified by a superscript ‘+’ and ‘-’ on its event symbol, respectively, and additionally followed by a superscript ‘#’ provided the event is forcible.

We now define the various notions of control strings (and events) w.r.t the transition structure of a TDES G . Given an arbitrary nonempty string $s = \sigma_1\sigma_2 \cdots \sigma_k \in \Sigma^*$ which is a suffix of some string of $L(G)$, and where $\sigma_i \in \Sigma$ for all i ($1 \leq i \leq k$), string s is said to be

- controllable if, for some i ($1 \leq i \leq k$), $\sigma_i \in \Sigma_{hib}$, or $\sigma_i = tick$ and is unambiguously preemptable;
- uncontrollable if, for all i ($1 \leq i \leq k$), $\sigma_i \in \Sigma_u$, or $\sigma_i = tick$ and is non-preemptable;
- ambiguously controllable if, for all i ($1 \leq i \leq k$), $\sigma_i \in \Sigma_u$, or $\sigma_i = tick$ and is not unambiguously preemptable, and for some j ($1 \leq j \leq k$), $\sigma_j = tick$ and is ambiguously preemptable;
- preemption-unambiguous if, for all i ($1 \leq i \leq k$), either $\sigma_i \in \Sigma_{act}$, or $\sigma_i = tick$ and is not ambiguously preemptable.

Note that a string s as defined above is either controllable, uncontrollable or ambiguously controllable. Therefore, it is not controllable if it is either uncontrollable or ambiguously

controllable, or equivalently, for all i ($1 \leq i \leq k$), $\sigma_i \in \Sigma_u$, or $\sigma_i = tick$ and is not unambiguously preemptable. An ambiguously controllable event is an event $tick$. A preemption-unambiguous string does not contain ambiguously preemptable ticks, and an uncontrollable string is preemption-unambiguous.

2.4.4 Time Fidelity of TDES Model

For an automaton G of the type (2.4.2) modeling a TDES, the following are its qualitative temporal properties.

Property 2.1 (Persistence of time). *Let $q = \delta(s, q_0)$. Then $(\Sigma_{act}(q) = \emptyset) \implies \delta(tick, q)!$.*

Proof. For a state $q = (a, \{t_\beta \mid \beta \in \Sigma_{act}\})$, assume $\Sigma_{act}(q) = \emptyset$. Then, to prove that $\delta(tick, q)!$, according to TP (2.4.4), we need to show that for all $\beta \in \Sigma_{spe}$ such that $\delta_{act}(\beta, a)!$, $t_\beta > 0$. Since $\Sigma_{act}(q) = \emptyset$ iff for all $\beta \in \Sigma_{act}$, $\neg\delta(\beta, q)!$, it follows by the assumption and (2.4.3) that, for all $\beta \in \Sigma_{spe}$ such that $\delta_{act}(\beta, a)!$, $t_\beta > u_\beta - l_\beta \geq 0$, i.e., $t_\beta > 0$. Hence the property. $\square$

The property states that a time event $tick$ is always eligible at a reachable state with no eligible activity events. By Property 2.1, the continual time elapse that persists even during the transience or absence of system activity is modeled.

Property 2.2 (Time invariance of event eligibility). *Let $q = \delta(s, q_0)$ and $q' = \delta(tick, q)$. Then $\Sigma_{act}(q) \subseteq \Sigma_{act}(q')$.*

Proof. Let $q' = \delta(tick, q)$, with $q = (a, \{t_\beta \mid \beta \in \Sigma_{act}\})$ and $q' = (a', \{t'_\beta \mid \beta \in \Sigma_{act}\})$. Since $q' = \delta(tick, q)$ and $\neg\delta_{act}(tick, a)!$, $a' = a$. Therefore, for $\beta \in \Sigma_{act}$, $\delta_{act}(\beta, a)!$ iff $\delta_{act}(\beta, a')!$. Since $\Sigma_{act} = \Sigma_{spe} \dot{\cup} \Sigma_{rem}$, to show that for $\beta \in \Sigma_{act}$, $\beta \in \Sigma_{act}(q) \implies \Sigma_{act}(q')$, we need to prove two cases, as follows:

- *Case 1:* Suppose $\beta \in \Sigma_{act}(q) \cap \Sigma_{spe}$. Then $\delta(\beta, q)!$ and by (2.4.3), $\delta_{act}(\beta, a)!$ & $0 \leq t_\beta \leq u_\beta - l_\beta$. Since $\delta(tick, q)!$, $t_\beta > 0$ by (2.4.4). Since $\delta_{act}(\beta, a)!$ & $\beta \in \Sigma_{spe}$ & $t_\beta > 0$,

the occurrence of *tick* only decreases timer t_β by one unit, i.e., $t'_\beta = t_\beta - 1$. Hence, since $0 \leq t'_\beta \leq u_\beta - l_\beta$ & $\delta_{act}(\beta, a')!$, $\delta(\sigma, q')!$ by (2.4.3), and therefore $\beta \in \Sigma_{act}(q')$.

- *Case 2:* Suppose $\beta \in \Sigma_{act}(q) \cap \Sigma_{rem}$. Then $\delta(\beta, q)!$ and by (2.4.3), $\delta_{act}(\beta, a')!$ & $t_\beta = 0$. Since $\delta_{act}(\beta, a')!$ & $\beta \in \Sigma_{rem}$ & $t_\beta = 0$, the occurrence of *tick* does not decrease the timer t_β further, i.e., $t'_\beta = t_\beta = 0$. Hence, since $t'_\beta = 0$ & $\delta_{act}(\beta, a')!$, $\delta(\beta, q')!$ by (2.4.3), and therefore $\beta \in \Sigma_{act}(q')$.

Hence the property. $\square$

The property states that eligible activity events at a reachable state remain eligible at another following the time elapse of a tick. By Property 2.2, the continual execution of an activity event as time elapses is effectively modeled. Applying this property iteratively, tick occurrences represent the elapse of time until some eligible event occurs instantaneously.

Given the uncontrollable event set Σ_u in the control-theoretic setting formulated, an important system model property strengthening Property 2.1 may now be presented.

Property 2.3 (Uncontrollable persistence of time). *Let $q = \delta(s, q_0)$. Then $(\Sigma_{act}(q) \cap \Sigma_u = \emptyset) \implies \delta(tick, q)!$.*

Proof. For a state $q = (a, \{t_\beta \mid \beta \in \Sigma_{act}\})$, assume $\Sigma_{act}(q) \cap \Sigma_u = \emptyset$. Then, since $\Sigma_{act}(q) \cap \Sigma_u = \emptyset$ iff $\Sigma_{act}(q) = \emptyset$ or $(\Sigma_{act}(q) \neq \emptyset \text{ \& } \Sigma_{act}(q) \subseteq \Sigma_{hib})$, we need to prove two cases, as follows:

- *Case 1:* Suppose $\Sigma_{act}(q) = \emptyset$. Then, by Property 2.1, it follows that $\delta(tick, q)!$.
- *Case 2:* Suppose $\Sigma_{act}(q) \neq \emptyset$ & $\Sigma_{act}(q) \subseteq \Sigma_{hib}$. Then to prove that $\delta(tick, q)!$, according to TP (2.4.4), we show that for every event $\beta \in \Sigma_{spe}$ such that $\delta_{act}(\beta, a')!$, $t_\beta > 0$, as follows:

Since $\Sigma_{hib} \subseteq \Sigma_{rem}$, $\Sigma_{act}(q) \subseteq \Sigma_{rem}$. Together with the fact that $\Sigma_{rem} \cap \Sigma_{spe} = \emptyset$, it follows that $\Sigma_{act}(q) \cap \Sigma_{spe} = \emptyset$, meaning that $\neg\delta(\beta, q)!$ for all $\beta \in \Sigma_{spe}$. Therefore, by (2.4.3), it follows that, for all $\beta \in \Sigma_{spe}$ such that $\delta_{act}(\beta, a')!$, $t_\beta > u_\beta - l_\beta \geq 0$, i.e., $t_\beta > 0$.

Hence the property. □

The property states that the event *tick* is always eligible at a reachable state with no eligible uncontrollable event. This means the evolution of time in the model G does not halt regardless of the absence of activity events or the disablement of all eligible prohibitable events at a reachable state.

Properties 2.1 to 2.3 present new supporting insights for our research. They provide a clearer understanding of the TDES model G (2.4.2) [74] reviewed. In fact, Properties 2.2 and 2.3 together with ALF (2.4.5) of TDES G model system time fidelity, characterizing time progression as never halting an executing activity event (Property 2.2) and unstoppable (Property 2.3 and ALF (2.4.5)). Equivalently, the model G is said to possess sound system (event-) timing semantics; or the time *tick* is said to be Σ -uninterrupting.

We turn to the notion of a (control) specification language, an arbitrary constraint on which a supervisor is to be synthesized to restrict (the behavior of) a TDES as specified. To fix the notion in automata, we define a specification TTG C for a TDES G as a trim automaton that shares the same event set as the TDES G . This TTG is said to prescribe the specification language $L_m(C)$ for restricting G to within the language $L_m(C \sqcap G)$. Note that the essence of the control requirements by specification TTG C is in the ‘composed specification TTG $C \sqcap G$, which prescribes intra-system restrictions. As these restrictions include prohibitions on activity events and tick preemption in general, $C \sqcap G$ need not satisfy Properties 2.2 and 2.3 of system time fidelity.

2.4.5 Supervisory Control of TDES’s

For a sublanguage $L \subseteq L(G)$ (having the same event set Σ), let $\Sigma^L(q) = \{\sigma \in \Sigma \mid \delta(\sigma, q) \neq \emptyset \text{ and } s\sigma \in \bar{L}\}$ be the set of eligible events at state q w.r.t Σ and the string $s \in \bar{L}$, such that $q = \delta(s, q_0)$. Then a (specification) language $K \subseteq \Sigma^*$ is said to be controllable

w.r.t G if, for all $s \in \bar{L}$ with $\bar{L} = \bar{K} \cap L(G)$,

$$\Sigma^L(q) \supseteq \begin{cases} \Sigma(q) \cap (\Sigma_u \cup \{tick\}), & \text{if } \Sigma^L(q) \cap \Sigma_{for} = \emptyset \\ \Sigma(q) \cap \Sigma_u, & \text{if } \Sigma^L(q) \cap \Sigma_{for} \neq \emptyset. \end{cases}$$

Intuitively, it means that following an arbitrary string $s \in \bar{L}$, the TDES G does not slip out (of $\bar{L}$ and hence $\bar{K}$) on an uncontrollable event, and any *tick* that it may slip out on can be preempted without the TDES slipping out as a result. In general, K may not be controllable w.r.t G , but the supremal (or largest) controllable marked sublanguage of the TDES G that lies within K exists. This sublanguage can be generated by a trim automaton computed as $Supcon(G, K)$ ^{2.2} [74, 12], and is exactly L if K is controllable and $L = K \cap L(G) \subseteq L_m(G)$.

$Supcon(G, K)$ is a timed supervisor automaton S with the same event set Σ , and is said to be nonblocking (for TDES G) since $\overline{L_m(S)} = L(S)$ for a trim and hence coreachable $S = Supcon(G, K)$. As a supervisor that can generate the marked language of $Supcon(G, K)$ in conjunction with TDES G , $S = Supcon(G, K)$ is said to be optimal or maximally permissive (w.r.t G under language K). To exercise supervision on G , the supervisor S can ‘disable’ events in $\Sigma_c = \Sigma_{hib} \cup \{tick\}$, i.e., disable prohibitable activity events and preempt *tick*, where appropriate.

Let $\bar{G}$ be TDES G but with all its states marked. It follows that if $S = Supcon(\bar{G}, \bar{K})$, then $L_m(S) = L(S)$; and by definition, an arbitrary language Z for which $\bar{Z} \cap L(G) = L(S)$ is controllable w.r.t G . Such a ‘safety’ supervisor S obtained for Z is not guaranteed to be nonblocking for the original TDES G , unless $\overline{Z \cap L_m(G)} = L(S)$.

A control law for a TDES G specifies a set of controllable events to be disabled following an input string $s \in L(G)$. Formally, a control law f is a function $f : L(G) \rightarrow 2^\Sigma$, such

^{2.2}Note that, since TDES G is a (finite-state) TTG, the language of interest for control synthesis, $K \cap L_m(G)$, is a regular language and can thus be modeled by a TTG. In the algorithmic computation [12] of $Supcon(G, K)$, K can be practically expressed as a regular language by a specification TTG.

that for all $s \in L(G)$, with $q = \delta(s, q_0)$,

$$\Sigma(q) - f(s) \supseteq \begin{cases} \Sigma(q) \cap (\Sigma_u \cup \{tick\}), & \text{if } \Sigma_{for}(q) \subseteq f(s) \\ \Sigma(q) \cap \Sigma_u, & \text{if } \Sigma_{for}(q) \not\subseteq f(s).^{2,3} \end{cases}$$

The prefix-closed language generated from imposing f on G is denoted by $L(f, G)$ and defined as

$$\varepsilon \in L(f, G), (\forall s \in L(f, G))(\forall s\sigma \in L(G))s\sigma \notin L(f, G) \Leftrightarrow \sigma \in f(s).$$

It follows that the timed supervisor S defined above for a TDES G can be implemented by a control law f such that $L(f, G) = L(S)$.

We conclude this section with a remark regarding a practical relaxation of ALF (2.4.5) in TDES G (2.4.2) and the resulting design considerations for obtaining real-time control behavior.

Remark 2.1. Consider an arbitrary (reachable) TTG $(Q, \Sigma, \delta, -, -)$, such that $(\forall q \in Q)(\forall s \in \Sigma_{act}^+)(\delta(s, q) = q) \implies$

$$(\exists s' \in \Sigma^*)(\exists w \in \Sigma^*\{tick\}\Sigma^*)s' < s \ \& \ \delta(s'w, q) = q. \quad (2.4.6)$$

In our theoretical development, we adhere to system ALF (2.4.5) [74]. However, ALF (2.4.5) in TDES G (2.4.2) may not be met in practice but can often be relaxed to Condition (2.4.6). This relaxation may also lead to physically meaningful control behavior (with regard to eventual real-time tick execution). But that is provided the TTG synthesized [74] for the supremal controllable sublanguage of a given specification w.r.t TDES G also satisfies Condition (2.4.6), and the underlying system mechanism that is unmodeled can execute activity events and break out of every activity loop that exists in the controlled system onto a transition path of an ‘outer’ loop in which a tick occurs, as seen in the practical system

^{2,3}Note that $(\Sigma_{for}(q) \not\subseteq f(s))$ is equivalent to $(f(s) \neq \Sigma_{for}(q) \cup f(s))$.

modeling and control design of some real-time system applications [82, 83] of the control theory [74] reviewed. ■

2.5 Moore Reachability Tree for Conceptual Computation

Given a Moore automaton (G_{lo}, V) , where $G_{lo} = (Q, \Sigma, \delta, q_0, Q_m)$ is reachable. The (Moore) reachability tree [12] generated for (G_{lo}, V) is the Moore automaton $(G_{lo,t}, V_t)$, of which:

- $G_{lo,t} = (Q_t, \Sigma, \delta_t, n_0, Q_{m,t})$, such that $L(G_{lo,t}) = L(G_{lo})$ and $L_m(G_{lo,t}) = L_m(G_{lo})$, where Q_t and $Q_{m,t}$ are called the infinite set of nodes and marked nodes, respectively, and each node is identified with a string $s \in L(G_{lo})$ by a bijection $node : L(G_{lo}) \rightarrow Q_t : s \mapsto node(s)$, such that the initial or root node $n_0 = node(\varepsilon)$, and, extended to Σ^* , $\delta_t(\varepsilon, n) = n$ where $n = node(s')$ for an $s' \in L(G_{lo})$, and $(\forall \sigma \in \Sigma)(\forall s \in \Sigma^*)\delta_t(s\sigma, n) = \delta_t(\sigma, \delta_t(s, n))$, and is defined as $node(s's\sigma)$ if $n' = \delta_t(s, n)!$ & $\delta_t(\sigma, n')!$.
- $V_t : Q_t \rightarrow T \cup \{\tau_o\}$ is the corresponding vocalization map, such that for an arbitrary $s \in L(G_{lo})$, $V_t(\delta_t(s, n_0)) = V(\delta(s, q_0))$.

Clearly, $L(G_{lo,t}) = L(G_{lo})$ and $L_m(G_{lo,t}) = L_m(G_{lo})$, and conceptually, translations between a Moore automaton and its Moore reachability tree can be made [12]. Corresponding with a state of G_{lo} under V , under V_t , a node of the Moore tree automaton (or simply tree) $G_{lo,t}$ is silent if it outputs τ_o , and vocal if it outputs a high-level event in T ; and $Q_{voc,t} \subseteq Q_t$ denotes the vocal node set of $G_{lo,t}$. Note that every string of $L(G_{lo})$ can be uniquely identified by a node in the tree, and vice versa.

Chapter 3

System Abstraction for On-line Logical Hierarchical Control

Since its founding [35], the supervisory control of discrete-event systems (DES's) [12, 4, 2] has evolved into an increasingly important science of automation for many engineering control problems, with particular application success reported for automated manufacturing systems (see e.g., [84, 85, 86]). In this chapter, for problems where command and control is required, a new theoretical and algorithmic basis for operational design of hierarchical discrete-event control is investigated and discussed.

The chapter is organized as follows. We start by reviewing related research in the literature in Section 3.1, and state our research motivation in Section 3.2. In Section 3.3, the system-output control feasibility and decomposition of a DES into a controllable subsystem and an uncontrollable subsystem, given a command reference input, are first formulated; and are illustrated by an example. Based on these results, the implementability of the new command and control design for hierarchical control is investigated in Section 3.4, and a resultant control algorithm is developed with its design complexity examined. Section 3.5 concludes the chapter.

3.1 Hierarchical Control of DES's: A State of the Art

According to Grigorov in his unpublished 2005 survey paper [87], there are four major approaches in the literature on hierarchical control of logical DES's. These approaches include bottom-up design, state aggregation, top-down design and interface-based design. In this section, we provide an overview of research under each of these approaches.

Bottom-up Design

The founding contribution to hierarchical control research, as well as to the bottom-up design approach, is reported in [1]. The framework development is a scalable two-level design, and is an architectural extension of the monolithic control theory of DES's initiated in [35].

In a basic two-level hierarchy, the DES at the high level is an abstraction of the (real) DES at the low level, by a reporter map projecting system languages from the low level to the high level. Broadly speaking, two types of event reporter maps have been proposed to model a DES hierarchy, namely, virtual [1, 12, 88] and natural [69, 70, 89, 90] projections. By the former projection, the high-level DES in a two-level hierarchy is modeled by virtual events, i.e., events abstracted or projected as a symbolically different set from that of the low-level DES. By the latter projection, the high-level DES is modeled by a subset of the events defined for the low-level DES.

The founding work of [1] is based on the virtual projection paradigm. In this research, the low-level DES is first constructed as a Moore automaton [71], in which some states become vocal states outputting high-level events while other states remain non-vocal. Outputs by vocal states represent significant changes in the low-level system that are deemed important to the level above. The high-level prefix-closed and marked languages are regular languages projected from those of the low-level. In order to apply the control theory [35] for the high level, the reporter map is designed or refined such that every high-level event is unambiguously controllable or uncontrollable. This fundamental requirement is

embodied in the concept of output-control consistency. Using this concept as the cornerstone, the research interest is on how to construct or refine a DES hierarchy so that an arbitrary controllable specification task, given by a prefix-closed or marked sublanguage of the high-level DES, can be fully realized by control at the low level.

It turns out that the hierarchy needs to be structured or restructured to possess some form of consistency. In the first efforts [1], a theory for the concept of hierarchical consistency is developed, by which an arbitrary controllable specification task, given by a prefix-closed sublanguage of the high-level DES, can be fully realized by low-level control. To achieve such consistency, it is sufficient for the low-level DES to satisfy a stronger concept called strict output-control consistency, which subsumes the condition that no two arbitrary vocal states are partners. Intuitively, two vocal states are not partners if they vocalize virtual outputs that can be independently controlled. To address the blockingness issue, an extension to a theory for the concept of hierarchical consistency with marking is developed [67, 72, 12], by which an arbitrary controllable specification task, given by a marked sublanguage of the high-level DES, can be fully realized by low-level nonblocking control. The theory includes formulating perhaps the weakest known sufficiency conditions for hierarchical consistency with marking [12]. A more specialized concept called hierarchical consistency with nonblockingness has also been developed [12, 67]. This specialized concept, for which additional structural conditions have been developed [12], ensures that an arbitrary controllable specification task, given by a marked sublanguage of the high-level DES, can be fully realized (with nonblockingness) by low-level control without marking.

Given some consistent hierarchy, and a consistency-supported high-level specification (on the high-level DES), the corresponding high-level supervision can be implemented by synthesizing a low-level supervisor automaton from some equivalent low-level specification (on the low-level DES). This low-level specification for control synthesis is obtained by applying the inverse reporter map on the high-level specification. Procedural implementations of this inverse map are described in [72] for specifications given by (regular) prefix-closed and marked sublanguages of the high-level DES.

Among research efforts addressing different issues in the virtual projection paradigm, hierarchical control has been extended to handle partial observation of the low-level DES [88], and to ensure control robustness [91]. It has also been generalized to handle partial observation of the low-level system modeled as a fuzzy DES [59]. Besides, different multi-level extensions of the two-level control hierarchy have been developed and investigated [75, 68].

Related work based on the other paradigm of natural projection [69, 89, 92] develops different scalable methods of subsystem synthesis aimed at reducing the computational effort. Research efforts include proposing or enhancing different hierarchical design methods for nonblocking control synthesis, by exploiting modularity (e.g., [75, 93, 94, 95]) and decentralization (e.g., [96, 97, 98, 99, 70, 100, 101, 89]) of DES's in a hierarchical structure. In essence, unlike a centralized controller that has to be constructed to act fully on the overall system, modular or decentralized controllers are synthesized to exercise their control only on that part of the DES that matches their event set. Hierarchical control has also been extended to handle partial observation of the low-level DES [90], and to general ω -languages using natural projection defined for infinite strings [102, 103, 104].

State Aggregation

State aggregation is another interesting approach. The first known approach describes state aggregation by the concept of partition machines [105] and its variant [106]. In [105], the system state set is aggregated by partitioning it into *blocks* or subsets of states. Each instance of such a system partition is called a partition machine. A partition machine is said to be controllable if, for two arbitrary states belonging to (or aggregating) two different blocks, called a source state and a target state, there exists a string of events that can bring the system from the source state to the target state. In other words, controllability in the state aggregation approach implies reachability of states. In more detail, two additional concepts have been developed: in-block controllability and between-block controllability. A partition machine is said to be in-block controllable if, for an arbitrary block,

there exists a directed string connecting two arbitrary states of the block; and it is said to be between-block controllable if there exists a directed string connecting two arbitrary blocks. A block is then considered as a superstate, implying that the system abstraction can be realized with a state set consisting of these superstates. An important result is that, for an arbitrary partition machine that is in-block controllable, the system abstraction is controllable provided the partition machine is between-block controllable. With a partition machine, a specification, which is given in terms of desired control trajectories, can be decomposed into high-level and low-level components, and its controllability can be verified structurally. This partition machine approach is extended to incorporate event controllability and uncontrollability [107], and the issues of nonblockingness are subsequently studied [108]. Important applications of the partition machine approach include the control of hybrid system [109, 110], and the computation of optimal paths [111, 112].

State aggregation is also employed in a bottom-up hierarchical setup [69, 113]. In this framework, the low-level system is modeled by a standard finite-state machine [3], while the high-level system is equipped with a Γ -control structure which is first proposed in [114]. A subset of the low-level event set is distinguished as the high-level event set; the latter set contains relevant events deemed more critical than the other events for the high level. The state set is divided into blocks. Two states belong to the same block if there exist some strings of irrelevant events connecting them. A block is represented by a single state at the high level. Intuitively, in this formulation, a Γ -control structure for the high-level system that is valid is a construction where, for every block and every input state of the block, the corresponding element of the control structure contains high-level control options. Each high-level option is a possible set of events leading out of the block, following (high-level) strings in the natural projection on an arbitrary supremal controllable sublanguage within the block with the input state as initial state, and that either contains strings reaching a marked state or leading out of the block, or contains strings leading out of the block and do not pass through a marked state inside the block. And, for every such supremal controllable sublanguage which contains a marked string, the corresponding control option is marked.

An objective of this approach is to achieve system hierarchical consistency with minimum or no refinement of the hierarchy. To which, a theory of hierarchical consistency in a linguistic framework is developed in terms of a high-level Γ -compatible sublanguage, including an algorithm for synthesis of the low-level supervisor given a high-level supervisor, either off-line (to obtain the low-level supervisor) or on-line (to obtain the event-controls for every block that is reached).

Top-down Design

The first known approaches of top-down design are inspired by the concept of statecharts introduced in [115]. In these approaches, generally, the system is a finite-state model that is recursively defined from a system model at the highest level to subsystems at the lowest level. Each successive refinement is a ‘zooming in’ of each system state at one level to a lower level that reveals more details defined about the state. In general, a state at one level hides its more detailed information of substates and substate transitions in the lower levels. In formally structuring the system top-down, states are classified into the three categories originally defined for statecharts, namely, basic states, AND states and OR states. Basic states cannot be refined while AND and OR states are composite states that can be refined into one or multiple subsystems at the immediate lower level. A subsystem can be either a single finite-state machine, if it belongs to an OR superstate, or a composition of finite-state machines, if it belongs to an AND superstate. With this setup, modules belonging to the same AND superstate can execute concurrently, implying that a certain degree of parallelism can be modeled. Hierarchical state machines [116], hierarchical finite state machines [117, 118] and state tree structures [119, 120] are among the most well-known statechart-inspired models.

In [116], the authors define hierarchical state machines with components as described above, and consider a special class of hierarchical state machines called asynchronous hierarchical state machines, which satisfies the condition that the event sets of two arbitrary subsystems are disjoint. A *configuration* of a hierarchical state machine is defined by a

tuple (or collection) of states, with one state from each level of the machine, and shows the states the machine is in at the same time. Using this definition, two problems are addressed in [116], namely, the reachability of configurations, and the forbidden configurations. In [117, 118], a variant of hierarchical state machines called hierarchical finite state machines is proposed. In this model variant, the original model is restricted such that any transition from a high-level state to a lower-level state can only enter a state at one level lower, and vice versa. For this model variant, the authors highlight and investigate the issues of non-blockingness, complexity and optimal control, and present various algorithms to address those issues.

In [119], for supervisory control synthesis, state tree structures is first proposed for representing (or encoding) DES's modeled by finite automata, and control specifications given in the form of logic predicates. In [120], to overcome the issues of state tree structures raised in [119], including problems of handling AND superstates, complexity of non-blocking synthesis, as well as the restrictions inherent in hierarchical state machines and hierarchical finite state machines, the authors propose a method that efficiently leverages binary decision diagrams [121] for control computation of DES's that have large state sets. As demonstrated in recent research, cast in the framework of state tree structures [120], localized control can be more efficiently synthesized for very large systems [122].

Other top-down design approaches have also been proposed. In [123], the authors propose a language-based approach, with the DES modeled top-down by a hierarchy of subsystems, with its behaviors represented by the recursion concept of a structured language. The event sets at different levels of abstraction in the linguistic hierarchy are mutually disjoint, and the structured language generated by the upper level contains events belonging to that level, with tokens embedded that represent the languages of subsystems in the lower level. In this framework, language containment and controllability are formulated by recursive definitions, and a recursive procedure for computing a supremal controllable sublanguage is proposed.

In addressing the issue of synthesis complexity, recent efforts include three-level co-

ordination synthesis of control [124] and its extension to handle control under partial observation [125] for modular DES's. Herewith, a modular DES is a DES consisting of a finite number of subsystems. In a top-down fashion, to meet some system 'safety' property specified by a prefixed-closed language, the three-level hierarchical approach synthesizes a coordinator at the first level, and then decomposes it into several coordinators at the second level, with one for each *a priori* fixed group of the subsystems of the DES. Finally, proceeding to the third level, it synthesizes a solution set of supervisors, with one for each subsystem of each group of subsystems, and one for each group coordinator combined with the first-level coordinator. In this line of research [124, 125], the (prefix-closed language) specification decomposability and controllability conditions for the existence of supervisors, under full and partial observation, are presented and discussed. The results in [124] are applied to decentralized control [126].

Interface-based Design

In interface-based design [127, 128, 86, 129, 130], the standard two-level hierarchy setup is a system consisting of three components, namely, a high-level system, an interface and a low-level system. The event set of the whole system is partitioned into the set of high-level events for the high-level system, the set of low-level events for the low-level system, the set of 'request' events, and the set of 'answer' events. The latter two subsets form the event set for the interface, and is the common event set of the three system components. By this formulation, the high-level and the low-level systems communicate strictly through the interface.

The interface-based approach targets complex systems whose internal components can be decomposed into independent components. If the 3-component interfaced system satisfies the formulated properties of interface consistency and level-wise nonblockingness, the system is nonblocking. Interface consistency can be verified by algorithms developed in [131]. A pair of a high-level and a low-level specification, for synthesizing a supervisor each at the respective level, is called level-wise controllable if, 1) the event set of each spec-

ification is localized to the events of their respective level and the interface, 2) the low-level specification together with the interface is controllable for the low-level system and 3) the high-level specification is controllable for the high-level system together with the interface. If a given system is interface consistent and level-wise controllable, then the combined or synchronized specification is controllable for the whole system. These results can be extended to multiple low-level subsystems working in parallel and interacting with the high level via separate interfaces (see [127, 128] for mathematical details or the survey [87] for a good intuitive understanding). One application of this approach is reported in [86].

Recently, the issues of synthesis [129] and flexibility [130] are further investigated, including a multi-level extension [73] of the two-level interface-based hierarchy that is shown to more greatly reduce synthesis complexity and improve reconfigurability, due to the partitioning of smaller modules of the system.

3.2 Towards On-line Command and Control: A

Motivation

As overviewed, the current literature on logical hierarchical control is quite extensive. However, we note that, in the founding bottom-up design approach [12], the command and control structure, theoretically conceptualized for hierarchical control as depicted in Figure 1.4, is not implemented as such for runtime operation, once hierarchical consistency (HC) between a low-level DES G_{lo} and its high-level abstraction G_{hi} is assured^{3.1}. Instead, given a high-level specification E for DES G_{hi} , hierarchical control according to high-level supervision $S_{hi} = Supcon(G_{hi}, E)$ is actually implemented using a low-level supervisor automaton S_{lo} ^{3.2}, which is either $Supcon(G_{lo}, \theta^{-1}(E))$ or $Supcon(\overline{G}_{lo}, \theta^{-1}(L(S_{hi})))$, where $\overline{G}_{lo}$ is G_{lo} but with all its states marked, and the associated control data. Besides the is-

^{3.1}The reader is referred to the literature for details of HC [12, 1], hierarchical consistency with marking (HCM) [12, 67, 72], and a more specialized concept of HCM [12] relevant to this research (mentioned later in Remark 3.1).

^{3.2}It can be computed as $Supcon(G_{lo}, \theta^{-1}(E))$ if HCM holds; and as $Supcon(\overline{G}_{lo}, \theta^{-1}(L(S_{hi})))$ if the more specialized concept [12] of HCM holds.

Figure 3.1: Operational designs for hierarchical control

sue of synthesis complexity, such a consolidated ‘flat’ design, shown in Figure 3.1(a), may not be adequate for two other reasons. Firstly, command and control may be the inherent mode of runtime operation for the engineering control problem of interest. Secondly, there is limited operational clarity when command is not explicitly linked to control on-line to show what stepwise controls are exercised in carrying out the commands from above. Such on-line causal clarity of command over control is important to designers, when subjecting the control system to validation tests by simulation and observation.

In this chapter, an operational design linking command and control is proposed as depicted in Figure 3.1(b), bringing the concept depicted in Figure 1.4 to realization. Referring to Figure 3.1(b), the design uses the high-level supervisor automaton S_{hi} and a f_{com} -output control law μ , also called a command-to-control ‘transfer’ function, to continually translate command data in f_{com} to control data during runtime. Clearly, this design obviates the need for the low-level supervisor automaton S_{lo} . The details and discussion of this alternative approach are presented in the rest of the chapter.

3.3 System-Output Control with Command Input

3.3.1 Preliminaries

Consider the feedback control loop shown in Figure 3.2.

Figure 3.2: Feedback control of a DES with command input and system output

The closed loop interconnecting the controller and DES allows the former to modify the latter's dynamics based on feedback. In a standard setting, this loop applies a principle of feedback and control, namely, the control design of dynamics through feedback. Set in the Moore DES paradigm, the design dynamics refers to that generated by the system under control, without vocalizing or outputting via the reporter the virtual events specified in the command reference input f_{com} (from the instant the reference is set). Based on the system feedback string s generated under control, the controller continually computes the next control input.

The fundamental problem, then, is whether a feasible output control law μ for the reference input f_{com} exists for controller realization or implementation. To explain more formally later, a feasible output control law can prevent the vocalization of events specified in the reference input set. In the rest of this section, this is theoretically investigated for a single reference and an extension to some regulated sequences of references.

3.3.2 Problem Approach: Uncontrollable and Controllable States

Without loss of generality, in this research, a Moore automaton is always constructed such that its initial state is non-vocal or silent.

Now, consider a (reachable) Moore DES G_{lo} with $\theta : L(G_{lo}) \rightarrow T^*$. For $q \in Q_{voc}$, $V(q) = \tau$, let

$$B_u[q, V(q)] = \{p \in Q - \{q_0\} \mid (\exists t \in \Sigma_u^*)(\delta(t, p) = q \ \& \ (\forall t' < t)V(\delta(t', p)) = \tau_o)\}.$$

Excluding the initial state $q_0 \in Q$, $B_u[q, V(q)]$ defines a set of τ -uncontrollable states which are non-vocal other than the state q , and each non-vocal state can reach state q via a string of uncontrollable events defined at non-vocal states.

The complete set of τ -uncontrollable states is

$$B_u^\tau = \bigcup_{V(q)=\tau} B_u[q, V(q)]. \quad (3.3.1)$$

Now, define

$$B_c[q, V(q)] = Q - B_u[q, V(q)].$$

Then

$$B_c^\tau = \bigcap_{V(q)=\tau} B_c[q, V(q)] = Q - B_u^\tau$$

is said to define the complete set of τ -controllable states if the following condition holds for virtual event $\tau \in T$:

$$(\forall q' \in Q_{voc} \cup \{q_0\})(\forall w \in \Sigma^+)(V(\delta(w, q')) = \tau) \Rightarrow w \notin \Sigma_u^*. \quad (3.3.2)$$

Informally, Condition (3.3.2) asserts that from the initial state or any vocal state, a nonempty string leading to a state vocalizing the high-level event τ is not uncontrollable.

If Condition (3.3.2) holds, then the following condition also holds:

$$\begin{aligned} &(\forall s\sigma \in L(G_{lo}), \sigma \in \Sigma_c)(\forall t \in \Sigma_u^*) \\ &(V(\delta(s\sigma t, q_0)) = \tau) \Rightarrow (\forall t' < t)V(\delta(s\sigma t', q_0)) = \tau_o. \end{aligned} \quad (3.3.3)$$

Informally, Condition (3.3.3) asserts that if a nonempty string of uncontrollable events immediately following a controllable event leads the DES to a reachable state q vocalizing the high-level event τ , all the states except q via which the uncontrollable string traverses are non-vocal.

In what follows, w.r.t $f_{com} \subseteq T$ of a DES G_{lo} ,

$$B_c^{f_{com}} = \bigcap_{\tau \in f_{com}} B_c^\tau = Q - B_u^{f_{com}} \quad (3.3.4)$$

is said to be the (f_{com}) -controllable state set, where

$$B_u^{f_{com}} = \bigcup_{\tau \in f_{com}} B_u^\tau \quad (3.3.5)$$

is the (f_{com}) -uncontrollable state set, if the following condition holds for f_{com} :

$$(\forall q' \in Q_{voc} \cup \{q_0\})(\forall w \in \Sigma^+)(V(\delta(w, q')) \in f_{com}) \Rightarrow w \notin \Sigma_u^*. \quad (3.3.6)$$

That Condition (3.3.6) holds implies that the following condition also holds^{3.3}:

$$\begin{aligned} &(\forall s\sigma \in L(G_{lo}), \sigma \in \Sigma_c)(\forall t \in \Sigma_u^*) \\ &(V(\delta(s\sigma t, q_0)) \in f_{com}) \Rightarrow (\forall t' < t)V(\delta(s\sigma t', q_0)) = \tau_o. \end{aligned} \quad (3.3.7)$$

Then the command-to-control ‘transfer’ function μ , as shown in the proposed operational design depicted in Figure 3.1(b), can be written as

$$\begin{aligned} \mu(f_{com}, s) = &\{\sigma \in \Sigma_c \mid (\exists t \in \Sigma_u^*)(V(\delta(s\sigma t, q_0)) \in f_{com} \\ &\& (\forall t' < t)V(\delta(s\sigma t', q_0)) = \tau_o)\}. \end{aligned} \quad (3.3.8)$$

This function is an output control law that is said to be ‘permissive’, in that it disables a controllable event only when the event occurrence can otherwise lead the system uncontrollably to a state vocalizing a virtual event in the reference input f_{com} . Based on the characterization of B_u^τ (3.3.1) and the definition of $B_u^{f_{com}}$ (3.3.5), this law can be rewritten as

$$\mu(f_{com}, s) = \{\sigma \in \Sigma_c \mid \delta(s\sigma, q_0)! \& \delta(s\sigma, q_0) \in B_u^{f_{com}}\}. \quad (3.3.9)$$

^{3.3}That (3.3.2) implies (3.3.3) and (3.3.6) implies (3.3.7) can be proved by contradiction. The proofs are simple and straightforward, and hence are omitted.

3.3.3 Output Control Feasibility

Theorem 3.1. Consider a Moore DES (G_{lo}, V) constructed with $\theta : L(G_{lo}) \rightarrow T^*$, and an arbitrary $f_{com} \subseteq T$. Then for all $s \in L(G_{lo})$, μ (3.3.9) is the permissive control law for (f_{com}, s) such that $V(\delta(s, q_0)) \in f_{com}$ or $\delta(s, q_0) \in B_c^{f_{com}}$ implies

$$(\forall \sigma \in \Sigma_u(\delta(s, q_0))) \delta(s\sigma, q_0) \notin B_u^{f_{com}}$$

iff $B_c^{f_{com}}$ is the controllable state set.

Proof. (If) The sufficiency proof proceeds as follows. For $s \in L(G_{lo})$, let $q' = \delta(s, q_0)$. There are two cases to consider. To prove by contradiction for each case, assume that there exists a $\sigma \in \Sigma_u(q')$ such that $\delta(\sigma, q') \in B_u^{f_{com}}$.

- *Case 1:* Suppose $q' \in B_c^{f_{com}} - (Q_{voc} \cup \{q_0\})$.

Then by definition of $B_u^{f_{com}}$ and the assumption, it follows that $q' \in B_u^{f_{com}}$. Therefore, $q' \in B_u^{f_{com}} \cap B_c^{f_{com}}$, contradicting the fact that $B_u^{f_{com}} \cap B_c^{f_{com}} = \emptyset$.

- *Case 2:* Suppose $q' \in B_c^{f_{com}} \cap (Q_{voc} \cup \{q_0\})$ (implying $V(q') \notin f_{com}$), or $V(q') \in f_{com}$ (i.e., $q' \in B_u^{f_{com}} \cap Q_{voc}$).

Then by definition of $B_u^{f_{com}}$ and the assumption, there is an uncontrollable string $w = \sigma t \in \Sigma_u^+$ such that $V(\delta(\sigma t, q')) \in f_{com}$, contradicting the fact that w contains a controllable event by Condition (3.3.6).

(Only If) The necessity proof proceeds as follows. Given an arbitrary $B_c^{f_{com}}, f_{com} \subseteq T$:

- Suppose, for every string $s \in L(G_{lo})$ such that $q' = \delta(s, q_0) \in B_c^{f_{com}}$, or $V(q') \in f_{com}$ and hence $q' \in Q_{voc}$,

$$(\forall \sigma \in \Sigma_u(q')) \delta(\sigma, q') \notin B_u^{f_{com}}.$$

Since by definition, $B_u^{f_{com}} \cup B_c^{f_{com}} = Q$ and $B_u^{f_{com}} \cap B_c^{f_{com}} = \emptyset$, $\delta(\sigma, q') \notin B_u^{f_{com}}$ iff $\delta(\sigma, q') \in B_c^{f_{com}}$. By definition, for $V(q) = \tau \in f_{com}$, $B_u[q, \tau]$ of $B_u^\tau \subseteq B_u^{f_{com}}$ contains

non-vocal states (excluding the initial state $q_0 \in Q$) and the state $q \in Q_{voc}$, such that each non-vocal member state can reach state q via a string of uncontrollable events defined at non-vocal member states. As a result, if $q' = \delta(s, q_0) \in B_c^{f_{com}}$ or $V(q') \in f_{com}$, then every event $\sigma \in \Sigma$ for which $\delta(\sigma, q') \in B_u^{f_{com}}$ must be controllable; and so is it if $q' \in B_c^{f_{com}} \cap (Q_{voc} \cup \{q_0\})$ or $V(q') \in f_{com}$. Therefore, every string $w \in \Sigma^+$ that can bring every such $q' \in Q$ to some state q vocalizing a $\tau \in f_{com}$, i.e., $V(\delta(w, q')) = V(q) = \tau \in f_{com}$, must contain at least one controllable event σ_c , such that $w = s'\sigma_c t$ for some $s' \in \Sigma^*$ and $t \in \Sigma_u^*$. Together with the fact that $q_0 \in B_c^{f_{com}}$ and G_{lo} is reachable, it follows that Condition (3.3.6) holds, since $w = s'\sigma_c t \notin \Sigma_u^*$.

Hence the theorem. $\square$

Intuitively, Theorem 3.1 states the necessary and sufficient condition for the control law (3.3.9) to be feasible for (f_{com}, s) . Being feasible means the output control law can always prevent the system from next entering $B_u^{f_{com}}$ if it is in a vocal state that just vocalized an event in f_{com} or in $B_c^{f_{com}}$, and thus guarantees subsequent nonoccurrence of events in f_{com} .

3.3.4 The Decomposition Theorem

In partitioning the states of a Moore DES $G_{lo} = (Q, \Sigma, \delta, q_0, Q_m)$ w.r.t an arbitrary $f_{com} \subseteq T$, it can be said that $B_c^{f_{com}}$ (3.3.4) induces a subsystem, denoted by $G_c^{f_{com}}$, that is controllable if it is the controllable state set, and $B_u^{f_{com}}$ (3.3.5) induces the corresponding uncontrollable subsystem denoted by $G_u^{f_{com}}$. The pair $(G_c^{f_{com}}, G_u^{f_{com}})$ is then called a feasible system decomposition.

Formally, the subsystems $G_c^{f_{com}}$ and $G_u^{f_{com}}$ can be modeled as automata^{3.4} derived from the Moore automaton G_{lo} . For the subsystem $G_c^{f_{com}} = (Q_c, \Sigma, \delta_c, Q_{0,c}, Q_{m,c})$, the state set is $Q_c = B_c^{f_{com}} \subseteq Q$, the transition function $\delta_c : \Sigma \times Q_c \rightarrow Q_c$ is a restriction of δ to $\Sigma \times Q_c$, the initial state set is $Q_{0,c} = Q_c \cap (Q_{voc} \cup \{q_0\})$, defining an initial state as where f_{com} is

^{3.4}Note that, with some abuse of notation, the initial state q_0 in the usual 5-tuple formalization of an automaton is extended to a state set.

computed following system initialization or an occurrence of a high-level event that is not in f_{com} , and the marked state set is $Q_{m,c} = Q_m \cap Q_c$.

For the uncontrollable subsystem $G_u^{f_{com}} = (Q_u, \Sigma, \delta_u, Q_{0,u}, Q_{m,u})$, the state set is $Q_u = B_u^{f_{com}} \subseteq Q$, the transition function $\delta_u : \Sigma \times Q_u \rightarrow Q_u$ is a restriction of δ to $\Sigma \times Q_u$, the initial state set is $Q_{0,u} = \{q \in Q_u \mid (\exists q' \in B_c^{f_{com}})(\exists \sigma \in \Sigma)q = \delta(\sigma, q')\}$, defining an initial state as the first state of entry into the subsystem, and the marked state set is $Q_{m,u} = Q_m \cap Q_u$.

Each subsystem can be represented by a (possibly unconnected) subgraph of the state-transition graph representing the DES G_{lo} .

Below, Theorem 3.2 states that a (feasible) system decomposition, as conceptually depicted in Figure 3.3, exists for a DES G_{lo} that is output-control consistent (OCC), with command input $f_{com} \subseteq T_c$.

The proof of Theorem 3.2 requires the following lemma.

Lemma 3.1. *For every $\tau \in T_c$ of an OCC DES G_{lo} , Condition (3.3.2) holds.*

Proof. Consider a string $s' \in L(G_{lo})$, where $q' = \delta(s', q_0) \in Q_{voc} \cup \{q_0\}$. For an OCC G_{lo} , if the string s' can be extended to a state vocalizing an event $\tau \in T_c$, i.e., $(\exists w \in \Sigma^+)(q = \delta(s'w, q_0) = \delta(w, q') \in Q_{voc} \text{ and } V(q) = \tau \in T_c)$, then $w = s\sigma_c t$ for some $s \in \Sigma^*$, $\sigma_c \in \Sigma_c$ and $t \in \Sigma_u^*$. This in turn implies that Condition (3.3.2) holds since $w = s\sigma_c t \notin \Sigma_u^*$. Hence the lemma. $\square$

Figure 3.3: Conceptual decomposition of a DES

Theorem 3.2. *An OCC Moore automaton G_{lo} , w.r.t an arbitrary $f_{com} \subseteq T_c$, can be decomposed into a controllable subsystem $G_c^{f_{com}}$ and an uncontrollable subsystem $G_u^{f_{com}}$.*

Proof. For a Moore automaton G_{lo} that is OCC, by Lemma 3.1, Condition (3.3.2) holds for every $\tau \in T_c$. It is easy to show that, for all $\tau \in f_{com} \subseteq T_c$, the conjunctions of Condition (3.3.2) constitute Condition (3.3.6). Hence, for an arbitrary $f_{com} \subseteq T_c$, the set $B_c^{f_{com}}$ (3.3.4) is controllable, inducing a controllable subsystem $G_c^{f_{com}}$, with $B_u^{f_{com}}$ (3.3.5), where $B_c^{f_{com}} \cap B_u^{f_{com}} = \emptyset$, inducing the corresponding uncontrollable subsystem $G_u^{f_{com}}$. Hence the theorem. $\square$

Note that, upon computing an f_{com} immediately following system initialization or a high-level event occurrence, the low-level system G_{lo} , w.r.t the decomposition $(G_c^{f_{com}}, G_u^{f_{com}})$, will be in a state $q \in Q_{0,c} \cup Q_u \cap Q_{voc}$.

Illustrative Example

Consider the OCC DES $G_{lo} = (Q, \Sigma, \delta, q_0, Q_m)$ with high-level event set T for system output, as adapted from [1] and shown in Figure 3.4. The DES model is represented by an edge-labeled directed graph with a state represented by a node, and a transition $\delta(\sigma, q) = q'$ by a directed edge from state q to q' labeled with the symbol σ of an event whose occurrence it represents. The symbol for a controllable and an uncontrollable event is indicated with a superscript '+' and '-', respectively. $\Sigma = \{\sigma_i^+ \mid i = 1, 2, 3, 4\} \cup \{\sigma_j^- \mid j = 5, 6, 7, 8, 9\}$, and $T = \{\tau_i^+ \mid i = 1, 2, 3\} \cup \{\tau_4^-\}$. Every state is denoted by a number. The initial state $q_0 = 0$ is represented by a node with an entering arrow, a marked state by a darkened node, and a vocal state by a node containing the symbol of an event that it vocalizes.

The τ -uncontrollable state sets for T_c are given in the following: $B_u^{\tau_1^+} = B_u[1, V(1)] \cup B_u[8, V(8)]$; $B_u^{\tau_2^+} = B_u[5, V(5)]$ and $B_u^{\tau_3^+} = B_u[7, V(7)]$, where $B_u[q, V(q)]$ is respectively depicted in Figure 3.4. One can easily verify that every arbitrary $f_{com} \subseteq T_c$ decomposes the OCC G_{lo} into a controllable and an uncontrollable part (Theorem 3.2), such that when the system is in a state of the controllable part (induced by $Q - B_u^{f_{com}}$) or a state that has

Figure 3.4: An example OCC DES G_{lo} and its τ -uncontrollable state sets, $\tau \in T_c$

vocalized a high-level event in f_{com} , it can be prevented from entering the uncontrollable part by disabling some controllable low-level events (Theorem 3.1), and hence disallowing the events in f_{com} as desired.

3.4 Command and Control Operational Design

3.4.1 The Implementability Theorem

The implementability result is now established for the command and control design shown in Figure 3.1(b). The result is based on the feasibility and decomposition Theorems 3.1 and 3.2, respectively. Formally, it shows that the design can be realized for a controllable high-level specification, w.r.t a high-level DES G_{hi} obtained under a language map θ on a low-level DES G_{lo} that is OCC.

Theorem 3.3. *Given that an OCC Moore automaton G_{lo} is constructed with $\theta : L(G_{lo}) \rightarrow$*

T^* such that $L(G_{lo}) \mapsto L(G_{hi})$ and $L_m(G_{lo}) \mapsto L_m(G_{hi})$; and $E \subseteq T^*$. Let $S_{hi} = \text{Supcon}(G_{hi}, E)$ and $S_{lo} = \text{Supcon}(\overline{G}_{lo}, \theta^{-1}(L(S_{hi})))$, where $\overline{G}_{lo}$ is G_{lo} but with all its states marked. Assume $L(S_{hi}) \neq \emptyset$. Then

$$L(\mu, G_{lo}) = L(S_{lo}),$$

where $\mu(f_{com}, s)$ (3.3.9) is the permissive control law, with $s \in L(S_{lo})$ and $f_{com} = \text{Condat}(S_{hi}, \theta(s))$. ■

Proof. Given that an OCC Moore automaton G_{lo} is constructed with $\theta : L(G_{lo}) \rightarrow T^*$ such that $L(G_{lo}) \mapsto L(G_{hi})$ and $L_m(G_{lo}) \mapsto L_m(G_{hi})$; and that for some $E \subseteq T^*$, $S_{hi} = \text{Supcon}(G_{hi}, E)$ and $S_{lo} = \text{Supcon}(\overline{G}_{lo}, \theta^{-1}(L(S_{hi})))$, where $\overline{G}_{lo}$ is G_{lo} but with all its states marked, and $f_{com} = \text{Condat}(S_{hi}, \theta(s))$ for an $s \in L(S_{lo})$.

For an arbitrary output subset $f \subseteq T_c$ of the OCC G_{lo} , by Theorem 3.2, the controllable state set B_c^f exists.

Let $L(S'_{hi}) = \theta(L(S_{lo}))$ and $f'_{com} = \text{Condat}(S'_{hi}, \theta(s)) \subseteq T_c$ with $s \in L(S_{lo})$. It follows that $L(S'_{hi}) \subseteq L(S_{hi})$ since $f'_{com} \supseteq f_{com}$ in general for the same control data $\text{Condat}(S_{lo}, s)$ following $s \in L(S_{lo})$. In what follows, it can be shown that

$$B_c^{f_{com}} = B_c^{f'_{com}} \cup [(\bigcup_{\tau \in f'_{com} - f_{com}} B_u^\tau) - B_u^{f_{com}}]. \quad (3.4.1)$$

By the characterization of the controllable state set $B_c^{f'_{com}}$ and the controllability of $L(S_{lo})$, for an arbitrary controlled string $s \in L(S_{lo})$,

$$V(\delta(s, q_0)) \notin f'_{com} \implies \delta(s, q_0) \in B_c^{f'_{com}}.$$

In other words, $V(\delta(s, q_0)) \in f'_{com}$ or $\delta(s, q_0) \in B_c^{f'_{com}}$. However, since vocal states for different $\tau \in T_c$ reside only in their respective B_u^τ , for $V(\delta(s, q_0)) \in f'_{com} - f_{com}$, $\delta(s, q_0) \in$

$[(\bigcup_{\tau \in f'_{com} - f_{com}} B_u^\tau) - B_u^{f_{com}}]$. Together with (3.4.1), it follows that

$$V(\delta(s, q_0)) \in f_{com} \text{ or } \delta(s, q_0) \in B_c^{f_{com}}. \quad (3.4.2)$$

Since $B_c^{f_{com}}$ is a controllable set, by Theorem 3.1, $V(\delta(s, q_0)) \in f_{com}$ or $\delta(s, q_0) \in B_c^{f_{com}}$ implies $\delta(s\sigma, q_0) \notin B_u^{f_{com}}$ for all $\sigma \in \Sigma_u(\delta(s, q_0))$. Applying (3.4.2) and the fact that $Q = B_u^{f_{com}} \cup B_c^{f_{com}}$ with $B_u^{f_{com}} \cap B_c^{f_{com}} = \emptyset$, the result is that, for all $s \in L(S_{lo})$,

$$(\forall \sigma \in \Sigma_u(\delta(s, q_0))) \delta(s\sigma, q_0) \in B_c^{f_{com}}.$$

Now, together with the controllability of $L(S_{lo})$, it follows that for all $\sigma \in \Sigma_u$, for all $s \in L(S_{lo})$, where $\delta(s\sigma, q_0)!$,

$$\delta(s\sigma, q_0) \in B_c^{f_{com}} \text{ iff } s\sigma \in L(S_{lo}).$$

Next, it needs to be proved that, for all $\sigma \in \Sigma_c$, for all $s \in L(S_{lo})$, where $\delta(s\sigma, q_0)!$,

$$\delta(s\sigma, q_0) \in B_c^{f_{com}} \text{ iff } \delta(s\sigma, q_0) \notin B_u^{f_{com}} \text{ iff } s\sigma \in L(S_{lo}),$$

as follows:

- (If) $s\sigma \in L(S_{lo})$ and $\sigma \in \Sigma_c$. Assume $\delta(s\sigma, q_0) \in B_u^{f_{com}}$. This implies there is a $\tau \in f_{com} = \text{Condat}(S_{hi}, \theta(s))$, unambiguously controllable since G_{lo} is OCC, that can uncontrollably occur by vocalization, contradicting the fact that $\theta(s)\tau \notin L(S_{hi})$. Thus $\delta(s\sigma, q_0) \notin B_u^{f_{com}}$ or equivalently, $\delta(s\sigma, q_0) \in B_c^{f_{com}}$.
- (Only If) $s \in L(S_{lo})$, $\delta(s\sigma, q_0)!$, $\delta(s\sigma, q_0) \in B_c^{f_{com}}$ and $\sigma \in \Sigma_c$. There are two cases:
 - Case 1: $\theta(s\sigma) = \theta(s) \in L(S_{hi})$.

Since $B_c^{f_{com}}$ is controllable, and the controllable $L(S_{lo})$ is supremally permissive w.r.t $\theta^{-1}(L(S_{hi}))$, $s\sigma \in L(S_{lo})$.

– Case 2: $\theta(s\sigma) = \theta(s)\tau$ for some $\tau \notin f_{com}$, and therefore $\theta(s)\tau \in L(S_{hi})$.

Assuming $s\sigma \notin L(S_{lo})$ contradicts the fact that $\theta(s)\tau \in L(S_{hi})$. Thus, $s\sigma \in L(S_{lo})$.

Hence for all $\sigma \in \Sigma$, for all $s \in L(S_{lo})$, where $\delta(s\sigma, q_0)!$,

$$(s \in L(S_{lo}) \ \& \ \delta(s\sigma, q_0) \in B_c^{f_{com}}) \text{ iff } s\sigma \in L(S_{lo}).$$

Assuming that $L(S_{hi}) \neq \emptyset$, $\varepsilon \in L(S_{hi})$ and therefore $\varepsilon \in L(S_{lo})$. Thus, with μ (3.3.9) as the permissive control law for f_{com} according to

$$\mu(f_{com}, s) = \{\sigma \in \Sigma_c \mid \delta(s\sigma, q_0)! \ \& \ \delta(s\sigma, q_0) \in B_u^{f_{com}}\}$$

such that $(\forall \sigma \in \Sigma_u(\delta(s, q_0)))\delta(s\sigma, q_0) \in B_c^{f_{com}}$, it follows by induction on the string $s \in L(S_{lo})$, beginning with $s := \varepsilon \in L(S_{lo})$, that, for all $\sigma \in \Sigma$,

$$s\sigma \in L(\mu, G_{lo}) \text{ iff } s\sigma \in L(S_{lo});$$

or that $L(\mu, G_{lo}) = L(S_{lo})$. Hence the theorem. $\square$

Since $B_u^{f_{com}} = \bigcup_{\tau \in f_{com}} B_u^\tau$ according to (3.3.5), under Theorem 3.3, B_u^τ (3.3.1) for all $\tau \in T_c$ form the set of τ -components that is said to provide the ‘control technology’ for implementing command and control using the law μ (3.3.9).

Remark 3.1. Note that $L(S_{hi}) = \overline{L_m(S_{hi})}$. In general, $\theta(L(S_{lo})) \subseteq L(S_{hi})$. Thus Theorem 3.3 alone does not imply that the high-level controllable sublanguage $L(S_{hi})$ can be fully met by low-level control via the law μ (3.3.9), and without high-level blocking during run-time caused by a low-level string $s \in L(S_{lo})$ that cannot be extended, by any $s' \in \Sigma^*$, to $ss' \in L(S_{lo})$ such that $\theta(ss') \in L_m(S_{hi})$. To meet such a high-level (nonblocking) expectation through the control law μ , the assurance of both HC [1] and hierarchical nonblockingness (N) [12] - a more specialized concept (HCN) of HCM - must first be provided, for which

additional structural conditions have been developed [12]. ■

3.4.2 Algorithmic Procedures

Based on Theorem 3.3, hierarchical control depicted in Figure 3.1(b) can be realized by a control algorithm computing the law (3.3.9) on-line, with a command and a control level coupled by top-down command f_{com} and bottom-up event vocalization feedback V implementing the reporter map θ .

The control algorithm contains two procedures **HI-manager** and **LO-operator**, shown respectively in Figures 3.5 and 3.6. Through the command of the high-level DES G_{hi} by **HI-manager**, **LO-operator** computes on-line the control of the low-level DES G_{lo} . Without loss of generality, the algorithm is assumed to be non-terminating, hence the use of the unconditional ‘while’ loop in the two procedures.

Procedure HI-manager (High Level)	
	begin
1	initialize
2	Initialize state in S_{hi} ;
3	Use initial state x_0 to retrieve off-line Condat f_{com} ;
4	Send f_{com} down to low level;
5	while true do
6	Wait to receive high-level event occurrence (from T);
7	Update state in S_{hi} ;
8	Use current state x to retrieve off-line Condat f_{com} ;
9	Send f_{com} down to low level;
	end

Figure 3.5: Procedural realization for high-level controller C_{hi}

Let ξ and x_0 be the transition function and initial state of the high-level supervisor automaton S_{hi} , respectively. The following notation is used in the procedures: For an evolving $t \in L(G_{hi})$ and $s \in L(G_{lo})$, $\theta(s) = t$, the current high-level state of S_{hi} is $x = \xi(t, x_0)$; the current low-level state of G_{lo} is $q = \delta(s, q_0)$; the command or high-level control data to transmit is $f_{com}(S_{hi}, x) = \text{Condat}(S_{hi}, t)$ and the translated low-level control data is

$$f_{con}(f_{com}, q) = \mu(f_{com}, s), \quad (3.4.3)$$

Procedure L0-operator (Low Level)	
	begin
1	initialize
2	Initialize state in G_{lo} ;
3	Wait to receive updated f_{com} from high level;
4	Use current f_{com} and state q to compute Condat f_{con} , and apply it to G_{lo} ;
5	while true do
6	Wait to receive low-level event occurrence (from Σ);
7	Update state in G_{lo} ;
8	if current state q is vocal then
9	Send vocalized event $V(q)$ up to high level;
10	Wait to receive updated f_{com} from high level;
11	Use current f_{com} and state q to compute Condat f_{con} , and apply it to G_{lo} ;
	end

Figure 3.6: Procedural realization for low-level controller C_{lo}

where μ is the control law (3.3.9).

To sharpen causal clarity to a finer level of ascertaining the control data from each individual event command, the law f_{con} can be refined to $f_{con}(f_{com}, q) = \bigcup_{\tau \in f_{com}} f_{con}(\tau, q)$, where $f_{con}(\tau, q) = \{\sigma \in \Sigma_c \mid \delta(\sigma, q)! \ \& \ \delta(\sigma, q) \in B_u^\tau\}$ is the τ -commanded control data set, i.e., the set of low-level events to be disabled as commanded by (the control data for virtual event) $\tau \in f_{com}$.

3.4.3 Computational Complexity

Computational Analysis for all B_u^τ , $\tau \in T_c$

Note that for an OCC DES G_{lo} , a non-vocal state that is in a set B_u^τ may not be in another set $B_u^{\tau'}$, for $\tau \in T_c$ and $\tau' \in T_c - \{\tau\}$. In what follows, the total time complexity analysis, of computing B_u^τ for all $\tau \in T_c$, is based on first determining the worst-case time complexity of constructing every B_u^τ , using Procedure BuStateSet-Compute in Figure 3.7 that does not exploit this fact. The result is a (well-defined) theoretical upper bound of the worst-case total time complexity.

To begin with, refer to Table 3.1 for the definition of n , n_{voc} , u and v . Let $a = n - n_{voc} + n_0^j$, where n_0^j is the number of vocal states outputting $\tau_{c_j} \in T_c$ under V , and u be the cardinality of the uncontrollable event set Σ_u . Then the iterative computations will produce

Figure 3.7: Procedural realization for computing B_u^τ for $\tau \in T_c$

the following complexity series:

$$(a - n_0)n_0u, (a - n_1)(n_1 - n_0)u, (a - n_2)(n_2 - n_1)u, \dots, (a - n_i)(n_i - n_{i-1})u \dots, 0,$$

where $n_0 = n_0^j$, $n_{i+1} = n_i + k_{i+1}$, with n_{i+1} and n_i denoting the cardinality of B_u^τ immediately after and before each **forloop** iteration $i \geq 0$ (from line 3 of BuStateSet-Compute in Figure 3.7), respectively; $(a - n_i)(n_i - n_{i-1})u$ is the time complexity incurred at iteration i , and $k_{i+1} \geq 0$ is the number of new non-vocal states added to the iterating B_u^τ following iteration i . The series stops at some iteration m with 0, where $n_m = n_{m-1}$ (i.e., no more non-vocal state was added at iteration $m - 1$). Let $n_{sil} = n - n_{voc}$. Then the series can be rewritten as

$$(n_{sil})n_0^j u, (n_{sil} - k_1)k_1 u, (n_{sil} - k_2)k_2 u, \dots, (n_{sil} - k_i)k_i u \dots, 0.$$

Computationally, the theoretical worst case occurs if the iterating B_u^τ increases by one new state per iteration until iteration $n_{sil} - 1$, at which $k_{n_{sil}} = 0$ (i.e., $k_{i+1} = 1$, $i < n_{sil} - 1$). In other words, the worst case occurs if the complexity series is

$$(n_{sil})n_0^j u, (n_{sil} - 1)u, (n_{sil} - 2)u, (n_{sil} - 3)u, \dots, (1)u, 0.$$

Reversing the series, the terms from 0 to $(n_{sil} - 1)u$ form an arithmetic series of n_{sil} terms with a difference of $1u$ between consecutive terms.

Summing the series and $(n_{sil})n_0^j u$ returns the result

$$\left(\frac{n_{sil}}{2}[n_{sil} - 1] + (n_{sil})n_0^j\right)u = 0.5un_{sil}(n_{sil} + 2n_0^j - 1).$$

Hence the worst-case complexity of Procedure BuStateSet-Compute is

$$O(un_{sil}^2 + (2n_0^j - 1)un_{sil})$$

or

$$O((2n_0^j - 1)u(n - n_{voc}) + u(n - n_{voc})^2)$$

for each $\tau_{c_j} \in T_c$. Hence, the overall complexity for all $\tau_{c_j} \in T_c$ is

$$O\left(u \sum_{j=1}^v [(2n_0^j - 1)(n - n_{voc}) + (n - n_{voc})^2]\right)$$

or

$$O(u(2\alpha n_{voc} - v)(n - n_{voc}) + vu(n - n_{voc})^2)$$

where $\alpha \leq 1$ and αn_{voc} refers to the total number of vocal states vocalizing events in T_c .

It follows that the worst-case total time complexity occurs in the limit where $\alpha = 1$ (i.e., $T = T_c$), and is

$$O(u(2n_{voc} - v)(n - n_{voc}) + vu(n - n_{voc})^2).$$

Off-line Complexity Reduction

It is assumed that the hierarchical structuring of a low-level DES to achieve HCN [12] (see Remark 3.1) is already applied. Under the same HCN setup for a trim low-level DES, the worst-case time complexity comparison is made between the existing design [12, 72] and the proposed design for hierarchical controller operation, as follows.

- To implement the existing controller design (realized by S_{lo} in Theorem 3.3 and the associated control data), the low-level specification automaton has to be computed [72] first for the inverse mapping θ^{-1} of the closure of the supremal controllable sublanguage of a given high-level specification $E \subseteq T^*$, before $Supcon$ and $Condat$ are computed accordingly for the low-level DES.
- To implement the proposed design (realized by the control algorithm in Figures 3.5 and 3.6 regulating f_{con} (3.4.3)), every τ -uncontrollable state set B_u^τ (3.3.1) for $\tau \in T_c$ needs to be computed for the low-level DES G_{lo} (to implement μ in Theorem 3.3 for f_{con}), along with $Supcon$ and $Condat$ for the high-level DES G_{hi} .

The various component complexities and the overall complexity are summarized in Table 3.1. The complexity of B_u^τ for all $\tau \in T_c$ is based on the preceding analysis. The rest are based on the results in [12, 72, 132].

The low-level specification automaton generating the language $\theta^{-1}(L(S_{hi})) \subseteq L(G_{lo})$, $S_{hi} = Supcon(G_{hi}, E)$, is constructed using the method developed in [72, p. 58], and is assumed to be of minimal state cardinality for the complexity analysis. This method, in the final step, obtains the trim specification automaton by computing a natural projection $P : \overline{L_a} \rightarrow \Sigma^*$, where $L_a \subseteq (\Sigma \cup T)^*$ is the prefix-closed language generated by the (trim) Cartesian product of two automata. One automaton is a T -embedded and completely state-marked version of the trim DES G_{lo} , and the other is a Σ -self-looped version of the automaton that is S_{hi} but is completely state-marked to model $L(S_{hi})$. Both these versions are computed in earlier steps of the method. For an optimistic (or best) worst-case complexity analysis of the existing design, it is assumed that the projection P is an observer of the (closed) language L_a . By the complexity results in [132] and using the notation defined in Table 3.1, the exponential worst case of computing natural projection, in time complexity of $O(2^{n_e n_{hi}(n_{voc} + n)})$ and returning an automaton of state cardinality $O(2^{n_e n_{hi}(n_{voc} + n)})$, can then be avoided. In fact, the automaton returned has worst-case state cardinality of only $n_e n_{hi}(n_{voc} + n)$. However, this projection step alone still incurs a high polynomial time com-

plexity of $O((n_e n_{hi}(n_{voc} + n))^4 + l(n_e n_{hi}(n_{voc} + n))^5 + l^2(n_e n_{hi}(n_{voc} + n))^7)$, as derived based on the complexity result in [132] for such an observer P .

Component $O(\cdot)$	For existing design [12, 72] [Figure 3.1(a)]	For proposed design [Figure 3.1(b)]
High-level <i>Supcon</i>	$l_{hi}(n_e n_{hi})^2$	$l_{hi}(n_e n_{hi})^2$
High-level <i>Condat</i>	—	$vn_e n_{hi}$
θ^{-1} mapping	$n_{voc} + ln_e n_{hi} +$ $n_e n_{hi}(n_{voc} + n) +$ $(n_e n_{hi})^4(n_{voc} + n)^4 +$ $l(n_e n_{hi})^5(n_{voc} + n)^5 +$ $l^2(n_e n_{hi})^7(n_{voc} + n)^7$	—
Low-level <i>Supcon</i>	$l(n_e n_{hi}(n_{voc} + n))^2$	—
Low-level <i>Condat</i>	$cn_e n_{hi}(n_{voc} + n)$	—
B_u^τ for all $\tau \in T_c$	—	$u(2n_{voc} - v)(n - n_{voc}) +$ $vu(n - n_{voc})^2$
Overall complexity	$n_{voc} + cn_e n_{hi}(n_{voc} + n) +$ $ln_e n_{hi}(1 + n_{voc} + n) +$ $n_e^2 n_{hi}^2(l_{hi} + l(n_{voc} + n)^2) +$ $n_e^4 n_{hi}^4(n_{voc} + n)^4 +$ $ln_e^5 n_{hi}^5(n_{voc} + n)^5 +$ $l^2 n_e^7 n_{hi}^7(n_{voc} + n)^7$	$vn_e n_{hi} + l_{hi} n_e^2 n_{hi}^2 +$ $u(2n_{voc} - v)(n - n_{voc}) +$ $vu(n - n_{voc})^2$

Nomenclature:

For G_{lo} : Cardinality	For G_{hi} : Cardinality
n : state set of G_{lo} with event set Σ	n_{hi} : state set of G_{hi} with event set T
n_{voc} : vocal state set of G_{lo}	n_e : state set of the trim automaton for E
l : event set Σ	l_{hi} : event set T
c : controllable event set Σ_c	v : controllable event set T_c
u : uncontrollable event set Σ_u	

Table 3.1: Control operational design of a high-level regular specification language $E \subseteq T^*$ for a HCN pair (G_{lo}, G_{hi}) : An off-line, worst-case time complexity comparison

The hierarchical construction for HCN of (G_{lo}, G_{hi}) [12] entails structuring the reporter map θ as an observer. It is assumed that G_{hi} is a minimal-state recognizer of the language $\theta(L_m(G_{lo}))$. From the results on observers [133] and using the notation defined in Table 3.1, it then follows that $n_{hi} \leq n$. Thus, implementing the proposed design incurs an off-line control synthesis complexity that is significantly lower in general due to the following:

1. the smaller state cardinality of the high-level DES involved in *Supcon* synthesis instead of the low-level DES;
2. the τ -uncontrollable state sets which can be computed in polynomial time complexity in the (non-initial) non-vocal state cardinality of the low-level DES; and
3. not having at all to compute the low-level specification automaton for the inverse mapping θ^{-1} of the closure of the supremal controllable sublanguage of a given high-level specification language.

Besides, computing τ -uncontrollable state sets is a one-off exercise for a given DES G_{lo} . For a different high-level specification, only the high-level supervisor needs to be recomputed. In this sense, B_u^τ (3.3.1) for $\tau \in T_c$ is reusable.

Low On-line Complexity

Not surprisingly, the potentially significant savings in off-line synthesis comes at the expense of incurring on-line low-level control (data) computation. Fortunately, however, this on-line time complexity is generally low, as explained below.

In the control algorithm, f_{con} (3.4.3) is recomputed each time an event in G_{lo} occurs, and in absorbing the cardinality of Σ_c in the constants of $O(\cdot)$, each on-line computation incurs, in the worst case, only a linear time complexity of $O(r)$, where r is the cardinality of the set $B_u^{f_{com}}$, given $f_{com} \subseteq T_c$.

In a related but different work [68], a baseline procedure for computing the control law (3.3.8) on-line has linear time complexity of $O(n)$ in the worst case, where n is the state cardinality of the low-level DES G_{lo} . When compared with this procedure, computing the

equivalent law f_{con} in $O(r)$ as used in the proposed design is faster. This is because the state cardinality r of a set $B_u^{f_{com}}$ is often very much smaller than the state cardinality n of the low-level DES.

3.5 Conclusion

In this chapter, based on the results of output control feasibility and system decomposition, the command and control design proposed for hierarchical control of logical DES's is shown to be implementable. The implementation entails the off-line computation of the high-level controlled DES and the reusable τ -uncontrollable state sets, instead of the larger scale, low-level controlled DES. This operational design is attractive as it offers the intrinsic merit of furnishing causal clarity of command over control during on-line operation, along with off-line complexity reduction and fast on-line computation. These practical advantages are useful for engineering control problems, where the required runtime operational mode is command and control, or the control solution needs to be subjected to validation tests by simulation and observation.

Chapter 4

System Abstraction for Real-Time Hierarchical Control: Consistency Formulation

In the general framework of formal languages and finite (or finite-state) automata, the seminal concept of hierarchical consistency of DES's [1] in a two-level control setup is suitably extended to TDES's in this chapter.

Based on architecturally the same two-level setup as the hierarchical control of logical DES's [1] reviewed, for the hierarchical control of TDES's, the low-level system is modeled by a Moore automaton, but that is constructed from a TDES (automaton) model proposed in [74], and a timed hierarchical reporter map modeling the information channel via which the low level drives the high level.

In this chapter, the research contributions presented include new timed concepts and theoretical conditions for the hierarchical consistency of TDES's. Contributions on hierarchical system synthesis of TDES's are presented in the next chapter.

This chapter is organized as follows. Section 4.1 overviews the state-of-the-art control research in TDES's. Section 4.2 follows up with a Moore system formulation for two-level

hierarchical control, and explains the need for system output-time fidelity. Section 4.3 defines the constituent concepts for the system concept of timed output-control consistency, by which the system abstraction at the high level possesses a natural timed control structure as the system at the low level. Together with these constituent concepts, Section 4.4 adds a timed concept of partner-freeness to formulate system sufficiency structures for hierarchical consistency between the two levels, without and with output-time fidelity guarantee. Section 4.5 concludes the chapter.

4.1 Control of TDES's: A State of the Art

There are several real-time control approaches using different TDES models proposed in the literature. Besides ‘timed transition graphs’ [74], other notable TDES models include ‘clock automata’ [134], ‘timed transition (semantics) models’ [6], ‘timed state automata’ [135], ‘timed automata’ used in [7, 8], and ‘timed Petri nets’ [136]. Timed transition graphs are a specialized class of finite automata formulated as system models for real-time supervisory control of TDES's in [74], and are able to represent a variety of timing issues for a useful range of control problems [12]. Despite the well-known complexity shortcomings of timed transition graphs as TDES models [137, 138], the real-time nonblocking control theory [74] and its subsequent developments form a mathematically rigorous body of conceptually rich work based on this graph model. These developments include work on supervisor reduction [138], efficient control synthesis using binary decision diagrams (BDD's) [139], control under partial observation [40, 140], nonblocking control with communication delay [141], specification automaton transparency for validation [142] and translation [143] from a class of real-time temporal logic, decentralized control [144, 145, 146], modular control [147, 148], localized or distributed control [149] and that with communication delay [150], and hierarchical control [78, 151]. In our research, we add to this intellectually promising body of real-time control research by extending the monolithic control theory for TDES's [74] to hierarchical control. The contributions include a number of new timed concepts

to support hierarchical consistency with output-time fidelity, within the same elementary framework and computational foundation for formal languages and finite automata.

Despite its importance as a control architecture, there is relatively little work on hierarchical control in a real-time framework. One related early effort [78] extends the bottom-up design approach of hierarchical control of logical DES's [1, 67, 12] to a timed version. In [78], the low and the high level in a hierarchy are each modeled by a timed transition (graph) model. The low-level system is a Moore TDES and the high-level system is an virtual abstraction of the low-level system. Earlier in Chapter 2, we have unearthed important system properties of time, namely, time is unstopable and does not halt an executing event, that are implicit in the TDES model from [74] that we adopt for our research. However, in [78], unlike the timed transition graph formulation of the TDES model [74], the system property of time not halting an executing event is relaxed for both levels of the hierarchy. Because of this relaxation, the tick event at both levels is akin to timeout in general, and may be treated like any other event. The research therefore does not consider system time fidelity, and in this aspect is fundamentally different from our work.

Another related effort [151, 152] extends the top-down design approach of hierarchical control using state tree structures [153] to a real-time version. In this approach, a given TDES model is of the type [74] possessing time fidelity, and is encoded (equivalently) into a timed state tree structure without time aggregation or abstraction for efficient BDD-based control synthesis. System time fidelity is a non-issue in this computational approach.

4.2 Towards a TDES Hierarchy with Time Fidelity

In a two-level hierarchical setup as in [1], the low-level TDES needs to be equipped with an output function that drives the high-level TDES model. To model a class of such low-level TDES's, a Moore automaton [71] is used.

4.2.1 Low-level TDES Modeling for Hierarchical Control

In general, a TDES model G (2.4.2) with event set Σ needs to be re-structured into a Moore automaton (G_{lo}, V) - an automaton^{4.1} $G_{lo} = (Q, \Sigma, \delta, q_0, Q_m)$ associated with an information channel defined by a vocalization map $V : Q \rightarrow T \dot{\cup} \{\tau_o\}$, where $T = T_{act} \dot{\cup} \{t_h\}$ - such that $L(G_{lo}) = L(G)$ and $L_m(G_{lo}) = L_m(G)$. T_{act} denotes the high-level (virtual) activity event set, t_h denotes a time aggregation of low-level ticks of the global clock in TDES G_{lo} , and the symbol τ_o denotes a ‘silent output’. For the low-level TDES G_{lo} , we henceforth replace *tick* by t_l to distinguish it as a low-level atomic time tick; therefore, $\Sigma = \Sigma_{act} \dot{\cup} \{t_l\}$.

Let w^n denote a string of $n \in \mathbb{N}$ consecutive occurrences of string w , with $w^0 = \varepsilon$; and w^* denote strings of finitely many occurrences of string w such that we write $sw^* \in L(G_{lo})$ if, for all $n \geq 0$, $sw^n \in L(G_{lo})$, and is such that $\delta(w, q) = q$, where $q = \delta(s, q_0) \in Q$. Then the Moore construction [71] of G_{lo} for the TDES G is based on a given timed reporter map - a virtual projection $\theta : L(G) \rightarrow T^*$, defined such that $\theta(\varepsilon) = \varepsilon$ and, for $\sigma \in \Sigma$ and $s\sigma \in L(G)$, $\theta(s\sigma)$ is either $\theta(s)$ or $\theta(s)\tau$ for some $\tau \in T$. The given map θ obeys the following time-output design laws:

Law 1: For $s(s't_ls'')^* \in L(G)$, and $s', s'' \in \Sigma^*$, $\theta(s(s't_ls'')^n) = \theta(s)(t't_h t'')^n$ for all $n \geq 0$, where $t', t'' \in T^*$.

Law 2: For $\sigma \in \Sigma$ and $s\sigma \in L(G)$, $\theta(s\sigma) = \theta(s)t_h \implies \sigma = t_l$.

The constraint by Law 1 means that G_{lo} must be constructed such that whenever a state $q = \delta(s, q_0)$ in G_{lo} has, traversing through it, a loop string containing a low-level time tick t_l , i.e., $\delta(s't_ls'', q) = q$ for some $s', s'' \in \Sigma^*$, the loop string $s't_ls''$ must traverse through a state in G_{lo} that outputs or vocalizes a high-level time tick t_h . In this sense, G_{lo} is t_h -responsive. The constraint by Law 2 means that the high-level tick t_h is a time output, in that it must be real time-driven, i.e., t_h is always a vocalization that immediately follows

^{4.1}Although the same 5-tuple notation is used as in Section 2.3.1, it should be clear in the context that the structure of G_{lo} is in general not the same as that of a given TDES G .

the execution of a low-level tick t_l in G_{lo} . With θ obeying the time-output design laws, the low-level TDES G_{lo} constructed is said to be time-output responsive.

For the constructed G_{lo} , the vocalization map V for every $s' \in L(G_{lo})$ is defined by

$$V(\delta(s', q_0)) = \begin{cases} \tau_o, & \text{if } s' = \varepsilon \text{ or } \delta(s', q_0) \notin Q_{voc} \\ \tau \in T, & \text{otherwise,} \end{cases}$$

where the selected subset $Q_{voc} \subseteq Q$, called vocal state set, is defined as follows. For $\sigma \in \Sigma$ and $s' = s\sigma$,

$$\delta(s\sigma, q_0) \begin{cases} \notin Q_{voc}, & \text{if } \theta(s\sigma) = \theta(s) \\ \in Q_{voc}, & \text{if } \theta(s\sigma) = \theta(s)\tau. \end{cases}$$

A conceptual procedure applicable for constructing a Moore TDES (G_{lo}, V) from a given TDES G and a reporter map θ , or simply a TDES (G, θ) , is prescribed in [1, 12]. In the graphical representation of G_{lo} and any Moore automaton in general, every vocal state is represented by a node containing the symbol of an event that it vocalizes.

The inverse reporter map for $t \in T^*$ is now defined as follows: $\theta^{-1}(t) = \{s \in L(G_{lo}) \mid \theta(s) = t\}$. In what follows, extending θ and θ^{-1} to $\theta(K) \subseteq T^*$ for $K \subseteq L(G_{lo})$ and $\theta^{-1}(E) \subseteq L(G_{lo})$ for $E \subseteq T^*$, respectively, we have: $\theta(K) = \{\theta(s) \mid s \in K\}$, and $\theta^{-1}(E) = \bigcup_{t \in E} \theta^{-1}(t)$.

The Moore automaton (G_{lo}, V) is simply referred to as G_{lo} when V is understood. Under the map V , G_{lo} outputs events in T to drive some high-level θ -image model G_{hi} whenever it reaches a vocal state $q \in Q_{voc}$, and otherwise outputs the silent symbol $\tau_o \notin T$ to signal no ‘significant’ change for the high level. Formally, model G_{hi} , the high-level image of G_{lo} , is an automaton such that $L(G_{hi}) = \{\theta(s) \mid s \in L(G_{lo})\}$ and $L_m(G_{hi}) = \{\theta(s) \mid s \in L_m(G_{lo})\}$. G_{hi} is said to generate events of T under the θ -map on $L(G_{lo})$. The pair (G_{lo}, G_{hi}) represents a two-level TDES hierarchy.

The vocal language of G_{lo} , denoted by $L_{voc}(G_{lo})$, is

$$L_{voc}(G_{lo}) = \{s \in L(G_{lo}) \mid s = \varepsilon \text{ or } \delta(s, q_0) \in Q_{voc}\},$$

which is the sublanguage of $L(G_{lo})$ containing the empty string ε and all the strings of $L(G_{lo})$, called vocal strings, that end in a state of Q_{voc} . In a richer characterization, let an arbitrary vocal string $s \in L(G_{lo})$, denoted by

$$s = \langle s', \sigma_i, x_i, k, \tau \rangle, \quad (4.2.1)$$

to be of the form

$$s = s' \sigma_1 \sigma_2 \cdots \sigma_k \text{ with } \sigma_i \in \Sigma (1 \leq i \leq k),$$

such that, where $s' = \varepsilon$,

- $V(\delta(\sigma_1 \sigma_2 \cdots \sigma_i, q_0)) = \tau_o \quad (1 \leq i \leq k-1),$
- $V(\delta(s, q_0)) = \tau \in T,$
- $x_0 = \delta(\varepsilon, q_0),$
- $x_i = \delta(\sigma_1 \sigma_2 \cdots \sigma_i, q_0) \quad (1 \leq i \leq k),$

and where $s' \in \Sigma^+$,

- $V(\delta(s', q_0)) \in T,$
- $V(\delta(s' \sigma_1 \sigma_2 \cdots \sigma_i, q_0)) = \tau_o \quad (1 \leq i \leq k-1),$
- $V(\delta(s, q_0)) = \tau \in T,$
- $x_0 = \delta(s', q_0),$
- $x_i = \delta(s' \sigma_1 \sigma_2 \cdots \sigma_i, q_0) \quad (1 \leq i \leq k).$

In every $s = \langle s', \sigma_i, x_i, k, \tau \rangle$ (4.2.1) of $L(G_{lo})$, $s' \in L(G_{lo})$ is called the reference prefix of string s , and is an empty string if x_0 is the initial low-level system state $q_0 \in Q$. Such a string $s \in L(G_{lo})$ is called a τ -string and has a suffix $\sigma_1\sigma_2 \cdots \sigma_k$ that runs from the initial state or a vocal state, via non-vocal states of G_{lo} , to a vocal state outputting the high-level event $\tau \in T$. This suffix is called the co-silent string of s (4.2.1).

As reviewed in Chapter 2, three of the qualitative temporal properties of the TDES model G (2.4.2), and hence of its Moore construction G_{lo} , are Properties 2.1 to 2.3, relating, respectively, to the ‘persistence of time’, the ‘time invariance of event eligibility’ and the ‘uncontrollable persistence of time’.

A fundamental result for the Moore TDES model G_{lo} follows.

Lemma 4.1. $L(G_{lo}) = \overline{L_{voc}(G_{lo})}$.

Proof. For a TDES model G_{lo} , that $\overline{L_{voc}(G_{lo})} \subseteq L(G_{lo})$ is straightforward. It remains to show that $L(G_{lo}) \subseteq \overline{L_{voc}(G_{lo})}$, as follows: By Property 2.1 and the finiteness of state set Q of the TDES G_{lo} , every $s'_p \in L(G_{lo})$ can be extended to some $s'w^* \in L(G_{lo})$, where $s'_p \leq s'$ and $w \in \Sigma^+$. Then since $\delta(w, q) = q$, where $q = \delta(s', q_0) \in Q$, by ALF (2.4.5) of G_{lo} , i.e., the fact that

$$(\forall q \in Q)(\forall s \in \Sigma_{act}^+, \delta(s, q)!) \delta(s, q) \neq q,$$

string $w \notin \Sigma_{act}^+$ and hence contains a tick t_l . By design Law 1 of the reporter map θ from which the time-output responsive G_{lo} is constructed, for $s'w^* \in L(G_{lo})$, it necessarily follows that $\theta(s'w) = \theta(s')t_1t_h t_2$, where $t_1, t_2 \in T^*$. Therefore, $\theta(s')t_1t_h \in L(G_{hi})$, and since $\theta(L(G_{lo})) = L(G_{hi})$, there exists a t_h -string $s'' \in L_{voc}(G_{lo})$ with $s' \leq s''$ such that $\theta(s'') = \theta(s')t_1t_h$. Then, since $s'_p \leq s'$, and therefore $s'_p \leq s''$, it follows that $s'_p \in \overline{L_{voc}(G_{lo})}$.

Hence the lemma. □

Two propositions for a TDES hierarchy (G_{lo}, G_{hi}) may now be presented.

Proposition 4.1. *Given a TDES hierarchy (G_{lo}, G_{hi}) , G_{hi} is activity-loop free.*

Proof. Given a TDES hierarchy (G_{lo}, G_{hi}) , where $G_{hi} \stackrel{\text{def}}{=} (X, T, \xi, x_0, -)$. By Property 2.1 and the finiteness of state set Q of the TDES G_{lo} , every $s'_p \in L(G_{lo})$ can be extended to some $s'w^* \in L(G_{lo})$, where $s'_p \leq s'$ and $w \in \Sigma^+$. Then since $\delta(w, q) = q$, where $q = \delta(s', q_0) \in Q$, by ALF (2.4.5) of G_{lo} , i.e., the fact that

$$(\forall q \in Q)(\forall s \in \Sigma_{act}^+, \delta(s, q)!) \delta(s, q) \neq q,$$

string $w \notin \Sigma_{act}^+$ and hence contains a tick t_l . By design Law 1 of the reporter map θ from which the time-output responsive G_{lo} is constructed, it follows that, for $s'w^* \in L(G_{lo})$, $\theta(s'w^n) = \theta(s')(t_1 t_h t_2)^n$ for all $n \geq 0$, where $t_1, t_2 \in T^*$. Since state set X of G_{hi} is finite, there exist an $n_1 \geq 0$ and an $n_2 \geq 1$ such that for all $n \geq 0$, $\theta(s')(t_1 t_h t_2)^{n_1} t_0^n \in L(G_{hi})$, where $t_0 = (t_1 t_h t_2)^{n_2}$, and is such that $\xi(t_0, x') = x'$, where $x' = \xi(\theta(s')(t_1 t_h t_2)^{n_1}, x_0) \in X$ (and we can thus write $\theta(s')(t_1 t_h t_2)^{n_1} t_0^* \in L(G_{hi})$). Therefore, $(\forall x \in X)(\forall t \in T^+, \xi(t, x)!) ((\xi(t, x) = x) \implies t \notin T_{act}^+)$. By contraposition,

$$(\forall x \in X)(\forall t \in T_{act}^+, \xi(t, x)!) \xi(t, x) \neq x.$$

Hence the proposition. □

Proposition 4.2. *Given a TDES hierarchy (G_{lo}, G_{hi}) , G_{hi} obeys Property 2.1.*

Proof. Given a TDES hierarchy (G_{lo}, G_{hi}) , where $G_{hi} \stackrel{\text{def}}{=} (X, T, \xi, x_0, -)$. Then, for an arbitrary $t \in L(G_{hi})$ and hence an arbitrary state $x = \xi(t, x_0) \in X$, we need to prove that Property 2.1 for G_{hi} , i.e.,

$$(T_{act}(x) = \emptyset) \implies \xi(t_h, x)!$$

holds, as follows: Since $\theta(L(G_{lo})) = L(G_{hi})$, there must exist a string $s \in L_{voc}(G_{lo})$ such that $\theta(s) = t$. By Property 2.1 for G_{lo} , there exists some $\sigma \in \Sigma$ such that $s\sigma \in L(G_{lo})$. Furthermore, by Lemma 4.1, there must also exist a string $w \in \Sigma^*$ such that $s\sigma w \in L_{voc}(G_{lo})$ and is some τ -string of $L(G_{lo})$, where $\tau \in T$ and string s is its reference prefix. With $\theta(s\sigma w) = t\tau$ and thus $\xi(\tau, \xi(t, x_0))!$, it follows that if $T_{act}(\xi(t, x_0)) = \emptyset$, then $\tau \notin T_{act}$

which means $\tau \in T - T_{act}$, i.e., $\tau = t_h$. Hence the proposition. $\square$

In other words, the passage of aggregated time, as represented by the ticking of t_h , is continual in the (uncontrolled) high-level TDES G_{hi} , in that the tick t_h is always eligible in the absence of activity events at a state of G_{hi} .

4.2.2 System Abstraction: Need for Output-Time Fidelity

The transition of a high-level time tick $t_h \in T$ in a system model abstraction [78] denotes the passage of some low-level time ticks of t_l . Time abstraction (or state vocalization of $t_h \in T$) is qualitative if it signals an amount of low-level time elapsed that is possibly irregular but deemed important by hierarchical design, in which case it is said to apply a non-periodic timescale (between the high and low level). Time abstraction is quantitative if a periodic timescale $1 : n$ reminiscent of that in [138] is applied, which is a fixed time ratio of 1 high-level tick of t_h for every n low-level ticks of t_l , where integer $n \geq 1$. However, be it qualitative or quantitative, to lay a sound design foundation for timed hierarchical control, we postulate that high-level time (or output-time) fidelity must also be upheld in the control design of a hierarchical abstraction for a base or low-level TDES model under the real-time control-theoretic setting [74] reviewed. This is so that the event timing feature, of *specifying a real-time requirement for control that is naturally in congruence with time fidelity of the TDES model* as laid in [74], can be extended to system abstraction. By this, we mean that a real-time specification such as ‘an activity event must complete execution within one time tick since it started’ can be prescribed in terms of $t_h \in T$ for the system abstraction, with the sublanguage generated by the high-level timed specification on the system abstraction not violating the intended high-level timing semantics of the specification. Otherwise, we would often need to go beneath the abstraction level to examine or re-examine the low-level Moore system structure, to ascertain if desired timing requirements are correctly prescribed.

Example 4.1. Consider an example depicted in Figure 4.1. For the TDES G given in Figure 4.1(a), two system abstractions are proposed, as shown in Fig 4.1(b). Both the

abstractions possess ALF and Property 2.3^{4.2}. But Abstraction 1 violates Property 2.2, whereas Abstraction 2 does not and hence possesses time fidelity.

Now, to illustrate the need for time fidelity in system abstraction, consider a high-level specification *TTG Spec*, as shown in Figure 4.1(c). It asserts that a high-level activity event τ must complete execution in not more than one high-level time tick upon event eligibility or initiation. As shown in Figure 4.1(d), the sublanguage due to the specification on Abstraction 1 (without time fidelity) is represented by *TTG 1*, whereas that due to the same specification on Abstraction 2 (with time fidelity) is represented by *TTG 2*. Clearly, the timing semantics of *TTG 1* is incorrect or unsound against the intended timing requirement of ‘at most one tick for τ -completion’ prescribed by *Spec*, as τ appears disabled after a high-level tick, whereas *TTG 2* is. ■

^{4.2}Note that in this example, the high-level tick t_h is eligible at every state in both the system abstractions, hence their satisfying Property 2.3.

(a) Example TDES G

(b) Moore TDES's constructed from G under different timed reporter maps

(c) High-level abstractions of G and a high-level specification TTG, Spec

(d) TTG's of the high-level system sublanguages generated for Spec

Figure 4.1: Implications on the timing semantics of a high-level specification on system abstractions without and with high-level time fidelity: An illustration

To specify real-time high-level specifications for hierarchical control without incorrect high-level timing semantics due to the system abstraction, the problem of interest is to construct not only a system abstraction possessing time fidelity such as Abstraction 2, but also one endowed with a natural timed control structure that subsumes time fidelity, as will be elaborated in the next section. Put simply, our intent is to preserve real-time system dynamics at the abstraction or high level with conceptually the same real-time control-theoretic setting as assumed or given at the low level. Reviewed in Chapter 2, the assumed setting is the real-time system control basis under which the control synthesis method for TDES's [74] is developed. Importantly, it is the necessary basis on which an arbitrary proper control specification for a given TDES can always be stated. A real-time specification is said to be proper if, in conjunction with a given TDES, it generates a sublanguage of sound timing semantics (against the specified high-level timings), for *real-time* and not simply *timed* control synthesis, against which the supervisor synthesized can be unambiguously understood in terms of permitting or restricting the specified real-time durations for activity events.

4.3 Timed Control Structure

To admit control for a TDES hierarchy (G_{lo}, G_{hi}) , the high-level activity event set T_{act} of G_{hi} is partitioned into the prohibitible event set T_{hib} and the uncontrollable event set T_u , and into the forcible event set T_{for} and the non-forcible event set $T_{act} - T_{for}$; with controllable event set $T_c = T_{hib} \dot{\cup} \{t_h\}$. However, the two high-level control-theoretic partitions may not be unambiguous and correct under a given control-theoretic setting and (transition) structure of G_{lo} . Even if they do, the structure of G_{hi} is a TTG that might not possess time fidelity (w.r.t t_h), although the TDES G_{lo} , constructed from a given TDES G (2.4.2) and a reporter map θ , does (w.r.t t_l).

For real-time high-level control of (G_{lo}, G_{hi}) , G_{lo} in general needs to be structurally refined so that G_{hi} is endowed with a natural timed control structure (w.r.t subsets T_{hib} , T_u ,

T_{for} , $T_{act} - T_{for}$ and $\{t_h\}$), i.e., so that every high-level event $\tau \in T_{act}$ defined and output by G_{lo} is unambiguously prohibitable or uncontrollable if it is in T_{hib} or in T_u , respectively, and is unambiguously forcible or non-forcible if it is in T_{for} or in $T_{act} - T_{for}$, respectively, and the time tick $t_h \in T$ is T -uninterrupting. The Moore transition structure of the TDES G_{lo} is defined to be timed output-control consistent if G_{hi} possesses such a natural timed control structure.

In what follows, we present the theoretical development of the fundamental system concept of timed output-control consistency, to lay a time fidelity foundation for feasible hierarchical control of TDES's. We first conceptualize and explain the essence of the formulation: The TDES G_{lo} is output-control consistent (OCC) if it is activity output-control consistent (AOCC) and output-force consistent (OFC); and timed OCC (TOCC) if it is OCC and output time-compliant (OTC):^{4.3}

- Being AOCC means that every high-level event $\tau \in T_{act}$ (defined, and output by G_{lo}) is unambiguously prohibitable or uncontrollable. As stated in Proposition 4.3 below, for an AOCC G_{lo} , Property 2.3 holds for the resultant G_{hi} .
- Being OFC means that every high-level event $\tau \in T_{act}$ is unambiguously forcible or non-forcible. By the standard model of event forcing [74], a high-level event is unambiguously forcible provided, when enforced, it can always preempt the time tick t_h .
- Being OTC means that, in driving G_{hi} from below via the vocalization map V , the high-level time tick evolution must not render an eligible high-level activity event ineligible, as with the respect accorded to low-level ticks and activity events by the low-level model G_{lo} . As stated in Proposition 4.5 below, for an OTC G_{lo} that is also AOCC, the resultant G_{hi} is a TDES model with time fidelity.

The formulation of these three concepts entails the system definition of vocal string structure (4.2.1).

^{4.3}Note that the abbreviations of concepts introduced herewith and later are in adjectival form.

4.3.1 Activity Output-Control Consistency

Definition 4.1 (Activity output-control consistency). A TDES G_{lo} is said to be AOCC if, for every τ -string $\langle s', \sigma_i, x_i, k, \tau \rangle \in L(G_{lo})$ with $\tau \in T_{act}$, it is the case that

- if $\tau \in T_{hib}$, then $\sigma_1 \cdots \sigma_k$ is controllable, i.e., for some i ($1 \leq i \leq k$), $\sigma_i \in \Sigma_{hib}$ or $(\sigma_i = t_l \ \& \ \Sigma(x_{i-1}) \cap \Sigma_{for} \cap \Sigma_u \neq \emptyset)$,
- if $\tau \in T_u$, then $\sigma_1 \cdots \sigma_k$ is uncontrollable, i.e., for all i ($1 \leq i \leq k$), $\sigma_i \in \Sigma_u$ or $(\sigma_i = t_l \ \& \ \Sigma(x_{i-1}) \cap \Sigma_{for} = \emptyset)$.

For an AOCC G_{lo} , $\tau \in T_{hib}$ only if there is a low-level event in the co-silent string, of every τ -string of $L(G_{lo})$, which is prohibitable or is a tick t_l at a state where an uncontrollable and forcible event is also eligible. $\tau \in T_u$ only if every low-level event in the co-silent string, of every τ -string of $L(G_{lo})$, is uncontrollable or is a tick t_l at a state where no forcible event is eligible.

Time, represented by tick t_h , is uncontrollably persistent in the high-level abstraction G_{hi} of an AOCC G_{lo} , and this fact is formalized in Proposition 4.3.

Proposition 4.3. Given a TDES hierarchy (G_{lo}, G_{hi}) and that G_{lo} is AOCC, G_{hi} obeys Property 2.3.

Proof. Given a TDES hierarchy (G_{lo}, G_{hi}) , where G_{lo} is AOCC and $G_{hi} \stackrel{\text{def}}{=} (X, T, \xi, x_0, -)$. Then, for an arbitrary string $t \in L(G_{hi})$ and hence an arbitrary state $x = \xi(t, x_0) \in X$, we need to prove Property 2.3 for G_{hi} , i.e.,

$$(T_{act}(x) \cap T_u = \emptyset) \implies \xi(t_h, x)!$$

holds, as follows: Since $\theta(L(G_{lo})) = L(G_{hi})$, there must exist a string $s \in L_{voc}(G_{lo})$ such that $\theta(s) = t$. By Property 2.1 for G_{lo} , there exists some $\sigma \in \Sigma$ such that $s\sigma \in L(G_{lo})$. Furthermore, by Lemma 4.1, there must also exist a string $w \in \Sigma^+$, where $\sigma \leq w$, such that $sw \in L_{voc}(G_{lo})$ and is some τ -string of $L(G_{lo})$, where $\tau \in T$ and string s is its reference

prefix. It then follows that one such string $w = \sigma_1\sigma_2 \cdots \sigma_k$ exists that is uncontrollable or ambiguously controllable, i.e., it contains only uncontrollable activity events or t_l 's that are not unambiguously preemptable, found as follows:

Let $\sigma_0 = \varepsilon$. Then for each i , ($0 \leq i \leq k-1$), if $\Sigma(\delta(s\sigma_0\sigma_1\sigma_2 \cdots \sigma_i, q_0)) \cap \Sigma_u \neq \emptyset$, select a $\sigma_{i+1} \in \Sigma_u$ such that $\sigma_{i+1} \in \Sigma(\delta(s\sigma_0\sigma_1\sigma_2 \cdots \sigma_i, q_0))$. Otherwise, select a $\sigma_{i+1} = t_l$ that is defined by Property 2.3 for G_{lo} , such that $\sigma_{i+1} \in \Sigma(\delta(s\sigma_0\sigma_1\sigma_2 \cdots \sigma_i, q_0))$.

With $\theta(s\sigma_1\sigma_2 \cdots \sigma_k) = t\tau$, it is clear that $\xi(\tau, \xi(t, x_0))!$. Because G_{lo} is AOCC, by Definition 4.1, $\tau \notin T_{hib}$, since the co-silent string $w = \sigma_1\sigma_2 \cdots \sigma_k$ of the τ -string $sw \in L(G_{lo})$ is either uncontrollable, implying $\tau \in T_u$, or ambiguously controllable, implying $\tau = t_h$. Therefore, it can only be that $\tau \in T_u \cup \{t_h\}$. It thus follows that if $T_{act}(\xi(t, x_0)) \cap T_u = \emptyset$, then $\tau \notin T_u$, implying that $\tau = t_h$. Hence the proposition. $\square$

4.3.2 Output-Force Consistency

We first define and explain two supporting concepts, before defining the structure of an OFC G_{lo} .

Definition 4.2 (Preemptability of t_h by $\tau \in T_{act}$). *Consider an arbitrary τ -string (4.2.1) of $L(G_{lo})$ with reference prefix $s' \in L(G_{lo})$ and $\tau \in T_{act}$. Then t_h is said to be unambiguously preemptable w.r.t (s', τ) if, for every t_h -string $\langle w, \alpha_j, z_j, h, t_h \rangle \in L(G_{lo})$ such that $\theta(w) = \theta(s')$ and $w \in L(G_{lo})$ is the reference prefix of some τ -string (4.2.1) of $L(G_{lo})$, there exists a τ -string $\langle w, \sigma_i, x_i, k, \tau \rangle \in L(G_{lo})$, with $\alpha_0\alpha_1 \cdots \alpha_p = \sigma_0\sigma_1 \cdots \sigma_p$ for some p ($0 \leq p < \min(h, k)$) where $\alpha_0 = \sigma_0 = \varepsilon$, and $z_n \notin \{x_i \mid (0 \leq i \leq k-1)\}$ for all n ($p < n < h$), such that*

- $\alpha_{p+1} \in \Sigma_{hib}$ or $(\alpha_{p+1} = t_l \ \& \ \sigma_{p+1} \in \Sigma_{for})$.

Consider an arbitrary nonempty string $r = \alpha_{p+1}\alpha_{p+2} \cdots \alpha_h$ that leads, from a state x_p lying along the transitions defined by the co-silent string $\sigma_1\sigma_2 \cdots \sigma_k$ of some τ -string of $L(G_{lo})$, $\tau \in T_{act}$, that exists, to a vocal state outputting t_h , via subsequent non-vocal states that are not lying along the transitions defined by the co-silent string. Then, in words, w.r.t

the reference prefix s' of a τ -string of $L(G_{lo})$, t_h is unambiguously preemptable if every t_h -string of $L(G_{lo})$ with reference prefix w , such that $\theta(w) = \theta(s')$ and string w is also the reference prefix of some τ -string of $L(G_{lo})$, has such a string $\sigma_1\sigma_2\cdots\sigma_p r$ if $0 < p < \min(h, k)$ and r if $p = 0$, as its suffix, with α_{p+1} in string r either a prohibitable event or a t_l which can be preempted by a forcible event σ_{p+1} that lies along the co-silent string of the τ -string that exists.

Definition 4.3 (Preemptability of t_h by τ -mirage, $\tau \in T_{act}$). *Consider an arbitrary τ -string (4.2.1) of $L(G_{lo})$ with reference prefix $s' \in L(G_{lo})$ and $\tau \in T_{act}$. Then t_h is said to be unambiguously preemptable w.r.t the mirage of (s', τ) if, for all $w \in L(G_{lo})$, $w = \varepsilon$ or $\delta(w, q_0) \in Q_{voc}$, if $\theta(w) = \theta(s')$ and $\tau' \neq \tau$ for every τ' -string (4.2.1) of $L(G_{lo})$ with reference prefix $w \in L(G_{lo})$ and $\tau' \in T_{act}$, then for every t_h -string $\langle w, \alpha_j, z_j, h, t_h \rangle \in L(G_{lo})$, there exists some j ($1 \leq j \leq h$) such that*

$$\alpha_j \in \Sigma_{hib} \text{ or } (\alpha_j = t_l \ \& \ \Sigma(z_{j-1}) \cap \Sigma_{for} \cap \Sigma_u \neq \emptyset).$$

In words, consider an arbitrary τ -string of $L(G_{lo})$ with reference prefix s' and $\tau \in T_{act}$. Then t_h is unambiguously preemptable w.r.t the mirage of (s', τ) if, for every other string $w \in L_{voc}(G_{lo})$ that has the same θ -image as string s' , but is not the reference prefix of any τ -string of $L(G_{lo})$, every t_h -string of $L(G_{lo})$ with string w as its reference prefix has its co-silent string either containing a prohibitable event, or a t_l which can be preempted by a forcible event that is uncontrollable.

Definition 4.4 (Output-force consistency). *A TDES G_{lo} is said to be OFC if, for every τ -string (4.2.1) of $L(G_{lo})$ with reference prefix $s' \in L(G_{lo})$ and $\tau \in T_{act}$, for which there exists a t_h -string with reference prefix $w' \in L(G_{lo})$ such that $\theta(w') = \theta(s')$, $\tau \in T_{for}$ iff t_h is unambiguously preemptable w.r.t (s', τ) and its mirage.*

For an OFC G_{lo} , $\tau \in T_{for}$ if $\tau \in T_{act}$ can unambiguously preempt the tick t_h , whenever the former is virtually enforced at a high-level state where τ and t_h are eligible, reached

following an underlying vocal string that has the same θ -image as the reference prefix of an arbitrary τ -string of $L(G_{lo})$. Otherwise, $\tau \in T_{act} - T_{for}$.

4.3.3 Output Time-Compliance

Output time-compliance can be achieved by a proper design or redesign of the reporter map θ , and is formally defined as follows.

Definition 4.5 (Output time-compliance). *A TDES G_{lo} is said to be OTC if, for every τ -string of $L(G_{lo})$ with reference prefix s' and $\tau \in T_{act}$, if there exists a t_h -string of $L(G_{lo})$ with reference prefix r_1 such that $\theta(s') = \theta(r_1)$, then there exists a τ -string of $L(G_{lo})$ with reference prefix r_2 such that $\theta(r_2) = \theta(r_1)t_h$.*

Put another way, if G_{lo} is OTC, the essence is that, as with the ticking of t_l in G_{lo} , the ticking of t_h is never the cause of activity event ineligibility in G_{hi} . This is formalized in Proposition 4.4.

Proposition 4.4. *Given a TDES hierarchy (G_{lo}, G_{hi}) , G_{hi} obeys Property 2.2 iff G_{lo} is OTC.*

Proof. Given a TDES hierarchy (G_{lo}, G_{hi}) , where $G_{hi} \stackrel{\text{def}}{=} (X, T, \xi, x_0, -)$. That G_{hi} obeys Property 2.2, i.e., for an arbitrary string $t \in L(G_{hi})$, and hence an arbitrary state $x = \xi(t, x_0) \in X$, if $x' = \xi(t_h, x) \in X$, then $T_{act}(x) \subseteq T_{act}(x')$, (equivalently) means that if $\tau \in T_{act}(\xi(t, x_0))$ and $t_h \in T(\xi(t, x_0))$, then $\tau \in T_{act}(\xi(tt_h, x_0))$, i.e., if $t\tau, tt_h \in L(G_{hi})$, where $\tau \in T_{act}$, then $tt_h\tau \in L(G_{hi})$. Since $\theta(L(G_{lo})) = L(G_{hi})$, there always exists an $s \in L(G_{lo})$ such that $\theta(s) = t$. Together, it means that for an arbitrary τ -string of $L(G_{lo})$ with reference prefix s' , where $\theta(s') = t$, if there exists a t_h -string of $L(G_{lo})$ with reference prefix r_1 such that $\theta(r_1) = t$ and therefore $\theta(s') = \theta(r_1)$, then there exists a τ -string of $L(G_{lo})$ with reference prefix r_2 such that $\theta(r_2) = tt_h$ and therefore $\theta(r_2) = \theta(r_1)t_h$. By Definition 4.5, this means that G_{lo} is OTC. Hence the proposition. $\square$

Proposition 4.5. *Given a TDES hierarchy (G_{lo}, G_{hi}) and that G_{lo} is AOCC and OTC, G_{hi} is a TDES model with time fidelity.*

Proof. Given a TDES hierarchy (G_{lo}, G_{hi}) , by Proposition 4.1, G_{hi} is activity-loop free. Since G_{lo} is AOCC, by Proposition 4.3, G_{hi} satisfies Property 2.3. And since G_{lo} is OTC, by Proposition 4.4, G_{hi} satisfies Property 2.2. Therefore, since G_{hi} is activity-loop free and satisfies Properties 2.2 and 2.3 of a TDES model, it is a TDES model with time fidelity. Hence the proposition. $\square$

4.3.4 Timed Output-Control Consistency

Based on the concepts of activity output-control consistency, output-force consistency, and output time-compliance, the two concepts of output-control consistency may now be defined.

Definition 4.6 (Output-control consistency). *A TDES G_{lo} is said to be OCC if it is AOCC and OFC; and TOCC if it is OCC and OTC.*

The foregoing theoretical development culminates in the following theorem.

Theorem 4.1. *Given a TDES hierarchy (G_{lo}, G_{hi}) and that G_{lo} is TOCC, G_{hi} is a TDES model with time fidelity.*

Proof. Given a TDES hierarchy (G_{lo}, G_{hi}) where G_{lo} is TOCC. By Definition 4.6, a TOCC G_{lo} is necessarily AOCC and OTC. Hence the result by Proposition 4.5. $\square$

Importantly, along with the unambiguous control properties of high-level activity events, the abstraction G_{hi} of a TOCC G_{lo} provides a real-time basis of generally coarser time granularity that is decoupled from the low level, for which high-level specification TTG's can be independently specified for feasible hierarchical control design.

4.4 Hierarchical Consistency: Theoretical Conditions

A core high-level supervisor expectation issue for timed hierarchical control design inherited from the untimed version [1] is explained, and two versions of hierarchical consistency

for TDES's, without and with output-time fidelity guarantee, are then defined to address the issue. Before that, a conceptual computation tool is reviewed, and a timed concept of partner-freeness is subsequently introduced as the absence, of vocal-state partners illustrated using this tool, to complete the timed systems synthesis framework for hierarchical consistency.

To explain and illustrate concepts, we may use the concept of a reachability tree reviewed in Chapter 2. In what follows, the terminology, drawing notation and concepts formulated for state and state-transitions of G_{lo} carry over to nodes and node-transitions of the corresponding tree $G_{lo,t}$. It should be understood that, in referring to a τ -string $\langle s', \sigma_i, x_i, k, \tau \rangle \in L(G_{lo,t})$, x_0 and x_i ($1 \leq i \leq k$) are nodes defined by the transition function δ_t of $G_{lo,t}$ over the reference prefix s' and string $s'\sigma_1\sigma_2 \cdots \sigma_i$, respectively, and w.r.t the root node n_0 .

4.4.1 Consistency for Hierarchical Control

Consider a two-level TDES hierarchy (G_{lo}, G_{hi}) where G_{lo} is OCC. Given a high-level specification $E \subseteq T^*$, the optimal high-level timed supervisor synthesized for its prefix closure $\overline{E}$ w.r.t G_{hi} is $S_{hi} = \text{Supcon}(\overline{G}_{hi}, \overline{E})$. Let

$$K = \theta^{-1}(L(S_{hi})) \subseteq L(G_{lo}),$$

the (low-level) maximal sublanguage of $L(G_{lo})$ whose projection under the timed reporter map θ is $L(S_{hi})$. In general, K is prefix-closed but not controllable w.r.t G_{lo} . Consider the low-level timed supervisor S_{lo} synthesized for K w.r.t G_{lo} , given by

$$S_{lo} = \text{Supcon}(\overline{G}_{lo}, K).$$

Then since $L(S_{lo}) \subseteq \theta^{-1}L(S_{hi})$, it follows that in general,

$$\theta(L(S_{lo})) \subseteq L(S_{hi}). \quad (4.4.1)$$

Inclusion (4.4.1) asserts that the projection of the prefix-closed language generated by G_{lo} under the supervision of S_{lo} is a sublanguage of the prefix-closed language generated by G_{hi} under the high-level (virtual) supervision of S_{hi} . Indeed, (4.4.1) may turn out to be strict, in which case the low-level system G_{lo} under the supervision of S_{lo} cannot meet the expectation of high-level supervisor S_{hi} .

The basis for hierarchical control design for a two-level TDES hierarchy (G_{lo}, G_{hi}) requires the equality in (4.4.1), i.e., $\theta(L(S_{lo})) = L(S_{hi})$. In what follows, two concepts of hierarchical consistency are defined.

Definition 4.7 (Hierarchical consistency). *A TDES hierarchy (G_{lo}, G_{hi}) is said to be*

1. *hierarchically consistent (HC)^{4.4} if, $(\forall E)(E \subseteq T^*)$*

$$(S_{hi} = \text{Supcon}(\overline{G}_{hi}, \overline{E}))(S_{lo} = \text{Supcon}(\overline{G}_{lo}, \theta^{-1}(L(S_{hi}))))\theta(L(S_{lo})) = L(S_{hi});$$

2. *and HC with output-time fidelity (HC-OTF) if, additionally, G_{hi} possesses high-level time fidelity.*

4.4.2 Vocal-State Partnership and Strictness of Output-Control

Consistency

That Inclusion (4.4.1) may be strict can be explained using the concept of vocal-state partnership for OCC DES's [12] extended to OCC TDES's. Intuitively, if two vocal states of an OCC TDES are vocal-state partners, they output two different, eligible controllable outputs that cannot in general be independently controlled, in that, in taking a low-level control action necessary to disable or preempt one output, it is possible that this action also prevents the other output from occurring next. This partnership concept is formalized as follows.

^{4.4}Note that unlike in Chapter 3 where HC abbreviates the concept, here it abbreviates the adjectival form for the same concept.

Definition 4.8 (Control-dependent vocal states). *For a TDES G_{lo} , let $q_1, q_2 \in Q_{voc}$, where either $V(q_1)$ or $V(q_2)$ is an event of T_{act} . Then (q_1, q_2) is said to be a pair of control-dependent vocal states over $[s', w\sigma s'', <_{s_2}^{s_1}, j]$, where $s', w, s'', s_1, s_2 \in \Sigma^*$ and $\sigma \in \Sigma$, if, for all $i \in \{1, 2\}$, there exists a $V(q_i)$ -string of $L(G_{lo})$, with common reference prefix s' and co-silent string of the form $w_i = w\sigma s'' s_i$, with $q = \delta(s', q_0)$ and $q_i = \delta(w_i, q)$, such that the following three conditions hold:*

CDS1) σ is not uncontrollable, i.e., $\sigma \in \Sigma_{hib}$ or $\sigma = t_l$ & $\Sigma(\delta(s'w, q_0)) \cap \Sigma_{for} \neq \emptyset$.

CDS2) $s'' \in (\Sigma_u \cup \{t_l\})^$ and is uncontrollable, i.e., $\forall s_a \in \Sigma^*$, if $s_a t_l \leq s''$ then*

$$\Sigma(\delta(s'w\sigma s_a, q_0)) \cap \Sigma_{for} = \emptyset.$$

CDS3) $\exists j \in \{1, 2\}$ such that $s_j \in (\Sigma_u \cup \{t_l\})^+$ and is not controllable, i.e., $\forall s_b \in \Sigma^$, if*

$$s_b t_l \leq s_j \text{ then } \Sigma(\delta(s'w\sigma s'' s_b, q_0)) \cap \Sigma_{for} \cap \Sigma_u = \emptyset.$$

Note that a pair (q_1, q_2) of states in G_{lo} may be control-dependent over several strings structures of the form $[s', w\sigma s'', <_{s_2}^{s_1}, j]$.

Definition 4.9 (Vocal-state partnership). *For a TDES G_{lo} , let $q_1, q_2 \in Q_{voc}$, where either $V(q_1)$ or $V(q_2)$ is an event of T_{act} . Then (q_1, q_2) is said to be a pair of vocal-state partners if $V(q_1) \neq V(q_2)$ and (q_1, q_2) is a pair of control-dependent states.*

Figure 4.2: Vocal-state partners q_1 and q_2 of a TDES G_{lo} : They are depicted respectively by nodes n_1 and n_2 in a subtree of the system reachability tree, with $V(q_1) = V_t(n_1) = \tau_1$ and $V(q_2) = V_t(n_2) = \tau_2$, where $\tau_1 \neq \tau_2$ and $\sigma \in \Sigma_{hib}$ or is a preemptable tick.

Together with Definition 4.8, Definition 4.9 for a pair of arbitrary vocal-state partners q_1 and q_2 of a TDES (G_{lo}, V) are graphically depicted in Figure 4.2, respectively by nodes n_1 and n_2 in a subtree of the reachability tree generated for G_{lo} . As defined, this tree is also a Moore automaton, denoted by $(G_{lo,t}, V_t)$, with $G_{lo,t} \stackrel{\text{def}}{=} (Q_t, \Sigma, \delta_t, n_0, Q_{m,t})$ and vocal node set $Q_{voc,t} \subseteq Q_t$, except that its transition function is defined over an infinite set of elements called nodes instead of a finite set of states, with a different node representing a possibly duplicate state of G_{lo} reached by a different string of $L(G_{lo})$. In the figure, in shorthand drawing notation, a string traversing between two pertinent nodes in the subtree is graphically represented by a directed edge, as for an event, without showing the intermediate nodes and transitions, and has no double bars (//) across it only if the intermediate nodes that the string traverses through are all non-vocal. Each dotted line indicates a catenation of strings between two vocal nodes. As depicted, Condition CDS1 asserts that, along the string $w\sigma s''$ defining the transitions via non-vocal nodes from vocal node n of $G_{lo,t}$ corresponding to vocal state $q = \delta(s', q_0)$ of G_{lo} , the event σ is prohibitable, or is a tick t_l that is not non-preemptable. Condition CDS2 asserts that every event along the string s'' is uncontrollable, or is a non-preemptable tick. Condition CDS3 asserts that the string s_j for some $j \in \{1, 2\}$, defining the transitions from non-vocal node n_b corresponding to non-vocal state $\delta(w\sigma s'', q)$ of G_{lo} , via non-vocal nodes to the vocal node n_j that outputs $\tau_j \in T$, i.e., $V_t(n_j) = \tau_j = V(q_j)$, contains no prohibitable events and no unambiguously preemptable ticks. Therefore, along this string s_j , that τ_j cannot be prevented from occurring is definite only if no tick along the transitions defined by the string s_j is ambiguously preemptable.

For discussion's sake, let $j = 1$. Then under such vocal-state partnership of (q_1, q_2) as depicted by (n_1, n_2) in Figure 4.2, if the TDES G_{lo} is OCC, then $\tau_i \in T_c$ for all $i \in \{1, 2\}$, and the low-level control action guaranteed to prevent τ_1 from occurring - by disabling or preempting an event along the transitions in G_{lo} defined by the string $w\sigma$ - will also prevent τ_2 from occurring. In other words, for an OCC G_{lo} , the control of τ_1 and τ_2 is generally not independent if the associated pair (q_1, q_2) are vocal-state partners.

It follows that, to guarantee independence of high-level control, it is sufficient for an

OCC TDES G_{lo} to be free of vocal-state partners.

Definition 4.10 (Partner-freeness). *A TDES G_{lo} is said to be partner-free (PF) if it does not contain vocal-state partners.*

Two strict versions of output-control consistency follow.

Definition 4.11 (Strictness of output-control consistency). *A TDES G_{lo} is said to be strictly OCC (SOCC) if it is OCC and PF. It is said to be strictly TOCC (STOCC) if it is SOCC and OTC, or equivalently, TOCC and PF.*

Before proceeding, we summarize the bottom-up development of system concepts and their constituent relationships for timed hierarchical control in a self-explanatory figure, Figure 4.3.

Figure 4.3: Hierarchical system concepts and their constituent relationships for a Moore TDES model G_{lo}

4.4.3 Hierarchical Consistency Theorem

We are now ready to state the structural conditions for the consistency of a two-level TDES hierarchy.

Theorem 4.2. *A TDES hierarchy (G_{lo}, G_{hi}) is*

1. *HC if G_{lo} is SOCC, and*
2. *HC-OTF if G_{lo} is STOCC.*

Proof. Given a TDES hierarchy (G_{lo}, G_{hi}) . The proof for the two cases proceeds as follows.

Case 1: G_{lo} is SOCC. By Definition 4.11, G_{lo} is OCC and PF. For an arbitrary $E \subseteq T^*$, let $S_{hi} = \text{Supcon}(\overline{G_{hi}}, \overline{E})$ and $S_{lo} = \text{Supcon}(\overline{G_{lo}}, \theta^{-1}(L(S_{hi})))$. By Definition 4.7-1, to show that (G_{lo}, G_{hi}) is HC, we need to show that $\theta(L(S_{lo})) = L(S_{hi})$, as follows:

Suppose $L(S_{hi}) = \emptyset$. Then $L(S_{lo}) = \emptyset$ and trivially,

$$\theta(L(S_{lo})) = \theta(\emptyset) = \emptyset = L(S_{hi}).$$

In the rest of the proof for this case, suppose $L(S_{hi}) \neq \emptyset$. Then $L(S_{lo}) \neq \emptyset$. Otherwise, $\text{Supcon}(\overline{G_{lo}}, \theta^{-1}(L(S_{hi}))) = \text{EMPTY}$. By Lemma 4.1 that $L(G_{lo}) = \overline{L_{\text{voc}}(G_{lo})}$ and the definition of θ^{-1} , for every $s'\sigma \in L(G_{lo})$ where $\sigma \in \Sigma$,

$$s' \in \theta^{-1}(L(S_{hi})) \ \& \ s'\sigma \notin \theta^{-1}(L(S_{hi})) \implies s'\sigma \in L_{\text{voc}}(G_{lo}).$$

Therefore, that $\text{Supcon}(\overline{G_{lo}}, \theta^{-1}(L(S_{hi}))) = \text{EMPTY}$ implies

$$L(S_{hi}) \neq \emptyset \implies L(S_{hi}) \supset L(\text{Supcon}(\overline{G_{hi}}, L(S_{hi}))).$$

In turn, this means $L(S_{hi})$ is empty or not controllable, contradicting the supposition that $L(S_{hi})$ is not empty and controllable.

By Inclusion (4.4.1), $\theta(L(S_{lo})) \subseteq L(S_{hi})$. It remains to show that $\theta(L(S_{lo})) \supseteq L(S_{hi})$. To do that, we now suppose $\theta(L(S_{lo})) \subset L(S_{hi})$ and show a contradiction of the fact that the given G_{lo} is PF, as follows:

Since $L(S_{hi}) - \theta(L(S_{lo})) \neq \emptyset$, let t be a string of $L(G_{hi})$ such that

$$t \in L(S_{hi}) - \theta(L(S_{lo})).$$

Since $L(S_{lo}) \neq \emptyset$, $\varepsilon \in L(S_{lo})$ and hence $\varepsilon \in \theta(L(S_{lo}))$. Since $L(G_{hi})$ is prefix-closed and $\varepsilon \in \theta(L(S_{lo}))$, the longest prefix t' of t exists such that

$$t' < t \text{ and } t' \in \theta(L(S_{lo})).$$

Let $s \in \theta^{-1}(t)$ and $\delta(s, q_0) \in Q_{voc} \cup \{q_0\}$. It follows that $s \notin L(S_{lo})$. Since $L(S_{lo}) \neq \emptyset$ and is prefix-closed, the longest prefix s' of s exists such that $s' \in L(S_{lo})$ and $\delta(s', q_0) \in Q_{voc} \cup \{q_0\}$. Let $t'' = \theta(s')$. Then $t'' \leq t'$ and therefore $t'' \in L(S_{hi})$. Let $w \in \Sigma^+$ such that $s'w \leq s$, $\delta(s'w, q_0) \in Q_{voc} \cup \{q_0\}$ and $\theta(s'w) = \theta(s')\tau_1$ for some $\tau_1 \in T$. Since $L(S_{lo})$ is controllable, $s' \in L(S_{lo})$ and $s'w \notin L(S_{lo})$, it follows that $(\exists w')(\exists \sigma \in \Sigma)w'\sigma \leq w$, $s'w' \in L(S_{lo})$ and

$$\sigma \in \Sigma_{hib} \text{ or } (\sigma = t_l) \& (\exists \gamma \in \Sigma_{for}, \delta(s'w'\gamma, q_0)!) s'w'\gamma \in L(S_{lo})).$$

Otherwise, $w \in (\Sigma_u \cup t_l)^*$ and contains no unambiguously preemptable t_l , and is such that, due to the controllability of $L(S_{lo})$, every event of $\Sigma_{hib} \cap \Sigma_{for}$ that can preempt a t_l in w exits the boundary of $L(S_{lo})$, i.e., $(\forall w', w't_l \leq w) s'w' \in L(S_{lo})$ and

$$(\forall \gamma \in \Sigma_{hib} \cap \Sigma_{for}, \delta(s'w'\gamma, q_0)!) s'w'\gamma \notin L(S_{lo}),$$

and we have $s'w \in L(S_{lo})$, hence contradicting the maximality of s' w.r.t inclusion in $L(S_{lo})$. Thus $w \in (\Sigma^*)(\Sigma_{hib} \cup \{t_l\})(\Sigma_u \cup \{t_l\})^*$, such that the event of w in $(\Sigma_{hib} \cup \{t_l\})$ is either prohibitable or a t_l that can be preempted by some forcible event that does not exit the

boundary of $L(S_{lo})$, and every event of $\Sigma_{hib} \cap \Sigma_{for}$ that can preempt a t_l in its suffix string in $(\Sigma_u \cup \{t_l\})^*$ that contains no unambiguously preemptable t_l exits the boundary of $L(S_{lo})$. Now, let $w' \in \Sigma^*$ and $\sigma \in (\Sigma_{hib} \cup \{t_l\})$ be such that $s'w'\sigma \leq s'w$, $s'w' \in L(S_{lo})$ and $s'w'\sigma \notin L(S_{lo})$, i.e., σ must be disabled to stay in $L(S_{lo})$.

In what follows, there must exist a string $v \in (\Sigma_u \cup \{t_l\})^*$ that contains no unambiguously preemptable t_l , and where every event of $\Sigma_{hib} \cap \Sigma_{for}$ that can preempt a t_l in v exits the boundary of $L(S_{lo})$, such that $\delta(s'w'\sigma v, q_0) \in Q_{voc} \cup \{q_0\}$, $\theta(s'w'\sigma v) = \theta(s')\tau_2$ for some $\tau_2 \in T$ and $\theta(s'w'\sigma v) \notin L(S_{hi})$, because having no such string v contradicts the fact that $L(S_{lo})$ is the supremal controllable sublanguage of $\theta^{-1}(L(S_{hi}))$ w.r.t $\overline{G}_{lo}$.

Let $q_1 = \delta(s'w, q_0)$ and $q_2 = \delta(s'w'\sigma v, q_0)$ and. Thus $V(q_1) = \tau_1$ and $V(q_2) = \tau_2$. Since v extends from $s'w'\sigma$ and is not controllable in general, the longest common prefix string $s'w'' \in \Sigma^*$ of $s'w$ and $s'w'\sigma v$ exists such that $s'w''w_1 = s'w$ and $s'w''v_1 = s'w'\sigma v$ where $w_1, v_1 \in \Sigma^*$. It follows that $w'' = w'\sigma s_1$ for some $s_1 \in \Sigma^*$. Since v is not controllable, s_1 is not controllable. It follows that there must exist a string $s'' \in \Sigma^*$ that is the longest suffix of s_1 that is uncontrollable, i.e., $w'\sigma s_1 = w'\sigma s_2 s''$, where s_2 is not controllable (i.e., uncontrollable or ambiguously controllable). Therefore (q_1, q_2) is a pair of control-dependent vocal states over $[s', w'\sigma s_2 s'', <_{v_1}^{w_1}, -]$ since v_1 is not controllable because v is not controllable, and therefore satisfies Condition CDS3 of Definition 4.8; and

- if s_2 is uncontrollable, then σ satisfies Condition CDS1 and string $s_2 s''$ satisfies Condition CDS2, of Definition 4.8; and
- if s_2 is ambiguously controllable and thus is of the form $s'_1 \sigma' s''_2$ for some $s'_1, s''_2 \in (\Sigma_u \cup \{t_l\})^*$, where s''_2 is uncontrollable and $\sigma' = t_l$ is ambiguously preemptable, then event σ' satisfies Condition CDS1, and string $s''_2 s''$ satisfies Condition CDS2, of Definition 4.8.

Since $\theta(s'w) = t''\tau_1 \in L(S_{hi})$ and $\theta(s'w'\sigma v) = t''\tau_2 \notin L(S_{hi})$, it follows that $\tau_1 \neq \tau_2$ and therefore $V(q_1) \neq V(q_2)$. Together with the fact that (q_1, q_2) is a pair of control-dependent

vocal states, (q_1, q_2) is a pair of vocal-state partners by Definition 4.9, contradicting the fact that the given G_{lo} is PF.

Case 2: G_{lo} is STOCC. Then, by Definition 4.11, it is necessary that G_{lo} is SOCC, and therefore by the proof of Case 1 above, (G_{lo}, G_{hi}) is HC. It also necessary that G_{lo} is TOCC, and therefore by Theorem 4.1, G_{hi} possesses high-level time fidelity. Combining, it follows that (G_{lo}, G_{hi}) is HC-OTF by Definition 4.7-2.

Hence the theorem. □

4.4.4 An Illustrative Example: Hierarchical Mission Control of a Robotic Camera System

As an illustration of the STOCC system concept for building a hierarchy that is HC-OTF, we now present an example of a simplified robotic camera system. One may think of it as

Figure 4.4: Camera system: Moore ATG model and associated event timings

a module on board a drone or an unmanned aerial vehicle, for use in a surveillance mission along a designated flying route. This module is to be organized as a hierarchy (G_{lo}, G_{hi}) , constructed from a given TDES (G, θ) that is represented by a Moore ATG G_{act} and the associated event timing information as shown in Figure 4.4, with $\theta : L(G) \rightarrow T^*$ and $T_{act} = \{\tau_1, \tau_2\}$, under a periodic timescale 1 : 2. The event set $\Sigma_{act} = \{\sigma_1, \sigma_2, \sigma_3\}$ of G is partitioned with $\Sigma_{hib} = \{\sigma_1, \sigma_3\}$ and $\Sigma_{for} = \{\sigma_2\}$. The definitions of the activities and events are given in Table 4.1.

Following the construction as shown in Figure 4.5, it can be verified that G_{lo} is:

Models	Event	Activity
G_{lo}	σ_1 : Detected new object localized σ_2 : Camera set σ_3 : Camera clicked	a_0 : Scanning and detecting new suspicious moving or stationary object of interest a_1 : Setting camera shutter speed and aperture based on scene lighting a_2 : Improving camera setting and zooming in on localized object
G_{hi}	τ_1 : New object seen τ_2 : Photo taken	

Table 4.1: Symbol definitions for a robotic camera system hierarchy (G_{lo}, G_{hi})

- OCC, by setting $T_{hib} = \{\tau_2\}$ and $T_u = \{\tau_1\}$; $T_{for} = \{\tau_2\}$ and $\tau_1 \notin T_{for}$.
- PF, due to the absence of a partnership structure as depicted in Figure 4.2, by determining that for all state pairs (q_1, q_2) with $V(q_1), V(q_2) \in T$, every pair of $V(q_1)$ -string and $V(q_2)$ -string with common reference prefix and co-silent strings with common prefix s , has either $V(q_1) = V(q_2) = t_h$ or $s = \varepsilon$.
- OTC, by observing that G_{hi} obeys Property 2.2 and following Proposition 4.4.

Therefore, in the constructed hierarchy (G_{lo}, G_{hi}) , G_{lo} is STOCC by Definition 4.11, and hence the hierarchy is HC-OTF by Theorem 4.2.

A system abstraction is very useful if it can abstract away unnecessary low-level language details and provide unambiguous control information of interest abstracted as languages of high-level events. As this example hierarchy shows, the abstraction G_{hi} provides a clear understanding of the application mission-level driven by the underlying real TDES G_{lo} , and allows real-time requirements to be more readily identified and specified at the high level.

As a specification example over G_{hi} , consider a controllable specification asserting that, continually, the robotic camera, upon seeing a new object, is to take a photo with no further

Figure 4.5: Camera system: A constructed hierarchy (G_{lo}, G_{hi})

t_h -time delay as soon as it is ready to do so (following a t_h -setup^{4.5}). This specification may be prescribed by a TTG (say, MC) that structurally is G_{hi} (see Figure 4.5(a)), but with the self-loop t_h -transition at state 3 removed. A low-level supervisor S_{lo} for the TDES G_{lo} can be synthesized for which $\theta(L(S_{lo})) = L(MC)$.

In driving home the point, a hierarchy that is HC-OTF is necessary if we need to synthesize low-level supervisor solutions that fully realize the prefix closure of controllable,

^{4.5}Examining G_{lo} for the curious reader, this high-level tick models the minimum time required by the underlying system for auto-adjusting the setting of the camera shutter speed and aperture to anticipate a clear photo finish.

high-level real-time specifications w.r.t the high-level TDES model, without violating their intended high-level timing semantics.

4.5 Conclusion

In this chapter, we have presented a new framework for hierarchical consistency of TDES's. The sufficiency conditions for ensuring hierarchical consistency without and with output-time fidelity guarantee are formulated. An example explains the conditions for hierarchical consistency with output-time fidelity, laying a sound real-time high-level specification design foundation for hierarchical control that, in this aspect is fundamentally different from related work [78] which does not assure or ignores the issues of system time fidelity. In the next chapter, we address the issues of Moore system synthesis under the real-time hierarchical control framework presented herein.

Chapter 5

System Abstraction for Real-Time Hierarchical Control: Consistency Synthesis

In this chapter, we complete the theoretical development of the real-time hierarchical control framework proposed in the previous chapter by addressing the issues of Moore system synthesis or refinement.

In general, a Moore TDES (G_{lo}, V) constructed from given hierarchical system information (G, θ) is not STOCC. Refining it to be so by modifying the associated map V for G_{lo} turns out to be a challenging research problem. By system refinement or synthesis, we refer to redefining the map V over G_{lo} , *without removing any given high-level activity and timing information*. It can be easily deduced that a TDES G_{lo} refined as such remains time-output responsive (in the sense of not invalidating the time-output design Laws 1 and 2). Therefore, the system concepts and their constituent relationships defined for a given TDES G_{lo} , as summarized in Figure 4.3, are also applicable to a refined TDES, and so are the definitions and results presented from Sections 4.2 to 4.4. Where required, a refined TDES will be referred to by the same symbol, G_{lo} , to imply that it remains time-output

responsive as the given TDES G_{lo} , in all the theoretical proofs of synthesis results.

For clarity of description, every system refinement (procedure or method) will be defined, and thought of as being ‘implemented’ in terms of refinement of the Moore reachability tree introduced in Section 2.5, i.e., redefining the map V over G_{lo} is made by redefining the corresponding V_t over its tree $G_{lo,t}$. The system refinement is therefore conceptual.^{5.1}

This chapter is organized as follows. Using string-wise definitions of output-control properties defined in Section 5.1, we investigate the issues and present the results of structural existence and synthesis (or refinability) for SOCC systems from Sections 5.2 to 5.4, and point out and discuss the abstraction anomalies to be removed for obtaining STOCC systems in Section 5.5. Along with it, we also investigate the conditions under which timescale is preserved following a system refinement. In essence, these are the system structural conditions under which a refinement does not introduce a new t_h -string for the refined TDES. As established in Theorem 4.2, the structure of an SOCC system is sufficient for hierarchical consistency with no output-time fidelity guarantee, while that of an STOCC system is sufficient for hierarchical consistency with output-time fidelity. Based on the existence and synthesis results for SOCC-systems, in Section 5.6, it is shown that a class of Moore TDES’s exists whose structure can be refined to achieve hierarchical consistency with output-time fidelity, and is a linear subclass of a mildly restrictive class of TDES’s whose structure can be refined to achieve hierarchical consistency with no output-time fidelity guarantee. In Section 5.7, a detailed example illustrates how a Moore TDES of the linear subclass studied can be sufficiently refined to achieve hierarchical consistency with output-time fidelity. The chapter continues with a discussion on generalizing and scaling to multiple levels the consistency of a two-level hierarchy in Section 5.8. Section 5.9 concludes the chapter.

^{5.1}In principle, in place of reachability trees, more efficient and compact representations are available for practical implementation that can be stored [12, 154].

5.1 String-wise Control Partitions of Outputs

At this juncture, it is useful to bring in some string-wise definitions for the event-control properties of $\tau \in T$, as follows. Given an arbitrary τ -string $s = \langle s', \sigma_i, x_i, k, \tau \rangle \in L(G_{lo})$:

- $\tau \in T$ is said to be controllable w.r.t s if the co-silent string of s is controllable, i.e., for some i ($1 \leq i \leq k$), $\sigma_i \in \Sigma_{hib}$ or $(\sigma_i = t_l \ \& \ \Sigma(x_{i-1}) \cap \Sigma_{for} \cap \Sigma_u \neq \emptyset)$.
- $\tau \in T$ is said to be uncontrollable w.r.t s if the co-silent string of s is uncontrollable, i.e., for all i ($1 \leq i \leq k$), $\sigma_i \in \Sigma_u$ or $(\sigma_i = t_l \ \& \ \Sigma(x_{i-1}) \cap \Sigma_{for} = \emptyset)$.
- $\tau \in T$ is said to be ambiguously controllable w.r.t s if the co-silent string of s is ambiguously controllable, i.e., for all i ($1 \leq i \leq k$), $\sigma_i \in \Sigma_u$ or $(\sigma_i = t_l \ \& \ \Sigma(x_{i-1}) \cap \Sigma_{for} \cap \Sigma_u = \emptyset)$, and for some i ($1 \leq i \leq k$), $\sigma_i = t_l \ \& \ \Sigma(x_{i-1}) \cap \Sigma_{for} \cap \Sigma_{hib} \neq \emptyset$.
- $\tau \in T_{act}$ is said to be forcible w.r.t s' if t_h is unambiguously preemptable w.r.t (s', τ) and its mirage, and is not forcible w.r.t s' otherwise, in either case that there exists a t_h -string with reference prefix $s'' \in L(G_{lo})$ such that $\theta(s'') = \theta(s')$, or else it is said to be force-don't-care w.r.t s' .

In the above, all the definitions except the last are for events in $T = T_{act} \dot{\cup} \{t_h\}$. A $\tau \in T_{act}$ that is force-don't-care w.r.t the reference prefix s' of a given τ -string is said to be definable as either forcible or not forcible w.r.t s' . It follows that the definitions induce two string-wise control partitions of Moore outputs. In one partition, an arbitrary $\tau \in T$ is either controllable, uncontrollable or ambiguously controllable w.r.t to every of its τ -strings, and in the other, an arbitrary $\tau \in T_{act}$ is either forcible or non-forcible w.r.t to the reference prefix of every of its τ -strings.

5.2 OCC-System Refinability and Refinement

In what follows, a TDES is said to be AOCC-, OFC-, and OCC-system refinable, if it can be refined to be AOCC, OFC and OCC, respectively.

Theorem 5.1. *A TDES G_{lo} is not AOCC-system refinable iff there exists a τ -string*

$\langle s', \sigma_i, x_i, k, \tau \rangle \in L(G_{lo})$ with $\tau \in T_{act}$, such that

- *$\sigma_1 \cdots \sigma_{k-1}$ is not controllable, i.e., for all i ($1 \leq i \leq k-1$), $\sigma_i \in \Sigma_u$ or ($\sigma_i = t_l$ & $\Sigma(x_{i-1}) \cap \Sigma_{for} \cap \Sigma_u = \emptyset$), and*
- *σ_k is ambiguously controllable, i.e., $\sigma_k = t_l$ and is ambiguously preemptable.*

Proof. (If): Suppose there exists a τ -string $s = \langle s', \sigma_i, x_i, k, \tau \rangle \in L(G_{lo})$ with $\tau \in T_{act}$, such that

- for all i ($1 \leq i \leq k-1$),

$$\sigma_i \in \Sigma_u \text{ or } (\sigma_i = t_l \text{ \& } \Sigma(x_{i-1}) \cap \Sigma_{for} \cap \Sigma_u = \emptyset), \text{ and}$$

- $\sigma_k = t_l$ and is ambiguously preemptable.

Then, the τ -string $s \in L(G_{lo})$ does not satisfy any of the respective condition for $\tau \in T_{hib}$ and $\tau \in T_u$ stated in Definition 4.1, and it follows that G_{lo} is not AOCC. Regardless of any Moore transition redefinition along the co-silent string $\sigma_1 \sigma_2 \cdots \sigma_{k-1}$ of s , that $\sigma_k = t_l$ is ambiguously preemptable remains, and hence there will always exist a τ -string of $L(G_{lo})$, with reference prefix $s' \sigma_1 \cdots \sigma_i$ for some i ($1 \leq i \leq k-1$), for the refined G_{lo} that does not satisfy any of the respective condition for $\tau \in T_{hib}$ and $\tau \in T_u$ stated in Definition 4.1, implying that the refined G_{lo} is not AOCC. Hence, the given G_{lo} is not AOCC-system refinable.

(Only if): Suppose G_{lo} is not AOCC-system refinable, and is therefore not AOCC. Together, they mean that there exists a τ -string $s = \langle s', \sigma_i, x_i, k, \tau \rangle \in L(G_{lo})$ with $\tau \in T_{act}$, that

- does not satisfy any of the respective condition for $\tau \in T_{hib}$ and $\tau \in T_u$ stated in Definition 4.1 and thus

$$\text{-- for all } i \text{ } (1 \leq i \leq k), \sigma_i \in \Sigma_u \text{ or } (\sigma_i = t_l \text{ \& } \Sigma(x_{i-1}) \cap \Sigma_{for} \cap \Sigma_u = \emptyset), \text{ and}$$

- for some i ($1 \leq i \leq k$), $(\sigma_i = t_l \ \& \ \Sigma(x_{i-1}) \cap \Sigma_{for} \cap \Sigma_{hib} \neq \emptyset)$,
- and no Moore transition redefinition along its co-silent string $\sigma_1\sigma_2 \cdots \sigma_{k-1}$ can be made to satisfy the condition for $\tau \in T_u$ as stated in Definition 4.1, implying the additional condition that

$$\sigma_k = t_l \ \& \ \Sigma(x_{k-1}) \cap \Sigma_{for} \cap \Sigma_{hib} \neq \emptyset.$$

In other words, if G_{lo} is not AOCC-system refinable, by logical conjunction, there exists a τ -string $s = \langle s', \sigma_i, x_i, k, \tau \rangle \in L(G_{lo})$ with $\tau \in T_{act}$ such that

- for all i ($1 \leq i \leq k-1$), $\sigma_i \in \Sigma_u$ or $(\sigma_i = t_l \ \& \ \Sigma(x_{i-1}) \cap \Sigma_{for} \cap \Sigma_u = \emptyset)$, and
- $\sigma_k = t_l$ and

$$(\Sigma(x_{k-1}) \cap \Sigma_{for} \cap \Sigma_u = \emptyset) \ \& \ (\Sigma(x_{k-1}) \cap \Sigma_{for} \cap \Sigma_{hib} \neq \emptyset),$$

i.e., is ambiguously preemptable.

Hence the theorem. □

Noting that by definition, $\sigma \in \Sigma$ is not ambiguously controllable if $\sigma \in \Sigma_{act}$, or $\sigma = t_l$ and is non-preemptable or unambiguously preemptable, a logically straightforward corollary of Theorem 5.1 follows.

Corollary 5.1. *A TDES G_{lo} is AOCC-system refinable iff, for every τ -string $\langle s', \sigma_i, x_i, k, \tau \rangle \in L(G_{lo})$ with $\tau \in T_{act}$, the following condition holds: If $\sigma_1 \cdots \sigma_{k-1}$ is not controllable, then σ_k is not ambiguously controllable.*

We now present a conceptual procedure named Procedure **OCC-SR** for a TDES (G_{lo}, V) . We then show, by the proof of Theorem 5.2 below, that Procedure **OCC-SR** can be applied for OCC-system refinement of an AOCC-system refinable TDES (G_{lo}, V) .

The procedure is defined over the reachability tree $(G_{lo,t}, V_t)$, as follows: With $T_{act} = \{\tau_n \mid 1 \leq n \leq \kappa\}$, of set cardinality $|T_{act}| = \kappa$:

Step 1) Let $\{\alpha_n \mid 1 \leq n \leq \kappa\}$, $\{\beta_n \mid 1 \leq n \leq \kappa\}$ be the sets of new high-level outputs, where $\{\alpha_n\} \cap T_{act} = \emptyset$ and $\{\beta_n\} \cap T_{act} = \emptyset$ initially, and $\{\alpha_n\} \cap \{\beta_n\} = \emptyset$. For all n ($1 \leq n \leq \kappa$) and for every τ_n -string $s = \langle s', \sigma_i, x_i, k, \tau_n \rangle$ of $L(G_{lo,t})$, if τ_n is controllable w.r.t s , redefine $V_t(\delta_t(s, n_0)) = \alpha_n$; if τ_n is uncontrollable w.r.t s , redefine $V_t(\delta_t(s, n_0)) = \beta_n$; and if τ_n is ambiguously controllable w.r.t s , for some i ($1 \leq i \leq k-1$) such that $\sigma_i = t_l \ \& \ \Sigma(x_{i-1}) \cap \Sigma_{for} \cap \Sigma_{hib} \neq \emptyset$, and for all j ($i < j \leq k$), $\sigma_j \in \Sigma_u$ or $(\sigma_j = t_l \ \& \ \Sigma(x_{j-1}) \cap \Sigma_{for} = \emptyset)$, redefine $V_t(\delta_t(s' \sigma_1 \cdots \sigma_i, n_0)) = t_h$, and redefine $V_t(\delta_t(s, n_0)) = \beta_n$. By refining $(G_{lo,t}, V_t)$ as such, each original $\tau_n \in T_{act}$ is replaced by α_n or β_n accordingly, such that the modified set $T_{act} \subseteq \{\alpha_n, \beta_n\}$.

Step 2) Let $\{\overline{\alpha_n} \mid 1 \leq n \leq \kappa\}$, $\{\overline{\beta_n} \mid 1 \leq n \leq \kappa\}$ be the sets of new high-level outputs, where $\{\overline{\alpha_n}\} \cap T_{act} = \emptyset$ and $\{\overline{\beta_n}\} \cap T_{act} = \emptyset$ initially, and $\{\overline{\alpha_n}\} \cap \{\overline{\beta_n}\} = \emptyset$, $\{\overline{\alpha_n}\} \cap \{\alpha_n\} = \emptyset$ and $\{\overline{\beta_n}\} \cap \{\beta_n\} = \emptyset$. For each $\gamma \in \{\alpha_n, \beta_n\} \cap T_{act}$ and for every γ -string of $L(G_{lo,t})$ with reference prefix s' , if γ is not forcible w.r.t s' , then for every γ -string s of $L(G_{lo,t})$ with reference prefix w such that $\theta(w) = \theta(s')$, redefine $V_t(\delta_t(s, n_0)) = \overline{\gamma}$; and if γ is force-don't-care w.r.t s' , then for every γ -string s of $L(G_{lo,t})$ with reference prefix w such that $\theta(w) = \theta(s')$, redefine $V_t(\delta_t(s, n_0)) = \overset{x}{\gamma} \in \{\overline{\gamma}, \gamma\}^{5.2}$. By further refining $(G_{lo,t}, V_t)$ as such, each $\alpha_n \in T_{act}$ is further replaced by either α_n or $\overline{\alpha_n}$, each $\beta_n \in T_{act}$ is further replaced by either β_n or $\overline{\beta_n}$, such that the further modified set $T_{act} \subseteq \{\alpha_n, \beta_n, \overline{\alpha_n}, \overline{\beta_n}\}$.

Following the procedure, $T_{hib} = \{\alpha_n, \overline{\alpha_n}\} \cap T_{act}$; $T_u = \{\beta_n, \overline{\beta_n}\} \cap T_{act}$; and $T_{for} = \{\alpha_n, \beta_n\} \cap T_{act}$. In the maximal case, each enumerated τ_n ($1 \leq n \leq \kappa$) is replaced by four distinct outputs α_n , $\overline{\alpha_n}$, β_n and $\overline{\beta_n}$, and the maximal cardinality of the new T_{act} is 4κ .

Theorem 5.2. *A TDES G_{lo} is AOCC-system refinable iff it is OCC-system refinable.*

Proof. (If): Suppose a given TDES G_{lo} is OCC-system refinable. Then G_{lo} can be refined to be OCC, and hence AOCC and OFC by Definition 4.6. That the TDES G_{lo} can be refined

^{5.2}An event denoted by symbol $\overset{x}{\gamma}$ is simply called a force-don't-care event, and is either $\overline{\gamma}$ or γ .

to be AOCC implies it is AOCC-system refinable.

(*Only if*): Suppose a given TDES G_{lo} is AOCC-system refinable. It is sufficient to show that using Procedure **OCC-SR**, it can be refined to be AOCC, and then OFC without violating the established AOCC-system property, and hence OCC by Definition 4.6. The necessity proof proceeds as follows:

- Show that the given TDES G_{lo} can be refined to be AOCC:

An arbitrary $\tau_n \in T_{act}$ is, in general, either ambiguously controllable, controllable or uncontrollable w.r.t every τ_n -string of $L(G_{lo})$. By Corollary 5.1, for every τ -string $\langle s', \sigma_i, -, k, \tau \rangle \in L(G_{lo})$ with $\tau \in T_{act}$, the prefix $\sigma_i \cdots \sigma_{k-1}$ of its co-silent string is either controllable or ambiguously controllable, and its terminal event $\sigma_k \in \Sigma$ is, string-wise, either controllable or uncontrollable. Hence every τ_n -string $s = \langle s', \sigma_i, -, k, \tau_n \rangle$ of $L(G_{lo})$, w.r.t which $\tau_n \in T_{act}$ is ambiguously controllable, has the longest prefix $s'\sigma_1 \cdots \sigma_p$ for some p ($1 \leq p \leq k-1$) - at which $\sigma_p = t_l$ and is ambiguously preemptable, and beyond which the suffix $\sigma_{p+1} \cdots \sigma_k$ is uncontrollable. With the new event output notation accordingly defined, and over the reachability tree constructed for G_{lo} , Step 1 of Procedure **OCC-SR** labels such a prefix as a t_h -string and such a τ_n -string s as a β_n -string, of $L(G_{lo})$, with the new t_h -string $s'\sigma_1 \cdots \sigma_p$ as its reference prefix; and as a result, the new β_n -string s now satisfies the condition for an activity event in T_u as stated in Definition 4.1 (of activity output-control consistency).

The step also relabels every other τ_n -string $s \in L(G_{lo})$, w.r.t which $\tau_n \in T_{act}$ is controllable or uncontrollable, as an α_n -string or a β_n -string, of $L(G_{lo})$, respectively, and as a result, the new α_n - or β_n -string s now satisfies the condition for an activity event in T_{hib} or T_u , respectively, as stated in Definition 4.1.

It thus follows that Step 1 of Procedure **OCC-SR** refines the given TDES G_{lo} to be AOCC according to Definition 4.1, with the new set of high-level activity outputs $T_{act} \subseteq \{\alpha_n, \beta_n\}$ partitioned into $T_{hib} = \{\alpha_n\} \cap T_{act}$ and $T_u = \{\beta_n\} \cap T_{act}$.

- Show that the AOCC-system refined G_{lo} can be further refined to be OFC without

violating the established AOCC-system property:

With additional new event output notation accordingly defined, and over the reachability tree of the now AOCC-system G_{lo} , Step 2 of Procedure **OCC-SR** relabels α_n -strings and β_n -strings as $\overline{\alpha_n}$ - and $\overline{\beta_n}$ -strings, of $L(G_{lo})$, accordingly as needed, such that the refined G_{lo} becomes OFC according to Definition 4.4. As the step entails only output relabeling, it thus follows that every newly formed $\overline{\alpha_n}$ - or $\overline{\beta_n}$ -string of $L(G_{lo})$, like their α_n - or β_n -string counterpart, retains satisfying the condition for an activity event in T_{hib} or T_u , respectively, as stated in Definition 4.1.

It thus follows that Step 2 of Procedure **OCC-SR** refines the TDES G_{lo} to be OFC without violating the AOCC-system property established by Step 1, and hence the refined G_{lo} is OCC according to Definition 4.6, with $T_{act} \subseteq \{\alpha_n, \beta_n, \overline{\alpha_n}, \overline{\beta_n}\}$ partitioned into $T_{hib} = \{\alpha_n, \overline{\alpha_n}\} \cap T_{act}$ and $T_u = \{\beta_n, \overline{\beta_n}\} \cap T_{act}$; and partitioned with $T_{for} = \{\alpha_n, \beta_n\} \cap T_{act}$.

Hence the theorem. □

In subsequent references, Steps 1 and 2 of Procedure **OCC-SR** may be separately referred to as Subprocedures **AOCC-SR** and **OFC-SR**, respectively. Henceforth, when we say a TDES is OCC-, AOCC-, and OFC-system refinable, we now mean, more specifically, that it can be refined to be OCC, AOCC and OFC, using Procedure **OCC-SR** and Subprocedures **AOCC-SR** and **OFC-SR**, respectively.

Corollary 5.2. *An AOCC TDES G_{lo} is OCC-system refinable with timescale preservation.*

Proof. An AOCC G_{lo} is trivially AOCC-system refinable. By Theorem 5.2, G_{lo} is OCC-system refinable, i.e., it can be refined to be OCC by applying Procedure **OCC-SR**. Thus, as G_{lo} is already AOCC, it remains to apply Subprocedure **OFC-SR** of Procedure **OCC-SR** to refine G_{lo} to be OFC. Since Subprocedure **OFC-SR** refines by high-level activity event relabeling only, and therefore does not introduce any new t_h -string when refining G_{lo} , it follows that G_{lo} can be refined to be OCC with timescale preservation. Hence the corollary. □

Proposition 5.1. *Given a TDES G_{lo} that is AOCC-system refinable. Then G_{lo} is AOCC-system refinable with timescale^{5.3} preservation iff no τ -string of $L(G_{lo})$ with $\tau \in T_{act}$ exists w.r.t which τ is ambiguously controllable.*

Proof. Given a TDES G_{lo} that is AOCC-system refinable, i.e., it can be refined to be AOCC using Subprocedure **AOCC-SR**. As procedurally defined, Subprocedure **AOCC-SR** creates a new t_h -string iff there exists a τ -string of $L(G_{lo})$ with $\tau \in T_{act}$ w.r.t which τ is ambiguously controllable. Thus:

(If): No τ -string $s \in L(G_{lo})$ with $\tau \in T_{act}$ exists w.r.t which τ is ambiguously controllable implies no new t_h -string is introduced by Subprocedure **AOCC-SR**, which in turn implies the timescale of the given G_{lo} is preserved.

(Only if): That the timescale of G_{lo} is preserved by Subprocedure **AOCC-SR** implies no new t_h -string is introduced by the procedure, which in turn implies no τ -string of $L(G_{lo})$ with $\tau \in T_{act}$ exists w.r.t which τ is ambiguously controllable.

Hence the proposition. □

5.3 PF-System Refinement and SOCC-System

Refinability

As implied in Section 4.4, the terminology and concepts formulated for state pairs of G_{lo} carry over to node pairs of the corresponding tree $G_{lo,t}$.

In logical hierarchical control, a method over the system's Moore reachability tree is developed [1, 12] to break up vocal-node partners, by first finding them via breadth-first search of the tree. To refine a TDES (G_{lo}, V) so that it becomes free of vocal-state partners, however, it is discovered that, in finding vocal-node partners over $[s', w\sigma s'', <_{s_2}^{s_1}, j]$, similarly by breadth-first search of the tree $(G_{lo,t}, V_t)$, and breaking them up, new vocal-node partners may be introduced. This is because in breaking them up, the map V_t needs to be redefined so that the node $\delta_t(s'w\sigma, n_0)$ outputs t_h if $\sigma \in \Sigma(\delta_t(s'w, n_0))$ is an ambiguously

^{5.3}Note that a given timescale can be periodic or non-periodic.

preemptable tick, and otherwise, as in the partners-breakup method for logical hierarchical control [1, 12], outputs a given new activity event. The following example illustrates this issue.

Example 5.1. Consider a subtree of the reachability tree $(G_{lo,t}, V_t)$ for a Moore TDES G_{lo} , as depicted in Figure 5.1(a) with $s' \in L_{voc}(G_{lo,t})$, $w, s'' \in \Sigma^*$, $\sigma_i \in \Sigma_{act}$ for all $i \in \{1, 2, 3, 4, 5\}$ and $\tau_1, \tau_2 \in T_{act}$. The figure uses the same shorthand drawing notation for Figure 4.2. Suppose $\sigma_1, \sigma_2, \sigma_3$ and σ_5 are prohibitable, σ_4 is uncontrollable and σ_3 is also forcible.

(a) Partners (n_2, n_3)

(b) Partners (n_2, n_3) removed with new partners (n_1, n_4) introduced

Figure 5.1: An example showing that partnership removal can introduce new partnership

The subtree, according to Definition 4.9, has a pair of vocal-node partners, namely, (n_2, n_3) . To remove their partnership, V_t can be redefined such that the non-vocal node n_4 (reachable by string $s'ws_1s''t_l$) becomes a vocal node outputting t_h , as depicted in Figure 5.1(b). However, in so doing, a new pair of vocal-node partners is introduced, namely, (n_1, n_4) , as depicted in Figure 5.1(b). ■

In general, the issue is due to vocal-state partners being ‘hidden’ by certain pairs of control-dependent vocal states outputting the same activity event, formalized as follows.

Definition 5.1 (Hidden vocal-state partnership). *For a TDES G_{lo} , let $q_1, q_2 \in Q_{voc}$, where either $V(q_1)$ or $V(q_2)$ is an event of T_{act} . Then (q_1, q_2) is said to be a pair of partner-hiding states if $V(q_1) = V(q_2) \in T_{act}$ and (q_1, q_2) is a pair of control-dependent states over $[s', w\sigma s'', <_{s_2}^{s_1}, j]$, for which $(\exists s_c \in \Sigma^*)(s_c t_l < s_j \ \& \ \Sigma(\delta(s' w\sigma s'' s_c, q_0)) \cap \Sigma_{for} \cap \Sigma_{hib} \neq \emptyset)$.*

In words, two system vocal states q_1 and q_2 outputting the same activity event are partner-hiding if they are control dependent over $[s', w\sigma s'', <_{s_2}^{s_1}, j]$ in the system, such that an ambiguously preemptable tick exists at $q'_j = \delta(s' w\sigma s'' s_c, q_0)$ along some $s_c \in \Sigma^*$ for which $s_c t_l < s_j$, for which q_i and q'_j , $i \in \{1, 2\}$ and $i \neq j$, are vocal-state partners if q'_j were to output t_h .

To break up partners without introducing new partners, using bread-first search strategy, conservatively, pairs of vocal-state partners and partner-hiding states need to be broken up. In what follows, a conceptual procedure for PF-system refinement, named Procedure **PF-SR**, is defined over the reachability tree $(G_{lo,t}, V_t)$ of a TDES (G_{lo}, V) , as follows: For each pair (n_1, n_2) of vocal-node partners or partner-hiding nodes over $[s', w\sigma s'', <_{s_2}^{s_1}, j]$ in $G_{lo,t}$, detected with $s' w\sigma \in L(G_{lo,t})$ found by breadth-first search of the tree $G_{lo,t}$ starting from its root node, it is the case that

1. if $\sigma \in \Sigma(\delta_t(s' w, n_0))$ is ambiguously controllable, i.e., $\sigma = t_l$ and is ambiguously preemptable, then redefine $V_t(\delta_t(s' w\sigma, n_0)) = t_h$;
2. else redefine $V_t(\delta_t(s' w\sigma, n_0)) = \tau_p$, where $\tau_p \notin T$ is the given new activity event.

Definition 5.2 (Partner-hiding-state freeness). *A TDES G_{lo} is said to be partner-hiding-state free (PHF) if it does not contain partner-hiding (vocal) states.*

Therefore, by Definitions 4.10 and 5.2, Procedure **PF-SR** refines a TDES G_{lo} into a system that is not only PF, but also PHF.

Henceforth, a TDES is said to be SOCC-system refinable if it can be refined to be SOCC using the (ordered) application of Procedure **PF-SR** followed by Procedure **OCC-SR**, or either one of these two procedures.

5.4 SOCC-System Synthesis Theorems

Theorem 5.3. *A TDES G_{lo} is SOCC-system refinable if it is AOCC-system refinable and, over each $[s', w\sigma s'', <_{s_2}^{s_1}, j]$ of every pair (q_1, q_2) of vocal-state partners or partner-hiding states, it is the case that for $i, j, k \in \{1, 2\}$, $i \neq j$, and $s_k = s'_k \alpha_k$ where $\alpha_k \in \Sigma$,*

- *if $V(q_i) \neq t_h$ and s'_i is not controllable, then α_i is preemption-unambiguous;*
- *if $V(q_j) \neq t_h$, then α_j is uncontrollable.*

Proof. Given an AOCC-system refinable TDES G_{lo} with the structural conditions as specified. It is sufficient to show that, by applying Procedure **PF-SR** followed by Procedure **OCC-SR**, the given TDES G_{lo} can be refined to be PF without violating AOCC-system refinability, and then further refined to be OCC without violating the established PF-system property, and hence SOCC by Definition 4.11. The proof proceeds as follows:

- Show that the given TDES G_{lo} can be refined to be PF without violating AOCC-system refinability:

For each structure $[s', w\sigma s'', <_{s_2}^{s_1}, j]$ of every pair (q_1, q_2) of vocal-state partners (Definition 4.9) or partner-hiding states (Definition 5.1), both of which are control-dependent vocal states (Definition 4.8) in G_{lo} , Procedure **PF-SR**, in computing over the Moore reachability tree constructed for (G_{lo}, V) , labels string $s'w\sigma$ as a τ_p -string where τ_p is a new high-level activity output, only if in the co-silent string $w\sigma$ of the new τ_p -string, either $\sigma \in \Sigma_{hib}$ or $\sigma = t_l$ and is unambiguously preemptable. Therefore, the new τ_p -string with reference prefix s' satisfies the condition for AOCC-system refinability (required by each τ -string of $L(G_{lo})$ for every high-level activity event τ), as stated in Corollary 5.1.

Next, let $s_j = s'_j \alpha_j$ for $s'_j \in \Sigma^*$ and $\alpha_j \in \Sigma$, if $V(q_j) = \tau_j \in T_{act}$. Then, since s_j is not controllable by Definition 4.8 for control-dependent vocal states that vocal-state partners and partner-hiding states are, together with the condition that α_j is

uncontrollable, it follows that the new τ_j -string with reference prefix $s'w\sigma$ and co-silent string $s''s'_j\alpha_j$ satisfies the condition for AOCC-system refinability as stated in Corollary 5.1.

For $i \in \{1, 2\}, i \neq j$, let $s_i = s'_i\alpha_i$ for $s'_i \in \Sigma^*$ and $\alpha_i \in \Sigma$. It follows that if $V(q_i) = \tau_i \in T_{act}$ and s'_i is not controllable, then, that α_i is preemption-unambiguous implies α_i is either controllable or uncontrollable. Consequently, this implies that the new τ_i -string with reference prefix $s'w\sigma$ and co-silent string $s''s'_i\alpha_i$ satisfies the condition for AOCC-system refinability as stated in Corollary 5.1.

Hence, by refining G_{lo} as such using Procedure **PF-SR**, the refined G_{lo} is not only PF and PHF, implying it is PF, but also remains AOCC-system refinable according to Corollary 5.1.

- Show that the PF-system refined G_{lo} can be further refined to be OCC without violating the established PF-system property:

Since the refined G_{lo} is AOCC-system refinable, by Theorem 5.2, it is OCC-system refinable using Procedure **OCC-SR**.

An arbitrary $\tau_n \in T_{act}$ is, in general, either ambiguously controllable, controllable or uncontrollable w.r.t every τ_n -string of $L(G_{lo})$. As defined, over the reachability tree constructed for G_{lo} , Subprocedure **AOCC-SR** of Procedure **OCC-SR** simply relabels, accordingly, τ_n -strings of $L(G_{lo})$, w.r.t which $\tau_n \in T_{act}$ is controllable or uncontrollable. Effectively, no vocal-state partners are created herewith.

It remains to show that, in Subprocedure **AOCC-SR** relabeling τ_n -strings of $L(G_{lo})$ w.r.t which each $\tau_n \in T_{act}$ is ambiguously controllable, effectively no vocal-state partners are also created, as follows:

We note that a τ_n -string $s = s'\sigma_1 \cdots \sigma_k \in L(G_{lo})$, where $\sigma_i \in \Sigma$ for all i ($1 \leq i \leq k$) and s' is its reference prefix, w.r.t which $\tau_n \in T_{act}$ is ambiguously controllable, has the longest prefix $s'\sigma_1 \cdots \sigma_p$ for some p ($1 \leq p \leq k - 1$) - at which $\sigma_p = t_l$ and is ambiguously preemptable, and beyond which the suffix $\sigma_{p+1} \cdots \sigma_k$ is uncontrollable.

Herewith, Subprocedure **AOCC-SR** labels such a prefix as a t_h -string and re-labels τ_n -string s as some β_n -string accordingly, with the new t_h -string $s'\sigma_1 \cdots \sigma_p$ as its reference prefix. It then follows that, to prove by contradiction, assume that, due to the preceding refinement, q_1 is the new vocal state outputting the t_h and q_2 is some originally existent vocal state outputting an activity event such that (q_1, q_2) forms a pair of vocal-state partners over some structure $[s', w\sigma s'', <_{s_2}^{s_1}, j]$, where $q_1 = \delta(s'w\sigma s''s_1, q_0)$, $q_2 = \delta(s'w\sigma s''s_2, q_0)$ and $j \in \{1, 2\}$. Associating this structure with the form of the string s , the co-silent string of the t_h -string is $w\sigma s''s_1 = \sigma_1 \cdots \sigma_p$, which is ambiguously controllable as σ_p is, and so $\sigma_1 \cdots \sigma_{p-1}$ is either uncontrollable or ambiguously controllable. Together with Definition 4.8 for control-dependent vocal states that vocal-state partners are, it can only be that $\sigma = t_l$ and is ambiguously preemptable (i.e., σ is ambiguously controllable), and so the string $\sigma_1 \cdots \sigma_{p-1}$ must be ambiguously controllable. Thus, it follows that s_1 is not controllable and we may let $j = 1$.

In what follows, since state q_1 resides along the co-silent string of the initially given τ_n -string s , there exists a string $w'_1 \in (\Sigma_u \cup \{t_l\})^+$ such that $s = s'w\sigma s''s_1w'_1$. And since w'_1 is the co-silent string of the newly formed β_n -string s , $w'_1 = \sigma_{p+1} \cdots \sigma_k$ and is therefore uncontrollable. Since s_1 is not controllable and w'_1 is uncontrollable, it follows that the vocal state reachable by string s outputting τ_n initially and vocal state q_2 must initially be a pair of control-dependent vocal states over $[s', w\sigma s'', <_{s_2}^{s_1w'_1}, j]$, according to Definition 4.8. It follows that, if they output the same activity events, they form a pair of partner-hiding states by Definition 5.1, or otherwise form a pair of vocal-state partners by Definition 4.9, contradicting the fact that the G_{lo} is PF and PHF.

Therefore, refining G_{lo} to be AOCC using Subprocedure **AOCC-SR** does not introduce new vocal-state partners, implying the AOCC-system refined G_{lo} remains PF.

Finally, as defined, over the reachability tree constructed for G_{lo} , Subprocedure

OFC-SR of Procedure **OCC-SR** simply relabels τ_n -strings of $L(G_{lo})$ accordingly, and hence does not create new vocal states and therefore does not introduce new vocal-state partners.

All in all, the OCC-system refined G_{lo} remains PF.

Hence the theorem. □

A corollary of Theorem 5.3 follows.

Corollary 5.3. *An OCC TDES G_{lo} is SOCC-system refinable if, over each $[s', w\sigma s'', <_{s_2}^{s_1}, j]$ of every pair (q_1, q_2) of vocal-state partners or partner-hiding states, it is the case that for $i, j \in \{1, 2\}, i \neq j$,*

- *if $V(q_i) \neq t_h$, then s_i is preemption-unambiguous,*
- *if $V(q_j) \neq t_h$, then s_j is uncontrollable;*

and with timescale preservation if

- *$\sigma \in \Sigma$ is controllable.*

Proof. Given an OCC TDES G_{lo} with the conditions as specified for each structure $[s', w\sigma s'', <_{s_2}^{s_1}, j]$ of every pair (q_1, q_2) of vocal-state partners (Definition 4.9) or partner-hiding states (Definition 5.1), both of which are control-dependent vocal states (Definition 4.8) in G_{lo} . Taken together, the structural conditions (and these include the given fact that G_{lo} is OCC) can be logically shown to be stronger than the sufficiency conditions stated in Theorem 5.3 for an SOCC-system refinable TDES. It thus follows that G_{lo} is SOCC-system refinable.

Note that the application of Procedure **PF-SR** followed by Procedure **OCC-SR** is as implied by Theorem 5.3, in refining G_{lo} to be SOCC. Thus if, additionally, the event $\sigma \in \Sigma$ in the structure $[s', w\sigma s'', <_{s_2}^{s_1}, j]$ is controllable, then together with all the specified structural conditions of the given G_{lo} , it can be shown that applying Procedure **PF-SR** will not introduce any new t_h -string or new τ -string, of $L(G_{lo})$, w.r.t which $\tau \in T_{act}$ is

ambiguously controllable. As a result, applying Procedure **OCC-SR** subsequently will also not introduce any new t_h -string, thereby preserving the timescale.

Hence the corollary. □

5.5 STOCC-System Synthesis: A Discussion

Consider a hierarchy (G_{lo}, G_{hi}) built based on the proposed formulation. Suppose G_{lo} is SOCC and hence OCC. Based on the foregoing theoretical development, Property 2.3 and ALF (2.4.5) are satisfied for G_{hi} . Suppose we want the tick t_h for the high-level model G_{hi} to model real time, i.e., G_{hi} to possess time fidelity. What then remains to attain high-level time fidelity is that G_{hi} must also satisfy Property 2.2, and this is so provided G_{lo} is or can be refined to be OTC while remaining SOCC, and hence STOCC - a sufficient condition for HC-OTF. Any violation of Property 2.2 by G_{hi} is caused by either of two anomalies in the abstraction of the original Moore TDES^{5.4}, namely, either, upon a t_h -occurrence, an eligible high-level activity event becomes ineligible or it has one of its event-control properties modified.

W.r.t high-level time fidelity, the temporal dynamics of an SOCC G_{lo} being not OTC is deemed erroneous as the real-time soundness of all specifications w.r.t its abstraction model G_{hi} is not guaranteed. This necessitates a redesign of the reporter map θ by refining (G_{lo}, V) that removes the abstraction anomalies as well, and this general problem of existence and synthesis (by refinement) of an STOCC G_{lo} is a challenging one. In the next section, we study the SOCC-system refinability of an existent class of TDES's, and show that a 'linear' subclass formulated is STOCC-system refinable.

5.6 Hierarchical Consistency for NTU/NTI Systems

We consider a class of Moore TDES's, that we call Next-output Terminal-control Unambiguous (NTU) systems. NTU systems impose some output-system design or modeling

^{5.4}By original, we refer to the low-level TDES prior to undergoing Procedure **OCC-SR**.

restrictions in the resultant class of TDES hierarchies. A special subclass of NTU systems, called Non-terminal Time-control Invariant (NTI) systems, is also defined. In what follows, the SOCC-system synthesis of NTU TDES's is formally proved. A further restricted linear subclass of NTI systems is also characterized, which, importantly, lends itself to STOCC-system synthesis of linear NTI systems as also formally proved, and that entails a neat strategy of arbitrarily removing the abstraction anomalies identified without violating SOCC-system refinability in obtaining STOCC systems.

Definition 5.3 (NTU and NTI systems). *Let $s = s'\sigma_1\sigma_2 \cdots \sigma_k$ be an arbitrary τ -string of $L(G_{lo})$ with reference prefix s' and $\sigma_i \in \Sigma$ for all i ($1 \leq i \leq k$). Then the TDES G_{lo} is said to be NTU if, for each τ -string $s \in L(G_{lo})$ of every $\tau \in T_{act}$, the terminal event σ_k , if it is a tick t_l , is either non-preemptable or unambiguously preemptable. The TDES G_{lo} is said to be NTI if it is NTU and, for each τ -string $s \in L(G_{lo})$ of every $\tau \in T_{act} \cup \{t_h\}$, every (non-terminal) σ_i ($1 \leq i < k$) that is a tick t_l is either non-preemptable or unambiguously preemptable.*

For an NTU or NTI TDES G_{lo} , the terminal event $\sigma \in \Sigma$ in every τ -string $s\sigma \in L(G_{lo})$ of every output $\tau \in T_{act}$ is either an unambiguously preemptable or a non-preemptable tick, or an activity event (which is either prohibitable or uncontrollable). In other words, the terminal σ -control of the next activity output τ is unambiguous. For an NTI TDES G_{lo} , additionally, except the terminal tick t_l of every t_h -string of $L(G_{lo})$, the control preemptability of tick t_l elsewhere (i.e., whether t_l elsewhere can be preempted or not) is always the same under arbitrary system control dynamics. In other words, referring to t_l as a system non-terminal time tick whenever it is not the terminal event of a t_h -string of $L(G_{lo})$, the (low-level preemptive) control of non-terminal time ticks is invariant.

Remark 5.1. *We should point out that NTU TDES's are not a limited class of hierarchical systems. It is first discussed in [78] that, in general, a given TDES can be redesigned to become free of activity events, each of which is both forcible and prohibitable. A Moore TDES G_{lo} with $\Sigma_{for} \cap \Sigma_{hib} = \emptyset$ or equivalently, $\Sigma_{for} \subseteq \Sigma_w$, is clearly NTI since all its time*

ticks are control invariant, and hence is NTU.

In more detail, the system redesign can be done by explicitly modeling two different sets of circumstances for each event $\sigma \in \Sigma_{for} \cap \Sigma_{hib}$, in which it is known that for proper system operation, the same control action on the eligible event σ under the same set of circumstances entered, of either non-disabling or non-forcing, is mandatory and remains so following every consecutive tick, unless the system exits the circumstances and this happens when an activity event occurs. Therefore, two circumstantial (uncontrollable activity) events for the event σ need to be introduced to signify the system entering the respective circumstances. The non-forcing control action is also deemed mandatory at every system state where σ is eligible but not the tick t_l . So in essence, the redesign involves redefining the system state-transition function, such that in the redesigned system,

- *the two circumstantial events are defined at the same arbitrary state reachable by a system string s , where σ and t_l are both not eligible, and following each circumstantial event is a state where σ and t_l both become eligible for the first time since transiting temporally along some nonempty suffix of the string s , and*
- *that*
 - *every σ -transition on which non-disabling control is mandatory is relabeled as a σ_u -transition, where σ_u is the new forcible and uncontrollable event,*
 - *every σ -transition on which non-forcing control is mandatory is relabeled as a σ_c -transition, where σ_c is the new non-forcible and prohibitible event.*

■

5.6.1 SOCC-System Synthesis for NTU/NTI Systems

For NTU/NTI systems, two important results follow.

Theorem 5.4. *An NTU TDES G_{lo} is SOCC-system refinable.*

Proof. Given an NTU TDES G_{lo} . By Definition 5.3 of an NTU TDES and Corollary 5.1, G_{lo} is AOCC-system refinable.

Over $[s', w\sigma s'', <_{s_2}^{s_1}, j]$ of every pair (q_1, q_2) of vocal-state partners (Definition 4.9) or partner-hiding states (Definition 5.1), both of which are control-dependent vocal states (Definition 4.8) in G_{lo} , and letting $s_k = s'_k \alpha_k$ where $\alpha_k \in \Sigma$, $k \in \{1, 2\}$, it is the case that, since the given G_{lo} is NTU,

- if $V(q_i) \neq t_h$, then α_i is preemption-unambiguous;
- if $V(q_j) \neq t_h$, then α_j is uncontrollable.

Together, it follows that the NTU TDES G_{lo} satisfies the sufficiency conditions stated in Theorem 5.3 for an SOCC-system refinable TDES. Hence the theorem. $\square$

Theorem 5.5. *An NTI TDES G_{lo} is SOCC-system refinable with timescale preservation.*

Proof. An NTI TDES G_{lo} is NTU by Definition 5.3. Therefore, by Theorem 5.4, the TDES G_{lo} is SOCC-system refinable. This implies the application of Procedure **PF-SR** followed by Procedure **OCC-SR** in refining G_{lo} to be SOCC. Being NTI implies that the co-silent string of every τ -string of $L(G_{lo})$, where $\tau \in T_{act}$, is preemption-unambiguous. Therefore, applying Procedure **PF-SR** will not introduce any new t_h -string or new τ -string, of $L(G_{lo})$, w.r.t which $\tau \in T_{act}$ is ambiguously controllable. As a result, applying Procedure **OCC-SR** subsequently will also not introduce any new t_h -string, thereby preserving the timescale. Hence the theorem. $\square$

5.6.2 STOCC-System Synthesis for Linear NTI Systems

Fundamental to the linearity characterization of an NTI system is the system concept of linear time control-invariance, which requires the following definition.

Definition 5.4 (Timed-output-state control uniformity). *A TDES G_{lo} is said to be timed-output-state control uniform (w.r.t T_{act}) if, for all $\tau \in T_{act}$, if there exists a τ -string of $L(G_{lo})$ with reference prefix s' and a t_h -string of $L(G_{lo})$ with reference prefix w' such that $\theta(w') =$*

$\theta(s')$, then for every τ -string $s \in L(G_{lo})$ with reference prefix s'' such that $\theta(s'') = \theta(s')$, τ has the same controllability property w.r.t s and the same forcibility property w.r.t s'' .

The concept definition follows.

Definition 5.5 (Linear time control-invariance). *A TDES G_{lo} is linear time control-invariant (w.r.t T_{act}) if it is timed-output-state control uniform and NTI.*

Thus, a TDES G_{lo} is linear time control-invariant in the sense that

- it is timed-output-state control uniform, and thus at an arbitrary high-level state where a $\tau \in T_{act}$ and t_h are eligible, and the state is reachable from below by a string $s' \in L_{voc}(G_{lo})$, the string-wise control properties of $\tau \in T_{act}$ over every τ -string of $L(G_{lo})$ with reference prefix s'' such that $\theta(s'') = \theta(s')$ are the same; and
- it is NTI, and thus at an arbitrary low-level state, whenever a tick t_l is eligible, the control preemptability of the tick t_l under arbitrary system control dynamics is always the same if the tick t_l is not a terminal event of a t_h -string of $L(G_{lo})$.

Note that because a linear time control-invariant TDES is NTI, string-wise, no $\tau \in T_{act}$ is ambiguously controllable.

The two possible abstraction anomalies, of an eligible $\tau \in T_{act}$ becoming ineligible, and of it having one of its event-control properties modified upon a t_h -occurrence, are formalized as t_h -preemptability and t_h -property-modifiability, respectively, as follows.

Definition 5.6 (t_h -preemptability of $\tau \in T_{act}$). *Consider an arbitrary τ -string of $L(G_{lo})$ with $\tau \in T_{act}$ and reference prefix s' . Then τ is said to be t_h -preemptable w.r.t s' if there exists a t_h -string $w \in L(G_{lo})$ with reference prefix w' such that $\theta(s') = \theta(w')$, but there is no τ -string of $L(G_{lo})$ with reference prefix r such that $\theta(r) = \theta(w)$.*

Intuitively, t_h -preemptability of $\tau \in T_{act}$ characterizes the situation where a high-level event τ that is eligible is ‘preempted’ or becomes ineligible following an occurrence of t_h at a high-level state where τ and t_h are eligible.

A proposition relating system output time-compliance and t_h -preemptability follows.

Proposition 5.2. *A TDES G_{lo} is OTC iff, for every τ -string of $L(G_{lo})$ with reference prefix s' , where $\tau \in T_{act}$, τ is not t_h -preemptable w.r.t s' .*

Proof. For an arbitrary τ -string of $L(G_{lo})$ with reference prefix s' , where $\tau \in T_{act}$, by Definition 5.6 of t_h -preemptability, τ is not t_h -preemptable w.r.t s'

- provided that for every t_h -string $w \in L(G_{lo})$ with reference prefix w' such that $\theta(s') = \theta(w')$, there exists a τ -string of $L(G_{lo})$ with reference prefix r such that $\theta(r) = \theta(w)$;
- provided that if there exists a $tt_h \in \theta(L(G_{lo}))$ such that $\theta(s') = t$, then there exists a τ -string of $L(G_{lo})$ with reference prefix r such that $\theta(r) = \theta(w')t_h$;
- provided that, if there exists a t_h -string of $L(G_{lo})$ with reference prefix w' such that $\theta(s') = \theta(w')$, then there exists a τ -string of $L(G_{lo})$ with reference prefix r such that $\theta(r) = \theta(w')t_h$; and

provided that G_{lo} is OTC by Definition 4.5 of an OTC TDES. Hence the proposition. $\square$

Next is the t_h -property-modifiability of a $\tau \in T_{act}$. Essentially, it means that, for a TDES hierarchy (G_{lo}, G_{hi}) where $G_{hi} \stackrel{\text{def}}{=} (X, T, \xi, x_0, -)$, there is some reachable state $x \in X$ with $\tau \in T_{act}(x)$, and a state $x' = \xi(t_h, x) \in X$, for which there is no τ -string $s_2 \in L(G_{lo})$ with reference prefix s'' such that $x' = \xi(\theta(s''), x_0)$ and $\theta(s'') = tt_h$ for some $t \in T^*$ such that $x = \xi(t, x_0)$, and for which τ has the same controllability and forcibility properties w.r.t s_2 and s'' , respectively, as it respectively has w.r.t any τ -string $s_1 \in L(G_{lo})$ and s' , where s' is the reference prefix of s_1 such that $\theta(s') = t$. In this chapter, for a clearer exposition, it suffices to formally define this anomaly for a linear time control-invariant TDES G_{lo} .

Definition 5.7 (t_h -property-modifiability of $\tau \in T_{act}$). *Consider an arbitrary τ -string $s_1 \in L(G_{lo})$ with $\tau \in T_{act}$ and reference prefix s' , where TDES G_{lo} is linear time control-invariant. Then τ is said to be t_h -property-modifiable w.r.t s' if there exists a t_h -string $w \in L(G_{lo})$ with reference prefix w' , where $\theta(s') = \theta(w')$, and there exists a τ -string $s_2 \in L(G_{lo})$ with reference prefix s'' , where $\theta(s'') = \theta(w)$ such that, for every such $s_2 \in L(G_{lo})$,*

- either, τ is controllable w.r.t s_1 iff τ is not controllable w.r.t s_2 ,
- or, τ is forcible w.r.t s' iff τ is not forcible w.r.t s'' .

Intuitively, characterizing for a linear time control-invariant TDES G_{lo} , the t_h -property-modifiability of $\tau \in T_{act}$ refers to one ‘uniform’ string-wise event-control property (of either controllability or forcibility) of the high-level event τ changing completely, following an occurrence of t_h at a high-level state where τ and t_h are eligible.

Under system linear time control-invariance, and in the design spirit of minimizing the τ -strings of every $\tau \in T_{act}$, for which t_h -property modifiability holds w.r.t their reference prefixes, Definition 5.7 can be re-expressed in the following proposition.

Proposition 5.3. *Consider an arbitrary τ -string $s_1 \in L(G_{lo})$ with $\tau \in T_{act}$ and reference prefix s' , and an arbitrary t_h -string $w \in L(G_{lo})$ with reference prefix w' , where $\theta(s') = \theta(w')$. Let s_2 be an arbitrary τ -string of $L(G_{lo})$ with reference prefix s'' , where $\theta(s'') = \theta(w)$. Then, for a linear time control-invariant TDES G_{lo} , τ is t_h -property-modifiable w.r.t s' if, in the case where a t_h -string $w_1 \in L(G_{lo})$ with reference prefix w'_1 exists such that $\theta(w'_1) = \theta(w)$, there exists an $s_2 \in L(G_{lo})$ such that*

- either, τ is controllable w.r.t s_1 iff τ is not controllable w.r.t s_2 ,
- or, τ is forcible w.r.t s' iff τ is not forcible w.r.t s'' ,

and in the otherwise case, for every $s_2 \in L(G_{lo})$,

- τ is controllable w.r.t s_1 iff τ is not controllable w.r.t s_2 .

Proof. Given a linear time control-invariant TDES G_{lo} . Consider an arbitrary τ -string $s_1 \in L(G_{lo})$ with $\tau \in T_{act}$ and reference prefix s' , and an arbitrary t_h -string $w \in L(G_{lo})$ with reference prefix w' , where $\theta(s') = \theta(w')$. Let s_2 be an arbitrary τ -string of $L(G_{lo})$ with reference prefix s'' , where $\theta(s'') = \theta(w)$. Then there are two cases to consider:

Case 1: A t_h -string $w_1 \in L(G_{lo})$ with reference prefix w'_1 exists such that $\theta(w'_1) = \theta(w)$.

By Definition 5.5, since G_{lo} is linear time control-invariant, it is timed-output-state control uniform, implying by Definition 5.4 that for every such τ -string $s_2 \in L(G_{lo})$ (with

reference prefix s'' such that $\theta(s'') = \theta(w)$, τ has the same controllability w.r.t s_2 and the same forcibility w.r.t s'' . Hence, in this case, all such $s_2 \in L(G_{lo})$ can be equivalently reduced to one such τ -string s_2 in Definition 5.7 of t_h -property-modifiability of $\tau \in T_{act}$, for which

- either, τ is controllable w.r.t s_1 iff τ is not controllable w.r.t s_2 ,
- or, τ is forcible w.r.t s' iff τ is not forcible w.r.t s'' .

Case 2: No t_h -string $w_1 \in L(G_{lo})$ with reference prefix w'_1 exists such that $\theta(w'_1) = \theta(w)$.

In this otherwise case, for every τ -string $s_2 \in L(G_{lo})$, $\tau \in T_{act}$ is force-don't-care w.r.t the reference prefix s'' and therefore can be trivially defined as either forcible or not forcible w.r.t s'' . Therefore, the statement

τ is forcible w.r.t s' iff τ is not forcible w.r.t s''

in Definition 5.7 can be falsified.

Combining these two cases and rewriting Definition 5.7, the result follows. Hence the proposition. □

Based on Proposition 5.2, we may define a stronger concept of an OTC-system.

Definition 5.8 (Control output time-compliance). *A TDES G_{lo} is said to be control OTC if every $\tau \in T_{act}$ is neither t_h -preemptable nor t_h -property-modifiable w.r.t the reference prefix of each τ -string of $L(G_{lo})$.*

We now define the relative index for the starting state of the longest suffix of the co-silent string of a t_h -string in a TDES G_{lo} that may exist, along which the TDES will never diverge from entering a state outputting t_h .

Definition 5.9 (Output-time attractor limit). *The attractor limit for a t_h -string*

$\langle s', \sigma_i, -, k, t_h \rangle \in L(G_{lo})$, if it exists, is the smallest index b ($1 \leq b < k$) starting which the prefix $s'\sigma_1 \cdots \sigma_i$ for every i ($b \leq i < k$) cannot be extended to a τ -string of $L(G_{lo})$, where $\tau \in T_{act}$.

Intuitively, if no attractor limit b ($1 \leq b < k$) exists for a t_h -string $\langle s', \sigma_i, x_i, k, t_h \rangle \in L(G_{lo})$, it means that every prefix $s'\sigma_1 \cdots \sigma_i$ ($1 \leq i < k$) can be extended to some τ -string of $L(G_{lo})$, where $\tau \in T_{act}$. If it does, then evolving from TDES state x_0 , it is only after entering state x_b that the evolution towards state x_k or other vocal state that outputs t_h is guaranteed.

Two more system concepts, for a linear time control-invariant TDES, follow.

Definition 5.10. (Output-control determinism and anomalous output-time linear blockability): Consider a linear time control-invariant TDES G_{lo} . For every τ -string $\langle s', \sigma_i, x_i, k, \tau \rangle \in L(G_{lo})$ with $\tau \in T_{act}$, and for every τ' -string $\langle s', \alpha_j, -, h, \tau' \rangle \in L(G_{lo})$ with $\tau' \in T$, such that $\alpha_0 \alpha_1 \cdots \alpha_p = \sigma_0 \sigma_1 \cdots \sigma_p$ for some p ($0 \leq p < \min(h, k)$) where $\alpha_0 = \sigma_0 = \varepsilon$, the TDES G_{lo} is said to be

1. *output-control deterministic*, if the following condition holds: If $\tau' \neq \tau$, then for all i ($1 \leq i \leq p$), $\sigma_i \in \Sigma_u$ or $(\sigma_i = t_l \ \& \ \Sigma(x_{i-1}) \cap \Sigma_{for} = \emptyset)$; and
2. *anomalous output-time linearly blockable*, if the following condition holds: If, w.r.t s', τ is either t_h -preemptable or t_h -property-modifiable, then
 - if $\tau' = t_h$, the attractor limit b ($p + 1 \leq b < h$) exists such that for some j ($b \leq j < h$), $\alpha_j \in \Sigma_{act}$, and
 - for every t_h -string $\langle w', \beta_j, -, l, t_h \rangle \in L(G_{lo})$, where $\theta(w') = \theta(s')$ and w' cannot be extended (by a nonempty string) to a τ -string of $L(G_{lo})$, the attractor limit b' ($1 \leq b' < l$) exists such that for some j ($b' \leq j < l$), $\beta_j \in \Sigma_{act}$.

In words, for a linear time control-invariant TDES G_{lo} , it is output-control deterministic if, for every τ -string, $\tau \in T_{act}$ and for every τ' -string both with reference prefix s' , if $\tau \neq \tau'$ and their co-silent strings share the first p low-level events, then each shared event is either uncontrollable or t_l that is not preemptable. Intuitively, it ensures that every high-level prohibitable event can always be solely disabled and every preemptable t_h can always be solely preempted. G_{lo} is anomalous output-time linearly blockable if, for every τ -string

with reference prefix s' , $\tau \in T_{act}$, if τ is either t_h -preemptable or t_h -property-modifiable w.r.t s' , then for every t_h -string with reference prefix s' whose co-silent string shares first p events with the co-silent string of the τ -string, there must exist an attractor limit b for the t_h -string such that $b > p$ and the suffix of the co-silent string of the t_h -string started from b must contain a low-level activity event. For every t_h -string with reference prefix w' that cannot extend to a τ -string, there exists an attractor limit b' for the t_h -string such that the suffix of the co-silent string of the t_h -string started from b' contains a low-level activity event. Intuitively, anomalous output-time linearly blockability asserts that at a high-level state where t_h and an activity event are eligible, if the occurrence of t_h preempts the activity event or changes its controllability or forcibility property, then there must exist critical low-level activity events whose occurrence discontinues or alters the nature of the on-going high-level activity w.r.t time, respectively.

We are now ready to define a linear NTI system, and present its STOCC-system synthesis results.

Definition 5.11 (Linear NTI system). *A TDES G_{lo} is said to be linear NTI if it is linear time control-invariant, output-control deterministic and anomalous output-time linearly blockable.*

A lemma follows.

Lemma 5.1. *A linear NTI TDES G_{lo} is PF and PHF.*

Proof. For a linear NTI TDES G_{lo} , consider an arbitrary τ -string $\langle s', \sigma_i, x_i, k, \tau \rangle \in L(G_{lo})$, where $\tau \in T_{act}$ and an arbitrary τ' -string $\langle s', \alpha_j, -, h, \tau' \rangle \in L(G_{lo})$, where $\tau' \in T$, such that $\alpha_0 \alpha_1 \cdots \alpha_p = \sigma_0 \sigma_1 \cdots \sigma_p$ for some p ($0 \leq p < \min(h, k)$) where $\alpha_0 = \sigma_0 = \varepsilon$, and let $q_1 = \delta(s' \sigma_1 \sigma_2 \cdots \sigma_k, q_0) \in Q_{voc}$ and $q_2 = \delta(s' \alpha_1 \alpha_2 \cdots \alpha_h, q_0) \in Q_{voc}$. The proof then proceeds as follows.

To prove that G_{lo} is PF: Since G_{lo} is output-control deterministic by Definition 5.11 of a linear NTI TDES, it follows by Definition 5.10-1 of output-control determinism that, if $\tau \neq \tau'$, then for all i ($1 \leq i \leq p$), $\sigma_i \in \Sigma_u$ or $(\sigma_i = t_l \ \& \ \Sigma(x_{i-1}) \cap \Sigma_{for} = \emptyset)$. This implies

that string $\sigma_1 \cdots \sigma_p$ is uncontrollable. Hence, (q_1, q_2) is not a pair of control-dependent states as Condition CDS1 of Definition 4.8 is not satisfied. Therefore (q_1, q_2) is not a pair of vocal-state partners by Definition 4.9. By Definition 4.10, since G_{lo} does not have vocal-state partners, it is PF.

To prove that G_{lo} is PHF: By Definition 5.11, G_{lo} is also linear time control-invariant and therefore NTI by Definition 5.5. This implies that if $\tau = \tau' \in T_{act}$, the co-silent strings $\sigma_1 \cdots \sigma_k$ and $\alpha_1 \cdots \alpha_h$ of the τ -string and τ' -string of $L(G_{lo})$, respectively, do not contain an ambiguously preemptable tick t_l , and hence (q_1, q_2) is not a pair of partner-hiding states by Definition 5.1. By Definition 5.2, since G_{lo} does not have partner-hiding states, it is PHF.

Hence the lemma. $\square$

We now present a conceptual method named Method **STOCC-LNTI-SR** for a linear NTI TDES (G_{lo}, V) . The method uses another conceptual method named Method **COTC-SR**, which is presented first.

Method **COTC-SR** is defined over the reachability tree $(G_{lo,t}, V_t)$ as follows: For each $\tau \in T_{act}$, and for every τ -string $s = \langle s', \sigma_i, -, k, \tau \rangle \in L(G_{lo})$ for which τ is t_h -preemptable or t_h -property-modifiable w.r.t its reference prefix s' :

Step 1) Add a new activity output γ to T_{act} .

Step 2) For each t_h -string $\langle w', \alpha_j, -, h, t_h \rangle \in L(G_{lo})$ where $\theta(w') = \theta(s')$, find an index j such that $(b \leq j < h)$ and $\alpha_j \in \Sigma_{act}$, where b is the attractor limit of the t_h -string, and redefine $V_t(\delta_t(w' \alpha_1 \alpha_2 \cdots \alpha_j, n_0)) = \gamma$.

Method **STOCC-LNTI-SR** for a linear NTI TDES (G_{lo}, V) is now outlined in two steps, as follows:

Step 1) Refine the TDES G_{lo} by applying Method **COTC-SR**.

Step 2) Refine the model G_{lo} further by first applying Procedure **OCC-SR**, and thereafter fixing each force-don't-care event $\overset{x}{\gamma} \in \{\bar{\gamma}, \gamma\}$ in every $tt_h \overset{x}{\gamma} \in \theta(L(G_{lo}))$ as $\bar{\gamma}$ if $t\bar{\gamma} \in \theta(L(G_{lo}))$; and otherwise γ if $t\gamma \in \theta(L(G_{lo}))$.

Henceforth, a TDES is said to be STOCC-system refinable if it can be refined to be STOCC using Method **STOCC-LNTI-SR**.

Theorem 5.6. *A linear NTI TDES G_{lo} is STOCC-system refinable.*

Proof. Given a linear NTI TDES G_{lo} , which by Definition 5.11 is linear time control-invariant (Definition 5.5) [i.e., NTI (Definition 5.3) and timed-output-state control uniform (Definition 5.4)], output-control deterministic (Definition 5.10-1) and anomalous output-time linearly blockable (Definition 5.10-2).

We first show that, by applying Step 1 (i.e., Method **COTC-SR**) of Method **STOCC-LNTI-SR**, the given linear NTI TDES G_{lo} can be refined to be control OTC without violating the linear NTI-system property, as follows: Consider an arbitrary τ -string $s = \langle s', \sigma_i, x_i, k, \tau \rangle \in L(G_{lo})$, where $\tau \in T_{act}$.

- Show that the given G_{lo} can be refined to be control OTC:

If $\tau \in T_{act}$ is t_h -preemptable or t_h -property-modifiable w.r.t s' , Definitions 5.6 and 5.7 together imply that there must exist a t_h -string $w = \langle w', \alpha_j, -, h, t_h \rangle$ such that $\theta(w') = \theta(s')$. It follows that w' may or may not be the reference prefix of some τ -string of $L(G_{lo})$. Therefore, we have two cases to consider:

Case 1: The string w' is the reference prefix of some τ -string.

Then since τ is either t_h -preemptable or t_h -property-modifiable w.r.t s' , τ is also t_h -preemptable or t_h -property-modifiable w.r.t w' . By Definition 5.10-2 of anomalous output-time linearly blockability, it follows that there exists an attractor limit (of Definition 5.9), which is an index b such that along the co-silent string $\alpha_1 \cdots \alpha_h$ of w , for some j ($b \leq j < h$), $\alpha_j \in \Sigma_{act}$. Hence, in applying Method **COTC-SR**, one such index j can be found for redefining $V(\delta(w' \alpha_1 \alpha_2 \cdots \alpha_j, q_0)) = \gamma$, where γ is the new activity output introduced.

Case 2: The string w' is not the reference prefix of any τ -string.

Then since τ is either t_h -preemptable or t_h -property-modifiable w.r.t s' , by Definition 5.10-2 of anomalous output-time linearly blockability, it follows that there exists an attractor limit which is an index b' , such that along the co-silent string $\alpha_1 \cdots \alpha_h$ of w , for some j ($b' \leq j < h$), $\alpha_j \in \Sigma_{act}$. Hence similarly, in applying Method **COTC-SR**, one such index j can be found for redefining $V(\delta(w'\alpha_1\alpha_2 \cdots \alpha_j, q_0)) = \gamma$, where γ is the new activity output introduced.

In (effectively) redefining the vocalization map V as such, Method **COTC-SR** refines the TDES G_{lo} such that every $\tau \in T_{act}$ for the refined G_{lo} is no longer t_h -preemptable or t_h -property-modifiable w.r.t s' . Hence the refined G_{lo} does not contain a τ -string where τ is either t_h -preemptable or t_h -property-modifiable w.r.t its reference prefix, and therefore is control OTC by Definition 5.8.

- Show that the refined G_{lo} remains a linear NTI system:

To prove that G_{lo} remains NTI: As deduced from Definition 5.3, G_{lo} is NTI provided every event along the co-silent string of an arbitrary τ -string of $L(G_{lo})$ is not an ambiguously preemptable tick t_l , and so is every non-terminal event along the co-silent string of an arbitrary t_h -string of $L(G_{lo})$. For the given NTI G_{lo} , without relabeling or unlabeled any existing t_h -string (i.e., redefining it as a non- t_h -string or a non-vocal string, respectively), and only introducing each new γ -string as prescribed for each new γ added to T_{act} , Method **COTC-SR** clearly does not change this provision for the refined G_{lo} ; and hence the refined G_{lo} remains NTI.

To prove that G_{lo} remains timed-output-state control uniform: From the proof above showing that the given G_{lo} can be refined to be control OTC, it is clear that, for any new activity output γ , and therefore any new γ -string of $L(G_{lo})$ with reference prefix s' introduced by Method **COTC-SR**, there is no t_h -string with reference prefix w' such that $\theta(s') = \theta(w')$ in the refined G_{lo} , and the co-silent string of every τ -string in the given G_{lo} remains the same in the refined G_{lo} . Therefore, the refined G_{lo} remains timed-output-state control uniform by Definition 5.4.

At this juncture, we have proved that the refined G_{lo} is linear time control-invariant (Definition 5.5), to which Definition 5.10 is applicable.

To prove, by contradiction, that G_{lo} remains output-control deterministic: Suppose the refined G_{lo} is not output-control deterministic. This means that there exists a γ -string $\langle s', \sigma_i, x_i, k, \gamma \rangle$ and a τ' -string $\langle s', \alpha_j, -, h, \tau' \rangle$ where γ is a new activity output, $\tau' \in T$ and $\tau' \neq \gamma$, such that $\alpha_0 \alpha_1 \cdots \alpha_p = \sigma_0 \sigma_1 \cdots \sigma_p$ for some p ($0 \leq p < \min(h, k)$) where $\alpha_0 = \sigma_0 = \varepsilon$, and for some i ($1 \leq i \leq p$), $\sigma_i \notin \Sigma_u$ and $\sigma_i = t_l \implies \Sigma(x_{i-1}) \cap \Sigma_{for} \neq \emptyset$. Then since before redefining as γ , $V(\delta(s' \sigma_1 \sigma_2 \cdots \sigma_k, q_0)) = \tau_o$, it must be that in the given G_{lo} , there exists a t_h -string $\langle s', \sigma'_j, x'_j, k', t_h \rangle$ such that $\sigma_1 \sigma_2 \cdots \sigma_k = \sigma'_1 \sigma'_2 \cdots \sigma'_k$ and $V(x'_1) = \cdots = V(x'_{k'-1}) = \tau_o$. With Method **COTC-SR** redefining similarly for every t_h -string with reference prefix w' such that $\theta(w') = \theta(s')$, it must be that $\tau' \in T_{act}$. Together, it means that in the given G_{lo} , there exists a t_h -string $\langle s', \sigma'_j, x'_j, k', t_h \rangle$ and a τ' -string $\langle s', \alpha_j, -, h, \tau' \rangle$ where $\tau' \neq t_h$, such that $\alpha_0 \alpha_1 \cdots \alpha_p = \sigma'_0 \sigma'_1 \cdots \sigma'_p$ for some p ($0 \leq p < \min(k', h)$) and for some j ($1 \leq j \leq p$), $\sigma_j \notin \Sigma_u$ and $\sigma_j = t_l \implies \Sigma(x'_{j-1}) \cap \Sigma_{for} \neq \emptyset$, contradicting the fact that the given G_{lo} is output-control deterministic. Hence, the refined G_{lo} is output-control deterministic by Definition 5.10-1.

To prove that G_{lo} remains anomalous output-time linearly blockable: As proved above, the refined G_{lo} is control OTC (Definition 5.8). It follows by Definition 5.10-2 that it is trivially anomalous output-time linearly blockable.

Together, the refined G_{lo} that is control OTC and hence OTC by Definition 5.8 and Proposition 5.2, is a linear NTI system by Definition 5.11.

We then show, by applying Step 2 of Method **STOCC-LNTI-SR**, that the TDES G_{lo} can be further refined to be SOCC without violating the established OTC-system property, and hence to be STOCC by Definition 4.11, as follows:

Remaining linear NTI, the OTC-system refined G_{lo} is, by Lemma 5.1, PF and PHF, which is the same as a TDES refined using Procedure **PF-SR**. Therefore, by Theorem 5.4,

G_{lo} , an NTI and therefore NTU TDES by Definition 5.3, can be further refined to be SOCC by applying the remaining Procedure **OCC-SR**. Because G_{lo} is NTI, no co-silent string of every τ -string of $L(G_{lo})$, where $\tau \in T_{act}$, is ambiguously controllable, and hence no new t_h -string is introduced by Procedure **OCC-SR** in Step 2 of Method **STOCC-LNTI-SR**. Together with the fact that the TDES G_{lo} to be further refined is control OTC, after applying Step 2 of Method **STOCC-LNTI-SR**, no (control relabeled) $\gamma \in T_{act}$ in the refined G_{lo} is t_h -preemptable or t_h -property-modifiable, string-wise. Hence the SOCC-system refined G_{lo} remains control OTC and hence OTC by Definition 5.8 and Proposition 5.2.

Hence the theorem. □

A corollary of Theorem 5.6 follows.

Corollary 5.4. *A linear NTI TDES G_{lo} is STOCC-system refinable with timescale preservation.*

Proof. By Theorem 5.6, a linear NTI TDES G_{lo} is STOCC-system refinable (using Method **STOCC-LNTI-SR**). Only Subprocedure **AOCC-SR** of Procedure **OCC-SR** used in Method **STOCC-LNTI-SR** introduces new t_h -strings as needed. Let $\tau \in T_{act}$. For an arbitrary $s \in L_{voc}(G_{lo})$, Subprocedure **AOCC-SR** labels an $swt_l \in L(G_{lo})$ as a new t_h -string of $L(G_{lo})$ with the same vocal string s as its reference prefix, only if wt_l , where the tick t_l is ambiguously preemptable, is the strict prefix of the co-silent string of some τ -string of $L(G_{lo})$ (w.r.t which τ is ambiguously controllable). But since G_{lo} is linear NTI and therefore NTI by Definition 5.11 of a linear NTI TDES, every tick t_l that exists along the co-silent string of an arbitrary τ -string of $L(G_{lo})$ is not ambiguously preemptable, as deduced from Definition 5.3 of an NTI TDES. This means that no τ -string of $L(G_{lo})$ exists whose co-silent string contains an ambiguously preemptable tick t_l , and hence no new t_h -string is introduced in the methodical STOCC-system refinement of this G_{lo} , preserving the given timescale. Hence the corollary. □

5.7 Hierarchical Control of a Photocopying System - a Linear NTI System

The STOCC-system synthesis for linear NTI TDES's is illustrated, with the necessity for output-time fidelity reiterated, using a simplified but non-trivial photocopying machine that takes a photo snapshot of every properly placed document page and saves it as software image file.

5.7.1 System Description

The machine is a system composition G of two real-time component TDES's: a photocopier G_1 and a page positioner G_2 . The ATG's $G_{1,act}$ and $G_{2,act}$ with their associated timing information, by which the respective components G_1 and G_2 (not shown) are constructed, are shown in Figures 5.2(a) and 5.2(b); and the composed system $G = G_1 ||| G_2$ is shown in Figure 5.2(c). The event set $\Sigma_{act} = \{\sigma_i \mid 1 \leq i \leq 6\}$ is partitioned with $\Sigma_{hib} = \{\sigma_3\}$ and $\Sigma_{for} = \{\sigma_5\}$. The definitions of the system events are given in the 'Events' row of Table 5.1.

The dynamics of the system components are described as follows. Following a 1-tick joint initialization or re-initialization of sensors, the system components are both ready to begin the next photocopying cycle. When a document page in the input tray is detected (σ_1), the page positioner and photocopier are jointly alerted. Following, the page positioner takes 1 tick to ready itself, and up to 1 subsequent tick to pull the page from the input tray and position it in the photocopy area (σ_5). Concurrently, the photocopier takes 2 ticks to set up, and up to 2 more ticks to photocopy and save the photocopied as a software file (σ_4). Upon executing σ_4 , the photocopier may clear any page in the photocopy area into the input tray (σ_3), or following a 1 tick-delay, take up to 1 more tick to clear the page in the photocopy area into the output tray (σ_2). Upon executing σ_5 followed by a tick, the page positioner readies itself for the next document page (σ_6).

	Symbol: Meaning
Events	σ_1 : next document page in input tray detected σ_2 : page in photocopy area cleared into output tray σ_3 : photocopy area cleared moving any page there into input tray σ_4 : photocopy-as-software-file action executed σ_5 : page pulled from input tray and positioned in photocopy area σ_6 : ready for next photocopying cycle
Outputs (given)	τ_1 : next page placed for photocopying (vocalized after σ_1) τ_2 : page processed (vocalized after σ_2) τ_3 : page photocopy assured (vocalized after $\sigma_1\sigma_5$) τ_4 : page failed to be photocopied (vocalized after $\sigma_1\sigma_4$) τ_5 : page left in photocopy area (vocalized after $\sigma_3\sigma_5$)
Output (added)	τ_6 : page to be re-processed (vocalized after σ_3)

Table 5.1: Symbol definitions of events and outputs for the photocopying system

The intricate timed interleaving of the events between the system components is captured in the TDES model G .

5.7.2 Initial Hierarchical System Design

Suppose that in the initial hierarchical system design, we are given the set of high-level activity events, such that each is defined as an output by vocal states of a Moore version G_{lo} shown in Figure 5.2(d) of the composed system G , entered following a respective low-level activity event or a string of low-level activity events (in ATG $G_{1,act} \parallel G_{2,act}$). The system outputs of interest pertain to the photocopying cycle. The definitions of these system outputs are given in the ‘Outputs (given)’ row of Table 5.1. The outputs correspond to ‘next page placed for photocopying’, signalled as high-level event τ_1 when σ_1 occurs; ‘page processed’ signalled as τ_2 when σ_2 occurs; ‘page photocopy assured’ signalled as τ_3 when σ_5 is the next activity event to occur after σ_1 ; ‘page failed to be photocopied’ signalled as τ_4 when σ_4 is the next activity event to occur after σ_1 ; and ‘page left in photocopy area’

signalled as τ_5 when σ_5 occurs immediately after σ_3 .

(a) Photocopier ATG $G_{1,act}$ and associated event timings (b) Page positioner ATG $G_{2,act}$ and associated event timings

(c) Composed system model G

(d) Given Moore system model G_{lo}

(e) An output-set enlarged Moore system model G_{lo} for STOCC-system design

Figure 5.2: Photocopying machine models for hierarchical control system design

5.7.3 Linear NTI System - A Verification

In the following, we verify that the given G_{lo} of Figure 5.2(d) with $T_{act} = \{\tau_i \mid 1 \leq i \leq 5\}$ is linear NTI (Definition 5.11).

- G_{lo} is NTI (Definition 5.3) because no t_l in G_{lo} is ambiguously preemptable (Definition 2.1).
- G_{lo} is timed-output-state control uniform (Definition 5.4) since:
 - For each $\gamma \in T_{act} - \{\tau_2\}$ and for an arbitrary γ -string $s \in L(G_{lo})$ with reference prefix s' , there does not exist a γ -string $w \in L(G_{lo})$, $w \neq s$, with reference prefix s'' such that $\theta(s'') = \theta(s')$. Therefore, each γ trivially satisfies the condition required by every $\tau \in T_{act}$ for timed-output-state control uniformity.
 - For $\tau_2 \in T_{act}$, consider an arbitrary τ_2 -string of $L(G_{lo})$ with reference prefix s' . For each

$$s' \in R_2 = \{t_l\sigma_1 t_l t_l \sigma_4, t_l\sigma_1 t_l t_l \sigma_5, t_l\sigma_1 t_l t_l \sigma_5\},$$

there exists a t_h -string of $L(G_{lo})$ with reference prefix s' ; and for every τ_2 -string $s \in L(G_{lo})$ with reference prefix s'' such that $\theta(s'') = \theta(s')$ (where we note that $s'' = s'$), τ_2 stays uncontrollable w.r.t the given s and forcible w.r.t the string s'' . It can be inferred from R_2 - the set of representative reference prefixes for τ_2 - and the structural regularity of (finite-state) G_{lo} that the reference prefix of every τ_2 -string is the reference prefix of some t_h -string, both of $L(G_{lo})$, and for every τ_2 -string $s \in L(G_{lo})$ with reference prefix s'' , τ_2 is uncontrollable w.r.t s and forcible w.r.t s'' . Thus, τ_2 satisfies the condition required by every $\tau \in T_{act}$ for timed-output-state control uniformity.

- G_{lo} is output-control deterministic (Definition 5.10-1) since:
 - For each $\gamma \in T_{act} - \{\tau_5\}$ and for every γ -string $s \in L(G_{lo})$, since its co-silent string contains only events in $\Sigma_u \cup \{t_l\}$, of which every t_l present is non-

preemptable, trivially, γ satisfies the condition required by every τ -string of $L(G_{lo})$, $\tau \in T_{act}$, for output-control determinism.

- For $\tau_5 \in T_{act}$, for the τ_5 -string $s'\sigma_3\sigma_5 \in L(G_{lo})$ with reference prefix $s' = t_l\sigma_1t_lt_l\sigma_4$ and co-silent string $\sigma_3\sigma_5$, since $s'\sigma_3$ cannot be extended into states vocalizing outputs other than τ_5 , this τ_5 -string trivially satisfies the condition required by every τ -string of $L(G_{lo})$, $\tau \in T_{act}$, for output-control determinism. It can be inferred from the representative reference prefix $s' = t_l\sigma_1t_lt_l\sigma_4$ for τ_5 and the structural regularity of G_{lo} that every τ_5 -string of $L(G_{lo})$ satisfies the condition required by every τ -string of $L(G_{lo})$, $\tau \in T_{act}$, for output-control determinism.
- G_{lo} is anomalous output-time linearly blockable (Definition 5.10-2) since (string-wise), no $\tau \in T_{act}$ is t_h -property modifiable, and only $\tau_2, \tau_5 \in T_{act}$ are t_h -preemptable:
 - τ_2 is t_h -preemptable w.r.t $s' \in R_2$ (with R_2 defined earlier above), τ_5 is t_h -preemptable w.r.t $t_l\sigma_1t_lt_l\sigma_4 \in R_2$. It can be inferred from the representative reference prefix set R_2 for $\{\tau_2, \tau_5\}$ and the structural regularity of G_{lo} that $\tau_2, \tau_5 \in T_{act}$ are t_h -preemptable w.r.t every of their reference prefixes.

Along the co-silent string of every t_h -string with the same reference prefix as a τ -string of $L(G_{lo})$, where $\tau \in \{\tau_2, \tau_5\}$, there exists a non-vocal state entered via a transition of event σ_3 , from which the system cannot be extended into a state vocalizing activity outputs. Thus, for every such t_h -string, an attractor limit exists (at every state entered via a transition of σ_3).

In what follows, we explain how to refine the given TDES G_{lo} to build a consistent hierarchy, but which is one that has some design time anomalies.

5.7.4 SOCC-System Synthesis and High-Level Time Fidelity Issues

The given G_{lo} is NTI and hence NTU. By Theorem 5.4, it is SOCC-system refinable using Procedure **PF-SR** followed by Procedure **OCC-SR**. By Lemma 5.1, since the given G_{lo} is linear NTI, it is PF and PHF. To refine it into an SOCC-system, it remains to apply Procedure **OCC-SR**, to relabel accordingly and unambiguously associate every $\tau \in T_{act}$ with the event-control properties. Note that following Step 1 of Procedure **OCC-SR**, τ_4 becomes β_4 ; and following Step 2, string-wise, β_4 is found to be always force-don't-care, and defaulted to non-forcible with relabel $\overline{\beta_4}$. The OCC-system refined G_{lo} is also partner-free, and hence is SOCC.

The refined hierarchy (G_{lo}, G_{hi}) , where G_{lo} is SOCC, is shown in Figure 5.3. By Theorem 4.2, it is HC.

Note that in the abstracted model G_{hi} shown in Figure 5.3(a), the eligibility of giving assurance that a page can be photocopied (β_3) is invariant under high-level time tick transition. Importantly, this aspect of timing semantics captures a critical fact that any high-level tick delay in giving such assurance can result in the imminent and uncontrollable possibility of page-photocopy failure ($\overline{\beta_4}$) while the assurance is still in progress. The eligibility of a page placed and ready for photocopying ($\overline{\beta_1}$) is also invariant under high-level time tick transition, and that of $\overline{\beta_4}$ is trivially so. However, high-level activity events β_2 and $\overline{\alpha_5}$ do not comply with such timed eligibility invariance, and thus G_{hi} violates Property 2.2, or equivalently, the STOCC G_{lo} violates the OTC-system property.

Violating Property 2.2 although Property 2.3 and ALF (2.4.5) are satisfied, model G_{hi} in Figure 5.3(a) does not possess time fidelity. It follows that, as similarly illustrated in the example system depicted in Figure 4.1, a high-level real-time specification such as ensuring ‘at most one high-level tick for page processing (β_2) completion’ has unsound timing semantics w.r.t G_{hi} . Certain high-level specifications might still have sound timing semantics, but without going underneath the abstraction to understand the low-level system dynamics of the non-OTC G_{lo} , studying the high-level model G_{hi} alone poses difficulty

Figure 5.3: A hierarchical photocopying system (G_{lo}, G_{hi}) that is HC

for a high-level control designer to identify and prescribe with confidence any correct and required high-level control specification for G_{hi} . As a matter of fact, one wonders what unabstracted low-level activity event or string of low-level events occurs along with the tick t_h , causing the system G_{hi} in Figure 5.3(a) to, upon the t_h -occurrence, cancel the processing (β_2) or prevent a page from being left in the photocopy area ($\overline{\alpha_5}$).

In what follows, we explain how to refine the given TDES G_{lo} in Fig 5.2(d) to build a consistent hierarchy with output-time fidelity.

5.7.5 STOCC-System Synthesis

Because the given G_{lo} is linear NTI as established earlier, by Theorem 5.6, it is STOCC-system refinable (using Method **STOCC-LNTI-SR**). From a design perspective, the missing high-level information is the signal that a document page needs to be re-processed. So it turns out that, although technically not necessarily the only way, we may introduce a new high-level event τ_6 to represent this information, to be output by every state entered upon the occurrence of σ_3 . With this event introduction, listed in the ‘Output (added)’ row of Table 5.1, the modified G_{lo} , as shown in Figure 5.2(e), becomes OTC with no $\tau \in T_{act}$ that is string-wise t_h -preemptable or t_h -property-modifiable, as shown in Figure 5.2(e). Applying Procedure **OCC-SR** to this modified G_{lo} , every $\tau \in T_{act}$ is relabeled accordingly and unambiguously associated with the event-control properties. Again, note that, following Step 1 of Procedure **OCC-SR**, τ_2, τ_4, τ_5 and τ_6 become $\beta_2, \beta_4, \alpha_5$ and α_6 , respectively; and following Step 2, string-wise, $\beta_2, \beta_4, \alpha_5$ and α_6 are found to be always force-don’t-care, and defaulted to non-forcible, with relabels $\overline{\beta_2}, \overline{\beta_4}, \overline{\alpha_5}$ and $\overline{\alpha_6}$, respectively.

The desired refined hierarchy (G_{lo}, G_{hi}) , where G_{lo} is STOCC, is shown in Figure 5.4. By Theorem 4.2, it is HC-OTF.

Figure 5.4: A hierarchical photocopying system (G_{lo}, G_{hi}) that is HC-OTF

5.7.6 Hierarchical Control Specification

As a specification example over G_{hi} shown in Figure 5.4(a), we may now assert the requirement that every document page is to be photocopied once without failure. The specification TTG for this requirement is shown in Figure 5.5. A high-level supervisor (not shown) may be synthesized using standard real-time control theory [74].

Figure 5.5: A (high-level) specification TTG for the model G_{hi} in Figure 5.4(a)

Note that a specialized untimed hierarchical nonblockingness result that entails the reporter map to be a system marked language observer (see [12] for details) is applicable to TDES's within the same framework of formal languages and finite automata. Although hierarchical consistency, as formalized by Definition 4.7 and Theorem 4.2, does not deal with marked states, it can be shown by applying this nonblockingness result that the photocopying system hierarchy (G_{lo}, G_{hi}) in Figure 5.4 admits nonblocking high-level supervisor that can be realized or implemented by a corresponding low-level supervisor generating prefix-closed sublanguages.

5.8 Framework Generalization and Scalability

Our theory development of formulating and building a consistent control hierarchy began with adopting the control-theoretic formulation of G (2.4.2) [74] as the base TDES model and our formulation of the timed reporter map θ . This puts our contribution in the context of a useful control-theoretic system model possessing time fidelity that G (2.4.2) is for real-time system and control design [74], with the added control-theoretic postulation of $\Sigma_{hib} \subseteq \Sigma_{rem}$ being sufficient for proving Property 2.3. It should be clear that the theory

part on hierarchical consistency with output-time fidelity still applies as long as a given base TDES model possesses time fidelity, and the part on that without output-time fidelity guarantee still applies even if the base TDES model has Property 2.2 relaxed, in which case we are using a TDES model where *tick* still represents time but is not always ‘behaviorally’ real time as it can be a timeout (see Footnote 1.1 on page 10) - an event denoting a time elapse in simultaneity with some implied action whose transition may disrupt the eligibility of activity events.

In concluding, we note that the hierarchical consistency for two levels may be extended to multiple levels. Once hierarchical consistency is achieved, either of the type without or with output-time fidelity guarantee as desired for, say $(G_{lo,0}, G_{hi,0})$ - the base level and initial level up - by refining accordingly, the Moore TDES $G_{lo,0}$ that is NTU or linear NTI, respectively, the constructions may be repeated by first assigning state outputs in $G_{hi,0}$ (according to the time-output design laws and respective NTU or linear NTI system modeling constraints) to obtain a Moore TDES $G_{lo,1}$ as desired, and then bringing in the next higher level, $G_{hi,1}$. Clearly, by similarly refining the TDES $G_{lo,1}$ obtained, the hierarchical consistency of the same type for $(G_{lo,1}, G_{hi,1})$ as attained for $(G_{lo,0}, G_{hi,0})$ can be achieved without disturbing the consistency of $(G_{lo,0}, G_{hi,0})$. In principle, therefore, as with the logical framework [1], our real-time framework is vertically scalable.

5.9 Conclusion

In this chapter, supporting SOCC-system existence and synthesis results are developed. With the aid of these supporting results, the new results for SOCC-system synthesis for hierarchical consistency, and STOCC-system synthesis for hierarchical consistency with output-time fidelity, are proved for the class of NTU systems and its subclass of linear NTI systems, respectively. An example is presented to demonstrate the concepts and system synthesis for a linear NTI TDES.

Chapter 6

Conclusion & Future Research

6.1 System Abstraction for Logical Hierarchical Control

The first part of this thesis, presented in Chapter 3, considers the problem of on-line command and control of logical DES's. Based on the results of output control feasibility and system decomposition, the command and control design proposed for hierarchical control of DES's is shown to be implementable. The implementation entails the off-line computation of the high-level controlled DES and the reusable τ -uncontrollable state sets, instead of the larger scale, low-level controlled DES. This operational design is attractive as it not only offers the intrinsic merit of furnishing causal clarity of command over control during on-line operation, but is also shown to significantly reduce off-line synthesis complexity and achieve fast on-line computation. The major off-line complexity reduction comes from doing away with computation involving the inverse mapping θ^{-1} . The fast on-line control computation is due to linear-time complexity of the main on-line load of computing low-level control data $f_{con}(f_{com}, -)$ (3.4.3); this computation involves simply checking state membership of the uncontrollable state set $B_u^{f_{com}}$ (3.3.5). These practical advantages are useful for engineering control problems, where the required runtime operational mode is command and control, or the control solution needs to be subjected to validation tests by

simulation and observation.

One direction for future work is to consider deploying on-line limited lookahead control [46] at the high level of the command and control hierarchy. Doing so, it might be possible to decrease off-line computation further without significantly increasing on-line computation. Besides, by using on-line control at the high-level and treating every high-level specification as a control task, the foundation laid in this research could pave the way towards full on-line hierarchical control of DES's that can support a flexible and operationally clearer command-to-control transfer for sequential or nonconcurrent multi-tasking.

Another interesting direction for future work is to extend the command and control synthesis theory for DES's presented in this thesis to TDES's. This entails an investigation into, and a formulation of output uncontrollable and controllable state sets that can handle forcible events and tick preemption.

6.2 System Abstraction for Real-Time Hierarchical Control

The second part of this thesis, presented in Chapters 4 and 5, considers the problem of system abstraction for hierarchical consistency of timed DES's (TDES's) with output-time fidelity. In the theoretical development, the concepts of output-control consistency and partner-freeness for hierarchical control are generalized, from untimed DES's [1] to TDES's [78] where time fidelity need not be respected in the sense of not obeying Property 2.2; and the foundation is then augmented with the new concept of output time-compliance for a class of Moore TDES's that possesses time fidelity, to develop a new real-time control-theoretic framework for hierarchical control where output-time fidelity is also respected, i.e., the framework of hierarchical consistency with output-time fidelity. Under this framework, supporting SOCC-system existence and synthesis results are presented, on which the results, of SOCC-system synthesis for hierarchical consistency and STOCC-system synthesis for hierarchical consistency with output-time fidelity, are proved for the mildly restrictive

class of NTU systems and its subclass of linear NTI systems, respectively.

This research adds to the conceptually rich real-time control theory for DES's developed in the TTG-modeling formalism that originated with the work of [74]. Using TTG modeling anchors the control theory in the same framework of formal languages and finite automata as its untimed counterpart, and the key intellectual merits lie in a unified theoretical foundation. Formalized over controllable, high-level prefix-closed system sublanguages as in the logical version [1], hierarchical consistency does not ensure control nonblockingness at the high level by low-level control implementation; only the prefix-closure of high-level nonempty controllable sublanguages can be realized, unless some hierarchical observer condition holds as briefly mentioned at the end of Section 5.7. Using the key system concepts of output-control consistency developed in this thesis, the logical theory of hierarchical consistency with marking [12] can be extended to a timed framework, under which high-level nonblocking supervisor can be implemented by low-level nonblocking supervisor.

Not unexpectedly, a serious trade-off of the TTG modeling formalism is the inevitable incurrence of large computational complexities by the key hierarchical-system synthesis procedures conceptualized in this thesis, and by the real-time extendible untimed control synthesis procedures [72], when dealing with large-scale base TDES's. That these complexities arise due primarily to system composition of component automaton-models is well understood, and they are exacerbated by tick transitions and the Moore automaton structure of TTG's used for system modeling. To graduate from theory to practice, future research will need to address and mitigate the complexity issues. Three promising and mutually supporting approaches come to mind. One is to adapt and cast the control theory onto a more efficient modeling formalism (e.g., State Tree Structures [153] and extended finite automata [155, 156]), and for which synthesis procedures can be implemented using symbolic computation with binary decision diagrams that has found success in terms of efficiency improvement over traditional automata-based approaches for synthesizing monolithic supervisors. Two is to extend to real time and leverage on the untimed synthesis approach

for on-line command and control [157]. Three is to generalize the theory to decentralized hierarchical abstraction of modular TDES's by extending the idea and method from an un-timed version [92], of building the consistent abstraction of a system from the composition of consistent abstractions of its system components.

Author's Publications

Journal Papers

1. Q. H. Ngo and K. T. Seow, "A Hierarchical Consistency Framework for Real-Time Supervisory Control," 2016. (Submitted)
2. Q. H. Ngo and K. T. Seow, "Command and control of discrete-event systems: Towards on-line hierarchical control based on feasible system decomposition," *IEEE Transactions on Automation Science and Engineering*, vol. 11, no. 4, pp. 1218-1228, October 2014.

Conference Papers

1. Q. H. Ngo and K. T. Seow, "A Time Fidelity Control Foundation for Hierarchical Discrete-Event Systems," in *Proceedings of the IEEE International Conference on Automation Science and Engineering (CASE'14)*, August 2014, Taipei, Taiwan, pp. 443-448.
2. Q. H. Ngo and K. T. Seow, "Hierarchical control of discrete-event systems: A new command and control design based on feasible system decomposition," in *Proceedings of the IEEE International Conference on Automation Science and Engineering (CASE'12)*, Seoul, Korea, August 2012, pp. 674-679.

Bibliography

- [1] H. Zhong and W. M. Wonham, “On the consistency of hierarchical supervision in discrete-event systems,” *IEEE Transactions on Automatic Control*, vol. 35, no. 10, pp. 1125–1134, October 1990.
- [2] C. G. Cassandras and S. Lafortune, *Introduction to Discrete Event Systems*. Springer, 2008.
- [3] P. J. Ramadge and W. M. Wonham, “The control of discrete event systems,” *Proceedings of the IEEE*, vol. 77, no. 1, pp. 81–98, January 1989.
- [4] B. Hruz and M. Zhou, *Modeling and control of Discrete-Event Dynamic Systems: With Petri nets and other tools*. Springer, 2007, vol. 59.
- [5] J. Zhao, Y.-L. Chen, Z. Chen, F. Lin, C. Wang, and H. Zhang, “Modeling and control of discrete event systems using finite state machines with variables and their applications in power grids,” *Systems & Control Letters*, vol. 61, no. 1, pp. 212–222, 2012.
- [6] J. S. Ostroff and W. M. Wonham, “A framework for real-time discrete event control,” *IEEE Transactions on Automatic Control*, vol. 35, no. 4, pp. 386–397, April 1990.
- [7] H. Wong-Toi and G. Hoffman, “The control of dense real-time discrete event systems,” in *Proceedings of the 30th IEEE International Conference on Decision and Control*, Brighton, England, December 1988, pp. 1527–1528.

- [8] R. Alur and D. L. Dill, “A theory of timed automata,” *Theoretical computer science*, vol. 126, no. 2, pp. 183–235, 1994.
- [9] J. Wang, *Timed Petri nets: Theory and Application*. Springer Science & Business Media, 2012, vol. 9.
- [10] M. Lawford and W. M. Wonham, “Supervisory control of probabilistic discrete event systems,” in *Proceedings of the 36th Midwest Symposium on Circuits and Systems*, Detroit, Michigan, U.S.A, August 1993, pp. 327–331.
- [11] R. Kumar and V. K. Garg, “Control of stochastic discrete event systems modeled by probabilistic languages,” *IEEE Transactions on Automatic Control*, vol. 46, no. 4, pp. 593–606, 2001.
- [12] W. M. Wonham, *Supervisory Control of Discrete-Event Systems*. Systems Control Group, University of Toronto, Canada, 2016 (Updated annually), <http://www.control.toronto.edu/cgi-bin/dldes.cgi>.
- [13] R. Kumar and V. K. Garg, *Modeling and Control of Logical Discrete Event Systems*. Kluwer Academic Publishers, 1995, the Kluwer International Series in Engineering and Computer Science.
- [14] J. Moody and P. J. Antsaklis, *Supervisory control of discrete event systems using Petri nets*. Springer Science & Business Media, 2012, vol. 8.
- [15] Y.-C. Wu and S. Lafortune, “Synthesis of insertion functions for enforcement of opacity security properties,” *Automatica*, vol. 50, no. 5, pp. 1336–1348, 2014.
- [16] Y.-C. Wu, K. A. Sankararaman, and S. Lafortune, “Ensuring privacy in location-based services: An approach based on opacity enforcement,” in *Proceedings of the 12th International Workshop of Discrete Event Systems*, vol. 12, Cachan, France, May 2014, pp. 33–38.

- [17] N. B. Hadj-Alouane, S. Lafrance, F. Lin, J. Mullins, and M. M. Yeddes, “On the verification of intransitive noninterference in multilevel security,” *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, vol. 35, no. 5, pp. 948–958, 2005.
- [18] N. B. Hadj-Alouane, S. Lafrance, F. Lin, J. Mullins, and M. Yeddes, “Characterizing intransitive noninterference for 3-domain security policies with observability,” *IEEE Transactions on Automatic Control*, vol. 50, no. 6, pp. 920–925, June 2005.
- [19] E. Dallal and S. Lafortune, “On most permissive observers in dynamic sensor activation problems,” *IEEE Transactions on Automatic Control*, vol. 59, no. 4, pp. 966–981, April 2014.
- [20] P. Antsaklis, “On control and cyber-physical systems: challenges and opportunities for discrete event and hybrid systems,” in *Proceedings of the 9th International Workshop on Discrete Event Systems*, Goteborg, Sweden, 2008, pp. 1–1.
- [21] H. Hu and M. Zhou, “A petri net-based discrete-event control of automated manufacturing systems with assembly operations,” *IEEE Transactions on Control Systems Technology*, vol. 23, no. 2, pp. 513–524, 2015.
- [22] H. Yue, K. Xing, H. Hu, W. Wu, and H. Su, “Robust supervision using shared-buffers in automated manufacturing systems with unreliable resources,” *Computers & Industrial Engineering*, vol. 83, pp. 139–150, 2015.
- [23] B. A. Brandin, “The real-time supervisory control of an experimental manufacturing cell,” *IEEE Transactions on Robotics and Automation*, vol. 12, no. 1, pp. 1–14, February 1996.
- [24] R. Malik and R. Leduc, “Hierarchical modelling of manufacturing systems using discrete event systems and the conflict preorder,” *Discrete Event Dynamic Systems: Theory and Applications*, vol. 25, no. 1, pp. 177–201, 2015.

- [25] H. Liao, S. Lafortune, S. Reveliotis, Y. Wang, and S. Mahlke, “Optimal liveness-enforcing control for a class of petri nets arising in multithreaded software,” *IEEE Transactions on Automatic Control*, vol. 58, no. 5, pp. 1123–1138, 2013.
- [26] H. Liao, Y. Wang, J. Stanley, S. Lafortune, S. Reveliotis, T. Kelly, and S. Mahlke, “Eliminating concurrency bugs in multithreaded software: A new approach based on discrete-event control,” *IEEE Transactions on Control Systems Technology*, vol. 21, no. 6, pp. 2067–2082, 2013.
- [27] Y. Wang, P. Liu, T. Kelly, S. Lafortune, S. Reveliotis, and C. Zhang, “On atomicity enforcement in concurrent software via discrete event systems theory,” in *Proceedings of the 51st IEEE International Conference on Decision and Control*, Hawaii, U.S.A, December 2012, pp. 7230–7237.
- [28] K. T. Seow, M. Pasquier, and M. L. Hong, “Supervising Passenger Land-Transport Systems,” *IEEE Transactions on Intelligent Transportation Systems*, 1999, submitted Feb 1999.
- [29] —, “Vehicle Route-Sequence Planning For Passenger Travel Service,” *Transportation Science*, 1999, submitted Nov 1999.
- [30] —, “Hierarchical Route Planning of Pickup-and-Delivery : Towards A Basis For Tasks-Directed Vehicle Route Guidance,” *International Journal of Artificial Intelligence in Engineering*, 2000, submitted Mar 2000.
- [31] E. Dallal, A. Colombo, D. D. Vecchio, and S. Lafortune, “Supervisory control for collision avoidance in vehicular networks using discrete event abstractions,” in *Proceedings of the American Control Conference*, Washington, DC, U.S.A, 2013, pp. 4380–4386.
- [32] M. Dotoli, N. Epicoco, M. Falagario, and G. Cavone, “A timed petri nets model for intermodal freight transport terminals,” in *Proceedings of the 12th International Workshop on Discrete Event Systems*, vol. 12, Cachan, France, 2014, pp. 176–181.

- [33] J. Zhao, Y.-L. Chen, Z. Chen, F. Lin, C. Wang, and H. Zhang, “Modeling and control of discrete event systems using finite state machines with variables and their applications in power grids,” *Systems & Control Letters*, vol. 61, no. 1, pp. 212–222, 2012.
- [34] B. Zhao, F. Lin, C. Wang, X. Zhang, M. P. Polis, and L. Y. Wang, “Power distribution network management using networked control of timed discrete event systems,” in *Proceedings of the IEEE 53rd Annual Conference on Decision and Control (CDC)*, Los Angeles, CA, U.S.A, 2014, pp. 5949–5954.
- [35] P. J. Ramadge and W. M. Wonham, “Supervisory control of a class of discrete event processes,” *SIAM Journal of Control and Optimization*, vol. 25, no. 1, pp. 206–230, January 1987.
- [36] E. Chen and S. Lafortune, “Dealing with blocking in supervisory control of discrete event systems,” in *Proceedings of the 28th IEEE International Conference on Decision and Control*, Tampa, Florida, U.S.A, December 1989, pp. 117–122.
- [37] M. Fabian and R. Kumar, “Mutually nonblocking supervisory control of discrete event systems,” *Automatica*, vol. 36, no. 12, pp. 1863–1869, 2000.
- [38] L. Feng, K. Cai, and W. M. Wonham, “A structural approach to the non-blocking supervisory control of discrete-event systems,” *The International Journal of Advanced Manufacturing Technology*, vol. 41, no. 11, pp. 1152–1168, 2008.
- [39] M. Heymann and F. Lin, “On-line control of partially observed discrete event systems,” *Discrete Event Dynamic Systems: Theory and Applications*, vol. 4, no. 3, pp. 221–236, 1994. [Online]. Available: <http://dx.doi.org/10.1007/BF01438708>
- [40] F. Lin and W. M. Wonham, “Supervisory control of timed discrete-event systems under partial observation,” *IEEE Transactions on Automatic Control*, vol. 40, no. 3, pp. 558–562, 1995.

- [41] F. Lin, “Robust and adaptive supervisory control of discrete event systems,” *IEEE Transactions on Automatic Control*, vol. 38, no. 12, pp. 1848–1852, December 1993.
- [42] S. E. Bourdon, M. Lawford, and W. M. Wonham, “Robust nonblocking supervisory control of discrete-event systems,” *IEEE Transactions on Automatic Control*, vol. 50, no. 12, pp. 2015–2021, 2005.
- [43] A. Saboori and S. Hashtrudi-Zad, “Robust nonblocking supervisory control of discrete-event systems under partial observation,” *Systems & control letters*, vol. 55, no. 10, pp. 839–848, 2006.
- [44] J. G. Thistle and W. M. Wonham, “Supervision of infinite behaviour of discrete event systems,” *SIAM Journal of Control and Optimization*, vol. 32, no. 4, pp. 1098–1113, July 1994.
- [45] S.-L. Chung, S. Lafortune, and F. Lin, “Limited lookahead policies in supervisory control of discrete event systems,” *IEEE Transactions on Automatic Control*, vol. 37, no. 12, pp. 1921–1935, December 1992.
- [46] N. B. Hadj-Alouane, S. Lafortune, and F. Lin, “Variable lookahead supervisory control with state information,” *IEEE Transactions on Automatic Control*, vol. 39, no. 12, pp. 2398–2410, December 1994.
- [47] S.-L. Chung, S. Lafortune, and F. Lin, “Supervisory control using variable lookahead policies,” *Discrete Event Dynamic Systems: Theory and Applications*, vol. 4, no. 3, pp. 237–268, July 1994.
- [48] T. Ushio, “On-line control of discrete event systems with a maximally controllable and observable sublanguage,” *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. 82, no. 9, pp. 1965–1970, 1999.

- [49] F. Boroomand and S. Hashtrudi-Zad, “A limited lookahead policy in robust non-blocking supervisory control of discrete event systems,” in *Proceedings of the American Control Conference*, Washington, DC, U.S.A, June 2013, pp. 935–939.
- [50] M. T. Pham, A. Dhananjayan, and K. T. Seow, “On specification transparency: Towards a formal framework for human designer comprehensibility of discrete-event control specifications in finite automata,” *IEEE Transactions on Systems, Man and Cybernetics: Systems*, vol. 43, no. 1, pp. 139–148, January 2013.
- [51] K. T. Seow, “Integrating temporal logic as a state-based specification language for discrete-event control design in finite automata,” *IEEE Transactions on Automation Science and Engineering*, vol. 4, no. 3, pp. 451–464, 2007.
- [52] K. Rudie and W. M. Wonham, “Think globally, act locally : Decentralized supervisory control,” *IEEE Transactions on Automatic Control*, vol. 37, no. 11, pp. 1692–1708, November 1992.
- [53] S. Takai and T. Ushio, “Reliable decentralized supervisory control of discrete event systems,” *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 30, no. 5, pp. 661–667, 2000.
- [54] T.-S. Yoo and S. Lafortune, “A general architecture for decentralized supervisory control of discrete-event systems,” *Discrete Event Dynamic Systems: Theory and Applications*, vol. 12, no. 3, pp. 335–377, 2002.
- [55] W. M. Wonham and P. J. Ramadge, “Modular supervisory control of discrete-event systems,” *Mathematics of Control, Signals and Systems*, vol. 1, no. 1, pp. 13–30, January 1988.
- [56] T. Yamasaki and T. Ushio, “Decentralized supervisory control of discrete event systems based on reinforcement learning,” *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. 88, no. 11, pp. 3045–3050, 2005.

- [57] R. C. Hill, D. M. Tilbury, and S. Lafortune, “Modular supervisory control with equivalence-based abstraction and covering-based conflict resolution,” *Discrete Event Dynamic Systems : Theory and Applications*, vol. 20, no. 1, pp. 139–185, March 2010.
- [58] K. Hiraishi, “On solvability of a decentralized supervisory control problem with communication,” *IEEE Transactions on Automatic Control*, vol. 54, no. 3, pp. 468–480, March 2009.
- [59] A. Jayasiri, G. K. I. Mann, and R. G. Gosine, “Modular supervisory control and hierarchical supervisory control of fuzzy discrete-event systems,” *IEEE Transactions on Automation Science and Engineering*, vol. 9, no. 2, pp. 353–364, April 2012.
- [60] F. Lin and W. M. Wonham, “Decentralized control and coordination of discrete-event systems with partial observation,” *IEEE Transactions on Automatic Control*, vol. 35, no. 12, pp. 1330–1337, 1990.
- [61] K. T. Seow, C. Ma, and M. Yokoo, “Multiagent planning as control synthesis,” in *Proceedings of the Third International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS’04)*, Columbia University, New York City, U.S.A, July 2004, pp. 972–979.
- [62] K. T. Seow, M. T. Pham, C. Ma, and M. Yokoo, “Coordination planning: Applying control synthesis methods for a class of distributed agents,” *IEEE Transactions on Control Systems Technology*, vol. 17, no. 2, pp. 405–415, March 2009.
- [63] M. T. Pham and K. T. Seow, “Discrete-event coordination design for distributed agents,” *IEEE Transactions on Automation Science and Engineering*, vol. 9, no. 1, pp. 70–82, 2012.
- [64] M. T. Pham, “Discrete-event multiagent coordination: Framework and algorithms,” School of Computer Engineering, Nanyang Technological University, Singapore, Doctor of Philosophy (Ph.D) Thesis, August 2011.

- [65] K. Cai and W. M. Wonham, "Supervisor localization: a top-down approach to distributed control of discrete-event systems," *IEEE Transactions on Automatic Control*, vol. 55, no. 3, pp. 605–618, 2010.
- [66] R. Zhang, K. Cai, Y. Gan, and W. M. Wonham, "Distributed supervisory control of discrete-event systems with communication delay," *Discrete Event Dynamic Systems: Theory and Applications*, pp. 1–31, 2014.
- [67] K. C. Wong and W. M. Wonham, "Hierarchical control of discrete-event systems," *Discrete Event Dynamic Systems : Theory and Applications*, vol. 6, no. 3, pp. 241–273, July 1996.
- [68] K. T. Seow, "Organizational control of discrete-event systems: A hierarchical multi-world supervisor design," *IEEE Transactions on Control Systems Technology*, vol. 22, no. 1, pp. 23–33, January 2014.
- [69] A. E. C. da Cunha and J. E. R. Cury, "Hierarchical supervisory control based on discrete event systems with flexible marking," *IEEE Transactions on Automatic Control*, vol. 52, no. 12, pp. 2242–2253, December 2007.
- [70] K. Schmidt, M. H. de Queiroz, and J. E. R. Cury, "Hierarchical and decentralized multitasking control of discrete event systems," in *Proceedings of the 46th IEEE International Conference on Decision and Control*, New Orleans, LA, U.S.A, December 2007, pp. 5936–5941.
- [71] S. Eilenberg, *Automata, Languages and Machines : Volume A*. Academic Press, New York, 1974.
- [72] S. Yi, "Hierarchical supervision with nonblocking," Graduate Department of Electrical and Computer Engineering, University of Toronto, Canada, Master of Applied Science (MASc) Thesis, June 2004.

- [73] R. C. Hill, J. E. R. Cury, M. H. de Queiroz, D. M. Tilbury, and S. Lafortune, “Multi-level hierarchical interface-based supervisory control,” *Automatica*, vol. 46, no. 7, pp. 1152–1164, July 2010.
- [74] B. A. Brandin and W. M. Wonham, “Supervisory control of timed discrete-event systems,” *IEEE Transactions on Automatic Control*, vol. 39, no. 2, pp. 329–341, February 1994.
- [75] K. Q. Pu, “Modeling and control of discrete-event systems with hierarchical abstraction,” Graduate Department of Electrical and Computer Engineering, University of Toronto, Canada, Master of Applied Science (MAsc) Thesis, March 2000.
- [76] E. A. Lee, “Computing needs time,” *Communications of the ACM*, vol. 52, no. 5, pp. 70–79, 2009.
- [77] ———, “CPS foundations,” in *Proceedings of the 47th Design Automation Conference (DAC 2010)*. ACM, June 2010, pp. 737–742.
- [78] K. C. Wong and W. M. Wonham, “Hierarchical control of timed discrete-event systems,” *Discrete Event Dynamic Systems : Theory and Applications*, vol. 6, no. 3, pp. 275–306, July 1996.
- [79] W. M. Wonham and P. J. Ramadge, “On the supremal controllable sublanguage of a given language,” *SIAM Journal of Control and Optimization*, vol. 25, no. 3, pp. 637–659, May 1987.
- [80] J. E. Hopcroft and J. D. Ullman, *Introduction to Automata Theory, Languages and Computation*. Reading, MA : Addison-Wesley, 1979.
- [81] C. G. Cassandras and S. Lafortune, “Ch 2 : Languages and Automata,” in *Introduction to Discrete Event Systems*, 2nd ed. Springer-Verlag, New York, 2008, pp. 53–132.

- [82] S.-J. Park and K.-H. Cho, “Real-time preemptive scheduling of sporadic tasks based on supervisory control of discrete event systems,” *Information Sciences*, vol. 178, no. 17, pp. 3393–3401, September 2008.
- [83] S.-J. Park and J.-M. Yang, “Supervisory control for real-time scheduling of periodic and sporadic tasks with resource constraints,” *Automatica*, vol. 45, no. 11, pp. 2597–2604, November 2009.
- [84] H. Hu, M. Zhou, Z. Li, and Y. Tang, “Deadlock-free control of automated manufacturing systems with flexible routes and assembly operations using petri nets,” *IEEE Transactions on Industrial Informatics*, vol. 9, no. 1, pp. 109–121, February 2013.
- [85] J. S. Lee, M. Zhou, and P. L. Hsu, “A petri-net approach to modular supervision with conflict resolution for semiconductor manufacturing systems,” *IEEE Transactions on Automation Science and Engineering*, vol. 4, no. 4, pp. 584–588, October 2007.
- [86] R. J. Leduc, , M. Lawford, and P. Dai, “Hierarchical interface-based supervisory control of flexible manufacturing system,” *IEEE Transactions on Control Systems Technology*, vol. 14, no. 4, pp. 654–668, July 2006.
- [87] L. Grigorov, “Hierarchical control of discrete-event systems,” School of Computing, Queen’s University, Canada,” Unpublished survey paper, 2005. [Online]. Available: <http://research.cs.queensu.ca/home/grigorov/depth.pdf>
- [88] M. Z. Fekri and S. Hashtrudi-Zad, “Hierarchical supervisory control of discrete-event systems under partial observation,” in *Proceedings of the 48th IEEE International Conference on Decision and Control*, Shanghai, China, December 2009, pp. 181 –186.
- [89] K. Schmidt and C. Breindl, “Maximally permissive hierarchical control of decentralized discrete event systems,” *IEEE Transactions on Automatic Control*, vol. 56, no. 4, pp. 723–737, April 2011.

- [90] O. Boutin, J. Komenda, T. Masopust, K. Schmidt, and J. H. van Schuppen, “Hierarchical control with partial observations: Sufficient conditions,” in *Proceedings of the 51th IEEE International Conference on Decision and Control*, Orlando, FL, U.S.A., December 2011, pp. 1817 – 1822.
- [91] M. Z. Fekri and S. Hashtrudi-Zad, “Hierarchical robust supervisory control of discrete-event systems,” in *Proceedings of the American Control Conference*, Seattle, Washington, U.S.A., June 2008, pp. 1178–1183.
- [92] A. E. C. da Cunha, J. E. R. Cury, and B. H. Krogh, “An assume-guarantee reasoning for hierarchical coordination of discrete event systems,” in *Proceedings of the 6th International Workshop on Discrete-Event Systems*, Zaragoza, Spain, October 2002, pp. 75–80.
- [93] R. C. Hill and D. M. Tilbury, “Modular supervisory control of discrete-event systems with abstraction and incremental hierarchical construction,” in *Proceedings of the 8th International Workshop on Discrete Event Systems*, Ann Arbor, Michigan, U.S.A., July 2006, pp. 399–406.
- [94] K. Schmidt and C. Breindl, “On maximal permissiveness of hierarchical and modular supervisory control approaches for discrete event systems,” in *Proceedings of the 9th International Workshop on Discrete Event Systems*, Goteborg, Sweden, May 2008, pp. 462–467.
- [95] J. Komenda, T. Masopust, and J. H. van Schuppen, “Bottom-up approach to multi-level supervisory control with coordination,” in *Proceedings of the European Control Conference*, Strasbourg, France, 2014, pp. 2715–2720.
- [96] K. Schmidt, J. Reger, and T. Moor, “Hierarchical control for structural decentralized DES,” *Proceedings of the IFAC Workshop on Discrete Event Systems*, pp. 289–295, September 2004.

- [97] K. Schmidt, T. Moor, and S. Perk, “A hierarchical architecture for nonblocking control of decentralized discrete event systems,” in *Proceedings of the 13th Mediterranean Conference on Control and Automation*, Limassol, Cyprus, June 2005, pp. 902–907.
- [98] K. Schmidt and T. Moor, “Marked-string accepting observers for the hierarchical and decentralized control of discrete event systems,” in *Proceedings of the 8th International Workshop on Discrete-Event Systems*, Ann Arbor, MI, U.S.A, July 2006, pp. 413–418.
- [99] K. Schmidt, H. Marchand, and B. Gaudin, “Modular and decentralized supervisory control of concurrent discrete event systems using reduced system models,” in *Proceedings of the 8th International Workshop on Discrete-Event Systems*, Ann Arbor, MI, U.S.A, July 2006, pp. 149–154.
- [100] K. Schmidt, T. Moor, and S. Perk, “Nonblocking hierarchical control of decentralized discrete event systems,” *IEEE Transactions on Automatic Control*, vol. 53, no. 10, pp. 2252–2265, November 2008.
- [101] L. Feng and W. M. Wonham, “Supervisory control architecture for discrete-event systems,” *IEEE Transactions on Automatic Control*, vol. 53, no. 6, pp. 1449–1461, 2008.
- [102] T. Moor, J. Raisch, and J. M. Davoren, “Admissibility criteria for a hierarchical design of hybrid control systems,” in *Proceedings of the IFAC Conference on Analysis and Design of Hybrid Systems*, St. Malo, Brittany, France, June 2003, pp. 389–394.
- [103] S. Perk, T. Moor, and K. Schmidt, “Hierarchical discrete event systems with inputs and outputs,” in *Proceedings of the 8th International Workshop on Discrete Event Systems*, Ann Arbor, Michigan, U.S.A, July 2006, pp. 427–432.

- [104] C. Baier and T. Moor, “A hierarchical and modular control architecture for sequential behaviours,” *Discrete Event Dynamic Systems : Theory and Applications*, vol. 25, no. 1, pp. 95–124, 2014.
- [105] P. E. Caines and Y. J. Wei, “The hierarchical lattices of a finite machine,” *Systems & Control Letters*, vol. 25, no. 4, pp. 257–263, July 1995.
- [106] P. E. Caines, V. Gupta, and G. Shen, “The hierarchical control of st-finite-state machines,” *Systems & Control Letters*, vol. 32, no. 4, pp. 185–192, 1997.
- [107] P. E. Caines, P. Hubbard, and G. Shen, “State aggregation and hierarchical supervisory control,” in *Proceedings of the 36th IEEE Conference on Decision and Control*, vol. 4, San Diego, CA, U.S.A, December 1997, pp. 3590–3591.
- [108] P. Hubbard and P. E. Caines, “Dynamical consistency in hierarchical supervisory control,” *IEEE Transactions on Automatic Control*, vol. 47, no. 1, pp. 37–52, 2002.
- [109] M. S. Shaikh and P. E. Caines, “On the optimal control of hybrid systems: Optimization of trajectories, switching times, and location schedules,” in *Hybrid systems: Computation and control*. Springer, 2003, pp. 466–481.
- [110] M. K. Helwa and P. E. Caines, “Hierarchical control of piecewise affine hybrid systems,” in *Proceedings of the 53rd IEEE International Conference on Decision and Control*, Los Angeles, CA, U.S.A, December 2014, pp. 3950–3956.
- [111] G. Shen and P. E. Caines, “Control consistency and hierarchically accelerated dynamic programming,” in *Proceedings of the 37th IEEE Conference on Decision and Control*, vol. 2, Tampa, FL, U.S.A, December 1998, pp. 1686–1691.
- [112] —, “Hierarchically accelerated dynamic programming for finite-state machines,” *IEEE Transactions on Automatic Control*, vol. 47, no. 2, pp. 271–283, 2002.

- [113] C. R. Torrico and J. E. R. Cury, “Hierarchical supervisory control of discrete event systems based on state aggregation,” in *Proceedings of the 15th IFAC World Congress*, vol. 15, Barcelona, Spain, 2002.
- [114] J. E. R. Cury, C. R. Torrico, and A. E. C. da Cunha, “A new approach for supervisory control of discrete event systems,” in *Proceedings of the European Control Conference*, vol. 1, Porto, Portugal, September 2001, pp. 1595–1599.
- [115] D. Harel, “Statecharts: A visual formulation for complex systems,” *Science of Computer Programming*, vol. 8, no. 3, pp. 231–274, June 1987.
- [116] Y. Brave and M. Heymann, “Control of discrete event systems modeled as hierarchical state machines,” *IEEE Transactions on Automatic Control*, vol. 38, no. 12, pp. 1803–1819, December 1993.
- [117] H. Marchand and B. Gaudin, “Supervisory control problems of hierarchical finite state machines,” in *Proceedings of the 41st IEEE Conference on Decision and Control*, vol. 2, Las Vegas, U.S.A, 2002, pp. 1199–1204.
- [118] B. Gaudin and H. Marchand, “Supervisory control of product and hierarchical discrete event systems,” *European Journal of Control*, vol. 10, no. 2, pp. 131–145, 2004.
- [119] B. Wang, “Top-down design for RW supervisory control theory,” Graduate Department of Electrical and Computer Engineering, University of Toronto, Canada, Master of Applied Science (MASc) Thesis, 1995.
- [120] C. Ma and W. M. Wonham, “Control of state tree structures,” in *Proceedings of the 11th Mediterranean Conference on Control and Automation*, Rhodes, Greece, June 2003, p. 6.
- [121] S. Akers, “Binary decision diagrams,” *IEEE Transactions on Computers*, vol. 100, no. 6, pp. 509–516, 1978.

- [122] K. Cai and W. M. Wonham, “Supervisor localization of discrete-event systems based on state tree structures,” *IEEE Transactions on Automatic Control*, vol. 59, no. 5, pp. 1329–1335, May 2014.
- [123] P. Gohari-Moghadam and W. M. Wonham, “A linguistic framework for controlled hierarchical DES,” in *Proceedings of the 4th International Workshop on Discrete-Event Systems*. Cagliari, Italy: IEE Computing and Control Division, August 1998, pp. 207–212.
- [124] J. Komenda, T. Masopust, and J. H. van Schuppen, “Multilevel coordination control of modular DES,” in *Proceedings of the 52nd IEEE Conference on Decision and Control*, Florence, Italy, December 2013, pp. 6323–6328.
- [125] —, “Multilevel coordination control of partially observed modular DES,” in *Proceedings of the American Control Conference*, Chicago, IL, U.S.A, 2015, pp. 384–389.
- [126] J. Komenda and T. Masopust, “Decentralized supervisory control with communicating supervisors based on top-down coordination control,” in *Proceedings of the IEEE 53rd Annual Conference on Decision and Control*, Los Angeles, CA, U.S.A, December 2014, pp. 5149–5155.
- [127] R. J. Leduc, B. A. Brandin, M. Lawford, and W. M. Wonham, “Hierarchical interface-based supervisory control - Part I: Serial case,” *IEEE Transactions on Automatic Control*, vol. 50, no. 9, pp. 1322–1334, September 2005.
- [128] R. J. Leduc, M. Lawford, and W. M. Wonham, “Hierarchical interface-based supervisory control-Part II: Parallel case,” *IEEE Transactions on Automatic Control*, vol. 50, no. 9, pp. 1336–1348, 2005.
- [129] R. J. Leduc, P. Dai, and R. Song, “Synthesis method for hierarchical interface-based supervisory control,” *IEEE Transactions on Automatic Control*, vol. 54, no. 7, pp. 1548–1560, July 2009.

- [130] R. Malik and R. J. Leduc, "Hierarchical interface-based supervisory control using the conflict preorder," in *Proceedings of the 11th International Workshop on Discrete-Event Systems*, Guadalajara, Mexico, October 2012, pp. 163–168.
- [131] R. J. Leduc, "Hierarchical interface-based supervisory control," Graduate Department of Electrical and Computer Engineering, University of Toronto, Canada, Doctor of Philosophy (Ph.D) Thesis, 2002.
- [132] K. C. Wong, "On the complexity of projections of discrete-event systems," in *Proceedings of the 4th International Workshop on Discrete-Event Systems*. Cagliari, Italy: IEE Computing and Control Division, August 1998, pp. 201–206.
- [133] K. C. Wong and W. M. Wonham, "On the computation of observers in discrete-event systems," *Discrete Event Dynamic Systems : Theory and Applications*, vol. 14, no. 1, pp. 55–107, January 2004.
- [134] Y. Brave and M. Heymann, "Formulation and control of real time discrete event processes," in *Proceedings of the 27th IEEE International Conference on Decision and Control*, Austin, Texas, U.S.A, December 1988, pp. 1131–1132.
- [135] C. G. Cassandras, *Discrete Event Systems : Modeling and Performance Analysis*. Richard D. Irwin, Inc., and Aksen Associates, Inc., 1993.
- [136] D. D. Cofer and V. K. Garg, "Supervisory control of real-time discrete-event systems using lattice theory," *IEEE Transactions on Automatic Control*, vol. 41, no. 2, pp. 199–209, February 1996.
- [137] S. L. Knap, "Modelling and Control of Timed Discrete-Event Systems and its Applications to Scheduling," Department of Electrical and Computer Engineering, Queen's University at Kingston Kingston, Ontario, Canada, Doctor of Philosophy (Ph.D) Thesis, January 2001.

- [138] P. Gohari and W. M. Wonham, “Reduced supervisors for timed discrete-event systems,” *IEEE Transactions on Automatic Control*, vol. 48, no. 7, pp. 1187–1198, July 2003.
- [139] A. Saadatpoor and W. M. Wonham, “State based control of timed discrete event systems using binary decision diagrams,” *Systems & Control Letters*, vol. 56, no. 1, pp. 62–74, 2007.
- [140] K. Cai, R. Zhang, and W. M. Wonham, “On relative observability of timed discrete-event systems,” in *Proceedings of the 12th International Workshop on Discrete-Event Systems*, Cachan, France, May 2014, pp. 208–213.
- [141] S.-J. Park and K.-H. Cho, “Nonblocking supervisory control of timed discrete event systems under communication delays: The existence conditions,” *Automatica*, vol. 44, no. 4, pp. 1011–1019, April 2008.
- [142] A. Dhananjayan and K. T. Seow, “A formal transparency framework for validation of real-time discrete-event control requirements modeled by timed transition graphs,” *IEEE Transactions on Human-Machine Systems*, vol. 45, no. 3, pp. 350–361, June 2015.
- [143] —, “A metric temporal logic specification interface for real-time discrete-event control,” *IEEE Transactions on Systems, Man and Cybernetics: Systems*, vol. 44, no. 9, pp. 1204–1215, September 2014.
- [144] M. Nomura and S. Takai, “Decentralized supervisory control of timed discrete-event systems,” *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. 94, no. 12, pp. 2802–2809, 2011.
- [145] —, “A synthesis method for decentralized supervisors for timed discrete-event systems,” *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. 96, no. 4, pp. 835–839, 2013.

- [146] W. H. Sadid, S. Hashtrudi-Zad, and S. L. Ricker, “Decentralized control of timed discrete-event systems under bounded delay communication,” in *Proceedings of the IEEE Conference on Control Applications*, Juan Les Antibes, France, October 2014, pp. 1795–1800.
- [147] T. J. Ho, “A method for the modular synthesis of controllers for timed discrete-event systems,” *International Journal of Control*, vol. 76, no. 5, pp. 520–535, 2003.
- [148] G. Schafaschek, M. H. de Queiroz, and J. E. R. Cury, “Local modular supervisory control of timed discrete-event systems,” in *Proceedings of the 12th International Workshop on Discrete Event Systems*, vol. 12, Cachan, France, 2014, pp. 271–277.
- [149] R. Zhang, K. Cai, Y. Gan, Z. Wang, and W. M. Wonham, “Supervision localization of timed discrete-event systems,” *Automatica*, vol. 49, no. 9, pp. 2786–2794, 2013.
- [150] R. Zhang, K. Cai, and W. M. Wonham, “Delay-robustness in distributed control of timed discrete-event systems based on supervisor localization,” in *Proceedings of the 53rd IEEE International Conference on Decision and Control*, Los Angeles, CA, U.S.A, December 2014, pp. 6719 – 6724.
- [151] A. Saadatpoor, “Timed state tree structures: Supervisory control and fault diagnosis,” Graduate Department of Electrical and Computer Engineering, University of Toronto, Canada, Doctor of Philosophy (Ph.D) Thesis, 2009.
- [152] A. Saadatpoor, C. Ma, and W. M. Wonham, “Supervisory control of timed state tree structures,” in *Proceedings of the American Control Conference*, Seattle, Washington, U.S.A, June 2008, pp. 477–482.
- [153] C. Ma and W. M. Wonham, *Nonblocking Supervisory Control of State Tree Structures. Lecture Notes in Control and Information Sciences, Vol 317*. Springer-Verlag, New York, 2005.

- [154] H. Zhong and W. M. Wonham, “Hierarchical control of discrete-event systems: Computation and examples,” in *Proceedings of the 27th Annual Allerton Conference on Communication, Control, and Computing*, University of Illinois, Champaign-Urbana, Illinois, U.S.A, September 1989, pp. 511–519.
- [155] Z. Fei, S. Miremadi, K. Åkesson, and B. Lennartson, “Efficient symbolic supervisor synthesis for extended finite automata,” *IEEE Transactions on Control Systems Technology*, vol. 22, no. 6, pp. 2368–2375, June 2014.
- [156] S. Miremadi, Z. Fei, K. Åkesson, and B. Lennartson, “Symbolic supervisory control of timed discrete event systems,” *IEEE Transactions on Control Systems Technology*, vol. 23, no. 2, pp. 584–597, February 2015.
- [157] Q. H. Ngo and K. T. Seow, “Command and control of discrete-event systems: Towards on-line hierarchical control based on feasible system decomposition,” *IEEE Transactions on Automation Science and Engineering*, vol. 11, no. 4, pp. 1218–1228, October 2014.