2010

Control and diagnosis of real-time systems under finite-precision measurement of time

Songyan Xu
Iowa State University

Control and diagnosis of real-time systems under finite-precision measurement of time

by

Songyan Xu

A dissertation submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Major: Electrical Engineering

Program of Study Committee:
Ratnesh Kumar, Major Professor
Nicola Elia
Manimaran Govindarasu
Samik Basu
Pavan Aduri

Iowa State University

Ames, Iowa

2010

DEDICATION

I would like to dedicate this dissertation to my dear parents Yajun Wu and Guoshun Xu, and my sister Songyang Xu. Without their endless love, concern, understanding, encouragement and support, I would not have been able to complete this work.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

A discrete event system (DES) is an event-driven system that evolves according to abrupt occurrences of discrete changes (events). The domain of such systems encompasses aspects of many man-made systems such as manufacturing systems, telephone networks, communication protocols, traffic systems, embedded software, asynchronous hardware, robotics, etc.

Supervisory control theory for DESs studies the existence and synthesis of the supervisory controllers, namely, supervisors that restrict the system behaviors by dynamically disabling certain controllable events so that the controlled close-loop system could behave as desired. Extensive work on supervisory control of untimed DESs exists and the extension to the timed setting has been reported in the literature. In this dissertation, we study the supervisory control of dense-time DESs in which the digital-clocks of finite-precision are employed to observe the event occurrence times, thereby relaxing the assumption of the prior works that time can be measured precisely. In our setting, the passing of time is measured using the number of ticks generated by a digital-clock and we allow the plant events and digital-clock ticks to occur concurrently. We formalize the notion of a control policy that issues the control actions based on the observations of events and their occurrence times as measured using a digital-clock, and show that such a control policy can be equivalently represented as a "digitalized"-automaton, namely, an untimed-automaton that evolves over the events (of the plant) and ticks (of the digital-clock). We introduce the notion of observability with respect to the partial observations of time resulting from the use of a digital-clock, and show that this property together with controllability serves as a necessary and sufficient condition for the existence of a supervisor to enforce a real-time specification on a dense-time discrete event plant. The observability condition presented in the dissertation is very different from the one arising due to a partial

observation of events since a partial observation of time is in general nondeterministic (the number of ticks generated in any time interval can vary from execution to execution of a digital-clock). We also present a method to verify the proposed observability and controllability conditions, and an algorithm to compute a supervisor when such conditions are satisfied. Furthermore we examine the lattice structure of a class of timing-mask observable languages, and show that the proposed observability is not preserved under intersection but preserved under union.

Fault diagnosis for DESs is to detect the occurrence of a fault so as to enable any corrective actions. It is crucial in automatic control of large complex man-made systems and has attracted considerable attention in the literature of reliability engineering, control and computer science. For the event-driven systems with timing-requirements such as manufacturing systems, communication networks, real-time scheduling and traffic systems, fault diagnosis involves detecting the timing-faults, besides the sequence-faults. This requires monitoring timing and sequence of events, both of which may only be partially observed in practice. In this dissertation, we extend the prior works on fault diagnosis of timed DESs by allowing time to be partially observed using a digital-clock which measures the advancement of time with finite precision by the number of ticks. For the diagnosis purposes, the set of nonfaulty timed-traces is specified as another timed-automaton that is deterministic. We show that the set of timed-traces observed using a digital-clock with finite precision is regular, i.e., can be represented using a finite (untimed) automaton. We also show that the verification of diagnosability (the ability to detect the execution of a faulty timed-trace within a bounded time delay) as well as the off-line synthesis of a diagnoser are decidable by reducing these problems to the untimed setting. The reduction to the untimed setting also suggests an effective method for the off-line computation of a diagnoser as well as its on-line implementation for diagnosis. The aforementioned results are further extended to the nondeterministic setting, i.e., diagnosis of dense-time DESs using digital-clocks under nondeterministic event observation mask. We introduce the notion of "lifting" (associating each event with each of its nondeterministic observations), and show that diagnosis of dense-time DESs employing digital-clocks to observe event occurrence

times under nondeterministic event observation mask can be reduced to that of the deterministic setting, i.e., diagnosis of the lifted dense-time DESs under the deterministic lifted event observation mask, and hence can be further reduced to diagnosis of the untimed setting.

CHAPTER 1. INTRODUCTION

1.1 Discrete Event Systems: Untimed and Timed

A discrete event system (DES) is an event-driven system which evolves according to abrupt occurrences of discrete changes (events). The examples of the discrete events include completion of a transaction in a database system, transmission of a message in a communication network, occurrence of a failure in a manufacture system, etc. The domain of discrete event systems encompasses aspects of many man-made systems such as manufacturing systems, telephone networks, communication protocols, traffic systems, embedded software, asynchronous hardware, robotics, etc [66].

Discrete event systems can be classified into two types: untimed and timed DESs. In the case of untimed DESs, the system behavior is only determined by the sequence of the events executed, whereas in the case of timed DESs, the system behavior is determined by the logic order of the events as well as the times at which the events occur. An untimed model is used when the system under study needs to meet the qualitative goals such as orderly occurrence of events, whereas a timed model is used when the quantitative goals such as a response rate in a traffic system and an average delay in a communication network need be satisfied.

In contrast to the continuous systems in which the system state changes continuously with the evolvement of time, the state transition mechanism of DESs is event-driven: state transitions occur at asynchronous discrete time instants instantaneously in response to events, and in between event occurrences, the system state remains unchanged [19]. Unlike the continuous control systems, the state space of a DES consists of a set of discrete states. Further, the states as well as the state transitions of DESs may take symbolic values rather than real values. For instance, the state of a machine can be *on* or *off*, and a machine can change its state from *on*

to *off* upon the occurrence of an event *shutting down*. Owing to these features, discrete event systems cannot be adequately described using differential or difference equations. Instead, formal language and state machine are used to describe the behavior of DESs at the logic level of abstraction. A language is a collection of the sequences of events that can be executed. A state machine is an alternative way to represent a language model. Petri net and automaton are the two state machine modeling formalisms for DESs that are widely used in the literature. In this dissertation, we choose automaton as the modeling formalism for DESs.

1.2 Control of Untimed and Timed DESs

Supervisory control theory for DESs was initiated in the work reported in [106]. It studies the existence and synthesis of supervisory controller, also called supervisor that restricts the system behaviors by dynamically disabling certain controllable events as a function of the observed event-trace so that the controlled system behaves as desired.

Extensive work on supervisory control of untimed systems/specifications exists. See for example centralized supervisory control under complete or partial observations in which the specification is expressed by a single language in [106, 107, 137, 148, 80, 75, 22, 68, 117, 98, 152], and the extension to modular supervisory control in which multiple modular supervisors are combined to achieve the desired system behavior expressed by the intersection of two or more elementary specification languages in [134, 135, 30, 105, 20, 109, 36, 61]. Since a real control system is usually complex and physically distributed, it is impractical to design a centralized supervisor for such real applications. Therefore a decentralized/distributed control architecture need be employed. Similar to the modular supervisory control, multiple supervisors are synthesized for decentralized/distributed supervisory control. In the case of decentralized control, the local supervisors do not communicate with each other, whereas in the case of distributed control, the local supervisors can exchange their local observations for the purpose of control. Unlike the modular supervisory control, each individual supervisor of decentralized/distributed supervisory control has its own set of controllable events, observation mask and a desired controlled behavior to achieve. The work on the decentralized/distributed super-

visory control can be found in [79, 81, 112, 133, 131, 132, 111, 48, 50, 11, 147, 145, 103, 154]. In [9, 10, 28, 122, 90, 90, 92], the issues of control arising due to bounded communication delays were studied. Supervisory control of DESs with special purposes has been addressed in the following works. See for example the work on robust control in [78, 1, 24, 113], on adaptive control in [25, 35], on optimal control in [96, 95, 76, 65, 67, 115, 18, 8, 85, 62], on fault-tolerant control in [1, 94, 21, 45, 110, 34, 89, 129, 128, 130], and on directed control in [43, 42]. The above works focus on control of deterministic discrete event systems, i.e., the systems in which given a state and an event that occurs at the state, the destination state reached following the occurrence of the event can be uniquely determined. However, in certain systems, due to the unmodeled system dynamics and/or partial observations, the current state and ongoing event are not sufficient to uniquely determine its next state. For instance, while a manufacturing system is performing a task, an undetectable failure may occur. This can be modeled by a nondeterministic transition which leads to two successor states based on whether a failure occurred. The studies on supervisory control of the nondeterministic DESs can be found in [83, 116, 69, 70, 69, 70, 72, 39, 40, 49, 99, 52, 150, 151]. And control in the temporal logic setting has been examined in [77, 56, 12, 118, 13].

The aforementioned works studied control of untimed DESs. The extensions to the timed setting have been reported in the works as follows. In [17, 16, 82] a discretized model of time was used: the time advances when a "tick" transition of the timer occurs. Such a discretized-time model is not as general as a dense-time model [4, Section 2.3.2], which requires the continuous time to be approximated by a priori fixing the smallest measurable time unit. Such approximation limits the accuracy of the modeling for a physical system. Also the inclusion of clock tick transitions in timed DESs results in the state-space explosion problem. To avoid state-space explosion, the author of [91] proposed the notion of eligible time bound and showed the controllability and observability conditions for the existence of a supervisor. Moreover the possibility of preempting a "tick" transition by a *forcible* event as in [16, 82] is an artifact arising due to the discretized model of time, and thus is unrealistic. Another approach for control of timed DESs is based on a dense-time model: the event occurrence

times are represented by real numbers and increase monotonically without bound. A dense-time model is a more natural model for physical systems that operate over continuous time. The examples of modelings of dense-time DESs include timed automaton proposed by Alur and Dill and timed Petri nets proposed by Leveson and Stoley. The first use of dense-time model (timed automaton) for supervisory control of dense-time DESs was proposed in [136], where the authors introduced the notion of controllability in the timed setting as a condition for the existence of a supervisor, and also presented a method to verify the controllability condition by reducing to reachability over an untimed region-automaton [6]. The supervisory control using dense-time model was later studied in [37], in which the timed automaton is completely discretized into region automaton, and then the discrete synthesis problem is solved. Certain techniques for efficient representations of region-automaton have been presented in the literature. For instance, in [59] two special types of events Set and Exp, which represent setting and expiring of clocks respectively, are used to transform a timed automaton into a small-sized equivalent finite state automaton, namely Set-Exp-Automaton (SEA). The real-time control problem is then solved by adapting the non-real-time control method. Since the state of a SEA changes with the passing of time only when a clock reaches a value it is compared to in a timing constraint of the corresponding timed automaton, in practice the size of SEA does not increase significantly with the magnitudes of the constants used in timing constraints. The extension to real-time control under partial observation was reported in [58]. In [86], the timed supervisory control method based on sampling region graphs of the plant and specifications, and constructing a discrete finite subautomaton, called grid automaton was presented. The aforementioned techniques can be used to expedite the controllability test of [136]. In [7, 84, 125], symbolic methods for controller synthesis in the setting of game timed-automata have been proposed. The game timed-automaton approach is different from Ramadge and Wonham framework: supervisory control is considered as a game between a supervisor with a desired specification and a plant with the goal to drive the controlled plant to a set of safe states. Moreover, [71] employed prioritized synchronization to exercise control in the dense-time setting. Recently, an extension of the supervisory control to a class of timed

DESs modeled by (max,+) automata, which can be viewed as special timed automata, is proposed in [60, 93].

The above cited works on control of dense-time systems *assume* that the event occurrence times can be measured precisely. Such assumption on infinite-precision measurement of time can greatly simplify the design and analysis of systems. However, measuring time with arbitrary precision is not practically possible. In our work, we relax the assumption that time can be measured with arbitrary precision by letting a supervisor employ a digital-clock to measure event occurrence times in form of the number of ticks generated by the digital-clock. (See also [3] where finite-precision clocks are investigated.) Under finite-precision measurement of time, correctness of control may get lost. The application of digital-clocks to verification and testing of real-time systems modeled by timed automaton can be found in the works such as [38, 87, 74, 64, 63]. And the research on semantics of control that can be implemented with digital-clocks of finite-precision can be found in [139, 138].

The following example illustrates that owing to the partial observability of time resulting from the use of a digital-clock, controllability condition of [136] alone is not sufficient for the existence of a supervisor.

Example 1 Consider a plant G and a specification language K as shown in Figure 1.1. Suppose all events are controllable, so that any specification language including K is controllable. G can execute a earlier than 2 units of time, followed by which it can execute b. The specification requires that b be disabled if a occurs at or before 1.5 units of time. Under a precise observation of time, this can be achieved by observing the occurrence time of a and disabling b accordingly. Suppose time is measured by a digital-clock which ticks every $\Delta \pm \delta$ units of time, where $\Delta = 1$ and $\delta = 0.1$. Then the timed-trace $(a, 1.1 < t_1 \leq 1.5)$ is indistinguishable from the timed-trace $(a, 1.5 < t_2 < 1.8)$, for both are observed as "tick.a". Since b must be disabled following (a, t_1) whereas it must be enabled following (a, t_2), no supervisor for enforcing the desired specification exists when time is measured using the digital-clock described above.

The above example serves to motivate the study of supervisory control of dense-time discrete event systems using finite-precision digital-clocks for measuring time. We start by for-

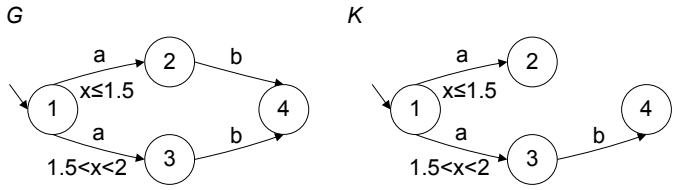

Figure 1.1 Timed-automata of plant G (left) and specification K (right)

malizing the notion of a control policy that employs observations of events and their occurrence times as measured using a digital-clock for computing the control actions, and show that it can be equivalently represented as a "digitalized"-automaton, namely, an untimed-automaton that evolves over events (of the plant) and ticks (of the digital-clock). We introduce the notion of *observability with respect to the partial observation of time* resulting from the use of a digital-clock, and show that this property together with controllability serves as a necessary and sufficient condition for the existence of a control policy to enforce a real-time specification on a dense-time discrete event plant. The observability condition presented in the paper is very different from the one arising due to a partial observation of events since a partial observation of time is in general nondeterministic (the number of ticks generated in any time interval can vary from execution to execution of a digital-clock). We present a method to check the proposed observability and controllability conditions, and also present a method to synthesize a supervisor when these conditions are satisfied.

1.3 Diagnosis of Untimed and Timed DESs

Fault diagnosis is crucial for large complex control systems such as manufacturing systems, computer and communication networks, power systems, circuits, traffic systems, heating, ventilation and air conditioning systems, etc. It has attracted considerable attention from the literature of reliability engineering, computer science and control. The task of diagnosis is to detect the occurrence of faults from the observations of the system behavior so as to enable any corrective actions.

Diagnosis of untimed DESs was first studied in [114], in which the notion of diagnosability, i.e., the property that requires the occurrence of a failure to be detected within bounded time

delay, was proposed. And an exponential algorithm to check diagnosability was also proposed in [114]. The polynomial tests for checking diagnosability were given in [46, 146]. A stronger notion of state-observability has been examined in [88]. Moreover, diagnosis in the probabilistic setting (stochastic DESs) can be found in [120], and in the Petri net setting in [33, 32]. These cited works studied centralized diagnosis: a centralized diagnoser is responsible for fault/failure detection. However many large complex systems are physically distributed, which results in variable communication delays and communication errors. Therefore centralized diagnosis is not appropriate for these systems. The architecture in which diagnosis can be preformed at each local site is needed. Diagnosis of discrete-event systems in the decentralized setting has been reported in [27, 101, 127, 119, 73], and in the distributed setting can be seen in [108, 14, 102, 104]. Decentralized diagnosis can be viewed as a special case of distributed diagnosis. In decentralized diagnosis, the local diagnosers do not communicate with each other, whereas in distributed diagnosis either diagnosers communicate with each other directly or through a coordinator which is responsible for fusing local observations and making the final diagnosis decision. In [29], the assumption of zero communication delay made in [114] was *mildly* relaxed by allowing at most one-step communication delay. Decentralized/distributed diagnosis under bounded communication delay was further studied in [124, 144, 100, 104, 102]. Diagnosis of repeatable/intermittent-failures was investigated in [57, 144, 55, 23, 153]. The notion of diagnosability was extended to $[1, \infty]$-diagnosability to allow diagnosis of a failure each time it occurred. To facilitate generalization of failure specifications, diagnosis in the temporal logic setting was studied in [53, 55]. And decentralized diagnosis using inference-based ambiguity management has been reported in [73, 119].

The above cited works study diagnosis of untimed discrete event systems. However most applications of discrete event systems (such as manufacturing systems, communication network, real-time scheduling, traffic systems, etc.) possess timing properties (such as latency). Correctness of these real-time applications depend on not only the correctness of the sequence of events executed, but also the correctness of the event occurrence times. For instance, a controller of a manufacturing system may be required to issue a correct control within an al-

lowable time interval; the messages transmitted in a communication network may be required to be delivered within an acceptable delay. Diagnosis of such event-driven systems with timing-requirements involves detecting the timing-faults, besides the sequence-faults. This requires monitoring both timing and sequence of events. Therefore the priori works on diagnosis of untimed discrete event system are inadequate in monitoring or diagnosing timing-faults that arise in real-time applications. There has been some research on diagnosis of timed discrete event systems, which includes diagnosis in the discrete-time setting [149] and in the dense-time setting [41, 26, 123, 15]. It is known that the class of discrete-time systems is a subclass of dense-time ones as modeled by timed-automata [5]. Failure diagnosis of dense-time models was first examined in [123]. It was assumed that while a diagnoser has partial observation of events, it is able to measure time perfectly. It was shown that the verification of diagnosability in this setting is decidable and on-line diagnosis can be effectively performed, whereas no comments were made about the off-line synthesis of a diagnoser. The failure diagnosis of timed-automata under partial observation of events and perfect observation of time was also studied in [15]. The main focus was on the synthesis of diagnosers which are realizable as deterministic timed-automata. A procedure for constructing a diagnoser off-line for a subclass of dense-time automaton was proposed in [31].

These earlier works on failure diagnosis of timed DESs assume that a diagnoser has partial observation of events but can measure (or observe) time precisely. In practice, however, time can only be measured with finite precision. The following example illustrates that a system, that is diagnosable under the assumption that time is measured precisely, may become undiagnosable when time can only be measured with certain finite precision.

Example 2 Consider the timed automaton model G shown in Figure 1.2, in which f is a faulty event and unobservable, and u is a nonfaulty event and unobservable. The faulty event f can occur after 0.5 units of time in G, whereas the nonfaulty event u occurs before 0.5 units of time. Following the occurrence of the faulty event, G can execute observable event a at 1.6 units of time. whereas following the nonfaulty event, event a can be executed at 1.6 units of time. It can be checked that this system is diagnosable if time could be measured with

arbitrary precision based on the occurrence time of event a: a (resp., no) fault has occurred if a can be observed at 1.6 (resp., 1.2) units of time. Suppose time could be measured with only a finite precision, say using a digital clock that ticks every one unit of time. Then G is no longer diagnosable. This is because a faulty trace $(f, 1.1)(a, 1.6)$ cannot be distinguished from a nonfaulty trace $(u, 0.1)(a, 1.2)$, both of which produce the same observation, namely, "tick" followed by a.

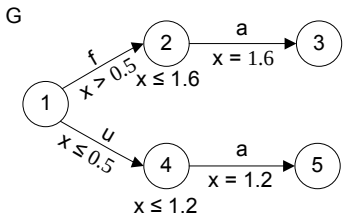

Figure 1.2 Timed automaton model of a discrete event system

This motivates us to study the diagnosis problem of dense-time discrete event systems in which digital-clocks with finite precision are used to measure the event occurrence times. In our work, we model the finite precision observability of time using a digital-clock that measures time discretely by executing ticks, the logic of which is governed by a timed-automaton. We show that the set of timed-traces of a dense timed-automaton observed using a finite precision digital-clock is regular, i.e., can be represented using a finite (untimed) automaton. We show that the verification of diagnosability (ability to detect the execution of a faulty timed-trace within a bounded time delay) as well as the off-line synthesis of a diagnoser are decidable by reducing these problems to the untimed setting. The reduction to the untimed setting also suggests an effective method for the off-line computation of a diagnoser as well as its on-line implementation for diagnosis. The aforementioned results are further extended to the nondeterministic setting, i.e., the diagnosis of dense-time DESs using digital-clocks under nondeterministic event observation mask. We introduce the notion of *lifting*, (associating each event with each of its nondeterministic observations) and show that diagnosis of dense-time DESs in the nondeterministic setting can be reduced to that of "lifted" dense-time DESs under deterministic event observation mask, and hence can be further reduced to that of lifted

untimed DESs in the deterministic setting.

The same problem was also independently studied in [2]. The authors of [2] additionally studied the existence of a digital-clock that ensures the diagnosability of a dense-time system, whereas we additionally study the diagnosis problem where failure is specified more generally, namely as the violation of a real-time specification language (in contrast, in [2] failure was simply modeled as the execution of a faulty transition). We also further study the diagnosis problem in the nondeterministic setting, i.e., diagnosis of dense-time DESs using digital-clocks under nondeterministic event observation mask. The other differences are as below: (i) We explicitly define the notion of a timing-mask of a timed-trace (namely its observation when time is measured using a digital-clock). It captures nondeterminism of the untimed observations of a timed trace as observed using a digital-clock explicitly. Whereas no such explicit definition is given in [2]. (ii) We point out that the set of behaviors of a dense-time discrete event system observed employing a digital-clock for the measurement of time is not prefix-closed. This is owing to the fact that plant events and digital-clock ticks can occur simultaneously. The non-prefix-closure of the set of observed behaviors was not noticed in [2]. The construction of an automaton, that accepts the event/timing-masked language of a plant (namely the set of observations of the plant timed-traces when events are partially observed and time is measured using a digital-clock) given in [2, Proposition 3] would require additional marking and refinement. (iii) We establish an equivalence between diagnosability of a timed DES employing a digital clock to observe event occurrence times and that of an untimed DES. Therefore the diagnosis problem in the dense-time setting can be solved by its reduction to the untimed setting. In particular this suggests an algorithm to construct a diagnoser when the given system is diagnosable. In contrast, how to construct a diagnoser when a given system is diagnosable is not described in [2] (whether the necessary and sufficient condition of [2, Proposition 7] is also enough to construct a diagnoser is not discussed and hence open).

1.4 Organization of Dissertation

The dissertation is organized as follows.

In Chapter 2, we present the notations and preliminaries which are necessary for the dissertation.

In Chapter 3, we define the behavior of a dense timed-automaton when the event occurrence times are measured using a digital-clock of finite precision that measures time discretely by generating ticks. We show that the set of timed-traces in which the event occurrence times are measured through a digital-clock constitutes a regular language, i.e., can be represented by a finite (untimed) automaton.

In Chapter 4, we study the supervisory control of dense-time discrete event systems using finite-precision digital-clocks to observer event occurrence times. We start by introducing the notion of *control compatible* and *timing-mask compatible* control policies that do not disable uncontrollable events and also respect the timing-mask associated with a digital-clock, and show that a compatible control policy can be represented as a "digitalized"-automaton (an untimed-automaton in which the passing of time occurs discretely in form of the occurrences of ticks). We then introduce the notion of *observability with respect to the partial observation of time* resulting from the use of a digital-clock, and show that this property together with controllability serves as a necessary and sufficient condition for the existence of a control policy to enforce a real-time specification on a dense-time discrete event plant. The observability condition presented in the paper is very different from the one arising due to a partial observation of events since a partial observation of time is in general nondeterministic (the number of ticks generated in any time interval can vary from execution to execution of a digital-clock). Finally we present a method to check the proposed observability and controllability conditions, and also present a method to synthesize a supervisor when these conditions are satisfied. We further examine the lattice structure of a class of timing-mask observable languages, and show that timing-mask observability is not preserved under intersection but preserved under union.

In Chapter 5, we study the diagnosis of dense-time discrete event systems using finite-precision digital-clocks to observer event occurrence times. Two diagnosis problems are investigated: (i) diagnosis of timed discrete event system modeled by timed-automaton with both timing and event observation masks, and (ii) diagnosis with dense-time specification which

specifies the nonfailure behavior. We show that the verification of diagnosability (ability to detect the execution of a faulty timed-trace within a bounded time delay) as well as the off-line synthesis of a diagnoser are decidable by reducing these problems to the untimed setting. The reduction of the diagnosis problem to the untimed setting also suggests an effective method for the off-line computation of a diagnoser as well as its on-line implementation for diagnosis. The aforementioned results are further extended to the nondeterministic setting, i.e., diagnosis of dense-time DESs using digital-clocks under nondeterministic event observation mask. We introduce the notion of lifting and show that diagnosis of dense-time DESs in the nondeterministic setting can be reduced to diagnosis of lifted dense-time DESs under deterministic event observation mask, and hence can be further reduced to diagnosis of lifted untimed DESs in the deterministic setting.

In Chapter 6, we summarize the work and conclude with the discussions of future work. We also summarize the other works that have done during the Phd studies of the author.

CHAPTER 2. NOTATION AND PRELIMINARIES

In this chapter, we present the notations and preliminaries which are necessary for the dissertation: the notions of untimed and timed-language, untimed and timed-automaton, non-speedingness, non-slowingness, etc. The interested readers are recommended to refer to [66, 19, 4] for more details on untimed/timed DESs and untimed/timed automaton.

2.1 Untimed Language and Automaton

Let Σ denote a set of events. A trace over Σ is a sequence $s = \sigma_1 \cdots \sigma_n$ where $\sigma_i \in \Sigma$ for $i = 1, \cdots, n$. We use Σ^* to denote the set of all event-traces over Σ, including the trace of zero-length ϵ. A subset $L \subseteq \Sigma^*$ is called a language over Σ. Given a trace s, $|s|$ is used to denote the length of s. A trace $s \in \Sigma^*$ is said to be a *prefix* of a trace $t \in \Sigma^*$ if for some trace $u \in \Sigma^*$, $t = su$. This is denoted by $s \leq t$. If $s \leq t$ and $|s| < |t|$, then s is said to be a *proper prefix* of t. The *prefix-closure* of $L \subseteq \Sigma^*$, denoted $pr(L)$, is the set of all prefixes of the traces in L. L is called *prefix-closed* or simply *closed* if $pr(L) = L$. Given a language K, K is said to be *relative closed* (with respect to a language L) if $pr(K) \cap L = K$.

Given $\widehat{\Sigma} \subseteq \Sigma$, the projection of an untimed-trace over $\widehat{\Sigma}$ is denoted by $\Pi_{\widehat{\Sigma}}(\cdot)$, which is inductively defined as

$$\Pi_{\widehat{\Sigma}}(\epsilon) := \epsilon,$$

$$\Pi_{\widehat{\Sigma}}(s\sigma) := \begin{cases} \Pi_{\widehat{\Sigma}}(s)\sigma & \text{if } \sigma \in \widehat{\Sigma} \\ \Pi_{\widehat{\Sigma}}(s) & \text{otherwise} \end{cases}$$

where $s \in \Sigma^*$, and $\sigma \in \Sigma$.

It can be concluded that $\Pi_{\widehat{\Sigma}}(s) \in \Sigma^*$.

For control purpose, the event set Σ is partitioned into the set of uncontrollable as well as controllable events: $\Sigma = \Sigma_u \cup (\Sigma - \Sigma_u)$. Given a language K, K is said to *controllable* (with respect to a language L and a set of uncontrollable events Σ_u) if

$$\forall s \in pr(K), \sigma \in \Sigma_u \; s.t. \; s\sigma \in L \;\Rightarrow\; s\sigma \in pr(K).$$

The property of controllability requires that the execution of an uncontrollable event σ following a trace s in $pr(K)$ must either result in a trace in $pr(K)$, or a violation of the language L, i.e, resulting in a trace in $\Sigma^* - L$.

In many real applications, it is difficult for a supervisor to observe all the events that the plant executes due to lack of sensors, or due to the presence of faulty sensors. Therefore certain events may be indistinguishable or even completely unobservable to a supervisor. Such partial observations over plant events can be captured by the notion of an *event observation mask*, a function mapping each event to a set of "observable events" denote by certain observation symbols: $M : \Sigma \cup \{\epsilon\} \to \Lambda \cup \{\epsilon\}$, where Λ is the set of observed symbols, and $M(\epsilon) = \epsilon$. An untimed-trace $s = \sigma_1 \cdots \sigma_n$ is observed through the event-mask M as $M(s) = M(\sigma_1) \cdots M(\sigma_n)$. Given an untimed closed language $L \subseteq \Sigma^*$, the event-masked language $M(L)$ is defined by $M(L) := \{M(s) \mid s \in L\}$. Event observation mask M is said to be *deterministic*, if $\forall \sigma \in \Sigma$, $|M(\sigma)| = 1$. Here $|\cdot|$ denotes the size of a set, i.e., the number of elements belonging to the set. Consider a language K, and a deterministic observation mask M. K is said to be *observable* (with respect to a language L and an observation-mask M) if

$$\forall s, t \in pr(K), \sigma \in \Sigma \; s.t. \; M(s) = M(t), s\sigma \in pr(K), t\sigma \in L \;\Rightarrow\; t\sigma \in pr(K).$$

The property of observability requires that if the execution of an event σ following a trace s in $pr(K)$ results in a trace remaining in $pr(K)$, then the execution of an event σ following a trace t in $pr(K)$, which is indistinguishable from s, must either result in a trace in $pr(K)$, or a trace in $\Sigma^* - L$. An observation-mask M is said to be *nondeterministic* if $\forall \sigma \in \Sigma$, $|M(\sigma)| \geq 1$, and $\exists \sigma' \in \Sigma$ s.t $|M(\sigma')| > 1$. An event may have multiple possible observations under a nondeterministic observation mask. Such nondeterminism of observations on events is due to the limited sensing capabilities of sensors, or due to the presence of faulty sensors.

A language is a formal way to describe the behaviors of discrete event systems. It specifies all the admissible sequences of events that a DES can execute. However it is not always easy to specify or work with a "simple" representation of languages. Therefore an alternative modeling formalism to define and manipulate languages so that we can analyze and manipulate any arbitrarily complex language is needed [19]. An automaton is such graphical device with well-defined rules to represent languages.

An untimed automaton G is a five-tuple, $G = (X, \Sigma, \alpha, X_0, X_m)$, where

- X is a set of states,
- Σ is a finite set of events,
- $\alpha : X \times \Sigma \to X$ is a set of transitions,
- $X_0 \subseteq X$ is a set of initial states, and
- $X_m \subseteq X$ is a set of marked or accepting states.

For a transition $r \in \alpha$, r is in form of (x, σ, x'), where x is the source state, σ is the event label, and x' is the destination state.

An untimed-automaton is said to be *deterministic* if for all $x, x' \in X, \sigma \in \Sigma : |\{(x, \sigma, x') \in \alpha\}| \leq 1$, and otherwise it is said to be *nondeterministic*. The language generated (resp., accepted) by an automaton G consists of the traces executable in G starting from an initial state (resp., and ending in a final state in X_m), and is denoted $L(G)$ (resp., $L_m(G)$). It has been shown that for any nondeterministic automaton, there always exists an equivalent deterministic automaton, i.e., the generated as well as marked language represented by the two automata are the same. A *deadlock* appears if an automaton reaches a state $x \notin X_m$, and no further execution can be executed at x. Whereas a *livelock* appears when an automaton reaches a set of unmarked states, and there is no transition going out of such a set. If either deadlock or livelock can happen, then an automaton is said to be *blocking*, in which case the marked states of the system may never be reached. Therefore the property of absence of blockingness is expected. Given an automaton G, G is said to be *nonblocking* if $pr(L_m(G)) = pr(L(G))$.

The property of nonblockingness requires that any generated trace can be eventually extended to a sequence belonging to its marked behavior.

Given automaton $G_i = (X_i, \Sigma_i, \alpha, X_{0i}, X_{mi})$ for $i = 1, 2$, the composed automaton of G_1 and G_2 is defined as $G_1||G_2 := (X_1 \times X_2, \Sigma_1 \cup \Sigma_2, \alpha, X_{01} \cup X_{02}, X_{m1} \times X_{m2})$, where the set of transitions α is defined by

- $(x_i, \sigma, x_i') \in \alpha_i$, and $\sigma \in \Sigma_1 \cap \Sigma_2$, $\Rightarrow$ $((x_1, x_2), \sigma, (x_1', x_2')) \in \alpha$,
- $(x_1, \sigma, x_1') \in \alpha_1$, and $\sigma \in \Sigma_1 - \Sigma_2$, $\Rightarrow$ $((x_1, x_2), \sigma, (x_1', x_2)) \in \alpha$,
- $(x_2, \sigma, x_2') \in \alpha_2$, and $\sigma \in \Sigma_2 - \Sigma_1$, $\Rightarrow$ $((x_1, x_2), \sigma, (x_1, x_2')) \in \alpha$.

Synchronous composition of automata only allows transitions on common events. It can be used to model the joint behavior of the concurrently operating automata. Given a plant $G = (X, \Sigma, \alpha, X_0, X_m)$ and a supervisor $S = (Y, \Sigma, \beta, Y_0, Y_m)$, the controlled behavior of G under S can be represented by their synchronous composition $G\|S = (X \times Y, \Sigma, \gamma, X_0 \times Y_0, X_m \times Y_m)$, where $((x, y), \sigma, (x', y')) \in \gamma$ if and only if $(x, \sigma, x') \in \alpha$ and $(y, \sigma, y') \in \beta$.

The automaton representation is more convenient than enumeration of all the admissible sequences of events in the language. However in general a language is an infinite set, which may not be represented by finite state machines. We call a language that admits a finite state machine presentation a *regular* language. An automaton is unable to represent a non-regular language since it would require infinite number of states. Certain class of non-regular languages can be modeled by the other modeling formalism of DESs mentioned before, Petri nets.

2.2 Timed Language and Automaton

Let $\Re^+$ denote a set of nonnegative real numbers, and $\mathcal{N}$ denote a set of natural numbers. A *timed-trace* over an event set Σ is a sequence $\nu = (\sigma_1, t_1) \cdots (\sigma_n, t_n)$ where for $i = 1, \cdots, n$, $t_i \in \Re^+$; for $i = 1, \cdots, n-1$, $t_i \leq t_{i+1}$ and $\sigma_i \in \Sigma$; and for $i = n$, $\sigma_i \in \overline{\Sigma}$, where $\overline{\Sigma} := \Sigma \cup \{\epsilon\}$. Its corresponding *untimed-trace* is denoted as $\nu^{untime} = \sigma_1 \cdots \sigma_n$, in which event occurrence times are abstracted. We use $T(\nu) := t_n$ to denote the final time instant in ν. For $\nu = \epsilon$,

$T(\nu) := 0$. We denote the set of all timed-traces as $\mathcal{T}$. A subset of $\mathcal{T}$ is called a timed-language. For a timed-language $L \subseteq \mathcal{T}$, we use $pr(L) \subseteq \mathcal{T}$ to denote the set of all prefixes of the timed-traces belonging to L. $L \subseteq \mathcal{T}$ is said to be *prefix-closed*, or simply *closed* if $pr(L) = L$. Given a timed-language K, K is said to be *relative closed* (with respect to a language L) if $pr(K) \cap L = K$. A timed-language L is said to be *nonforcing* if an event is never forced to occur in L, i.e., a passing of time is always allowed: $\forall \nu \in pr(L)$ and $\infty \neq t > T(\nu)$, it holds that $\nu(\epsilon, t) \in pr(L)$. Note that time is fully observable under an event mask, therefore a timed-trace $\nu = (\sigma_1, t_1) \cdots (\sigma_n, t_n)$ is observed through an event-mask M as $M(\nu) = (M(\sigma_1), t_1) \cdots (M(\sigma_n), t_n)$. For a timed-language $L \subseteq \mathcal{T}$, its corresponding event-mask language $M(L)$ is defined by $M(L) := \{M(\nu) \mid \nu \in L\}$.

Given $\widehat{\Sigma} \subseteq \Sigma$, the projection of a timed-trace over $\widehat{\Sigma}$ is denoted by $\Pi_{\widehat{\Sigma}}(\cdot)$ and inductively defined as:

$$\Pi_{\widehat{\Sigma}}(\epsilon) := \epsilon,$$

$$\Pi_{\widehat{\Sigma}}(\nu(\sigma, t)) := \begin{cases} \Pi_{\widehat{\Sigma}}(\nu)(\sigma, t) & \text{if } \sigma \in \widehat{\Sigma} \\ \Pi_{\widehat{\Sigma}}(\nu)(\epsilon, t) & \text{otherwise} \end{cases}$$

where $\nu \in \mathcal{T}$, $\sigma \in \overline{\Sigma}, t \in \Re^+$. For $t \leq t' \in R^+$, we treat a "no event at t" followed by "an event $\sigma \in \Sigma$ at t'" to be equivalent to just "the event σ at t'". I.e., we treat $(\epsilon, t)(\sigma, t')$ to be equivalent to the timed-trace (σ, t'). And so it can be concluded that $\Pi_{\widehat{\Sigma}}(\nu) \in \mathcal{T}$.

A timed discrete event system can be modeled by a *timed-automaton* $A = (Q, \Sigma, \Xi, \Upsilon, I, Q_0, Q_m)$, where

- Q is a finite set of discrete states (locations);
- Σ is a finite set of events;
- Ξ is a finite set of clocks;
- $\Upsilon \subseteq Q \times \Sigma \times \Phi \times 2^{\Xi} \times Q$ is a set of transitions. Here Φ is the set of clock constraints. A clock constraint $\phi \in \Phi$ is a Boolean formula over atomic constraints of the form $\xi \sim c$ or $\xi_1 - \xi_2 \sim c$, where $\xi, \xi_1, \xi_2 \in \Xi$, $\sim \in \{\leq, <, =, >, \geq\}$, and c is a rational constant. Each

transition $\upsilon \in \Upsilon$ is a tuple (q, σ, ϕ, r, q') with q being the source discrete state, σ being the event associated with the transition, ϕ being a clock constraint representing the guard condition of the transition, r being the set of clocks to be reset by the transition, and q' being the destination discrete state.

- $I : Q \to \Phi$ is the invariant function, which assigns invariant conditions (belonging to Φ) to discrete states;
- $Q_0 \subseteq Q$ is the set of initial states;
- $Q_m \subseteq Q$ is the set of final states.

A *time-assignment* is a function $v : \Xi \to \Re^+$ assigning a nonnegative real value to each clock. Constants may be added to a time assignment: $(v + c)(\xi) := v(\xi) + c$. $[r \mapsto 0]v$ defines a time assignment which maps each clock in $r \subseteq \Xi$ to 0 and keeps all other clocks unchanged. Under this assignment we say that the clocks in r are *reset*. We use 0_v to denote the time assignment which maps every clock to 0.

A run of a timed-automaton A over a timed-trace $\nu = (\sigma_1, t_1) \cdots (\sigma_n, t_n)$ is a sequence of the form $(q_0, v_0) \cdots \overset{(\sigma_i, t_i)}{\to} (q_i, v_i) \cdots \overset{(\sigma_n, t_n)}{\to} (q_n, v_n)$ where $q_0 \in Q_0$; for each $i = 0, \cdots, n$, $q_i \in Q$, and time-assignment v_i satisfies the following requirements:

- **Initialization:** $v_0 = 0_v$;
- **Invariance:** $\forall i = 0, \cdots, n$, $\forall t \in [0, t_{i+1} - t_i]$, $v_i + t$ satisfies $I(q_i)$, where $t_0 = 0$.
- **Consecution:** $\forall i = 1, \cdots, n$, $\exists (q_{i-1}, \sigma_i, \phi_i, r_i, q_i) \in \Upsilon$ such that $v_{i-1} + t_i - t_{i-1}$ satisfies ϕ_i and $v_i = [r_i \mapsto 0](v_{i-1} + t_i - t_{i-1})$; if $\sigma_n \neq \epsilon$ then there is a tuple $(q_{n-1}, \sigma_n, \phi_n, r_n, q_n) \in \Upsilon$ such that $v_{n-1} + t_n - t_{n-1}$ satisfies ϕ_n and $v_n = [r_n \mapsto 0](v_{n-1} + t_n - t_{n-1})$, otherwise $q_n = q_{n-1}$ and $v_n = v_{n-1} + t_n - t_{n-1}$.

A timed-automaton A generates a finite timed-trace ν if A has a run over ν; it generates an infinite timed-trace ν if it generates all finite prefixes of ν. A generated finite timed-trace ν is accepted by A if a corresponding run over ν ends in a final state in Q_m; an infinite timed-trace is accepted by ν if a corresponding run over ν visits the set of final states infinitely

often. The *timed language* generated (resp., marked) by A, denoted by $L(A)$, (resp., $L_m(A)$) is the set of all the timed-traces generated (resp. marked) by A. A is said to be nonforcing if $L(A)$ is nonforcing. The *generated untimed language* of A is denoted by $L^{untime}(A) = \{\nu^{untime} \mid \nu \in L(A)\}$. Similarly, the *marked untimed language* of A is denoted by $L_m^{untime}(A) = \{\nu^{untime} \mid \nu \in L_m(A)\}$. Given timed-automata A and R, R is said to be *closed relative to* A if $pr(L_m(R)) \cap L_m(A) = L_m(R)$.

An untimed-automaton can be considered as a special timed-automaton with an empty set of clocks. Given a timed-automaton A, its corresponding region-automaton $\mathcal{R}(A) := (Q_{\mathcal{R}}, \Sigma, \Upsilon_{\mathcal{R}}, Q_{\mathcal{R}0}, Q_{\mathcal{R}m})$ is a nondeterministic untimed-automaton whose states consist of clock-regions together with the locations of A, and whose paths mimic the runs of A in a certain way. Here $\mathcal{R}$ denotes the set of clock-regions of A, $Q_{\mathcal{R}} = Q \times \mathcal{R}$, $Q_{\mathcal{R}0} = Q_0 \times 0_v$, $Q_{\mathcal{R}m} = Q_m \times \mathcal{R}$ and $((q, \alpha), \sigma, (q', \alpha')) \in \Upsilon_{\mathcal{R}}$ if and only if $\exists (q, \sigma, \phi, r, q') \in \Upsilon$ and $\exists \alpha'' \in \mathcal{R}$ such that (i) α'' is a time successor of α, (ii) α'' satisfies ϕ, and (iii) $\alpha' = [r \mapsto 0]\alpha''$. A clock region is an equivalence class of clock interpretations induced by the equivalence relation $\sim$. For $t \in \Re$, let $\lfloor t \rfloor$ and $fract(t)$ denote the integral part and the fractional part of t respectively. For each $x \in \Xi$, let c_x be the largest integer c such that $x \leq c$ or $c \leq x$ is a subformula of some clock constraints in Φ. The equivalence relation $v \sim v'$ holds if and only if the conditions below hold [6]:

1. For each $x \in \Xi$, either $\lfloor v(x) \rfloor$ and $\lfloor v'(x) \rfloor$ are the same, or both $v(x)$ and $v'(x)$ are greater than c_x.

2. For all $x, y \in \Xi$ with $v(x) \geq c_x$ and $v(y) \geq c_y$, $fract(v(x)) \leq fract(v(y))$ iff $fract(v'(x)) \leq fract(v'(y))$.

3. For all $x \in \Xi$ with $v(x) \leq c_x$, $fract(v(x)) = 0$ iff $fract(v'(x)) = 0$.

The following table lists the clock regions of a timed automaton with two clocks x and y with $c_x = 2$ and $c_y = 1$, which include 6 points, 14 open line segments and 8 open regions.

From [6], we have the number of clock regions is bounded by $|\Xi|! \cdot 2^{|\Xi|} \prod_{x \in \Xi}(2c_x + 2)$ [6, Lemma 4.5]. Note this result is obtained by restricting ourselves to the timed automata with

Table 2.1 Clock regions

	Clock Regions
Points	$(0,0)$, $(1,0)$, $(2,0)$, $(0,1)$, $(1,1)$, $(1,2)$
Open line segments	$[x=0, 0<y<1]$, $[x=0, y>1]$, $[0<x<1, y=1]$, $[0<x=y<1]$, $[0<x<1, y=0]$, $[x=1, 0<y<1]$, $[x=1, y>1]$, $[1<x<2, y=1]$, $[0<x-1=y<1]$, $1<x<2, y=0$, $[x=2, 0<y<1]$, $[x=2, y>1]$, $[x>2, y=1]$, $[x>2, y=0]$
Open regions	$[0<y<x<1]$, $[0<x<y<1]$, $[0<x<1, y>1]$, $[0<y<x-1<1]$, $[0<x-1<y<1]$, $[1<x<2, y>1]$, $[x>2, y>1]$, $[x>2, 0<y<1]$

the clock constraints only involving integer constants. For the timed automata in which the clock constraints are allowed to involve comparisons with rational constants, the least common multiple of the rational constants can be chosen so as to convert the constraints with rational constants to the ones with integral constants. In the following, we primarily consider the timed automata with integral constraints.

The following result from [6] shows the regularity of the corresponding untimed language of a timed automaton.

Theorem 1 *[6]*Given a timed automaton A, there exists an untimed automaton which accepts $L^{untime}(A)$.

Remark 1 From [6], we have the untimed language of a timed-automaton A can be represented by its region-automaton $\mathcal{R}(A)$, i.e., $L^{untime}(A) = L(\mathcal{R}(A))$. This further implies the marked untimed language of a timed-automaton A is regular, i.e., $L_m^{untime}(A) = L_m(\mathcal{R}(A))$.

Given a region automaton $\mathcal{R}(A)$, there exists a deterministic untimed-automaton $det[\mathcal{R}(A)] := (2^{Q_{\mathcal{R}}}, \Sigma, \widetilde{\Upsilon}, \{Q_{\mathcal{R}0}\}, \mathcal{Q}_m)$, where $(\tilde{Q}, \sigma, \tilde{Q}') \in \widetilde{\Upsilon}$ if $\widetilde{Q}' = \bigcup_{q \in \tilde{Q}:(q,\sigma,q') \in \Upsilon_{\mathcal{R}}} \{q'\}$ and $\mathcal{Q}_m := \{\tilde{Q} \subseteq Q_{\mathcal{R}} \mid \tilde{Q} \cap Q_{\mathcal{R}m} \neq \emptyset\}$, such that $L(\mathcal{R}(A)) = L(det[\mathcal{R}(A)]) \subseteq \Sigma^*$ and $L_m(\mathcal{R}(A)) = L_m(det[\mathcal{R}(A)]) \subseteq \Sigma^*$.

Given timed-automata $A_i = (Q_i, \Sigma_i, \Xi_i, \Upsilon_i, I_i, Q_{i0}, Q_{im})$, $i = 1, 2$, with disjoint Ξ_i, their composition is defined as $A_1 \| A_2 = (Q_1 \times Q_2, \Sigma_1 \cup \Sigma_2, \Xi_1 \cup \Xi_2, \Upsilon, I, Q_{10} \times Q_{20}, Q_{1m} \times Q_{2m})$ where $I(q_1, q_2) = I(q_1) \wedge I(q_2)$ and the transition set Υ is the smallest set with the property:

1. $\forall \sigma \in \Sigma_1 \cap \Sigma_2$, $\forall (q_1, \sigma, \phi_1, r_1, q_1') \in \Upsilon_1$, $\forall (q_2, \sigma, \phi_2, r_2, q_2') \in \Upsilon_2$, $((q_1, q_2), \sigma, \phi_1 \wedge \phi_2, r_1 \cup r_2, (q_1', q_2')) \in \Upsilon$.

2. $\forall \sigma \in \Sigma_1 - \Sigma_2$, $\forall (q_1, \sigma, \phi_1, r_1, q_1') \in \Upsilon_1$, $\forall q_2 \in Q_2$, $((q_1, q_2), \sigma, \phi_1, r_1, (q_1', q_2)) \in \Upsilon$.

3. $\forall \sigma \in \Sigma_2 - \Sigma_1$, $\forall (q_2, \sigma, \phi_2, r_2, q_2') \in \Upsilon_2$, $\forall q_1 \in Q_1$, $((q_1, q_2), \sigma, \phi_2, r_2, (q_1, q_2')) \in \Upsilon$.

Next we introduce the notions of non-speedingness (also called non-zenoness) and non-slowingness. The former requires that too many transitions shall not occur in a short time interval, whereas the latter requires that too few transitions shall not occur in a long time interval.

Definition 1 An infinite timed-trace $\nu = (\sigma_1, t_1) \cdots (\sigma_n, t_n) \cdots$ is said to be *non-speeding* or *non-zeno* if for every interval $[t, t+T] \subseteq \Re^+$ exists a count $N_{t,T} > 0$ such that

$$\forall i < j : [t_i, t_j] \subset [t, t+T] \Rightarrow j - i < N_{t,T}.$$

ν is said to be *uniformly non-speeding* if $N_{t,T}$ is independent of t. A timed language is said to be (uniformly) non-speeding if all its infinite timed-traces are (uniformly) non-speeding. A timed-automaton is (uniformly) non-speeding if its generated timed-language is (uniformly) non-speeding. In the following, it is assumed that a system model is non-speeding by default.

An infinite timed-trace $\nu = (\sigma_1, t_1) \cdots (\sigma_n, t_n) \cdots$ is said to be *non-slowing* if for every count set $[n, n+N] := \{n+k | 0 \leq k \leq N\} \subseteq \mathcal{N}$ exists an interval $T_{n,N} \in \Re^+$ such that

$$\forall i < j : [i, j] \subset [n, n+N] \Rightarrow t_j - t_i < T_{n,N}.$$

ν is said to be *uniformly non-slowing* if $T_{n,N}$ is independent of n. A timed language is said to be (uniformly) non-slowing if its each finite timed-trace possesses an infinite timed-trace extension, and its each infinite timed-trace is (uniformly) non-slowing. A timed-automaton is (uniformly) non-slowing if its generated timed-language is (uniformly) non-slowing.

For a non-slowing timed-language K, it holds that for each $\nu \in K$, exists $t > T(\nu)$ such that $\nu(\epsilon, t) \notin K$.

CHAPTER 3. TIMING-MASKED LANGUAGE AND ITS REGULARITY

In this chapter, we define the behavior of a dense timed-automaton when the event occurrence times are measured using a digital-clock of finite precision that measures time discretely by generating ticks. We show that the set of timed-traces in which the event occurrence times are measured through a digital-clock constitutes a regular language, i.e., can be represented by a finite (untimed) automaton.

3.1 Timing-Masked Language

A digital-clock is a generator of "ticks" (used to measure time with a finite-precision). It can be modeled as a timed-automaton $C = (Q_c, \{\tau\}, \Xi_c, \Upsilon_c, I_c, Q_{c0}, Q_c)$, in which $\tau \notin \Sigma$ denotes the "tick" event.

Note since time can never be stopped, a digital-clock C is required to be "non-speeding" (also known as non-Zeno), i.e., C cannot generate infinitely many tick-transitions within a finite time interval. Dually, since C should continue to generate ticks (for the measurement purpose), it is required to be "non-slowing", i.e., C cannot generate only a finitely many ticks in an infinite time interval.

Given a digital-clock C and a timed-trace ν, we use $M_C(\nu)$ to denote the set of all possible timing-masked observations of ν, where the passing of time is measured in form of the number of ticks generated by the digital-clock C.

Definition 2 Given a digital-clock C, the *timing-mask* M_C associated with C is defined as

follows: For a timed-trace $\nu = (\sigma_1, t_1) \cdots (\sigma_n, t_n)$,

$$\begin{aligned} M_C(\nu) := \{ \quad & \tau^{k_1}\sigma_1 \cdots \tau^{k_n}\sigma_n\tau^b \mid \\ & \exists \nu_1 \cdots \nu_{n+1} \in L(C) \text{ s.t.} \\ & \forall i \leq n+1, \nu_i = (\tau, t_i^1) \cdots (\tau, t_i^{k_i}), \\ & \forall i \leq n, t_i \in [t_i^{k_i}, t_{i+1}^1], \\ & b \in \{0,1\}, (b = 1 \Leftrightarrow t_n = t_{n+1}^1)\}, \end{aligned}$$

where $\tau^0 := \epsilon$ and $\tau^{i+1} := \tau \cdot \tau^i$ for all $i \geq 0$.

Note in the definition above, $\tau^{k_i}, i = 1, \ldots, n+1$ denotes the number of ticks that can occur in the interval $[t_{i-1}, t_i]$ (where $t_0 := 0$). Accordingly a timed-trace $\nu_1 \ldots \nu_{n+1}$ belongs to $L(C)$, where for each $i \leq n$, $\nu_i = (\tau, t_i^1) \ldots (\tau, t_i^{k_i})$ has k_i ticks, and moreover it holds that $t_i \in [t_i^{k_i}, t_{i+1}^1]$. Note it is possible that the occurrence of a tick coincides with that of an event σ_i of the timed-trace ν. Then according to the interleaving semantics, this is observed either as σ_i followed by the tick or, as the tick followed by σ_i. The timing-mask function includes the both possibilities. In particular it is possible that a tick transition occurs at the last event occurrence time t_n, and so (following the interleaving semantics) the observation of ν can consist of a single tick after the last event σ_n.

The following example illustrates Definition 2.

Example 3 Consider a digital-clock C with a period of 2 seconds and a jitter of 1 second as shown in Figure 3.1 (a) and a timed-trace $\nu = (a, 5.5)$. The timed-traces that C can execute by 5.5 seconds and ν are shown in Figure 3.1 (b) (d). From the definition of timing-mask, we have $M_C(\nu) = \{\tau a, \tau\tau a, \tau a\tau\}$. This is because $(\tau, t_1)(\tau, t_2) \in L(C)$ for $t_1 \in [2, 3]$ and $t_2 \in [4, 6]$. And so it is possible to observe one or two ticks before the occurrence of a. Also since C may tick at 5.5 seconds, synchronously with a, we have $\tau a\tau \in M_C(\nu)$ as well.

The *timing-mask generated language* (resp., *timing-masked marked language*) of a timed-automaton A, denoted by $M_C(L(A)) = \{M_C(\nu) | \nu \in L(A)\}$ (resp., $M_C(L_m(A)) = \{M_C(\nu) | \nu \in L_m(A)\}$), consists of all timing-masked observations of the timed-traces generated (resp.,

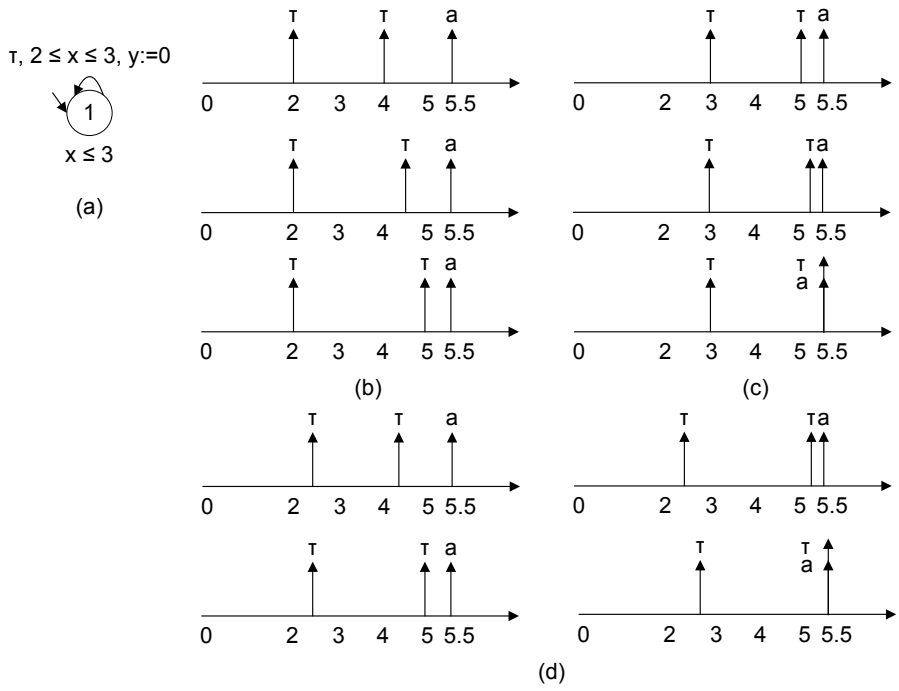

Figure 3.1 Timed-automaton of C and timing-masked observations of ν

marked) by A. It is obvious that a timing-masked language is a language over $\Sigma \cup \{\tau\} := \Sigma^\tau$. Since τ is just a symbol representing tick, $M_C(L(A))$ and $M_C(L_m(A))$ can be viewed as untimed languages over Σ^τ.

Remark 2 $M_C(\nu)$ consists of all the untimed observations of a timed-trace ν as observed through the digital-clock C. Since the number of ticks generated in any time interval can vary from execution to execution of the digital-clock, the timing-mask observation of a timed-trace is in general non-unique, i.e., $M_C(\cdot)$ is set-valued, and hence we also refer $M_C(\cdot)$ as being nondeterministic. (In contrast, an event mask function is typically deterministic yielding a unique observation for any executed trace [80, 66].)

Note if a tick event occurs simultaneously with an event $\sigma \in \Sigma$, then $\ldots\tau\sigma\ldots$ as well as $\ldots\sigma\tau\ldots$ must be included in the corresponding timing-masked observations. Thereby a timing-masked language of even a prefix-closed language need not be prefix-closed. For example, given $L = \{(a,1)(\epsilon,t_1),(\epsilon,t_2)\}$ with $t_1 > 1$, $t_2 \geq 0$ and a digital-clock which ticks every one time unit, we have $M_C(L) = \{\tau^*, \tau a\tau^*, a\tau\tau^*\}$. Note $a \in pr(M_C(L)) - M_C(L)$,

showing the non prefix-closure of $M_C(L)$.

3.2 Regularity of Timing-Masked Language

We present the regularity result of a marked timing-masked language in this section. A timing-masked language is in general not prefix-closed. In the following we first show that the prefix of the timing-masked generated language of a dense timed-automaton is regular.

Proposition 1 Given a timed-automaton A and a digital-clock timed-automaton C, let M_C be the timing-mask associated with C. Then $pr(M_C(L(A))) \subseteq (\Sigma^\tau)^*$ is a regular untimed language.

Proof: Let $A\|C$ be the product timed-automaton of A and C. In light of Theorem 1, it suffices to show that $pr(M_C(L(A))) = L^{untime}(A\|C)$.

We first prove the containment $pr(M_C(L(A))) \subseteq L^{untime}(A\|C)$. Pick $\nu_c = \tau^{k_1}\sigma_1 \cdots \tau^{k_l}\sigma_l\tau^r \in pr(M_C(L(A)))$. There exists $\nu'_c = \nu_c\tau^{k_{l+1}-r}\sigma_{l+1} \cdots \tau^{k_m}\sigma_m\tau^b \in M_C(L(A))$. So there exist $\mu'_A = (\sigma_1, t_1)\cdots(\sigma_l, t_l)(\sigma_{l+1}, t_{l+1})\cdots(\sigma_m, t_m)(\epsilon, t) \in L(A)$ and $\mu'_C = (\tau, t_1^1)\cdots(\tau, t_l^{k_l})(\tau, t_{l+1}^1)\cdots(\tau, t_m^{k_m})$ $\cdots(\tau, t_{m+1}^{b+1}) \in L(C)$ such that $\nu'_c \in M_C(\mu'_A)$ and $t_i \in [t_i^{k_i}, t_{i+1}^1]$ for $i = 1, \cdots, m$, $t \in [t_{m+1}^b, t_{m+1}^{b+1})$ (if $b = 0$, $t_{m+1}^b = t_m^{k_m}$). We need to consider the following two cases.

Case 1: $r \geq 1$. If $t_{l+1} = t_{l+1}^r$, i.e., σ_{l+1} and τ occur simultaneously, then from $\mu_A = (\sigma_1, t_1)\cdots(\sigma_l, t_l)(\sigma_{l+1}, t_{l+1}) \in pr(\mu'_A) \subseteq L(A)$ and $\mu_C = (\tau, t_1^1)\cdots(\tau, t_{l+1}^r) \in pr(\mu'_C) \subseteq L(C)$, we have $\rho = (\tau, t_1^1)\cdots(\tau, t_1^{k_1})(\sigma_1, t_1)\cdots(\tau, t_l^{k_l})(\sigma_l, t_l)(\tau, t_{l+1}^1)\cdots(\tau, t_{l+1}^r) \in pr(L(A\|C)) = L(A\|C)$. If $t_{l+1} \in (t_{l+1}^r, t_{l+2}^1]$, then we have $\rho \in L(A\|C)$.

Case 2: $r = 0$. If $t_l = t_{l+1}^1$, i.e., σ_l and τ occur simultaneously, then from $\mu_A = (\sigma_1, t_1)\cdots(\sigma_l, t_l) \in pr(\mu'_A) \subseteq L(A)$ and $\mu_C = (\tau, t_1^1)\cdots(\tau, t_l^{k_l}) \in pr(\mu'_C) \subseteq L(C)$, we have $\rho = (\tau, t_1^1)\cdots(\tau, t_1^{k_1})(\sigma_1, t_1)\cdots(\tau, t_l^{k_l})(\sigma_l, t_l) \in pr(L(A\|C)) = L(A\|C)$. If $t_l \in [t_l^{k_l}, t_{l+1}^1)$, we have $\rho \in L(A\|C)$.

In either case, we have $\nu_c = \rho^{untime} \in L^{untime}(A\|C)$.

Next we prove the reverse containment, $L^{untime}(A\|C) \subseteq pr(M_C(L(A)))$. Pick $\nu_c = \tau^{k_1}\sigma_1 \cdots \tau^{k_l}\sigma_l\tau^r \in L^{untime}(A\|C)$. There exists $\rho = (\tau, t_1^1)\cdots(\tau, t_1^{k_1})(\sigma_1, t_1)\cdots(\tau, t_l^{k_l})(\sigma_l, t_l)$

$\cdots (\tau, t^r_{l+1}) \in L(A\|C)$. Then we have $\mu_A = (\sigma_1, t_1)\cdots(\sigma_l, t_l) \in L(A)$, $\mu_C = (\tau, t^1_1)\cdots(\tau, t^r_{l+1}) \in L(C)$ and $t_i \in [t^{k_i}_i, t^1_{i+1}]$ for $i = 1, \cdots, l$. We need to consider the following two cases.

Case 1: $r \geq 1$. If $\mu_A(\sigma_{l+1}, t_{l+1}) \in L(A)$ and $t_{l+1} = t^r_{l+1}$, i.e., σ_{l+1} and τ occur simultaneously, then $\nu_c \in pr(M_C(\mu_A(\sigma_{l+1}, t_{l+1})))$. Otherwise, we have $\mu_A(\epsilon, t^r_{l+1}) \in L(A)$ and $\nu_c \in M_C(\mu_A(\epsilon, t^r_{l+1}))$.

Case 2: $r = 0$. If $t_l = t^1_{l+1}$, i.e., σ_l and τ occur simultaneously, then $\nu_c \in pr(M_C(\mu_A))$. Otherwise, $\nu_c \in M_C(\mu_A)$.

In either case, we have $\nu_c \in pr(M_C(L(A)))$. ■

Remark 3 Since the untimed-language of a timed-automaton can be generated by its region-automaton [6], it follows from Proposition 1 and the construction of a region-automaton [6] that the maximum number of states in the generator of $pr(M_C(L(A)))$ is linear in the number of states in A and C and exponential in the number of clocks and the encodings of the clock constraints in A and C. Instead of using a region-automaton, one may use a zone-automata which in practice possess a smaller state-space.

Proposition 1 establishes that the prefix closure of a timing-mask language is regular. In the following, we show that a timing-masked language itself is also regular.

Proposition 2 Given a timed-automaton A and a digital-clock timed-automaton C, let M_C be the timing-mask associated with C. Then $M_C(L(A)) \subseteq (\Sigma^\tau)^*$ is a regular untimed language.

Proof: From Proposition 1, we have that given a timed-automaton A, $pr(M_C(L(A))) = L^{untime}(A\|C)$. For any $\nu \in M_C(L(A)$, from $\nu \in pr(M_C(L(A)))$, then by following ν, a certain state x in $\mathcal{R}(A\|C)$ must be reached. If there exists a trace $\nu' \in pr(M_C(L(A))) - M_C(L(A))$, by executing which x is also reached, then x needs to be split into two copies, namely, a marked copy x reachable by ν and an unmarked copy x' reachable by ν'. Otherwise, we mark the state x. Then by this means, the automaton $\mathcal{R}(A\|C)$ is augmented by certain unmarked states. The size (i.e., the number of the states) of the resulting augmented automaton is at most twice of that of the original automaton $\mathcal{R}(A\|C))$ since each state in $\mathcal{R}(A\|C))$ is either unmarked, or marked, or split to the marked and unmarked copies. Further, since only the states reachable

by $\nu \in M_C(L(A)))$ are marked in the resulting augmented automaton, it accepts the language $M_C(L(A))$. Therefore we have the timing-masked language $M_C(L(A)))$ is regular. ■

With Proposition 2 in hand, it is easy to conclude that the timing-masked language of a marked timed-language is also regular. Combining the above results, we have the following theorem.

Theorem 2 Given a timed-automaton A and a digital-clock timed-automaton C, let M_C be the timing-mask associated with C. Then $M_C(L(A)) \subseteq (\Sigma^\tau)^*$ as well as $M_C(L_m(A)) \subseteq (\Sigma^\tau)^*$ are regular.

From the proof of Proposition 2, we notice that in order to decide whether a state of $\mathcal{R}(A\|C))$ should be marked, unmarked or split, we need know whether the state can be reached by a trace $\nu' \in pr(M_C(L(A))) - M_C(L(A))$. In order to identify such traces, the region-automaton $\mathcal{R}(A\|C)$ need be refined so that it is able to accept the non prefix-closed language $M_C(L_(A))$. This can be done in two steps. First, the transitions of $\mathcal{R}(A\|C)$ are extended to include ϵ-labeled transitions which keep track of the passing of time. This helps us identify the "forcing" states where the passing of time is not possible, and only a transition in Σ^τ can occur. The necessity for identifying the forcing states are explained as follows. Suppose a state x can be reached by a trace $\nu \in M_C(L_(A))$ and $\nu\sigma\tau, \nu\tau\sigma \in M_C(L_(A))$. If σ and τ are both forced, then we have the prefixes $\nu\sigma$ and $\nu\tau$ are not in $M_C(L_(A))$, and thus the states reached by these prefixes should be unmarked. Whereas if either σ or τ is not forced, then both of the prefixes are accepted by $M_C(L_(A))$. And so it is necessary to identify the forcing states. The resulting automaton is called an *extended region-automaton*, denoted by $\mathcal{R}^\epsilon(A\|C)$, and consists of a tuple $(Q_\mathcal{R}, \overline{\Sigma}^\tau, \Upsilon_{\mathcal{R}^\epsilon}, Q_{\mathcal{R}0}, Q_{\mathcal{R}m})$, where $\forall((q_A, q_C, \alpha), \sigma, (q'_A, q'_C, \alpha')) \in \Upsilon_\mathcal{R}$: $((q_A, q_C, \alpha), \epsilon, (q_A, q_C, \alpha_0))$, $\cdots$, $((q_A, q_C, \alpha_i), \epsilon, (q_A, q_C, \alpha_{i+1}))$, $\cdots$, $((q_A, q_C, \alpha_k), \sigma, (q'_A, q'_C, \alpha')) \in \Upsilon_{\mathcal{R}^\epsilon}$, where $\alpha_i \in \mathcal{R}$, α_{i+1} is the immediate time-successor of α_i for $i = 0, \cdots, k-1$ and α_0 (resp., α') is the immediate time-successor of α (resp., α_k).

In the second step, an extended region automaton is further refined to identify "event-pending" states and "tick-pending" states. Such states cannot be a final state since a certain

transition (either tick or event) is still "pending". To identify these states, we first introduce the notions of "forcing","tick-concurrent" and "pending" states.

Definition 3 Consider the extended region automaton $\mathcal{R}^{\epsilon}(A\|C)$. Let $\Sigma^{\tau}(q) \subseteq \overline{\Sigma}^{\tau}$ denote the set of events defined at state q of $\mathcal{R}^{\epsilon}(A\|C)$. q is said to be

- *forcing* if $\epsilon \notin \Sigma^{\tau}(q)$;
- *tick-concurrent* if forcing and exists $\sigma \in \Sigma$ such that $\{\tau, \sigma\} \subseteq \Sigma^{\tau}(q)$;
- *tick-pending* with respect to its predecessor q^{-} if forcing and $(q^{-}, \sigma, q) \in \Upsilon_{\mathcal{R}^{\epsilon}}$ such that q^{-} is tick-concurrent and
  - either all predecessors of q are forcing,
  - or $(q_1^{-}, \sigma_1, q), (q_2^{-}, \epsilon, q) \in \Upsilon_{\mathcal{R}^{\epsilon}}$ with q_1^{-} forcing and q_2^{-} nonforcing;
- *event-pending* with respect to its predecessor q^{-} if forcing and $(q^{-}, \tau, q) \in \Upsilon_{\mathcal{R}^{\epsilon}}$ such that q^{-} is tick-concurrent and
  - either for any predecessor q^{--} of q^{-}, $(q^{--}, \epsilon, q^{-}) \in \Upsilon_{\overline{\mathcal{R}}^{\epsilon}}$,
  - or $(q_1^{--}, \sigma_1, q^{-}), (q_2^{--}, \epsilon, q^{-}) \in \Upsilon_{\mathcal{R}^{\epsilon}}$ with q_2^{--} nonforcing.

Remark 4 It can be inferred that a forcing predecessor of a tick-pending state q must be tick-concurrent (since tick is enabled at q). Also, in the second clause of event-pending state, it can be inferred that if q_1^{--} is forcing, then q^{-} is a tick-pending state with respect to its predecessor q_1^{--}.

Note any tick/event-pending state cannot be a final state if it is reached along predecessors which render it a pending state (since some concurrently enabled transition is still pending to occur). And so each tick/event-pending state may be duplicated to make another copy, which cannot be marked. This is formalized in the following algorithm.

Algorithm 1 Given a plant A and a digital-clock C, the algorithm for constructing the *refined region-automaton* $\overline{\mathcal{R}}^{\epsilon}(A||C) := (Q_{\overline{\mathcal{R}}^{\epsilon}}, \Sigma^{\tau}, \Upsilon_{\overline{\mathcal{R}}^{\epsilon}}, Q_{\overline{\mathcal{R}}^{\epsilon}0}, Q_{\overline{\mathcal{R}}^{\epsilon}m})$, is presented as follows.

1. Obtain the region-automaton $\mathcal{R}(A\|C)$.

2. Obtain the extended region-automaton $\mathcal{R}^{\epsilon}(A\|C)$.

3. Construct the states $Q_{\overline{\mathcal{R}}^{\epsilon}}$. $\forall q \in Q_{\mathcal{R}^{\epsilon}}$:

   - if $q \in Q_{\mathcal{R}^{\epsilon}m}$ and tick-pending, then:
     - if all predecessors of q are forcing, $(q,0) \in Q_{\overline{\mathcal{R}}^{\epsilon}}$;
     - otherwise, $q, (q,0) \in Q_{\overline{\mathcal{R}}^{\epsilon}}$;
   - if $q \in Q_{\mathcal{R}^{\epsilon}m}$ and event-pending such that $(q^-, \tau, q) \in \Upsilon_{\mathcal{R}^{\epsilon}}$ with q^- being tick-concurrent, then:
     - if for any predecessor q^{--} of q^-, $(q^{--}, \epsilon, q^-) \in \Upsilon_{\overline{\mathcal{R}}^{\epsilon}}$, then $(q,0) \in Q_{\overline{\mathcal{R}}^{\epsilon}}$;
     - otherwise, $q, (q,0) \in Q_{\overline{\mathcal{R}}^{\epsilon}}$;
   - otherwise, $q \in Q_{\overline{\mathcal{R}}^{\epsilon}}$;

4. Construct the transitions $\Upsilon_{\overline{\mathcal{R}}^{\epsilon}}$. $\forall q \in Q_{\mathcal{R}^{\epsilon}}$:

   - if $q \in Q_{\mathcal{R}^{\epsilon}m}$ and tick-pending such that $(q^-, \sigma, q) \in \Upsilon_{\mathcal{R}^{\epsilon}}$ with q^- tick-concurrent, then:
     - if all predecessors of q are forcing, then:
       $(q^-, \sigma, (q,0)) \in \Upsilon_{\overline{\mathcal{R}}^{\epsilon}}$ (resp., $((q^-,0), \sigma, (q,0)) \in \Upsilon_{\overline{\mathcal{R}}^{\epsilon}}$) if $q^- \in Q_{\overline{\mathcal{R}}^{\epsilon}}$ (resp., $(q^-,0) \in Q_{\overline{\mathcal{R}}^{\epsilon}}$);
     - if $(q_1^-, \sigma_1, q), (q_2^-, \epsilon, q) \in \Upsilon_{\mathcal{R}^{\epsilon}}$ with q_1^- forcing and q_2^- nonforcing, then:
       $(q_1^-, \sigma_1, (q,0)) \in \Upsilon_{\overline{\mathcal{R}}^{\epsilon}}$ (resp., $((q_1^-,0), \sigma_1, (q,0)) \in \Upsilon_{\overline{\mathcal{R}}^{\epsilon}}$) if $q_1^- \in Q_{\overline{\mathcal{R}}^{\epsilon}}$ (resp., $(q_1^-,0) \in Q_{\overline{\mathcal{R}}^{\epsilon}}$), and $(q_2^-, \epsilon, q) \in \Upsilon_{\overline{\mathcal{R}}^{\epsilon}}$;
   - if $q \in Q_{\mathcal{R}^{\epsilon}m}$ and event-pending such that $(q^-, \tau, q) \in \Upsilon_{\mathcal{R}^{\epsilon}}$ with q^- being tick-concurrent, then:
     - if for any predecessor q^{--} of q^-, $(q^{--}, \epsilon, q^-) \in \Upsilon_{\overline{\mathcal{R}}^{\epsilon}}$, then:
       $(q^-, \tau, (q,0)) \in \Upsilon_{\overline{\mathcal{R}}^{\epsilon}}$;

– if $(q_1^{--}, \sigma_1, q^-), (q_2^{--}, \epsilon, q^-) \in \Upsilon_{\mathcal{R}^\epsilon}$ with q_1^{--} forcing (resp. nonforcing) and q_2^{--} nonforcing, then:

$((q^-, 0), \tau, q)$(resp. $(q^-, \tau, q)), (q^-, \tau, (q, 0)) \in \Upsilon_{\overline{\mathcal{R}}^\epsilon}$;

- otherwise, $(q^-, \sigma, q) \in \Upsilon_{\overline{\mathcal{R}}^\epsilon}$ (resp., $((q^-, 0), \sigma, q) \in \Upsilon_{\overline{\mathcal{R}}^\epsilon}$) if $(q^-, \sigma, q) \in \Upsilon_{\mathcal{R}^\epsilon}$ and $q^- \in Q_{\overline{\mathcal{R}}^\epsilon}$ (resp., $(q^-, 0) \in Q_{\overline{\mathcal{R}}^\epsilon}$);

5. Construct the marked states $Q_{\overline{\mathcal{R}}^\epsilon{}_m}$.

   $q \in Q_{\overline{\mathcal{R}}^\epsilon{}_m}$ iff $q \in Q_{\mathcal{R}^\epsilon{}_m} \cap Q_{\overline{\mathcal{R}}^\epsilon}$.

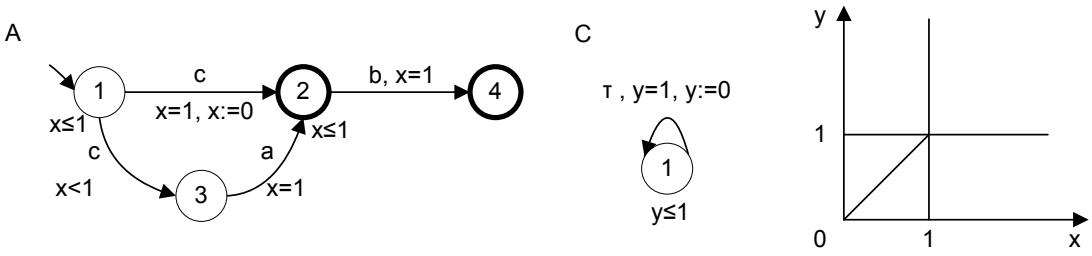

Figure 3.2 Timed automata of A,C and clock regions

The following example illustrates Algorithm 1.

Example 4 Given a timed-automaton A, digital-clock automaton C and the clock regions of $A||C$ as shown in Figure 3.2. The extended region-automaton $\mathcal{R}^\epsilon(A\|C)$ is shown in Figure 3.3, in which for clarity the location label for C is omitted (this does not cause any ambiguity since C has a single location). The corresponding refined region-automaton $\overline{\mathcal{R}}^\epsilon(A||C)$ is shown in Figure 3.4.

According to Algorithm 1, state $(2, (x = 0, y = 1))$ (where 2 is marked in A), is identified as a tick-pending state and unmarked since its *only* predecessor state $(1, (x = y = 1))$ is a tick-concurrent state. Similarly, state $(4, (x = y = 1))$ (where 4 is marked in A) is also identified as a tick-pending state, and so is unmarked. State $(2, (x = y = 1))$, identified as a tick-pending state, is duplicated to yield a marked copy $(2, (x = y = 1))$ and an unmarked copy $(2, 0, (x = y = 1))$. The unmarked copy is reached from the forcing predecessor $(3, (x = y = 1))$, whereas the marked copy is reached from nonforcing predecessor $(2, (0 < x = y < 1))$. The incoming transitions of state $(2, (x = y = 1))$ are thus accordingly split. State $(2, (x = 1, y = 0))$,

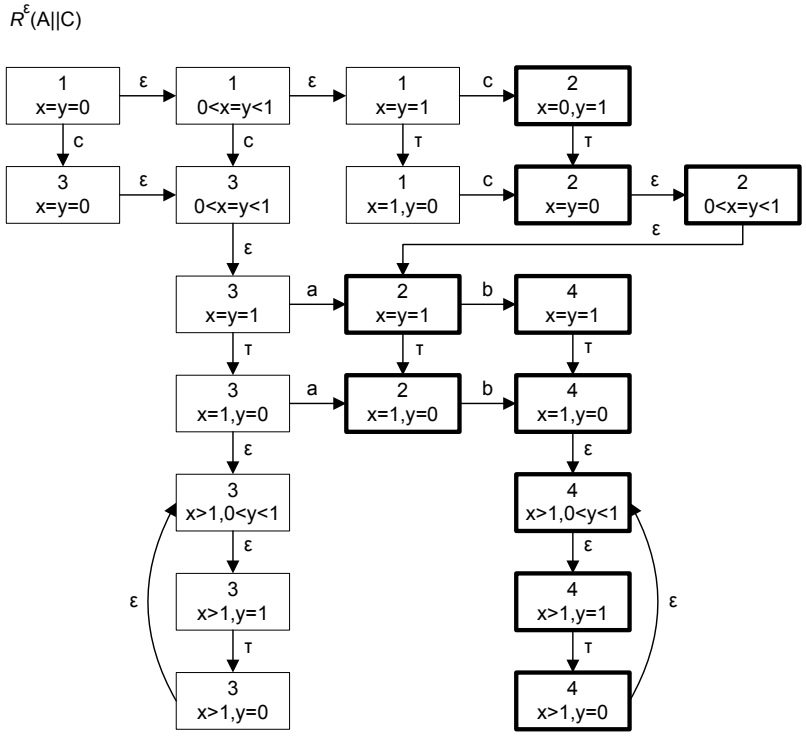

Figure 3.3 Extended region-automaton of $\mathcal{R}^{\epsilon}(A||C)$

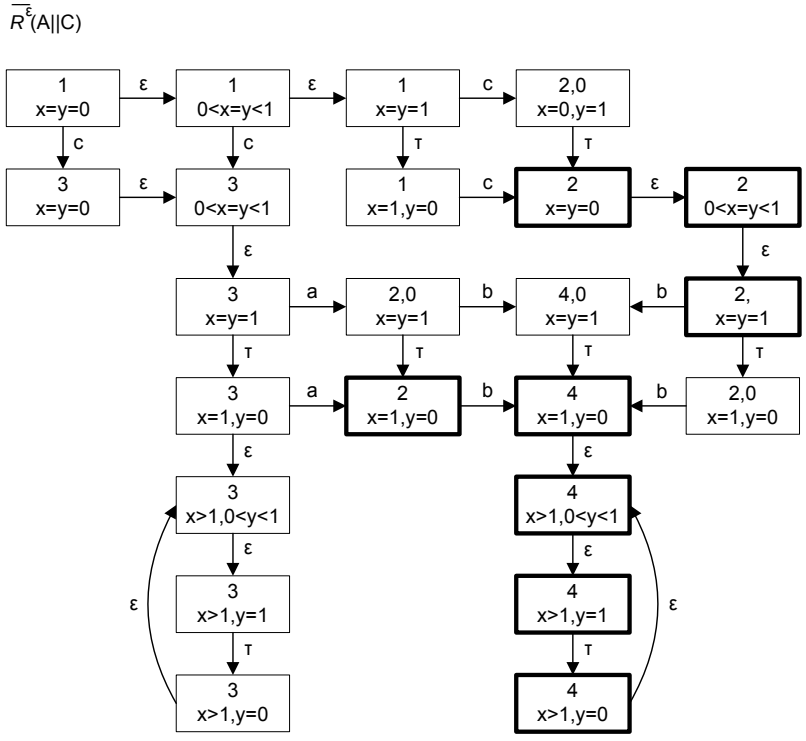

Figure 3.4 Refined region-automaton of $\overline{\mathcal{R}}^{\epsilon}(A||C)$

identified as an event-pending state, is also duplicated since its tick-concurrent predecessor $(2, (x = y = 1))$ possesses a forcing predecessor $(3, (x = y = 1))$ as well as a nonforcing predecessor; also the incoming transitions are accordingly split.

CHAPTER 4. SUPERVISORY CONTROL OF DENSE-TIME DESs USING DIGITAL-CLOCKS

In this chapter, we study the supervisory control of dense-time discrete event systems using finite-precision digital-clocks to observer event occurrence times. We start by introducing the notion of *control compatible* and *timing-mask compatible* control policies that do not disable uncontrollable events and also respect the timing-mask associated with a digital-clock, and show that a compatible control policy can be represented as a "digitalized"-automaton (an untimed-automaton in which the passing of time occurs discretely in form of the occurrences of ticks). We then introduce the notion of *observability with respect to the partial observation of time* resulting from the use of a digital-clock, and show that this property together with controllability serves as a necessary and sufficient condition for the existence of a control policy to enforce a real-time specification on a dense-time discrete event plant. The observability condition presented in the paper is very different from the one arising due to a partial observation of events since a partial observation of time is in general nondeterministic (the number of ticks generated in any time interval can vary from execution to execution of a digital-clock). Finally we present a method to check the proposed observability and controllability conditions, and also present a method to synthesize a supervisor when these conditions are satisfied. Further we examine the lattice structure of a class of timing-mask observable languages, and show that timing-mask observability is not preserved under intersection but preserved under union.

4.1 Compatible Control Policy

In this work we assume that plants as well as specifications are nonforcing. This is a practical assumption since it is unclear how to even precisely know a certain time so as to

enforce a certain event at that time.

Let $\Gamma = \{\widehat{\Sigma} \subseteq \overline{\Sigma}\}$ denote the set of control actions. Each control action consists of events in Σ to be enabled, and possibly ϵ to indicate that a passage of time is being allowed (i.e., an event is not being forced to occur).

A control policy for a plant with a generated language L is a partial function $f : L \to \Gamma$, mapping a timed-trace to a certain control action. We use $f(\nu)$ to denote the control action issued by f following a timed-trace ν. The generated controlled language, denoted L/f, is defined inductively as follows:

$$\epsilon \in L/f; \quad [\nu \in L/f, \nu(\sigma, T(\nu)) \in L, \sigma \in f(\nu)] \Leftrightarrow [\nu(\sigma, T(\nu)) \in L/f].$$

Letting L_m denote the marked plant language, the marked controlled language is given by $L_m/f := L_m \cap L/f$. A control policy f is said to be *nonforcing* if a passing of time is always allowed, i.e., for any $\nu \in L$, $\epsilon \in f(\nu)$. It is said to be *nonblocking* if $pr(L_m/f) = L/f$. Nonblockingness requires that any generated timed-trace of the controlled language must not be blocked from being able to extend to a marked timed-trace of the controlled language.

To understand the restriction imposed by a timing mask on a control policy, we also consider another type of control policy that is defined over the observations of a plant language. For a timed-trace $\nu \in L$, the set of enabled events following ν will consist of those events that are enabled after an enabled observation in $M_C(\nu)$. Additionally, if a tick can occur at $T(\nu)$, i.e., at the very end of ν, then the set of enabled events following ν will also include those events that are enabled following an enabled observation of ν prior to the occurrence of the very last tick at $T(\nu)$, as those events remain enabled even at the instance $T(\nu)$ (since no other observations occur in the interim, and so the control action cannot change).

This motivates us to introduce the set of extended timing-mask observations (or simply extended observations) of ν, which consists of all timing-mask observations of ν, together with the observations of ν prior to the occurrence of the very last tick at $T(\nu)$:

Definition 4 Given a digital-clock C, the *extended timing-mask observation* of ν, denoted $M_C^+(\nu)$, is defined as $M_C^+(\nu) := M_C(\nu) \cup \{\nu_c \mid \nu_c\tau \in M_C(\nu), \forall \sigma \in \Sigma : \nu_c\sigma\tau \in M_C(\nu(\sigma, T(\nu)))\}$.

Note the second term in the union of Definition 4 consists of the observations of ν prior to the very last tick at $T(\nu)$. Under the non-forcingness of L, it is easy to see that $M_C^+(L) = pr(M_C(L))$, i.e., the set of extended observations of L equals all observations of L together with all their prefixes. Also it follows from Definition 4 that given a timed trace $\nu = (\sigma_1, t_1)\cdots(\sigma_n, t_n)$, if $\nu_c = \tau^{k_1}\sigma_1\cdots\tau^{k_n}\sigma_n\tau^b \in M_C^+(\nu)$, then $\overline{\nu}_c = \tau^{k_1}\sigma_1\cdots\tau^{k_i}\sigma_i \in M_C^+(\overline{\nu})$ for $\overline{\nu} = (\sigma_1, t_1)\cdots(\sigma_i, t_i)$.

The lemma below asserts certain properties of the extended timing-mask observations, which follow from Definitions 2 and 4.

Lemma 1 Given a digital-clock C, the following holds for $\sigma \in \Sigma$ and timed-traces ν, $\nu(\sigma, T(\nu))$:

- $\forall \nu_c \in M_C^+(\nu)$, $\exists \nu'_c \in M_C(\nu(\sigma, T(\nu))$ such that either $\nu'_c = \nu_c\sigma$ or $\nu'_c = \nu_c\sigma\tau$.
- $\forall \nu'_c \in M_C(\nu(\sigma, T(\nu)))$, $\exists \nu_c \in M_C^+(\nu)$ such that either $\nu_c\sigma = \nu'_c$ or $\nu_c\sigma\tau = \nu'_c$.
- $\forall \nu_c \in M_C^+(\nu)$, either $\nu_c \in M_C(\nu)$ or $\nu_c\tau \in M_C(\nu)$.

Consider a control policy $\tilde{f} : M_C^+(L) \to \Gamma^\tau$ defined over the set of extended observations (equivalently, over the set of observations together with their prefixes, for $M_C^+(L) = pr(M_C(L))$), where $\Gamma^\tau := \{\widehat{\Sigma} \subseteq \overline{\Sigma}^\tau\}$. Then the controlled-behavior under such a control policy can be defined as follows:

$$\nu = (\sigma_1, t_1)\cdots(\sigma_n, t_n) \in L,\ \nu \in L/\tilde{f} \Leftrightarrow \exists \nu_c = \tau^{k_1}\sigma_1\cdots\tau^{k_n}\sigma_n\tau^b \in M_C^+(\nu) \text{ s.t.}$$
$$\sigma_1 \in \tilde{f}(\tau^{k_1}), \forall i = 2, \cdots, n:\ \sigma_i \in \tilde{f}(\tau^{k_1}\sigma_1\cdots\tau^{k_{i-1}}\sigma_{i-1}\tau^{k_i}).$$

We say that $\tilde{f}$ is nonforcing if for any untimed trace ν_c for which $\tilde{f}(\nu_c)$ is defined, $\epsilon, \tau \in \tilde{f}(\nu_c)$.

A nonforcing control policy $f : L \to \Gamma$ over a plant language is able to respect the restrictions imposed by a timing-mask if and only if it is equivalent to a control policy $\tilde{f} : M_C^+(L) \to \Gamma^\tau$ over the extended observations of a plant language, in the sense that the two yield the same controlled-language. For this to occur, a control policy f needs to satisfy a certain restriction which we identify below as the property of (L, M_C)-compatibility. The basic idea is simple: If each extended-observation of $\nu \in L$ is either an extended-observation of a timed-trace where $\sigma \in \Sigma$ is disabled, or possesses a prefix that is an extended observation

of a disabled timed-trace, then σ must be disabled following ν (since in this case there exists no enabled extended observation of ν where σ is enabled).

The following definition formalizes the notion of (L, M_C)-compatibility, in which $[M_C^+(L - L/f)](\Sigma^\tau)^*$ denotes the set of extended observations whose prefixes share an extended observation of a disabled trace. The definition also defines (L, Σ_u)-compatibility which captures the restriction that uncontrollable events cannot be disabled.

Definition 5 Let L be a generated timed-language of a plant, $\Sigma_u \subseteq \Sigma$ be a set of uncontrollable events, M_C be the timing-mask associated with a digital-clock C. A control policy f is said to be

- (L, M_C)-*compatible*, if $\forall \nu \in L/f$, $\sigma \in \Sigma$ such that $\nu(\sigma, T(\nu)) \in L$: if exists $H \subseteq L/f$ such that $\forall \nu' \in H, \nu'(\sigma, T(\nu')) \in L - L/f$, and $M_C^+(\nu) \subseteq M_C^+(H) \cup [M_C^+(L - L/f)](\Sigma^\tau)^*$, then $\sigma \notin f(\nu)$.
- (L, Σ_u)-*compatible*, if $\forall \nu \in L/f$, $\sigma \in \Sigma_u$ such that $\nu(\sigma, T(\nu)) \in L$: $\nu(\sigma, T(\nu)) \in L/f$.
- (L, Σ_u, M_C)-*compatible*, if f is (L, Σ_u)-compatible and (L, M_C)-compatible.

The following example illustrates the concept of timing-mask compatibility.

Example 5 Consider the generated timed-language L of a plant G, and a digital-clock C as shown in Figure 4.1.

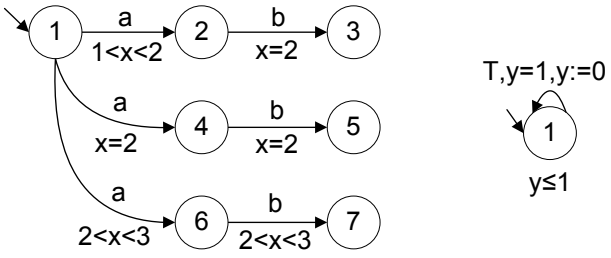

Figure 4.1 Timed-automata of plant G (left) and digital-clock C (right)

Suppose events a, b are controllable. Let f be a control policy that enables a after $(\epsilon, 1 < t \leq 2)$ whereas disables a after $(\epsilon, 2 < t < 3)$, and enables b after $(a, 2)$ whereas disables b after

$(a, 1 < t < 2)(\epsilon, 2)$. Note the extended observations of an enabled timed-trace $(a, 2) \in L/f$ are given by $M_C^+((a, 2)) = \{\tau a, \tau a \tau, \tau\tau a\}$ (for $M_C((a, 2)) = \{\tau a \tau, \tau\tau a\}$). Since the extended observation $\tau\tau a$ of $(a, 2)$ is shared by a disabled timed-trace $(a, 2 < t_1 < 3) \in L - L/f$ (for $M_C^+((a, t_1)) = M_C((a, t_1)) = \{\tau\tau a\}$), for b to be enabled after $(a, 2)$, it needs to be enabled following the extended observations in $M_C^+((a, 2)) - [M_C^+(L - L/f)](\Sigma^\tau)^*$, i.e., $\{\tau a, \tau a \tau\}$. However, b is disabled after the timed-trace $(a, 1 < t_2 < 2)(\epsilon, 2) \in L/f$ for which the extended observation is given by $M_C^+((a, t_2)(\epsilon, 2)) = \{\tau a, \tau a \tau\}$. It follows that f is not timing-mask compatible. On the other hand, if b were to be enabled following $(a, t_2)(\epsilon, 2)$, the resulting control policy would then be timing-mask compatible.

In the following we show that whenever f is (L, M_C)-compatible, the controlled language under f can also be achieved under $\tilde{f}$, and vice versa.

Proposition 3 Let L be the generated timed-language of a nonforcing plant, M_C be the timing-mask associated with a digital-clock C.

1. Given a (L, M_C)-compatible nonforcing control policy f, there exists a nonforcing control policy $\tilde{f}$ such that $L/\tilde{f} = L/f$.

2. Given a nonforcing control policy $\tilde{f}$, there exists a (L, M_C)-compatible nonforcing control policy f such that $L/f = L/\tilde{f}$.

Proof: We start by proving the first assertion. Let $K := pr(M_C(L/f)) - [M_C^+(L - L/f)](\Sigma^\tau)^*$. Since $pr(M_C(L/f))$ is prefix-closed and $[M_C^+(L - L/f)](\Sigma^\tau)^*$ is suffix-closed, K is prefix-closed. From $\epsilon \in pr(M_C(L/f))$ and nonforcing-ness of L and f, $\epsilon \notin M_C^+(L - L/f)$. This implies K is nonempty. Let $S = (Q, \Sigma^\tau, \Upsilon, Q_0, Q)$ be an untimed-automaton with $L(S) = K$. $\forall \nu_c \in L(S)$, we have $\nu_c\tau \in L(S)$, i.e., $\nu_c\tau^* \in L(S)$. This can be proved by contradiction as follows. Suppose $\nu_c\tau \notin L(S)$. Then from $\nu_c \in pr(M_C(L))$ and nonforcing-ness of f, $\nu_c\tau \in pr(M_C(L))$. Therefore $\nu_c\tau \in [M_C^+(L - L/f)](\Sigma^\tau)^*$. Note $\nu_c \notin [M_C^+(L - L/f)](\Sigma^\tau)^*$. Then it must be the case $\nu_c\tau \in M_C^+(L - L/f)$. It follows that exists $\nu' \in L - L/f$ s.t. $\nu_c\tau \in M_C^+(\nu')$. Then $\exists \nu'' \in pr(\nu')$ s.t. $\nu''(\epsilon, T(\nu')) = \nu'$ and $\nu_c \in M_C^+(\nu'')$. Note $\nu'' \in L - L/f$.

Then $\nu_c \in M_C^+(L - L/f)$, which yields a contradiction. Define a nonforcing control policy $\tilde{f}$: $\forall \nu_c \in L(S)$, $\tilde{f}(\nu_c) := \{\sigma \in \Sigma^\tau | \nu_c\sigma \in L(S)\} \cup \{\epsilon\}$. We claim that $L/\tilde{f} = L/f$.

We first show $L/f \subseteq L/\tilde{f}$ by contradiction. Pick $\nu = (\sigma_1, t_1)\cdots(\sigma_n, t_n) \in L/f$ (Without loss of generality, $\sigma_i \in \Sigma$ for $i = 1, \cdots, n-1$, and $\sigma_n \in \overline{\Sigma}$). Suppose $\nu \notin L/\tilde{f}$. Then $\forall \nu_c = \tau^{k_1}\sigma_1 \cdots \tau^{k_n} \in M_C^+((\sigma_1, t_1)\cdots(\epsilon, t_n))$, if exists $j \leq n-2$ s.t. $\sigma_{j+1} \notin \tilde{f}(\tau^{k_1}\sigma_1 \cdots \tau^{k_i}\sigma_i\tau^{k_{j+1}})$, then $\overline{\nu}_c\sigma_{j+1} \notin L(S)$ for $\overline{\nu}_c = \tau^{k_1}\sigma_1 \cdots \tau^{k_i}\sigma_i\tau^{k_{j+1}}$. Note $\overline{\nu}_c\sigma_{j+1} \in M_C^+(\overline{\nu})$, where $\overline{\nu} = (\sigma_1, t_1)$ $\cdots(\sigma_j, t_j)(\sigma_{j+1}, t_{j+1}) \in pr(\nu)$. We have $\overline{\nu}_c\sigma_{j+1} \in pr(M_C(L/f))$. Thereby $\overline{\nu}_c\sigma_{j+1} \in [M_C^+(L - L/f)](\Sigma^\tau)^*$. This implies $\nu_c \in [M_C^+(L - L/f)](\Sigma^\tau)^*$. On the other hand, if $\forall j \leq n-2$, $\sigma_{j+1} \in \tilde{f}(\tau^{k_1}\sigma_1 \cdots \tau^{k_i}\sigma_i\tau^{k_{j+1}})$ $(\sigma_1 \in \tilde{f}(\tau^{k_1}))$, then $\overline{\nu} = (\sigma_1, t_1)\cdots(\sigma_{n-1}, t_{n-1}) \in L/\tilde{f}$ and $\tau^{k_1}\sigma_1 \cdots \tau^{k_{n-1}}\sigma_{n-1} \in L(S)$. It follows that $\nu_c \in L(S)$ and $\overline{\nu}(\epsilon, t_n) \in L/\tilde{f}$. From $\nu \notin L/\tilde{f}$, we have $\sigma_n \notin \tilde{f}(\nu_c)$, i.e., $\nu_c\sigma_n \notin L(S)$. This further implies $\nu_c\sigma_n \in [M_C^+(L - L/f)](\Sigma^\tau)^*$. Note $\nu_c \in L(S)$, $\nu_c\sigma_n \in M_C^+(L - L/f)$. Therefore there exists $\nu' = (\sigma_1, t_1')\cdots(\sigma_n, t_n') \in L - L/f$ s.t. $\nu_c\sigma_n \in M_C^+(\nu')$. Note $\nu_c \in M_C^+(\overline{\nu}')$, where $\overline{\nu}' = (\sigma_1, t_1')\cdots(\sigma_{n-1}, t_{n-1}')(\epsilon, t_n')$. This together with $\overline{\nu}' \in L$ and $\nu_c \in L(S)$ implies $\overline{\nu}' \in L/f$. Let $H := \{\overline{\nu}'\}$. Then $\forall \overline{\nu}' \in H$, $\overline{\nu}' \in L/f$, $\overline{\nu}'(\sigma_n, T(\nu')) \in L - L/f$ and $\nu_c \in M_C^+H$. Therefore $M_C^+(\nu) \subseteq M_C^+(H) \cup [M_C^+(L - L/f)](\Sigma^\tau)^*$. Then from (L, M_C)-compatibility of f, $\sigma_n \notin f((\sigma_1, t_1)\cdots(\epsilon, t_n))$, i.e., $\nu \notin L/f$, which yields a contradiction.

We next show $L/\tilde{f} \subseteq L/f$ by contradiction. Pick $\nu = (\sigma_1, t_1)\cdots(\sigma_n, t_n) \in L/\tilde{f}$. Suppose $\nu \notin L/f$. Then $\forall \nu_c \in M_C^+(\nu)$, $\nu_c \in M_C^+(L - L/f)$. Therefore $\nu_c \notin L(S)$. However on the other hand, from $\nu \in L/\tilde{f}$, there exists $\nu_c = \tau^{k_1}\sigma_1 \cdots \tau^{k_n}\sigma_n \in M_C^+(\nu)$ s.t. $\sigma_1 \in \tilde{f}(\tau^{k_1})$, and for $i = 2, \cdots, n$, $\sigma_i \in \tilde{f}(\tau^{k_1}\sigma_1 \cdots \tau^{k_{i-1}}\sigma_{i-1}\tau^{k_i})$. This implies $\nu_c \in L(S)$. A contradiction arrives.

Then we prove the second assertion. Define a nonforcing control policy f: $\forall \nu \in \mathcal{T}$, $f(\nu) := \bigcup_{\nu_c \in M_C^+(\nu)}\{\sigma \in \Sigma \mid \sigma \in \tilde{f}(\nu_c)\} \cup \{\epsilon\}$, where $\nu_c = \tau^{k_1}\sigma_1 \cdots \tau^{k_n}\sigma_n\tau^b \in M_C^+(\nu)$ s.t. $\sigma_1 \in \tilde{f}(\tau^{k_1})$, and for $i = 2, \cdots, n$, $\sigma_i \in \tilde{f}(\tau^{k_1}\sigma_1 \cdots \tau^{k_{i-1}}\sigma_{i-1}\tau^{k_i})$. We claim that $L/f = L/\tilde{f}$ and f is (L, M_C)-compatible .

From the definition of f and $\tilde{f}$, it is easy to see $L/f = L/\tilde{f}$. Next we show f is (L, M_C)-compatible. Pick $\nu = (\sigma_1, t_1)\cdots(\sigma_n, t_n) \in L/f$, $\sigma \in \Sigma$ s.t. $\nu(\sigma, T(\nu)) \in L$. Suppose $\exists H \subseteq L/f$ s.t. $\forall \nu' \in H$, $\nu'(\sigma, T(\nu')) \in L - L/f$ and $M_C^+(\nu) \subseteq M_C^+(H) \cup [M_C^+(L - L/f)](\Sigma^\tau)^*$. We need

show $\sigma \notin f(\nu)$. Pick $\nu_c = \tau^{k_1}\sigma_1 \cdots \tau^{k_n}\sigma_n\tau^b \in M_C^+(\nu)$ s.t. $\sigma_1 \in \tilde{f}(\tau^{k_1})$, and for $i = 2, \cdots n$, $\sigma_i \in \tilde{f}(\tau^{k_1}\sigma_1 \cdots \tau^{k_{i-1}}\sigma_{i-1}\tau^{k_i})$. It must be true that $\nu_c \in M_C^+(H)$. This is because all the plant events are observable. Therefore for any trace ν' s.t. $\nu_c \in M_C^+(\nu')$, ν' must be in form of $(\sigma_1, t_1') \cdots (\sigma_n, t_n')$. Note $\sigma_1 \in \tilde{f}(\tau^{k_1})$, and $\forall i = 2, \cdots n$, $\sigma_i \in \tilde{f}(\tau^{k_1}\sigma_1 \cdots \tau^{k_{i-1}}\sigma_{i-1}\tau^{k_i})$, $\nu' \in L/f$. Similarly for any trace $\overline{\nu}'$ s.t. $\exists \overline{\nu}_c \in pr(\nu_c) \cap M_C^+(\overline{\nu}')$, we have $\overline{\nu}' \in L/f$. This implies $\nu_c \notin [M_C^+(L - L/f)](\Sigma^\tau)^*$. And so we have $\nu_c \in M_C^+(H)$. Note $\forall \nu' \in H$, $\nu'(\sigma, T(\nu')) \in L - L/f$. Therefore $\sigma \notin \tilde{f}(\nu_c)$, i.e., $\sigma \notin f(\nu)$. ■

From the above proposition, it is clear that the timing-mask compatibility is a requirement for a control policy defined over a plant language to be realizable as a control policy defined over the observations of a plant language.

4.2 Representation of Compatible Control Policy

In the following, we provide a representation of a compatible control policy as a digitalized-automaton (an untimed automaton that evolves over the events of the plant and the ticks of the digital-clock).

We first need to define the following two compatibility properties of an untimed-automaton: Given an untimed-automaton $S = (Q, \Sigma^\tau, \Upsilon, Q_0, Q_m)$, S is said to be $\{\tau\}$-*compatible* if $\forall q \in Q$, $\exists q' \in Q$ s.t. $(q, \tau, q') \in \Upsilon$. Given a generated timed-language L, S is said to be $(pr(M_C(L)), \Sigma_u)$-*compatible* if $\forall \nu_c \in L(S)$, $\sigma \in \Sigma_u$ and $\nu_c\sigma \in pr(M_C(L))$, then $\nu_c\sigma \in L(S)$.

Lemma 2 below shows that a compatible control policy can be represented by a "digitalized"-automaton.

Lemma 2 Let L and L_m be the generated and marked timed-languages of a nonforcing plant G, Σ_u be a set of uncontrollable events, and M_C be the timing-mask associated with a digital-clock C.

- Given a (L, M_C)-compatible nonforcing control policy f, there exists a $\{\tau\}$-compatible untimed-automaton $S = (Q, \Sigma^\tau, \Upsilon, Q_0, Q)$ such that $\Pi_\Sigma(L(G \| S \| C)) = L/f$; $\Pi_\Sigma(L_m(G \| S \| C)) = L_m/f$.

- Given a (L, Σ_u, M_C)-compatible nonforcing control policy f, there exists a $(pr(M_C(L)), \Sigma_u)$-compatible and $\{\tau\}$-compatible untimed-automaton $S = (Q, \Sigma^\tau, \Upsilon, Q_0, Q)$ such that $\Pi_\Sigma(L(G\|S\|C)) = L/f$; $\Pi_\Sigma(L_m(G\|S\|C)) = L_m/f$.

Proof: Define the language $K := pr(M_C(L/f)) - [M_C^+(L - L/f)](\Sigma^\tau)^*$. From the proof of Proposition 3, we know K is nonempty and prefix-closed. Let $S = (Q, \Sigma^\tau, \Upsilon, Q_0, Q)$ be an untimed-automaton with $L(S) = K$. From the proof of Proposition 3, we have S is $\{\tau\}$-compatible. We claim that given a (L, M_C)-compatible control policy f, $\Pi_\Sigma(L(G\|S\|C)) = L/f$.

To show $\Pi_\Sigma(L(G\|S\|C)) = L/f$, we first show $\nu \in \Pi_\Sigma(L(G\|S\|C))$ implies $\nu \in L/f$ by induction on the length of ν. $\epsilon \in \Pi_\Sigma(L(G\|S\|C)) \cap L/f$, establishing the base step. For the induction step, pick $\nu = \overline{\nu}(\sigma, t) \in \Pi_\Sigma(L(G\|S\|C))$. We need to show $\nu \in L/f$. For $\sigma = \epsilon$, from nonforcing-ness of f, $\overline{\nu}(\epsilon, t) \in L/f$. For $\sigma \in \Sigma$, suppose for contradiction that $\overline{\nu}(\sigma, t) \notin L/f$. Then $\overline{\nu}(\sigma, t) \in L - L/f$. Thereby $\forall \nu_c \in M_C^+(\overline{\nu}(\epsilon, t))$, $\nu_c\sigma \in M_C^+(\overline{\nu}(\sigma, t)) \subseteq M_C^+(L - L/f)$, i.e., $\nu_c\sigma \notin L(S)$. On the other hand, from $\overline{\nu}(\sigma, t) \in \Pi_\Sigma(L(G\|S\|C))$, exists $\nu_c' \in M_C^+(\overline{\nu}(\epsilon, t))$ s.t. $\nu_c'\sigma \in L(S)$, which yields a contradiction.

We next show $\nu \in L/f$ implies $\nu \in \Pi_\Sigma(L(G\|S\|C))$ by induction on the length of ν. $\epsilon \in \Pi_\Sigma(L(G\|S\|C)) \cap L/f$, establishing the base step. For the induction step, pick $\nu = \overline{\nu}(\sigma, t) \in L/f$. We need to show $\nu \in \Pi_\Sigma(L(G\|S\|C))$. For $\sigma = \epsilon$, from induction hypothesis $\overline{\nu} \in \Pi_\Sigma(L(G\|S\|C))$, $\exists \overline{\nu}_c \in M_C(\overline{\nu})$ s.t. $\overline{\nu}_c \in L(S)$, i.e., $\overline{\nu}_c\tau^* \in L(S)$. Therefore $\overline{\nu}(\epsilon, t) \in \Pi_\Sigma(L(G\|S\|C))$. For $\sigma \in \Sigma$, suppose for contradiction that $\overline{\nu}(\sigma, t) \notin \Pi_\Sigma(L(G\|S\|C))$. Then $\forall \nu_c \in M_C^+(\overline{\nu}(\epsilon, t))$, $\nu_c\sigma \notin L(S)$. Note $\nu_c\sigma \in pr(M_C(\overline{\nu}(\sigma, t))) \subseteq pr(M_C(L/f))$. We have $\nu_c\sigma \in [M_C^+(L - L/f)](\Sigma^\tau)^*$. If $\exists \overline{\nu}_c \in pr(\nu_c)$ s.t. $\overline{\nu}_c \in M_C^+(L - L/f)$, then $\nu_c \in [M_C^+(L - L/f)](\Sigma^\tau)^*$. On the other hand, if $\forall \overline{\nu}_c \in pr(\nu_c)$, $\overline{\nu}_c \notin M_C^+(L - L/f)$, then it must be the case that $\nu_c\sigma \in M_C^+(L - L/f)$. Thereby $\exists \nu'(\sigma, T(\nu')) \in L - L/f$ s.t. $\nu_c\sigma \in M_C^+(\nu'(\sigma, T(\nu')))$. It follows from Lemma 1 that $\nu_c \in M_C^+(\nu')$. Then since $\nu_c \notin M_C^+(L - L/f)$ together with $\nu' \in L$ implies $\nu' \in L/f$, exists $H := \{\nu'\} \subseteq L/f$ s.t. $\forall \nu' \in H$, $\nu'(\sigma, T(\nu')) \in L - L/f$ and $\nu_c \in M_C^+(H)$. Therefore $M_C^+(\overline{\nu}(\epsilon, t)) \subseteq M_C^+(H) \cup [M_C^+(L - L/f)](\Sigma^\tau)^*$. Then from (L, M_C)-compatibility of f, we have $\sigma \notin f(\overline{\nu}(\epsilon, t))$, which contradicts the assumption $\overline{\nu}(\sigma, t) \in L/f$.

Next we claim that given a (L, Σ_u, M_C)-compatible control policy f, S is $(pr(M_C(L)), \Sigma_u)$-compatible. Pick $\nu_c \in L(S)$, $\sigma \in \Sigma_u$ s.t. $\nu_c\sigma \in pr(M_C(L))$. Suppose for contradiction that $\nu_c\sigma \notin L(S)$. Then $\exists \nu(\sigma, T(\nu)) \in L$ s.t. $\nu_c\sigma$ or $\nu_c\sigma\tau \in M_C(\nu(\sigma, T(\nu)))$, which implies $\nu_c \in M_C^+(\nu)$. Therefore $\nu \in \Pi_\Sigma(L(G\|S\|C)) = L/f$. It follows from (L, Σ_u)-compatibility of f that $\nu(\sigma, T(\nu)) \in L/f$. That is, $\nu_c\sigma \in pr(M_C(L/f))$. Then from $\nu_c\sigma \notin L(S)$, we have $\nu_c\sigma \in [M_C^+(L - L/f)](\Sigma^\tau)^*$. Note $\nu_c \in L(S)$, it must be the case that $\nu_c\sigma \in M_C^+(L - L/f)$. Thereby $\exists \nu'(\sigma, T(\nu')) \in L - L/f$ s.t. $\nu_c\sigma \in M_C^+(\nu'(\sigma, T(\nu')))$. From Lemma 1, we have $\nu_c \in M_C^+(\nu')$. Then since f is (L, Σ_u)-compatible and $\nu_c \notin M_C^+(L - L/f)$ together with $\nu' \in L$ implies $\nu' \in L/f$, we have $\nu'(\sigma, T(\nu')) \in L/f$. A contradiction arrives.

Finally, since $\Pi_\Sigma(L(G\|S\|C)) = L/f$ and $\Pi_\Sigma(L_m(G\|S\|C)) = L_m \cap \Pi_\Sigma(L(G\|S\|C)$, we can claim that $\Pi_\Sigma(L_m(G\|S\|C)) = L_m/f$. ■

We next show that the control exercised by a digitalized-automaton can be characterized as a compatible control policy.

Lemma 3 Let L and L_m be the generated and marked timed-languages of a nonforcing plant G, Σ_u be a set of uncontrollable events, and M_C be the timing-mask associated with a digital-clock C.

- Given a $\{\tau\}$-compatible untimed-automaton $S = (Q, \Sigma^\tau, \Upsilon, Q_0, Q)$, there exists a (L, M_C)-compatible nonforcing control policy f such that $L/f = \Pi_\Sigma(L(G\|S\|C))$; $L_m/f = \Pi_\Sigma(L_m(G\|S\|C))$.
- Given a $(pr(M_C(L)), \Sigma_u)$-compatible and $\{\tau\}$-compatible untimed-automaton $S = (Q, \Sigma^\tau, \Upsilon, Q_0, Q)$, there exists a (L, Σ_u, M_C)-compatible nonforcing control policy f such that $L/f = \Pi_\Sigma(L(G\|S\|C))$; $L_m/f = \Pi_\Sigma(L_m(G\|S\|C))$.

Proof: Define a nonforcing control policy f: $\forall \nu \in \mathcal{T}$, $f(\nu) := \bigcup_{\nu_c \in M_C^+(\nu)} \{\sigma \in \Sigma \mid \nu_c\sigma \in L(S)\} \cup \{\epsilon\}$. We claim that given a $\{\tau\}$-compatible untimed-automaton S, f is (L, M_C)-compatible and $L/f = \Pi_\Sigma(L(G\|S\|C))$.

To show $L/f = \Pi_\Sigma(L(G\|S\|C))$, we first show $\nu \in \Pi_\Sigma(L(G\|S\|C))$ implies $\nu \in L/f$ by induction on the length of ν. The base step trivially holds since $\epsilon \in \Pi_\Sigma(L(G\|S\|C)) \cap L/f$.

For the induction step, pick $\nu = \overline{\nu}(\sigma, t) \in \Pi_\Sigma(L(G\|S\|C))$. Then $\exists \overline{\nu}_c \in M_C^+(\overline{\nu}(\epsilon, t))$ s.t. $\overline{\nu}_c\sigma \in L(S)$, i.e., $\sigma \in f(\overline{\nu}(\epsilon, t))$. This together with $\overline{\nu}(\epsilon, t) \in L/f$ implies $\nu \in L/f$.

We next show $\nu \in L/f$ implies $\nu \in \Pi_\Sigma(L(G\|S\|C))$ by induction on the length of ν. The base step trivially holds since $\epsilon \in \Pi_\Sigma(L(G\|S\|C)) \cap L/f$. For the induction step, pick $\nu = \overline{\nu}(\sigma, t) \in L/f$. For $\sigma = \epsilon$, from induction hypothesis $\overline{\nu} \in \Pi_\Sigma(L(G\|S\|C))$, exists $\overline{\nu}_c \in M_C(\overline{\nu})$ s.t. $\overline{\nu}_c \in L(S)$, and further $\overline{\nu}_c\tau^* \in L(S)$. Thereby $\overline{\nu}(\epsilon, t) \in \Pi_\Sigma(L(G\|S\|C))$. For $\sigma \in \Sigma$, since $\exists \overline{\nu}_c \in M_C^+(\overline{\nu}(\epsilon, t))$ s.t. $\overline{\nu}_c\sigma \in L(S)$, we have $\nu \in \Pi_\Sigma(L(G\|S\|C))$.

To show f is (L, M_C)-compatible, pick $\nu \in L/f$, $\sigma \in \Sigma$ s.t. $\nu(\sigma, T(\nu)) \in L$. Suppose $\exists H \subseteq L/f$ s.t. $\forall \nu' \in H$, $\nu'(\sigma, T(\nu')) \in L - L/f$ and $M_C^+(\nu) \subseteq M_C^+(H) \cup [M_C^+(L - L/f)](\Sigma^\tau)^*$. $\forall \nu_c \in M_C^+(\nu)$, if $\nu_c \in [M_C^+(L - L/f)](\Sigma^\tau)^*$, then $\exists \overline{\nu}_c \in pr(\nu_c)$ s.t. $\overline{\nu}_c \in M_C^+(L - L/f)$. Thereby $\exists \nu' \in L - L/f$ s.t. $\overline{\nu}_c \in M_C^+(\nu')$. Then since $\nu' \notin L/f = \Pi_\Sigma(L(G\|S\|C))$ implies $\overline{\nu}_c \notin L(S)$, we have $\nu_c\sigma \notin L(S)$. On the other hand, if $\nu_c \in M_C^+(H)$, then $\exists \nu' \in H$ s.t. $\nu' \in L/f$, $\nu'(\sigma, T(\nu')) \in L - L/f$ and $\nu_c \in M_C^+(\nu')$. From $\nu'(\sigma, T(\nu')) \notin L/f = \Pi_\Sigma(L(G\|S\|C))$, we have $\nu_c\sigma \notin L(S)$. Therefore $\sigma \notin f(\nu)$.

Next we claim that given a $(pr(M_C(L)), \Sigma_u)$-compatible and $\{\tau\}$-compatible untimed-automaton S, f is (L, Σ_u)-compatible.

Pick $\nu \in L/f$, $\sigma \in \Sigma_u$ s.t. $\nu(\sigma, T(\nu)) \in L$. From $L/f = \Pi_\Sigma(L(G\|S\|C))$, $\exists \nu_c \in M_C(\nu)$ s.t. $\nu_c \in L(S)$. Then since $\nu_c\sigma \in M_C(\nu(\sigma, T(\nu))) \subseteq pr(M_C(L))$ and S is $(pr(M_C(L)), \Sigma_u)$-compatible, we have $\nu_c\sigma \in L(S)$. This further implies $\nu(\sigma, T(\nu)) \in \Pi_\Sigma(L(G\|S\|C)) = L/f$.

Finally, since $L/f = \Pi_\Sigma(L(G\|S\|C))$ and $\Pi_\Sigma(L_m(G\|S\|C)) = L_m \cap \Pi_\Sigma(L(G\|S\|C))$, we have $L_m/f = \Pi_\Sigma(L_m(G\|S\|C))$. ■

Combining the results of Lemma 2 and 3, the following theorem shows that a (L, Σ_u, M_C)-compatible control policy can be equivalently represented by a $(pr(M_C(L)), \Sigma_u)$-compatible and $\{\tau\}$-compatible digitalized-automaton.

Theorem 3 Let L and L_m be the generated and marked timed-languages of a nonforcing plant G, Σ_u be a set of uncontrollable events, and M_C be the timing-mask associated with a digital-clock C.

- Given a (L, Σ_u, M_C)-compatible nonforcing control policy f, there exists a $(pr(M_C(L)), \Sigma_u)$-

compatible and $\{\tau\}$-compatible untimed-automaton S over Σ^τ such that $\Pi_\Sigma(L(G\|S\|C)) = L/f$; $\Pi_\Sigma(L_m(G\|S\|C)) = L_m/f$.

- Given a $(pr(M_C(L)), \Sigma_u)$-compatible and $\{\tau\}$-compatible untimed-automaton S over Σ^τ, there exists a (L, Σ_u, M_C)-compatible nonforcing control policy f such that $L/f = \Pi_\Sigma(L(G\|S\|C))$; $L_m/f = \Pi_\Sigma(L_m(G\|S\|C))$.

4.3 Existence of Compatible Control Policy

In this section, we introduce the properties of (L, Σ_u)-controllability and (L, M_C)-observability, and show that they serve as a necessary and sufficient condition for the existence of a compatible control policy that enforces a given real-time specification on a dense-time discrete event system, using digital-clocks to measure event occurrence times. Similar to the controllability of the untimed setting, (L, Σ_u)-controllability of a timed specification language requires that the execution of a feasible uncontrollable event following a legal timed-trace must also be legal. (L, M_C)-observability, on the other hand, is very different from the observability property of the untimed setting owing to the facts that (i) a timing-mask is nondeterministic, and (ii) an event can occur simultaneously with a tick. (L, M_C)-observability requires that if σ is feasible following ν, and if each extended-observation of ν is either an extended-observation of a timed-trace where σ is illegal or possesses a prefix which is also an extended-observation of an illegal timed-trace, then σ must be illegal after ν. (This is analogous to the (L, M_C)-compatibility property introduced earlier.)

Definition 6 Let L be the generated timed-language of a plant, $K \subseteq L$ be a timed specification language, Σ_u be the set of uncontrollable events, and M_C be the timing-mask associated with a digital-clock C. K is said to be

- (L, Σ_u)-*controllable*, if $\forall \nu \in pr(K)$, $\sigma \in \Sigma_u$ such that $\nu(\sigma, T(\nu)) \in L$: $\nu(\sigma, T(\nu)) \in pr(K)$.
- (L, M_C)-*observable* if $\forall \nu \in pr(K)$, $\sigma \in \Sigma$, such that $\nu(\sigma, T(\nu)) \in L$: if exists $H \subseteq pr(K)$ s.t. $\forall \nu' \in H, \nu'(\sigma, T(\nu')) \in L - pr(K)$ and $M_C^+(\nu) \subseteq M_C^+(H) \cup [M_C^+(L - pr(K))](\Sigma^\tau)^*$, then $\nu(\sigma, T(\nu)) \in L - pr(K)$.

Note that K is controllable (resp., observable) if and only if $pr(K)$ is controllable (resp., observable). In the following, we show that controllability together with timing-mask observability serves as a necessary and sufficient condition for the existence of a compatible control policy.

Theorem 4 Let L and L_m be the generated and marked timed-languages of a nonforcing plant G, $K \subseteq L$ be a timed specification language, $\Sigma_u \subseteq \Sigma$ be a set of uncontrollable events, and M_C be the timing-mask associated with a digital-clock C.

- There exists a (L, Σ_u, M_C)-compatible nonforcing control policy f such that $L/f = K$ if and only if $\emptyset \neq K = pr(K) \subseteq L$, K is nonforcing, (L, Σ_u)-controllable and (L, M_C)-observable.

- There exists a (L, Σ_u, M_C)-compatible, nonblocking, and nonforcing control policy f such that $L_m/f = K$ if and only if $\emptyset \neq K = pr(K) \cap L_m$, K is nonforcing, (L, Σ_u)-controllable and (L, M_C)-observable.

Proof: We start by proving the first assertion. To show the sufficiency, define a nonforcing control policy f as follows: $\forall \nu \in \mathcal{T}$, $f(\nu) := \bigcup_{\nu_c \in M_C^+(\nu)} \{\sigma \mid \nu_c\sigma \in K_C := pr(M_C(K)) - [M_C^+(L-K)](\Sigma^\tau)^*\}$. Following the proof of Proposition 3, K_C is prefix-closed, nonempty and $\forall \nu_c \in K_C$, $\nu_c\tau^* \in K_C$.

To show $L/f = K$, we first show that $\nu \in L/f$ implies $\nu \in K$ by induction on the length of ν. $\epsilon \in K \cap L/f$, establishing the base step. For the induction step, pick $\nu = \overline{\nu}(\sigma, t) \in L/f$. For $\sigma \in \overline{\Sigma}_u$, from nonforcing-ness and (L, Σ_u)-controllability of K, we have $\nu \in K$. For $\sigma \in \Sigma - \Sigma_u$, suppose for contradiction that $\nu \notin K$, i.e., $\nu \in L - K$. Then $\forall \nu_c \in M_C^+(\overline{\nu}(\epsilon, t))$, $\nu_c\sigma \in M_C^+(\overline{\nu}(\sigma, t))$ i.e., $\nu_c\sigma \in [M_C^+(L-K)](\Sigma^\tau)^*$. This implies $\nu_c\sigma \notin K_C$. Thereby $\sigma \notin f(\overline{\nu}(\epsilon, t))$, i.e., $\overline{\nu}(\sigma, t) \notin L/f$, which yields a contradiction.

We next show that $\nu \in K$ implies $\nu \in L/f$ by induction on the length of ν. $\epsilon \in K \cap L/f$, establishing the base step. For induction step, pick $\nu = \overline{\nu}(\sigma, t) \in K$. For $\sigma = \epsilon$, from nonforcing-ness of f, we have $\overline{\nu}(\epsilon, t) \in L/f$. For $\sigma \in \Sigma$, suppose for contradiction that $\nu \notin L/f$, i.e., $\nu \in L - L/f$. Then $\forall \nu_c \in M_C^+(\overline{\nu}(\epsilon, t))$, $\nu_c\sigma \in M_C^+(L - L/f)$, i.e., $\nu_c\sigma \notin K_C$.

Further since $\nu \in K$ together with $\nu_c\sigma$ or $\nu_c\sigma\tau \in M_C(\overline{\nu}(\sigma,t))$ implies $\nu_c\sigma \in pr(M_C(K))$, we have $\nu_c\sigma \in [M_C^+(L-K)](\Sigma^\tau)^*$. We need to analyze two cases: (i) $\exists\overline{\nu}_c \in pr(\nu_c)$ s.t. $\overline{\nu}_c \in M_C^+(L-K)$ and (ii) $\nu_c\sigma \in M_C^+(L-K)$. Following the proof of Lemma 2, we have either $\nu_c \in [M_C^+(L-K)](\Sigma^\tau)^*$ or exists $H := \{\nu'\} \subseteq K$ s.t. $\forall\nu' \in H$, $\nu'(\sigma,T(\nu')) \in L-K$ and $\nu_c \in M_C^+(H)$. From (L,M_C)-observability and prefix-closure of K, we have $\nu \notin K$, which yields a contradiction.

Then we show that f is (L,Σ_u,M_C)-compatible. To show (L,Σ_u)-compatibility of f, pick $\nu \in L/f$, $\sigma \in \Sigma_u$ s.t. $\nu(\sigma,t) \in L$. Since K is controllable and prefix-closed, $\nu(\sigma,t) \in pr(K) = L/f$. To show (L,M_C)-compatibility of f, pick $\nu \in L/f$, $\sigma \in \Sigma$ s.t. $\nu(\sigma,T(\nu)) \in L$. Suppose exists $H \subseteq L/f$ s.t. $\forall\nu' \in H$, $\nu'(\sigma,T(\nu')) \in L-L/f$ and $M_C^+(\nu) \subseteq M_C^+(H) \cup [M_C^+(L-L/f)](\Sigma^\tau)^*$. Then since K is (L,M_C)-observable and prefix-closed, $\nu(\sigma,T(\nu)) \in L-pr(K) = L-L/f$, i.e., $\sigma \notin f(\nu)$.

To show the necessity, since L/f is a generated language and $L/f = K$, we have $\emptyset \neq K = pr(K) \subseteq L$. Further since f is nonforcing, so is K. To show (L,Σ_u)-controllability of K, pick $\nu \in pr(K)$, $\sigma \in \Sigma_u$ s.t. $\nu(\sigma,T(\nu)) \in L$. Since $L/f = K = pr(K)$ and f is Σ_u-compatible, $\sigma \in f(\nu(\epsilon,t))$, i.e., $\nu(\sigma,T(\nu)) \in L/f = pr(K)$. To show (L,M_C)-observability of K, pick $\nu \in pr(K)$, $\sigma \in \Sigma$ s.t. $\nu(\sigma,T(\nu)) \in L$. Suppose exists $H \subseteq pr(K)$ s.t. $\forall\nu' \in H$, $\nu'(\sigma,T(\nu')) \in L-pr(K)$ and $M_C^+(\nu) \subseteq M_C^+(H) \cup [M_C^+(L-pr(K))](\Sigma^\tau)^*$. Then since $L/f = K = pr(K)$ and f is (L,M_C)-compatible, $\sigma \notin f(\nu)$, i.e., $\nu(\sigma,T(\nu)) \notin L/f = pr(K)$.

We next prove the second assertion. We first show the sufficiency. Since $\emptyset \neq K = pr(K) \cap L_m$ and $pr(L_m) = L$, we have $\emptyset \neq pr(K) \subseteq L$. Further since K is (L,Σ_u)-controllable and (L,M_C)-observable, so is $pr(K)$. Then from the first assertion, there exists a compatible nonforcing control policy f s.t. $L/f = pr(K)$. Thereby $L_m/f = L_m \cap L/f = L_m \cap pr(K) = K$.

We next show the necessity. Since $L_m/f = K$, we have $\emptyset \neq K$. Further since f is nonblocking, we have $L/f = pr(K)$. Then from the first assertion, K is (L,Σ_u)-controllable and (L,M_C)-observable. Furthermore, $K = L_m/f = L_m \cap L/f = L_m \cap pr(K)$. ■

Based on the results of Theorem 3 and 4, the following corollary provides a necessary and sufficient condition for the existence of a digitalized-automaton based (nonblocking) supervisor.

(A supervisor S is said to be *nonblocking* for a plant G if $L(G\|S) = pr(L_m(G\|S))$.)

Corollary 1 Let L and L_m be the generated and marked timed-languages of a nonforcing plant G, $K \subseteq L$ be a nonforcing timed specification language, Σ_u be a set of uncontrollable events, and M_C be the timing-mask associated with a digital-clock C.

- There exists a $\{\tau\}$-compatible and $(pr(M_C(L), \Sigma_u)$-compatible supervisor S over Σ^τ such that $\Pi_\Sigma(L(G\|S\|C)) = K$ if and only if $\emptyset \neq pr(K) = K \subseteq L$, K is nonforcing, (L, Σ_u)-controllable and (L, M_C)-observable.

- There exists a $\{\tau\}$-compatible and $(pr(M_C(L), \Sigma_u)$-compatible nonblocking supervisor S over Σ^τ such that $\Pi_\Sigma(L_m(G\|S\|C)) = K$ if and only if $\emptyset \neq K = pr(K) \cap L_m$, K is nonforcing, (L, Σ_u)-controllable and (L, M_C)-observable.

4.4 Verification of Existence Condition

From Theorem 4, to determine the existence of a compatible (nonblocking) control policy for a given timed specification language K, we need to check the properties of (L, Σ_u)-controllability and (L, M_C)-observability of K, or equivalently of $pr(K)$. We assume that $pr(K)$ can be generated by a deterministic timed-automaton $R = (Q^R, \Sigma, \Xi^R, \Upsilon^R, I^R, Q_0^R, Q^R)$. Note determinism is required to allow its complementation. (In general timed-automata languages are not closed under complementation [6].) In order to capture any violation of $pr(K)$, we need to complete R by introducing a dump state and certain transitions. The completed specification $\overline{R}$ is constructed as follows. $\overline{R} = (Q_R \cup \{dump\}, \Sigma, \Xi_R, \Upsilon_R \cup \Upsilon_{add}, I_{\overline{R}}, Q_{R0}, Q_{Rm})$, where $\forall q \in Q_R$, $I_{\overline{R}}(q) = I_R(q)$, $I_{\overline{R}}(dump) = \texttt{true}$, and the set of added transitions Υ_{add} is defined as

- $\forall q \in Q_R$, $\forall \sigma \in \Sigma$, if there are $n \geq 1$ out-going transitions from q labeled with σ, and let $\{\phi_\sigma^1, \cdots, \phi_\sigma^n\}$ be the set of guard conditions associated with those n transitions, then $(q, \sigma, \neg(\vee_{i=1}^n \phi_\sigma^i), \emptyset, dump) \in \Upsilon_{add}$; otherwise $(q, \sigma, \texttt{true}, \emptyset, dump) \in \Upsilon_{add}$.

- $\forall \sigma \in \Sigma$, $(dump, \sigma, \texttt{true}, \emptyset, dump) \in \Upsilon_{add}$.

It is easy to see that $\overline{R}$ can generate any timed-trace, and since R is deterministic, any timed-trace that violates the specification $pr(K)$ has a run that reaches the dump state. (I.e., $L(\overline{R}) = \mathcal{T}$ and $L_m(\overline{R}) = \mathcal{T} - pr(K)$.)

In the following, we present an algorithm for checking the controllability and timing-mask observability properties. The algorithm first constructs the region-automaton $\mathcal{R}(G\|\overline{R}\|C)$. From the result in [54], this region-automaton generates the language $pr(M_C(L(G\|\overline{R}))) = pr(M_C(L(G)))$. Thus when a plant G executes a certain timed-trace, one of its timing-masked observation will nondeterministically be seen by a controller, and this observation can be traced in $\mathcal{R}(G\|\overline{R}\|C)$. Since a region-automaton is in general nondeterministic, the corresponding states reached in $\mathcal{R}(G\|\overline{R}\|C)$ will be non-unique. If the execution of a certain event from any such state leads to the violation of the specification (namely, results in the reaching of a state in $\mathcal{R}(G\|\overline{R}\|C)$ with second coordinate as dump), then such an event must be disabled at all such states. This idea is formalized and illustrated below, and then its correctness is formally established.

Algorithm 2 Let L be the generated timed-language of a nonforcing plant G, $K \subseteq L$ be a nonforcing timed specification language represented by a deterministic timed-automaton R, Σ_u be a set of uncontrollable events, and M_C be the timing-mask associated with a digital-clock C. An algorithm for checking the existence condition for a compatible control policy, i.e., controllability and timing-mask observability, is presented as below.

1. Construct the automaton $\overline{R}$.

2. Construct the region-automaton $\mathcal{R}(G\|\overline{R}\|C)$. Note each state of $\mathcal{R}(G\|\overline{R}\|C)$ is of the form $(q_G, q_{\overline{R}}, q_C, r)$, where q_G is a state of G, $q_{\overline{R}}$ is a state of $\overline{R}$, q_C is a state of C, and r is a clock-region.

3. Determinize $\mathcal{R}(G\|\overline{R}\|C)$. The resulting deterministic automaton $det[\mathcal{R}(G\|\overline{R}\|C)]$ is denoted as $\widehat{S}$. We call a state $q_{\widehat{S}}$ of $\widehat{S}$ to be a dump state if exists $(q_G, dump, q_C, r) \in q_{\widehat{S}}$.

4. Construct an untimed-automaton S by disabling certain transitions in $\widehat{S}$: Disable $\sigma \in \Sigma$ at $q_{\widehat{S}} \in Q^{\widehat{S}}$ if its successor state is a dump state. Mark all states in resulting automaton

S.

5. Output 'yes' if $\Pi_\Sigma(L(G\|S\|C)) = pr(K)$ (note in this case S is $(pr(M_C(L)), \Sigma_u)$-compatible); else output 'no'.

The following example illustrates Algorithm 2.

Example 6 Consider a prefix-closed timed-specification language K represented by a timed-automaton R as shown in Figure 4.2, and a plant G and a digital-clock C given in Example 5.

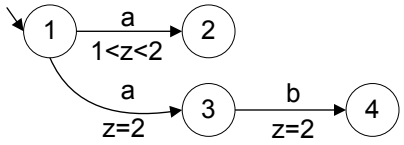

Figure 4.2 Timed-automaton of K

Suppose all events are controllable, so that K is control-compatible by default. Then we only need to check timing-mask observability of K. From the analysis of Example 5, we have each extended observation in $M_C^+((a,2)) = \{\tau a, \tau a \tau, \tau\tau a\}$ is either an extended observation of an illegal timed-trace ($\tau\tau a \in M_C^+((a, 2 < t < 3))$) or an extended observation of a timed-trace where b is disabled ($\tau a, \tau a \tau \in M_C^+((a, 1 < t < 2)(\epsilon, 2))$). On the other hand, since b is legal following $(a,2)$ (i.e., $(a,2)(b,2) \in pr(K)$), we have that K is not timing-mask observable. Let us verify this using Algorithm 2.

We first compute the region-automaton $R(G\|\overline{R}\|C)$, which is as shown in Figure 4.3. For simplicity of illustration, the discrete-state of C and clock variable z of R are omitted (since C has a single discrete-state and the value of z is the same as the value of the clock variable x of G). We next construct an untimed-automaton S by disabling the events that lead to the dump states in the determinized region-automaton $det[\mathcal{R}(G\|\overline{R}\|C)]$. The resulting automaton S is shown in Figure 4.4. Note S is control-compatible by default since all events are controllable. It can be checked that $\Pi_\Sigma(L(G\|S\|C)) = \{(a, 1 < t \leq 2)(\epsilon, t' > t), (\epsilon, t \geq 0)\} \neq pr(K)$. Then according to Algorithm 2, K is not timing-mask observable as expected.

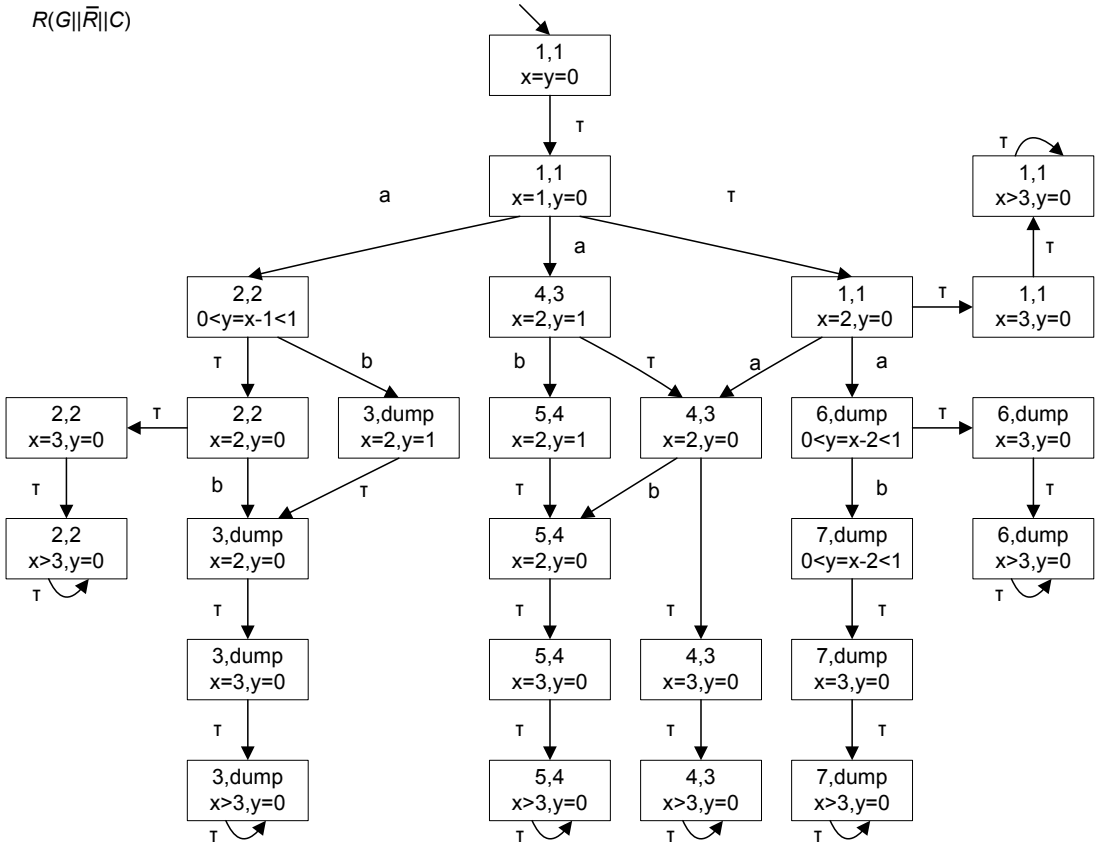

Figure 4.3 Region-automaton of $G\|\overline{R}\|C$

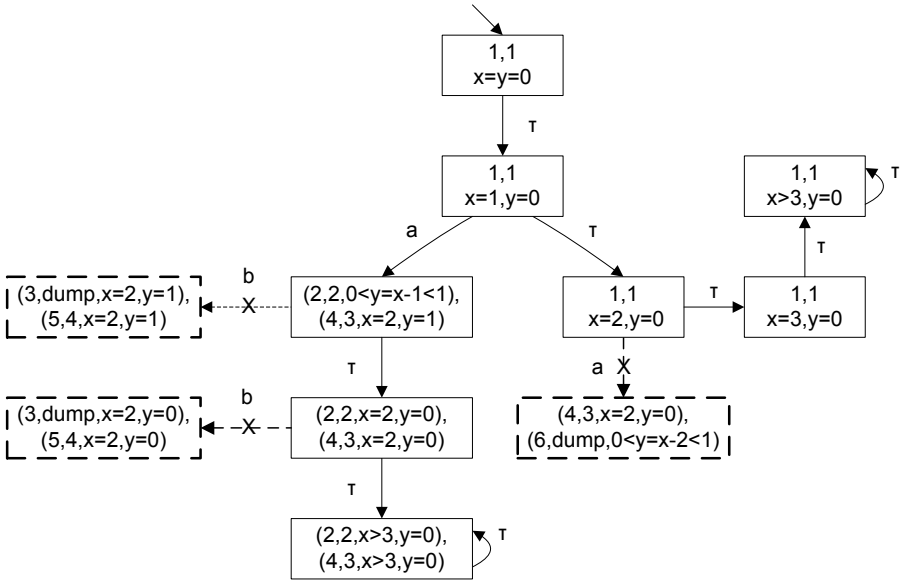

Figure 4.4 Constructed untimed-automaton S

From the construction of S and non-forcingness of L and K, the untimed-automaton S constructed by Algorithm 2 is $\{\tau\}$-compatible. The correctness of Algorithm 2 is next established.

Theorem 5 Let L be the generated timed-language of a nonforcing plant G, $\emptyset \neq K \subseteq L$ be a deterministic nonforcing timed specification language, Σ_u be a set of uncontrollable events, M_C be the timing-mask associated with a digital-clock C, and S be the untimed-automaton constructed by Algorithm 2. Then K is (L, Σ_u)-controllable and (L, M_C)-observable if and only if $\Pi_\Sigma(L(G\|S\|C)) = pr(K)$, and in which case S is $(pr(M_C(L)), \Sigma_u)$-compatible.

Proof: We first show the sufficiency. To show (L, Σ_u)-controllability of K, pick $\nu \in pr(K)$, $\sigma \in \Sigma_u$ s.t. $\nu(\sigma, T(\nu)) \in L$. Then since $\nu \in pr(K) = \Pi_\Sigma(L(G\|S\|C))$, exists $\nu_c \in M_C(\nu)$ s.t. $\nu_c \in L(S)$. Further since S is $(pr(M_C(L)), \Sigma_u)$-compatible and $\nu_c\sigma \in M_C(\nu(\sigma, T(\nu))) \subseteq pr(M_C(L))$, we have $\nu_c\sigma \in L(S)$, which implies $\nu(\sigma, T(\nu)) \in \Pi_\Sigma(L(G\|S\|C)) = pr(K)$. To show (L, M_C)-observability of K, pick $\nu \in pr(K)$, $\sigma \in \Sigma$ s.t. $\nu(\sigma, T(\nu)) \in L$. Suppose exists $H \subseteq pr(K)$ s.t. $\forall \nu' \in H$, $\nu'(\sigma, T(\nu')) \in L - pr(K)$ and $M_C^+(\nu) \subseteq M_C^+(H) \cup [M_C^+(L - pr(K))](\Sigma^\tau)^*$. $\forall \nu_c \in M_C^+(\nu)$, if $\nu_c \in M_C^+(H)$, then $\exists \nu' \in H$ s.t. $\nu_c \in M_C^+(\nu')$ and $\nu'(\sigma, T(\nu')) \in L - pr(K) \notin \Pi_\Sigma(L(G\|S\|C))$. That is, $\nu_c\sigma$. On the other hand, if $\nu_c \in [M_C^+(L - pr(K))](\Sigma^\tau)^*$, then $\exists \overline{\nu}_c \in pr(\nu_c)$ s.t. $\overline{\nu}_c \in M_C^+(L - pr(K))$, i.e., $\exists \nu' \in L - pr(K) \notin \Pi_\Sigma(L(G\|S\|C))$ s.t. $\overline{\nu}_c \in M_C^+(\nu')$. So $\overline{\nu}_c \notin L(S)$, which implies $\nu_c\sigma \notin L(S)$. Thereby $\nu(\sigma, T(\nu)) \notin \Pi_\Sigma(L(G\|S\|C)) = pr(K)$.

We next show the necessity. We first show $\nu \in \Pi_\Sigma(L(G\|S\|C))$ implies $\nu \in pr(K)$ by induction on the length of ν. $\epsilon \in \Pi_\Sigma(L(G\|S\|C)) \cap pr(K)$, establishing the base step. For the induction step, pick $\nu = \overline{\nu}(\sigma, t) \in \Pi_\Sigma(L(G\|S\|C))$. For $\sigma \in \overline{\Sigma}_u$, from nonforcing-ness and (L, Σ_u)-controllability of K, we have $\nu \in pr(K)$. For $\sigma \in \Sigma - \Sigma_u$, suppose for contradiction that $\nu \notin pr(K)$. Then $\nu \in L - pr(K)$. It follows that $\forall \nu'_c \in M_C(\nu)$, $\nu'_c \notin L(S)$ (since dump state is reached following ν'_c). Thereby $\nu \notin \Pi_\Sigma(L(G\|S\|C))$. A contradiction arrives.

Then we show that $\nu \in pr(K)$ implies $\nu \in \Pi_\Sigma(L(G\|S\|C))$ by induction on the length of ν. $\epsilon \in \Pi_\Sigma(L(G\|S\|C)) \cap pr(K)$, establishing the base step. For induction step, pick $\nu = \overline{\nu}(\sigma, t) \in pr(K)$. For $\sigma = \epsilon$, from $\{\tau\}$-compatibility of S, we have $\overline{\nu}(\epsilon, t) \in \Pi_\Sigma(L(G\|S\|C))$. For $\sigma \in \Sigma$,

suppose for contradiction that $\nu \notin \Pi_\Sigma(L(G\|S\|C))$. Then $\forall \nu_c \in M_C^+(\overline{\nu}(\epsilon,t))$, $\nu_c\sigma \notin L(S)$. We need to consider the following two cases.

Case 1: $\nu_c \in L(S)$, $\nu_c\sigma \notin L(S)$. Then exists a run $q_0^{\widehat{S}} \xrightarrow{\nu_c} (q_G, q_R, q_C, r) \xrightarrow{\sigma} (q'_G, dump, q'_C, r')$. Thereby $\nu_c \in pr(M_C(pr(K)))$ and $\nu_c\sigma \in pr(M_C(L - pr(K)))$.

If $\nu_c \in M_C(pr(K))$ and $\nu_c\sigma \in M_C(L - pr(K))$, then exists $\nu'(\sigma, T(\nu')) \in L - pr(K)$ s.t. $\nu_c\sigma \in M_C(\nu'(\sigma, T(\nu')))$. It follows $\nu_c \in M_C(\nu')$. Then from $q_0^{\widehat{S}} \xrightarrow{\nu_c} (q_G, q_R, q_C, r)$, we have $\nu' \in pr(K)$.

If $\nu_c \in M_C(pr(K))$ and $\nu_c\sigma \notin M_C(L - pr(K))$, then it must be the case that σ occurs simultaneously with τ, i.e., $\nu_c\sigma\tau \in M_C(L - pr(K))$. It follows that exists $\nu'(\sigma, T(\nu')) \in L - pr(K)$ s.t. $\nu_c\sigma\tau \in M_C(\nu'(\sigma, T(\nu')))$. This implies $\nu_c\tau \in M_C(\nu')$. Then since $q_0^{\widehat{S}} \xrightarrow{\nu_c} (q_G, q_R, q_C, r)$, we have $\nu' \in pr(K)$.

If $\nu_c \notin M_C(pr(K))$ and $\nu_c\sigma \notin M_C(L - pr(K))$, then it must be the case that the last event of ν_c occurs simultaneously with σ and τ, i.e., $\nu_c\tau \in M_C(pr(K))$ and $\nu_c\sigma\tau \in M_C(L - pr(K))$. It follows that exists $\nu'(\sigma, T(\nu')) \in L - pr(K)$ s.t. $\nu_c\sigma\tau \in M_C(\nu'(\sigma, T(\nu')))$ and $\nu_c\tau \in M_C(\nu')$. Then since $q_0^{\widehat{S}} \xrightarrow{\nu_c} (q_G, q_R, q_C, r)$, we have $\nu' \in pr(K)$.

On the other hand, if $\nu_c \notin M_C(pr(K))$ and $\nu_c\sigma \in M_C(L - pr(K))$, then it must be the case that the very last tick of ν_c is still pending. Thereby $\nu_c\sigma \notin M_C(L - pr(K))$, whereas $\nu_c\sigma\tau \in M_C(L - pr(K))$. So this is not a possible scenario.

Case 2: $\nu_c \notin L(S)$. Then since $\nu \in pr(K)$ implies $\nu_c \in L(\widehat{S})$, it must be the case that certain transition of ν_c is disabled. Thereby $\exists \overline{\nu}_c\overline{\sigma} \in pr(\nu_c)$, $\overline{\sigma} \in \Sigma$ s.t. $\overline{\nu}_c \in L(S)$ and $\overline{\nu}_c\overline{\sigma} \notin L(S)$. It follows that exists a run $q_0^{\widehat{S}} \xrightarrow{\overline{\nu}_c} (q_G, q_R, q_C, r) \xrightarrow{\overline{\sigma}} (q'_G, dump, q'_C, r')$. This implies $\overline{\nu}_c\overline{\sigma} \in pr(M_C(L - pr(K)))$. Thereby $\exists \nu''(\overline{\sigma}, T(\nu'')) \in L - pr(K)$ s.t. $\overline{\nu}_c\overline{\sigma}$ or $\overline{\nu}_c\overline{\sigma}\tau \in M_C(\nu''(\overline{\sigma}, T(\nu'')))$, i.e., $\overline{\nu}_c\overline{\sigma} \in M_C^+(\nu''(\overline{\sigma}, T(\nu'')))$. This implies $\nu_c \in [M_C^+(L - pr(K))](\Sigma^\tau)^*$.

Therefore we have $M_C^+(\overline{\nu}(\epsilon,t)) \subseteq M_C^+(H) \cup [M_C^+(L - pr(K))](\Sigma^\tau)^*$, where $H := \{\nu'\} \subseteq pr(K)$ s.t. $\forall \nu' \in H$, $\nu'(\sigma, T(\nu')) \in L - pr(K)$. Then since K is (L, M_C)-observable, $\nu \notin pr(K)$, which yields a contraction.

Finally, we show that S is $(pr(M_C(L)), \Sigma_u)$-compatible. Pick $\nu_c \in L(S)$, $\sigma \in \Sigma_u$ s.t. $\nu_c\sigma \in pr(M_C(L))$. Suppose for contradiction that $\nu_c\sigma \notin L(S)$. Then since certain dump

state is reached by $\nu_c\sigma$, exists $\nu(\sigma, T(\nu)) \in L - pr(K)$ s.t. $\nu_c\sigma \in pr(M_C(\nu(\sigma, T(\nu))))$. It follows $\nu_c \in M_C^+(\nu)$, which further implies $\nu \in \Pi_\Sigma(L(G\|S\|C)) = pr(K)$. Since K is (L, Σ_u)-controllable, $\nu(\sigma, T(\nu)) \in pr(K)$. A contradiction arrives. ■

Remark 5 Theorem 4 additionally requires checking the nonemptiness, and prefix or relative closure of K (which requires a language containment check). A language containment check can be reduced to an emptiness check (under the determinism of the timed-automaton of the larger language), and the emptiness check is well known for timed-automata languages [6].

When the proposed existence condition of Corollary 1 is satisfied, Algorithm 2 presents a method to compute a compatible digitalized-automaton based (nonblocking) supervisor. This is illustrated by the following example.

Example 7 Consider a prefix-closed specification K' as shown in Figure 4.5, and a plant G and a digital-clock C given in Example 5. Suppose a is controllable and b is uncontrollable. It can be checked using Algorithm 2 that K' is controllable and timing-mask observable. The untimed-automaton S constructed by Algorithm 2 is shown in Figure 4.5. It serves as a $\{\tau\}$-compatible and $(pr(M_C(L), \Sigma_u)$-compatible supervisor that satisfies $\Pi_\Sigma(L(G\|S\|C)) = K'$.

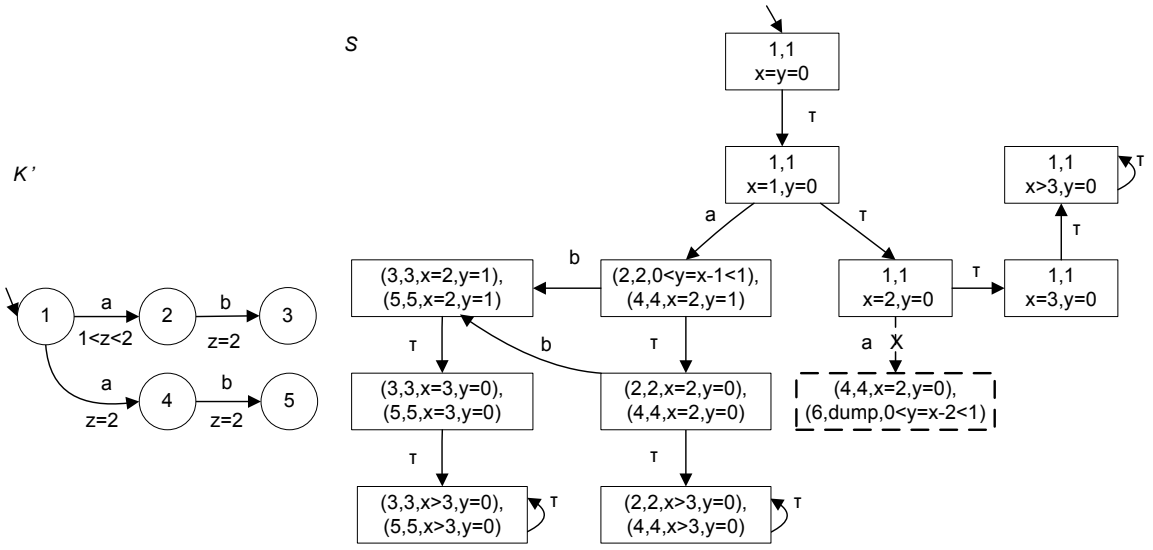

Figure 4.5 Timed-automaton of K' (left) and supervisor S (right)

4.5 Properties of Timing-Masked Observability

In this section, we examine the lattice structure of a class of (L, M_C)-observable languages. We show below that timing-mask observability is not closed under intersection but closed under union. In order to show this property, we first show in the following lemma that if a trace is accepted by the prefix language $pr(K)$, then it must possess certain untimed observation(s) not shared by an illegal trace of $L - pr(K)$.

Lemma 4 Let L be the generated timed-language of nonforcing plant G, M_C be the timing-mask associated with a digital-clock C, and $K \subseteq L$ be a (L, M_C)-observable timed language. Then for any $\nu \in pr(K)$, it holds that $M_C^+(\nu) \subsetneq [M_C^+(L - pr(K))](\Sigma^\tau)^*$.

Proof: We show this by contradiction. Pick $\nu = (\sigma_1, t_1)\cdots(\sigma_{n+1}, t_{n+1}) \in pr(K)$. Suppose $M_C^+(\nu) \subseteq [M_C^+(L - pr(K))](\Sigma^\tau)^*$. For any $\nu_c = \tau^{k_1}\sigma_1 \cdots \tau^{k_{n+1}}\sigma_{n+1}\tau^b \in M_C^+(\nu)$, we need consider the following two cases.

Case 1: $\exists \overline{\nu}_c = \tau^{k_1}\sigma_1 \cdots \tau^{k_j}\sigma_j \in pr(\nu_c)$ with $j \leq n$ s.t. $\overline{\nu}_c \in [M_C^+(L - pr(K))](\Sigma^\tau)^*$. This implies $\tau^{k_1}\sigma_1 \cdots \tau^{k_n}\sigma_n\tau^{k_{n+1}}\tau^b \in [M_C^+(L - pr(K))](\Sigma^\tau)^*$.

Case 2: $\forall \overline{\nu}_c = \tau^{k_1}\sigma_1 \cdots \tau^{k_j}\sigma_j \in pr(\nu_c)$ with $j \leq n$, $\overline{\nu}_c \notin M_C^+(L - pr(K))(\Sigma^\tau)^*$. It can be concluded that either $\overline{\nu}_c = \tau^{k_1}\sigma_1 \cdots \tau^{k_{n+1}}\sigma_{n+1} \in M_C^+(L - pr(K))$ or $\nu_c \in M_C^+(L - pr(K))$. Then $\exists \mu = (\sigma_1, t'_1)\cdots(\sigma_{n+1}, t'_{n+1}) \in L - pr(K)$ s.t. $\overline{\nu}_c \in M_C^+(\mu)$ or $\nu_c \in M_C^+(\mu)$. And so exists $j \leq n$ s.t. $\overline{\mu} = (\sigma_1, t'_1)\cdots(\sigma_j, t'_j) \in pr(K)$, and $\overline{\mu}(\sigma_{j+1}, t'_{j+1}) \in L - pr(K)$. Suppose $j \leq n-1$. Note $\overline{\nu}'_c = \tau^{k_1}\sigma_1 \cdots \tau^{k_{j+1}}\sigma_{j+1} \in M_C^+(\overline{\mu}(\sigma_{j+1}, t'_{j+1}))$. We have $\overline{\nu}'_c \in [M_C^+(L - pr(K))]$. This conflicts with the condition of Case 2. Thus it must hold that $j = n$. This further implies $\overline{\mu}' = \overline{\mu}(\epsilon, t'_{n+1}) \in pr(K)$. Let $H = \{\overline{\mu}'\}$. We have $H \subseteq pr(K)$ and $\forall \overline{\mu}' \in H$, $\overline{\mu}'(\sigma_{n+1}, t'_{n+1}) \in L - pr(K)$. Then if $\overline{\nu} \in M_C^+(\mu)$, we have $\tau^{k_1}\sigma_1 \cdots \tau^{k_{n+1}} \in M_C^+(H)$ and $\tau^{k_1}\sigma_1 \cdots \tau^{k_{n+1}}\tau \in [M_C^+(L - pr(K))](\Sigma^\tau)^*$. If $\nu_c \in M_C^+(\mu)$, we have $\tau^{k_1}\sigma_1 \cdots \tau^{k_{n+1}}\tau^b \in M_C^+(H)$.

Then for $\overline{\nu} = (\sigma_1, t_1)\cdots(\epsilon, t_{n+1}) \in pr(K)$, $\overline{\nu}(\sigma_{n+1}, t_{n+1}) \in L$: $\forall \overline{\nu}_c = \tau^{k_1}\sigma_1 \cdots \tau^{k_{n+1}}\tau^b \in M_C^+(\overline{\nu})$, we have $\overline{\nu}_c \in M_C^+(H) \cup [M_C^+(L - pr(K))](\Sigma^\tau)^*$. Note K is (L, M_C)-observability. We have $\nu \in L - pr(K)$, which yields a contradiction. ∎

With Lemma 4 in hand, we are ready to establish the following theorem.

Theorem 6 Let L be the generated timed-language of nonforcing plant G, M_C be the timing-mask associated with a digital-clock C, and $K_i \subseteq L$ be (L, M_C)-observable timed languages.

1. In general $\bigcap K_i$ is not (L, M_C)-observable.

2. $\bigcup K_i$ is (L, M_C)-observable.

Proof: To see the first assertion, we consider the G, K_1, and K_2 as shown in Figure 4.6. We have $pr(K_1 \cap K_2) = \{(\epsilon, t), (a, 1)(\epsilon, t')\}$ with $t \geq 0$ and $t' \geq 1$, and $L - pr(K_1 \cap K_2) = \{(a, t_1 < 1)(\epsilon, t_1'), (a, < 1 < t_2 < 2)(\epsilon, t_2')\}$ with $t_i' \geq t_i$ for $i = 1, 2$. It can be checked that $K_1 \cap K_2$ is not (L, M_C)-observable. This is because for $\nu = (\epsilon, 1) \in pr(K_1 \cap K_2)$, exists $H = \{(\epsilon, 0.5), (\epsilon, 1.5)\} \subseteq pr(K_1 \cap K_2)$ s.t. $\forall \mu \in H$, $\mu(a, T(\mu)) \in L - pr(K_1 \cap K_2)$. Note $M_C^+(\nu) = \{\epsilon, \tau\} = M_C^+(H)$ and $[M_C^+(L - pr(K_1 \cap K_2))](\Sigma^\tau)^* = \{a(\Sigma^\tau)^*, \tau a(\Sigma^\tau)^*\}$. We have $M_C^+(\nu) \subseteq M_C^+(H) \cup [M_C^+(L - pr(K_1 \cap K_2))](\Sigma^\tau)^*$. Note (L, M_C)-observability of $pr(K_1 \cap K_2)$ will require $(a, 1) \in L - pr(K_1 \cap K_2)$, leading to a contradiction. And so we have $pr(K_1 \cap K_2)$ is not (L, M_C)-observable.

Next we show the second assertion. Pick $\nu \in pr(\bigcup_i K_i)$ and $\sigma \in \Sigma$ s.t. $\nu(\sigma, T(\nu)) \in L$. Suppose exists $H \subseteq pr(\bigcup_i K_i)$ s.t. $\forall \nu' \in H$, $\nu'(\sigma, T(\nu')) \in L - pr(\bigcup_i K_i)$, and $M_C^+(\nu) \subseteq M_C^+(H) \cup [M_C^+(L - pr(\bigcup_i K_i))](\Sigma^\tau)^*$. We need show $\nu(\sigma, T(\nu)) \in L - pr(\bigcup_i K_i)$. This can be proved by contradiction. Suppose there exists i s.t. $\nu(\sigma, T(\nu)) \in pr(K_i)$. Note $pr(\bigcup_i K_i) = \bigcup_i pr(K_i)$, i.e. prefix-closure is closed under union, and $[M_C^+(L - pr(\bigcup_i K_i))](\Sigma^\tau)^* \subseteq [M_C^+(L - pr(K_i))](\Sigma^\tau)^*$. We need consider the following cases.

Case 1: $H \subseteq pr(K_i)$. Then from (L, M_C)-observability of K_i, we have $\nu(\sigma, T(\nu)) \in L - pr(K_i)$.

Case 2: $H = H_1 \cup H_2$ with $H_1 \subseteq pr(K_i)$ and $H_2 \subseteq pr(\bigcup_j K_j) - pr(K_i) \subseteq L - pr(K_i)$. Note $M_C^+(H) \subseteq M_C^+(H_1) \cup M_C^+(L - pr(K_i))$. We have $M_C^+(\nu) \subseteq M_C^+(H_1) \cup [M_C^+(L - pr(K_i))](\Sigma^\tau)^*$. Then from (L, M_C)-observability of K_i, we have $\nu(\sigma, T(\nu)) \in L - pr(K_i)$.

Case 3: $H \subseteq pr(\bigcup_j K_j) - pr(K_i)$. Note $M_C^+(H) \subseteq M_C^+(L - pr(K_i))$. We have $M_C^+(\nu) \subseteq [M_C^+(L - pr(K_i))](\Sigma^\tau)^*$. However from Lemma 4, we have $M_C^+(\nu) \subsetneq [M_C^+(L - pr(K_i))](\Sigma^\tau)^*$.

From the above analysis, we know a contradiction arrives at each case. Therefore $\nu(\sigma, T(\nu)) \in L - pr(K_i)$ holds for any i, equivalently $\nu(\sigma, T(\nu)) \in L - pr(\bigcup_i K_i)$ as desired. ■

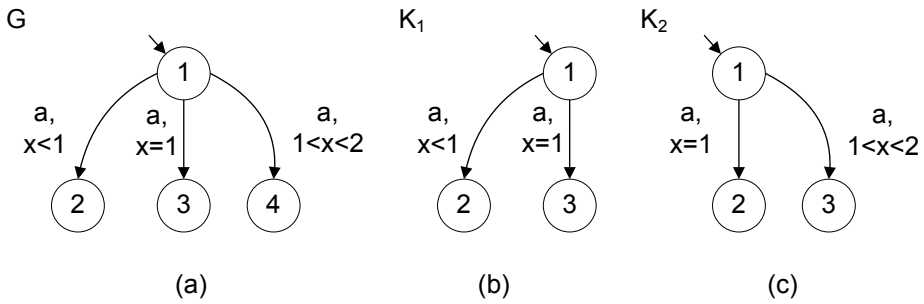

Figure 4.6 Timed-automata of G, K_1 and K_2

Theorem 6 shows that timing-mask observability is preserved under union but generally not preserved under intersection. Note prefix-closure as well as controllability are closed under union. It can be concluded that there exists a supremal closed, controllable and observable sublanguage for a given specification, however generally there does not exist an infimal controllable and observable superlanguage.

4.6 Conclusion

We studied the supervisory control of a dense-time discrete-event plant subject to a real-time specification, where the event occurrence times are observed by a finite-precision digital-clock. This is a realistic scenario compared to the earlier works which assumed that time can be measured precisely. In this work, (i) We provided a representation of a compatible control policy, that relies on a finite-precision measurement of time, in form of a digitalized-automaton in which time evolves discretely (in accordance with the available digital-clock). (ii) We introduced the notion of observability with respect to a timing-mask and showed that this property together with controllability serves as a necessary and sufficient condition for the existence of a compatible control policy (supervisor) enforcing a real-time specification on a dense-time discrete event plant. The observability condition presented in the paper is very different from the one arising due to a partial observation of events since a partial observation of time is generally nondeterministic. (iii) We presented a method to verify the proposed observability and controllability conditions, and an algorithm to compute a compatible supervisor when such conditions are satisfied. Further we examined the lattice structure of a class of

timing-mask observable languages, and showed that timing-mask observability is not preserved under intersection but preserved under union.

CHAPTER 5. DIAGNOSIS OF DENSE-TIME DESs USING DIGITAL-CLOCKS

Diagnosis is needed to detect the occurrence of a fault so as to enable any corrective actions. For event-driven systems with timing-requirements, diagnosis involves detecting the timing-faults, besides the sequence-faults. This requires monitoring timing and sequence of events, both of which may only be partially observed in practice. In this chapter, we study the diagnosis of dense-time discrete event systems using finite-precision digital-clocks to observer event occurrence times. Two diagnosis problems are investigated: (i) diagnosis of timed discrete event system modeled by timed-automaton with both timing and event observation masks, and (ii) diagnosis with dense-time specification which specifies the nonfailure behavior. We show that the verification of diagnosability (ability to detect the execution of a faulty timed-trace within a bounded time delay) as well as the off-line synthesis of a diagnoser are decidable by reducing these problems to the untimed setting. The reduction to the untimed setting also suggests an effective method for the off-line computation of a diagnoser as well as its on-line implementation for diagnosis. The aforementioned results are further extended to the nondeterministic setting, i.e., diagnosis of dense-time DESs using digital-clocks under nondeterministic event observation mask. We introduce the notion of *lifting* and show that diagnosis of dense-time DESs in the nondeterministic setting can be reduced to diagnosis of lifted dense-time DESs under deterministic event observation mask, and hence can be further reduced to diagnosis of lifted untimed DESs under deterministic event observation mask.

5.1 Diagnosis under Event and Timing Masks

In this section we study the diagnosis problem of dense-time discrete event systems modeled by timed-automata under timing as well as (deterministic) event observation masks. Recall that the timed-language to be diagnosed is generated by a plant and hence is prefix-closed. Similarly, a nonfaulty specification language is also prefix-closed.

Let $A = (Q, \Sigma, \Xi, \Upsilon, I, Q_0, Q_m)$ be the timed-automaton model of a system, M_C be the timing-mask associated with a digital-clock C, $M : \Sigma \cup \{\epsilon\} \to \Lambda \cup \{\epsilon\}$ be the event-mask, $\mathcal{F} = \{F_1, F_2, \cdots, F_m\}$ be the set of failure types and $\psi : \Sigma \to 2^{\mathcal{F}}$ be the fault assignment function for each event. Given a timed-trace $\nu = (\sigma_1, t_1) \cdots (\sigma_n, t_n)$, if $F_i \in \psi(\sigma_k)$ for some event σ_k in ν $(1 \leq k \leq n)$, then we say a fault of type F_i has occurred in ν and denote it as $F_i \in \nu$.

The observation of trace ν through both timing and event masks is denoted as

$$M \circ M_C(\nu) = \{M(\nu_c) \mid \nu_c \in M_C(\nu)\}$$

where $M(\nu_c)$ has the form of $\tau^{k_1} M(\sigma_1) \cdots \tau^{k_n} M(\sigma_n) \tau^b$ since "tick" event is observable through the event observation mask M, i.e., $M(\tau) = \tau$. And we have $M \circ M_C(\nu) = M_C \circ M(\nu)$. The *event and timing masked (generated) language* of an timed-automaton A is denoted by $M \circ M_C(L(A)) = \{M \circ M_C(\nu) \mid \nu \in L(A)\}$.

To introduce the faults, let $\mathcal{F} = \{F_1, F_2, \cdots, F_m\}$ be the set of fault types, $\psi : \Sigma \to 2^{\mathcal{F}}$ be the fault-type assignment function for each event, where $\psi(\sigma) = \emptyset$ means σ is a nonfaulty event, otherwise σ is a faulty event and $\psi(\sigma)$ is the set of fault types associated with σ. Hereafter, when we write that "a fault of type F_i has occurred", it will mean that some faulty event σ with $F_i \in \psi(\sigma)$ has occurred. For an untimed-trace $s = \sigma_1 \cdots \sigma_n$, if for some event $\sigma_k (1 \leq k \leq n)$ in the trace, $F_i \in \psi(\sigma_k)$, then we say that a fault of type F_i has occurred in s, and denote it as $F_i \in s$.

The definition of diagnosability for untimed discrete event systems is given below.

Definition 7 A language $H \subseteq \Sigma^*$ is said to be diagnosable with respect to an event mask M

and a fault assignment function ψ if the following holds:

$$(\forall F_i \in \mathcal{F})(\exists N_i > 0)$$

$$(\forall s = \sigma_1 \cdots \sigma_j \in H : F_i \in s)$$

$$(\forall s' = s\sigma_{j+1} \cdots \sigma_{j+n} \in H : n \geq N_i \text{ or } s' \text{ deadlocks})$$

$$(\forall w \in H : M(w) = M(s'))(F_i \in w)$$

A discrete event system is said to be diagnosable if its marked language is diagnosable.

Definition 7 states that an untimed system is diagnosable if the execution of any faulty event can be detected within a bounded delay (bounded number of transitions) from the observations of the system behavior (i.e., no nonfaulty behavior can produce the same observation). Polynomial algorithms for the test of the above diagnosability and the synthesis of the on-line diagnoser can be found in [47, 146], and in [57] respectively.

Remark 6 Note Definition 7 of fault diagnosability allows not only the detection of a fault, but also its *type*. If we choose to not discriminate among the fault-types (so that $\mathcal{F}$ is a singleton set), then Definition 7 will reduce to a more traditional definition of fault detectability (where the aim is to detect a fault, and not its type).

Next we give the definition of diagnosability in the timed setting.

Definition 8 A timed language L is said to be *diagnosable* with respect to the timing-mask M_C, the event-mask M and the fault assignment function ψ if the following holds:

$$(\forall F_i \in \mathcal{F})(\exists B_i \in \Re^+)$$

$$(\forall \nu = (\sigma_1, t_1) \cdots (\sigma_j, t_j) \in L : F_i \in \nu)$$

$$(\forall \nu' = \nu(\sigma_{j+1}, t_{j+1}) \cdots (\sigma_n, t_n) \in L : t_n \geq (t_j + B_i))$$

$$(\forall \mu \in L : M \circ M_C(\nu') \cap M \circ M_C(\mu)) \neq \emptyset)(F_i \in \mu)$$

A dense-time system A is said to be *diagnosable* if its marked timed language $L_m(A)$ is diagnosable.

Definition 8 states that a timed system is diagnosable if the execution of any faulty event can be detected within a bounded time delay from the event and timing-mask observations of the system (i.e., no nonfaulty behavior can produce the same observation).

The following example illustrates the consequence of the precision with which time can be measured on the property of diagnosability of a timed DES.

Example 8 We revisit the timed DES G of Example 2. As analyzed before, if time is measured using a digital-clock that ticks every one unit of time, then G is undiagnosable. Whereas, if time is measured using a digital-clock that ticks every 0.5 units of time, then G becomes diagnosable. This is because a faulty trace $(f, t_0)(a, 1.6)$ can be distinguished from a nonfaulty trace $(u, t_0')(a, 1.2)$ since the former will produce 3 ticks prior to the occurrence of a, whereas the latter will produce only 2 ticks prior to the occurrence of a.

In the following we show that the diagnosis problem of dense-time systems with both timing and event observation masks can be reduced to the diagnosis problem of untimed systems with only event observation mask. To establish the equivalence between the diagnosabilities of a timed language and its timing-masked language, the following simple lemma is needed.

Lemma 5 For any timed-trace $\nu \in L$, $F_i \in \nu$ if and only if $F_i \in \nu_c$ where $\nu_c \in M_C(\nu)$.

Lemma 5 can be obtained by following from the fact that a timing mask does not mask the the events (rather their timings).

Next we show that the diagnosability of a timed language is equivalent to the diagnosability of its timing-masked language.

Theorem 7 Let L be a prefix closed and uniformly non-speeding timed language, C be a uniformly non-speeding and non-slowing digital-clock, M_C be the timing-mask associated with digital-clock C, M be the event-mask, and ψ be the fault assignment function. L is diagnosable with respect to timing-mask M_C, event-mask M and fault assignment function ψ if and only if its timing-masked language $M_C(L)$ is diagnosable with respect to the event mask M and the fault assignment function ψ.

Proof: For the sufficiency, suppose $M_C(L)$ is diagnosable, i.e., for any F_i, there exists a N_i s.t. Definition 7 is satisfied. Since C is uniformly non-slowing, there exists an interval $T^C_{N_i+1} > 0$ s.t. the number of ticks generated during the interval $T^C_{N_i+1}$ is at least $N_i + 1$. Pick a faulty trace $\nu = (\sigma_1, t_1)\cdots(\sigma_l, t_l) \in L$ with $F_i \in \nu$, an extended trace $\nu' = \nu(\sigma_{l+1}, t_{l+1})\cdots(\sigma_m, t_m) \in L$ with $t_m - t_l \geq T^C_{N_i+1}$ and a trace $\mu \in L$ s.t. $M \circ M_C(\nu') \cap M \circ M_C(\mu) \neq \emptyset$, we need show $F_i \in \mu$.

From $\nu, \nu', \mu \in L$ and $M \circ M_C(\nu') \cap M \circ M_C(\mu) \neq \emptyset$, there exist $\nu_c = \tau^{k_1}\sigma_1 \cdots \tau^{k_l}\sigma_l\tau^{b_1} \in M_C(\nu)$, $\nu'_c = \nu_c\tau^{k_{l+1}-b_1}\tau_{l+1}\cdots\tau^{k_m}\sigma_m\tau^{b_2} \in M_C(\nu')$, $\rho = (\tau, t^1_1)\cdots(\tau, t^{k_l}_l)\cdots(\tau, t^{k_m}_m)(\tau, t^1_{m+1})(\tau, t^2_{m+1}) \in L(C)$ and $\mu_c \in M_C(\mu)$ s.t. $M(\nu'_c) = M(\mu_c)$ and $t_i \in [t^{k_i}_i, t^1_{i+1}]$ for $i \leq m$, $b_1, b_2 \in \{0, 1\}$. Also $|\nu'_c| - |\nu_c| \geq \sum_{i=l+1}^m k_i + b_2 - b_1$. The following four cases need to be considered.

Case 1: $b_1 = b_2 = 0$. Then $t_l \in [t^{k_l}_l, t^1_{l+1})$, $t_m \in [t^{k_m}_m, t^1_{m+1})$ and $t^1_{m+1} - t^{k_l}_l > t_m - t_l \geq T^C_{N_i+1}$. Note that C is uniformly non-slowing, $\sum_{i=l+1}^m k_i + 1 \geq N_i + 1$. So $|\nu'_c| - |\nu_c| \geq \sum_{i=l+1}^m k_i \geq N_i$.

Case 2: $b_1 = b_2 = 1$. Then $t_l = t^1_{l+1}$, $t_m = t^1_{m+1}$ and $t^2_{m+1} - t^1_{l+1} > t_m - t_l \geq T^C_{N_i+1}$. Note that C is uniformly non-slowing, $\sum_{i=l+1}^m k_i + 1 \geq N_i + 1$. So $|\nu'_c| - |\nu_c| \geq \sum_{i=l+1}^m k_i \geq N_i$.

Case 3: $b_1 = 0, b_2 = 1$. Then $t_l \in [t^{k_l}_l, t^1_{l+1})$, $t_m = t^1_{m+1}$ and $t^2_{m+1} - t^{k_l}_l > t_m - t_l \geq T^C_{N_i+1}$. Note that C is uniformly non-slowing, $\sum_{i=l+1}^m k_i + 2 \geq N_i + 1$. So $|\nu'_c| - |\nu_c| \geq \sum_{i=l+1}^m k_i + 1 \geq N_i$.

Case 4: $b_1 = 1, b_2 = 0$. Then $t_l = t^1_{l+1}$, $t_m \in [t^{k_m}_l, t^1_{m+1})$ and $t^1_{m+1} - t^1_{l+1} > t_m - t_l \geq T^C_{N_i+1}$. Note that C is uniformly non-slowing, $\sum_{i=l+1}^m k_i \geq N_i + 1$. So $|\nu'_c| - |\nu_c| \geq \sum_{i=l+1}^m k_i - 1 \geq N_i$.

In each case, $\nu_c, \nu'_c, \mu_c \in M_C(L)$, $F_i \in \nu_c$ (from Lemma 5), $|\nu'_c| - |\nu_c| \geq N_i$ and $M(\mu_c) = M(\nu'_c)$. Note that $M_C(L)$ is diagnosable, $F_i \in \mu_c$. And so we have $F_i \in \mu$, as desired.

For the necessity, suppose L is diagnosable, i.e., for any F_i, there exists a B_i s.t. Definition 8 is satisfied. Since L and C are uniformly non-speeding, there exist $N^L_{B_i}$ and $N^C_{B_i}$ s.t. the interval for generating $N^L_{B_i}$ (resp., $N^C_{B_i}$) number of events by L (resp., C) is at least B_i. Pick a faulty trace $\nu_c = \tau^{k_1}\sigma_1 \cdots \tau^{k_l}\sigma_l\tau^{b_1} \in M_C(L)$ with $F_i \in \nu_c$, an extended trace $\nu'_c = \nu_c\tau^{k_{l+1}-b_1}\cdots\tau^{k_m}\sigma_m\tau^{b_2} \in M_C(L)$ with $|\nu'_c| - |\nu_c| \geq N^L_{B_i} + N^C_{B_i} + 1$ and a trace $\mu_c \in M_C(L)$ s.t. $M(\mu_c) = M(\nu'_c)$. We need show $F_i \in \mu_c$.

From $\nu_c, \nu'_c, \mu_c \in M_C(L)$, there exist $\nu = (\sigma_1, t_1)\cdots(\sigma_l, t_l) \in L$, $\nu' = \nu(\sigma_{l+1}, t_{l+1})\cdots(\sigma_m, t_m) \in$

L and $\mu \in L$ s.t. $\nu_c \in M_C(\nu)$, $\nu'_c \in M_C(\nu')$, $\mu_c \in M_C(\mu)$ and $M \circ M_C(\nu') \cap M \circ M_C(\mu) \neq \emptyset$. And there exists $\rho = (\tau, t_1^1) \cdots (\tau, t_m^{k_m}) \cdots (\tau, t_{m+1}^1) \in L(C)$ s.t. $t_i \in [t_i^{k_i}, t_{i+1}^1]$ for $i \leq m$. Also we have $|\nu'_c| - |\nu_c| = (m-l) + (\sum_{i=l+1}^{m} k_i + b_2 - b_1)$ if $\sigma_l, \sigma_m \neq \epsilon$; $(m-l-1) + (\sum_{i=l+1}^{m} k_i + b_2 - b_1)$ if $\sigma_l \neq \epsilon$, $\sigma_m = \epsilon$ or $\sigma_l = \epsilon$, $\sigma_m \neq \epsilon$; $(m-l-2) + (\sum_{i=l+1}^{m} k_i + b_2 - b_1)$ if $\sigma_l, \sigma_m = \epsilon$. Note $b_1, b_2 \in \{0, 1\}$. The following four cases need be considered.

Case 1: $b_1 = b_2 = 0$. Then $t_l \in [t_l^{k_l}, t_{l+1}^1)$, $t_m \in [t_l^{k_m}, t_{m+1}^1)$. From $|\nu'_c| - |\nu_c| \geq N_{B_i}^L + N_{B_i}^C + 1$, either $m - l \geq N_{B_i}^L$ (resp., $m - l - 1 \geq N_{B_i}^L$, $m - l - 2 \geq N_{B_i}^L$) if $\sigma_l, \sigma_m \neq \epsilon$ (resp., $\sigma_l \neq \epsilon$, $\sigma_m = \epsilon$ or $\sigma_l = \epsilon$, $\sigma_m \neq \epsilon$, $\sigma_l, \sigma_m = \epsilon$) or $\sum_{i=l+1}^{m} k_i \geq N_{B_i}^C + 1$. Note that L is uniformly non-speeding, $m - l \geq N_{B_i}^L$ (if $\sigma_l, \sigma_m \neq \epsilon$) implies $t_m - t_l \geq B_i$; similarly, $m - l - 1 \geq N_{B_i}^L$ (if $\sigma_l \neq \epsilon$, $\sigma_m = \epsilon$ or $\sigma_l = \epsilon$, $\sigma_m \neq \epsilon$) implies $t_m - t_l \geq t_{m-1} - t_l \geq B_i$ or $t_m - t_l \geq t_m - t_{l+1} \geq B_i$; $m - l - 2 \geq N_{B_i}^L$ (if $\sigma_l, \sigma_m = \epsilon$) implies $t_m - t_l \geq t_{m-1} - t_{l+1} \geq B_i$; Note that C is uniformly non-speeding, $\sum_{i=l+1}^{m} k_i - 1 \geq N_{B_i}^C$ implies $t_m - t_l > t_m^{k_m} - t_{l+1}^1 \geq B_i$.

Case 2: $b_1 = b_2 = 1$. Then $t_l = t_{l+1}^1$, $t_m = t_{m+1}^1$. From the above analysis, we have $m - l \geq N_{B_i}^L$ ($m - l - 1 \geq N_{B_i}^L$ or $m - l - 2 \geq N_{B_i}^L$) implies $t_m - t_l \geq B_i$. On the other hand, $\sum_{i=l+1}^{m} k_i \geq N_{B_i}^C + 1$ together with uniformly non-speedingness of C implies $t_m - t_l = t_{m+1}^1 - t_{l+1}^1 \geq B_i$.

Case 3: $b_1 = 0$, $b_2 = 1$. Then $t_l \in [t_l^{k_l}, t_{l+1}^1)$, $t_m = t_{m+1}^1$. From the above analysis, we have $m - l \geq N_{B_i}^L$ ($m - l - 1 \geq N_{B_i}^L$ or $m - l - 2 \geq N_{B_i}^L$) implies $t_m - t_l \geq B_i$. On the other hand, $\sum_{i=l+1}^{m} k_i + 1 \geq N_{B_i}^C + 1$ together with uniformly non-speedingness of C implies $t_m - t_l > t_{m+1}^1 - t_{l+1}^1 \geq B_i$.

Case 4: $b_1 = 1$, $b_2 = 0$. Then $t_l = t_{l+1}^1$, $t_m \in [t_l^{k_m}, t_{m+1}^1)$. From the above analysis, we have $m - l \geq N_{B_i}^L$ ($m - l - 1 \geq N_{B_i}^L$ or $m - l - 2 \geq N_{B_i}^L$) implies $t_m - t_l \geq B_i$. On the other hand, $\sum_{i=l+1}^{m} k_i - 1 \geq N_{B_i}^C + 1$ together with uniformly non-speedingness of C implies $t_m - t_l > t_m^{k_m} - t_{l+1}^1 \geq B_i$.

In each case, $\nu, \nu', \mu \in L$, $F_i \in \nu$ (from Lemma 5), $T(\nu') - T(\nu) = t_m - t_l \geq B_i$ and $M \circ M_C(\nu') \cap M \circ M_C(\mu) \neq \emptyset$. Note that L is diagnosable, $F_i \in \mu$. And so we have $F_i \in \mu_c$, as desired. ■

Remark 7 It follows from Theorems 7 that the diagnosis problem of dense-time systems with

respect to both timing and event observation masks can be reduced to the diagnosis problem of untimed discrete event systems. Thus, the results for the diagnosis of untimed discrete event systems like [47, 146, 57] can be applied for the test of diagnosability and the synthesis of on-line as well as off-line diagnoser. In particular the diagnosability of a timed system with respect to the event and timing masks can be performed quadratically in the size of the states in the system and the digital-clock, and exponentially in the number of clocks and the encodings of the clock constraints in the system and the digital-clock.

Example 9 Consider the model of an air conditioning (AC) unit G along with its environment as shown in Figure 5.1 (a). When the environment temperature is "Hot", the AC unit is switched *on* within one unit of time, transmitting to "On" state. From this state either a transition to "Cool" state occurs within one unit of time, or the AC unit fails. In the former case, the AC unit is switched *off* after it has been running for one unit of time. When the AC unit is off, it can be switched *on* after the occurrence of the transition *hot*.

A diagnoser can observe all events except the event f, which represents the failure of the AC unit. Figure 5.1(b) depicts the model of a digital clock C that generates the tick events observed by the diagnoser to keep track of the time. The duration between two successive tick events is one unit of time. Figure 5.1(c) shows the composed automaton $G\|C$. The corresponding clock regions are shown in Figure 5.1(d). It can be checked that the AC unit G is uniformly non-speeding and the clock C is uniformly non-speeding and non-slowing.

From Theorem 7, the diagnosability of the AC unit G under the event and timing masks can be checked by checking the diagnosability of its (untimed) timing-masked language $M_C(L(G))$ under only the event mask. We first obtain the acceptor for the language $M_C(L(G))$ by constructing the refined region-automaton $\overline{\mathcal{R}}^{\epsilon}(G\|C)$ according to Algorithm 1. Next, we check the diagnosability of the untimed language $M_C(L(G))$ using a known algorithm (see Remark 7). A partial refined region-automaton, sufficient to check the diagnosability of $M_C(L(G))$ is shown in Figure 5.1(e). (The sequence of transitions starting from the AC unit state "Off" is omitted since it is not relevant to diagnosability analysis.) From Figure 5.1(e), it can be verified that if a fault (f) occurs after the occurrence of *on*, then all future transitions are tick transitions τ

(since no event is executable at the "Fault" state). On the other hand if a fault does not occur after the occurrence of *on*, then the *cool* event is observed following at most one tick transition τ. It follows that $M_c(L(G))$ is diagnosable with delay bound $N = 2$. So from Theorem 7, $L(G)$ is also diagnosable. This can be independently verified by choosing delay-bound $B = 1$: If following the observation of *on*, the *cool* event is not observed within $B = 1$ unit of time, then we can conclude that a fault has occurred. It should be noted that the above system is not diagnosable in the untimed setting since after the occurrence of a fault, no future observation can occur, and thus the ambiguity between the traces $on.f$ versus *on* (both of which produce the identical observation *on*) is never resolved.

5.2 Diagnosis with Dense-Time Specification

In this section we study the diagnosis problem where one dense timed-automaton is given as the system model and another dense timed-automaton as the specification model which specifies the nonfailure behavior. The task of diagnosis is to diagnose any faulty behavior of the system (with respect to the specification) within a bounded delay of its occurrence in the presence of both timing and event masks. This notion of diagnosability is captured by the following definition.

Definition 9 Given a timed system $G = (Q, \Sigma, \Xi, \Upsilon, I, Q_0, Q_m)$, a specification $R = (Q_R, \Sigma, \Xi_R, \Upsilon_R, I_R, Q_{R0}, Q_{Rm})$ closed relative to G, the timing mask M_C, and the event mask M, (G, R) is said to be diagnosable with respect to M_C and M if the following holds:

$$
\begin{aligned}
&(\exists B \in \Re^+) \\
&(\forall \nu = (\sigma_1, t_1) \cdots (\sigma_j, t_j) \in L_m(G) - L_m(R)) \\
&(\forall \nu' = \nu(\sigma_{j+1}, t_{j+1}) \cdots (\sigma_n, t_n) \in L_m(G) : \\
&t_n \geq (t_j + B)) \\
&(\forall \mu \in L_m(G) : M \circ M_C(\nu') \cap M \circ M_C(\mu)) \neq \emptyset) \\
&(\mu \notin L_m(R))
\end{aligned}
$$

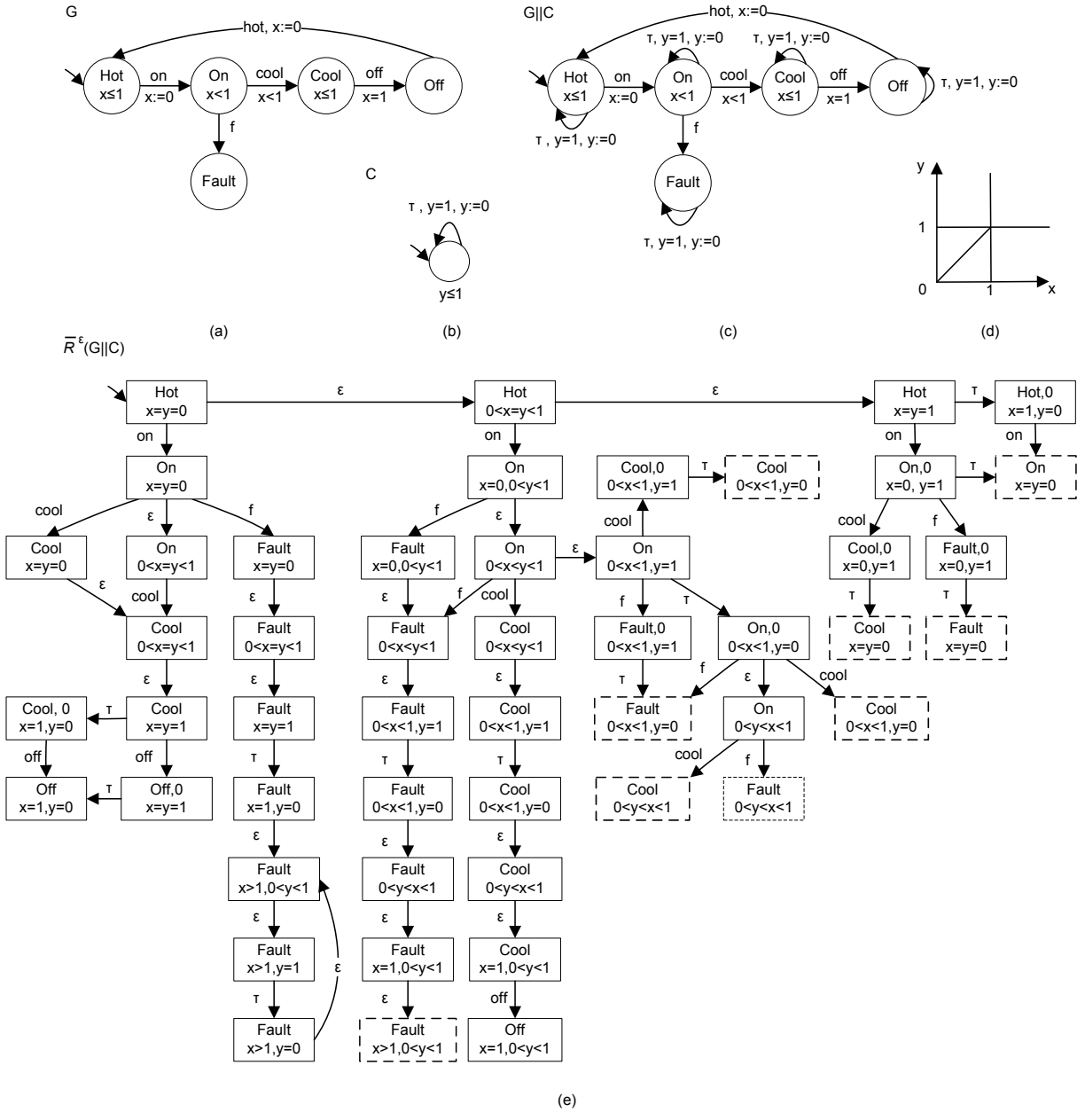

Figure 5.1 Models of the AC unit and digital-clock

Definition 9 states that a timed system and a specification of its nonfaulty behaviors are diagnosable if any violation of the given specification can be detected within a bounded time delay from the event and timing-mask observations of the system behavior (i.e., no nonfaulty behavior can produce the same observation).

Note $L_m(G)$ here represents those traces of interest that must be diagnosed. The relative-closure property requires that all prefixes of a nonfaulty trace that need to be diagnosed (i.e., are accepted by G) are themselves nonfaulty. This is a natural requirement.

For any deterministic specification R, the above diagnosis problem of a pair of timed-automata can be converted to the diagnosis problem of a single timed-automaton with a faulty event as defined in Definition 8. For this, we first complete the specification R by adding a *dump* state and all the missing transitions. The resulting completed specification model is denoted as $\overline{R}$. Next, we introduce in $\overline{R}$ a faulty event f, whose occurrence indicates the execution of a behavior violating the given specification. The resulting refined completed specification is denoted as $\overline{R}^f$. Then, we reduce the diagnosis problem of the pair (G, R) to that of the system $G\|\overline{R}^f$. Note a nonfaulty specification can always be accepted by a trim automaton, and we assume without loss of generality that R is trim, so that $pr(L_m(R)) = L(R)$.

From the construction of the completed specification $\overline{R}$ (refer to Section 4.4), we can see $\overline{R}$ accepts any timed-trace over the event set Σ and if a timed-trace leads to the state *dump*, then that trace is not marked by R (when R is deterministic), in which case it indicates a fault. In order to represent such a fault using a faulty event, we (i) "split" the *dump* state into $dump_1$ and $dump_2$ states, (ii) make all self-loop transitions of *dump* as self-loop transitions of $dump_2$, (iii) make all incoming non-self-loop transitions of *dump* as incoming transitions of $dump_1$, and (iv) add an outgoing transition on f from $dump_1$ to $dump_2$.

The refined complete specification $\overline{R}^f$ is defined as follows. $\overline{R}^f = (Q_R \cup \{dump_1, dump_2\}, \Sigma \cup \{f\}, \Xi_R \cup \{\xi_f\}, \Upsilon_R \cup \Upsilon^f_{add}, I^f_R, Q_{R0}, Q_{Rm} \cup \{dump_2\})$, where $\forall q \in Q_R$, $I^f_R(q) = I_R(q)$, $I^f_R(dump_1) = (\xi_f = 0)$, $I^f_R(dump_2) = \texttt{true}$, and the set of transitions Υ^f_{add} is defined as:

- $\forall q \in Q_R$, $\forall \sigma \in \Sigma$, if there are $n \geq 1$ out-going transitions from q labeled with σ, and let $\{\phi^1_\sigma, \cdots, \phi^n_\sigma\}$ be the set of guard conditions associated with those n transitions, then

$(q, \sigma, \neg(\vee_{i=1}^{n} \phi_\sigma^i), \{\xi_f\}, dump_1) \in \Upsilon_{add}^f$; otherwise $(q, \sigma, \texttt{true}, \{\xi_f\}, dump_1) \in \Upsilon_{add}^f$

- $\forall \sigma \in \Sigma$, $(dump_2, \sigma, \texttt{true}, \emptyset, dump_2) \in \Upsilon_{add}^f$
- $(dump_1, f, \xi_f = 0, \emptyset, dump_2) \in \Upsilon_{add}^f$

In the composed automaton $G \| \overline{R}^f$, we have only one failure type, i.e., $\mathcal{F} = \{F_1\}$, and the corresponding fault assignment function ψ_f is defined as $\psi_f(f) = \{F_1\}$ and $\psi_f(\sigma) = \emptyset$ for any $\sigma \in \Sigma$. The faulty event f is unobservable, i.e., $M(f) = \epsilon$. Also note $f \notin \Sigma$, and so the faulty event f occurs asynchronously in the composition $G \| \overline{R}^f$ (i.e., without the participation of G, whereas all other events occur synchronously) and immediately after the occurrence a violation of the specification.

From the construction of $G \| \overline{R}^f$, it can be proved that (G, R) is diagnosable according to Definition 9 if and only if $G \| \overline{R}^f$ is diagnosable according to Definition 8. (The diagnosis problem of the timed system $G \| \overline{R}^f$ can be further reduced to the diagnosis problem of its corresponding untimed system as described in the earlier sections.) To show this, we need the following lemmas.

Lemma 6 Given G and deterministic and relative-closed R, it holds that $\Pi_\Sigma(L(G \| \overline{R}^f)) = L(G)$ and $\Pi_\Sigma(L_m(G \| \overline{R}^f)) = L_m(G)$.

Proof: The first conclusion follows from the fact $L(G) \subseteq \Pi_\Sigma(L(\overline{R}^f)) = (\Sigma \times \Re^+)^*$.

Next we show the second conclusion. It follows from the definition of synchronous composition that $\Pi_\Sigma(L_m(G \| \overline{R}^f)) \subseteq L_m(G)$. To show the converse containment $L_m(G) \subseteq \Pi_\Sigma(L_m(G \| \overline{R}^f))$, pick $\nu \in L_m(G)$. If $\nu \in L_m(R)$, then $\nu \in L_m(G \| \overline{R}^f)$. On the other hand if $\nu \in L_m(G) - L_m(R)$, then from the relative-closure of R and the fact $pr(L_m(R)) = L(R)$ (since R is trim), we have $\nu \in L_m(G) - pr(L_m(R)) \cap L_m(G) = L_m(G) - L(R)$. Therefore ν must reach the *dump* state in $\overline{R}$ (since R is deterministic). This implies that there exists $\mu \in L(\overline{R}^f)$ such that $\Pi(\mu) = \nu$, and $f \in \mu$. Then μ reaches the state $dump_2$, which is a marked state of $\overline{R}^f$. Further since $\Pi(\mu) = \nu \in L_m(G)$, we have that $\mu \in L_m(G \| \overline{R}^f)$. Thus it can be concluded that $\nu \in \Pi_\Sigma(L_m(G \| \overline{R}^f))$. ■

Lemma 7 Given G and deterministic relative-closed R, any $\mu \in L_m(G\|\overline{R}^f)$ contains the faulty event f if and only if $\Pi_\Sigma(\mu) \in L_m(G) - L_m(R)$.

Proof: Pick $\mu \in L_m(G\|\overline{R}^f)$ with $f \in \mu$. From Lemma 6, $\Pi_\Sigma(\mu) \in L_m(G)$. Since μ contains the faulty event f, the execution of μ in $\overline{R}^f$ reaches state $dump_2$. Since the projected trace $\Pi_\Sigma(\mu)$ only erases the faulty event, the execution of $\Pi_\Sigma(\mu)$ reaches the *dump* state in $\overline{R}$. Therefore $\mu \in L_m(G\|\overline{R}^f)$ contains the faulty event f if and only if $\Pi_\Sigma(\mu) \in L_m(G)$ and its execution reaches the *dump* state in $\overline{R}$ (i.e., $\Pi_\Sigma(\mu) \notin L(R) = pr(L_m(R))$, for R is deterministic). Note $\Pi_\Sigma(\mu) \in L_m(G)$. Further from the relative-closure property of R, we have $\Pi_\Sigma(\mu) \notin L_m(R)$, as desired. ■

With Lemmas 6 and 7 in hand, we are ready to establish the following theorem.

Theorem 8 Given a system G, a deterministic relative closed specification R, a timing mask M_C, and an event mask M, (G, R) is diagnosable with respect to M_C and M if and only if $G\|\overline{R}^f$ is diagnosable with respect to M_C, M, and ψ_f.

Proof: For the sufficiency, suppose $G\|\overline{R}^f$ is diagnosable, i.e., there exists $B \in \Re^+$ such that Definition 8 is satisfied. Pick a trace $\nu = (\sigma_1, t_1)\cdots(\sigma_j, t_j) \in L_m(G) - L_m(R)$, an extended trace $\nu' = \nu(\sigma_{j+1}, t_{j+1})\cdots(\sigma_n, t_n) \in L_m(G)$ with $t_n \geq t_j + B$, and $\mu \in L_m(G)$ such that $M \circ M_C(\nu') \cap M \circ M_C(\mu) \neq \emptyset$. We need show $\mu \in L_m(G) - L_m(R)$.

Since $\nu \in L_m(G) - L_m(R)$, $\nu', \mu \in L_m(G)$, from Lemma 6 and 7, there exist $\widetilde{\nu}, \widetilde{\nu}', \widetilde{\mu} \in L_m(G\|\overline{R}^f)$ s.t. $\Pi_\Sigma(\widetilde{\nu}) = \nu$, $\Pi_\Sigma(\widetilde{\nu}') = \nu'$, $\Pi_\Sigma(\widetilde{\mu}) = \mu$, and $\widetilde{\nu}$ contains the faulty event f. Since the faulty transition occurs instantaneously (see the construction of $\overline{R}^f$), the last events in $\widetilde{\nu}$ and $\widetilde{\nu}'$ occur at the same times as the last events in ν and ν', i.e., $T(\widetilde{\nu}) = T(\nu)$ and $T(\widetilde{\nu'}) = T(\nu')$. Thus the last events $\widetilde{\nu}$ and $\widetilde{\nu}'$ are separated by at least the duration B. Note that f is unobservable under the event-mask M, $M \circ M_C(\widetilde{\mu}) = M \circ M_C(\Pi_\Sigma(\widetilde{\mu})) = M \circ M_C(\mu)$ and $M \circ M_C(\widetilde{\nu'}) = M \circ M_C(\nu')$. Since $G\|\overline{R}^f$ is diagnosable, $f \in \widetilde{\mu}$. Then from Lemma 7, we have $\mu \in L_m(G) - L_m(R)$.

For the necessity, suppose (G, R) is diagnosable, i.e., there exists $B \in \Re^+$ such that Definition 9 is satisfied. Pick a faulty trace $\widetilde{\nu} = (\sigma'_1, t_1)\cdots(\sigma'_j, t_j) \in L_m(G\|\overline{R}^f)$ where $\sigma'_i \in \Sigma \cup \{f\}$

and $\sigma'_k = f$ for some k, an extended trace $\widetilde{\nu}' = \widetilde{\nu}(\sigma'_{j+1}, t_{j+1}) \dots (\sigma'_n, t_n) \in L_m(G \| \overline{R}^f)$ with $t_n \geq t_j + B$, and $\widetilde{\mu} \in L_m(G \| \overline{R}^f)$ such that $M \circ M_C(\widetilde{\mu}) \cap M \circ M_C(\widetilde{\nu}') \neq \emptyset$. We need show $f \in \widetilde{\mu}$.

Since $\widetilde{\nu}, \widetilde{\nu}', \widetilde{\mu} \in L_m(G \| \overline{R}^f)$ and $f \in \widetilde{\nu}$, from Lemma 6 and 7, there exist $\nu, \nu', \mu \in L_m(G)$ s.t. $\nu = \Pi_\Sigma(\widetilde{\nu})$, $\nu' = \Pi_\Sigma(\widetilde{\nu}')$, $\mu = \Pi_\Sigma(\widetilde{\mu}) \in L_m(G)$ and $\nu \in L_m(G) - L_m(R)$. Since the projection Π_Σ only erases the faculty event f which occurs instantaneously, the last events in ν and ν' are separated by the same duration as the last events in $\widetilde{\nu}$ and $\widetilde{\nu}'$, namely by at least the duration B. Note that f is unobservable under event-mask M, $M \circ M_C(\mu) = M \circ M_C(\widetilde{\mu})$ and $M \circ M_C(\nu') = M \circ M_C(\widetilde{\nu}')$. Since (G, R) is diagnosable, $\mu \in L_m(G) - L_m(R)$. Then from Lemma 7, we have $f \in \widetilde{\mu}$. ■

Remark 8 It follows from Theorem 8 that diagnosability of (G, R), where G and R are such that $G \| \overline{R}^f$ is uniformly non-speeding and further R is deterministic and relative closed, with respect to the timing and event masks can be performed quadratically in the size of the states in G, R and the digital-clock C, and exponentially in the number of clocks and the encodings of the clock constraints in G, R and C.

The following example illustrates the equivalence between the diagnosability of (G, R) and that of $G || \overline{R}^f$.

Example 10 Consider the system G and the deterministic relative-closed specification R as shown in Figure 5.2. Suppose the digital clock C ticks with interval of one. From the specification R, we construct $\overline{R}$, $\overline{R}^f$ and $G || \overline{R}^f$, which are shown in Figure 5.2.

Suppose $M(a) = a$, $M(b) = b$, then (G, R) is diagnosable. This is because the trace in $L_m(G)$, which violates the specification $L_m(R)$, must be of the form $(a, t)(\epsilon, t')$ with $t < 1$. Such a trace is observed as $a\tau^*$. On the other hand, the trace with the same event observation and which satisfies the specification is of the form $(a, \overline{t})(\epsilon, \overline{t}')$ with $\overline{t} > 1$. Such a trace is is observed as $\tau^k a \tau^*$ for some $k \geq 1$. The conclusion about diagnosability of $G \| \overline{R}^f$ can be drawn as well by comparing a faulty trace $(a, t)(f, t)$ with $t < 1$ and a nonfaulty trace (a, t') with $t' > 1$.

Now suppose $M(a) = M(b) = a$, then (G, R) becomes undiagnosable. This is because a faulty trace $(a, 0.5)$ can not be distinguished from a nonfaulty trace $(b, 0.7)$. Both produce the same observation, a. Similarly, $G||\overline{R}^f$ is also undiagnosable since a faulty trace $(a, 0.5)(f, 0.5)$ can not be distinguished from a nonfaulty trace $(b, 0.7)$ (both produce the same observation, a).

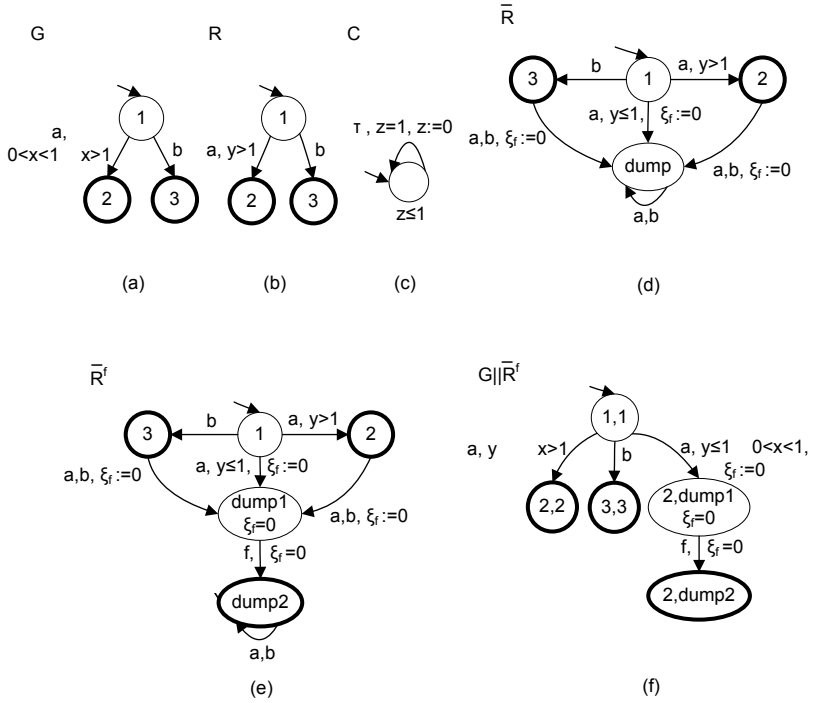

Figure 5.2 Automata of $G, R, C, \overline{R}, \overline{R}^f$ and $G||\overline{R}^f$

5.3 Extension to Nondeterministic Setting

In the above work on diagnosis of dense-time DESs using digital-clock, we assume that sensors are reliable so that they can be modeled as a deterministic (point-valued) observation mask. However in certain harsh environments such as nuclear systems and mobile systems, sensors may not be reliable [121]. This results in a nondeterministic (set-valued) observation mask. In this section, we extend the aforementioned results to the nondeterministic setting by allowing the event observation mask to be nondeterministic so as to capture unreliability of sensors.

5.3.1 Introduction to Lifting

To solve the diagnosis problem under nondeterministic event observations, we need to introduce the notion of *lifting*. Given an event set Σ and a nondeterministic observation mask M_n, a lifted-event can be viewed as a pair consisting of an event and its possible observation. The set of all *lifted-events* is given by $\tilde{\Sigma} := \{\sigma_\delta \mid \sigma \in \Sigma, \delta \in M_n(\sigma)\}$. Given a plant $G = (Q, \Sigma, \Xi, \Upsilon, I, Q_0, Q_m)$, its corresponding *lifted-plant*, denoted $\tilde{G}$, has the same state space as G, whereas each of its transitions is labeled by an event together with its possible observation: $\tilde{G} = (Q, \tilde{\Sigma}, \Xi, \tilde{\Upsilon}, I, Q_0, Q_m)$, where $(q, \sigma_\delta, \phi, r, q') \in \tilde{\Upsilon}$ if and only if $(q, \sigma, \phi, r, q') \in \Upsilon$ and $\delta \in M_n(\sigma)$.

The lifting operation can also be defined for languages: *Lifting of a language* $L \subseteq \mathcal{T}$, denoted $\tilde{L} \subseteq \tilde{\mathcal{T}}$, is defined as $\tilde{L} := \bigcup_{s \in L} \tilde{s}$, where the *lifting of a trace* s is inductively defined as follows:

$$\tilde{\epsilon} = \{\epsilon\}; \quad s \in \mathcal{T}, \sigma \in \Sigma : \widetilde{s(\sigma, t)} = \tilde{s}\{(\sigma_\delta, t) \mid \delta \in M_n(\sigma)\}.$$

It can be concluded that $L(\tilde{G}) = \widetilde{L(G)}$.

An observation mask over a lifted-language is called a *lifted-observation mask*. Given an observation mask M_n, its lifted observation mask, denoted $\tilde{M}_n$, is defined by $\tilde{M}_n(\sigma_\delta) := \delta$ for any $\sigma_\delta \in \tilde{\Sigma}$. $\tilde{M}_n$ can be generalized to lifted-traces and lifted-languages in a natural way. It can be seen that the observations of a language under an observation mask agree with those of its lifted-language under the lifted-observation mask: $M_n(L) = \tilde{M}_n(\tilde{L})$. Furthermore, a lifted-observation mask $\tilde{M}_n$ is always deterministic (regardless of whether its underlying observation mask M_n is deterministic or not).

To retrieve a physical event from its lifted version, a *projection* function (with a little abuse of notation Π) $\Pi : \tilde{\Sigma} \to \Sigma$ is extended to be defined in the domain of $\tilde{\Sigma}$ as $\Pi(\sigma_\delta) := \sigma$ for any $\sigma_\delta \in \tilde{\Sigma}$. Π can be generalized to the lifted-traces and lifted-languages in a natural way.

5.3.2 Diagnosis under Nondeterministic Event and Timing-Masks

In the following we extend the notion of diagnosability with event as well as timing masks to the nondeterministic setting, i.e., with nondeterministic event and timing masks.

Let $A = (Q, \Sigma, \Xi, \Upsilon, I, Q_0, Q_m)$ be the timed-automaton model of a system, M_C be the timing-mask associated with a digital-clock C, $M_n : \Sigma \cup \{\epsilon\} \to \Lambda \cup \{\epsilon\}$ be the nondeterministic event-mask, $\mathcal{F} = \{F_1, F_2, \cdots, F_m\}$ be the set of failure types and $\psi : \Sigma \to 2^{\mathcal{F}}$ be the fault assignment function for each event. The notion of diagnosability in the nondeterministic setting is defined as below.

Definition 10 A timed language L is said to be *diagnosable* with respect to the timing-mask M_C, the nondeterministic event-mask M_n and the fault assignment function ψ if the following holds:

$$(\forall F_i \in \mathcal{F})(\exists B_i \in \Re^+)$$
$$(\forall \nu = (\sigma_1, t_1) \cdots (\sigma_j, t_j) \in L : F_i \in \nu)$$
$$(\forall \nu' = \nu(\sigma_{j+1}, t_{j+1}) \cdots (\sigma_n, t_n) \in L : t_n \geq (t_j + B_i))$$
$$(\forall \mu \in L : M_n \circ M_C(\nu') \cap M_n \circ M_C(\mu)) \neq \emptyset)(F_i \in \mu)$$

A dense-time system A is said to be *diagnosable* if its marked timed language $L_m(A)$ is diagnosable.

Definition 10 is the similar to Definition 8 except that in Definition 10 the event observation mask is allowed to be nondeterministic.

Given the set of fault types $\mathcal{F}$, let $\tilde{\mathcal{F}} := \{F_\epsilon \mid F \in \mathcal{F}\}$ denote the lifted fault types. (Note $\forall F \in \mathcal{F}$, $M_n(F) = \epsilon$.) The lifted fault assignment function is defined by $\tilde{\psi} : \tilde{\Sigma} \to 2^{\mathcal{F}}$, and $\tilde{\psi}(\sigma_\delta) = \psi(\sigma)$ for any $\sigma \in \Sigma$, $\delta \in M_n(\sigma)$.

In the following we show that diagnosis of dense-time systems with both nondeterministic timing and event observation masks can be reduced to that of dense-time systems with deterministic event observation mask and nondeterministic timing observation mask (that is studied in the previous section). To establish the equivalence between the diagnosability of a timed language and that of its lifted timed language, the following simple lemma is needed.

Lemma 8 For any timed-trace $\nu \in L$, $F_i \in \nu$ if and only if $F_{i\epsilon} \in \tilde{\nu}$ where $\tilde{\nu} \in \tilde{L}$.

Lemma 8 can be obtained from the definition of lifting.

Next we show that the diagnosability of a timed language is equivalent to the diagnosability of its lifted timed language.

Theorem 9 Let L be a prefix closed and uniformly non-speeding timed language, C be a uniformly non-speeding and non-slowing digital-clock, M_C be the timing-mask associated with digital-clock C, M_n be the nondeterministic event-mask, and ψ be the fault assignment function. L is diagnosable with respect to timing-mask M_C, event-mask M_n and fault assignment function ψ if and only if its lifted timed language $\tilde{L}$ is diagnosable with respect to timing-mask M_C, event mask $\tilde{M}_n$ and the lifted fault assignment function $\tilde{\psi}$.

Proof: We start by showing the sufficiency. Pick $F_i \in \mathcal{F}$, $\nu = (\sigma_1, t_1)\cdots(\sigma_j, t_j) \in L : F_i \in \nu$, $\nu' = \nu(\sigma_{j+1}, t_{j+1})\cdots(\sigma_n, t_n) \in L : t_n \geq (t_j + B_i)$, and $\mu \in L : M \circ M_C(\nu') \cap M \circ M_C(\mu)) \neq \emptyset$. Then there exist $\tilde{\nu}$, $\tilde{\nu}'$, $\tilde{\mu} \in \tilde{L}$ s.t. $\Pi(\tilde{\nu}) = \nu$, $\Pi(\tilde{\nu}') = \nu'$, $\Pi(\tilde{\mu}) = \mu$, $F_{i\epsilon} \in \tilde{\nu}$ (from Lemma 8), and $\tilde{M}_n \circ M_C(\tilde{\nu}') \cap \tilde{M}_n \circ M_C(\tilde{\mu})) \neq \emptyset$ (from $\tilde{M}_n \circ M_C(\tilde{\nu}') = M \circ M_C(\tilde{\nu}')$, and $\tilde{M}_n \circ M_C(\tilde{\mu}) = M \circ M_C(\tilde{\mu})$). Note $\tilde{L}$ is diagnosable with respect to timing-mask M_C, event mask $\tilde{M}_n$ and the lifted fault assignment function $\tilde{\psi}$. Then we have $F_{i\epsilon} \in \tilde{\mu}$, equivalently, $F_i \in \mu$.

Next we show the necessity. Pick $F_{i\epsilon} \in \tilde{\mathcal{F}}$, $\tilde{\nu} = (\sigma_{1\delta_1}, t_1)\cdots(\sigma_{j\delta_j}, t_j) \in \tilde{L} : F_{i\delta_i} \in \tilde{\nu}$, $\tilde{\nu}' = \tilde{\nu}(\sigma_{j+1}, t_{j+1})\cdots(\sigma_n, t_n) \in \tilde{L} : t_n \geq (t_j + B_i)$, and $\tilde{M}_n \circ M_C(\tilde{\nu}') \cap \tilde{M}_n \circ M_C(\tilde{\mu})) \neq \emptyset$. Then there exist ν, ν', $\mu \in L$ s.t. $\Pi(\tilde{\nu}) = \nu$, $\Pi(\tilde{\nu}') = \nu'$, $\Pi(\tilde{\mu}) = \mu$, $F_i \in \nu$ (from Lemma 8), and $M_n \circ M_C(\nu') \cap M_n \circ M_C(\mu)) \neq \emptyset$) (from $M_n \circ M_C(\tilde{\nu}') = \tilde{M}_n \circ M_C(\tilde{\nu}')$, and $M_n \circ M_C(\tilde{\mu}) = \tilde{M}_n \circ M_C(\tilde{\mu})$). Note L is diagnosable with respect to timing-mask M_C, event mask M_n and the fault assignment function ψ. Then we have $F_i \in \mu$, equivalently, $F_{i\epsilon} \in \tilde{\mu}$. ■

The following example illustrates Theorem 9.

Example 11 Consider the system G and the digital clock C as shown in Figure 5.3. Suppose the nondeterministic observation mask M_n is defined as $M_n(a) = \{a_1, a_2\}$, $M_n(a_1) = a_1$, $M_n(a_2) = a_2$, $M_n(b) = b$, and $M_n(f) = \epsilon$. The lifted plant $\tilde{G}$ is shown in Figure 5.3 (c). From Definition 10, we have G is diagnosable. This is because in a nonfaulty trace where a_1 is observed, event b can be observed within 1 time unit upon the observation of a_1, whereas in a faulty trace where a_1 is also observed, event b can never be observed. Similarly, we have

the lifted timed language $L(\tilde{G})$ is also diagnosable. However if event b is unobservable, then G becomes non-diagnosable since a nonfaulty trace $(a, 0.5)(b, 1)$ cannot be distinguished from a faulty trace $(a_1, 0.3)(f, 0.6)(\epsilon, 1)$ (both produce the observation $a_1\tau$). Note the lifted nonfaulty trace $(a_{a_1}, 0.5)(b_\epsilon, 1)$ and the lifted faulty trace $(a_{1a_1}, 0.3)(f_\epsilon, 0.6)(\epsilon_\epsilon, 1)$ are indistinguishable, and so we have $L(\tilde{G})$ is not diagnosable.

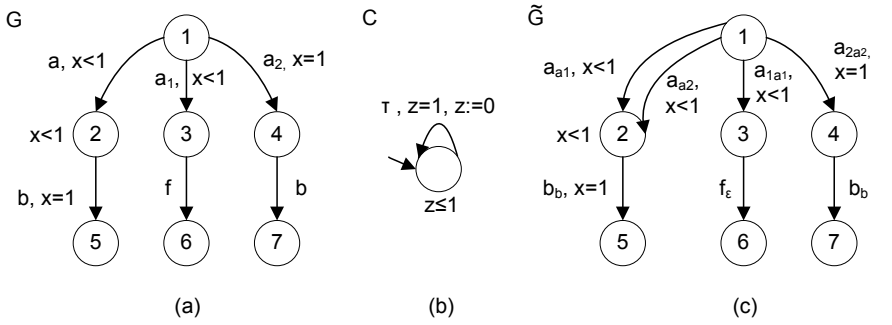

Figure 5.3 Automata of $G, C, \tilde{G}$

It follows from Theorem 9 and Theorem 7 that diagnosis of dense-time DESs in the nondeterministic setting can be further reduced to diagnosis of lifted untimed DESs in the deterministic setting. In result, the extensive results of diagnosis of untimed DESs can be applied to diagnosis of dense-time DESs under nondeterministic event and timing masks.

5.3.3 Diagnosis with Dense-Time Specification under Nondeterministic Event Mask

In the following we extend the prior result of diagnosis with dense time specification to the nondeterministic setting. The notion of diagnosability with respect to a given dense-time specification and nondeterministic observation mask is defined as below.

Definition 11 Given a timed system $G = (Q, \Sigma, \Xi, \Upsilon, I, Q_0, Q_m)$, a specification $R = (Q_R, \Sigma, \Xi_R, \Upsilon_R, I_R, Q_{R0}, Q_{Rm})$ closed relative to G, the timing mask M_C, and the nondeterministic event mask M_n, (G, R) is said to be diagnosable with respect to M_C and M_n if the following

holds:

$$
\begin{aligned}
&(\exists B \in \Re^{+}) \\
&(\forall \nu = (\sigma_1, t_1) \cdots (\sigma_j, t_j) \in L_m(G) - L_m(R)) \\
&(\forall \nu' = \nu(\sigma_{j+1}, t_{j+1}) \cdots (\sigma_n, t_n) \in L_m(G) : \\
&t_n \geq (t_j + B)) \\
&(\forall \mu \in L_m(G) : M_n \circ M_C(\nu') \cap M_n \circ M_C(\mu)) \neq \emptyset) \\
&(\mu \notin L_m(R))
\end{aligned}
$$

Definition 11 states that a timed system and a specification of its nonfaulty behaviors are diagnosable if any violation of the given specification can be detected within a bounded time delay from the event and timing-mask observations of the system behavior. This is similar to Definition 9 except that in Definition 11 the event observation mask is allowed to be nondeterministic.

Next we show that diagnosability of a timed language in the nondeterministic setting is equivalent to diagnosability of its lifted timed language in the deterministic setting. To show this, we need the following lemmas.

Lemma 9 Given G and deterministic and relative-closed R, it holds that $\Pi_\Sigma(L(\tilde{G} \| \widetilde{\overline{R}^f})) = L(G)$ and $\Pi_\Sigma(L_m(\tilde{G} \| \widetilde{\overline{R}^f})) = L_m(G)$.

Note in Lemma 9, $\widetilde{\overline{R}^f}$ denotes the lifted refined complete specification. Lemma 9 can be obtained following from Lemma 6 and the fact $\Pi_\Sigma(L(\tilde{G} \| \widetilde{\overline{R}^f})) = \Pi_\Sigma(L(G \| \overline{R}^f))$.

Lemma 10 Given G and deterministic relative-closed R, any $\tilde{\mu} \in L_m(\tilde{G} \| \widetilde{\overline{R}^f})$ contains the faulty event f_ϵ if and only if $\Pi_\Sigma(\mu) \in L_m(G) - L_m(R)$.

Lemma 10 can be shown from the construction of a lifted automaton and the fact $\Pi_\Sigma(\tilde{\mu}) = \Pi_\Sigma(\mu)$.

With Lemmas 6 and 7, we are able to establish the following theorem.

Theorem 10 Given a system G, a deterministic relative closed specification R, a timing mask M_C, and a nondeterministic event mask M_n, (G, R) is diagnosable with respect to M_C and M_n if and only if $\tilde{G}\|\widetilde{\overline{R}^f}$ is diagnosable with respect to M_C, $\tilde{M}_n$, and $\tilde{\psi}_f$.

Proof: For the sufficiency, suppose $\tilde{G}\|\widetilde{\overline{R}^f}$ is diagnosable. Pick a trace $\nu = (\sigma_1, t_1)\cdots(\sigma_j, t_j) \in L_m(G) - L_m(R)$, an extended trace $\nu' = \nu(\sigma_{j+1}, t_{j+1})\cdots(\sigma_n, t_n) \in L_m(G)$ with $t_n \geq t_j + B$, and $\mu \in L_m(G)$ such that $M_n \circ M_C(\nu') \cap M_n \circ M_C(\mu) \neq \emptyset$. We need show $\mu \in L_m(G) - L_m(R)$.

Since $\nu \in L_m(G) - L_m(R)$, $\nu', \mu \in L_m(G)$, from Lemma 9 and 10, there exist $\widetilde{\nu}, \widetilde{\nu}', \widetilde{\mu} \in L_m(\tilde{G}\|\widetilde{\overline{R}^f})$ s.t. $\Pi_\Sigma(\widetilde{\nu}) = \nu$, $\Pi_\Sigma(\widetilde{\nu}') = \nu'$, $\Pi_\Sigma(\widetilde{\mu}) = \mu$, and $f_\epsilon \in \widetilde{\nu}$ (from Lemma 8). Since the faulty transition occurs instantaneously, the last events in $\widetilde{\nu}$ and $\widetilde{\nu}'$ occur at the same times as the last events in ν and ν', i.e., $T(\widetilde{\nu}) = T(\nu)$ and $T(\widetilde{\nu'}) = T(\nu')$. Thus the last events $\widetilde{\nu}$ and $\widetilde{\nu}'$ are separated by at least the duration B. Note that f is unobservable under the event-masks M_n and $\tilde{M}_n$, $\tilde{M}_n \circ M_C(\widetilde{\mu}) = \tilde{M}_n \circ M_C(\Pi_\Sigma(\widetilde{\mu})) = M_n \circ M_C(\mu)$ and similarly $\tilde{M}_n \circ M_C(\widetilde{\nu'}) = M_n \circ M_C(\nu')$. From diagnosability of $\tilde{G}\|\widetilde{\overline{R}^f}$, $f_\epsilon \in \widetilde{\mu}$. Then from Lemma 10, we have $\mu \in L_m(G) - L_m(R)$.

For the necessity, suppose (G, R) is diagnosable. Pick a faulty trace $\widetilde{\nu} = (\sigma'_{1\delta_1}, t_1)\cdots(\sigma'_{j\delta_j}, t_j) \in L_m(\tilde{G}\|\widetilde{\overline{R}^f})$ where $\sigma'_k = f$ for some k $(1 \leq k \leq j)$, an extended trace $\widetilde{\nu}' = \widetilde{\nu}(\sigma'_{j+1\delta_{j+1}}, t_{j+1})\ldots(\sigma'_{n\delta_n}, t_n) \in L_m(\tilde{G}\|\widetilde{\overline{R}^f})$ with $t_n \geq t_j + B$, and $\widetilde{\mu} \in L_m(\tilde{G}\|\widetilde{\overline{R}^f})$ s.t. $\tilde{M}_n \circ M_C(\widetilde{\mu}) \cap \tilde{M}_n \circ M_C(\widetilde{\nu}') \neq \emptyset$. We need show $f_\epsilon \in \widetilde{\mu}$.

Since $\widetilde{\nu}, \widetilde{\nu}', \widetilde{\mu} \in L_m(\tilde{G}\|\widetilde{\overline{R}^f})$ and $f_\epsilon \in \widetilde{\nu}$, from Lemma 9 and 10, there exist $\nu, \nu', \mu \in L_m(G)$ s.t. $\nu = \Pi_\Sigma(\widetilde{\nu})$, $\nu' = \Pi_\Sigma(\widetilde{\nu}')$, $\mu = \Pi_\Sigma(\widetilde{\mu}) \in L_m(G)$ and $\nu \in L_m(G) - L_m(R)$. Since the projection Π_Σ only erases the lifted faculty event f_ϵ which occurs instantaneously, the last events in ν and ν' are separated by the same duration as the last events in $\widetilde{\nu}$ and $\widetilde{\nu}'$, namely by at least the duration B. Note that f is unobservable under event-masks M_n and $\tilde{M}_n$, $M_n \circ M_C(\mu) = \tilde{M}_n \circ M_C(\widetilde{\mu})$ and $M_n \circ M_C(\nu') = \tilde{M}_n \circ M_C(\widetilde{\nu}')$. From diagnosability of (G, R), $\mu \in L_m(G) - L_m(R)$. Then from Lemma 10, we have $f_\epsilon \in \widetilde{\mu}$. ■

5.4 Conclusion

We study the diagnosis problem of timed discrete event systems where the system as well as the nonfailure specification is modeled by a dense timed-automaton [6]. While it is meaningful for a system as well as its specification of nonfailure behavior to have a dense-time semantics, it is not practical for a diagnoser to be able to measure dense-time precisely. An imprecision in measurement of time can be viewed as partial observability of "time" just as the presence of imprecise sensors leads to a partial observability of events. A key observation we make is that diagnosability is preserved under timing mask. Based on this observation we have shown that diagnosis of dense-time systems can be reduced to one of untimed systems. Consequently, the results of untimed setting such as those reported in [47, 57] can be applied to perform the diagnosis of a dense-time system against a dense-time specification in the presence of partial observation of events as well as imprecise measurement of time. We further study the diagnosis problem in the nondeterministic setting, i.e., diagnosis of dense-time DESs using digital-clocks under nondeterministic event observation mask. We introduce the notion of lifting and show that diagnosis of dense-time DESs in the nondeterministic setting can be reduced to that of lifted dense-time DESs under deterministic event observation mask, and hence can be further reduced to that of lifted untimed DESs under deterministic event observation mask.

CHAPTER 6. CONCLUSION AND FUTURE WORK

6.1 Summary of Dissertation

In this dissertation, we studied supervisory control and diagnosis of dense-time discrete event systems which employ digital-clocks of finite-precision to observe event occurrence times. This is a realistic scenario compared to the prior work which assumed that time can be measured precisely. The main contributions of this dissertation are summarized as follows.

1. We introduced the notion of timing-mask associated with a digital-clock and formally defined a timing-masked language. A timing-masked language consists of all the possible untimed observations of timed traces in which the event occurrence times are observed using finite-precision digital-clocks. In contrast to the conventional partial observation of events, a timing-mask arising due to the use of digital-clocks to measure event occurrence times is set-valued, and hence is nondeterministic. The proposed notion of timing-mask captures the nondeterminism of the untimed observations as observed through using digital-clocks explicitly. And since plant events and clock ticks can occur simultaneously, a timing-masked language of a timed language is generally not prefix-closed (even if the timed language is prefix-closed).

2. We showed that a timing-masked language of a timed-automaton is regular, i.e. can be represented by an untimed automaton (over plant events and tick). An algorithm to construct the untimed automaton to accept the timing-masked language was also presented.

3. We formalized the notion of control and timing-mask compatible control policy that issues the control actions based on the observations of evens and their occurrence times

as measured using a finite-precision digital-clock. We showed that such control policy can be equivalently represented as a digitalized-automaton, namely, an untimed automaton that evolves over plant events and clock ticks.

4. We introduced the notion of observability with respect to the partial observation of time resulting from the use of digital-clocks, and showed that the proposed observability together with controllability serves as a necessary and sufficient condition for the existence of a supervisor to enforce a real-time specification on a dense-time discrete event plant. The observability condition proposed in our work (based on extended timing-mask) is different from the one arising due to a partial observation of events since a partial observation of time is in general nondeterministic (the number of ticks generated in any time interval can vary from execution to execution of a digital-clock). Further we examined the lattice structure of a class of timing-masked observable languages, and showed that the proposed observability is not preserved under intersection but preserved under union.

5. We proposed an algorithm to verify the proposed observability and controllability condition for the existence of a supervisor for dense-time DESs using digital-clocks. The algorithm requires a language containment check, which can be reduced to an emptiness check (under the determinism of the timed-automaton of the larger language). When the proposed existence conditions are satisfied, the proposed algorithm presents a method to compute compatible (nonblocking) supervisor.

6. We formalized the notions of dignosability for dense-time DESs under event and timing masks and diagnosability for dense-time DESs with a dense-time specification. The former requires the execution of any fault event can be detected within a bounded time delay from the event and timing-masked observations of the system behavior, whereas the latter requires any violation with respect to the specification can be detected within a bounded time delay in the presence of both event and timing masks.

7. We established an equivalence between diagnosability of a timed DES employing a digital-clock to observe event occurrence times and that of untimed DESs, and showed that the

verification of diagnosability as well as the off-line synthesis of a diagnoser are decidable. The reduction to the untimed setting also suggests an effective method for the off-line computation of a diagnoser as well as its on-line implementation for diagnosis: the results of diagnosis in untimed setting reported in the prior works can be applied to perform diagnosis of a dense-time system against a dense-time specification in the presence of partial observations of events as well as imprecise measurement of time.

8. The results were further extended to diagnosis in the nondeterministic setting, i.e., diagnosis of dense-time DESs using digital-clocks under nondeterministic event observation mask. We introduced the notion of lifting and showed that diagnosis of dense-time DESs in the nondeterministic setting can be reduced to diagnosis of lifted dense-time DESs under deterministic event observations, and hence can be further reduced to diagnosis of lifted untimed DESs in the deterministic setting.

6.2 Summary of Other Research Done during PhD

The research that were conducted and published during my Ph.D. studies but not included in this dissertation are summarized as follows.

6.2.1 Desynchronization

In this work, we studied the problem of "desynchronization", i.e., semantics-preserving "asynchronous implementation" of a "synchronous design". In a synchronous design, system components (which we model as input-output automata (I/O-automata)) communicate over synchronous channels, whereas in an asynchronous implementation, communication among components occurs over asynchronous channels (which we also model as I/O-automata). The presence of asynchronous communication can result in additional behavior that is not present under synchronous communication and can thus cause the semantics of a synchronous design not to be preserved under asynchronous implementation. We presented a framework based on I/O-automata, their compositions and their input/output responses to clearly formulate the notion of correctness: the set of responses to any input sequence must be the same in

the synchronous design and in the asynchronous implementation. We defined the simulation of I/O-automata, and argued that the simulation of the asynchronous implementation by a synchronous design is sufficient to guarantee the correctness of desynchronization. This is a new way of characterizing the correctness of desynchronization (as compared to the "iso-/endo-chrony" type conditions proposed in previous works). Under the practical assumption that the communication delay is bounded, the proposed simulation condition is algorithmically verifiable [143].

6.2.2 Asynchronous Implementation

Discrete event control is typically designed under the synchronous hypothesis that sensing and actuation incur zero delays, i.e., there exists zero delay between an event execution at a plant site and its observation at a controller site, and also between a control computation at a controller site and its enforcement at a plant site. An actual implementation, however, is asynchronous, introducing delays in sensing as well as actuation. A natural question that arises is what additional property must a given specification satisfy so that it remains implementable in spite of the delays introduced by an underlying asynchronous implementation platform. In this work, we formulated the problem of asynchronous implementation of synchronous control when both the sensing and actuation delays are bounded. Compared to our prior work [143] in which the main reason for the loss of semantics is that not all variables are present in all communications and the absence of certain variables can not be detected in asynchronous communication, in this work, the variables are present in all the communications and the reason for the loss of semantics is sensing as well as actuation delays. We introduced the notion of bounded-delay asynchronous composition to characterize the behavior of a controlled plant when the sensing and actuation delays are bounded. We introduced the notion of bounded-delay implementability and showed that this together with the existence conditions of the synchronous setting (namely controllability, closure, and nonemptiness) serves as a necessary and sufficient condition for the existence of a controller so that the controlled behavior under the asynchronous implementation remains the same as that under the synchronous

implementation. We presented an algorithm for checking the property of bounded-delay implementability, whose complexity is linear (resp., quadratic) in the size of the plant (resp., specification), and exponential in the delay bounds. We also examined the lattice structure of a set of bounded-delay implementable languages, and showed that it possesses maximal sublanguages and infimal superlanguages [140].

6.2.3 Distributed State Estimation

Knowledge of the current system state is crucial to many discrete event systems (DESs) applications such as control, diagnosis and prognosis. Due to limited sensing capabilities, the current state information is generally not available and needs to be estimated. In this work, we proposed a novel distributed state estimation algorithm for discrete event plants. Under the proposed algorithm, local sites maintain and update local state estimates based on their local observations of the plant behavior and the observations of the plant behavior sent from the other sites over communication channels with delays. For efficiency of storage, redundant history information about the possible plant evolution is truncated each time a local state estimate is updated. At each local site, the truncation is performed independently requiring no synchronization among the sites. The state estimate maintained at each of the local sites is shown to remain finite regardless of whether the system can execute an unbounded sequence of unobservable events. It is also shown that the proposed algorithm is sound and complete, i.e., each local estimate always contains the true current states (soundness), and it only contains the reachable states of the traces which give rise to a same history of observations (as received from the plant and the other local sites) as does the one executed by the plant (completeness). Also the proposed algorithm can support an architecture in which there is no communication from a certain site to certain other sites [141].

6.2.4 Control under Nondeterministic Event-Mask

In this work, we studied the supervisory control of a discrete event plant under nondeterministic partial observations. This has applicability in scenarios where sensors are unreliable,

thereby introducing nondeterminism in observations. Most prior work on control of discrete event systems assume that sensors are reliable so that they can be modeled as a deterministic (point-valued) observation mask. However in certain harsh environments such as nuclear systems, sensors may not be reliable. This results in a nondeterministic (set-valued) observation mask. In our setting, we allow the observation mask to be nondeterministic to capture unreliability of sensors. We introduced the notion of lifting, and converted the control problem under nondeterministic observations to the control problem of the lifted domain under deterministic observations. We showed that a supervisor exists if and only if exists a lifted specification that satisfies closure, safety, controllability, observability, together with a new property called conformity. The class of such lifted languages is not closed under union or intersection, but is closed under union over an increasing chain. The existence of a nonempty, closed, safe, conforming, controllable and observable lifted-specification serves as a constructive condition for the existence of a supervisor. However, finding such a lifted-specification can be difficult. We proposed a work around this difficulty by replacing the property of observability with the stronger property of normality. The corresponding class of languages is shown to be closed under union, and the nonemptiness of the supremal element serves as a constructive sufficient condition for the existence of a supervisor. We also provided an automaton representation of a control policy based on a nondeterministic observation mask [142].

6.2.5 Network Synthesis

The complexity of networked control/embedded systems is rising due to an increase in the number of nodes that must be interconnected. Therefore it is greatly desirable that automated methods be developed for the synthesis of embedded networks that are cost-effective and correct-by-construction. Since the routes assigned to various connections may overlap, scheduling of connections needs to be considered for network synthesis. In this work, we included the decisions of connection scheduling as part of the network synthesis problem so as to synthesize an embedded network, which not only can satisfy the end-to-end communication requirements, but also can satisfy the scheduling requirements, for networked control sys-

tems. We formulated the scheduling constraints using binary decision variables, and proposed a formal automated approach for the synthesis of a cost-effective and correct-by-construction communication network for embedded applications subject to a set of end-to-end communication constraints of latency, bandwidth, and error-rate, the geometry constraints arising due to a desired geographical placement of the network, and a set of scheduling constraints. We also applied the proposed approach for network synthesis with scheduling to the synthesis of wireless networks for centralized and distributed state estimation in building automation and control [97].

6.3 Future Work

6.3.1 Real-Time Control under Partial Event Observation

In the supervisory control problem, of dense-time discrete event systems using digital-clocks to observe event occurrence times, studied in this dissertation, we did not take the partial observation of events into account. However, in many situations it is difficult, or even impossible, for a supervisor to observe all the plant events due to lack of sensors or their high cost. Further, in certain harsh situations such as nuclear systems and mobile systems, sensors may not be reliable. And so a supervisor may not observe an event deterministically. Such nondeterminism of event observations can be captured by a nondeterministic event observation mask. Due to the nondeterminism of partial observations of time as measured using digital-clocks, the observation under timing-mask as well as deterministic/nondeterministic event-mask is also nondeterministic. The natures of the control problem under complete event observations and partial timing observations (as studied in this dissertation) and that of the control problem under partial event as well as timing-masked observations are quite different. For real-time control under complete event observations and partial time observations, we are able to identify the feasible observations of a timed-trace based on the plant behavior and the given specification: For any untimed observation ν_c of a legal timed-trace ν, if ν_c is shared

by an illegal timed-trace ν', then ν_c could not be a feasible observation of ν. However under partial observations on plant behavior as well as event occurrence times, control of discrete event systems has the additional challenge of deciding which observations of a trace of the specification must be enabled, i.e., the feasible observations. This is not straight forward since in this case an untimed observation of an illegal timed-trace could be shared by a legal timed-trace. Therefore it is difficult to identify the infeasible untimed observations from the violations with respect to the given specification.

Recently, the authors of [126] examined supervisory control of untimed DESs modeled by Mealy Machine in which control actions are determined based on nondeterministic output functions. The existence conditions for two kinds of special supervisors, namely, permissive and anti-permissive supervisors were proposed in [126]. Under permissive supervisors, a control action σ can be enabled following a trace s if there exists a trace s' which is indistinguishable from s such that σ can be enabled following s'. Whereas under anti-permissive supervisors, a control action σ has to be disabled following a trace s if there exists a trace s' which is indistinguishable from s such that the execution of σ following s' is illegal. The existence condition proposed in [126] is only sufficient for the existence of a supervisor. In [142], we investigated supervisory control of untimed DESs modeled by automaton under nondeterministic observation mask. We proposed a necessary and sufficient condition for the existence of a supervisory by converting control in the nondeterministic setting to control in the deterministic setting of *lifted* DESs. The idea of lifting can be applied to study the supervisory control problem of dense-time DESs with nondeterministic event as well as timing observation masks. A possible approach to solve the control problem under partial event and timing mask observations is to first focus on deterministic event observation mask and then extend the research to the nondeterministic setting. For control under nondeterministic event observation, one could lift the plant model and specification language by associating each plant event with its observation, and next check for the existence of a lifted specification which satisfies certain properties such as controllability and observability in the lifted setting. By this means we can convert control under nondeterministic event mask to control of lifted DESs under deterministic event mask.

6.3.2 Bounded Communication Delay

In this dissertation, we assumed that there is no communication delay in the control loop. However many real control systems are physically distributed and hence asynchronous, introducing delays in sensing as well as actuation. And with the rise of the complexity of the real control systems, maintaining global synchrony becomes quite expensive or infeasible. This leads to a disconnection between the design based on the synchronous assumption and a physical asynchronous implementation. Therefore an extended research on control/diagnosis of dense-time DESs using digital-clocks with bounded sensing and actuation delays is necessary. In our previous work [140] we introduced the notion of bounded-delay asynchronous composition to characterize the behaviors of an untimed controlled plant when the sensing and actuation delays are bounded. We assumed that the sensing and actuation delays are bounded by d_1 and d_2 respectively. That is, between the execution of an event by a plant and its observation by a supervisor, the plant can execute at most d_1 more events, whereas between the issuance of a control command by a supervisor and its reception by a plant, the plant can execute at most d_2 more events. And the control computation delay, if any, can be lumped with the actuation delay. The intuition behind the asynchronous composition is as follows: Suppose a plant G under the control of a supervisor S has executed a trace $s \in L(G)$ thus far and has evolved to a state x. Then a prefix t of s such that the length of the suffix trace $s\backslash t$ is at most $d_1 + 1$ would have been observed by supervisor S, and S would have evolved to a state y_1. Owing to the sensing delay, the plant has executed the trace $s\backslash t$ but its observation by the supervisor is still pending. At state y_1, the supervisor enables all events in the set $\Sigma(y_1)$. Due to the delay of actuation, the control witnessed by the plant however lags the one implied by the current state y_1 by at most d_2 steps. The state of S that actually determines the control at the current time is a state y_2 that S would have visited at most d_2 steps in past, when a prefix u of t such that the length of the suffix trance $t\backslash u$ is at most $d_2 + 1$ would have been executed by S. Here the suffix trace $t\backslash u$ denotes the trace that has already executed but whose control influence is yet to be witnessed by G owing to the actuation delay. Thus when at a current state x, plant G witnesses the control implied by the state y_2 of S, i.e., the set of

currently enabled events given by $\Sigma(y_2)$. The notion of asynchronous composition of untimed discrete event systems due to sensing and actuation delays defined in [140] is to capture the aforementioned three kinds of transitions: plant transition, supervisor current state updating transition, and supervisor control state updating transition. In order to formalize the effects of sensing and actuation delays on dense-time discrete event systems, we first need to formalize the controlled behavior of timed discrete event systems under bounded sensing and actuation delays. The notion of asynchronous composition defined in [140] can be extended to the timed setting by replacing the sensing and actuation delays in form of the number of transitions by the real-time delays. With the asynchronous composition in timed setting defined, we can further check whether the sensing and actuation delays can be equivalently aggregated into a single bounded delay for simplification. Then the problem is to study the existence of a supervisor/diagnosor of the dense-time discrete event system with bounded delays. For the control problem, a possible way is to check the existence of a delayed language of the given specification which is controllable and timing-mask observable. And for the diagnosis problem, a possible way is to combine the sensing delay with the behavior of the plant and then convert the diagnosis problem with bounded delay to diagnosis of the delayed plant which captures the plant behavior together with the sensing delay.

6.3.3 Implementable Control/Diagnosis under Finite-Precision Measurement of Time

In this dissertation the control/diagnosis problem we investigate is given a digital-clock and a specification if there exists a supervisor/diagnosor for the given discrete event plant. In practical applications, timed-activities such as computation/communication actions allow for a certain granularity for their occurrence-times and also a certain tolerance-bound. Given this, the clocks used for keeping track of time should be allowed certain granularity and tolerance of drift/jitter. Then an interesting question to ask could be: Given a real-time plant G and specification K, what is the clock-granularity (i.e., precision) and tolerance (i.e., drift/jitter) that can be allowed so that K can be enforced on G under the finite-precision measurement

of time (i.e., remaining controllable and timing-mask observable), or alternatively given a real-time plant G and a finite-precision digital-clock C, which class of specifications K can be implemented over G and C. Such challenging questions would require the analysis of the timed-model of digital-clocks and the flexibility in specification languages. We could first focus on the simple digital-clock that ticks at the rate of one time unit and has a drift of Δ, and then analyze the toleration of a specification, i.e., the duration of enabling/disabling an event.

6.3.4 Decentralized/Distributed Control and Diagnosis

The control and diagnosis problems studied in this dissertation focus on the centralized control/diagnosis architecture. However due to the physically distributed nature of certain systems, decentralized/distributed control/diagnosis may be desired. Therefore an interesting direction of the future research would be to study decentralized/distributed supervisory control and diagnosis problem of dense-time DESs using different digital-clocks. For decentralized/distributed supervisory control under complete event observation and partial time observations, a possible approach is to extend the proposed notion of timing-mask observability to co-observability with respect to the timing masks arising due to the use of different local digital-clock at each local site. For diagnosis under complete event observation and partial time observation, a possible approach is to check the reduction of co-diagnosability of timed DESs using digital-clocks to the untimed setting.

6.3.5 Temporal Logic

Temporal logic provides an effective means of specification. It is easier to specify and more user-friendly than the formal language/automata-based specifications. Temporal logic-based specification can capture the failures representing the violation of both liveness and safety properties, whereas the prior formal language/automaton-based specifications can only capture the failures representing the violation of the safety properties (such as the occurrence of a faulty event or the arrival at a failed state) [56, 51, 44]. An interesting direction of the future works could be to extend the prior work on control and diagnosis to dense-time DESs under

finite-precision timing observations with respect to the specifications in the timed temporal logic setting such as real-time temporal logic RTTL and timed computation tree logic TCTL.

6.3.6 State Explosion

In this dissertation timed-automaton is used to study supervisory control and diagnosis of dense-time DESs using digital-clocks. Timed-automaton suffers from the well known state explosion problem since the size of a timed-automaton is exponential to the number of its clock variables and maximum constant of the constraints. Therefore the computational complexity, especially for the applications to large scale discrete event systems, is an important issue. The approaches based on more computationally efficient representations of region-automaton such as Set-Exp automaton [59] and grid automaton [86] should be explored in the future.

ACKNOWLEDGEMENTS

First and foremost, I would like to express the sincerest and deepest appreciation to my major professor, Dr. Ratnesh Kumar, for his guidance, patience and support throughout my Ph.D. studies. I don't remember how many discussions we have had about my researches and how many revisions he has made for my papers and presentations. But I know behind each of my achievements, there are his directions and efforts. Without his insights and his persistent guidance and help, this dissertation would not have been possible. His rigorous scholarship, strong passion of exploring reasoning, and persistence of pursuing high standards will always inspire me.

I would also like to thank my committee members, Dr. Elia and Dr. Govindarasu from Department of Electrical and Computer Engineering, and Dr. Basu, and Dr. Aduri from Department of Computer Science, for their valuable advices and assistance on this work.

Last but not least, I would like to convey special thanks to my beloved parents and sister, and my friends for their understanding and support.

BIBLIOGRAPHY

[1] Robust and fault-tolerant supervisory control of discrete event systems with partial observation and model uncertainty. *International Journal of System Sciences*, 29(9):953–957, 1998.

[2] K. Altisen, F. Cassez, and S. Tripakis. Monitoring and fault-diagnosis with digital clocks. In *Application of Concurrency to System Design (ACSD'06)*, 2006.

[3] K. Altisen and S. Tripakis. Implementation of timed automata: an issue of semantics or modeling? In *Formal Modeling and Analysis of Timed Systems (FORMATS'05)*, pages 273–288, 2005.

[4] R. Alur. *Techniques for automatic verification of real-time systems.* PhD thesis, Department of Computer Science, Stanford University, 1991.

[5] R. Alur. Timed automata. In *11th International Conference on Computer-Aided Verification*, volume 1633 of *LNCS*, pages 8–22. Springer-Verlag, 1999.

[6] R. Alur and D. Dill. A theory of timed automata. *Theoretical Computer Science*, 126:183–235, 1994.

[7] E. Asarin, O. Maler, and A. Pnueli. Symbolic controller synthesis for discrete and timed systems. In *Hybrid Systems Computation and Control, LNCS 999*, pages 1–10. Springer Verlag, 1995.

[8] P. Astuti and B. J. McCarragher. Optimal control for the convergence of a class of discrete event systems. *International Journal of Control*, 67(6):1029–1046, 1997.

[9] S. Balemi. Communication delays in connections of input/output discrete event processes. In *31st IEEE Conference on Desicion and Control*, pages 3374–3379, Tucson,Arizon, Dec. 1992.

[10] S. Balemi. Input/output discrete event processes and communication delays. *Discrete Event Dynamical Systems: Theory and Applications*, 4(1):41–85, 1994.

[11] G. Barrett and S. Lafortune. Decentralized supervisory control with communicating controllers. *IEEE Transactions on Automatic Control*, 45(9):1620–1638, 2000.

[12] S. Basu and R. Kumar. Quotient based approach to control of nondeterministic discrete-event systems with μ-calculus specification. In *Proceedings of the 45th IEEE Conference on Decision and Control*, pages 6041–6046, San Diego, CA, December 2006.

[13] S. Basu and R. Kumar. Quotient-based control synthesis for partially observed non-deterministic plants with mu-calculus specifications. In *Proceedings of the 46th IEEE Conference on Decision and Control*, pages 5294–5299, New Orleanes, LA, December 2007.

[14] R. K. Boel and J. H. van Schuppen. Decentralized failure diagnosis for discrete-event systems with constrained communication between diagnosers. In *Proceedings of International Workshop on Discrete Event Systems*, 2002.

[15] P. Bouyer, F. Chevalier, and D. D'Souza. Fault diagnosis using timed automata. In *Proceeding of the 8th International Conference on Foundations of Software Science and Computation Structures (FoSSaCS'05)*, Edinburgh, 2005.

[16] B. A. Brandin and W. M. Wonham. Supervisory control of timed discrete event systems. *IEEE Transactions on Automatic Control*, 39(2):329–342, February 1994.

[17] Y. Brave and M. Heymann. Formulation and control of a class of real-time discrete-event processes. Technical Report EE-714, Department of Electrical Engineering, Technion-Israel Institute of Technology, Hafia, Israel, 1989.

[18] Y. Brave and M. Heymann. On optimal attraction in discrete-event processes. *Information Sciences: an International Journal*, 67(3):245–276, 1993.

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

[20] Y.-L. Chen, S. Lafortune, and F. Lin. Design of nonblocking modular supervisors using event priority functions. *IEEE Transactions on Automatic Control*, 45(3), March 2000.

[21] K.-H. Cho and J.-T. Lim. Failure diagnosis and fault tolerant supervisory control systems. *IEICE Transactions on Information and System*, E79-D(9):1223 – 1231, 1996.

[22] R. Cieslak, C. Desclaux, A. Fawaz, and P. Varaiya. Supervisory control of discrete event processes with partial observation. *IEEE Transactions on Automatic Control*, 33(3):249–260, 1988.

[23] O. Contant, S. Lafortune, and D. Teneketzis. Diagnosis of intermittent faults. *Discrete Event Dynamical Systems: Theory and Application*, 14:171–202, 2004.

[24] J. E. R. Cury and B. H. Krogh. Robustness of supervisors for discrete-event systems. *IEEE Transactions on Automatic Control*, 44(2):376–379, 1999.

[25] H. Darabi and M. A. Jafari. Adaptive supervisory control under sensor unavailability. In *Proceedings of 2000 IEEE International Conference on Robotics and Automation*, volume 4, pages 4115 – 4121, 2000.

[26] S. R. Das and L. E. Holloway. Characterizing a confidence space for discrete event timings for fault monitoring using discrete sensing and actuation signals. *IEEE Transactions on Systems, Man, and Cybernetics—Part A: Systems and Humans*, 30(1):52–66, 2000.

[27] R. Debouk, S. Lafortune, and D. Teneketzis. Coordinated decentralized protocols for failure diagnosis of discrete event systems. *Discrete Event Dynamical Systems: Theory and Applications*, 10:33–79, 2000.

[28] R. Debouk, S. Lafortune, and D. Teneketzis. On the effect of communication delays in failure diagnosis of decentralized discrete event systems. In *Proceedings of IEEE Conference on Decision and Control*, pages 2245–2251, Sydney, Australia, December 2000.

[29] R. Debouk, S. Lafortune, and D. Teneketzis. On the effect of communication delays in failure diagnosis of decentralized discrete event systems. *Discrete Event Dynamic Systems: Theory and Applications*, 13(3):263–289, 2003.

[30] M.H. deQueiroz and J.E.R. Cury. Modular supervisory control of large scale discrete-event systems. In *Proc. WODES.00*, page 103.110, 2000.

[31] H. Derbel, M. Yeddes, N.B. Hadj-Alouane, and H. Alla. Diagnosis of a class of timed discrete-event systems. In *Proceedings of the 8th International Workshop on Discrete Event Systems*, Ann Arbor, MI, 2006.

[32] M. Dotoli, M.P. Fanti, and A.M. Mangini. A fault monitor for automated manufacturing systems using a hybrid petri nets formalism. *Transactions of the Institute of Measurement and Control*, 31(19), 2009.

[33] M. Dotoli, M.P. Fanti, A.M. Mangini, and W. Ukovich. On-line fault detection in discrete event systems by petri nets and integer linear programming. *Automatica*, 45:2665–2672, 2009.

[34] E. Dumitrescu, A. Girault, and E. Rutten. Validating fault-tolerant behaviors of synchronous system specifications by discrete controller synthesis. In *IFAC Workshop on Discrete Event Systems, WODES'04*, Reims, France, September 2004.

[35] J. Flochova, P. Kolttrik, and P. Drozd. Modular adaptive approaches in DES failure diagnosis and control. In *Proceedings of 2004 IEEE International Conference on Systems, Man and Cybernetics*, volume 2, pages 1832 – 1836, 2004.

[36] B. Gaudin and H. Marchand. Modular supervisory control of a class of concurrent discrete event systems. In *Workshop on Discrete Event Systems, WODES'04*, September 2004.

[37] A. Gouin and J.L. Ferrier. Supervisory control of timed automata. In *European Control Conference (ECC)*, Karsruhe, Germany, 1999.

[38] T. Henzinger, Z. Manna, and A. Pnueli. What good are digital clocks? *Lecture Notes in Computer Science*, 632:545–558, 1992.

[39] M. Heymann and F. Lin. Nonblocking supervisory control of nondeterministic systems. *Operators, Systems, and Linear Algebra*, pages 96–110, 1997.

[40] M. Heymann and F. Lin. Discrete-event control of nondeterministic systems. *IEEE Transactions on Automatic Control*, 43(1):3–17, January 1998.

[41] L. E. Holloway and S. Chand. Distributed fault monitoring in manufacturing systems using concurrent discrete-event observations. *Integrated Computer-Aided Engineering*, 3(4):244–254, 1996.

[42] J. Huang and R. Kumar. Optimal directed control of discrete event systems. *IEEE Transactions on Automatic Control*, 53(7):1592–1603, 2008.

[43] J. Huang and R. Kumar. Optimal nonblocking directed control of discrete event systems. *IEEE Transactions on Automatic Control*, 53(7):1592–1603, 2008.

[44] Z. Huang, S. Bhattacharyya, V. Chandra, S. Jiang, and R. Kumar. Diagnosis of discrete event systems in rules-based model using first order linear temporal logic. In *Proceedings of 2004 American Control Conference*, pages 5114–5119, Boston, MA, June 2004.

[45] R. M. Jensen. DES controller synthesis and fault tolerant control: A survey of recent advances. Technical Report TR-2003-40, IT University of Copenhagen, 2003.

[46] S. Jiang, Z. Huang, V. Chandra, and R. Kumar. A polynomial time algorithm for diagnosability of discrete event systems. *IEEE Transactions on Automatic Control*, 46(8):1318–1321, 2001.

[47] S. Jiang, Z. Huang, V. Chandra, and R. Kumar. A polynomial time algorithm for diagnosability of discrete event systems. *IEEE Transactions on Automatic Control*, 46(8):1318–1321, 2001.

[48] S. Jiang and R. Kumar. Decentralized supervisory control of concurrent discrete event systems with partial observations. In *Proceedings of 1999 Annual Allerton Conference*, Urbana, IL, September 1999.

[49] S. Jiang and R. Kumar. Supervisory control of nondeterministic discrete event systems with driven events via masked prioritized synchronization. In *1999 IEEE Conference on Decision and Control*, pages 2212–2217, Phoenix, AZ, 1999.

[50] S. Jiang and R. Kumar. Decentralized control of discrete event systems with specializations to local control and concurrent systems. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 30(5):653–660, October 2000.

[51] S. Jiang and R. Kumar. Failure diagnosis of discrete event systems with linear-time temporal logic fault specifications. In *Proceedings of 2002 American Control Conference*, pages 128–133, Anchorage, Alaska, 2002.

[52] S. Jiang and R. Kumar. Supervisory control of nondeterministic discrete event systems with driven events via masked prioritized synchronization. *IEEE Transactions on Automatic Control*, 47(9):1438–1449, 2002.

[53] S. Jiang and R. Kumar. Failure diagnosis of discrete event systems with linear-time temporal logic fault specifications. *IEEE Transactions on Automatic Control*, 49(6):934–945, 2004.

[54] S. Jiang and R. Kumar. Diagnosis of dense-time systems using digital-clocks. In *Proceedings of the 25th American Control Conference*, pages 6051–6056, Minneapolis, MN, June 2006.

[55] S. Jiang and R. Kumar. Diagnosis of repeated failures for discrete event systems with linear-time temporal logic specifications. *IEEE Transactions on Automation Science and Engineering*, 3(1):47–59, 2006.

[56] S. Jiang and R. Kumar. Supervisory control of discrete event systems with CTL* temporal logic specification. *SIAM Journal on Control and Optimization*, 44(6):2079–2103, 2006.

[57] S. Jiang, R. Kumar, and H. E. Garcia. Diagnosis of repeated/intermittent failures in discrete event systems. *IEEE Transactions on Robotics and Automation*, 19(2):310–323, 2003.

[58] A. Khoumsi. Supervisory control of dense real-time discrete-event systems with partial observation. In *6th International Workshop on Discrete Event Systems*, Zaragoza, Spain, October 2002.

[59] A. Khoumsi. A supervisory control method for ensuring the conformance of real-time discrete event systems. *Discrete Event Dynamical Systems: Theory and Applications*, 15:397–431, 2005.

[60] J. Komenda, S. Lahaye, and J.L. Boimond. Control of (max,+) automata: logical and timing aspects. In *Proceedings of Workshop on Discrete Eevent Systems (WODES)*, Gothenburg, Sweden, 2008.

[61] J. Komenda, J.H. van Schuppen, B. Gaudin, and H. Marchand. Modular supervisory control with general indecomposable specification languages. In *Proceedings of the 44th IEEE Conference on Decision and Control, and the European Control Conference 2005*, 2005.

[62] Jan Komenda and Jan H. van Schuppen. Optimal solutions of modular supervisory control problems with indecomposable specification languages. In *Proceedings of the 8th International Workshop on Discrete Event Systems*, Ann Arbor, Michigan, USA, 2006.

[63] M. Krichen and S. Tripakis. Block-box conformance testing for real-time systems. In *Proceedings of 11th International Workshop on Model Checking of Software (SPIN'04)*, volume 2989 of *LNCS*. Springer, 2004.

[64] M. Krichen and S. Tripakis. State identification problems for timed automata. *Lecture Notes in Computer Science*, 3502:175–191, 2005.

[65] R. Kumar and V. K. Garg. Optimal control of discrete event dynamical systems using network flow techniques. In *Proceedings of 1991 Annual Allerton Conference*, pages 705–714, Urbana, IL, October 1991.

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

[67] R. Kumar and V. K. Garg. Optimal supervisory control of discrete event dynamical systems. *SIAM Journal of Control and Optimization*, 33(2):419–439, 1995.

[68] R. Kumar, V. K. Garg, and S. I. Marcus. On controllability and normality of discrete event dynamical systems. *Systems and Control Letters*, 17(3):157–168, 1991.

[69] R. Kumar and M. A. Shayman. Supervisory control of nondeterministic systems under partial observation. In *Proceedings of 1994 IEEE Conference on Decision and Control*, Orlando, FL, December 1994. 3649-3654.

[70] R. Kumar and M. A. Shayman. Nonblocking supervisory control of nondeterministic systems via prioritized synchronization. *IEEE Transactions on Automatic Control*, 41(8):1160–1175, August 1996.

[71] R. Kumar and M. A. Shayman. Supervisory control of real-time systems using prioritized synchronization. In R. Alur, T. Henzinger, and E. Sontag, editors, *Hybrid Systems III*, volume 1066 of *Lecture Notes in Computer Science*. Springer-Verlag, New York, 1996.

[72] R. Kumar and M. A. Shayman. Centralized and decentralized supervisory control of nondeterministic systems under partial observation. *SIAM Journal of Control and Optimization*, 35(2):363–383, March 1997.

[73] R. Kumar and S. Takai. Inference-based ambiguity management in decentralized decision-making: Decentralized diagnosis of discrete event systems. *IEEE Transactions on Automation Science and Engineering*, (3):479–491, 2009.

[74] M. Kwiatkowska, G. Norman, D. Parker, and J. Sproston. Performance analysis of probabilistic timed automata using digital clocks. In *Proceedings of Formal Modeling and Analysis of Timed Systems (FORMATS03)*, pages 105–120, 2003.

[75] Y. Li and W. M. Wonham. Controllability and observability in the state feedback control of discrete event systems. In *Proceedings of the 27th CDC*, pages 203–208, Austin, Texas, December 1988.

[76] F. Lin. A note on optimal supervisory control. In *Proceedings of 1991 IEEE International Symposium on Intelligent Control*, pages 227–232, Arlington, VA, August 1991.

[77] F. Lin. Analysis and synthesis of discrete event systems using temporal logic. *Control Theory and Advanced Technologies*, 9(1):341–350, 1993.

[78] F. Lin. Robust and adaptive supervisory control of discrete event systems. *IEEE Transactions on Automatic Control*, 38(12):1848–1852, 1993.

[79] F. Lin and W. M. Wonham. Decentralized supervisory control of discrete event systems. *Information Sciences*, 44:199–224, 1988.

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

[81] F. Lin and W. M. Wonham. Decentralized control and coordination of discrete-event systems with partial observation. *IEEE Transactions of Automatic Control*, 35(12):1330–1337, December 1990.

[82] F. Lin and W. M. Wonham. Supervisory control of timed discrete event systems under partial observation. *IEEE Transactions on Automatic Control*, 40(3):558–562, 1995.

[83] J.-Y. Lin and D. Ionescu. Verifying a class of nondeterministic discrete event systems in a generalized temporal logic. *IEEE Transactions on Systems, Man and Cybernetics*, 22(6):1461–1469, 1992.

[84] O. Maler, A. Pnueli, and J. Sifakis. On the synthesis of discrete controllers for timed systems. In *12th Annual Symposim of Theoretical Aspects of Computer Science*, Munich, Germany, 1995.

[85] H. Marchand, O. Boivineau, and S. Lafortune. Optimal control of discrete event systems under partial observation. In *Proceedings of IEEE Conference on Decision and Control*, pages 2335–2340, Orlando, FL, 2001.

[86] M. Nourelfath and A. Khoumsi. Grid automata and supervisory control of dense real-time discrete event systems. *Mathematics and Computers in Simulation*, 70:408–418, 2006.

[87] R.C. Olive. Conformance tests for real-time systems with timed automata specifications. *Formal Aspects of Computing*, 12:350–371, 2000.

[88] C. M. Özveren and A. S. Willsky. Observability of discrete event dynamical systems. *IEEE Transactions on Automatic Control*, 35(7):797–806, 1990.

[89] A. Paoli and S. Lafortune. Safe diagnosability for fault-tolerant supervision of discrete event systems. *Automatica*, 41(8):1335–1347, 2005.

[90] S. Park and K. Cho. Delay-robust supervisory control of discrete-event systems with bounded communication delays. *IEEE Transactions on Automatic Control*, 2006.

[91] S. Park and K. Cho. Supervisory control of timed discrete event systems under partial observation based on activity models and eligible time bounds. *Systems Control Letters*, (55):407–413, 2006.

[92] S. Park and K. Cho. Decentralized supervisory control of discrete event systems with communication delays based on conjunctive and permissive decision structures. *Automatica*, (43):738–747, 2007.

[93] S. Park, K. Cho, and J. Lim. Supervisory control of (max,+) automata: A behavioral approach. *Discrete Event Dynamic Systems*, (4):525–549, 2009.

[94] S.-J. Park and J.-T. Lim. Fault-tolerant robust supervisor for discrete event systems with model uncertainty and its application to a workcell. *IEEE Transactions on Robotics and Automation*, 15(2):386–391, 1999.

[95] K. M. Passino and P. J. Antsaklis. Near-optimal control of discrete event systems. In *Proceedings of 1989 Allerton Conference*, pages 915–924, Allerton, IL, September 1989.

[96] K. M. Passino and P. J. Antsaklis. On the optimal control of discrete event systems. In *Proceedings of the 28th IEEE Conference on Decision and Control*, pages 2713–2718, Tampa, FL, December 1989.

[97] A. Pinto, R. Kumar, and S. Xu. Synthesis of wireless time-triggered embedded networks for networked control systems. In *2009 IEEE Conference on Automation Science and Engineering*, Bangalore, India, August 2009.

[98] J. Prosser, M. Kam, and H. Kwatny. Supervisor synthesis for partially-observed discrete event systems. *IEEE Transactions on Automatic Control*, 43(11), 1998.

[99] W. Qiu and R. Kumar. Decentralized nondeterministic supervisory control of discrete event systems. In *Proceedings of IEEE Conference on Decision and Control*, Nassau, Bahama, 2004.

[100] W. Qiu and R. Kumar. Distributed failure diagnosis under bounded delay using immediate observation passing protocol. In *Proceedings of 2005 American Control Conference*, pages 1027–1032, Portland, OR, June 2005.

[101] W. Qiu and R. Kumar. Decentralized failure diagnosis of discrete event systems. *IEEE Transactions on Systems, Man, and Cybernetics—A*, 36(2):384–395, 2006.

[102] W. Qiu and R. Kumar. Distributed diagnosis under bounded-delay communication of immediately forwarded local observations. *IEEE Transactions on Systems, Man, and Cybernetics — Part A*, 38(3):628–643, 2008.

[103] W. Qiu, R. Kumar, and V. Chandra. Decentralized control of discrete event systems using prioritized composition with exclusion. *IEEE Transactions on Automatic Control*, 53(10):2425–2430, 2006.

[104] W. Qiu, R. Kumar, and S. Jiang. On decidability of distributed diagnosis under unbounded-delay communication. *IEEE Transactions on Automatic Control*, 52:114–116, January 2007.

[105] M. H. Queiroz and J. E. R. Cury. Modular control of composed systems. In *Proc. of 2000 American Control Conference*, 2000.

[106] P. J. Ramadge and W. M. Wonham. Supervisory control of a class of discrete event processes. *SIAM Journal of Control and Optimization*, 25(1):206–230, 1987.

[107] P. J. Ramadge and W. M. Wonham. The control of discrete event systems. *Proceedings of IEEE: Special Issue on Discrete Event Systems*, 77:81–98, 1989.

[108] S. L. Ricker and J. H. van Schuppen. Decentralized failure diagnosis with asynchronous communication between supervisors. In *Proceedings of the European Control Conference*, pages 1002–1006, 2001.

[109] K. Rohloff and S. Lafortune. Supervisor existence for modular discrete-event systems. In *Proceedings of the 2nd IFAC Conference on Control Systems Design*, Bratislava, Slovak Republic, September 2003.

[110] K. R. Rohloff. Sensor failure tolerant supervisory control. In *Proceedings of the 44th IEEE Conference on Decision and Control, and the European Control Conference 2005*, pages 3493 – 3498, 2005.

[111] K. Rudie and J. C. Willems. The computational complexity of decentralized discrete-event control problems. *IEEE Transactions on Automatic Control*, 40(7):1313–1319, July 1995.

[112] K. Rudie and W. M. Wonham. Think globally, act locally: decentralized supervisory control. *IEEE Transactions on Automatic Control*, 37(11):1692–1708, November 1992.

[113] A. Saboori and S. H. Zad. Robust nonblocking supervisory control of discrete-event systems under partial observation. *Systems Control Letters*, (55):839–848, 2006.

[114] M. Sampath, R. Sengupta, S. Lafortune, K. Sinaamohideen, and D. Teneketzis. Diagnosability of discrete event systems. *IEEE Transactions on Automatic Control*, 40(9):1555–1575, September 1995.

[115] R. Sengupta and S. Lafortune. A graph-theoretic optimal control problem for terminating discrete event processes. *Discrete Event Dynamic Systems: Theory and Applications*, 2(2):139–172, 1992.

[116] M. A. Shayman and R. Kumar. Supervisory control of nondeterministic discrete event dynamical systems. In *Proceedings of 1993 IEEE Conference on Decision and Control*, pages 1188–1193, San Antonio, TX, December 1993.

[117] M. A. Shayman and R. Kumar. A new framework for supervisory control. In *Proceedings of 1995 American Control Conference*, pages 1341–1345, Seattle, WA, June 1995.

[118] P. Tabuada and G. J. Pappas. Linear time logic control of discrete-time linear systems. *IEEE Transactions on Automatic Control*, (12):1862–1877, 2006.

[119] S. Takai and R. Kumar. Decentralized diagnosis for nonfailures of discrete event systems using inference-based ambiguity management. *IEEE Transactions on Systems, Man, and Cybernetics—Part A*. Accepted.

[120] D. Thorsley and D. Teneketzis. Diagnosability of stochastic discrete-event systems. *IEEE Transactions on Automatic Control*, 50(4):476–498, 2005.

[121] D. Thorsley, T. S. Yoo, and H. E. Garcia. Diagnosability of stochastic discrete-event systems under unreliable observations. In *Proceedings of American Control Conference*, 2008.

[122] S. Tripakis. Decentralized control of discrete event systems with bounded or unbounded delay communication. In *Proceedings of 2002 International Workshop on Discrete Event Systems*, pages 2–4, October 2002.

[123] S. Tripakis. Fault diagnosis for timed automata. In *Formal Techniques in Real Time and Fault Tolerant Systems*, volume 2469 of *Lecture Notes in Computer Science*. Springer Verlag, 2002.

[124] S. Tripakis. Decentralized control of discrete-event systems with bounded or unbounded delay communication. *IEEE Transactions on Automatic Control*, 49(9):1489–1501, 2004.

[125] S. Tripakis and K. Altisen. On-the-fly controller synthesis for discrete and dense-time systems. In *World Congress on Formal Method*, Toulouse, France, 1999.

[126] T. Ushio and S. Takai. Supervisory control of discrete event systems modeled by mealy automata with nondeterministic output functions. In *2009 American Control Conference*, St. Louise, MO, 2009.

[127] Y. Wang, T.-S. Yoo, and S. Lafortune. Decentralized diagnosis of discrete event systems using unconditional and conditional decisions. In *Proceedings of the 44th IEEE Conference on Decision and Control and 2005 European Control Conference*, pages 6298–6304, Seville, Spain, December 2005.

[128] Q. Wen, J. Huang, and R. Kumar. Synthesis of optimal fault-tolerant supervisor for discrete event systems. In *2008 American Control Conference*, Seattle, WA, June 2008.

[129] Q. Wen, R. Kumar, J. Huang, and H. Liu. Weakly fault-tolerant supervisory control of discrete event systems. In *Proceedings of the 26th American Control Conference*, pages 5649–5654, New York, NY, July 2007.

[130] Q. Wen, R. Kumar, J. Huang, and H. Liu. A framework for fault-tolerant control of discrete event systems. *IEEE Transactions on Automatic Control*, 53(9):1839–1849, 2008.

[131] Y. Willner and M. Heymann. Supervisory control of concurrent discrete-event systems. *International Journal of Control*, 54(5):1143–1169, 1991.

[132] K. C. Wong, J. G. Thistle, R. P. Malhame, and H.-H. Hoang. Supervisory control of distributed systems: Conflict resolution. In *Proceedings of IEEE Conference on Decsision and Control*, pages 3275–3280, Tampa, FL, 1998.

[133] K. C. Wong and J.H. van Schuppen. Decentralized supervisory control of discrete-event systems with communication. In *Proc. of WODES*, 1996.

[134] K. C. Wong and W. M. Wonham. Modular control and coordination of discrete event systems. *Discrete Event Dynamical Systems: Theory and Applications*, 8:247–297, 1998.

[135] K. C. Wong and W.M. Wonham. Modular control and coordination of discrete-event systems. *Journal of Discrete Event Dynamical Systems: Theory and Applications*, 1998.

[136] H. Wong-Toi and G. Hoffmann. The input-output control of real-time discrete event systems. In *Proceedings of 13th IEEE Real-Time Systems Symposium*, 1992.

[137] W. M. Wonham and P. J. Ramadge. On the supremal controllable sublanguage of a given language. *SIAM Journal of Control and Optimization*, 25(3):637–659, 1987.

[138] M.D. Wulf, L. D., N. Markey, and J.F. Raskin. Robustness and implementability of timed automata. In *Joint Conference Formal Modelling and Analysis of Timed Systems and Formal Techniques in Real-Time and Fault Tolerant System*, pages 118–133, 2004.

[139] M.D. Wulf, L. Doyen, and J.F. Raskin. Almost asap semantics: From timed models to timed implementations. *Hybrid Systems: Computation and Control, Lecture Notes in Computer Science*, 2993:296–310, 2004.

[140] S. Xu and R. Kumar. Asynchronous implementation of synchronous discrete event control. In *2008 Workshop on Discrete Event Systems*, Gotenborg, Sweden, May 2008.

[141] S. Xu and R. Kumar. Distributed state estimation in discrete event systems. In *2009 American Control Conference*, St. Louis, MO, June 2009.

[142] S. Xu and R. Kumar. Supervisory control of discrete event systems under nondeterministic partial observation. In *2009 IEEE Conference on Automation Science and Engineering*, Bangalore, India, August 2009.

[143] S. Xu, R. Kumar, S. Jiang, and S. Ramesh. A simulation condition for correct desynchronization. In *Proceedings of the 27th American Control Conference*, Seattle, WA, June 2008.

[144] T. Yoo and H. E. Garcia. Event diagnosis of discrete-event systems with uniformly and nonuniformly bounded diagnosis delays. In *Proceedings of 2004 American Control Conference*, pages 5102–5107, Boston, MA, June 2004.

[145] T. Yoo and S. Lafortune. A generalized framework for decentralized supervisory control of discrete event systems. In *Proceedings of 2000 International Workshop on Discrete Event Systems*, Ghent, Belgium, 2000.

[146] T. S. Yoo and S. Lafortune. Polynomial-time verification of diagnosability of partially observed discrete-event systems. *IEEE Transactions on Automatic Control*, 47(9):1491–1495, 2002.

[147] T. S. Yoo and S. Lafortune. Decentralized supervisory control with conditional decisions: Supervisor existence. *IEEE Transactions on Automatic Contrl*, 49(11):1886–1904, 2004.

[148] S. Young, D. Spanjol, and V. K. Garg. Control of discrete event systems modeled with deterministic Buchi automata. In *Proceedings of 1992 American Control Conference*, pages 2809–2813, Chicago, IL, 1992.

[149] S. H. Zad, R. H. Kwong, and W. M. Wonham. Fault diagnosis in discrete-event systems: Incorporating timing information. *IEEE Transactions on Automatic Control*, 50(7):1010–1015, 2005.

[150] C. Zhou and R. Kumar. Control of nondeterministic discrete event systems for simulation equivalence. *IEEE Transactions on Automation Science and Engineering*, 4:340–349, July 2007.

[151] C. Zhou and R. Kumar. A small model theorem for bisimilarity control under partial observation. *IEEE Transactions on Automated Science and Engineering*, 4(1):93–97, January 2007.

[152] C. Zhou and R. Kumar. Prioritized synchronization under mask for control and interaction of partially observed event-driven systems. *IEEE Transactions on Automation Science and Engineering*, 5:101–112, January 2008.

[153] C. Zhou and R. Kumar. Computation of diagnosable fault-occurrence indices for systems with repeatable-faults. *IEEE Transactions on Automatic Control*, (7):1477–1489, 2009.

[154] C. Zhou, R. Kumar, and R. S. Sreenivas. Decentralized modular control of concurrent discrete event systems. In *Proceedings of the 46th IEEE Conference on Decision and Control*, pages 5918–5923, New Orleanes, LA, December 2007.