A New Framework for Discrete-Event Systems

Other titles in Foundations and Trends® in Systems and Control

Adaptive Internal Models in Neuroscience
Mireille E. Broucke
ISBN: 978-1-68083-940-1

Finite-Time Stability Tools for Control and Estimation
Denis Efimov and Andrey Polyakov
ISBN: 978-1-68083-926-5

Generalized Coordination of Multi-robot Systems
Kazunori Sakurama and Toshiharu Sugie
ISBN: 978-1-68083-902-9

Analysis and Control for Resilience of Discrete Event Systems: Fault Diagnosis, Opacity and Cyber Security
João Carlos Basilio, Christoforos N. Hadjicostis and Rong Su
ISBN: 978-1-68083-856-5

Learning-Based Control: A Tutorial and Some Recent Results
Zhong-Ping Jiang, Tao Bian and Weinan Gao
ISBN: 978-1-68083-752-0

Synchronous Reinforcement Learning-Based Control for Cognitive Autonomy
Kyriakos G. Vamvoudakis and Nick-Marios T. Kokolakis
ISBN: 978-1-68083-744-5

A New Framework for Discrete-Event Systems

Kuize Zhang
University of Surrey
kuize.zhang@surrey.ac.uk

the essence of knowledge
Boston — Delft

Foundations and Trends® in Systems and Control

Published, sold and distributed by:
now Publishers Inc.
PO Box 1024
Hanover, MA 02339
United States
Tel. +1-781-985-4510
www.nowpublishers.com
sales@nowpublishers.com

Outside North America:
now Publishers Inc.
PO Box 179
2600 AD Delft
The Netherlands
Tel. +31-6-51115274

The preferred citation for this publication is

K. Zhang. *A New Framework for Discrete-Event Systems*. Foundations and Trends® in Systems and Control, vol. 10, no. 1-2, pp. 1–179, 2023.

ISBN: 978-1-63828-153-5

Editorial Scope

Topics

Foundations and Trends® in Systems and Control publishes survey and tutorial articles in the following topics:

- Control of:
  - Hybrid and Discrete Event Systems
  - Nonlinear Systems
  - Network Systems
  - Stochastic Systems
  - Multi-agent Systems
  - Distributed Parameter Systems
  - Delay Systems
- Filtering, Estimation, Identification
- Optimal Control
- Systems Theory
- Control Applications

Information for Librarians

Foundations and Trends® in Systems and Control, 2023, Volume 10, 4 issues. ISSN paper version 2325-6818. ISSN online version 2325-6826. Also available as a combined paper and online subscription.

Contents

A New Framework for Discrete-Event Systems

Kuize Zhang

University of Surrey, UK; kuize.zhang@surrey.ac.uk

ABSTRACT

Real-world problems are often formulated as diverse properties of different types of dynamical systems. Hence property verification and synthesis (i.e., enforcement) have been long-standing research interests. The motivations of writing this monograph lie in two aspects. First, we will develop an open-loop property enforcement framework for discrete-event systems. Second, we will propose a new model — labeled weighted automata over monoids.

The supervisory control framework initialized by Ramadge, Wonham, and Lin in the 1980s provides a closed-loop property enforcement framework for discrete-event systems which usually consist of discrete states and transitions between states caused by spontaneous occurrences of labeled (i.e., partially-observed) events. This framework can be fully realized in labeled finite-state automata (LFSAs). Plenty of theoretical and applied results under this framework have been obtained during the past three decades. However, there are several drawbacks in this framework which restrict the application of the framework to large-scale systems, e.g., all enforceable properties can be enforced in LFSAs in at least *exponential time*, showing that the enforcement algorithms in this framework do not scale well; this framework

---

Kuize Zhang (2023), "A New Framework for Discrete-Event Systems", Foundations and Trends® in Systems and Control: Vol. 10, No. 1-2, pp 1–179. DOI: 10.1561/2600000028.

cannot be fully realized in more complicated models such as labeled Petri nets and labeled timed automata, because supervisors/controllers in such models are generally not computable (with any complexity upper bound), which narrows the application range of this framework. In this monograph, we will develop an open-loop property enforcement framework for discrete-event systems which scales better and can be implemented in more models.

In order to implement this new framework, we develop a tool called *concurrent composition*, and use this tool to unify plenty of *inference-based* properties (e.g., detectability, diagnosability, predictability) and *concealment-based* properties (e.g., various notions of opacity) in discrete-event systems.[1] The negations of such properties are equivalently represented by the existence of special runs in the concurrent composition of two variants of a plant. Then, a property of interest can be enforced by choosing controllable events/transitions to disable in order to cut off all such runs violating the property. Our open-loop framework can be implemented in LFSAs in *polynomial time* for polynomially verifiable properties (e.g., strong detectability, diagnosability, predictability), and can also be fully realized (at least) in labeled Petri nets and labeled timed automata for decidable inference-based and concealment-based properties.

In the second aspect, we propose a new model called *labeled weighed automata over monoids* (LWAMs). LWAMs provide a natural generalization of LFSAs in the sense that each transition therein carries a weight from a monoid, the weight of a run (a sequence of consecutive transitions) is the product of the weights of the run's transitions. When weights are nonnegative real numbers, they could be interpreted as the time

[1]In the past, these inference-based properties were verified by using different methods, and based on two fundamental assumptions of deadlock-freeness (a plant will always run) and divergence-freeness (the running of a plant will always be eventually observed).

consumptions of the transitions' executions, so that LWAMs could be regarded as real-time systems. When weights are real vectors (hence the entries of the vectors could be negative), they can be interpreted as position deviations of a moving object along with the transitions' executions. We develop original techniques to compute three basic tools — *concurrent composition*, *observer*, and *detector* in LWAMs, and then design algorithms for verifying various notions of detectability. The research in LWAMs has just started. With these three tools, plenty of results in LFSAs obtained in the past three decades can be extended to LWAMs, including results on inference-based properties and concealment-based properties, as well as results obtained in the supervisory control framework. Our open-loop property enforcement framework, of course, can be fully implemented in LWAMs. Compared with LFSAs, LWAMs provide more accurate modeling scheme, hence have more applications. A challenging future direction lies in extending the formal verification and synthesis framework of cyber-physical systems from the core part of LFSAs-based to LWAMs-based.

1

Background and Motivation

1.1 Background

In the 1980s, P. Ramadge, W. Wonham, and F. Lin initialized the so-called *supervisory control framework* (Ramadge and Wonham, 1987; Lin and Wonham, 1988), which extends the analysis and synthesis framework from control systems (normally differential equations) to computer systems (formal languages) which are called *discrete-event systems* (DESs), where the counterparts of all kinds of (e.g., controllable or observable) subspaces in control systems are diverse sublanguages of formal languages. DESs usually consist of discrete states and transitions between states caused by spontaneous occurrences of labeled (aka partially-observed) events, and the formal languages of interest are the sets of label/output sequences generated by DESs. A transition is represented by the form $q_1 \xrightarrow{e(\sigma)} q_2$, indicating that when a DES is in state q_1 and event e occurs, the DES transitions to state q_2,[1] σ is the label/output of e, i.e., the observation when e occurs, particularly $\sigma = \epsilon$[2] implies e is unobservable. Hence DESs are autonomous (i.e., not

[1] q_2 need not be different from q_1.

[2] As usual, ϵ denotes the empty string.

driven by external factors) and nonlinear (Wonham and Cai, 2019). The supervisory control framework provides a controller synthesis method in a *closed-loop manner*. It tracks a sequence of observed outputs generated by a given DES, does state estimation according to the outputs, and meanwhile synthesizes control policies (called a *supervisor*) to restrict the behavior of the DES such that the modified DES satisfies a property of interest that the original DES does not satisfy. Hence, one prerequisite is that the property of interest is decidable, i.e., there is an algorithm for verifying the property. The verification problem is also called the analysis problem. The synthesis problem is also called the enforcement problem, i.e., for a DES $\mathcal{S}$ and a property P, $\mathcal{S}$ does not satisfy P, one modifies $\mathcal{S}$ in order to make it enforce P.

1.2 The First Motivation

During the past three decades, plenty of interesting properties with their variants in DESs have been proposed, investigated, and applied to many different areas such as heating, ventilation, and air conditioning (HVAC), traffic networks, automated manufacturing, tracking of mobile agents in sensor networks, etc. It is exciting that so many results have been obtained and different properties have remarkably different physical meanings, e.g., detectability implies that one *can* determine the current and subsequent states using observed output sequences (Shu *et al.*, 2007), diagnosability implies that one *can* determine the past occurrences of faulty events, using the observed output sequences (Sampath *et al.*, 1995), state-based opacity implies that one *cannot* determine the visit to some secret state, also by using the observed output sequences (Saboori and Hadjicostis, 2007). However, different properties have been verified using different methods and it is not known whether there are essential differences between them, particularly from a mathematical point of view. A first motivation of writing this monograph is to *perform subtractions on* DESs, i.e., using a streamlined mathematical framework to unify as many as properties, although they have diverse physical meanings. The first target of the monograph is to unify all properties without essential differences into one mathematical framework. Note that such subtractions will not reduce the existing realms and realms of results obtained in DESs, actually they will make the contents of DESs more *tidy*.

We develop a mathematical tool that we named *concurrent composition*[3] to implement the first target. Intuitively speaking, the concurrent composition of two labeled systems $\mathcal{S}_l$ and $\mathcal{S}_r$ aggregates any pair of a run[4] of $\mathcal{S}_l$ and a run of $\mathcal{S}_r$ with the same observation; the observations in any pair of their runs will be synchronized and their unobservable transitions will interleave. The concurrent composition will provide a unified mathematical framework for most *inference-based* properties and *concealment-based* properties. By inference-based we mean a property indicating that one can get further internal information from observations, e.g., detectability, diagnosability, predictability, etc. By concealment-based we mean a property indicating that one cannot get further internal information from observations, e.g., opacity. A preliminary work along this line in LFSAs refers to Zhang (2021a).

Because of the partially-observed feature of DESs, the properties therein can be naturally classified into the two basic categories of inference-based and concealment-based, other properties can be seen as variants of the properties in the two categories. In order to verify an inference-based property, we represent its *negation* as the existence of special runs in the corresponding concurrent composition, the rest is to check the existence of the special runs (see Section 2.3 and Zhang, 2021a). The advantages of this approach in LFSAs with respect to verification are two-fold. Firstly, it provides polynomial-time verification algorithms for most known inference-based properties except for those whose verification problems have been proven PSPACE-hard,[5] e.g., weak detectability (Zhang, 2017), strong periodic D-detectability (Balun and Masopust, 2021b). Secondly, it does not depend on any assumption. Note that the widely-used verification algorithms in the literature for verifying inference-based properties such as the detector method for

[3] Its form in labeled finite-state automata (LFSAs, as in Definition 2.5) is shown in Definition 2.6.

[4] A run is a sequence of transitions in which the terminating state of each transition is the same as the starting state of its very next transition.

[5] It is widely conjectured that a PSPACE-hard problem cannot be solved by any polynomial-time algorithm (Sipser, 1996), see Page 13.

strong detectability (Shu and Lin, 2011b), the twin-plant method (Jiang *et al.*, 2001) and the verifier method (Yoo and Lafortune, 2002) for diagnosability, and the verifier method (Genc and Lafortune, 2009) for predictability, though also run in polynomial time, all depend on two fundamental assumptions of deadlock-freeness (also called liveness, which means that an automaton will always run) and divergence-freeness (i.e., an automaton has no reachable unobservable transition cycle, which means that the running of an automaton will always be eventually observed). The reason lies in the fact that these methods were used to verify the properties themselves but not their negations. See Section 2.3.4 for detailed analysis. In order to verify a concealment-based property in an LFSA $\mathcal{A}$, e.g., opacity, we first compute the concurrent composition of $\mathcal{A}$ and its *observer*[6] (see Definition 2.7), and then check the reachability of some special state in the concurrent composition. Note that this idea of verification is directly derived from various definitions of opacity, and the derived algorithms are currently the most efficient (see Section 2.4).

In addition, with respect to verification, one major advantage of concurrent composition lies in that it can be extended to models that are more general than LFSAs, e.g., labeled weighted automata over monoids (Zhang, 2022), labeled Petri nets (Zhang and Giua, 2020b; Zhang *et al.*, 2020), and so on. The study on labeled weighted automata over monoids has just started. Welcome more and more researchers to join in this new research direction.

1.3 The Second Motivation

Before introducing a second motivation of writing this monograph, we recall the overall procedure of supervisor synthesis in the supervisory control framework (Ramadge and Wonham, 1987; Lin and Wonham, 1988; Wonham and Cai, 2019; Cassandras and Lafortune, 2008) (see Figure 1.1 for an illustration). Recall that the event set in a DES is an alphabet, i.e., a nonempty finite set E such that every sequence of elements of E has a unique decomposition of elements of E. For example,

[6]Actually the standard powerset/subset construction for determinizing a nondeterministic finite automaton with ε-transitions (Rabin and Scott, 1959; Sipser, 1996).

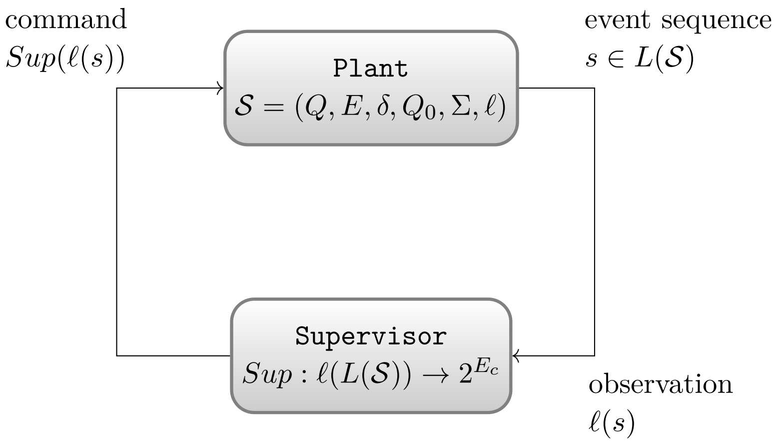

Figure 1.1: A sketch for suprevisory control.

$\{a, b\}$ and $\{0, 01\}$ are alphabets, but $\{0, 00\}$ is not, because $000 = 0\ 00 = 00\ 0$. An event set must be an alphabet because this guarantees that every generated event sequence cannot have two different interpretations. Consider $\{0, 00\}$ as a counterexample, if 000 were generated, then there are two interpretations: (1) 0 was first generated and then 00 was generated, or (2) 00 was first generated and then 0 was generated. Also recall that an event set E can be partitioned into two disjoint subsets E_c and E_{uc}, denoted by $E = E_c \dot\cup E_{uc}$, where E_c denotes the set of *controllable events* and E_{uc} the set of *uncontrollable events*. The occurrence of a controllable event can be forbidden, but the occurrence of an uncontrollable event cannot. Given a formal language L[7] and a DES G as one of its generators[8] such that G does not satisfy a property P of interest, one tracks an observed output sequence γ generated by G, does state estimate SE_γ according to γ, and then uses SE_γ to compute a subset C_{SE_γ} of controllable events that are in the transitions starting from the states of SE_γ. A supervisor $S : L \to 2^{E_c}$[9] is a mapping that sends an observed output sequence γ of L to C_{SE_γ} ($\subset E_c$).

[7]Defined by a subset of E^*, where E^* denotes the set of finitely long strings of elements of E.

[8]The set of finitely long output sequences generated by G is equal to L.

[9]The powerset of E_c, i.e., $2^{E_c} = \{E'_c | E'_c \subset E_c\}$.

The supervisor works in this way: whenever the current-state estimate is SE_γ (no matter the current observed output sequence is γ or not, i.e., there may exist different γ, γ' such that $SE_\gamma = SE_{\gamma'}$), one dynamically disables several controllable events of C_{SE_γ} to restrict the behavior of G, so that the closed-loop system (G, S) satisfies the property P. From the procedure, one can see that if G has finitely many states, then the supervisor S usually can be fully computed even if L is infinite, because during computation one can partition L into a finite number of disjoint nonempty subsets such that two sequences $\gamma, \gamma' \in L$ belong to the same subset if and only if $SE_\gamma = SE_{\gamma'}$ (in this case, S send them to the same subset of E_c). In this sense, S is a *nondeterministic finite automaton* (see Definition 2.2). However, if G has infinitely many states, usually the supervisor S cannot be fully computed, i.e., the supervisor control cannot be fully realized. In DESs, the widely-used models such as finite automata, Petri nets, timed automata, etc., have finitely many events, because their event sets are always alphabets; however, Petri nets and timed automata may have infinitely many states. As a result, although the supervisory control framework is a methodology that owns abundant intension, it is somehow air-castle. For the models that are more complicated than finite automata, generally the supervisory control cannot be fully realized. *The second motivation of writing this monograph is to develop a new synthesis framework that is applicable to remarkably larger classes of models that can represent DESs.*

Our new property synthesis framework is open-loop, i.e., one does not supervise output sequences generated by a DES as is done in the supervisory control framework. The philosophy of our framework is as follows (see Figure 1.2 for an illustration): (1) the negation of a property of interest is equivalently represented by the existence of special runs in the corresponding concurrent composition of a variant of the plant and another variant of the plant, (2) controllable events/transitions[10] are chosen to be disabled so that all these special runs violating the property will disappear. This synthesis method is quite simple and easily realizable on decidable properties of DESs, as there are only finitely

[10] A controllable transition is a transition whose event is controllable.

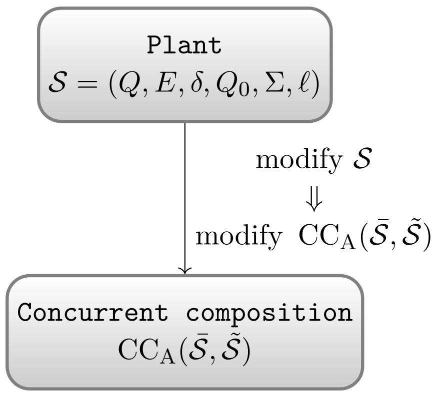

Figure 1.2: A sketch for the open-loop control.

many events. One can also easily check whether a property is enforceable: disabling all controllable events and then see whether the modified DES satisfies the property. If no, then the property of the DES cannot be enforceable in a large extent; otherwise, one can enforce the property by choosing several controllable events to disable according to specific scenarios. Of course, although for several DESs the property could be enforced after all controllable events being disabled, the remainder of such DESs might not be interesting any more because they might lose several interesting behaviors. Therefore, a better way is to choose as few as controllable events to disable. To be more refined, one can also choose concrete controllable transitions to disable instead of controllable events (in the latter coarser case, if one controllable event is disabled, then all transitions with the event will be disabled). In finite automata, there are finitely many transitions, so the synthesis method via choosing controllable transitions can be fully realized. However, for systems with infinitely many transitions such as unbounded Petri nets, the focus should be on controllable events or a finite subset of controllable transitions. To sum up, our open-loop framework will *perform additions* to DESs, because it provides a new property synthesis framework on dramatically larger classes of systems compared with the supervisory control framework. The second target of writing the monograph is to implement our open-loop property synthesis framework.

1.4 The Third Motivation

As mentioned above, DESs have LFSAs as their basic model. The supervisory control framework can be fully realized in LFSAs, but cannot be fully realized in models that are more general than LFSAs in general, because their observers are usually not computable, e.g., labeled timed automata and labeled Petri nets. A natural question to ask is: Are there a class of systems that are more general than LFSAs but the supervisory control framework can be fully realized in the class? This is almost equivalent to ask: Are there a class of systems that are more general than LFSAs but their observers are computable?

On the other hand, although LFSAs are the basic model of DESs, they do not show sufficiently accurate modeling. For example, when doing state estimation based on a sequence γ of observed labels, the time consumptions for the executions of unobservable transitions were usually assumed to be zero (Sampath *et al.*, 1995; Shu *et al.*, 2007). Timed automata are a natural generalization of finite automata in the sense that the executions of transitions are constrained by time intervals with rational endpoints. However, the observers of labeled timed automata are usually not computable.

Based on the above two points, it is very meaningful to find a class of systems that are more general than LFSAs but their observers are computable. It is challenging to do so. Consider a run $q_0 \xrightarrow{e_1/t_1} q_1 \xrightarrow{e_2/t_2} \cdots \xrightarrow{e_n/t_n} q_n$, in which after each "/" there is a weight for the corresponding transition. If the weight of a transition therein is considered as its time consumption, then the weight of the run is equal to $\sum_{i=1}^{n} t_i$. This semantics is similar to that in timed automata. Differently, here we consider a general monoid $\mathfrak{M} = (T, \otimes, \mathbf{1})$, where $\otimes$ is an associative binary operation on T and $\mathbf{1} \in T$ is an identity element. We extend LFSAs in the sense that each of its transitions carries a weight in a monoid and the weight of a run is the product of the weights of the transitions of the run. When $\mathfrak{M}$ is specified as $(\mathbb{R}_{\geq 0}, +, 0)$, where $\mathbb{R}_{\geq 0}$ denotes the set of nonnegative real numbers as usual, the weights can represent the time consumptions of the corresponding transitions and then the extended LFSA can represent a real-time system; while $\mathfrak{M}$ is specified as $(\mathbb{R}^n, +)$, where $\mathbb{R}^n$ denotes the set of n-dimensional real vectors, the weights can represent position deviations along with the transitions. That is, the weights have diverse physical meanings. In Section 5, we will prove that the observers of this kind of extended LFSAs over the monoid $(\mathbb{Q}^n, +)$ are computable by developing original computing techniques, where $\mathbb{Q}^n$ denotes the set of n-dimensional rational vectors. A general theory of the extended LFSAs over monoids will be given in Section 4, where the new class of automata are called *labeled weighted automata over monoids* (LWAMs).

1.5 Structure of the Monograph

In Section 2, we show the implementation of our unified concurrent-composition framework for DESs modeled by LFSAs as well as our open-loop property synthesis framework, in a centralized setting. In Section 3 we show the implementation for LFSAs in a decentralized setting. In Section 4, we will propose the new model — LWAMs. We will formulate the basic tools of concurrent composition, observer, and detector for LWAMs, and use them to derive necessary and sufficient conditions for several strong versions of detectability and weak versions of detectability. Particularly in Section 5, for labeled weighted automata (LWAs) over the monoid $(\mathbb{Q}^n, +)$, we will develop original methods (that can be implemented algorithmically) to compute the three basic tools, and hence prove that the necessary and sufficient conditions obtained in Section 4 are algorithmically implementable. Thus, our concurrent-composition framework and open-loop property synthesis framework will be fully extended to LMAs over the monoid $(\mathbb{Q}^n, +)$. Section 6 shows a brief outlook on the implementation of the open-loop property enforcement framework in labeled Petri nets and labeled timed automata.

1.6 Notation

Symbols $\mathbb{N}, \mathbb{Z}, \mathbb{Q}, \mathbb{Q}_{\geq 0}, \mathbb{R}$, and $\mathbb{R}_{\geq 0}$ denote the sets of nonnegative integers, integers, rational numbers, nonnegative rational numbers, real numbers, and nonnegative real numbers, respectively. $\mathbb{R}^n$ denotes the set of n-dimensional real column vectors. The symbol $(\cdot)^n$ also applies to the other sets of numbers. 0_n denotes the n-dimensional column vector with all entries 0. $[\![m, n]\!]$ denotes the set of integers no less than m and no greater than n. A finite nonempty set Σ is called an *alphabet* if every sequence of elements of Σ is a unique sequence of elements of Σ. For example, $\{0, 00\}$ is not an alphabet since $000 = 0\ 00 = 00\ 0$. For an alphabet Σ, elements of Σ are called *letters*, Σ^* and Σ^ω are used to denote the set of *words/strings* (i.e., finite-length sequences of elements of Σ) over Σ including the empty word ϵ and the set of *configurations* (i.e., infinite-length sequences of elements of Σ) over Σ, respectively. $\Sigma^+ := \Sigma^* \setminus \{\epsilon\}$. For a word $s \in \Sigma^*$, $|s|$ stands for its length, and we set $|s'| = +\infty$ for all $s' \in \Sigma^\omega$. For $s \in \Sigma^+$ and $k \in \mathbb{N}$, s^k and s^ω denote the concatenations of k copies of s and infinitely many copies of s, respectively. Analogously, the concatenation of two languages L_1 and L_2 is defined by $L_1L_2 := \{e_1e_2 | e_1 \in L_1, e_2 \in L_2\}$, where $L_1, L_2 \subset \Sigma^*$. For a word (configuration) $s \in \Sigma^*(\Sigma^\omega)$, a word $s' \in \Sigma^*$ is called a *prefix* of s, denoted as $s' \sqsubset s$, if there exists another word (configuration) $s'' \in \Sigma^*(\Sigma^\omega)$ such that $s = s's''$. In this case, s'' is called a *suffix* of s. For a set S, $|S|$ denotes its cardinality and 2^S its power set. Symbols $\subset$ and $\subsetneq$ denote the subset and strict subset relations, respectively. Symbol _ denotes an element that is not specified. For instance, consider $Q \times E \times Q$ with Q and E two nonempty sets, $(q, e, \_) \in Q \times E \times Q$ denotes a triple of $Q \times E \times Q$ whose first entry is q, second entry is e, and third entry can be any element of Q.

1.7 Preliminaries on Decidability and Complexity

We recall basic concepts on decidability and complexity (see Hopcroft and Ullman, 1969; Sipser, 1996; Immerman, 1988, etc.). Given two sets $\mathcal{A}$ and $\mathcal{B}$ such that $\mathcal{B} \subset \mathcal{A}$, a *decision problem* refers to whether there exists an *algorithm*[11] for determining whether a given $a \in \mathcal{A}$ belongs to $\mathcal{B}$. A decision problem is called *decidable* if an algorithm for solving this problem exists, and called *undecidable* otherwise. For example, the well-known Turing machine halting problem is undecidable. Decidable problems can be classified into different classes according the complexity

[11]Defined by a *halting Turing machine*.

of the algorithms solving them. For example, P (resp., NP, PSPACE, $\mathsf{NPSPACE}$, $\mathsf{EXPTIME}$, 2-$\mathsf{EXPTIME}$) denotes the class of the problems solvable by polynomial-time (resp., nondeterministic polynomial-time, polynomial-space, nondeterministic polynomial-space, exponential-time, doubly exponential-time) algorithms. NL denotes the class of problems solvable by nondeterministic logarithmic-space algorithms. coNL, coNP, and $\mathsf{coNPSPACE}$ denote the sets of problems whose complements belong to NL, NP, and $\mathsf{NPSPACE}$, respectively. It is known that (Sipser, 1996; Immerman, 1988) $\mathsf{NL} \subset \mathsf{P} \subset \mathsf{NP} \subset \mathsf{PSPACE} \subset \mathsf{EXPTIME}$, $\mathsf{P} \subset \mathsf{coNP} \subset \mathsf{PSPACE}$, $\mathsf{NL} = \mathsf{coNL}$, and $\mathsf{PSPACE} = \mathsf{NPSPACE} = \mathsf{coNPSPACE}$. It is also known that $\mathsf{NL} \subsetneq \mathsf{PSPACE}$ and $\mathsf{P} \subsetneq \mathsf{EXPTIME}$, but whether the rest of these containments are strict are long-standing open questions. It is widely conjectured all the other containments are strict.

A decision problem is called NP (resp., coNP, PSPACE, $\mathsf{EXPTIME}$)-hard if every problem in NP (resp., coNP, PSPACE, $\mathsf{EXPTIME}$) is polynomial time reducible to it. A problem is called X-complete if the problem belongs to the class X and is X-hard, where X can be P, NP, coNP, PSPACE, $\mathsf{EXPTIME}$, etc. Hence there exists no polynomial-time algorithm for solving an NP (resp., PSPACE)-complete problem unless $\mathsf{P} = \mathsf{NP}$ (resp., PSPACE). There is no polynomail-time algorithm for solving an $\mathsf{EXPTIME}$-hard problem since $\mathsf{P} \subsetneq \mathsf{EXPTIME}$. A decision problem is called NL-hard if every problem in NL is logarithmic space reducible to it.

References

Alfaro, L. de and T. Henzinger. (2001). “Interface automata”. *SIGSOFT Softw. Eng. Notes*. 26(5): 109–120.

Angulo, M., A. Aparicio, and C. Moog. (2019). “Structural accessibility and structural observability of nonlinear networked systems”. *IEEE Transactions on Network Science and Engineering*: online.

Atig, M. F. and P. Habermehl. (2009). “On Yen’s path logic for Petri nets”. In: *Reachability Problems*. Ed. by O. Bournez and I. Potapov. Berlin, Heidelberg: Springer Berlin Heidelberg. 51–63.

Balun, J. and T. Masopust. (2021a). “Comparing the notions of opacity for discrete-event systems”. *Discrete Event Dynamic Systems*. 31(4): 553–582.

Balun, J. and T. Masopust. (2021b). “On verification of D-detectability for discrete event systems”. *Automatica*. 133: 109884.

Balun, J. and T. Masopust. (2022). “On verification of weak and strong K-step opacity for discrete-svent systems”. In: *2022 16th International Workshop on Discrete Event Systems (WODES)*.

Basilio, J., C. Hadjicostis, and R. Su. (2021). “Analysis and Control for Resilience of Discrete Event Systems: Fault Diagnosis, Opacity and Cyber Security”. *Foundations and Trends® in Systems and Control*. 8(4): 285–443.

Broy, M., B. Jonsson, J. P. Katoen, L. Martin, and A. Pretschner. (2005). *Model-Based Testing of Reactive Systems: Advanced Lectures (Lecture Notes in Computer Science)*. Secaucus, NJ, USA: Springer-Verlag New York, Inc.

Bryans, J., M. Koutny, L. Mazaré, and P. Ryan. (2008). "Opacity generalised to transition systems". *International Journal of Information Security*. 7(6): 421–435.

Cabasino, M., A. Giua, S. Lafortune, and C. Seatzu. (2012). "A new approach for diagnosability analysis of Petri nets using verifier nets". *IEEE Transactions on Automatic Control*. 57(12): 3104–3117.

Caines, P. E., R. Greiner, and S. Wang. (1988). "Dynamical logic observers for finite automata". In: *Proceedings of the 27th IEEE Conference on Decision and Control*. 226–233 vol.1.

Caines, P. E., R. Greiner, and S. Wang. (1991). "Classical and Logic-Based Dynamic Observers for Finite Automata". *IMA Journal of Mathematical Control and Information*. 8(1): 45–80.

Cassandras, C. and S. Lafortune. (2008). *Introduction to Discrete Event Systems*. 2nd. Springer New York, NY.

Cassez, F. (2009). "The dark side of timed opacity". In: *Advances in Information Security and Assurance*. Ed. by J. H. Park, H.-H. Chen, M. Atiquzzaman, C. Lee, T.-h. Kim, and S.-S. Yeo. Berlin, Heidelberg: Springer Berlin Heidelberg. 21–30.

Cassez, F. (2012). "The complexity of codiagnosability for discrete event and timed systems". *IEEE Transactions on Automatic Control*. 57(7): 1752–1764.

Cassez, F., J. Dubreil, and H. Marchand. (2009). "Dynamic observers for the synthesis of opaque systems". In: *Automated Technology for Verification and Analysis*. Berlin, Heidelberg: Springer Berlin Heidelberg. 352–367.

Cassez, F., J. Dubreil, and H. Marchand. (2012). "Synthesis of opaque systems with static and dynamic masks". *Formal Methods in System Design*. 40(1): 88–115.

Cassez, F. and S. Tripakis. (2008). "Fault diagnosis with static and dynamic observers". *Fundamenta Informaticae*. 88(4): 497–540.

Chaum, D. (1988). “The dining cryptographers problem: Unconditional sender and recipient untraceability”. *Journal of Cryptology*. 1(1): 65–75.

Conte, G., C. Moog, and A. Perdon. (2007). *Algebraic Methods for Nonlinear Control Systems, 2nd Ed.* Springer-Verlag London.

Dong, W., X. Yin, K. Zhang, and S. Li. (2022). “On the verification of detectability for timed systems”. In: *2022 American Control Conference,* Atlanta, USA. accepted.

Dubreil, J., P. Darondeau, and H. Marchand. (2010). “Supervisory control for opacity”. *IEEE Transactions on Automatic Control*. 55(5): 1089–1100.

Falcone, Y. and H. Marchand. (2015). “Enforcement and Validation (at runtime) of Various Notions of Opacity”. *Discrete Event Dyn. Sys.: Theory & Apl.* 25: 531–570.

Garey, M. and D. Johnson. (1990). *Computers and Intractability: A Guide to the Theory of NP-Completeness*. USA: W. H. Freeman & Co.

Garey, M., D. Johnson, and L. Stockmeyer. (1976). “Some simplified NP-complete graph problems”. *Theoretical Computer Science*. 1(3): 237–267.

Genc, S. and S. Lafortune. (2009). “Predictability of event occurrences in partially-observed discrete-event systems”. *Automatica*. 45(2): 301–311.

Gill, A. (1962). *Introduction to Theory of Finite-State Machines*. New York, United States: McGraw–Hill Education – Europe.

Góes, R., B. Rawlings, N. Recker, G. Willett, and S. Lafortune. (2018). “Demonstration of Indoor Location Privacy Enforcement using Obfuscation”. In: vol. 51. No. 7. 145–151.

Grädel, E. (1988). “Subclasses of Presburger arithmetic and the polynomial-time hierarchy”. *Theoretical Computer Science*. 56(3): 289–301.

Haar, S., S. Haddad, T. Melliti, and S. Schwoon. (2017). “Optimal constructions for active diagnosis”. *Journal of Computer and System Sciences*. 83(1): 101–120.

Hadjicostis, C. (2020). *Estimation and Inference in Discrete Event Systems. Communications and Control Engineering*. Springer Nature Switzerland AG.

Han, X., K. Zhang, J. Zhang, Z. Li, and Z. Chen. (2023). "Strong current-state and initial-state opacity of discrete-event systems". *Automatica*. 148: 110756 (1–8).

Hopcroft, J. and J. Ullman. (1969). *Formal Languages and Their Relation to Automata*. USA: Addison-Wesley Longman Publishing Co., Inc.

Immerman, N. (1988). "Nondeterministic space is closed under complementation". *SIAM Journal on Computing*. 17(5): 935–938.

Isidori, A. (1995). *Nonlinear Control Systems. Communications and Control Engineering*. Springer-Verlag London.

Ji, Y., X. Yin, and S. Lafortune. (2019). "Opacity enforcement using nondeterministic publicly-known edit functions". *IEEE Transactions on Automatic Control*. 64(10): 4369–4376.

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

Kalman, R. (1963). "Mathematical description of linear dynamical systems". *Journal of the Society for Industrial and Applied Mathematics Series A Control*. 1(12): 152–192.

Kibangou, A., F. Garin, and S. Gracy. (2016). "Input and state observability of network systems with a single unknown Input". In: vol. 49. No. 22. 37–42.

Kozen, D. (1977). "Lower Bounds for Natural Proof Systems". In: *Proceedings of the 18th Annual Symposium on Foundations of Computer Science. SFCS '77*. Washington, DC, USA: IEEE Computer Society. 254–266.

Kumar, R. and S. Takai. (2010). "Decentralized prognosis of failures in discrete event systems". *IEEE Transactions on Automatic Control*. 55(1): 48–59.

Lee, E. and S. Seshia. (2017). *Introduction to Embedded Systems— A Cyber-Physical Systems Approach*. 2nd. MIT Press.

Li, J., D. Lefebvre, C. Hadjicostis, and Z. Li. (2021). "Observers for a class of timed automata based on elapsed time graphs". *IEEE Transactions on Automatic Control*: online.

Lin, F. (2011). "Opacity of Discrete Event Systems and Its Applications". *Automatica*. 47(3): 496–503.

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

Ma, Z., X. Yin, and Z. Li. (2021). "Verification and enforcement of strong infinite- and K-step opacity using state recognizers". *Automatica*. 133: 109838.

Masopust, T. (2018). "Complexity of deciding detectability in discrete event systems". *Automatica*. 93: 257–261.

Masopust, T. and X. Yin. (2019). "Deciding detectability for labeled Petri nets". *Automatica*. 104: 238–241.

Mazaré, L. (2004). "Using unification for opacity properties". In: *Proceedings of the Workshop on Issues in the Theory of Security (WITS'04)*. 165–176.

Moore, E. (1956). "Gedanken-experiments on sequential machines". *Automata Studies, Annals of Math. Studies*. 34: 129–153.

Moulton, R., B. Hamgini, Z. Khouzani, R. Meira-Góes, F. Wang, and K. Rudie. (2022). "Using subobservers to synthesize opacity-enforcing supervisors". *Discrete Event Dynamic Systems*.

Nykänen, M. and E. Ukkonen. (2002). "The exact path length problem". *Journal of Algorithms*. 42(1): 41–53.

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

Papadimitriou, C. (1981). "On the complexity of integer programming". *J. ACM*. 28(4): 765–768.

Qiu, W. and R. Kumar. (2006). "Decentralized failure diagnosis of discrete event systems". *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*. 36: 384–395.

Rabin, M. and D. Scott. (1959). "Finite automata and their decision problems". *IBM Journal of Research and Development*. 3(2): 114–125.

Ramadge, P. J. (1986). "Observability of discrete event systems". In: *1986 25th IEEE Conference on Decision and Control*. 1108–1112.

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

Rampersad, N. and J. Shallit. (2010). "Detecting patterns in finite regular and context-free languages". *Information Processing Letters*. 110(3): 108–112.

Saboori, A. (2010). "Verification and Enforcement of State-Based Notions of Opacity in Discrete Event Systems". *PhD thesis*. University of Illinois at Urbana-Champaign.

Saboori, A. and C. Hadjicostis. (2007). "Notions of security and opacity in discrete event systems". In: *2007 46th IEEE Conference on Decision and Control*. 5056–5061.

Saboori, A. and C. Hadjicostis. (2009). "Verification of K-step opacity and analysis of its complexity". In: *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*. 205–210.

Saboori, A. and C. Hadjicostis. (2012a). "Opacity-enforcing supervisory strategies via state estimator constructions". *IEEE Transactions on Automatic Control*. 57(5): 1155–1165.

Saboori, A. and C. Hadjicostis. (2012b). "Verification of Infinite-Step Opacity and Complexity Considerations". *IEEE Transactions on Automatic Control*. 57(5): 1265–1269.

Saboori, A. and C. Hadjicostis. (2013). "Verification of initial-state opacity in security applications of discrete event systems". *Information Sciences*. 246: 115–132.

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

Savitch, W. J. (1970). "Relationships between nondeterministic and deterministic tape complexities". *Journal of Computer and System Sciences*. 4(2): 177–192.

Schrijver, A. (1986). *Theory of Linear and Integer Programming*. USA: John Wiley & Sons, Inc.

Shu, S. and F. Lin. (2011a). “Co-detectability of multi-agent discrete event systems”. In: *2011 Chinese Control and Decision Conference (CCDC)*. 1708–1713.

Shu, S. and F. Lin. (2011b). “Generalized detectability for discrete event systems”. *Systems & Control Letters*. 60(5): 310–317.

Shu, S. and F. Lin. (2013). “Enforcing Detectability in Controlled Discrete Event Systems”. *IEEE Transactions on Automatic Control*. 58(8): 2125–2130.

Shu, S., F. Lin, and H. Ying. (2007). “Detectability of discrete event systems”. *IEEE Transactions on Automatic Control*. 52(12): 2356–2359.

Sipser, M. (1996). *Introduction to the Theory of Computation*. 1st. International Thomson Publishing.

Sontag, E. (1979). “On the observability of polynomial systems, I: Finite-time problems”. *SIAM Journal on Control and Optimization*. 17: 139–151.

Tanwani, A., H. Shim, and D. Liberzon. (2013). “Observability for switched linear systems: characterization and observer design”. *IEEE Transactions on Automatic Control*. 58(4): 891–904.

Tong, Y., Z. Li, C. Seatzu, and A. Giua. (2018). “Current-state opacity enforcement in discrete event systems under incomparable observations”. *Discrete Event Dynamic Systems*. 28(2): 161–182.

Tripakis, S. (2002). “Fault diagnosis for timed automata”. In: *Formal Techniques in Real-Time and Fault-Tolerant Systems*. Ed. by W. Damm and E. .-. Olderog. Berlin, Heidelberg: Springer Berlin Heidelberg. 205–221.

Vadhan, S. and T. Wang. (2021). “Concurrent composition of differential privacy”. In: *Theory of Cryptography*. Ed. by K. Nissim and B. Waters. Cham: Springer International Publishing. 582–604.

Vadhan, S. and W. Zhang. (2022). “Concurrent composition theorems for differential privacy”. In: *Poster presentation in TPDP ICML.*

Wonham, W. (1985). *Linear Multivariable Control: A Geometric Approach, 3rd Ed.* Springer-Verlag New York.

Wonham, W. and K. Cai. (2019). *Supervisory Control of Discrete-Event Systems*. Springer International Publishing.

Wu, Y. (2014). “Verification and Enforcement of Opacity Security Properties in Discrete Event Systems”. *PhD thesis*. University of Michigan.

Wu, Y. and S. Lafortune. (2013). “Comparative analysis of related notions of opacity in centralized and coordinated architectures”. *Discrete Event Dynamic Systems*. 23(3): 307–339.

Wu, Y., V. Raman, S. Lafortune, and S. Seshia. (2016). “Obfuscator Synthesis for Privacy and Utility”. In: *NASA Formal Methods*. Ed. by S. Rayadurgam and O. Tkachuk. Cham: Springer International Publishing. 133–149.

Wu, Y., V. Raman, B. Rawlings, S. Lafortune, and S. Seshia. (2018). “Synthesis of Obfuscation Policies to Ensure Privacy and Utility”. *Journal of Automated Reasoning*. 60(1): 107–131.

Wu, Y., K. Sankararaman, and S. Lafortune. (2014). “Ensuring Privacy in Location-Based Services: An Approach Based on Opacity Enforcement”. In: vol. 47. No. 2. 33–38.

Yen, H. C. (1992). “A unified approach for deciding the existence of certain Petri net paths”. *Information and Computation*. 96(1): 119–137.

Yin, X. and S. Lafortune. (2016). “A uniform approach for synthesizing property-enforcing supervisors for partially-observed discrete-event systems”. *IEEE Transactions on Automatic Control*. 61(8): 2140–2154.

Yin, X. and S. Lafortune. (2017). “A new approach for the verification of infinite-step and K-step opacity using two-way observers”. *Automatica*. 80: 162–171.

Yin, X. and S. Li. (2019). “Supervisory control for delayed detectability of discrete event systems”. In: *2019 IEEE 15th International Conference on Automation Science and Engineering (CASE)*. 480–485.

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

Zhang, K. (2017). “The problem of determining the weak (periodic) detectability of discrete event systems is PSPACE-complete”. *Automatica*. 81: 217–220.

Zhang, K. (2021a). "A unified method to decentralized state detection and fault diagnosis/prediction of discrete-event systems". *Fundamenta Informaticae*. 181: 339–371.

Zhang, K. (2021b). "State-based opacity of real-time automata". In: *27th IFIP WG 1.5 International Workshop on Cellular Automata and Discrete Complex Systems (AUTOMATA 2021)*. Ed. by A. Castillo-Ramirez, P. Guillon, and K. Perrot. Vol. 90. *Open Access Series in Informatics (OASIcs)*. Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik. 12:1–12:15.

Zhang, K. (2022). "Detectability of labeled weighted automata over monoids". *Discrete Event Dynamic Systems*. 32(3): 435–494.

Zhang, K. and A. Giua. (2018). "Weak (approximate) detectability of labeled Petri net systems with inhibitor arcs". In: vol. 51. No. 7. 167–171.

Zhang, K. and A. Giua. (2019). "K-delayed strong detectability of discrete-event systems". In: *Proceedings of the 58th IEEE Conference on Decision and Control (CDC)*. 7647–7652.

Zhang, K. and A. Giua. (2020a). "Instant detectability of discrete-event systems". In: vol. 53. No. 2. 2137–2142.

Zhang, K. and A. Giua. (2020b). "On detectability of labeled Petri nets and finite automata". *Discrete Event Dynamic Systems*. 30(3): 465–497.

Zhang, K., L. Zhang, and L. Xie. (2020). *Discrete-Time and Discrete-Space Dynamical Systems. Communications and Control Engineering*. Springer International Publishing.

Zhang, T. and K. Zhang. (2022). "Eventual strong detectability of labeled weighted automata over monoids". In: vol. 55. No. 28. 270–275.