State estimation and verification of detectability and opacity in weighted automata

Aiwen Lai

UNIVERSITE BRETAGNE LOIRE

université angers

Thèse de Doctorat

Aiwen LAI

Mémoire présenté en vue de l'obtention du
grade de Docteur de l'Université d'Angers
Label europen
sous le sceau de l'Université Bretagne Loire

École doctorale : Mathématiques et Sciences et Technologies de l'Information et de la Conmmunication (MathSTIC)

Discipline : Automatique, section CNU 61
Unité de recherche : Laboratoire Angevin de Recherche en Ingénierie des Systèmes (LARIS)

Soutenue le October 2019 /09/30
Thèse n° : 1

State Estimation and Verification of Detectability and Opacity in Weighted Automata

JURY

Rapporteurs :	**M. Jan KOMENDA**, Professor, The Czech Academy of Sciences
	M. Stéphane GAUBERT, Directeur de recherche, INRIA
Examinateurs :	**M. Bertrand COTTENCEAU**, Professeur, Université d'Angers
	M^{me} Isabel DEMONGODIN, Professeur, Université d'Aix-Marseille
	M. Zhiwu LI, Professeur, Xidian University
Directeur de thèse :	**M. Sébastien LAHAYE**, Professeur, Université d'Angers
Co-directeur de thèse :	**M. Alessandro GIUA**, Professeur, University of Cagliari

Abstract

State estimation is an important and fundamental problem in the systems and control theory. In many real-world systems, it is not always possible to obtain directly the state information due to measurement noises, uncertainties or limited sensor availability. Therefore, estimating the state of a system is critical when one wants to make decisions in certain applications that rely on state information, such as supervisory control. State estimation aims at accurately characterizing the possible states by observing system's behaviour, i.e., the output information obtained during the evolution of the system.

Another important issue that is closely related to state estimation is the verification problem. Given a system, it is important to check if it has some desired properties. That is, verify in a formal way whether the studied system satisfies some expected requirements or specifications. Generally, we would like to propose algorithmic and provably correct procedures to achieve the above checking, and such a formal satisfaction checking problem is called the *verification problem*. Property verification has been extensively studied since it is important in many complicated and safety-critical real-world systems.

Automata and Petri nets, two important classes of Discrete Event Systems (DESs) models, have been intensively used to deal with state estimation and property verification over the past few decades. A DES is a discrete-state, event-driven dynamic system in which the state evolution depends entirely on the abrupt occurrence of asynchronous discrete events. Many real-world systems, e.g., queueing systems, computer systems, communication systems, traffic systems, are discrete event systems. DES models have been widely used to deal with state estimation and verification problems in the literature.

This thesis focuses on the state estimation and verification problems, including detectability and opacity verification, of DESs modeled as weighted automata (WAs). WAs are a quantitative extension of classical finite automata in which the transitions carry weights

belonging to a semiring. The weight associated to a transition can model, e.g., the cost, the resource (energy), the time needed or the probability for executing the transition. The main work of this thesis is as follows. (1) An online procedure is proposed to deal with the problem of state estimation for WAs. (2) The state estimation approach is extended to tackle the fault diagnosis problem. (3) Given an unambiguous weighted automaton (UWA), a formal procedure with exponential complexity (resp. polynomial complexity) based on the construction of observer (resp. detector) is introduced to verify its current-state detectability. (4) Given a UWA, an approach is proposed to verify its initial-state detectability and initial-state opacity.

Keywords: Discrete event system, weighted automata, state estimation, fault diagnosis, detectability, opacity.

Acknowledgement

First of all, I would like to express my deepest gratitude to my two supervisors, Prof. Sébastien Lahaye and Prof. Alessandro Giua, for their unwavering support, guidance. In particular, Prof. Lahaye gave me a lot of freedom to explore anything I am interested in. Like my friend or brother, he always patiently discusses issues with me, even things that seem naive. Prof. Giua made many valuable suggestions on this work. I clearly remember that when we met in Toulouse during the 2017 IFAC World Congress, he suggested me to study the state estimation of the max-plus automaton before considering the identification problem, which turns out to be good research topic. Without their instruction and patience, I would never have been able to complete the thesis.

In addition, I would like to express my sincere appreciation to Prof. Zhiwu Li, for his constant support, encouragement and trust. He always believes in me, and whenever I encounter some major choices that need to be decided, he can always give me valuable advice. I think he is more like a member of my family and gave me a lot of care.

I would also like to thank Prof. Jan Komenda and Prof. Bertrand Cottenceau for being the members of Individual Follow-up Committee of my thesis (CSI: Comité de Suivi Individuel) to monitor the progress of my research. Besides, Prof. Komenda gave me the idea to investigate the detectability of max-plus automata, which led to another contribution of the thesis. I sincerely thank Prof. Stéphane Gaubert and Prof. Isabel Demongodin for accepting to sit on my dissertation committee.

Finally, I sincerely thank my parents, Maosong Lai and Zhaozhu Lai, and my sisters Qiuju Lai, Qiuxiang Lai and Qiuliu Lai, for their constant support and love. I would not have been the person I am today without their support. I owe my fiancee, Ms. Jingfeng Lai, too much in past two and half years. I would like to express my deepest appreciation to jingfeng for her love, understanding and support. I love you, if there is a deadline, it will be 10,000 years.

Contents

List of Figures

List of Tables

1

Introduction

State estimation is a fundamental problem in discrete event systems (DES), and it has been extensively studied in the framework of automata and Petri nets. State estimation aims at accurately characterizing the possible states by observing system's behaviour, i.e., the output information obtained during the evolution of the system. Generally, state estimation can be partitioned into current-state estimation and initial-state estimation. In current-state estimation problem, we want to estimate all possible states the system can be in currently based on the observation. In initial-state estimation, we want to determine the set of initial states the system may start from.

Recently, the problem of state estimation has been investigated in terms of detectability. Current-state detectability requires that the current state of the system can always be detected uniquely (or unambiguously) within a finite delay, i.e., a finite number of event observations. Initial-state detectability (I-detectability) requires that the initial state can always be determined uniquely (or unambiguously) after the occurrence of finite sequence of events.

The state estimation plays a very important role in some applications of DESs. One related application is opacity. Opacity is a general information flow property characterizing whether or not the "secret" of a system can be inferred by an external observer, called intruder. Another important application is the fault diagnosis. For those DESs where the failures are modeled by certain states, the diagnosis problem can be considered as a state estimation problem.

In this chapter, we first present a literature review on state estimation, detectability, fault diagnosis and opacity, and then we give the motivation for our study. Finally, we describe the organization and contribution of this thesis.

Contents

1.1 Literature review

1.1.1 State Estimation and Detectability Verification Problems

The state estimation of a dynamic system is a fundamental issue in control theory. This problem has been investigated both in time driven systems and discrete event systems (DESs). In time driven systems, the state estimation aims at generating an estimate $\hat{x}(t)$ of the current state $x(t)$ at a time instant t while in DES, it consists in capturing the characterisation of possible states with respect to a finite set of observed outputs [Giua, 2011].

In general, the output information of the considered system must be given in order to address state estimation problem. Typically, there exists two different kinds of outputs since DES can be described by states and events whereas sensors can be associated to events or states. The corresponding outputs are event observation and state observation. In the case of event observation, events are partitioned into observable and unobservable subsets. For state observation, a subset of states to which the current states of system belong is known. Note that these two outputs can be combined, which means that there may exist event and state observations during the system evolution. Automata and Petri nets (PNs) are two well-known abstractions for DES. These two mathematical formalisms can be used to deal with the state estimation problem, which has been studied over the past few decades.

In the framework of automata, the embryonic form of state estimation problem first appeared in [Wonham, 1976]. Since then, this problem has been studied by many researchers using automata. The early work of investigating state estimation was done by Caines et al. [Caines and Wang, 1989], Ramadge [Ramadge, 1986] and Ozveren and Willsky [Ozveren and Willsky, 1990]. Caines et al. showed how it is possible to use the information contained in the past sequence of observation states and control inputs to calculate the set of states consistent with observation. Ramadge studied the issue of determining the current state of the system modeled by nondeterministic finite automaton and showed the possibility of designing an observer for a discrete event process. Ozveren and Willsky formulated the observability with an assumption that the current state of system is known. An approach is proposed to build an observer according to a finite string from outputs. Besides, they showed that an observer may have an exponential number of states although it can be built in polynomial time.

In the framework of PNs. Algorithms for state (or marking) estimaton are proposed for labeled PNs with silent or/and indistinguishable transitions in [Corona *et al.*, 2007; Giua

et al., 2005; Cabasino *et al.*, 2011]. A transition is said to be silent if its firing cannot be detected by external agent. Two transitions are called undistinguishable if they are labeled by the same symbol. In [Corona *et al.*, 2007; Giua *et al.*, 2005], the notion of *basis marking* is proposed to compute the consistent markings. The authors proved that consistent markings can be characterized by a linear system with a fixed structure that does not change as the length of the observed word increases. In [Cabasino *et al.*, 2011] the marking estimation problem for PNs in the presence of silent and indistinguishable transitions is addressed first in order to solve the diagnosis problem. We have to mention that there exists a survey [Giua and Seatzu, 2014] pertaining the main contributions of state estimation in the field of PNs. Li and Hadjicostis [Li and Hadjicostis, 2013] proposed a recursive algorithm whose complexity is polynomial with respect to the length of observation for calculating the minimum initial marking of a labeled Petri net.

There are fewer works dealing with state estimation in timed DES compared to logical DES framework. In [Bonhomme, 2013; Bonhomme, 2015], a method for current state estimation in P-time PNs is presented. A state observer is synthesized according to the corresponding untimed PN, and the schedulability of a firing sequence is defined. The candidate firing sequences of an observed word are obtained from the state observer. One schedulable candidate sequence corresponds to a consistent marking. The checking of schedulability of a sequence is completed by solving a linear programming problem. In [Declerck and Bonhomme, 2014], timed labeled PNs with observable transitions are described by algebraic models. An approach for reconstructing the sequence of unobservable transitions, i.e., completing an observed word into a fireable sequence which is consistent with this observation is developed therein. Timed choice-free PNs are studied in [Wang *et al.*, 2011]. The authors computed the set of basis markings, notion extended from [Corona *et al.*, 2007], and used the time equations to reduce this set by removing all markings that are inconsistent with the time information. Basile et al. [Basile *et al.*, 2013; Basile *et al.*, 2015] investigated the state estimation problem for time unlabeled and time labeled PNs respectively. An on-line approach based on the modified state class graph is presented therein. Note that the assumption of acyclicity of unobservable transitions is no more needed. The new assumption is that there is no cycle of unobservable transitions that can fire in a null time interval.

Besides the automata and PNs, the max-plus algebra plays an important role in some applications of timed DES [Komenda *et al.*, 2017; Komenda *et al.*, 2018]. Hardouin et al. [Hardouin *et al.*, 2010] investigated the state estimation of timed event graphs, a subclass

of timed PNs capturing only synchronization and delay phenomena. Timed event graphs can be represented by state-space equations in max-plus algebra, strongly reminiscent of discrete-time representations for conventional linear systems. The design of an observer matrix for timed event graphs is proposed in analogy to Luenberger observer for classical linear systems.

In some literature, the state estimation problem has been investigated in terms of detectability verification. Two common detectabilities are current-state detectability and I-detectability. Current-state detectability requires that the current state of the system can always be detected uniquely after observing a finite sequence of event observations. I-detectability requires that the initial state can always be determined uniquely within a finite delay.

In [Shu *et al.*, 2007; Shu and Lin, 2011; Shu and Lin, 2013], the detectability of non-probabilistic DESs based on the construction of the observer automaton is investigated. [Shu *et al.*, 2008; Keroglou and Hadjicostis, 2015; Keroglou and Hadjicostis, 2017; Yin, 2017] studied the detectability of probabilistic (stochastic) DESs. Note that the detectability in non-probabilistic systems characterizes whether the state of the system can be uniquely determined after a finite number of observations. While in the probabilistic framework, it characterises whether the probability of detecting the state of the system converges to one as the length of the observation increases.

Shu et al. [Shu and Lin, 2011] defined four types of detectabilities, i.e., strong detectability, detectability, strong periodic detectability, periodic detectability for nondeterministic finite automata (NFAs) so as to characterise different properties of current state estimation. More precisely, an NFA is strongly detectable if its current state and subsequent states can be uniquely determined, after observing a finite sequence of events, for all trajectories of the system. Note that a trajectory of a system is represented by an infinite sequences that can be generated by the system. An NFA is weakly detectable if its current state and subsequent states can be uniquely determined, after observing a finite sequence of events, for some trajectories of the system. An NFA is said to be strongly periodically detectable if the unique current state for all trajectories of the system can be periodically determined. Similarly, if the unique current state for some trajectories of the system can be periodically determined, then the system is said to be weakly periodically detectable. An observer is constructed as to derive necessary and sufficient conditions for checking these four detectabilities. In [Shu *et al.*, 2008] the DES is represented as a non-deterministic probabilistic automaton where probabilities are associated with the non-deterministic

transitions. A given probabilistic automaton is first converted into a non-probabilistic automaton, then necessary and sufficient conditions are proposed to check the strong and weak detectabilities. Keroglou and Hadjicostis [Keroglou and Hadjicostis, 2017; Keroglou and Hadjicostis, 2015] investigated the state estimation of a probabilistic automaton by defining the concepts of A-detectability and AA-detectability respectively. As the observed output symbols increase, the state estimation will become more accurate.

Shu and Lin [Shu and Lin, 2013] formulated the initial-state estimation problem of NFAs as I-detectability. It characterizes whether the initial state of an NFA can be uniquely determined after a finite number of events have been observed. An exponential algorithm is introduced for checking strong I-detectability based on the construction of an I-observer. In additional, a polynomial-time algorithm is introduced for checking weak I-detectability based on the construction of an I-detector. Yin [Yin, 2017] introduced the notion of SI-detectability as an extension of I-detectability in the stochastic setting by considering the probabilistic sensor failures. SI-detectability characterizes the convergence of the probability for determining the initial-state of a probabilistic finite state automaton. An algorithm for verifying the SI-detectability is proposed, and the complexity of this verification is proved to be PSPACE-complete.

Giua and Seatzu [Giua and Seatzu, 2002] investigated the detectability of PNs by assuming that the PN structure is known and the transition firings can be precisely observed. Masopust and Yin [Masopust and Yin, 2019b] explored the existence of algorithms to verify strong and weak detectability for labeled PNs. The authors showed that although there exists an algorithm for checking strong detectability, this algorithm is infeasible because it requires at least exponential space. Besides, it is proved that there is no algorithm for checking weak detectability.

We refer the reader to [Yin, 2019; Ghazel *et al.*, 2009; Ramírez-Treviño *et al.*, 2003; Ma *et al.*, 2017; Masopust, 2018; Masopust and Yin, 2019a; Sasi and Lin, 2018; Yin and Lafortune, 2017c; Zhang, 2017; Zhao *et al.*, 2019] for more references on state estimation and detectability verification.

1.1.2 Fault Diagnosis Problem

The fault diagnosis of discrete event systems (DES) has received a lot of attention in recent years. A fault is defined to be any deviation of a system or of one of its components from its normal behavior. According to the manner in which faults are reset after they occur, faults are classified into permanent and intermittent ones. A fault is permanent if

the recovery event occurs only due to a repair/replacement of the fault. On the contrary, a fault is intermittent if the recovery event can occur either spontaneously or through repair/replacement. In this thesis, we consider the case where all faults are permanent. In other words, the studied system cannot spontaneously move from the fault state to a normal state.

Generally, the goal of fault diagnosis is to achieve three complementary tasks [Zaytoon and Lafortune, 2013]: fault detection, fault isolation and fault identification. Fault detection aims at determining whether a system is in normal status or whether a fault has occurred. If a fault has occurred, fault isolation and identification aim at localizing the system component(s) causing the fault and identifying the nature of the fault respectively. In the thesis, we use the terminology of *fault diagnosis* to describe the objective of the detection of a fault class. Automata and Petri nets (PNs) have been intensively used to deal with fault diagnosis in the literature.

An overview of fault diagnosis approaches for DES is proposed in [Zaytoon and Lafortune, 2013]. The classification of diagnosis approaches with respect to different kinds of criteria is provided. These criteria include fault compilation (on-line, offline), modelling formalism (automata, PNs), fault representation (models including faulty behavior, fault-free modes), and decision structure (centralized, decentralized, distributed).

In [Sampath *et al.*, 1995], fault diagnosis and diagnosability of DES are addressed in an event-based framework, that is, the failures are treated as unobservable events. Given an observation, the fault diagnosis consists in determining if a fault has occurred, i.e., if an evolution containing a transition labeled by a fault event has produced the given observation. A deterministic finite automaton is diagnosable if the occurrence of any fault event can be detected after a finite number of steps. Necessary and sufficient conditions for diagnosability of a DES are proposed therein. Besides, the diagnosers are introduced to solve the diagnosis problem. This approach has been extended to diagnosis of timed DES where different time bounds can be associated to different transitions of the same event label [Chen and Provan, 1997]. In [Tripakis, 2002], the above framework is extended to timed automata. The notion of $\triangle-$diagnosability for timed automata is proposed. A timed automaton is $\triangle-$diagnosable if a fault can be detected with a delay of at most $\triangle$ time units after it occurred. Besides, an algorithm is presented to check such diagnosability for a given timed automaton, and it was shown that this checking has PSPACE-complete complexity. In [Bouyer *et al.*, 2005], the fault diagnosis is studied for deterministic timed automata (DTA) and event-recording timed automata (ERA). The complexities for checking the existence of

diagnoser for DTA and ERA are 2-EXPTIME-complete and PSPACE-complete respectively. Other important contributions for fault diagnosis in the framework of probabilistic automata have been presented in [Athanasopoulou and Hadjicostis, 2005; Fabre and Jezequel, 2010; Lunze and Schröder, 2001; Thorsley *et al.*, 2008], among others.

There exists work dealing with fault diagnosis in the state-based framework [Lin, 1994; Zad *et al.*, 2003; Zad *et al.*, 2005]. In a state-based approach, it is assumed that the states of the system can be partitioned according to the condition (normal or faulty status) of the system. The fault diagnosis problem then becomes to use system output information to determine the block of the normal/faults partition to which the state belongs. Zad et al. [Zad *et al.*, 2003] introduced state-based diagnoser to determine the occurrence of a failure mode based on the sequence of state outputs. The authors extended the above approach for solving fault diagnosis problem in timed DES [Zad *et al.*, 2005]. Diagnosability of failures is discussed, and necessary and sufficient condition for time-diagnosability are derived.

In [Wu and Hadjicostis, 2005], redundancy (additional places and the connections associated with them) is added into the original PN to enable fault identification using algebraic decoding techniques. Two types of faults have been considered: place faults that cause the corruption of the number of tokens in a certain place whereas transition faults that prevent tokens from being removed from (deposited at) the input (output) places of a particular transition. In other words, a place fault that corrupts the net marking, and a transition fault that causes an incorrect update of the marking after events occurrences. In this approach, the authors assumed that transition firings are not directly observable but that the system marking is periodically observable. Basile et al. [Basile *et al.*, 2009] studied fault diagnosis through on-line computing of the set of possible fault events required to explain the last observed event. The diagnoser was built by defining and solving integer linear programming (ILP) problems. The computational complexity was reduced by reformulating these ILP problems on specific parts of the considered PN that influence the occurrence of the observed event. In [Basile *et al.*, 2012], the ILP technique was adopted to deal with the $k-$diagnosability, a fault can be detected within a finite delay of $k-$steps, for bounded PNs. Necessary and sufficient condition for $k-$diagnosability is proposed. Moreover, there is no specific assumption on the structure of the subnet induced by unobservable transitions. In [Basile *et al.*, 2015], the modified state class graph is used to perform on-line fault diagnosis for labeled time PNs.

Cabasino et al. [Cabasino *et al.*, 2010; Cabasino *et al.*, 2011] presented fault diagnosis approaches for unlabeled and labeled PNs respectively using the notion of *basis marking*.

Given an observation ω, this approach characterizes the minimal sequences of unobservable transitions whose firing enable ω using linear algebraic constraints. A diagnosis state was associated to each observation and to each fault class by an on-line diagnosis algorithm. In the case of bounded PNs, the basis reachability graph can be computed offline and used to perform fast on-line fault diagnosis. In their recent work [Cabasino *et al.*, 2014], the diagnosability of labeled PNs is studied. Necessary and sufficient condition for diagnosability are proposed, and the basis reachability diagnoser and modified basis reachability graphs are introduced to test the diagnosability. Yin and Lafortune [Yin and Lafortune, 2017b] proved that checking diagnosability for labeled PNs is decidable and is EXPSPACE-complete.

Other important contributions for fault diagnosis in the framework of PNs have been presented in [Ramírez-Treviño *et al.*, 2007; Hernandez-Flores *et al.*, 2011; Wen *et al.*, 2005; Lefebvre and Delherm, 2007; Dotoli *et al.*, 2009; Benveniste *et al.*, 2003; Jiroveanu and Boel, 2010; Cong *et al.*, 2017], among others.

1.1.3 Opacity Verification Problem

Security and privacy are crucial properties in online services and network communications. Opacity is an important aspect of such properties. It characterizes whether the "secret" of a system can be inferred by an intruder via observing its behavior. According to the definition of secrets, opacity can be classified into state-based opacity and language-based opacity. Moreover, the state-based opacity can be further partitioned as initial-state opacity (I-opacity) ([Saboori and Hadjicostis, 2013b; Keroglou and Hadjicostis, 2013; Wu and Lafortune, 2013; Wang *et al.*, 2018; Tong *et al.*, 2017]), current-state opacity ([Saboori and Hadjicostis, 2007; Saboori and Hadjicostis, 2013a; Tong *et al.*, 2017; Cong *et al.*, 2018]), initial-and-final-state opacity ([Wu and Lafortune, 2013]) and K-step opacity ([Saboori and Hadjicostis, 2011]).

It is assumed the intruder has full knowledge of the system's structure but only partial observability. Based on its observations, the intruder constructs an estimate of the system's behavior. Given a secret as a set of states. A system is said to be current-state opaque if, for any observation, the intruder can never infer that the current state of the system is within the set of secret states. Given a secret as a language, i.e., a set of event sequences. A system is language opaque if the intruder cannot conclude that the evolution of the system belongs to the secret.

In this thesis, we focus on I-opacity. Given a secret as a set of states (Normally, the secret

is a subset of the set of initial states). A system is said to be initial-state opaque if, for any observation, the intruder cannot determine if the evolution of the system has started from a secret state. More precisely, I-opacity requires that the set of initial states estimated by the intruder for any observation should contain at least one element that does not belong to the secret . The I-opacity is first introduced for Petri nets by Bryans et al. [Bryans *et al.*, 2005], and extended to labeled transition systems in [Bryans *et al.*, 2008]. It is proved in [Bryans *et al.*, 2005] that the problem of verifying I-opacity in bounded Petri nets is decidable. In [Bryans *et al.*, 2008], general opacity (including the I-opacity) problems in transition systems are proved to be undecidable. Recently, the I-opacity for bounded labeled Petri nets is solved by Tong et al. in [Tong *et al.*, 2017] where the secret is defined as an arbitrary subset of the reachability set. A necessary and sufficient condition for checking I-opacity based on the construction of the *modified basis reachability graph* is proposed.

In the framework of automata, the I-opacity problem has been investigated in [Saboori and Hadjicostis, 2013b; Keroglou and Hadjicostis, 2013; Wu and Lafortune, 2013; Wang *et al.*, 2018]. Particularly, in [Saboori and Hadjicostis, 2013b], an initial-state estimator is constructed and it can be used to verify I-opacity of an NFA for both invariant-secret and varying-secret. That is, the structure of such an initial-state estimator does not need to be modified when the secret changes. The initial-state estimator has up to 2^{n^2} states where n is the number of states in the NFA. In addition, when the secret is fixed, the complexity of verifying I-opacity is reduced to $\mathcal{O}(4^n)$ by introducing the notion of verifier. Recently, Wu and Lafortune [Wu and Lafortune, 2013] prove that the initial state of an NFA can be estimated by the observer of its reverse automaton, and as a result, the verification complexity of I-opacity is reduced to $\mathcal{O}(2^n)$. In [Keroglou and Hadjicostis, 2013], I-opacity is studied in the framework of probabilistic finite automata.

Cassez [Cassez, 2009a] investigates the decidability of opacity verification in dense timed systems. It is shown that the opacity problem in timed automata is undecidable since it is already undecidable for event recording automata, which is a very restrictive subclass of timed automata. Recently, in [Wang *et al.*, 2018], the problem of I-opacity in real-time automata (timed automata with only one clock to be reset at each transition) is proved to be decidable by reducing it into the inclusion problem of regular languages.

Note that the state-based opacity verification problem is tightly related to state estimation problem. The verification of current-state opacity and I-opacity can be done by the construction of an observer (current state estimator) and an initial-state estimator respectively. Opacity is also closely related to anonymity [Sweeney, 2002], non-interference

[Hadj-Alouane *et al.*, 2005b; Hadj-Alouane *et al.*, 2005a], and secrecy [Takai and Kumar, 2009], which are other information flow security properties. Note that anonymity and secrecy can be formulated as opacity problems.

Jacob et al. [Jacob *et al.*, 2016] presents an overview of different notions of opacity in DESs. The decidability of opacity verification problems in different DES models and the complexity of decidable opacity checks are presented therein. The reader is referred to the [Yin, 2019; Saboori and Hadjicostis, 2011; Yin and Lafortune, 2017a; Saboori and Hadjicostis, 2008; Badouel *et al.*, 2007; Yin *et al.*, 2019; Bérard *et al.*, 2015] for more references on opacity verification in the area of DESs.

1.2 Motivation of the Thesis

From the above literature review, we know that state estimation, fault diagnosis, and detectability and opacity verification problems are very important in DESs. Most of the existing work on these topics is focused on logical DESs, that is, classical automata and PNs where there is no quantitative information associated with the firing of transitions in a PN or the occurrence of events in an automaton. However, in some real-world systems, the quantitative information is very important and necessary. In fact, the occurrence of an event or the state transition of the system takes some time or requires some energy or resources. For instance, in a manufacturing system, a product needs to undergo a series of processing before it goes on the market, and each processing consumes a certain amount of power and time.

Weighted finite automata (WAs) are classical NFAs in which the transitions carry weights belongs to a semiring. These weights may model, e.g., the cost involved when executing a transition, the amount of resources or time needed for this, or the probability or reliability of its successful execution [Droste *et al.*, 2009]. From here, we know that a WA is a DES model that takes into account the quantitative information of real-world systems.

For such a DES model, as in classical automata and PNs, we think it makes sense to investigate its state estimation as well as the property verification problems. In this thesis, on one hand, we proposed formal online algorithms to tackle the state estimation and fault diagnosis problems of WAs. On the other hand, the current-state detectability, I-detectability and I-opacity verification problems are studied for unambiguous weighted automata (UWAs).

1.3 Organization and Contribution of the Thesis

This thesis investigates the state estimation and detectability and opacity verification in the framework of WAs. The organization and main contributions of the thesis are summarized as follows:

- Chapter 2. Some basics on classical automata and weighted automata are presented in this chapter.
- Chapter 3. This chapter deal with state estimation of WAs. One on hand, we first give the definition of the set of states consistent with an observation and a given weight value. Then we propose algorithms to solve online the state estimation problem. The main idea behind the proposed algorithms originates from the fact that the dynamic behavior of a WA can be characterized by its state vector, solution of recurrent equations on words representing the sequence of occurring labels (events). On the other hand, the state estimation approach is extended to deal with fault diagnosis of a WA. First, for a given WA where faults are associated with unobservable events, we propose an algorithm that transforms it into another automaton, called augmented automaton. Then, we prove that solving fault diagnosis of the original WA is equivalent to estimating the states of the augmented automaton. In addition, we define the diagnosis states, "normal" or "faulty" or "uncertain", with respect to an observation, a given value and a fault class. Finally, an on-line algorithm is proposed to compute these diagnosis states.
- Chapter 4. The of current-state detectability verification problem is studied for UWAs in this chapter. We first define the concept of consistent states for an infinite sequence of labels, which represents a trajectory of the system. Then, inspired by the work in [Shu *et al.*, 2007; Shu and Lin, 2011], we define four types of detectabilities, i.e., strong detectability, weak detectability, strong periodic detectability, and weak periodic detectability, for WAs. This chapter can be divided into two parts. In the first part of this chapter, an exponential approach is presented to check strong and weak detectabilities. The main contributions of this part are as follows. (1) A novel algorithm is introduced to construct a finite state automaton (called observer) for a given UWA, and we prove that this observer can be used for the current-state estimation of the studied system. Note that this algorithm is first presented in this chapter, and will be adapted to deal with the verification problem of I-detectability and I-opacity for UWAs in Chapter 5. (2) Necessary and sufficient conditions based

on the constructed observer are derived for verifying the strong and weak four detectability properties of UWAs. In the second part of this chapter, a polynomial approach is introduced to check strong detectability and strong periodic detectability, or simply called strong detectabilities. The main contributions of this part are as follows. (1) For a given UWA, we construct its detector, which is a special finite state automaton over a weighted alphabet. The number of states of detector is polynomial with respect to the state space cardinality of original system. (2) Necessary and sufficient conditions based on the constructed detector are proposed for checking strong detectabilities of the studied UWA. (3) We present an example to illustrate that the conditions stated for observer for weak detectability and weak periodic detectability, simply called weak detectabilities, cannot be transposed to detector so as to check the weak detectabilities.

- Chapter 5. This chapter deals with the verification of I-detectability as well as the I-opacity verification for UWAs. (1) We extends the notion of I-detectability and I-opacity from logical DESs to the framework of WAs. Two types of I-detectability, namely strong Idetectability and weak I-detectability are defined for WAs. A WA is said to be strong I-detectable if we can uniquely determine its initial state for all trajectories after a finite number of observations. A WA is said to be weak I-detectable if we can uniquely determine its initial state for some trajectories after a finite number of observations. (2) For a given UWA, a formal procedure is proposed to construct its initial-state estimator (called I-observer). More precisely, given a UWA, an algorithm is first proposed to construct its augmented version, called augmented automaton, which is also a WA. Each node (state) of the augmented automaton is a pair of states of the original WA indicating that the first state of the pair is reachable from the second state in the system. Then, the algorithm that is first presented in Chapter 4 is used to construct the current-state estimator of augmented automaton, and we prove that this estimator can serve as the initial-state estimator of original system. (3) Necessary and sufficient conditions are developed for verifying I-detectability and I-opacity of the studied UWA based on the constructed I-observer.

- Chapter 6. Conclusion and future directions related to this thesis are described in this Chapter.

2

Preliminaries

In this chapter, we recall some basics of automata and weighted automata. For more details, we fefer the reader to [Cassandras and Lafortune, 2009] and [Droste *et al.*, 2009].

Contents

2.1 Automata

In this section we recall some basics of automata. For more details, we fefer the reader to [Cassandras and Lafortune, 2009].

2.1.1 Alphabet and Language

Definition 2.1. An alphabet E is a finite and empty set of labels. The number of labels that E contains is called its cardinality and is denoted by $|E|$.

Definition 2.2. A string ω defined on E is a finite sequence of labels in E. The number of labels of that ω contains is called its length and is denoted by $|\omega|$. The set of all finite strings defined over alphabet E is denoted by E^*, including the *empty string* having length zero.

In this thesis, the empty string is denoted by ϵ instead of ε because ε will be used as a neutral element of a semiring.

Definition 2.3. The concatenation of two strings $\omega_1 \in E^*$ and $\omega_2 \in E^*$ is a new string, denoted by $\omega = \omega_1 \cdot \omega_2$ or simply by $\omega = \omega_1\omega_2$, composed by the sequence of labels in ω_1 followed by the sequence of labels in ω_2.

Definition 2.4. If a string $\omega \in E^*$ can be represented as $\omega = \nu\gamma\eta$ with $\nu\gamma, \eta \in E^*$, then string ν is called a prefix of ω, string γ is called a substring of ω, and string η is called a suffix of ω.

Example 2.1. Consider an alphabet $E = \{a, b, c\}$. Its cardinality is $|E| = 3$. Strings $\omega_1 = ab$, $\omega_2 = abc$ and $\omega_3 = abac$ defined on E have length $|\omega_1| = 2$, $|\omega_2| = 3$ and $|\omega_3| = 4$ respectively. The concatenation of ω_1 and ω_2 is $\omega_1 \cdot \omega_2 = ababc$. Consider string $\omega_3 = abac$. Its prefixes are ϵ, a, ab, aba and $abac$. Its suffixes are ϵ, c, ac, bac and $abac$. Its substrings are: all its prefixes, all its suffixes and b, a and ba.

Languages are defined as the sets of strings over an alphabet. In other words, L is said to be a language over alphabet E if it is a subset of E^*, i.e., $L \subseteq E^*$.

Definition 2.5. A language L defined on an alphabet E is a set of strings on this alphabet. Its cardinality, i.e., the number of strings that it contains, is denoted by $|L|$.

Example 2.2. Consider an alphabet $E = \{a, b, c\}$ and the following sets of strings.

$$L_1 = \{aa, ab, ac, ca\}\,,\ L_2 = \{a, b, c\}\,,\ L_3 = \{\epsilon\}\,,\ L_4 = \emptyset,\ L_5 = \{ab, d\}\,.$$

We find that sets L_1, L_2, L_3 and L_4 are languages defined on alphabet E. Set L_5 is not a language defined on E since it contains an element d that does not belong to E. Language L_1 consists of four strings, i.e., $|L_1| = 4$. Language L_2 consists of three strings, i.e., $|L_1| = 3$, which is consistent with the alphabet. Language L_3 only consists of the empty string. Language L_3 is the empty set that does not contain any strings.

Sequence projection

Definition 2.6. Let $E' \subseteq E$ be a subset of E. The natural projection of E on E' is denoted by $P_{E'} : E^* \to E'^*$ and is defined as:

$$\begin{cases} P_{E'}(\epsilon) = \epsilon \\ P_{E'}(a) = a, & if\ a \in E'; \\ P_{E'}(a) = \epsilon, & if\ a \in (E \setminus E'); \\ P_{E'}(sa) = P_{E'}(s)P_{E'}(a), & s \in E^*,\ a \in E. \end{cases} \tag{2.1}$$

The inverse of projection $P_{E'}$ is denoted by $P_{E'}^{-1} : E'^* \to E^*$ and is defined as:

$$P_{E'}^{-1}(\omega) = \{\sigma \in E^* \mid P_{E'}(\sigma) = \omega\} \subseteq E^*. \tag{2.2}$$

From the above definitions, we know that the projection operator $P_{E'}(\sigma)$, $\sigma \in E^*$, removes all the labels that are not in E' from string σ, while its inverse projection $P_{E'}^{-1}(\omega)$ associates to a sequence ω the set of all strings whose projection on E' is ω.

Example 2.3. Let $E = \{a, b, c\}$, $E' = \{b, c\}$ and $\sigma = abcbac$. The projection of σ on E' is $P_{E'}(\sigma) = bcbc$. Let $\omega = bc \in E'^*$. Then $P_{E'}^{-1}(\omega) = \{a^i b a^j c a^k\}$ where $i, j, k \in \mathbb{N} = \{0, 1, 2, \cdots\}$.

2.1.2 Automaton Models

In this subsection, we introduce the notions of non deterministic finite automata and deterministic finite automata.

Definition 2.7. A *nondeterministic finite automaton* (NFA) is a 5-tuple

$$G = (Q, E, \delta, Q_i, Q_m)$$

where

- Q is a finite set of states;
- E is a finite alphabet;
- $\delta : Q \times E \to 2^Q$ is a (partial) transition relation;
- $Q_i \subseteq Q$ is the set of initial states;
- $Q_m \subseteq Q$ is the set of final states.

An automaton can be described by a graph in which each state corresponds to a node, and is represented by a circle. In particular, the initial state is specified by an input arrow, and a final state is represented by a double circle.

Example 2.4. Fig. 2.1 is the graphical representation of an NFA $G = (Q, E, \delta, Q_i, Q_m)$ with $Q = \{1, 2, 3, 4\}$, $E = \{a, b, c, d\}$, $Q_i = \{3, 4\}$ and $Q_m = \{2\}$. Its transition function is given in Table 2.1.

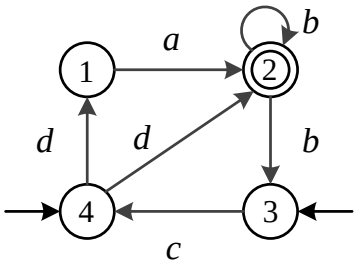

Figure 2.1 – Nondeterministic finite automaton

δ	a	b	c	d
1	2	-	-	-
2	-	$\{2, 3\}$	-	-
3	-	-	4	-
4	-	-	-	$\{1, 2\}$

Table 2.1 – Transition relation of the NFA in Fig. 2.1

The transition relation $\delta : Q \times E \to 2^Q$ can be extended to strings $\delta : Q \times E^* \to 2^Q$, which is defined as follows.

$$\begin{cases} \delta(q, \epsilon) = a, & for\ \ all\ q \in Q; \\ \delta(q, \sigma a) = \delta(\delta(q, \sigma), a), & for\ \sigma \in E^*\ and\ a \in E. \end{cases}$$

Note that for a subset of state set, e.g., $Q' \subseteq Q$, and a label $a \in E$, $\delta(Q', a)$ is defined as

$$\delta(Q', a) = \bigcup_{q' \in Q'} \delta(q', a).$$

Definition 2.8. Given an NFA $G = (Q, E, \delta, Q_i, Q_m)$, the set of labels enabled in state $q \in Q$ is defined as

$$\mathbb{E}(q) = \{a \in E \mid \delta(q, a) \text{ is defined}\}.$$

Definition 2.9. Consider a state $q \in Q$ in an NFA $G = (Q, E, \delta, Q_i, Q_m)$. State $q \in Q$ is said to be:

- *reachable* (or *accessible*) *from* state $q' \in Q$ if there is a string ω such that $q \in \delta(q', \omega)$. A state q reachable from an initial state $q_0 \in Q_i$ is simply called *reachable*;
- *co-reachable* (or *co-accessible*) *to* state $q' \in Q$ if there is a string ω such that $q' \in \delta(q, \omega)$. A state q co-reachable to a final state $q_m \in Q_m$ is simply called *co-reachable*;
- *dead* if no transition is enabled in state q, i.e., $\mathbb{E}(q) = \emptyset$.

Definition 2.10. Consider an NFA $G = (Q, E, \delta, Q_i, Q_m)$. It is said to be:

- *reachable* if all its states are reachable;
- *co-reachable* if all its states are co-reachable;
- *trim* if it is reachable and co-reachable.

In this thesis, the studied automaton is always supposed to be accessible. We use $G = (Q, E, \delta, Q_i)$ to represent an NFA if no final state is specified in the system.

In Def. 2.7, if the the initial states set Q_i is a singleton, i.e., $|Q_i| = 1$, and the transition relation δ is a one to one function: $\delta : Q \times E \to Q$, then the defined automaton becomes a *deterministic finite automaton* (DFA). The following is the formal definition of a DFA.

Definition 2.11. A *deterministic finite automaton* (DFA) is a 5-tuple

$$G = (Q, E, \delta, q_0, Q_m)$$

where objects Q, E, Q_m have the same interpretation as in the definition of NFA, $\delta : Q \times E \to Q$ is a one to one transition function, $q_0 \in Q$ is the initial state, and $Q_m \subseteq Q$ is the set of final states.

The transition function δ of a DFA $G = (Q, E, \delta, q_0, Q_m)$ can be extended to string $\delta : Q \times E^* \to Q$ in the same way as in NFA. From the definitions of NFA and DFA, it is clear that a DFA is a special NFA in which there is only one initial state, and the transition relation is a one to one function. As in the case of NFA, we use $G = (Q, E, \delta, q_0)$ to represent a DFA if no final state is specified.

Example 2.5. Fig. 2.2 is the graphical representation of a DFA $G = (Q, E, \delta, q_0, Q_m)$ with $Q = \{1, 2, 3, 4\}$, $E = \{a, b, c, d\}$, $q_0 = 4$ and $Q_m = \{2\}$. Its transition function is given in Table 2.2.

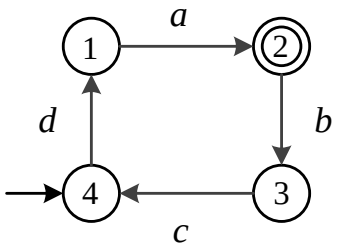

Figure 2.2 – Deterministic finite automaton

δ	a	b	c	d
1	2	-	-	-
2	-	3	-	-
3	-	-	4	-
4	-	-	-	1

Table 2.2 – Transition function of the DFA in Fig. 2.2

Now we give the definition of language generated by an NFA.

Definition 2.12. Given an NFA $G = (Q, E, \delta, Q_i)$, its generated language is defined as

$$L(G) = \{\sigma \in E^* \mid \exists q_0 \in Q_i : \delta(q_0, \sigma) \text{ is defined}\}.$$

2.2 Weighted Automata

In this section we recall some basics of weighted automata, which is the formalism used in this paper. Weighted automata are classical automata where the transitions carry weights belonging to a semiring. For more details we refer the reader to [Droste *et al.*, 2009].

2.2.1 Semirings

A semiring is a specific algebraic structure [Baccelli *et al.*, Wiley 1992; Gondran and Minoux, 2008; Heidergott *et al.*, 2014; Hardouin *et al.*, 2018]. The following is the definition of this structure.

Definition 2.13. A set $\mathcal{D}$ equipped with a binary operation $\oplus : \mathcal{D} \times \mathcal{D} \to \mathcal{D}$ is a *monoid*, denoted by $(\mathcal{D}, \oplus)$, if

- $\oplus$ is associative, i.e., $(a \oplus b) \oplus c = a \oplus (b \oplus c)$ for any $a, b, c \in \mathcal{D}$:
- $\oplus$ has a neutral element ε, i.e., $a \oplus \varepsilon = \varepsilon \oplus a = a$ for any $a \in \mathcal{D}$.

Monoid $(\mathcal{D}, \oplus)$ is said to be commutative if $\oplus$ is commutative, i.e., $a \oplus b = b \oplus a$ for any $a, b \in \mathcal{D}$.

Definition 2.14. A *semiring* is a quintuple $\mathbb{S} = (\mathcal{D}, \oplus, \otimes, \varepsilon, e)$ composed by a set $\mathcal{D}$, two binary operations $\oplus$ and $\otimes$ on $\mathcal{D}$, and two constant values $\varepsilon, e \in \mathcal{D}$ satisfying the following four axioms:

- $(\mathcal{D}, \oplus)$ is a commutative monoid with neutral element ε (also called zero element), i.e., $a \oplus b = b \oplus a$, $(a \oplus b) \oplus c = a \oplus (b \oplus c)$, and $\varepsilon \oplus a = a \oplus \varepsilon = a$ hold for any $a, b, c \in \mathcal{D}$;
- $(\mathcal{D}, \otimes)$ is a monoid with neutral element e (also called identity element), i.e., $(a \otimes b) \otimes c = a \otimes (b \otimes c)$, and $e \otimes a = a \otimes e = a$ hold for any $a, b, c \in \mathcal{D}$;
- the distributive laws $(a \oplus b) \otimes c = (a \otimes c) \oplus (b \otimes c)$ and $c \otimes (a \oplus b) = (c \otimes a) \oplus (c \otimes b)$ hold for any $a, b, c \in \mathcal{D}$;
- ε is absorbing for $\otimes$, i.e., $\varepsilon \otimes a = a \otimes \varepsilon = \varepsilon$.

A semiring $\mathbb{S}$ is said to be idempotent if addition $\oplus$ is idempotent, i.e., $a \oplus a = a$ holds for every $a \in \mathcal{D}$. An idempotent semiring is also called a *dioid*.

Example 2.6. The structure

$$\mathbb{S} = (\mathbb{R} \cup \{-\infty\}, \max, +, -\infty, 0)$$

is a typical instance of idempotent semiring, called max-plus semiring. By definition, $\mathcal{D} = \mathbb{R} \cup \{-\infty\}$, $\varepsilon = -\infty$ and $e = 0$. The maximum plays the role of addition $\oplus$, i.e., $\oplus = \max$, and the conventional addition plays the role of multiplication $\otimes$, i.e., $\otimes = +$. For any $a, b \in \mathcal{D}$, we have $a \otimes b = a + b$ and $a \oplus b = \max(a, b)$. Max-plus semiring is also called Max-plus algebra, and is often denoted by $\mathbb{R}_{max}$.

Example 2.7. Other important semirings include: The set of nonnegative integers equipped with the usual addition and usual multiplication operations, i.e., $\mathbb{S} = (\mathbb{N}, +, \times, 0, 1)$. Boolean semiring $(\{0, 1\}, \vee, \wedge, 0, 1)$, tropical semiring $(\mathbb{R} \cup \{+\infty\}, \min, +, +\infty, 0)$, max-plus semiring $(\mathbb{R} \cup \{-\infty\}, \max, +, -\infty, 0)$ and probability semiring $([0, 1], \max, \times, 0, 1)$.

The set of matrices with m rows and n columns over $\mathbb{S} = (\mathcal{D}, \oplus, \otimes, \varepsilon, e)$ is denoted by $\mathbb{S}^{m \times n}$. For matrices $A, B \in \mathbb{S}^{m \times n}$ and $C \in \mathbb{S}^{m \times n}$, the matrix sum and product under semiring

framework are defined in the following way:

$$\begin{aligned}[A \oplus B]_{ij} &\triangleq A_{ij} \oplus B_{ij} \\ [A \otimes C]_{ij} &\triangleq \bigoplus_{k=1}^{n} \left(A_{ik} \otimes C_{kj}\right)\end{aligned} \tag{2.3}$$

2.2.2 Weighted Automaton Models

Weighted automata (WAs) are a quantitative extension of finite state automata, which are originally introduced by Schützenberger in [Schützenberger, 1961]. More precisely, a WA is a finite state automaton in which each transition carries a weight that belongs to a semiring $\mathbb{S}$ [Schützenberger, 1961; Droste *et al.*, 2009]. The weight associated to a transition can model, e.g., the cost, the resource (energy), the time needed or the probability for executing the transition.

Definition 2.15. A *weighted automaton* (WA) over a semiring $\mathbb{S} = (\mathcal{D}, \oplus, \otimes, \varepsilon, e)$ is a tuple $G = (Q, E, \alpha, \mu, \beta)$ where

- Q and E are respectively a non-empty finite set of states and an alphabet;
- $\alpha \in \mathbb{S}^{1\times|Q|}$ is a row vector specifying the initial weights. A state $q \in Q$ is said to be an initial state iff $\alpha_q \neq \varepsilon$, and α_q is the corresponding initial weight;
- μ: $E \to \mathbb{S}^{|Q|\times|Q|}$ is a morphism representing the state transitions given by the family of matrices $\mu(a) \in \mathbb{S}^{|Q|\times|Q|}$, $a \in E$. For any string $\omega = e_1 \cdots e_k \in E^*$, we have $\mu(\omega) = \mu(e_1) \otimes \mu(e_2) \cdots \otimes \mu(e_k)$.
- $\beta \in \mathbb{S}^{|Q|\times 1}$ is a column vector specifying the final weights. A state q is said to be a final state iff $\beta_q \neq \varepsilon$, and β_q is the corresponding final weight.

Note that $|Q|$ represents the number of states of G and $\mathbb{S}$ can be any semiring in the above definition. In particular, it can be the Boolean semiring $(\{0,1\}, \vee, \wedge, 0, 1)$, in which case the corresponding WA is a classical finite automaton. Besides, if $\mathbb{S}$ represents the max-plus semiring, i.e., $\mathbb{S} = \mathbb{R}_{max} = (\mathbb{R} \cup \{-\infty\}, \max, +, -\infty, 0)$, then the corresponding WA becomes a max-plus automaton.

From the above definition, we know that a WA may be nondeterministic because there may be more than one element different from ε in α or/and in a row of $\mu(a)$ for any $a \in E$. Therefore, one can say that WA are classical NFA in which the transitions carry weights belonging to a semiring. We simply write $G = (Q, E, \alpha, \mu)$ if there is no final state in a WA.

A WA G can be associated with a useful graphical representation, which is a valued multigraph:

- Q corresponds to the set of nodes;
- an initial state $q \in Q$ (i.e., $\alpha_q \neq \varepsilon$) is characterized by an input arrow labeled by α_q representing its initial weight. Let us denote $Q_i \subseteq Q$ the set of initial states;
- a final state $q \in Q$ (i.e., $\beta_q \neq \varepsilon$) is characterized by an output arrow labeled by β_q representing its final weight;
- there exists an arrow from q to q', labeled by $a/\mu(a)_{qq'}$, iff $\mu(a)_{qq'} \neq \varepsilon$, $a \in E$. It represents the state transition when label a occurs. In this paper, $\mu(a)_{qq'}$ is interpreted as the activation weight [1] for label a before it can occur for the transition from q to q'.

Note that we assume the states of G are well ordered by natural numbers. With an abuse of notation, we denote $\mu(a)_{qq'}$ the element in the q^{th} row and q'^{th} column of the matrix $\mu(a)$ for any $a \in E$. The following is an equivalent representation of a WA $G = (Q, E, \alpha, \mu, \beta)$.

Definition 2.16. A WA $G = (Q, E, \alpha, \mu, \beta)$ can be equivalently defined by

$$G = (Q, E, t, Q_i, Q_m, \varrho, \rho)$$

where

- $t : Q \times E \times Q \to \mathcal{D}$ is the transition function, which is defined as $t(q, a, q') \triangleq \mu(a)_{qq'}$, for any $q, q' \in Q$;
- $Q_i \triangleq \{q \in Q \mid \alpha_q \neq \varepsilon\}$ is the set of initial states;
- $Q_m \triangleq \{q \in Q \mid \beta_q \neq \varepsilon\}$ is the set of final states;
- $\varrho : Q_i \to \mathcal{D}$ is the function of initial weights $\varrho(q) \triangleq \alpha_q$, for any $q \in Q_i$;
- $\rho : Q_m \to \mathcal{D}$ is the function of final weights $\rho(q) \triangleq \beta_q$, for any $q \in Q_m$.

Note that items α and β in Def. 2.15 are replaced by the set of initial states Q_i and the set of final states Q_m. Functions ϱ and ρ are adopted to specify the weights of initial and final states. Throughout this thesis, we denote by Q_i the set of initial states of a WA G. We simply write $G = (Q, E, t, Q_i, \varrho)$ if no final state is specified.

Example 2.8. Fig. 2.3 shows the graphical representation of a WA $G = (Q, E, \alpha, \mu, \beta)$ with set of states $Q = \{1, 2, 3, 4\}$, alphabet $E = \{a, b, c, d\}$, transitions $\mu(a)_{1,2} = 2$, $\mu(b)_{2,2} = 1$, $\mu(b)_{2,3} = 6$, $\mu(c)_{3,4} = 3$, $\mu(d)_{4,1} = 2$, $\mu(d)_{4,2} = 1$, initial weights vector $\alpha = (\varepsilon, \varepsilon, 2, e)$, and final weights vector $\beta = (\varepsilon, 3, \varepsilon, \varepsilon)$. The other values for $\mu(a)$, $\mu(b)$, $\mu(c)$ and $\mu(d)$ are equal to ε.

1. Note that the activation weight can represent an amount of time, resources (energy), or a cost, depending on the interpretation of weights adapted in the underlying work.

The equivalent representation of this automaton is $G = (Q, E, t, Q_i, Q_m, \varrho, \rho)$ with $Q = \{1,2,3,4\}$, $E = \{a,b,c,d\}$, $Q_i = \{3,4\}$, $Q_m = \{2\}$, $\varrho(1) = \varrho(2) = \varepsilon$, $\varrho(3) = 2$, $\varrho(4) = e$, $\rho(1) = \rho(3) = \rho(4) = \varepsilon$, $\rho(2) = 3$. The transitions $t(q_i, a, q_j) = \varepsilon$ for any $q_i, q_j \in \{1,2,3,4\}$ and any $a \in E$, except $t(1,a,2) = 2$, $t(2,b,2) = 1$, $t(2,b,3) = 6$, $t(3,c,4) = 3$, $t(4,d,1) = 2$ and $t(4,d,2) = 1$.

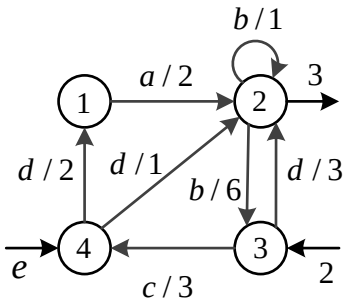

Figure 2.3 – Weighted automaton

Now we introduce the notion of unambiguity property for a WA.

Definition 2.17. Given a WA $G = (Q, E, \alpha, \mu)$, a path of length k is defined as a sequence of transitions

$$\pi = (q_1, e_1, q_2)\,(q_2, e_2, q_3) \cdots (q_k, e_k, q_{k+1}) \tag{2.4}$$

where $q_i \in Q$, $e_i \in E$ and $\mu(e_i)_{q_i q_{i+1}} \neq \varepsilon$ for $i = 1, \cdots, k$.

Path π is said to be labeled by string $e_1 e_2 \cdots e_k$, and π is a circuit if q_1 coincides with q_{k+1}. We use notation $p \overset{\omega}{\rightsquigarrow} q$ to represent the set of paths labeled by string $\omega \in E^*$ leading from state p to state q. By convention, if ω equals the empty string ϵ, i.e., $\omega = \epsilon$, then $p \overset{\omega}{\rightsquigarrow} q$ is the empty set, i.e., $p \overset{\omega}{\rightsquigarrow} q = \emptyset$. For any $P, R \subseteq Q$, we denote by $P \overset{\omega}{\rightsquigarrow} R$ the union of $p \overset{\omega}{\rightsquigarrow} q$ for all $p \in P$ and $q \in R$.

Definition 2.18. A WA $G = (Q, E, \alpha, \mu)$ is said to be *unambiguous* if $\forall q \in Q$, $\forall \omega \in E^*$, $|Q_i \overset{\omega}{\rightsquigarrow} \{q\}| \leq 1$, where Q_i is the set of initial states of G.

In simple words, the unambiguity requires that for any state q of G and any string ω in E^*, there is at most one path labeled by ω leading to q from an initial state. WAs with unambiguity property are called *unambiguous weighted automata* (UWAs).

Remark 2.1. Different definitions can be found for the notion of unambiguity in the literature. In [Béal *et al.*, 2008], the unambiguity is defined as: $\forall p, q \in Q, \forall \omega \in E^*, |p \overset{\omega}{\rightsquigarrow} q| \leq 1$. That is, for any $p, q \in Q$ and any string $\omega \in E^*$, there exists at most one path leads from

p to q labeled by ω. In [Klimann *et al.*, 2004; Kirsten and Lombardy, 2009], an automaton is said to be unambiguous if $\forall \omega \in E^*, |Q_i \overset{\omega}{\rightsquigarrow} Q_m| \leq 1$, where Q_m is the set of final states. That is, the unambiguity requires that if for any string ω in E^*, there is at most one successful path[2] of label ω. The unambiguity given in the above two definitions and the unambiguity given in Def. 2.18 have the following relationship. (1) The unambiguity in Def. 2.18 is more restrictive than the first definition above if the automaton is accessible. That is, for a given accessible WA G, if $\forall q \in Q, \forall \omega \in E^*, |Q_i \overset{\omega}{\rightsquigarrow} \{q\}| \leq 1$, then $\forall p, q \in Q, \forall \omega \in E^*, |p \overset{\omega}{\rightsquigarrow} q| \leq 1$. This can be proved as follows. Assume $\exists p, q \in Q, \exists \omega \in E^*, |p \overset{\omega}{\rightsquigarrow} q| > 1$. Since G is accessible, then there is a path leading from an initial state to state p (assume this path is labeled by ν). Hence, there must exist more than one path labeled by string $\nu\omega$ leading from an initial state leading to state q. That is, we have $\exists q \in Q, \exists \omega \in E^*, |Q_i \overset{\omega}{\rightsquigarrow} \{q\}| > 1$ (here $\omega = \nu\omega$). The structure in Fig. 2.4 illustrates this visually. (2) The unambiguity in Def. 2.18 is more general than the second one if the automaton is co-accessible. That is, for a given co-accessible WA G, if $\forall \omega \in E^*, |Q_i \overset{\omega}{\rightsquigarrow} Q_m| \leq 1$, then $\forall q \in Q, \forall \omega \in E^*, |Q_i \overset{\omega}{\rightsquigarrow} \{q\}| \leq 1$. This can be proved as follows. Assume $\exists q \in Q, \exists \omega \in E^*, |Q_i \overset{\omega}{\rightsquigarrow} \{q\}| > 1$. Since G is accessible, then there is a path leading from state q to a final state (assume this path is labeled by string ν). Hence, there must exist more than one path labeled by string $\omega\nu$ leading from an initial state. The structure in Fig. 2.5 illustrates this process.

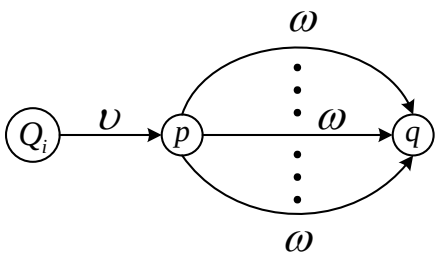

Figure 2.4 – Structure explaining the relation between the unambiguity in Def. 2.18 and in [*Béal et al.*, 2008].

Remark 2.2. Given a WA $G = (Q, E, \alpha, \mu)$, its unambiguity given in Def. 2.18 can be verified by some algorithms in the literature, such as [Allauzen *et al.*, 2011; Weber and Seidl, 1991]. In fact, algorithms in these two work are proposed for checking the unambiguity defined in [Klimann *et al.*, 2004]. That is, these algorithms are introduced to check if for any string ω belongs to E^*, there is at most one successful path labeled by ω. Although this unambiguity is different from the unambiguity defined in this thesis, according to their relationship stated in Remark 2.1, the algorithms in [Allauzen *et al.*, 2011; Weber and Seidl, 1991] can be used to

2. A path starts from an initial state and ends with a final state is called a successful path.

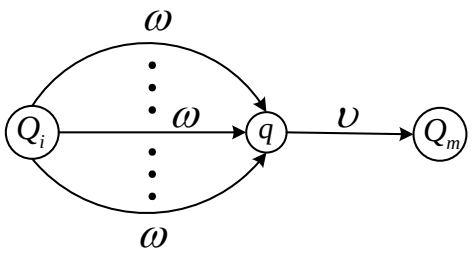

Figure 2.5 – Structure explaining the relation between the unambiguity in Def. 2.18 and in [*Klimann et al.*, 2004].

check the unambiguity in Def. 2.18. This can be done by considering a state q of WA G to be the unique final state when we want to check, for any string ω, whether there is at most one path labeled by ω leading to q from the initial states, that is, at most one successful path.

In the remaining of this thesis, the unambiguity property of a WA refers to the one defined in Def. 2.18.

Example 2.9. The WA presented in Fig. 2.6 is unambiguous. This automaton is a modification of the WA in Fig. 2.3 by removing the output transition of initial state 3 that is labeled by d.

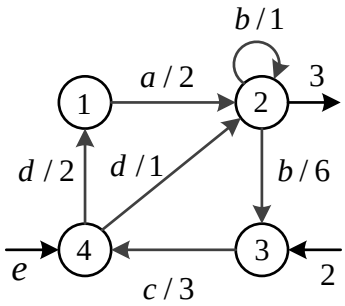

Figure 2.6 – Unambiguous weighted automaton

Remark 2.3. For a WA $G = (Q, E, \alpha, \mu)$ where all states are initial states, i.e., $Q_i = Q$, G is unambiguous iff every state in G has no two or more input transitions labeled by the same symbol. On one hand, suppose (q', a, q) and (q'', a, q) with $q', q'' \in Q$ and $a \in E$, are two input transitions of state q in G. Since q' and q'' are final states, then there exists a string $\omega = a$ such that $|Q_i \overset{\omega}{\rightsquigarrow} \{q\}| > 1$. That is G is not unambiguous. On the other hand, assume G is not unambiguous, i.e., $|Q_i \overset{\omega}{\rightsquigarrow} \{q\}| > 1$. Let a be the last label of ω. Then we know that there is more than one input transition of q that are labeled by a. In the case where $Q_i \neq Q$, it is easy to check that any state in G has no two or more input transitions labeled by the

same symbol is a sufficient condition for the unambiguity. Whereas, a WA is unambiguous does not mean that any state in G has no two or more input transitions with the same label.

Remark 2.4. As in the case of finite automata, a WA is said to be deterministic if it has a unique initial state and from any state, no two output transitions are labeled by the same symbol. From here we know that the determinism implies the unambiguity, but the reverse is not always true.

Example 2.10. Consider the WA shown in Fig. 2.7. This automaton is unambiguous. However, we find that one of its states, i.e., state 2, has two input transitions labeled by b. In addition, according to Remark 2.4, this automaton is not deterministic since it has two initial states and state 2 has two output transitions labeled by b.

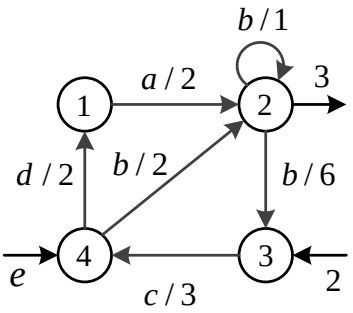

Figure 2.7 – Unambiguous weighted automaton in which state 2 has two input transitions labeled by b

The following subsection introduces the notions of state vector, weighted sequence generated by a path and weighted language generated by a WA.

2.2.3 Generated Language of Weighted Automaton

Definition 2.19. Given an arbitrary path $\pi = (q_1, e_1, q_2)\,(q_2, e_2, q_3)\cdots(q_k, e_k, q_{k+1})$ of a WA $G = (Q, E, \alpha, \mu)$, its weight is denoted by $\mathbb{W}(\pi)$, and is defined as

$$\mathbb{W}(\pi) = \begin{cases} \alpha_{q_1} \otimes \bigotimes\limits_{i=1,\cdots,k} \mu(e_i)_{q_i q_{i+1}}, & if\ q_1 \in Q_i; \\ \bigotimes\limits_{i=1,\cdots,k} \mu(e_i)_{q_i q_{i+1}}, & otherwise. \end{cases} \tag{2.5}$$

Note that if the first state in a path is an initial state, then its initial weight is included in the weight of the path.

Definition 2.20. Given a WA $G = (Q, E, \alpha, \mu)$, its state vector is denoted by $x(\omega) \in \mathbb{S}^{1\times|Q|}$, $\omega \in E^*$, and is defined as

$$x(\omega) = \begin{cases} \alpha, & if\ \omega = \epsilon; \\ \alpha \otimes \mu(\omega), & otherwise. \end{cases} \tag{2.6}$$

For any string $\omega \in E^*$, by considering the matrix sum and product defined in Eq. (2.3), it can be verified that $x(\omega)_q$ corresponds to the sum $\oplus$ of weights of paths labeled by ω from an initial state to state q:

$$x(\omega)_q = \bigoplus_{\pi \in Q_i \overset{\omega}{\rightsquigarrow} q} \mathbb{W}(\pi) = \bigoplus_{p \in Q_i} \alpha_p \otimes \mu(w)_{pq}. \tag{2.7}$$

Note that $x(\omega)_q$ is a real value, and we interpret it as the weight to reach state q according to ω. By convention, $x(\omega)_{q_k} = \varepsilon$ if q_k is not reachable via ω from an initial state. Again, different interpretations can be associated to the weights of WA $G = (Q, E, \alpha, \mu)$, such as the amount of time, resources (energy), or cost of firing the transitions. If the time interpretation is adopted, then we say that $x(\omega)_q$ is the time instant at which state q is reached when string ω is completed. If the energy interpretation is adopted, then $x(\omega)_q$ is interpreted as the energy consumed to reach state q by the occurrence of sequence ω.

In addition, $\otimes$ and $\oplus$ can represent different operations, depending on the underlying semiring $\mathbb{S}$. For example, in probability semiring, $\oplus$ represents the *max* operation, i.e., $\oplus = \max$, and $\otimes$ represents the usual multiplication, i.e., $\otimes = \times$. In max-plus semiring $\mathbb{R}_{max}$, $\oplus$ represents the *max* operation, i.e., $\oplus = \max$, and $\otimes$ is equal to the usual addition, i.e., $\otimes = +$.

Example 2.11. Consider the WA $G = (Q, E, \alpha, \mu)$ in Fig. 2.8 with $Q = \{1, 2, 3, 4\}$, $E = \{u, b\}$, $\mu(u)_{1,2} = 1$, $\mu(u)_{4,3} = 2$, $\mu(b)_{2,2} = 6$, $\mu(b)_{2,1} = 4$, $\mu(b)_{3,2} = 3$, $\alpha = (e, \varepsilon, \varepsilon, e)$. The other values for α, β, $\mu(b)$ and $\mu(u)$ are equal to ε. We assume that the underlying semiring of G is the max-plus semiring.

It can be checked that $x(u) = (\varepsilon, 1, 2, \varepsilon)$, $x(ub) = (5, 7, \varepsilon, \varepsilon)$, and $|Q_i \overset{ub}{\rightsquigarrow} 2| = 2$ since $\pi_1 = (1, u, 2)(2, b, 2)$ and $\pi_2 = (4, u, 3)(3, b, 2)$ are the two paths labeled by ub from an initial state to state 2. When u is completed, state 2 is reached with a weight value of 1 because $x(u)_2 = 1$. Besides, $x(ub)_2 = 7$ means that state 2 is reached with a weight value of 7 when ub is completed. Here we observe that labels along path π_1 occur as soon as the activation weights of the corresponding transitions have been reached. However, this

is not always the case. In particular, along path π_2, state 3 is first reached with a weight value of 2 since $x(u)_3 = 2$, and state 2 is then reached with a weight value of 7 when string ub is completed. Whereas the activation weight of label b for transition $(3, b, 2)$ is equal to $\mu(b)_{3,2} = 3$, this label here occurs 5 units of weight $(7 - 2 = 5)$ after state 3 is reached. Value 5 is the real weight needed for transferring from state 3 to state 2 along path π_2. This is consistent with the interpretation that $\mu(b)_{3,2}$ specifies the activation weight before label b can occur for this transition.

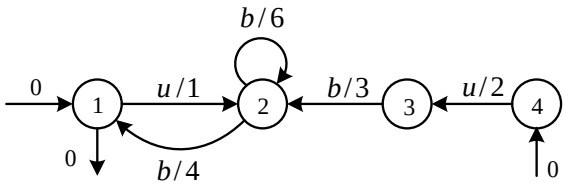

Figure 2.8 – Weighted automaton

Consider the occurrence of label $a \in E$ for transition (q, a, q'), $\mu(a)_{qq'} \neq \varepsilon$ in WA $G = (Q, E, \alpha, \mu)$. Suppose that state q is reached with a weight value of τ_q. It may happen that label a occurs with a weight value that is different from $\tau_q \otimes \mu(a)_{qq'}$. This can happen if $\exists q \in Q, \exists \omega \in E^* : |Q_i \overset{\omega}{\rightsquigarrow} q| \geq 1$, i.e., several paths labeled by ω leading from an initial state to state q. Such a configuration captures a synchronization occurring in the system, which is a common phenomenon in DESs.

Formally, considering the occurrence of label $a \in E$ for transition (q, a, q'), $\mu(a)_{qq'} \neq \varepsilon$, suppose that q is reached with a weight value of τ_q, it may happen that label a occurs with a weight value that is strictly greater than $\tau_q + \mu(a)_{qq'}$. This can happen if $\exists q \in Q$, $\exists \omega \in E^* : |Q_i \overset{\omega}{\rightsquigarrow} q| \geq 1$, i.e., several paths labeled by ω leading from an initial state to state q. Such a configuration captures a synchronization occurring in the system, which is a common phenomenon in DES.

Based on the above discussion, we can say that a WA describes a system characterized by synchronizations between concurrent sequential weighted processes. A path in a WA represents a concurrent process whose transition weights denote the activation weight before the transitions can fire. The weighted evolution of the automaton will be given by the synchronization of the evolutions of processes corresponding to paths with the same labeling and reaching the same state. Therefore, we associate to each path π of WA G a weighted sequence corresponding to this synchronization.

Definition 2.21. Given a path $\pi = (q_0, e_1, q_1)(q_1, e_2, q_2) \cdots (q_{k-1}, e_k, q_k)$ of a WA $G = (Q, E, \alpha, \mu)$ with $q_0 \in Q_i$, the weighted sequence generated by π is denoted by $\sigma(\pi) \in$

$(E \times \mathcal{D})^*$ and is defined as follows:

$$\sigma(\pi) = (e_1, \tau_1)(e_2, \tau_2) \cdots (e_k, \tau_k) \tag{2.8}$$

where τ_j is defined by $\tau_j = x(e_1, \cdots, e_j)_{q_j}$ for $j = 1, \cdots, k$.

We use notation $q_0 \overset{\sigma(\pi)}{\rightsquigarrow} q_k$ to denote the fact that path π starts from initial state q_0 and generates the weighted sequence $\sigma(\pi)$ reaching state q_k. Such a generated weighted sequence consists of pairs composed by a label and a weight value, i.e., (e_j, τ_j) with $e_j \in E$, $\tau_j \in \mathcal{D}$ for $j = 1, 2, \ldots, k$. In simple words, it specifies a sequence of labels, i.e., the unweighted sequence generated by π, and their occurrence weights.

If WA G is unambiguous, then $x(\omega)_q$ defined in Eq. (2.7) as well as the weighted sequence generated by a path π defined in Eq. (2.8) can be simplified as follows.

Definition 2.22. Consider an arbitrary string $\omega = e_1 \cdots e_k \in E^*$ and a state $q \in Q$ in UWA $G = (Q, E, \alpha, \mu)$. The value $x(\omega)_q$ defined in Eq. (2.7) can be simplified to:

$$x(\omega)_q = \begin{cases} \alpha_p \otimes \mu(\omega)_{pq}, & if\ q \text{ is reachable from p} \\ & \text{according to } \omega\ (i.e. |p \overset{\omega}{\rightsquigarrow} q| = 1); \\ \varepsilon, & otherwise. \end{cases} \tag{2.9}$$

This is simply because, for any $\omega \in E^*$ and any $q \in Q$, there exist at most one path labeled by ω leading from an initial state to state q in a UWA. As a result, the weighted sequence generated by a path can be simplified.

Definition 2.23. Given a path $\pi = (q_0, e_1, q_1)(q_1, e_2, q_2) \cdots (q_{k-1}, e_k, q_k)$ of a UWA $G = (Q, E, \alpha, \mu)$ with $q_0 \in Q_i$, the weighted sequence $(E \times \mathcal{D})^*$ generated by π defined in Eq. (2.8) can be simplified to:

$$\sigma(\pi) = (e_1, \tau_1)(e_2, \tau_2) \cdots (e_k, \tau_k) \tag{2.10}$$

where $\tau_1 = \alpha_{q_0} \otimes \mu(e_1)_{q_0 q_1}$, $\tau_i = \tau_{i-1} \otimes \mu(e_i)_{q_{i-1} q_i}$ for $i = 2, \cdots, k$.

Based on the definition of weighted sequence generated by a path π, we define the generated weighted language of a WA as follows.

Definition 2.24. Given a WA $G = (Q, E, \alpha, \mu)$, the generated weighted language $L(G)$ (set of all weighted sequences generated by G) is defined as:

$$L(G) = \{\sigma \in (E \times \mathcal{D})^* \mid \exists q \in Q, \exists \omega \in E^*,\ \exists \pi \in Q_i \overset{\omega}{\rightsquigarrow} q :\ \sigma(\pi) = \sigma\}. \tag{2.11}$$

For any weighted sequence $\sigma \in L(G)$, we denote by $(\mathbb{E}_f(\sigma), w_f(\sigma))$ the last pair in σ. That is, $\mathbb{E}_f(\sigma)$ is the last label of σ, and $w_f(\sigma)$ is the completion weight of σ.

Example 2.12. Consider again path $\pi_1 = (1,u,2)(2,b,2)$ and path $\pi_2 = (4,u,3)(3,b,2)$ of WA G in Fig. 2.8 with $Q_i = \{1,4\}$. Let max-plus semiring be the underlying semiring of G. Then, the generated weighted sequences are $\sigma_1 = \sigma(\pi_1) = (u,1)(b,7)$ and $\sigma_2 = \sigma(\pi_2) = (u,2)(b,7)$. Therefore, $(\mathbb{E}_f(\sigma_1), w_f(\sigma_1)) = (b,7)$ and $(\mathbb{E}_f(\sigma_2), w_f(\sigma_2)) = (b,7)$.

Remark 2.5. In the existing literature, e.g., [Droste and Gastin, 2007; Buchholz, 2008; Droste *et al.*, 2009], the behavior of a WA is usually represented by a mapping, called a formal power series. More precisely, for a WA G defined over semiring $\mathbb{S}$, its behavior is defined by the formal power series $l(G) : E^* \to \mathbb{S}$ with $L(G)(\omega) = \alpha \otimes \mu(\omega) \otimes \beta$, which assigns to each string a weight in semiring $\mathbb{S}$. Compared with such a formal series, the generated weighted language of G, defined in this thesis, gives more information on the evolution of the system. In fact, formal power series only compute the completion weight of a task (represented by a string), which is the sum $\oplus$ of the weights of its successful paths, while weighted sequence specifies a series of labels and their occurrence weights. In general and for some purposes, such as for the state estimation problem investigated in this thesis, more information is needed. Hence, from a practical point of view, the generated weighted language defined in the thesis is useful.

As in the classical automata, for any weighted sequences $\sigma_1, \sigma_2 \in (E \times \mathcal{D})^*$, notation $\sigma_1 \cdot \sigma_2$ is used to represent their concatenation.

Definition 2.25. The concatenation of two weighted sequences $\sigma_1, \sigma_2 \in (E \times \mathcal{D})^*$ is a new weighted sequence $\sigma_1 \cdot \sigma_2$ composed by the sequence of pairs in σ_1 followed by the sequence of pairs in σ_2.

Definition 2.26. If a weighted sequence $\omega \in (E \times \mathcal{D})^*$ can be represented as $\omega = \nu\gamma\eta$ with $\nu\gamma, \eta \in (E \times \mathcal{D})^*$, then weighted sequence ν is called a prefix of ω, weighted sequence γ is called a substring of ω, and weighted sequence η is called a suffix of ω.

The projection operation on $E' \subseteq E$, i.e., $P_{E'} : E^* \to E'^*$, defined in Eq. 2.1, is extended to weighted sequences $\sigma = (e_1,t_1)\cdots(e_k,t_k) \in (E \times \mathcal{D})^*$. The projection of a weighted sequence on $E' \subseteq E$ is denoted by $P_{E'} : (E \times \mathcal{D})^* \to (E' \times \mathcal{D})^*$ where $P(\sigma)$ is the weighted string obtained from σ by erasing all pairs corresponding to labels belonging to E'. For instance, consider $\sigma = (a,1)(u,4)(a,5)$ where $a \in E$ and $u \in E'$. Then, we have $P_{E'}(\sigma) =$

$(a,1)(a,5)$. In this thesis, we denote by $P_{E'}(L(G))$ the projection of the generated language of WA G on E', that is,

$$P_{E'}(L(G)) = \left\{\sigma \in (E' \times \mathcal{D})^* \mid \exists \sigma' \in L(G) : P_{E'}(\sigma') = \sigma\right\}. \tag{2.12}$$

If E' represents the set of observable labels, then $P_{E'}(L(G))$ is the set of all possible observations, i.e., all weighted sequences that can be obtained by an external agent during the evolution of system G.

In the case of classical finite automata, a *trajectory* of the system is represented by an infinite sequence of labels that the system may generate. The set of all possible trajectories of the system is called ω-language [Thistle and Wonham, 1994; Shu and Lin, 2011]. Similarly, given a WA G, a trajectory is represented by an infinite sequence of pairs (label, weight) that G may generate. The set of all possible trajectories of G defines the ω-language, which is denoted by $L^{\omega}(G)$. For an arbitrary finite or infinite sequence σ that can be generated by G, we denote $Pr(\sigma)$ the set of all its prefixes.

3

State Estimation and Fault Diagnosis of Weighted Automata

This chapter investigates the current-state estimation and fault diagnosis problems for weighted automata (WAs).

Contents

3.1 Introduction

In this chapter, we focus on current-state estimation and fault diagnosis problems for WAs where only partial labels (events) of the system are observable for the external agents. That is, the set of label is divided into two disjoint subsets, i.e., observable and unobservable subsets. We consider the case where the observations are static, i.e., an label is either observed or not observed at every state in which it can occur. Besides, the failures are associated to some of the unobservable labels, and supposed to be permanent. In other words, the system cannot spontaneously move from the fault state to a normal state. Given an observation and a weight value, state estimation consists in calculating the set of possible states that are consistent with them, and fault diagnosis is to determine whether a fault label has occurred or not. Note that the general terminology *fault diagnosis* refers to detecting the occurrence of fault labels in this chapter.

This chapter is organized as follows. Section 3.2 formally states the problem of state estimation and fault diagnosis and specifies our assumptions. In subsection 3.3.1, we propose algorithms to estimate the consistent current states for any observed weighted sequence over the operation of the system. Following the proposed algorithms, one numerical example is shown to illustrate them. In Subsection 3.3.2, we analyse the computational complexity of state estimation for a fixed-length observation using our approach. Section 3.3.3 shares the failure experience of building a structure like "observer" for state estimation. In Section 3.4, we extend the state estimation approach to deal with fault diagnosis. We first propose an algorithm to construct the augmented automaton of a WA, and another algorithm is presented to summarize how to perform on-line fault diagnosis. Finally, conclusions are drawn in Section 3.5.

3.2 Problem Statement

3.2.1 State Estimation

The studied WA is completely known, namely, initial states, state transitions, weights of transitions are available. The label set E is partitioned into two disjoint parts: the observable part E_o and the unobservable part E_{uo}. Considering that the firing of unobservable labels cannot be distinguished by an external agent observing the system evolution, we assume that all the unobservable labels are represented by the same symbol u, i.e., $E_{uo} = u$ and $E = E_o \cup \{u\}$. We make the following assumption when dealing with state estimation.

Assumption 3.1. There is no circuit labelled only by unobservable labels.

Assumption 3.1 implies that generated sequences of unobservable labels (strings composed exclusively of u) have finite length.

For any state $q \in Q$, we use notation $q^\bullet$ to represent the set of its output transitions, i.e., $q^\bullet = \left\{(q, a, q') | a \in E, q' \in Q : \mu(a)_{qq'} \neq \varepsilon\right\}$. Given a weighted sequence $\sigma = (e_1, t_1)(e_2, t_2) \cdots (e_k, t_k)$, we denote $lab(\sigma) = e_1 e_2 \cdots e_k$ the sequence of labels associated with σ, neglecting the occurrence weights. The fact that a label $a \in E$ appears in sequence $lab(\sigma)$ is denoted by $a \in lab(\sigma)$.

Given an observed weighted sequence $\sigma_o \in P(L(G))$ and a weight value τ, we define the set $C(\sigma_o, \tau)$ of $(\sigma_o, \tau)-$consistent states as the set of all possible states in which the system may be at τ after the observation of σ_o. We assume that there is no further observation after the last label in σ_o.

Definition 3.1. Given an observed weighted sequence $\sigma_o \in P(L(G))$ and a weight value $\tau \geq t_f(\sigma_o)$, the set of all $(\sigma_o, \tau)-$ *consistent states* is defined as

$$\begin{aligned} C(\sigma_o, \tau) = \{q \in Q \mid (\exists \sigma \in L(G), \exists q_0 \in Q_i : q_0 \overset{\sigma}{\rightsquigarrow} q, P(\sigma) = \sigma_o, t_f(\sigma) \leq \tau) \wedge \\ ((q^\bullet = \emptyset) \vee (\exists (q, a, q') \in q^\bullet : \tau < x(lab(\sigma)a)_{q'}))\} \end{aligned} \tag{3.1}$$

Note that $x(lab(\sigma)a)_{q'}$ is the element in state vector $x(lab(\sigma)a)$ that corresponds to state q'. From the above definition, a state q without output transition is a consistent state, if there exists a weighted sequence σ from an initial state to q such that the projection of σ coincides with the observation, and the completion weight of σ is less than or equal to given value τ. If q has output transition(s), then at least one of the output transitions is required to occur after τ.

Problem 3.1. The state estimation problem of a WA consists in finding a systematic approach to characterize the set $C(\sigma_o, \tau)$ for any observation σ_o and any time instant $\tau \geq t_f(\sigma_o)$.

Example 3.1. Consider again the automaton G in Fig. 2.8. Assume that the underlying semiring of G is the max-plus semiring. Let $\sigma_o = (b, 7)(b, 11)$ and $\tau = 11.5$. It can be verified that the set of consistent states is $C(\sigma_o, \tau) = \{1\}$. In fact, $\pi = (1, u, 2)(2, b, 2)(2, b, 1)$ is the unique path whose weighted sequence is consistent with σ_o. We have $\sigma = \sigma(\pi) = (u, 1)(b, 7)(b, 11)$, $1 \overset{\sigma}{\rightsquigarrow} 1$, $t_f(\sigma) = 11 \leq 11.5$, and $P(\sigma) = \sigma_o$. Moreover, the occurrence time instant of the output transition $(1, u, 2)$ of state 1 is $x(ubbu)_2 = 12$, which is greater than 11.5. Therefore, state 1 is consistent with σ_o and τ. □

3.2.2 Fault Diagnosis

The label set E is partitioned into two disjoint parts: the observable part E_o and the unobservable part E_{uo}. And E_{uo} is partitioned as $E_{uo} = E_f \cup E_{reg}$ where E_f is the set of fault labels [1] (modeling the faulty behavior), while E_{reg} is the set of regular labels that, although not observable, do not describe a faulty behavior. In addition to Assumption 3.1 in Subsection 3.2.1, we make another hypothesis when dealing with fault diagnosis.

Assumption 3.2. For any two different states $p, q \in Q$, there is at most one unobservable state transition from p to q.

Remark 3.1. Unobservable state transition is a transition that is labeled by an unobservable label. For state estimation described in Subsection 3.2.1, it is considered that an external agent cannot distinguish one unobservable label from the others, and all the unobservable labels are denoted with the same symbol u. In fault diagnosis problem, we admit different symbols for unobservable labels since fault and normal unobservable labels must be distinguished. Then Assumption 3.2 is needed to avoid the confusion between unobservable labels for a transition from state p to state q. □

Given an observed weighted sequence, the fault diagnosis problem is to detect the occurrence of some labels that belong to fault class E_f. In fault diagnosis problem, rather than being interested in finding the states in which the system can be, we are interested in determining the set of labels that may have occurred.

Definition 3.2. Given an observed weighted sequence $\sigma_o \in P(L(G))$ and a weight value $\tau \geq t_f(\sigma_o)$. The set of weighted sequences consistent with σ_o and τ is

$$\begin{aligned} S(\sigma_o, \tau) = \{\sigma \in L(G) \mid (P(\sigma) = \sigma_o, t_f(\sigma) \leq \tau) \wedge \\ (\exists q_0 \in Q_i, \exists q \in Q : q_0 \overset{\sigma}{\rightsquigarrow} q, (q^\bullet = \emptyset) \vee (\exists (q, a, q') \in q^\bullet : x(lab(\sigma)a)_{q'} > \tau))\} \end{aligned} \tag{3.2}$$

In simple words, weighted sequence σ is a consistent sequence, if its projection coincides with the observation, and its completion weight is less than or equal to τ. In addition, there must be a state q that can be reached by σ from an initial state such that it has no output transition or at least one of its output transitions is required to occur with a weight greater than τ.

1. Note that the set E_f could be further partitioned to r disjoint subsets, i.e., $E_f = E_f^1 \cup E_f^2 \cup \cdots \cup E_f^r$ representing different fault classes. However, in this chapter, we consider single fault class scenario for sake of clarity and without loss of generality.

We extend the notion of diagnoser for PNs proposed in [Cabasino *et al.*, 2011] to the framework of WAs.

Definition 3.3. A diagnoser is a function $\varphi : \left[(E \times \mathcal{D})^*\right] \times \mathcal{D} \times E_f$ that associates to each observation $\sigma_o \in P(L(G))$, to each given weight τ, and to fault class E_f, a diagnosis state.

- $\varphi(\sigma_o, \tau, E_f) = N$ if $\forall \sigma \in S(\sigma_o, \tau)$ and $\forall e_f \in E_f$, we have $e_f \notin lab(\sigma)$.
  In such a case fault cannot have occurred, because none of the generated sequences consistent with the observation contains fault labels belonging to class E_f.
- $\varphi(\sigma_o, \tau, E_f) = F$ if $\forall \sigma \in S(\sigma_o, \tau)$, $\exists e_f \in E_f$ such that $e_f \in lab(\sigma)$.
  In such a case fault must have occurred, because all generated sequences consistent with the observation contain at least one fault label belonging to class E_f.
- $\varphi(\sigma_o, \tau, E_f) = U$ if the following conditions are satisfied:
  (i) if $\exists \sigma \in S(\sigma_o, \tau)$, $\exists e_f \in E_f$ such that $e_f \in lab(\sigma)$;
  (ii) if $\exists \sigma' \in S(\sigma_o, \tau)$ such that $\forall e_f \in E_f$, we have $e_f \notin lab(\sigma)$.
  In such a case a fault label belonging to E_f may have occurred or not, namely, it is uncertain.

Problem 3.2. The fault diagnosis problem of a WA consists in finding a systematic approach to compute the diagnosis state for any observation σ_o, any given weight $\tau \geq t_f(\sigma_o)$ and the fault class E_f.

3.3 State Estimation of Weighted Automata

3.3.1 An Online State Estimation Approach

In this subsection we introduce an approach to solve Problem 3.1.

Definition 3.4. Given a WA $G = (Q, E, \alpha, \mu, \beta)$, a pair (q_j, ω_j) with $q_j \in Q$ and $\omega_j \in E^*$ is said to be *compatible* with a weighted sequence $\sigma_o = (e_1, t_1)(e_2, t_2) \cdots (e_j, t_j) \in P(L(G))$, $j \geq 1$, if there exists path π labeled by ω_j ending with e_j from an initial state q_0 in G such that $P(\sigma(\pi)) = \sigma_o$ and $q_0 \overset{\sigma(\pi)}{\rightsquigarrow} q_j$. We define $W(\sigma_o)$ as the set of all pairs compatible with σ_o.

We first define two functions $UpdateW$ and $ConsistentSet$, which are given in Algorithms 3.1 and 3.2 respectively. Then we propose the propositions to show that the current state according to system's evolution can be correctly calculated by combining the functions $UpdateW$ and $ConsistentSet$.

Function $UpdateW$ in Algorithm 3.1 is employed to update the compatible set each time a new pair (e, t) is observed.

Algorithm 3.1: Update the compatible set

Input: $W(\sigma_o)$, a pair $(e,t) \in (E_o \times \mathbb{R})$.
Output: $W(\sigma_o \cdot (e,t))$.

function $UpdateW((e,t), W(\sigma_o))$
$W(\sigma_o \cdot (e,t)) := \emptyset$
for each $(q',\omega') \in W(\sigma_o)$ **do**
  for each $q \in Q$ **do**
    for $i = 0$ to $|Q|-1$ **do**
      if $\left[x(\omega') \otimes \mu(u^i) \otimes \mu(e)\right]_q = t$ and $\left[\mu(u^i) \otimes \mu(e)\right]_{q'q} \neq \varepsilon$ **then**
        $W(\sigma_o \cdot (e,t)) := W(\sigma_o \cdot (e,t)) \cup \{(q, \omega' u^i e)\}$
      end if
    end for
  end for
end for
return$(W(\sigma_o \cdot (e,t)))$
end function

Proposition 3.1. Assume $W(\epsilon) = \{(q,\epsilon) \mid q \in Q_i\}$. Called for successive pairs in sequence $\sigma_o = (e_1,t_1)(e_2,t_2)\cdots(e_j,t_j) \in P(L(G))$, function $UpdateW$ returns $W(\sigma_o)$.

Proof. Proposition 3.1 can be proved by induction on the length of the observation. $W(\epsilon) = \{(q,\epsilon) \mid q \in Q_i\}$ is the compatible set for the empty weighted sequence. Let $\sigma_o' = (e_1,t_1)(e_2,t_2)\cdots(e_{j-1},t_{j-1})$, and assume that function $UpdateW$ returns $W(\sigma_o')$ when it has been called for successive pairs in σ_o'. When called for (e_j,t_j), function $UpdateW$ first computes the state vectors $\left[x(\omega') \otimes \mu(u^i) \otimes \mu(e_j)\right]_q$ and $\left[\mu(u^i) \otimes \mu(e_j)\right]_{q'q}$, $i = 0,\cdots,|Q|-1$, for each pair $(q',\omega') \in W(\sigma_o')$ and for any $q \in Q$. Then, any pair $(q,\omega' u^i e_j)$ is included in set $W(\sigma_o)$ if $\left[x(\omega') \otimes \mu(u^i) \otimes \mu(e_j)\right]_q = t_j$ and $\left[\mu(u^i) \otimes \mu(e_j)\right]_{q'q} \neq \varepsilon$. Note that the condition $\left[x(\omega') \otimes \mu(u^i) \otimes \mu(e_j)\right]_q = t_j$ ensures that state q can be reached with the weight t_j via string $\omega' u^i e_j$ and the condition $\left[\mu(u^i) \otimes \mu(e_j)\right]_{q'q} \neq \varepsilon$ ensures that there exists a path labeled by $u^i e_j$ from state q' to q. Since (q',ω') is compatible, the above two conditions ensure that $(q,\omega' u^i e_j)$ is compatible with σ_o. Therefore, function $UpdateW$ returns $W(\sigma_o)$ when it has been called for successive pairs in σ_o. □

Function *ConsistentSet* in Algorithm 3.2 is employed to determine the set of states consistent with an observation σ_o and a weight $\tau \geq t_f(\sigma_o)$. Although we assume that no observable label occurs between $t_f(\sigma_o)$ and τ, unobservable labels may occur between them. This requires us to enumerate all the possible unobservable transitions in continuation of compatible pairs in $W(\sigma_o)$.

Proposition 3.2. For any $\sigma_o \in P(L(G))$ whose compatible set is $W(\sigma_o)$ calculated by Algorithm 3.1, and for a weight $\tau \geq t_f(\sigma_o)$, the set $C(\sigma_o,\tau)$ returned by function

Algorithm 3.2: Compute the set of consistent states

Input: $W(\sigma_o)$, $\tau \geq t_f(\sigma_o)$.
Output: $C(\sigma_o, \tau)$.

function *ConsistentSet*$(W(\sigma_o), \tau)$
$C := \emptyset$
for each $(q', \omega') \in W(\sigma_o)$ **do**
  for $i = 0$ to $|Q| - 1$ **do**
    $Q_\tau := \emptyset$
    for each $q \in Q$ **do**
      if $\varepsilon \neq \left[x(\omega') \otimes \mu(u^i)\right]_q \leq \tau$ and $\left[\mu(u^i)\right]_{q'q} \neq \varepsilon$ **then**
        $Q_\tau := Q_\tau \cup \{q\}$
      end if
    end for
    for $q \in Q_\tau$ **do**
      $T_q := \emptyset$
      if $q^\bullet = \emptyset$ **then**
        $C := C \cup \{q\}$
      else
        for each $(q, a, q'') \in q^\bullet$ **do**
          $T_q = T_q \cup \left\{ \left[x(\omega') \otimes \mu(u^i) \otimes \mu(a)\right]_{q''} \right\}$
          if $\exists t \in T_q$ such that $t > \tau$ **then**
            $C := C \cup \{q\}$
          end if
        end for
      end if
    end for
  end for
end for
return$(C(\sigma_o, \tau) = C)$
end function

ConsistentSet is equal to $C(\sigma_o, \tau)$ defined in Eq. (3.1).

Proof. According to Algorithm 3.1, we know that $W(\sigma_o)$ is composed of all pairs (q', ω') such that q' is reached with a weight $t_f(\sigma_o)$ from an initial state by a path π labeled by ω' ending with the last label in σ_o, and $P(\sigma(\pi)) = \sigma_o$. For each pair $(q', \omega') \in W(\sigma_o)$ and for any $q \in Q$, function *ConsistentSet* computes the state vectors $\left[x(\omega') \otimes \mu(u^i)\right]_q$ and $\left[\mu(u^i)\right]_{q'q}$, $i = 0, \cdots, |Q| - 1$. Conditions $\varepsilon \neq \left[x(\omega') \otimes \mu(u^i)\right]_q \leq \tau$ and $\left[\mu(u^i)\right]_{q'q} \neq \varepsilon$ ensure that state q can be reached by a generated weighted sequence σ such that $P(\sigma) = \sigma_o$ and $t_f(\sigma) \leq \tau$. Besides, the existence of transition $(q, a, q'') \in q^\bullet$ such that $\left[x(\omega') \otimes \mu(u^i) \otimes \mu(a)\right]_{q''} > \tau$ guarantees that at least one of the occurrence weights of the output transitions of q is greater than τ. Condition $q^\bullet = \emptyset$ ensures that q has no output transition. All states satisfying the above conditions is included in $C(\sigma_o, \tau)$, which coincides with the definition of $(\sigma_o, \tau)-$consistent states. □

Using functions proposed in Algorithms 3.1 and 3.2, Algorithm 3.3 is introduced to deal with the state estimation problem during the operation of the system.

Algorithm 3.3: Online state estimation

Input: $G = (Q, E, \alpha, \mu, \beta)$.
Output: $C(\sigma_o, \tau)$.

1: $\sigma_o := \epsilon$, $W(\sigma_o) := \emptyset$
2: **for** each $q \in Q_i$ **do**
3: $W(\sigma_o) := W(\sigma_o) \cup \{(q, \epsilon)\}$
4: **end for**
5: **while** state estimation is required **do**
6: **if** observation (e, t) arrives **then**
7: $W(\sigma_o \cdot (e, t)) := UpdateW((e, t), W(\sigma_o))$
8: $\sigma_o := \sigma_o \cdot (e, t)$
9: **end if**
10: **if** the set of consistent states is required **then**
11: Let τ be the current weight
12: $C(\sigma_o, \tau) := ConsistentSet(W(\sigma_o), \tau)$
13: **end if**
14: **end while**

Proposition 3.3. Algorithm 3.3 provides an online solution to Problem 3.1.

Proof. Algorithm 3.3 initializes $W(\epsilon) = \{(q, \epsilon) \mid q \in Q_i\}$ since all initial states can be reached via an empty string (no observation has occurred). Each time a new observable label occurs, function $UpdateW$ is called to update the compatible set, and, when needed, the consistent states is determined by calling function *ConsistentSet*. According to Propositions 3.1 and 3.2, Algorithm 3.3 does provide a solution to Problem 3.1. □

Example 3.2. Let us consider again WA G in Fig. 2.8. Let us assume that the weighted sequence $\sigma_o = (b,5)(b,10)$ is observed and that the set of consistent states is required at a given weight $\tau = 14$. It can be checked that $\alpha = (e,\varepsilon,\varepsilon,e)$, $\beta = (e,\varepsilon,\varepsilon,\varepsilon)$,

$$\mu(u) = \begin{pmatrix} \varepsilon & 1 & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & 2 & \varepsilon \end{pmatrix}, \ \mu(b) = \begin{pmatrix} \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ 4 & 6 & \varepsilon & \varepsilon \\ \varepsilon & 3 & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon \end{pmatrix}.$$

1) Initialize $W(\epsilon) = \{(1,\epsilon),(4,\epsilon)\}$.

2) Update compatible set when observation $(b,5)$ arrives:

Firstly, compute $\mu(u^i)$, $i = 0,1,2,3$:

$$\mu(u^0) = \begin{pmatrix} e & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & e & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & e & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & e \end{pmatrix}, \mu(u^2) = \mu(u^3) = \begin{pmatrix} \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon \end{pmatrix}.$$

Secondly, for pair $(1,\epsilon)$, compute $x(\epsilon) \otimes \mu(u^i) \otimes \mu(b)$ and $\left[\mu(u^i) \otimes \mu(b)\right]_{1,\bullet}$ (the row corresponding to state 1 in matrix $\mu(u^i) \otimes \mu(b)$), $i = 0,1,2,3$: $x(\epsilon) \otimes \mu(u) \otimes \mu(b) = (5,7,\varepsilon,\varepsilon)$; $x(\epsilon) \otimes \mu(u^i) \otimes \mu(b) = (\varepsilon,\varepsilon,\varepsilon,\varepsilon)$, for $i = 0,2,3$; $[\mu(u) \otimes \mu(b)]_{1,\bullet} = (5,7,\varepsilon,\varepsilon)$; $\left[\mu(u^i) \otimes \mu(b)\right]_{1,\bullet} = (\varepsilon,\varepsilon,\varepsilon,\varepsilon)$, for $i = 0,2,3$. According to function $UpdateW$, any pair (q,u^ib) is recorded as an element of the compatible set if $\left[x(\epsilon) \otimes \mu(u^i) \otimes \mu(b)\right]_q = 5$ and $\left[\mu(u^i) \otimes \mu(b)\right]_{1,q} \neq \varepsilon$. In this case, the only pair $(1,ub)$ is obtained as an element of the updated set W since $[x(\epsilon) \otimes \mu(u) \otimes \mu(b)]_1 = 5$ and $[\mu(u) \otimes \mu(b)]_{1,1} = 5 \neq \varepsilon$.

Thirdly, for pair $(4,\epsilon)$, compute $x(\epsilon) \otimes \mu(u^i) \otimes \mu(b)$ and $\left[\mu(u^i) \otimes \mu(b)\right]_{4,\bullet}$, $i = 0,1,2,3$: $x(\epsilon) \otimes \mu(u) \otimes \mu(b) = (5,7,\varepsilon,\varepsilon)$; $x(\epsilon) \otimes \mu(u^i) \otimes \mu(b) = (\varepsilon,\varepsilon,\varepsilon,\varepsilon)$, for $i = 0,2,3$; $[\mu(u) \otimes \mu(b)]_{4,\bullet} = (\varepsilon,5,\varepsilon,\varepsilon)$; $\left[\mu(u^i) \otimes \mu(b)\right]_{4,\bullet} = (\varepsilon,\varepsilon,\varepsilon,\varepsilon)$, for $i = 0,2,3$.

Similar to the above process, any pair (q,u^ib) is added into the updated set W if $\left[x(\epsilon) \otimes \mu(u^i) \otimes \mu(b)\right]_q = 5$ and $\left[\mu(u^i) \otimes \mu(b)\right]_{4,q} \neq \varepsilon$. In this case, no pair is recorded. Hence, function $UpdateW$ returns $W((b,5)) = \{(1,ub)\}$.

3) Update compatible set when observation $(b,10)$ arrives: Similar to step 2), any pair (q,ubu^ib) is recorded as an element of the new compatible set if $\left[x(ub) \otimes \mu(u^i) \otimes \mu(b)\right]_q = t_2 = 10$ and $\left[\mu(u^i) \otimes \mu(b)\right]_{1,q} \neq \varepsilon$. In this case, the unique pair $(1,ubub)$ is recorded, and function $UpdateW$ returns $W((b,5)(b,10)) = \{(1,ubub)\}$.

4) Compute set of consistent states required at $\tau = 14$: For pair $(1,ubub)$, compute

$x(ubub) \otimes \mu(u^i)$ and $\left[\mu(u^i)\right]_{1,\bullet}$, $i = 0,1,2,3$: $x(ubub) \otimes \mu(u^0) = (10,12,\varepsilon,\varepsilon)$; $x(ubub) \otimes \mu(u) = (\varepsilon,11,\varepsilon,\varepsilon)$; $x(ubub) \otimes \mu(u^i) = (\varepsilon,\varepsilon,\varepsilon,\varepsilon)$, for $i = 2,3$; $\left[\mu(u^0)\right]_{1,\bullet} = (e,\varepsilon,\varepsilon,\varepsilon)$; $[\mu(u)]_{1,\bullet} = (\varepsilon,1,\ \varepsilon,\varepsilon)$; $\left[\mu(u^i)\right]_{1,\bullet} = (\varepsilon,\varepsilon,\varepsilon,\varepsilon)$, for $i = 2,3$.

According to Algorithm 3.2, we get $Q_\tau = \{1,2\}$. State 1 is not consistent with $\sigma_o = (b,5)(b,10)$ and $\tau = 14$ since it has only one output transition labeled by u leading to state 2, and $x(ababa)_2 = 11 < \tau$. State 2 is consistent with σ_o and τ because one of its output transitions occur with a weight $x(ababab)_2 = 17$ greater than τ. Therefore $C(\sigma_o,\tau) = \{2\}$. □

3.3.2 Computational Complexity Analysis

Complexity of Our Online State Estimation Approach

In this subsection, we discuss the computational complexity of estimating all possible states consistent with an observation $\sigma_o = (e_1,t_1)(e_2,t_2)\cdots(e_k,t_k) \in P(L(G))$ and a given value $\tau \geq t_f(\sigma_o) = t_k$ using Algorithm 3.3.

Function $UpdateW$ is used to update the compatible set each time a label is observed. Let $\sigma_o' = (e_1,t_1)(e_2,t_2)\cdots(e_{j-1},t_{j-1})$ and $\sigma_o'' = (e_1,t_1)(e_2,t_2)\cdots(e_j,t_j)$ with $1 \leq j \leq k$. In order to calculate $W(\sigma_o'')$ from $W(\sigma_o')$ when observation (e_j,t_j) arrives, we have to check whether the pair $(q,\omega' u^i e_j)$, $i = 0,\cdots,|Q|-1$, satisfies $\left[x(\omega') \otimes \mu(u^i) \otimes \mu(e_j)\right]_q = t_j$ and $\left[\mu(u^i) \otimes \mu(e_j)\right]_{q'q} \neq \varepsilon$ for each pair (q',ω') belonging to $W(\sigma_o')$. Therefore, computing $W(\sigma_o'')$ requires $2 \times |Q|^2 \times |W(\sigma_o')|$ comparisons where $|W(\sigma_o'')|$ is the cardinality of $W(\sigma_o')$. In the worst case, it is equal to $2 \times |Q|^{j+2}$ since $|W(\sigma_o')| \leq |Q|^j$. There are k observed labels in σ_o, which means that function $UpdateW$ should be called k times. Thus, the total number of comparisons for calculating $W(\sigma_o)$ is at most $\sum_{j=1}^{k} 2 \times |Q|^{j+2}$.

The function $ConsistentSet$ is called to calculate all states consistent with observation σ_o and τ. Any state $q \in Q$ is a consistent state if there exists a pair $(q',\omega') \in W(\sigma_o)$ and an integer number $l \in [0,|Q|-1]$ such that $\varepsilon \neq \left[x(\omega') \otimes \mu(u^l)\right]_q \leq \tau$ and $\left[\mu(u^l)\right]_{q'q} \neq \varepsilon$. Besides, the occurrence weights of output transitions of q need to be compared with τ. In the worst case, the number of comparisons required to check this for all states in the worst case is $(3+|E|) \times |Q|^{k+3}$.

Therefore, in the worst case, the total complexity of solving the problem of state estimation for a fixed-length observation $\sigma_o = (e_1,t_1)(e_2,t_2)\cdots(e_k,t_k)$ and a given value $\tau \geq t_k$ using Algorithm 3.3 is $\mathcal{O}(\sum_{j=1}^{k} 2 \times |Q|^{j+2} + (3+|E|) \times |Q|^{k+3})$.

Remark 3.2. The computational complexity, in terms of the number of comparisons, of

our online algorithm grows exponentially with the length of the observation. This is because each time a new label is observed, the compatible set should be updated from the previous one. For an observation $\sigma_o = (e_1, t_1)(e_2, t_2)\cdots(e_k, t_k)$, all possible sequences of unobservable transitions interleaved with it should be taken into account. That is, at worst, we have to consider all sequences $u^{i_1}e_1u^{i_2}e_2\cdots u^{i_k}e_k$ with $i_1, \cdots, i_k = 0, 1, \cdots |Q| - 1$ to determine $W(\sigma_o)$. This implies that in the worst case $|W(\sigma_o)| = |Q|^{k+1}$. □

Complexity of Some approaches for State Estimation or Related Problems

WAs automata, such as max-plus automata, can be considered as a class of timed DES models if time interpretation is associated to the weights of transitions. The complexity of our state estimation approach is due to the fact that we are considering a timed model. After a careful analysis of the literature, we can make the following comments on the complexity of some existing approaches for state estimation or related problems.

(1) In the case of automata, there exist online algorithms that are polynomial in the number of states of the system for detectability [Shu and Lin, 2011] and for online fault diagnosis [Jiang *et al.*, 2001; Yoo and Lafortune, 2002].

More precisely, in [Shu and Lin, 2011], an approach whose complexity is polynomial in the number of states $|Q|$ of an automaton G is presented to check (strong) detectability of G. A new automaton called detector G_{det}, the cardinality of its state space is bounded by $1 + |Q| + |Q|(|Q| - 1)/2$, is first constructed. Then necessary and sufficient condition is proposed to check the ability to determine the current and subsequent states of G. Clearly, once G_{det} is built, it can be used to perform online state estimation with a polynomial complexity in the length of observation and in the number of states of G. Yoo and Lafortune [Yoo and Lafortune, 2002] check diagnosability by constructing a verifier and Jiang et al. [Jiang *et al.*, 2001] check diagnosability by constructing a new automaton G_d based on the concurrent composition of two plant models. These algorithms are polynomial in the number of states of the system and also in the number of fault classes. Online fault diagnosis can be performed on verifier and G_d with a polynomial complexity.

(2) For the diagnosis of place/transition nets, online approaches are typically based on integer linear programming (ILP) techniques which are known to be NP-complete, but they do not need to enumerate the reachability space.

For instance, in [Basile *et al.*, 2009; Dotoli *et al.*, 2009], the authors present an online algorithm solving at each observed event some ILP problems in order to determine the occurrence of different faults. Since, no polynomial algorithm for ILP is known to exist, so

the online computational effort of these proposed algorithms in these work can be high (may not be polynomial). They also prove that the computational complexity can be reduced to be polynomial if the unobservable subnet of the studied Petri nets enjoys suitable properties (for instance, the subnet is an acyclic state machine and the studied Petri nets system is bounded). In Cabasino et al. [Cabasino *et al.*, 2010], the basis reachability graph is offline constructed to reduce the complexity of online fault diagnosis.

(3) When timed models are considered, the complexity of online approaches for state estimation and fault diagnosis (usually) becomes exponential.

For instance, Basile et al. [Basile *et al.*, 2015] investigate state estimation for time labeled Petri nets based the construction of modified state class graph (MSCG). The complexity comes from two folds: 1) the number of nodes of the MSCG increases exponentially with the system complexity (net structure, and number of tokens in the initial marking). 2) Based on the MSCG, a process consists of two main steps is introduced to determine the set of consistent states for a given observed sequence of events σ_o and a given time instant τ. More precisely, during the first step, the author compute the set of paths in the MSCG that are logically consistent with the observation σ_o. The complexity of finding all such paths is $\mathcal{O}(|T|^{l_{max}})$ where l_{max} is the length of the longest path in the MSCG that is logically consistent with the observation σ_o and the given value of τ. During the second step, an integer linear programming problems with $1 + |T| + 3 \times l_{max} \times |T| + l_{max}$ constraints is formulated to establish if the above paths are timing consistent with all the labels in σ_o. It is clear that l_{max} increases as the length of the observation σ_o increases. Hence the total complexity of this approach is exponential in the length of the observation σ_o.

Tripakis [Tripakis, 2002] and Bouyer et al. [Bouyer *et al.*, 2005] investigate the diagnosability for timed automata. The algorithm proposed in [Tripakis, 2002] is based on state estimation in a timed automaton with ε-transitions (unobservable transitions), its complexity to diagnose faults for an observation is doubly exponential in the size of the plant and in the size of the observation. An algorithm based on regions (and no more on zones) with a complexity exponential in both the size of the plant and of the observation could be proposed as well. Bouyer et al. [Bouyer *et al.*, 2005] deals with the diagnosability for deterministic timed automata (DTA) and event-recording timed automata (ERA). The authors prove that the checking of existence of diagnoser in the class DTA (with bounded resources) is 2EXPTIME-complete whereas it is PSPACE-complete for the class ERA (with bounded resources). Moreover if a diagnoser exists, it is possible to construct one: the size of such a diagnoser is doubly exponential in the granularity and in the size of the plant, but

only linear in the length of the observation.

3.3.3 Some Attempts of Constructing Observer for State Estimation

As discussed in subsection 3.3.2, our proposed approach has an exponential complexity in the length of the observation, which may be an issue during the implementation phase. However, there may be approaches (similar to the construction of the observer [Sampath *et al.*, 1995; Shu and Lin, 2011] or similar structure [Cabasino *et al.*, 2010; Basile *et al.*, 2015]) that would move the complexity to the offline phase. But the complexity of this may be even worse than exponential. Once this is done, the complexity of the online phase would be minimal. In fact, we tried but did not succeed in building such structures that can minimize the complexity of the online estimation process. Let us sketch some of the attempts and explain informally why they are not conclusive.

(1) Our first idea was to build a WA G_{obs} (possibly nondeterministic) playing the role of "observer" in the sense that its alphabet is restricted to observable labels and that it can generate the same observation. A straightforward construction could be to define the transition matrices by

$$\mu_{obs}(a) = \bigoplus_{i=1,\cdots,|Q|-1} \Big(\mu(u^i) \otimes \mu(a)\Big), \text{ for all } a \in E_o$$

If we consider the following simple WA G in Figure 3.1 where u is unobservable and b is observable. Assume G is defined over the max-plus semiring. Then, we have

$$\mu(u) = \begin{pmatrix} \varepsilon & 1 & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon \end{pmatrix}, \ \mu(b) = \begin{pmatrix} \varepsilon & \varepsilon & 1 \\ \varepsilon & \varepsilon & 1 \\ \varepsilon & 2 & \varepsilon \end{pmatrix},$$

$$\mu_{obs}(b) = \mu(b) \oplus (\mu(u) \otimes \mu(b)) = \begin{pmatrix} \varepsilon & \varepsilon & 2 \\ \varepsilon & \varepsilon & 1 \\ \varepsilon & 2 & \varepsilon \end{pmatrix}.$$

The resulting "observer" G_{obs} is depicted in Figure 3.2. G_{obs} is not satisfactory since it cannot generate, for instance, the observation $(b,1)(b,3)$ which can be generated by G.

(2) Another possibility could be to define a specific alphabet for G_{obs} composed of symbols corresponding to strings $u^i a$ for each $a \in E_o$ and $i = 0, 1, \cdots, |Q| - 1$ (remember that there is no circuit labeled only by unobservable labels). Transition matrices for G_{obs} can

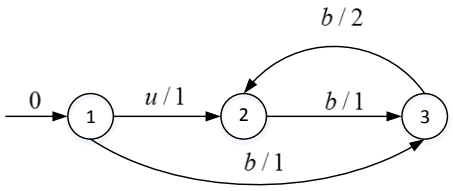

Figure 3.1 – Weighted automaton G

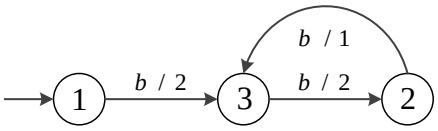

Figure 3.2 – G_{obs} with $\mu_{obs}(b) = \bigoplus_{i=1} \left(\mu(u^i) \otimes \mu(b)\right)$, *for* $i = 0, 1$

be defined by

$$\mu_{obs}(u^i a) = \mu(u^i a), \text{ for } a \in E_o$$

The idea behind is that letters appearing in possible observation from G are the projections of letters into the alphabet of G_{obs}. In this case, for example, the "observer" G_{obs} of the WA G in Fig. 3.1 is depicted in Figure 3.3.

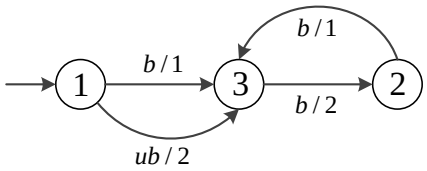

Figure 3.3 – G_{obs} with $\mu_{obs}(u^i b) = \mu(u^i b)$, for $i = 0, 1$

Then, based on G_{obs}, to check if state q_k is consistent with a given observation $(a_1, t_1)(a_2, t_2) \cdots (a_k, t_k)$ we not only need to verify that there exist $i_1, i_2, \cdots, i_k \in \{0, 1, \cdots, |Q| - 1\}$ such that $x_{obs}((u^{i_1}a_1)(u^{i_2}a_2) \cdots (u^{i_k}a_k))_{q_k} = t_k$ but also verify that

$$\exists q_{k-1} \text{ s.t. } x_{obs}((u^{i_1}a_1)(u^{i_2}a_2) \cdots (u^{i_{k-1}}a_{k-1}))_{q_{k-1}} = t_{k-1} \text{ and } \mu_{obs}(u^{i_k}a_k)_{q_{k-1}q_k} \neq \varepsilon$$

$$\vdots$$

$$\exists q_1 \text{ s.t. } x_{obs}(u^{i_1}a_1)_{q_1} = t_1 \text{ and } \mu_{obs}(u^{i_1}a_1)_{q_0 q_1} \neq \varepsilon \text{ with } q_0 \in Q_i$$

Finally, this leads to the same enumerations as in our algorithms, and complexity is not enhanced.

(3) We also tried to iteratively build "observer" G_{obs} whose states would reflect the

consistent states in the original WA G. In other words, G_{obs} now is a finite automaton, and each of its states q^{obs} is a subset of states of G, i.e., $q^{obs} \subseteq Q$. G_{obs} can be built as follows:

- the initial state of G_{obs} is defined to be the set of initial states of G;
- for any state q^{obs} of G_{obs}, a state transition is defined for each possible "observed transition".

For example, consider the WA G in Fig. 2.8, its "observer" G_{obs} obtained according to the above principle is depicted in Fig. 3.4.

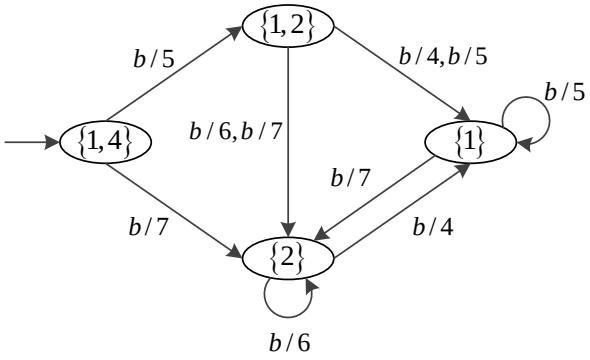

Figure 3.4 – G_{obs} of weighted automaton G in Fig. 2.8

One of the problems with such construction is that the resulting "observer" may generate trajectories that are not possible observations in the original automaton. In this example, $(b, 5)(b, 11)$ is generated by G_{obs}, but it is impossible in G.

We have investigated a lot of different ways for building an "observer" through a WA (or even a finite automaton with tailored labellings), but so far, we have not succeeded. Nevertheless, the feasibility of constructing an observer or similar structures for state estimation of a WA can be considered as a future work.

3.4 Fault Diagnosis of Weighted Automata

In the framework of logical DES, Zad et al. [Zad *et al.*, 2003] pointed out that the fault diagnosis problem in event-based framework can always be transformed to an equivalent problem in a state-based framework. In other words, fault diagnosis can be reduced to a state estimation problem. Inspired by this, in this section, we prove that the event-based fault diagnosis of a WA is also equivalent to solving state estimation problem. Given a WA, we first propose an algorithm to construct its augmented version, called augmented automaton, respect to a fault class. Then, the diagnosis state with respect to a fault class

is associated to each observation and a weight value using the state estimation approach presented in subsection 3.3.1.

3.4.1 Construction of the Augmented Automaton

Suppose that we want to determine the occurrence of some labels belonging to fault class E_f. We transform the original WA $G = (Q, E, \alpha, \mu, \beta)$ into $G^* = \left(Q^*, E, t^*, Q_i^*, Q_m^*, \varrho^*, \rho^*\right)$, where $Q^* \subseteq Q \times \{0,1\}$ is the set of states, $t^* : Q^* \times E \times Q^* \to \mathbb{R}_{max}$ is the transition function, $Q_i^* \subseteq Q^*$ (resp. $Q_m^* \subseteq Q^*$) is the set of initial states (resp. final states), $\varrho^* : Q_i^* \to \mathbb{R}_{max}$ (resp. $\rho^* : Q_m^* \to \mathbb{R}_{max}$) is the function of initial weights (resp. final weights).

Note that, each element $q^* \in Q^*$ is a pair (q, f) where q is a state symbol of G, and f is a boolean variable. Variable f is used to mark the state status of G^*. More precisely, (q, f) is a normal (resp. fault) state when f is equal to 0 (resp. 1). In other words, Q^* is partitioned into two parts, i.e., $Q^* = Q_N^* \cup Q_f^*$ where Q_N^* and Q_f^* are the set of normal and faulty states, respectively. G^* is called the augmented automaton of G with respect to fault class E_f. In the worst case, the number of states of G^* is $|Q^*| = 2 \times |Q|$.

For automaton $G = (Q, E, \alpha, \mu, \beta)$, we denote $R(q, a)$ the set of reachable states for $a \in E$ from state q, i.e., $R(q, a) = \left\{q' \in Q \mid \mu(a)_{qq'} \neq \varepsilon\right\}$. Now we define the set of reachable states for $a \in E$ from state (q, f) in G^* as: $R^*((q, f), a) = \{(q', f) \mid q' \in R(q, a)\}$, if $a \notin E_f$, and $R^*((q, f), a) = \{(q', 1) \mid q' \in R(q, a)\}$, if $a \in E_f$.

Algorithm 3.4 is used to construct the augmented automaton of G with respect to fault class E_f.

Steps 2-5 of Algorithm 3.4 enables us to compute the set of initial states and to specify the function of initial weights for G^*. In its graphical representation, these states are represented by nodes. The initial states of G are initial states of G^* and have a value 0 defining that no fault label has occurred at the beginning of the system evolution. We use stack *Queue* to store these nodes. Now, for each element (q, f) in *Queue*, we calculate all states that can be reached by labels $a \in E$, i.e., $R^*((q, f), a)$. If a pair $(q', f') \in R^*((q, f), a)$ is not contained in the previous nodes, a new node is added to G^*. The arc going from (q, f) node to (q', f') node is labeled by $a/\mu(a)_{qq'}$. At step 18, (q', f') is added into stack *Queue*. The procedure is iterated until stack *Queue* becomes empty. Steps 24-27 of Algorithm 3.4 determine the set of final states and the function of final weights.

Example 3.3. Consider the WA G in Fig. 3.5. Assume that $E_o = \{a, b, c\}$, $E_{uo} = E_f \cup E_{reg}$ with $E_f = \{\epsilon_f\}$ and $E_{ref} = \{\epsilon_n\}$. After applying Algorithm 3.4, the augmented automaton G^* is shown in Fig. 3.6 where the fault states are highlighted in red. □

Algorithm 3.4: Construction of the augmented automaton

Input: A WA $G = (Q, E, \alpha, \mu, \beta)$ and the fault class E_f.
Output: An augmented automaton G^* of G.

function $TransformG(G, E_f)$
Let $Q_i^* = \emptyset$
for each $q \in Q_i$ **do**
  $Q_i^* = Q_i^* \cup \{(q, 0)\}$, $\varrho^*((q, 0)) = \alpha_q$
end for
Let $Q^* = Q_i^*$, $Queue = Q_i^*$
while $Queue \neq \emptyset$ **do**
  Dequeue (q, f) from $Queue$
  for each $a \in E$ **do**
    if $a \in E_f$ **then**
      $R^*((q, f), a) = \{(q', 1) \mid q' \in R(q, a)\}$
    else
      $R^*((q, f), a) = \{(q', f) \mid q' \in R(q, a)\}$
    end if
    for each $(q', f') \in R^*((q, f), a)$ **do**
      if $(q', f') \notin Q^*$ **then**
        $Q^* = Q^* \cup \{(q', f')\}$
        Enqueue (q', f') in $Queue$
      end if
      $t^*((q, f), a, (q', f')) = \mu(a)_{qq'}$
    end for
  end for
end while
Let $Q_m^* = \emptyset$
for each $(q, f) \in Q^*$ such that $\beta_q \neq \varepsilon$ **do**
  $Q_m^* = Q_m^* \cup \{(q, f)\}$, $\rho^*((q, f)) = \beta_q$
end for
Let $G^* = \left(Q^*, E, t^*, Q_i^*, Q_m^*, \varrho^*, \rho^*\right)$
return(G^*)
end function

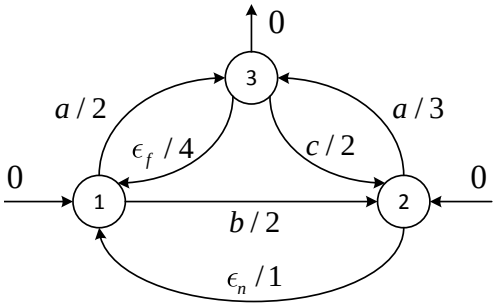

Figure 3.5 – A weighted automaton G

Lemma 3.1. The generated weighted language of G is equal to the generated weighted language of G^*, i.e., $L(G) = L(G^*)$.

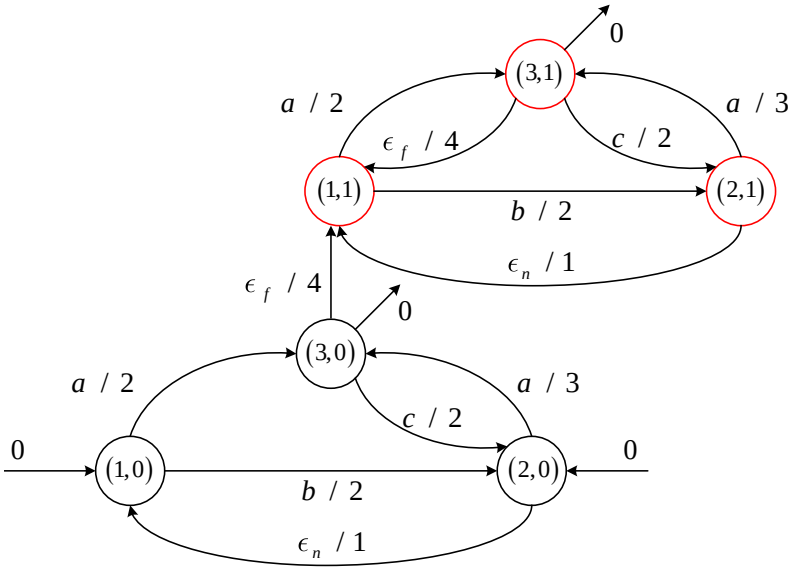

Figure 3.6 – The augmented automaton G^* of G in Fig. 3.5

Proof. We need to prove for any weightedd sequence σ, $\sigma \in L(G) \Leftrightarrow \sigma \in L(G^*)$.

(Only If) For any $\sigma = (e_1, \tau_1)(e_2, \tau_2) \cdots (e_k, \tau_k) \in L(G)$, by the definition of generated weighted language of G, there must be at least one path from one initial state, and the weighted sequence it generates is equal to σ. Assume $\pi = (q_0, e_1, q_1)(q_1, e_2, q_2) \cdots (q_{k-1}, e_k, q_k)$ is such a path. By Algorithm 3.4, there must exist the following path in G^*:

$$\pi' = ((q_0, 0), e_1, (q_1, f_1))((q_1, f_1), e_2, (q_2, f_2)) \cdots ((q_{k-1}, f_{k-1}), e_k, (q_k, f_k))$$

where $(q_0, 0) \in Q_i^*$, and for $i = 1, \cdots, k$, $f_i = 1$ if $e_i \in E_f$, otherwise, $f_i = f_{i-1}$. Besides, the weights of transitions of π' are $t^*((q_{i-1}, f_{i-1}), e_i, (q_i, f_i)) = \mu(e_i)_{q_{i-1}q_i}$, for $i = 1, \cdots k$. Now we assume that the weighted sequence generated by π' is different from σ, i.e., $\sigma(\pi') \neq (e_1, \tau_1)(e_2, \tau_2) \cdots (e_k, \tau_k)$. This implies that there exists at least one $i \in \{1, \cdots, k\}$, such that $x(e_1 \cdots e_i)_{(q_i, f_i)} \neq \tau_i$, which contradicts $x(e_1 \cdots e_i)_{q_i} = \tau_i$. Therefore, σ is generated by π', i.e., $\sigma \in L(G^*)$. Hence, $\forall \sigma \in L(G) \Rightarrow \sigma \in L(G^*)$.

(If) For any $\sigma = (e_1, \tau_1)(e_2, \tau_2) \cdots (e_k, \tau_k) \in L(G^*)$, there must be at least one path from one initial state in G^*, and its associated weighted sequence is equal to σ. Assume the following is such a path:

$$\pi = ((q_0, 0), e_1, (q_1, f_1))((q_1, f_1), e_2, (q_2, f_2)) \cdots ((q_{k-1}, f_{k-1}), e_k, (q_k, f_k))$$

where $f_i \in \{0, 1\}$, $i = 1, \cdots, k$. By Algorithm 3.4, there must exist the path $\pi' =$

$(q_0, e_1, q_1)(q_1, e_2, q_2) \cdots (q_{k-1}, e_k, q_k)$ in G where $q_0 \in Q_i$. Besides, the weights of transitions of π' are $\mu(e_i)_{q_{i-1}q_i} = t^*((q_{i-1}, f_{i-1}), e_i, (q_i, f_i))$, for $i = 1, \cdots k$. Similar to the prove process of (Only If), we know that σ is generated by π', i.e., $\sigma \in L(G)$. Hence, $\forall \sigma \in L(G^*) \Rightarrow \sigma \in L(G)$. □

Intuitively, according to Algorithm 3.4, G^* does not change the state transitions of G because the reachable states of any state (q, f) in G^* is defined by the reachable states of q in G, i.e., $R^*((q, f), a)$, $a \in E$, is calculated based on $R(q, a)$. Besides, the transition weights in G^* are equal to the corresponding weights in G since $t^*((q, f), a, (q', f')) = \mu(a)_{qq'}$. Finally, G and G^* have the same initial states with identical weights, and the same final weights.

3.4.2 Diagnosis State Determination

This subsection shows that solving the diagnosis problem for G is equivalent to solving the state estimation problem for its augmented automaton G^*.

Given an observed weighted sequence $\sigma_o \in P(L(G))$, obtained from WA G, and a weight $\tau \geq t_f(\sigma_o)$, the set of all consistent states $C(\sigma_o, \tau)$ in G is defined in Eq. (3.1). In this section, we denote $C^*(\sigma_o, \tau)$ the set of states consistent with σ_o and τ in augmented WA G^* of G with respect to fault class E_f.

Lemma 3.2. Consider an observed weighted sequence $\sigma_o \in P(L(G))$, a given weight τ, and fault class E_f.

- $\varphi(\sigma_o, \tau, E_f) = N$ if $C^*(\sigma_o, \tau) \subseteq Q_N^*$;
- $\varphi(\sigma_o, \tau, E_f) = F$ if $C^*(\sigma_o, \tau) \subseteq Q_f^*$;
- $\varphi(\sigma_o, \tau, E_f) = U$ if $C^*(\sigma_o, \tau) \cap Q_N^* \neq \emptyset$ and $C^*(\sigma_o, \tau) \cap Q_f^* \neq \emptyset$.

Proof. Let $S^*(\sigma_o, \tau)$ denotes the set of sequences, generated by G^*, that are consistent with σ_o and τ. By Lemma 3.1, we have $S^*(\sigma_o, \tau) = S(\sigma_o, \tau)$. From Definition 3.2 and Definition 3.1, we know that for each consistent state $q^* = (q, f) \in Q^*$, there must be a corresponding consistent sequence by which it is reached. Assume $\sigma = (e_1, \tau_1)(e_2, \tau_2) \cdots (e_k, \tau_k) \in L(G^*)$ is such a sequence. According to Algorithm 3.4, we know that q^* belongs to Q_N^* iff there exists at least one $i \in \{1, 2, \cdots k\}$ such that $e_i \in E_f$. Therefore, according to Definition 3.3 of diagnosis states, Lemma 3.2 is established. □

In fact, all states of the set $C^*(\sigma_o, \tau)$ are outside (resp. inside) the set Q_f^*, which definitely means that fault class E_f cannot (resp. must) have occurred. On the other hand, we cannot conclude whether a label in E_f has occurred or not if $C^*(\sigma_o, \tau)$ contains normal and faulty

states. Lemma 3.2 shows that the problem of detecting the occurrence of a fault label belonging to E_f in G reduces to a state estimation problem for the WA G^*.

Algorithm 3.5 summarizes the online procedure for solving fault diagnosis problem.

Algorithm 3.5: On-line fault diagnosis

1: $G^* = TransformG(G, E_f)$ /* Add comments in the way */
2: Let $\sigma_o = \epsilon$
3: **while** fault diagnosis is required **do**
4: **if** observation (e, t) arrives **then**
5: Let $\sigma_o = \sigma_o \cdot (e, t)$
6: **end if**
7: **if** the diagnosis state is required **then**
8: Let τ be the current given weight
9: Calculate $C^*(\sigma_o, \tau)$ in G^* using state estimation algorithm in Section 3.3
10: **if** $C^*(\sigma_o, \tau) \subseteq Q_N^*$ **then**
11: $\varphi(\sigma_o, \tau, E_f) = N$
12: **else if** $C^*(\sigma_o, \tau) \subseteq Q_f^*$ **then**
13: $\varphi(\sigma_o, \tau, E_f) = F$
14: **else**
15: $\varphi(\sigma_o, \tau, E_f) = U$
16: **end if**
17: **end if**
18: **end while**

Step 1 of Algorithm 3.5 calls Function $TransformG$ to transform G into G^* with respect to fault class E_f. Initially, the observation is ϵ (nothing has been observed at the beginning of system's evolution). Each time a new observable label occurs, the observation is updated as the concatenation of previous observation and the new occurred label (with its occurrence weight), and, when needed, the diagnosis state is determined based on the consistent states obtained by calling the state estimation algorithm.

Remark 3.3. As pointed out in Remark 3.1, the algorithm proposed in Subsection 3.3.1 for state estimation uses the single symbol u for all the unobservable labels. It is then required to replace all the labels corresponding to unobservable labels in the augmented automaton G^* by u to be able to call the state estimation algorithm at step 9.

Example 3.4. Consider again the WA G in Fig. 3.5, where $E_o = \{a, b, c\}$, $E_{uo} = E_f \cup E_{reg}$ with $E_f = \{\epsilon_f\}$ and $E_{ref} = \{\epsilon_n\}$. Let observation $\sigma_o = (a, 3)(c, 5)(a, 8)$ and $\tau = 12$. We want to determine the occurrence of fault class E_f, i.e., determine the diagnosis state value $\varphi(\sigma_o, \tau, E_f)$. First, we have to calculate the augmented automaton G^* of G with respect to E_f, which is presented in Fig. 3.6. Then, calculate all states that consistent with $\sigma_o = (a, 3)(c, 5)(a, 8)$ and $\tau = 12$ in G^*. Applying the approach proposed in

Subsection 3.3.1, we get $C(\sigma_o, \tau) = \{(1,1)\}$. Since $C^*(\sigma_o, \tau)$ contains no normal state, then the diagnosis state $\varphi(\sigma_o, \tau, E_f) = F$. In fact, from G we know that $S(\sigma_o, \tau) = \{(a,3)(c,5)(a,8)(\epsilon_f,12), (a,3)(c,5)(\epsilon_n,6)(a,8)(\epsilon_f,12)\}$, which means that fault label ϵ_f must have occurred if $\sigma_o = (a,3)(c,5)(a,8)$ is observed at $\tau = 12$. □

Example 3.5. Consider again the WA G in Fig. 2.8, where $E_o = \{b\}$ and $E_f = E_{uo} = \{a\}$. Its augmented automaton G^* calculated by applying Algorithm 3.4 is shown in Fig. 3.7 in which the faulty states are highlighted in red. Let observation $\sigma_o = (b,5)(b,10)$ and $\tau = 14$. Then, we obtain the set of consistent states $C(\sigma_o, \tau) = \{(2,1)\}$ by applying the approach proposed in Subsection 3.3.1. Since $C(\sigma_o, \tau)$ contains no normal state, then the diagnosis state $\varphi(\sigma_o, \tau, E_f) = F$. In fact, from G we know that $S(\sigma_o, \tau) = \{ababa\}$, which means that fault label a must have occurred if $\sigma_o = (b,5)(b,10)$ is observed at $\tau = 14$.

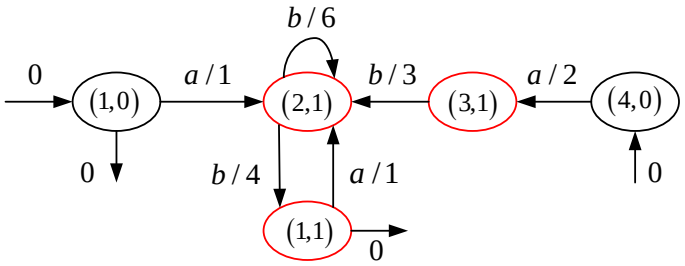

Figure 3.7 – The augmented automaton G^* of G in Fig. 2.8

3.5 Conclusion

This chapter copes with the problems of state estimation and fault diagnosis in the framework of WAs. Once a weighted sequence is observed and a weight value is given, the state estimation problem consists in determining the states in which the system may be, and the fault diagnosis is to determine if a fault label has occurred or not. One one hand, an algorithm based on the analysis of state vector is proposed to deal with state estimation online. One the other hand, the state estimation approach is extended to cope with fault diagnosis problem. For a given WA, we first proposed an algorithm that transforms it into its augmented version with respect to a fault class. Then we prove that the diagnosis problem can be reduced to the problem of state estimation. Finally, an algorithm is presented to summarize the online procedure for detecting the occurrence of some labels that belong to a given fault class.

The work in this chapter has been published as follows:

A. Lai, S. Lahaye and A. Giua. "State estimation of max-plus automata with unobservable events", In *Automatica*, Vol. 105, pp. 36–42, 2019 [Lai *et al.*, 2019a].

A. Lai, S. Lahaye and A. Giua. "A two-step approach for fault diagnosis of max-plus automata", In *proceeding of the 6th International Conference on Control, Decision and Information Technologies* (CoDIT), 2019 [Lai *et al.*, 2019b].

4

Current-State Detectability Verification for UWAs

This chapter studies the current-state detectability verification for unambiguous weighted automata (UWAs).

Contents

4.1 Introduction

chapter 3 deals with the current-state estimation problem, which aims at accurately characterizing the set of possible current states of the system based on the partial observation of its evolution. In this chapter, we deal with the detectability problem for UWAs. The problem is to determine if, after a finite number of observations, the current state can be uniquely determined. That is, the set of possible current states is reduced to a singleton. Four types of detectabilities, namely, strong detectability, weak detectability, strong periodic detectability, and weak periodic detectability are defined in terms of different requirements for current state estimation. On one hand, given a UWA G, we propose an approach based on the construction of observer G_{obs} of which the number of states is exponential with respect to the number of states of G to verify the four detectabilities. On the other hand, we introduce another approach based on the notion of detector G_{det} of which the number of states is polynomial with respect to the state space cardinality of G to verify the strong detectabilities, i.e., strong detectability and strong periodic detectability.

The structure of this chapter is as follows. Section 4.2 formally defines the problem of detectability and gives the definitions of four types of detectabilities. In Section 4.3, given a UWA G, we detail the process of constructing the observer, i.e., current-state estimator, of G. Note that such an observer is a DFA over a weighted alphabet. In Section 4.3.2, necessary and sufficient conditions based on constructed observer are introduced for checking the detectabilities of a UWA. Section 4.4 details the construction process of the detector of a UWA and then proposes a Lemma to illustrate its relationship with the observer built in Section 4.3. In Section 4.5, we present the necessary and sufficient conditions based on the detector for verifying strong detectabilities of UWAs. Section 4.6 summarizes the conclusion of the work in this chapter.

4.2 Problem Formulation

In this chapter we restrict our attention to a WA G with identity initial weights[1]. The label set E is partitioned into two disjoint parts: the observable part E_o and the unobservable part E_{uo}. Considering that the firing of unobservable labels cannot be detected by an external agent, we assume that all unobservable labels are represented by symbol u, i.e., $E = E_o \cup$

1. The coefficients in α different from ε are all equal to e. This assumption is without loss of generality since an automaton with non-identity initial weights can always be transformed into an equivalent automaton with identity initial weights by adding new states and by considering these weights as state transitions durations associated to new fictive initial labels.

$\{u\}$. We also make the following assumptions on the studied WA G when dealing with current-state detectability:

Assumption 4.1. G is unambiguous.

Assumption 4.2. G is deadlock free, that is, for any state, there exists at least one output transition: $(\forall q \in Q)(\exists a \in E, q' \in Q)\mu(a)_{qq'} \neq \varepsilon$.

Assumption 4.3. There is no circuit labelled only by unobservable labels in G.

Assumption 4.1 requires that for any state q of G and any string $\omega \in E^*$, there is at most one path labeled by ω leading to q from the initial states. However, Assumption 4.1 does not imply that an observed weighted sequence can be generated by a single path. For example, assuming that $\otimes$ is the usual addition $+$, the WA G in Fig. 4.1 is such that $\sigma_o = (b, 0.5)(b, 0.8)$ can be generated by two paths, i.e., $\pi_1 = (1, u, 2)(2, b, 2)(2, b, 2)$ and $\pi_2 = (4, u, 3)(3, b, 3)(3, b, 3)$. In addition, Assumption 4.2 implies that the length of the generated weighted sequence becomes infinite as the system evolves indefinitely. By Assumption 4.3, for an infinite sequence, the length of its projection is also infinite.

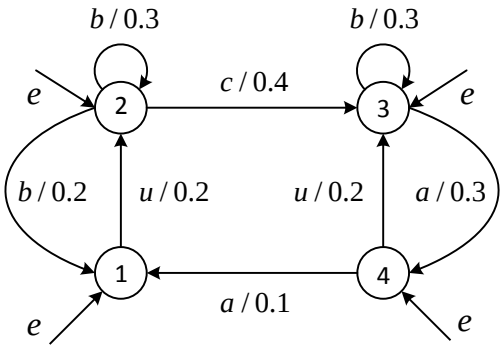

Figure 4.1 – Unambiguous weighted automaton G

4.2.1 Consistent State

In a logical DES, a *trajectory* of the system consists in an infinite sequence of labels that the system may generate. The set of all possible trajectories of the system is called ω-language [Thistle and Wonham, 1994]. Similarly, given a WA G, a trajectory consists in an infinite sequence of (label, weight) pairs that G may generate. The set of all possible trajectories of G defines the ω-language $L^\omega(G)$.

In Section 3.3, we investigate the problem of state estimation for WAs. Briefly, for a finite weighted sequence σ_o and a given value $\tau \geq w_f(\sigma_o)$, the goal is to determine the set of

possible current states that are consistent with σ_o and τ, which includes the states possibly reached by unobservable transitions after $\mathbb{E}_f(\sigma_o)$ leading to weights less than or equal to τ.

In this chapter, we check whether the set of possible current states at the end of weighted sequence $\sigma_o \in P(L(G))$ shrinks to a singleton, and if this continues with next observations. We then focus on the states that can be reached at the completion of σ_o for the following definition of consistent states. This is without restriction since the possible subsequent unobservable transitions (after $\mathbb{E}_f(\sigma_o)$) are considered through possible observations in continuation of σ_o.

Definition 4.1. Given an observed weighted sequence $\sigma_o \in P(L(G))$, the set of all σ_o-*consistent states* is defined as

$$C(\sigma_o) = \{q \in Q \mid \exists \sigma \in L(G), \exists q_0 \in Q_i : q_0 \overset{\omega}{\rightsquigarrow} q,\ P(\sigma) = \sigma_o,\ \mathbb{E}_f(\sigma) = \mathbb{E}_f(\sigma_o)\}. \tag{4.1}$$

In simple words, a state q is consistent with observation σ_o, if there exists a generated weighted sequence σ ending with the last label in σ_o leading to q such that the projection of σ coincides with σ_o.

Remark 4.1. In this chapter, we check whether we have the ability to know exactly the current state and the subsequent states for some or all trajectories of the system. Since the length of a weighted sequence can be infinite, the completion weight of the sequence can also be infinite. In this case, it does not make sense to consider a value greater than the completion weight of an observation when calculating the set of consistent states. That is why we consider a definition of consistent states that is different from the one given in Eq. (3.1). □

Example 4.1. Consider again the WA G in Fig. 4.1 with $\otimes = +$, $E_o = \{b, c\}$ and $E_{uo} = \{u\}$. Given $\sigma_o = (b, 0.5)(b, 0.8)$, it can be verified that the set of consistent states is $C(\sigma_o) = \{2, 3\}$. In fact, two different paths from an initial state have weighted sequences that are consistent with σ_o, namely, $\pi_1 = (1, u, 2)(2, b, 2)(2, b, 2)$ and $\pi_2 = (4, u, 3)(3, b, 3)(3, b, 3)$. Considering π_1, we have $\sigma_1 = \sigma(\pi_1) = (u, 0.2)(b, 0.5)(b, 0.8)$, $1 \overset{\sigma_1}{\rightsquigarrow} 2$ and $\mathbb{E}_f(\sigma_1) = \mathbb{E}_f(\sigma_o) = b$. Therefore, state 2 is consistent with σ_o. Similarly, state 3 is a consistent state by considering path π_2. □

4.2.2 Detectability notions in Weighted Automata

In this subsection, we extend the detectability problem defined for logical DESs modeled by NFAs in [Shu and Lin, 2011] to the framework of WAs.

For an arbitrary finite or infinite sequence σ, we denote $Pr(\sigma)$ the set of all its prefixes. Four types of detectabilities for WAs are defined as follows.

Definition 4.2 (Strong Detectability)**.** A WA G is strongly detectable with respect to a projection P if one can determine, after a finite number of pair observations, the current state and subsequent states of the automaton for all trajectories of the automaton. That is,

$$(\exists n \in \mathbb{N})(\forall \sigma \in L^{\omega}(G))(\forall \sigma' \in Pr(\sigma))|P(\sigma')| > n \Rightarrow |C(P(\sigma'))| = 1.$$

Definition 4.3 (Weak Detectability)**.** A WA G is detectable with respect to a projection P if one can determine, after a finite number of pair observations, the current state and subsequent states of the automaton for some trajectories of the automaton. That is,

$$(\exists n \in \mathbb{N})(\exists \sigma \in L^{\omega}(G))(\forall \sigma' \in Pr(\sigma))|P(\sigma')| > n \Rightarrow |C(P(\sigma'))| = 1.$$

Definition 4.4 (Strong Periodic Detectability)**.** A WA G is strongly periodically detectable with respect to a projection P if one can periodically determine the current state of the system for all trajectories of the automaton. That is,

$$(\exists n \in \mathbb{N})(\forall \sigma \in L^{\omega}(G))(\forall \sigma' \in Pr(\sigma))(\exists \sigma'' \in (E \times \mathcal{D})^{*})$$
$$\sigma'\sigma'' \in Pr(\sigma) \wedge |P(\sigma'')| < n \wedge |C(P(\sigma'\sigma''))| = 1.$$

Definition 4.5 (Weak Periodic Detectability)**.** A WA G is periodically detectable with respect to a projection P if one can periodically determine the current state of the system for some trajectories of the automaton. That is,

$$(\exists n \in \mathbb{N})(\exists \sigma \in L^{\omega}(G))(\forall \sigma' \in Pr(\sigma))(\exists \sigma'' \in (E \times \mathcal{D})^{*})$$
$$\sigma'\sigma'' \in Pr(\sigma) \wedge |P(\sigma'')| < n \wedge |C(P(\sigma'\sigma''))| = 1.$$

Problem 4.1. Solving the detectability problem of a WA aims to find a systematic way to check if the automaton is strongly detectable, detectable, strongly periodically detectable, and periodically detectable.

4.3 Observer-Based Approach

In this section, we first construct an observer that can be used to perform the current-state estimation for a UWA. Then, based on the constructed observer, the necessary and sufficient conditions for verifying the four detectabilities are proposed.

4.3.1 Construction of the Observer

In this subsection, we first construct an observer, denoted by G_{obs}, for an unambiguous WA G. In detail, the observer G_{obs} is a finite state automaton over a weighted alphabet $E_{obs} \subset E_o \times \mathcal{D}$, which is built so that it has the following structural properties: G_{obs} has only one initial state, and from a given state no two transitions of G_{obs} are labeled by the same weighted label $(a, t_a) \in E_{obs}$. That is, G_{obs} is a DFA over the weighted alphabet E_{obs}. Let us stress, however, that from a state in G_{obs} there may exist several output transitions labeled by the same label $a \in E_o$ but with different weights t_a. In this case, the external agent can distinguish the transitions from the different associated weights. Then, we prove that G_{obs} can be used to estimate the consistent states for any infinite or finite weighted sequence observed from system G.

Algorithm 4.1 details the construction of such an observer G_{obs} for a given unambiguous automaton G. This algorithm is first presented here and it will be adapted to deal with verification of the I-detectability in next chapter.

Algorithm 4.1 can be explained as follows. Step 2 determines the initial state of the observer G_{obs}, which is equal to the set of initial states of G. Stack *Queue* is employed to store it. At step 5, an element q_{obs} (which corresponds to a subset of Q) is removed from *Queue* for analysis. For each label $a \in E_o$, for each state $q \in q_{obs}$, and for each $i \in \{0, 1, \cdots, |Q| - 1\}$: we first include the value $\mu(u^i a)_{qq'}$ (resp. $(q', \mu(u^i a)_{qq'})$) in the set W_a (resp. Ω_a) for q' that can be reached from q via the occurrence of $u^i a$ (q' is reachable from q according to $u^i a$ if and only if $\mu(u^i a)_{qq'} \neq \varepsilon$). Note that when the *for loop* on i is completed, W_a consists of the weights of all paths in G labeled by $u^i a$ from state q, Ω_a contains all states that can be reached by paths labeled by $u^i a$ from q and the corresponding weights $\mu(u^i a)_{qq'}$ associated with the paths. At steps 16-18, for each $t \in W_a$, any state $q' \in Q$ is included to q'_{obs} if there exists a pair (q', τ) belonging to Ω_a such that $\tau = t$. If q'_{obs} is not contained in Q_{obs} (the current set of states for G_{obs}), then it is added in Q_{obs}, an arc labeled by (a, t) from node q_{obs} to node q'_{obs} is defined, and q'_{obs} is added in stack *Queue*. This process is repeated until stack *Queue* becomes empty.

Algorithm 4.1: Construction of the observer

Require: An unambiguous WA $G = (Q, E, \alpha, \mu, \beta)$ over semiring $\mathbb{S}$.
Ensure: The observer G_{obs} of G.

function *ObserverG*(G)
  Let $q_{i,obs} = Q_i$, $E_{obs} = \emptyset$, $Q_{obs} = \{q_{i,obs}\}$
  Enqueue $q_{i,obs}$ in *Queue*
  while *Queue* $\neq \emptyset$ **do**
    Dequeue q_{obs} from *Queue* /* Note that q_{obs} is a subset of Q */
    for each $a \in E_o$ **do**
      Let $W_a = \emptyset$, $\Omega_a = \emptyset$
      for each $q \in q_{obs}$ **do**
        for $i = 0, 1, \cdots, |Q| - 1$ **do**
          for each q' s.t. $\mu(u^i a)_{qq'} \neq \varepsilon$ **do**
            $W_a = W_a \cup \left\{\mu(u^i a)_{qq'}\right\}$
            $\Omega_a = \Omega_a \cup \left\{(q', \mu(u^i a)_{qq'})\right\}$
          end for
        end for
      end for
      for each $t \in W_a$ **do**
        $E_{obs} = E_{obs} \cup \{(a, t)\}$
        $q'_{obs} = \{q' \in Q \mid (q', t) \in \Omega_a\}$
        if $q'_{obs} \in Q_{obs}$ **then**
          $\delta_{obs}(q_{obs}, (a, t)) = q'_{obs}$
        else
          $Q_{obs} = Q_{obs} \cup \{q'_{obs}\}$
          $\delta_{obs}(q_{obs}, (a, t)) = q'_{obs}$
          Enqueue q'_{obs} in *Queue*
        end if
      end for
    end for
  end while
  return $G_{obs} = (Q_{obs}, E_{obs}, \delta_{obs}, q_{i,obs})$
end function

Note that each state q_{obs} of G_{obs} is a non-empty subset of Q, i.e., $q_{obs} \neq \emptyset \wedge q_{obs} \subseteq Q$. Therefore, in the worst case, the number of states of G_{obs} is $2^{|Q|} - 1$, and Algorithm 4.1 always terminates. As a result, the computational complexity of constructing the observer using Algorithm 4.1 is $\mathcal{O}(2^{|Q|})$. As usual for finite state automata, we extend the transition function $\delta_{obs} : Q_{obs} \times E_{obs} \to Q_{obs}$ to strings $\delta_{obs} : Q_{obs} \times E^*_{obs} \to Q_{obs}$.

Example 4.2. Consider the unambiguous WA G in Fig. 4.1 and $\otimes = +$. After applying Algorithm 4.1, we obtain the observer G_{obs} shown in Fig. 4.2. □

Let E^*_{obs} be the set of strings over weighted alphabet E_{obs} including (ϵ, e) corresponding to empty string ϵ and the identity weight value. The language generated by the observer

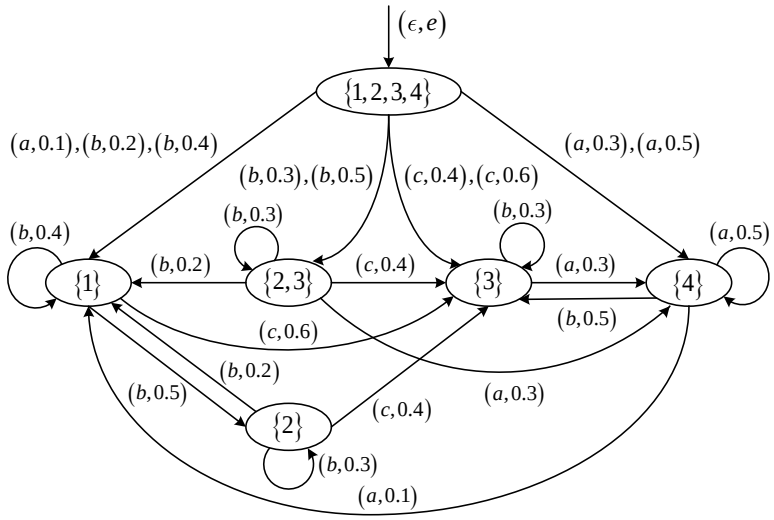

Figure 4.2 – Observer G_{obs} of G in Fig. 4.1

$G_{obs} = (Q_{obs}, E_{obs}, \delta_{obs}, q_{i,obs})$ is defined as:

$$L(G_{obs}) = \{\omega \in E_{obs}^* \mid \exists q_{obs} \in Q_{obs} : \delta_{obs}(q_{i,obs}, \omega) = q_{obs}\}.$$

It should be noticed that the language generated by G_{obs} is a subset of E_{obs}^*, i.e., $L(G_{obs}) \subseteq E_{obs}^*$. While the language generated by G is a subset of $(E \times \mathcal{D})^*$, i.e., $L(G) \subseteq (E \times \mathcal{D})^*$. For any observed weighted sequence $\sigma_o = (a_1, \tau_1)(a_2, \tau_2) \cdots (a_n, \tau_n) \in P(L(G))$, we define $\sigma_o^{elem} = (a_1, \tau_1)(a_2, \tau_2 \otimes^{-1} \tau_1) \cdots (a_k, \tau_k \otimes^{-1} \tau_{k-1})$ to denote its equivalent notation in E_{obs}^* where $\tau_k \otimes^{-1} \tau_{k-1}$ representing the weight[2] for the elementary transition according to a_k, $k = 2, 3, \cdots, n$. Similarly, for any sequence $\omega = (a_1, \tau_1)(a_2, \tau_2) \cdots (a_n, \tau_n) \in L(G_{obs})$, its equivalent notation in $(E \times \mathcal{D})^*$ can be defined by $\omega^{cum} = (a_1, \tau_1)(a_2, \tau_2') \cdots (a_n, \tau_n')$ where $\tau_k' = \tau_1 \otimes \cdots \otimes \tau_k$, $k = 2, 3, \cdots, n$, representing the cumulated weight for sequence $a_1 a_2 \cdots a_k$. We denote $L^{cum}(G_{obs})$ the equivalent notation of $L(G_{obs})$, that is,

$$L^{cum}(G_{obs}) = \{\sigma \in (E \times \mathcal{D})^* \mid \exists \omega \in L(G_{obs}) : \omega^{cum} = \sigma\}.$$

Similarly, we define

$$P(L(G))^{elem} = \{\sigma \in (E_o \times \mathcal{D})^* \mid \exists \sigma_o \in P(L(G)) : \sigma_o^{elem} = \sigma\}.$$

The following proposition shows that for any observation generated by system G, its consistent states can be determined by the observer G_{obs}.

2. Note that $\tau_k \otimes^{-1} \tau_{k-1}$ is defined as the value $x \in \mathcal{D}$ such that $x \otimes \tau_{k-1} = \tau_k$, with τ_k and $\tau_{k-1} \in \mathcal{D} \setminus \{\varepsilon\}$.

Proposition 4.1. The set of consistent states in G after observing the weighted sequence $(a_1, \tau_1)(a_2, \tau_2) \cdots (a_k, \tau_k)$ is given by $\delta_{obs}(q_{i,obs}, (a_1, \tau_1)(a_2, \tau_2 \otimes^{-1} \tau_1) \cdots (a_k, \tau_k \otimes^{-1} \tau_{k-1}))$, where $q_{i,obs}$ and δ_{obs} are the initial state and transition function of observer G_{obs}.

Proof. We prove the proposition by induction on the length of the observed weighted sequence.

(Base step.) The initial set of consistent states of G is given by $C(\epsilon) = q_{i,obs}$. Algorithm 4.1 states that $q_{i,obs} = Q_i$. That is, $q_{i,obs}$ is the set of states that can be reached from all initial states when no label has occurred, which coincides with the definition of $C(\epsilon)$.

(Inductive step.) Let $\sigma_o = (a_1, \tau_1)(a_2, \tau_2) \cdots (a_n, \tau_n) \in P(L(G))$ be an observed weighted sequence. Assume that $C(\sigma_o) = \delta_{obs}(q_{i,obs}, \sigma_o^{elem}) \in Q_{obs}$. We need to prove that if a new pair (a, τ) is observed in continuation of σ_o, then $C(\sigma_o \cdot (a, \tau)) = \delta_{obs}(C(\sigma_o), (a, \tau \otimes^{-1} \tau_n))$.

According to Algorithm 4.1, we have

$$\delta_{obs}(C(\sigma_o), (a, \tau \otimes^{-1} \tau_n)) = \{q \in Q \mid \exists q' \in C(\sigma_o), \\ \exists i \in \{0, 1, \cdots, |Q| - 1\} : \mu(u^i a)_{q'q} = \tau \otimes^{-1} \tau_n\}.$$

That is, $\delta_{obs}(C(\sigma_o), (a, \tau \otimes^{-1} \tau_n))$ consists of all states that can be reached from a state in $C(\sigma_o)$ via a path labeled by $u^i a$, $i \in \{0, 1, \cdots, \mid Q \mid -1\}$, and the weight associated with the path is equal to $\tau \otimes^{-1} \tau_n$. From Definition 4.1 of consistent states, we know that $C(\sigma_o)$ contains all states that can be reached by some generated weighted sequences ending with the last label in σ_o, and the projection of these generated weighted sequences is equal to σ_o. By assumption, $C(\sigma_o) = \delta_{obs}(q_{i,obs}, \sigma_o^{elem})$. Since G is assumed to be unambiguous , $\forall q \in C(\sigma_o)$ we have $q' \in C(\sigma_o \cdot (a, \tau))$ iff $\exists i \in \mathbb{N}$ such that $\tau = \tau_n \otimes \mu(u^i a)_{qq'}$. Therefore, $\delta_{obs}(C(\sigma_o), (a, \tau \otimes^{-1} \tau_n))$ consists of all states q that can be reached by a weighted sequence σ from an initial state such that $\mathbb{E}_f(\sigma) = \mathbb{E}_f(\sigma_o \cdot (a, \tau)) = a$ and $P(\sigma) = \sigma_o \cdot (a, \tau)$. This coincides with the definition of $C(\sigma_o \cdot (a, \tau))$. That is, $\delta_{obs}(C(\sigma_o), (a, \tau \otimes^{-1} \tau_n)) = C(\sigma_o \cdot (a, \tau))$. □

Proposition 4.1 indicates that observer G_{obs} can serve as the state estimator of WA G. Besides, from Proposition 4.1, we can claim that

$$\sigma_o \in P(L(G)) \Rightarrow \{\exists \omega \in E_{obs}^* : \delta_{obs}(q_{i,obs}, \omega) \neq \emptyset \wedge \omega^{cum} = \sigma_o\}.$$

This leads to the following corollary.

Corollary 4.1. The projection of language $L(G)$ generated by G is a subset of $L^{cum}(G_{obs})$, that is, $P(L(G)) \subseteq L^{cum}(G_{obs})$.

Note that $L^{cum}(G_{obs})$ is the equivalent presentation of the generated language of G_{obs} constructed by Algorithm 4.1.

The following lemma states the equivalence between the observed language generated by G and the language generated by observer G_{obs}.

Lemma 4.1. The projection of language $L(G)$ generated by G coincides with $L^{cum}(G_{obs})$, that is, $P(L(G)) = L^{cum}(G_{obs})$.

Proof. Using Corollary 4.1, it is sufficient to prove that $L^{cum}(G_{obs}) \subseteq P(L(G))$ or equivalently that $L(G_{obs}) \subseteq P(L(G))^{elem}$. Let $\omega = (a_1, \tau_1)(a_2, \tau_2)\cdots(a_n, \tau_n) \in E^*_{obs}$ such that $\delta_{obs}(q_{i,obs}, \omega) \neq \emptyset$. According to Algorithm 4.1, there must exist $q_0 \in Q_i$, $q_1, q_2, \cdots, q_n \in Q$ and $i_1, i_2, \cdots, i_n \in \{0, 1, \cdots, |Q|-1\}$ such that $\mu(u^{i_1}a_1)_{q_0q_1} = \tau_1, \mu(u^{i_2}a_2)_{q_1q_2} = \tau_2, \cdots, \mu(u^{i_n}a_n)_{q_{n-1}q_n} = \tau_n$. Defining $\pi = (q_0, u^{i_1}a_1, q_1)(q_1, u^{i_2}a_2, q_2)\cdots(q_{n-1}, u^{i_n}a_n, q_n)$, since automaton G is unambiguous, we have $\sigma(\pi) = (u^{i_1}a_1, \tau_1)(u^{i_2}a_2, \tau_1 \otimes \tau_2)\cdots(u^{i_n}a_n, \tau_1 \otimes \tau_2 \otimes \cdots \otimes \tau_n)$. This implies that $P(\sigma(\pi)) = (a_1, \tau_1)(a_2, \tau_1 \otimes \tau_2)\cdots(a_n, \tau_1 \otimes \tau_2 \otimes \cdots \otimes \tau_n)$ or equivalently that $P(\sigma(\pi))^{elem} = (a_1, \tau_1)(a_2, \tau_2)\cdots(a_n, \tau_n) = \omega$, i.e., $\omega \in P(L(G))^{elem}$. Therefore $L(G_{obs}) \subseteq P(L(G))^{elem}$ holds. □

Remark 4.2. From Proposition 4.1 and Lemma 4.1, it follows that the observer G_{obs} models the observable evolution of system G. On the one hand, for any observed sequence σ_o of G, all its consistent states, resulting of evolutions in G, can be characterised by the unique evolution labeled by σ_o^{elem} in G_{obs}. On the other hand, for any sequence ω of G_{obs}, there must be at least one evolution in G whose generated sequence coincides with ω. In addition, in the proof of Proposition 4.1 and Lemma 4.1, it is required that for any path $\pi = (q_0, u^{i_1}a_1, q_1)(q_1, u^{i_2}a_2, q_2)\cdots(q_{n-1}, u^{i_n}a_n, q_n)$ with $q_0 \in Q_i$, we have $\sigma(\pi) = (u^{i_1}a_1, \tau_1)(u^{i_2}a_2, \tau_1 \otimes \tau_2)\cdots(u^{i_n}a_n, \tau_1 \otimes \tau_2 \otimes \cdots \otimes \tau_n)$ where $\tau_k = \tau_{k-1} \otimes \mu(u^{i_k}a_k)_{q_{k-1}q_k}$ for $k = 1, 2, \cdots, n$ and $\tau_0 = 0$. Unambiguity of G is a sufficient condition for this property. This justifies Assumption 4.1 in this paper.

4.3.2 Criteria for Verifying Strong and Weak Detectabilities

In this subsection, we propose necessary and sufficient conditions, inspired by those presented in [Shu and Lin, 2011], to verify the four types of detectabilities for a UWA G.

Special attention must be paid to states belonging to elementary circuits because the trajectories of observer $G_{obs} = (Q_{obs}, E_{obs}, \delta_{obs}, q_{i,obs})$ can visit these states indefinitely. We

denote S_{ci} the set of all elementary circuits of G_{obs} as:

$$S_{ci} = \{(q_{obs}, s) \in Q_{obs} \times E_{obs}^* \mid \delta_{obs}(q_{obs}, s) = q_{obs} \wedge \\ |s| \geq 1 \wedge (\forall s' \in Pr(s) \text{ s.t. } s' \neq s : \delta_{obs}(q_{obs}, s') \neq q_{obs})\}.$$

Besides, a state of G_{obs} is defined as a subset of Q, and if this subset is a singleton then the state belongs to Q_{obs}^{single}, which is defined as:

$$Q_{obs}^{single} = \{q_{obs} \in Q_{obs} \mid |q_{obs}| = 1\}.$$

It should be noted that if a state in Q_{obs}^{single} is reached in G_{obs}, then there is a single consistent state for the corresponding observed weighted sequence. The following are the necessary and sufficient conditions for verifying the detectabilities of UWA $G = (Q, E, \alpha, \mu)$ based on its observer $G_{obs} = (Q_{obs}, E_{obs}, \delta_{obs}, q_{i,obs})$.

Theorem 4.1 (Criterion for Checking Strong Detectability)**.** A UWA G is strongly detectable with respect to projection P iff any state reachable from any elementary circuit in observer G_{obs} belongs to Q_{obs}^{single}, that is: $(\forall (q_{obs}, s) \in S_{ci})(\forall q' \in Q_{obs} \text{ s.t. } \exists \omega \in E_{obs}^*, \delta_{obs}(q_{obs}, \omega) = q')q' \in Q_{obs}^{single}$.

Proof. (If) Assume that G is not strongly detectable with respect to projection P, namely,

$$(\forall n \in \mathbb{N})(\exists \sigma \in L^\omega(G))(\exists \sigma' \in Pr(\sigma))|P(\sigma')| > n \wedge |C(P(\sigma'))| \neq 1.$$

Let $n = |Q_{obs}|$. Consider σ, σ' such that $\sigma' \in Pr(\sigma)$, $|P(\sigma')| > n$ and $|C(P(\sigma'))| \neq 1$. Since $|P(\sigma')| > n = |Q_{obs}|$, the sequence $P(\sigma')^{elem} \in E_{obs}^*$ must visit at least one elementary circuit $(q_{obs}, s) \in S_{ci}$ in G_{obs}. Let $v \in E_{obs}^*$ such that $\delta_{obs}(q_{i,obs}, v) = q_{obs}$ and $\omega \in E_{obs}^*$ such that $P(\sigma')^{elem} = v\omega$. By Proposition 4.1, we have $\delta_{obs}(q_{obs}, \omega) = C(P(\sigma'))$. By assumption $|C(P(\sigma'))| \neq 1$, we know that $\delta_{obs}(q_{obs}, \omega) \notin Q_{obs}^{single}$. Therefore, we can claim that $(\exists (q_{obs}, s) \in S_{ci})(\exists q' \in Q_{obs} \text{ s.t. } \exists \omega \in E_{obs}^*, \delta_{obs}(q_{obs}, \omega) = q')q' \notin Q_{obs}^{single}$.

(Only If) Assume $(\exists (q_{obs}, s) \in S_{ci})(\exists q' \in Q_{obs} \text{ s.t. } \exists \omega \in E_{obs}^*, \delta_{obs}(q_{obs}, \omega) = q')q' \notin Q_{obs}^{single}$. Let $v \in E_{obs}^*$ such that $\delta_{obs}(q_{i,obs}, v) = q_{obs}$. By assumption $\delta_{obs}(q_{obs}, \omega) \notin Q_{obs}^{single}$, we have $\delta_{obs}(q_{i,obs}, vs^j\omega) \notin Q_{obs}^{single}$ for $j \in \mathbb{N}$. Define σ' such that $P(\sigma')^{elem} = vs^j\omega$, by Proposition 4.1 and Lemma 4.1, we have $C(P(\sigma')) \neq 1$. Since j can be any arbitrary integer, we can conclude that

$$(\forall n \in \mathbb{N})(\exists \sigma \in L^\omega(G))(\exists \sigma' \in Pr(\sigma))|P(\sigma')| > n \wedge |C(P(\sigma'))| \neq 1.$$

That is, UWA G is not strongly detectable with respect to projection P. □

Remark 4.3. Due to Assumption 4.2, the finite state automaton G_{obs} constructed by Algorithm 4.1 is a (weakly connected) directed graph when ignoring the labels associated with its transitions. There exist algorithms to enumerate all the elementary circuits of such a directed graph in the literature, for instance in [Johnson, 1975]. These can be used to check whether any state in an elementary circuit of G_{obs} and any state reachable from such an elementary circuit belongs to Q_{obs}^{single}, that is to verify the strong detectability criterion stated in Theorem 4.1.

Theorem 4.2 (Criterion for Checking Detectability)**.** A UWA G is detectable with respect to projection P iff there is at least one elementary circuit composed of states which all belong to Q_{obs}^{single} in observer G_{obs}, that is: $(\exists(q_{obs}, s) \in S_{ci})(\forall \omega \in Pr(s))\delta_{obs}(q_{obs}, \omega) \in Q_{obs}^{single}$.

Proof. (If) We assume that $(\exists(q_{obs}, s) \in S_{ci})(\forall \omega \in Pr(s))\delta_{obs}(q_{obs}, \omega) \in Q_{obs}^{single}$. Let $v \in E_{obs}^*$ such that $\delta_{obs}(q_{i,obs}, v) = q_{obs}$. Define $\sigma \in P(L(G))$ such that $P(\sigma)^{elem} = vs^j$ where j is an arbitrary integer. By assumption $\delta_{obs}(q_{obs}, \omega) \in Q_{obs}^{single}$, we then have $\delta_{obs}(q_{i,obs}, vs^j\omega) \in Q_{obs}^{single}$. Let $n = |v| \in \mathbb{N}$, then by Proposition 4.1 and Lemma 4.1, we can claim that

$$(\exists n \in \mathbb{N})(\exists \sigma \in L^{\omega}(G))(\forall \sigma' \in Pr(\sigma))|P(\sigma')| > n \Rightarrow |C(P(\sigma'))| = 1.$$

That is, unambiguous automaton G is detectable with respect to projection P.

(Only If) Assume that G is detectable with respect to projection P, that is,

$$(\exists n \in \mathbb{N})(\exists \sigma \in L^{\omega}(G))(\forall \sigma' \in Pr(\sigma))|P(\sigma')| > n \Rightarrow |C(P(\sigma'))| = 1.$$

Since σ is an infinite sequence, then $P(\sigma)^{elem}$ must visit at least one elementary circuit $(q_{obs}, s) \in S_{ci}$ in G_{obs} infinitely often. Let $v \in E_{obs}^*$ such that $\delta_{obs}(q_{i,obs}, v) = q_{obs}$. By assumption $|C(P(\sigma'))| = 1$ and Proposition 4.1, we have $(\forall \omega \in Pr(s))\delta_{obs}(q_{i,obs}, v\omega) \in Q_{obs}^{single}$, which is equivalent to $(\forall \omega \in Pr(s))\delta_{obs}(q_{obs}, \omega) \in Q_{obs}^{single}$. Therefore, $(\exists(q_{obs}, s) \in S_{ci})(\forall \omega \in Pr(s))\delta_{obs}(q_{obs}, \omega) \in Q_{obs}^{single}$. □

Theorem 4.3 (Criterion for Checking Strong Periodic Detectability)**.** A UWA G is strongly periodically detectable with respect to projection P iff each elementary circuit of observer G_{obs} contains at least one state belonging to Q_{obs}^{single}, that is: $(\forall(q_{obs}, s) \in S_{ci})(\exists \omega \in Pr(s))\delta_{obs}(q_{obs}, \omega) \in Q_{obs}^{single}$.

Proof. (If) Assume that G is not strongly periodically detectable with respect to projection P, namely,

$$(\forall n \in \mathbb{N})(\exists \sigma \in L^{\omega}(G))(\exists \sigma' \in Pr(\sigma))(\forall \sigma'' \in (E \times \mathcal{D})^{*})$$
$$\sigma'\sigma'' \in Pr(\sigma) \wedge |P(\sigma'')| < n \Rightarrow |C(P(\sigma'\sigma''))| \neq 1.$$

Let $n = |Q_{obs}| + 1$ and $|P(\sigma'')| = |Q_{obs}|$. From the above assumption, we know that $P(\sigma)^{elem}$ must visit at least one elementary circuit $(q_{obs}, s) \in S_{ci}$ in G_{obs} such that it is the last elementary circuit passed by $P(\sigma'\sigma'')^{elem}$. Since $(\forall \sigma'' \in (E \times \mathcal{D})^{*})\sigma'\sigma'' \in Pr(\sigma) \wedge |P(\sigma'')| < |Q_{obs}| + 1 \Rightarrow |C(P(\sigma'\sigma''))| \neq 1$, by Proposition 4.1, we have $(\forall \omega \in Pr(s))\delta_{obs}(q_{obs}, \omega) \notin Q_{obs}^{single}$. Therefore, $(\exists (q_{obs}, s) \in S_{ci})(\forall \omega \in Pr(s))\delta_{obs}(q_{obs}, \omega) \notin Q_{obs}^{single}$.

(Only If) Assume that $(\exists (q_{obs}, s) \in S_{ci})(\forall \omega \in Pr(s))\delta_{obs}(q_{obs}, \omega) \notin Q_{obs}^{single}$. Let $v \in E_{obs}^{*}$ such that $\delta_{obs}(q_{i,obs}, v) = q_{obs}$. Define σ such that $P(\sigma)^{elem} = vs^{j}$ where j is an infinite integer and σ' such that $P(\sigma')^{elem} = v$. By assumption $\delta_{obs}(q_{obs}, \omega) \notin Q_{obs}^{single}$, we then have $\delta_{obs}(q_{i,obs}, vs^{j}\omega) \notin Q_{obs}^{single}$. Therefore, by Proposition 4.1 and Lemma 4.1, we can conclude that

$$(\forall n \in \mathbb{N})(\exists \sigma \in L^{\omega}(G))(\exists \sigma' \in Pr(\sigma))(\forall \sigma'' \in (E \times \mathcal{D})^{*})$$
$$\sigma'\sigma'' \in Pr(\sigma) \wedge |P(\sigma'')| < n \Rightarrow |C(P(\sigma'\sigma''))| \neq 1.$$

That is, unambiguous automaton G is not strongly periodically detectable with respect to projection P. □

Theorem 4.4 (Criterion for Checking Periodic Detectability)**.** A UWA G is periodically detectable with respect to projection P iff there is at least one elementary circuit in observer G_{obs} which contain at least one state belonging to Q_{obs}^{single}, that is: $(\exists (q_{obs}, s) \in S_{ci})(\exists \omega \in Pr(s))\delta_{obs}(q_{obs}, \omega) \in Q_{obs}^{single}$.

Proof. (If) Assume $(\exists (q_{obs}, s) \in S_{ci})(\exists \omega \in Pr(s))\delta_{obs}(q_{obs}, \omega) \in Q_{obs}^{single}$. Let $v \in E_{obs}^{*}$ such that $\delta_{obs}(q_{i,obs}, v) = q_{obs}$. Define σ such that $P(\sigma)^{elem} = vs^{j}$ where j is an arbitrary integer. By assumption $\delta_{obs}(q_{obs}, \omega) \in Q_{obs}^{single}$, we then have $\delta_{obs}(q_{i,obs}, vs^{j}\omega) \in Q_{obs}^{single}$. Let $n = |vs|$, then according to Proposition 4.1 and Lemma 4.1, we have

$$(\exists n \in \mathbb{N})(\exists \sigma \in L^{\omega}(G))(\forall \sigma' \in Pr(\sigma))(\exists \sigma'' \in (E \times \mathcal{D})^{*})$$
$$\sigma'\sigma'' \in Pr(\sigma) \wedge |P(\sigma'')| < n \wedge |C(P(\sigma'\sigma''))| = 1.$$

That is, unambiguous automaton G is periodically detectable with respect to projection P.

(Only If) Assume that G is periodically detectable with respect to projection P, that is,

$$(\exists n \in \mathbb{N})(\exists \sigma \in L^{\omega}(G))(\forall \sigma' \in Pr(\sigma))(\exists \sigma'' \in (E \times \mathcal{D})^{*})$$
$$\sigma'\sigma'' \in Pr(\sigma) \wedge |P(\sigma'')| < n \wedge |C(P(\sigma'\sigma''))| = 1.$$

Given an infinite sequence σ satisfying the conditions in the above equation, then $P(\sigma)^{elem}$ must visit at least one elementary circuit $(q_{obs}, s) \in S_{ci}$ of G_{obs} in which $|C(P(\sigma'\sigma''))| = 1$ holds for any $\sigma' \in Pr(\sigma)$ and for some $\sigma'' \in (E \times \mathcal{D})^{*}$. Let $v \in E^{*}_{obs}$ such that $\delta_{obs}(q_{i,obs}, v) = q_{obs}$. According to Proposition 4.1, we have $(\exists \omega \in Pr(s))\delta_{obs}(q_{i,obs}, v\omega) \in Q^{single}_{obs}$, which is equivalent to $(\exists \omega \in Pr(s))\delta_{obs}(q_{obs}, \omega) \in Q^{single}_{obs}$. Therefore, $(\exists (q_{obs}, s) \in S_{ci})(\exists \omega \in Pr(s))\delta_{obs}(q_{obs}, \omega) \in Q^{single}_{obs}$. □

Example 4.3. Consider the observer G_{obs} in Fig. 4.2 of UWA G in Fig. 4.1. By Theorem 4.1, G is not strongly detectable since there exists one elementary circuit, i.e., $\{2,3\} \xrightarrow{(b,0.3)} \{2,3\}$, with a state not in Q^{single}_{obs}. Because there are elementary circuits with all states in Q^{single}_{obs}, by Theorem 4.2, system G is detectable. For example, circuit $\{3\} \xrightarrow{(a,0.3)} \{4\} \xrightarrow{(b,0.5)} \{3\}$ has all its states in Q^{single}_{obs}. Because circuit $\{2,3\} \xrightarrow{(b,0.3)} \{2,3\}$ has its state in $Q_{obs} \setminus Q^{single}_{obs}$, by Theorem 4.3, G is not strongly periodically detectable. Because there are elementary circuits in G_{obs} which include at least one state belonging to Q^{single}_{obs}, by Theorem 4.4, automaton G is periodically detectable.

Example 4.4. Consider the UWA G in Fig. 4.3. Its observer G_{obs} computed by Algorithm 4.1 is shown in Fig. 4.4. From G_{obs}, after applying the above theorems, one can conclude that G is strongly detectable (hence, detectable, strongly periodically detectable and periodically detectable) since all states reachable from any elementary circuit of G_{obs} are entirely within Q^{single}_{obs}.

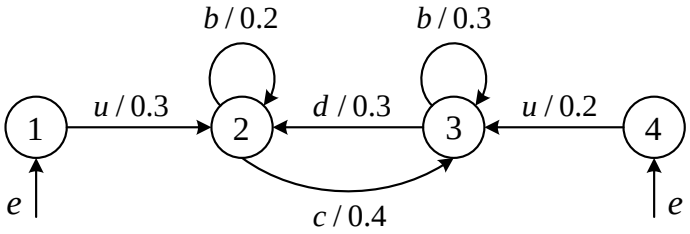

Figure 4.3 – Unambiguous weighted automaton G

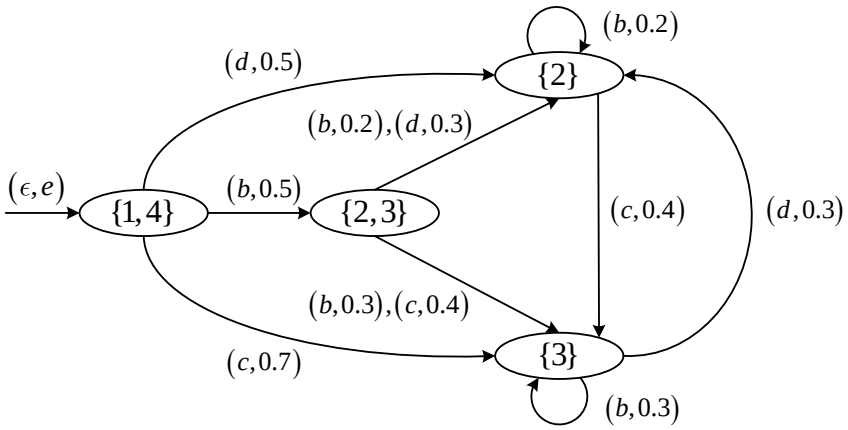

Figure 4.4 – Observer G_{obs} of automaton G in Fig. 4.3

4.3.3 Computational Complexity Analysis

From the procedure of constructing the observer G_{obs} for a UWA $G = (Q, E, \alpha, \mu)$, we know that, in the worst case, the number of states of G_{obs} is $2^{|Q|} - 1$. As a result, the computational complexity of constructing G_{obs} using Algorithm 4.1 is exponential with respect to the state space cardinality of G, that is, $\mathcal{O}(2^{|Q|})$.

Remark 4.4. For a UWA, in order to verify its detectabilities using Theorems 1-4, we have to construct its observer. Since the observer has at most $2^{|Q|} - 1$ states, the computational complexity of verifying the detectabilities is exponential with respect to the number of states of the underlying system.

The verification of detectabilities using observer-based criteria has a high computational complexity. Therefore, it is important to search for more efficient approaches to check detectabilities.

4.4 Detector-Based Approach

In this section, inspired by [Shu and Lin, 2011], the notion of detector is introduced for UWAs in order to reduce the verification complexity.

4.4.1 Construction of the Detector

Detector G_{det} is a finite state automaton whose structure relies on $G_{obs} = (X, E_{obs}, \delta, x_0)$. Each state of G_{det} is a subset of states of G, with a cardinality equals to one or two (except

for the initial state). The detector is defined in the following way:

$$G_{det} = (Y, E_{obs}, \xi, y_0) = Ac(Y', E_{obs}, \xi, y_0)$$

where y_0 is the unique initial state of detector G_{det}, defined as $y_0 = x_0 = Q_i$, $Ac(\cdot)$ represents the accessible part, i.e., $Ac(Y', E_{obs}, \xi, y_0)$ is the automaton obtained by removing all the states that are not reachable from the initial state y_0 as well as transitions associated with such states in automaton (Y', E_{obs}, ξ, y_0), and Y' is the set of subsets of Q that contains at most two elements, including the initial state, that is,

$$Y' = \{y' : y' \subseteq Q \wedge |y'| \leq 2\} \cup \{y_0\}.$$

The transition function $\xi : Y' \times E_{obs} \to 2^{Y'}$ is defined, for $y' \in Y'$ and $a \in E_{obs}$, based on the transition function δ of G_{obs} as follows:

$$\xi(y', a) = \begin{cases} \{\delta(y', a)\}, & if\ |\delta(y', a)| = 1; \\ \{y : y \subseteq \delta(y', a) \wedge |y| = 2\}, & if\ |\delta(y', a)| \geq 2; \\ \text{undefined}, & if\ |\delta(y', a)| = 0. \end{cases} \tag{4.2}$$

From the above description, we know that G_{det} is nondeterministic since any state in G_{obs} that contains more than two elements of Q is split into several states and each state contains two elements of Q. Moreover, the complexity of building G_{det} is polynomial since, in the worst case, the number of states of G_{det} is $|Y'| = 1 + |Q| + |Q|(|Q| - 1)/2$.

Example 4.5. Consider the automaton in Fig. 4.5 whose observer G_{obs} is shown in Fig.4.6. Its detector is depicted in Fig. 4.7. □

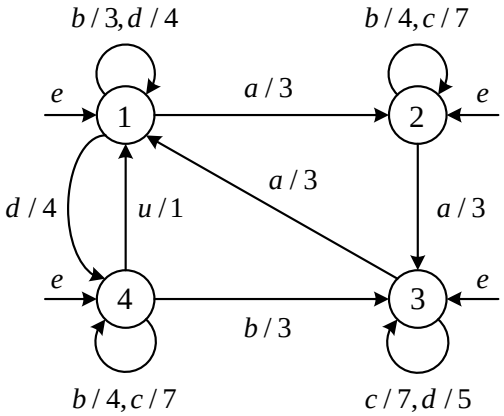

Figure 4.5 – An unweighted automaton G.

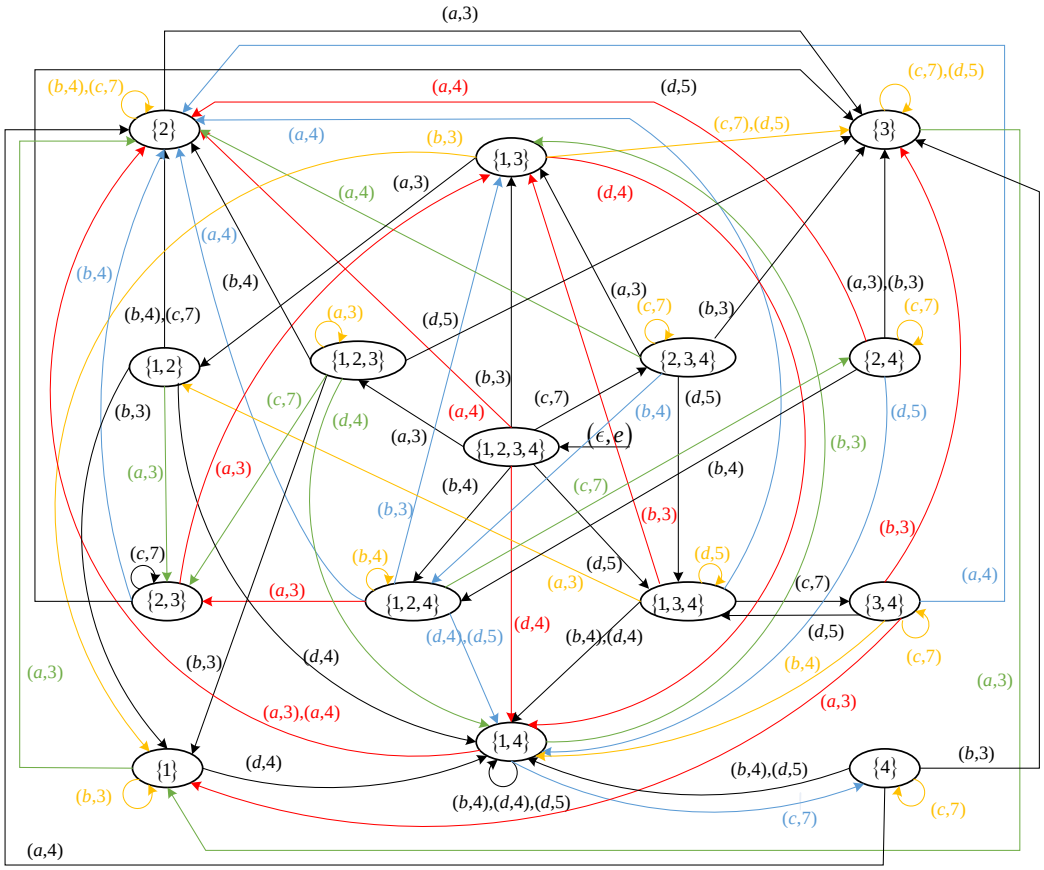

Figure 4.6 – The observer G_{obs} of automaton G in Fig. 4.5.

Let E^*_{obs} be the set of strings over weighted alphabet E_{obs} including (ϵ, e) corresponding to empty string ϵ and the identity weight value. We extend the transition function $\xi : Y \times E_{obs} \to 2^Y$ to strings $\xi : Y \times E^*_{obs} \to 2^Y$ in the usual way.

The following lemma states the relationship between observer G_{obs} and detector G_{det}.

Lemma 4.2. The observer G_{obs} and detector G_{det} have the following relation.

(1) For any $\omega \in E^*_{obs}$, if $|\delta(x_0, \omega)| = 1$ then $\xi(y_0, \omega) = \{\delta(x_0, \omega)\}$.

(2) For any $\omega \in E^*_{obs}$ and any $y \subseteq Q$ such that $|y| = 2$, $y \in \xi(y_0, \omega) \Leftrightarrow y \subseteq \delta(x_0, \omega)$.

Proof. According to the definition of ξ in Eq. (4.2), for all $y' \in Y$ and all $a \in E_{obs}$, it holds that

(F1) If $\delta(y', a)$ satisfies $|\delta(y', a)| = 1$, then $\xi(y', a) = \{\delta(x_0, a)\}$.

(F2) If $\delta(y', a)$ satisfies $|\delta(y', a)| \geq 2$, then for any state $y \in Q$ such that $|y| = 2$, $y \in \xi(y', a) \Leftrightarrow y \subseteq \delta(y', a)$.

We now prove this lemma by induction on the length $|\omega|$ of ω based on (F1) and (F2).

(Base step.) Consider $|\omega| = 1$, that is, $\omega = a \in E_{obs}$. Part (1) of the lemma can be proved by letting $y' = y_0$ in (F1), and part (2) of the lemma can be proved by letting $y' = y_0 = x_0$ in (F2).

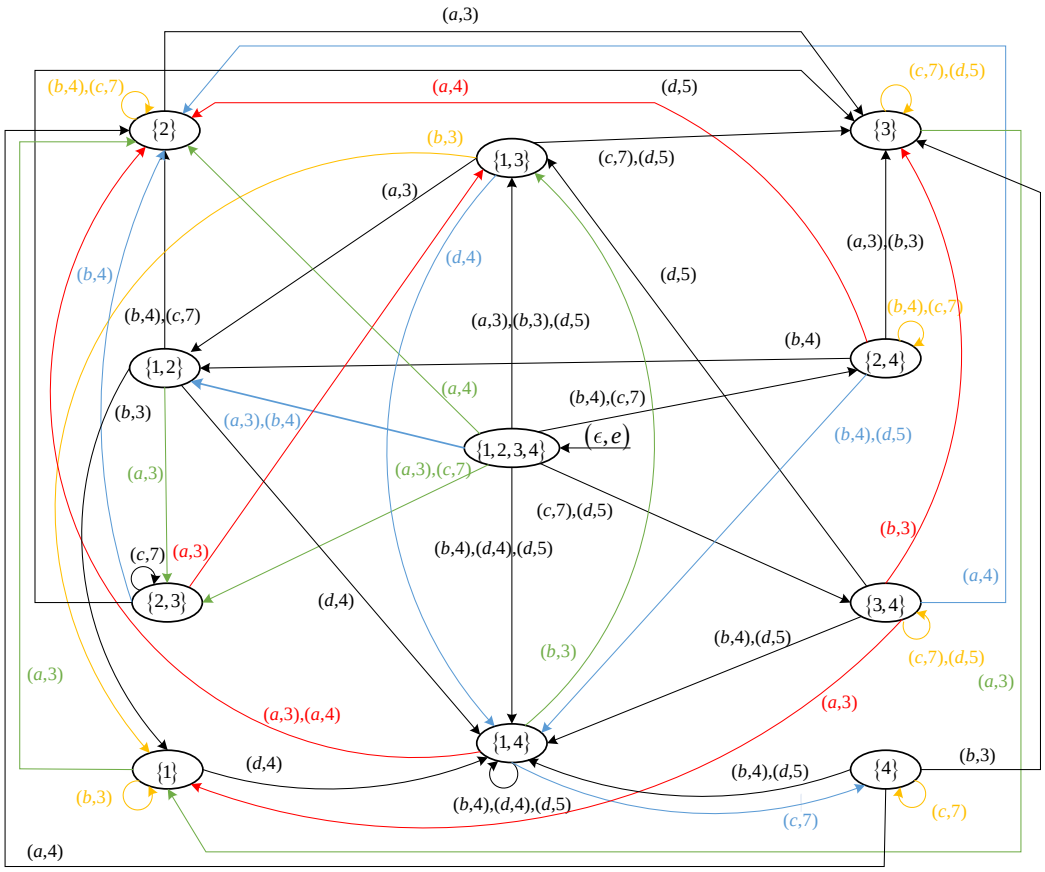

Figure 4.7 – The detector G_{det} of automaton G in Fig. 4.5.

(Induction hypothesis (IH).) Suppose that Lemma 1 holds for any $\omega \in E^*_{obs}$ such that $|\omega| \leq n$.

(Inductive step.) We now need to prove that Lemma 1 is true for any $\omega a \in E^*_{obs}$ such that $|\omega a| = n + 1$. Considering that the cardinality of set $\delta(x_0, \omega)$ and $\delta(x_0, \omega a)$ may be one or more than one, the following four possible cases need to be discussed.

Case 1: $|\delta(x_0, \omega)| = 1$ and $|\delta(x_0, \omega a)| = 1$. Since $|\delta(x_0, \omega a)| = 1$, we need to prove

$\xi(y_0, \omega a) = \{\delta(x_0, \omega a)\}$, which can be done as follows.

$$\begin{aligned}
\xi(y_0, \omega a) &= \xi(\xi(y_0, \omega), a) \\
&= \{y \in Y \mid \exists y' \in \xi(y_0, \omega) : y \in \xi(y', a)\} \\
&= \{y \in Y \mid \exists y' \in \{\delta(x_0, \omega)\} : y \in \xi(y', a)\} \\
&\quad \textit{(by IH and } |\delta(x_0, \omega)| = 1\textit{)} \\
&= \{y \in Y \mid y \in \xi(\delta(x_0, \omega), a)\} \\
&= \{y \in Y \mid y \in \{\delta(\delta(x_0, \omega), a)\}\} \\
&\quad \textit{(by (F1) and } |\delta(x_0, \omega a)| = 1\textit{)} \\
&= \{\delta(\delta(x_0, \omega), a)\} \\
&= \{\delta(x_0, \omega a)\}
\end{aligned}$$

Case 2: $|\delta(x_0, \omega)| \geq 2$ and $|\delta(x_0, \omega a)| = 1$. Since $|\delta(x_0, \omega a)| = 1$, we need to prove $\xi(y_0, \omega a) = \{\delta(x_0, \omega a)\}$, which can be done as follows.

$$\begin{aligned}
\xi(y_0, \omega a) &= \xi(\xi(y_0, \omega), a) \\
&= \{y \in Y \mid \exists y' \in \xi(y_0, \omega) : y \in \xi(y', a)\} \\
&= \{y \in Y \mid \exists y' \subseteq \delta(x_0, \omega) : y \in \xi(y', a)\} \\
&\quad \textit{(by IH and } |\delta(x_0, \omega)| \geq 2\textit{)} \\
&= \{y \in Y \mid \exists y' \subseteq \delta(x_0, \omega) : y \in \{\delta(y', a)\}\} \\
&\quad \textit{(by (F1) and}\,^{3}\ |\delta(x_0, \omega a)| = 1\textit{)} \\
&= \{y \in Y \mid \exists y' \subseteq \delta(x_0, \omega) : y = \delta(y', a)\} \\
&= \{\delta(\delta(x_0, \omega), a)\} \\
&= \{\delta(x_0, \omega a)\}
\end{aligned}$$

Case 3: $|\delta(x_0, \omega)| = 1$ and $|\delta(x_0, \omega a)| \geq 2$. Since $|\delta(x_0, \omega a)| \geq 2$, we need to prove for

3. Since $|\delta(x_0, \omega)| \geq 2$ and $|\delta(x_0, \omega a)| = 1$, then there exists $y' \subseteq \delta(x_0, \omega)$ such that $|\delta(y', a)| = 1$. Moreover, for any $y' \subseteq \delta(x_0, \omega)$, we have $|\delta(y', a)| = 1$ or $|\delta(y', a)| = 0$.

any $y \subseteq Q$ such that $|y| = 2$, $y \in \xi(y_0, \omega a) \Leftrightarrow y \subseteq \delta(x_0, \omega a)$, which can be done as follows.

$$\begin{aligned}
y \in \xi(y_0, \omega a) &\Leftrightarrow y \in \xi(\xi(y_0, \omega), a) \\
&\Leftrightarrow (\exists y' \in \xi(y_0, \omega)) y \in \xi(y', a) \\
&\Leftrightarrow (\exists y' \in \delta(x_0, \omega)) y \in \xi(y', a) \\
&\quad \textit{(by IH and } |\delta(x_0, \omega)| = 1\textit{)} \\
&\Leftrightarrow (\exists y' = \delta(x_0, \omega)) y \in \xi(y', a) \\
&\Leftrightarrow y \in \xi(\delta(x_0, \omega), a) \\
&\Leftrightarrow y \subseteq \delta(\delta(x_0, \omega), a) \\
&\quad \textit{(by (F2) and } |\delta(x_0, \omega a)| \geq 2 \\
&\Leftrightarrow y \subseteq \delta(x_0, \omega a)
\end{aligned}$$

Case 4: $|\delta(x_0, \omega)| \geq 2$ and $|\delta(x_0, \omega a)| \geq 2$. Since $|\delta(x_0, \omega a)| \geq 2$, we need to prove for any $y \subseteq Q$ such that $|y| = 2$, $y \in \xi(y_0, \omega a) \Leftrightarrow y \subseteq \delta(x_0, \omega a)$, which can be done as follows.

$$\begin{aligned}
y \in \xi(y_0, \omega a) &\Leftrightarrow y \in \xi(\xi(y_0, \omega), a) \\
&\Leftrightarrow (\exists y' \in \xi(y_0, \omega)) y \in \xi(y', a) \\
&\Leftrightarrow (\exists y' \subseteq \delta(x_0, \omega)) y \in \xi(y', a) \\
&\quad \textit{(by IH and } |\delta(x_0, \omega)| \geq 2\textit{)} \\
&\Leftrightarrow (\exists y' \subseteq \delta(x_0, \omega)) y \subseteq \delta(y', a) \\
&\quad \textit{(by (F2) and }^{4} |\delta(x_0, \omega a)| \geq 2\textit{)} \\
&\Leftrightarrow y \subseteq \delta(\delta(x_0, \omega), a) \\
&\Leftrightarrow y \subseteq \delta(x_0, \omega a)
\end{aligned}$$

Therefore, Lemma 4.2 is established. □

4.5 Creteria for Verifying Weak Detectabilities

In this section, we first show that strong detectability and strong periodic detectability of a UWA G can be efficiently checked by using its detector G_{det}. We then illustrate that the conditions stated for the observer for weak detectability and weak periodic detectability cannot be transposed to the detector.

4. More precisely, since $y \in \xi(y', a)$ and $|y| \geq 2$, according to the definition of ξ, then $|\delta(y', a)| \geq 2$. By (F2), we have $y \in \xi(y', a) \Leftrightarrow y \subseteq \delta(y', a)$.

We denote by EC_{det} the set of all elementary circuits of detector G_{det}, which is defined as:

$$EC_{det} = \{(y,s) \in Y \times E^*_{obs} \mid y \in \xi(y,s) \wedge |s| \geq 1 \\ \wedge (\forall s' \in Pr(s) \text{ s.t. } s' \neq s : y \notin \xi(y,s'))\}.$$

Besides, we denote by Y_{single} the set of singleton states of G_{det}, that is,

$$Y_{single} = \{y \in Y \mid |y| = 1\}.$$

Theorem 4.5 (Criterion for Checking Strong Periodic Detectability)**.** A UWA G is strongly periodically detectable with respect to projection P iff each elementary circuit of observer G_{det} contains at least one state belonging to Y_{single}, that is: $(\forall (y,s) \in EC_{det})(\exists \omega \in Pr(s))\xi(y,\omega) \subseteq Y_{single}$.

Proof. We have to prove that $(\forall (x,s) \in EC_{obs})(\exists \omega \in Pr(s))\delta(x,\omega) \in X_{single}$ iff $(\forall (y,s) \in EC_{det})(\exists \omega \in Pr(s))\xi(y,\omega) \subseteq Y_{single}$.

(Only If) Assume $(\forall (x,s) \in EC_{obs})(\exists \omega \in Pr(s))\delta(x,\omega) \in X_{single}$, then any elementary circuit contain at least one state $\delta(x,\omega) \in X_{single}$. Such a singleton state $\delta(x,\omega)$ will not be split in G_{det}. Formally, for any $(x,s) \in EC_{obs}$, let v be any string that leads to x from the initial state, that is, $\delta(x_0,v) = x$. By Lemma 1, $\xi(y_0, v\omega) = \{\delta(x_0, v\omega)\}$. According to the construction procedure of G_{det}, although G_{det} may have more elementary circuits than G_{obs}, all the elementary circuits in G_{det} must contain at least one singleton state. Therefore, if $(\forall (x,s) \in EC_{obs})(\exists \omega \in Pr(s))\delta(x,\omega) \in X_{single}$ then $(\forall (y,s) \in EC_{det})(\exists \omega \in Pr(s))\xi(y,\omega) \subseteq Y_{single}$.

(If) Assume that $(\forall (x,s) \in EC_{obs})(\exists \omega \in Pr(s))\delta(x,\omega) \in X_{single}$ is not true, i.e., $(\exists (x,s) \in EC_{obs})(\forall w \in Pr(s))\delta(x,w) \notin X_{single}$. Then we have the following path in G_{obs}.

$$x_1 \xrightarrow{a_1} x_2 \xrightarrow{a_2} x_3 \cdots \xrightarrow{a_{n-2}} x_{n-1} \xrightarrow{a_{n-1}} x_n$$

where $a_1, a_2, \cdots, a_{n-1} \in E_{obs}$, $n \in \mathbb{N}$ is a large enough number, and

$$\begin{cases} x_i = \delta(x_{i-1}, a_{i-1}), & i = 2,3,\cdots,n; \\ |x_i| \geq 2, & i = 1,2,\cdots,n. \end{cases}$$

Note that x_1 can be any state that belongs to elementary circuit (x,s). Let x'_n be a sub-state of x_n of cardinality two, that is, $x'_n \subseteq x_n$ and $|x'_n| = 2$. For x'_n and a_{n-1}, according to the construction procedure of G_{obs}, we can find $x'_{n-1} \subseteq x_{n-1}$ such that $x'_n \subseteq \delta(x'_{n-1}, a_{i-1})$ and

$|x'_{n-1}| = 2$. By the same way, we can find $x'_{n-2} \subseteq x_{n-2}, x'_{n-3} \subseteq x_{n-3}, \cdots, x'_1 \subseteq x_1$ such that

$$\begin{cases} x'_i \subseteq \delta(x'_{i-1}, a_{i-1}), & i = 2, 3, \cdots, n-1; \\ |x'_i| = 2, & i = 1, 2, \cdots, n-2. \end{cases}$$

According to the construction procedure of G_{det}, we have a path in G_{det} as

$$x'_1 \xrightarrow{a_1} x'_2 \xrightarrow{a_2} x'_3 \cdots \xrightarrow{a_{n-2}} x'_{n-1} \xrightarrow{a_{n-1}} x'_n.$$

Since n is large enough and G_{det} is finite, then there must be an elementary circuit in the above path such that each node consists of two states of the original system G. That is, $(\exists (y, s) \in EC_{det})(\forall w \in Pr(s))\xi(y, w) \nsubseteq Y_{single}$. □

Theorem 4.6 (Criterion for Checking Strong Detectability). A UWA G is strongly detectable with respect to projection P iff any state reachable from any elementary circuit in observer G_{obs} belongs to Y_{single}, that is, $(\forall (y, s) \in EC_{det})(\forall y' \in Y \text{ s.t. } \exists \omega \in E^*_{obs}, y' \in \xi(y, \omega))y' \in Y_{single}$.

Proof. We have to prove that $(\forall (x, s) \in EC_{obs})(\forall x' \in X \text{ s.t. } \exists \omega \in E^*_{obs}, x' = \delta(x, \omega))x' \in X_{single}$ iff $(\forall (y, s) \in EC_{det})(\forall y' \in Y \text{ s.t. } \exists \omega \in E^*_{obs}, y' \in \xi(y, \omega))y' \in Y_{single}$.

(Only If) Assume $(\forall (x, s) \in EC_{obs})(\forall x' \in X \text{ s.t. } \exists \omega \in E^*_{obs}, x' = \delta(x, \omega))x' \in X_{single}$, then in particular, each state in any elementary circuit of G_{obs} belongs to X_{single}, that is, $(\forall (x, s) \in EC_{obs})x \in X_{single}$. According to the construction of G_{det}, all these states will not be split in G_{det}. Formally, for any $(x, s) \in EC_{obs}$, let v be any string that leads to x from the initial state of G_{obs}, that is, $\delta(x_0, v) = x$. By Lemma 4.2, $\xi(y_0, v) = \{\delta(x_0, v)\}$. Since $(\forall x' \in X \text{ s.t. } \exists \omega \in E^*_{obs}, x' = \delta(x, \omega))x' \in X_{single}$, let $y = x$, and by Lemma 4.2, we get $(\forall y' \in Y \text{ s.t. } \exists \omega \in E^*_{obs}, y' \in \xi(y, \omega))y' \in Y_{single}$. Therefore, if $(\forall (x, s) \in EC_{obs})(\forall x' \in X \text{ s.t. } \exists \omega \in E^*_{obs}, x' = \delta(x, \omega))x' \in X_{single}$ is true, then $(\forall (y, s) \in EC_{det})(\forall y' \in Y \text{ s.t. } \exists \omega \in E^*_{obs}, y' \in \xi(y, \omega))y' \in Y_{single}$.

(If) Assume $(\forall (x, s) \in EC_{obs})(\forall x' \in X \text{ s.t. } \exists \omega \in E^*_{obs}, x' = \delta(x, \omega))x' \in X_{single}$ is not true, i.e., $(\exists (x, s) \in EC_{obs})(\exists x' \in X \text{ s.t. } \exists \omega \in E^*_{obs}, x' = \delta(x, \omega))x' \notin X_{single}$. Now we need to prove $(\exists (y, s) \in EC_{det})(\exists y' \in Y \text{ s.t. } \exists \omega \in E^*_{obs}, y' \in \xi(y, \omega))y' \notin Y_{single}$, which can be done by discussing the following cases.

Case 1: State x' belongs to (x, s) and (x, s) contains at least one state belongs to X_{single}. According to the construction procedure of G_{det}, there must exist a circuit in G_{det} containing a state that does not belong to Y_{single} which is obtained by splitting state x'.

Case 2: State x' belongs to (x, s) and all states in (x, s) do not belong to X_{single}. According

to the (If) part of Theorem 4.5, there exists a circuit in G_{det} containing a state that does not belong to Y_{single}.

Case 3: State x' does not belong to (x, s) and all states in (x, s) belong to X_{single}. Let $y = x$, then according to the construction procedure of G_{det}, there is a circuit (y, s) with all its states belonging to Y_{single} in G_{det}. Besides, since $x' \notin X_{single}$, by Lemma 4.2, we have $(\exists y' \in Y \text{ s.t. } \exists \omega \in E^*_{obs}, y' \in \xi(y, \omega)) y' \notin Y_{single}$ (y' is obtained by splitting state x'). □

Example 4.6. Consider again the WA G in Fig. 4.5. Its detector G_{det} is depicted in Fig. 4.7. Because there is elementary circuit, e.g., $\{1,3\} \xrightarrow{(d,4)} \{1,4\} \xrightarrow{(b,3)} \{1,3\}$, with all its states belong to $Y \setminus Y_{single}$ in G_{det}, according to Theorems 4.5 and 4.6, G is not strongly detectable and not strongly periodically detectable. □

In Section 4.3.2, it is proved that a UWA G is weakly detectable iff there is at least one elementary circuit composed of states that all belong to X_{single} in its observer G_{obs}, and is weakly periodically detectable iff there is at least one elementary circuit in G_{obs} which contain at least one state belonging to X_{single}. However, we cannot use the same conditions stated for detector G_{det} to check weak detectability and weak periodic detectability. The following example illustrates this.

Example 4.7. Consider the UWA G in Fig. 4.8. Its observer G_{obs} calculated by Algorithm 4.1 is depicted in Fig. 4.9. Its detector G_{det} is depicted in Fig. 4.10. According to G_{obs}, G is not weakly detectable as well as not weakly periodically detectable since each state of all elementary circuits belongs to $X \setminus X_{single}$. However, in G_{det}, there exists elementary circuit $\{7\} \xrightarrow{(c,1)} \{9\} \xrightarrow{(f,2)} \{7\}$ that has all its states in Y_{single}. Therefore, the fact that there exists a circuit with all its states in Y_{single} is neither sufficient nor necessary for weak detectability and weak periodic detectability. □

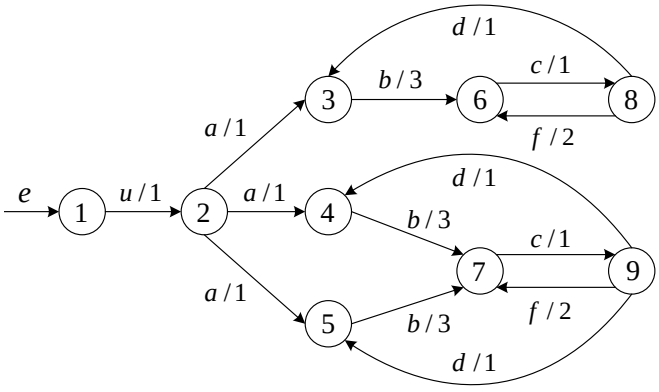

Figure 4.8 – An unambiguous weighted automaton G.

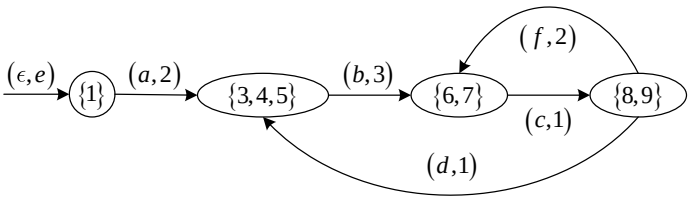

Figure 4.9 – The observer G_{obs} of G in Fig. 4.8.

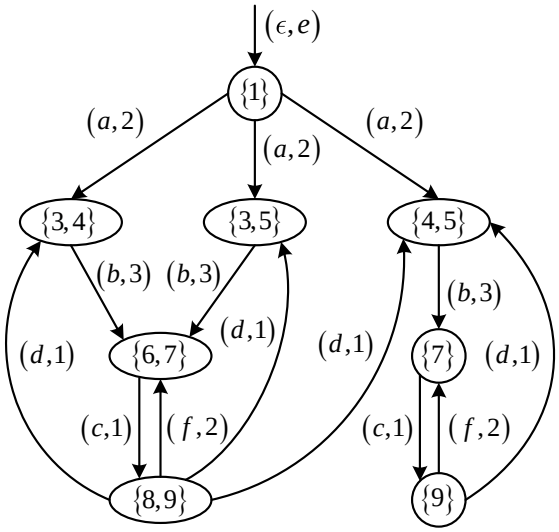

Figure 4.10 – The detector G_{det} of G in Fig. 4.8.

4.6 Conclusion

In this chapter, we deal with the current-state detectability problem for UWAs under partial event observation. An algorithm is proposed to build a DFA, called observer, having the ability to estimate the current states of the studied system for any infinite or finite observation. Inspired by the work in [Shu and Lin, 2011], criteria are proposed for checking four types of detectabilities. A disadvantage of the observer-based approach is that the verification complexity is very high since the observer's state number is exponential with respect to the number of states of the original system. In order to overcome this shortcoming, a computational complexity reduced approach is proposed to deal with strong detectabilities. Instead of using observer, this approach relies on the tool of detector whose size is polynomial with respect to the number of states of the original UWA. The complexity of checking strong detectability and strong periodic detectability by using a detector is polynomial time.

Part of the work in this chapter has been submitted as follows:

A. Lai, S. Lahaye and A. Giua. "Initial-state detectability and initial-state opacity of unambiguous weighted automata", *IEEE Transaction on Automatic Control* (submitted), 2019 [Lai *et al.*, 2019c].

5

I-Detectability and I-Opacity Verification for UWAs

This chapter focuses on the verification of initial-state detectability (I-detectability) and initial-state opacity (I-opacity) for unambiguous weighted automata (UWAs).

Contents

5.1 Introduction

In Chapter 4, we investigate the verification of current-state detectability for UWAs. Current-state detectable requires that system's current state can be uniquely determined. However, in some applications, instead of determining the current state of the system, one may be interested in knowing which initial states the system may start from. This is referred to as the initial-state estimation problem. In this chapter, this problem is investigated in terms of I-detectability, characterizing whether one can uniquely determine the initial state of a system after a finite number of observations. On the other hand, we study the I-opacity of UWAs, which is characterized by whether an intruder can infer that the evolution of the system has started from a secret initial state.

This chapter is structured as follows. Section 5.2 formally defines the problem of I-detectability and gives the definitions of two types of I-detectabilities, i.e., strong I-Detectability and Weak I-Detectability. In Subsection 5.3.1, given a UWA G, we detail the process of constructing the I-observer, i.e., initial-state estimator, of G. In Subsection 5.3.2, necessary and sufficient conditions based on constructed I-observer are introduced to verify the I-detectabilities of a UWA. Subsection 5.3.3 presents a necessary and sufficient condition for verifying I-opacity. In Subsection 5.3.4, we analyse the complexity of verification of I-detectability and I-opacity using our approach. Finally, conclusions are drawn in Section 5.4.

5.2 Problem Formulation

In this chapter, we assume that the studied WA $G = (Q, E, \alpha, \mu)$ have identity initial weights and that all unobservable labels are represented by symbol u, i.e., $E = E_o \cup \{u\}$. In addition, the assumptions in Chapter 4 are needed when dealing with the verification of I-detectability and I-opacity, i.e.,

- G is unambiguous;
- G is deadlock free, that is, for any state of the system, there exists at least one output transition: $(\forall q \in Q)(\exists a \in E, q' \in Q)\mu(a)_{qq'} \neq \varepsilon$;
- There is no circuit labeled only by unobservable labels in G.

5.2.1 I-Detectability

Definition 5.1. Given a UWA G, the set of possible initial states after observing $\sigma_o \in P(L(G))$ is defined as

$$I(\sigma_o) = \{q \in Q_i \mid \exists q' \in Q, \quad \exists \sigma \in L(G) : P(\sigma) = \sigma_o, q \overset{\sigma}{\rightsquigarrow} q'\}. \tag{5.1}$$

In simple words, $I(\sigma_o)$ consists of all initial states from which there exist generated weighted sequences such that their projections coincide with the observation.

Lemma 5.1. Let

$$I'(\sigma_o) = \{q \in Q_i \mid \exists q' \in Q, \exists \sigma \in L(G) : \quad P(\sigma) = \sigma_o, q \overset{\sigma}{\rightsquigarrow} q', \mathbb{E}_f(\sigma) = \mathbb{E}_f(\sigma_o)\}. \tag{5.2}$$

Then $I(\sigma_o) = I'(\sigma_o)$.

Proof. Let q_i be an initial state with $q_i \in I'(\sigma_o)$. According to Eqs. (5.1) and (5.2), $q_i \in I(\sigma_o)$. On the other hand, consider an initial state $q_0 \in I(\sigma_o)$. According to Eq. (5.1), there must exist a path π from q_0 and ending with $\mathbb{E}_f(\sigma_o)$ or an unobservable label such that $P(\sigma(\pi)) = \sigma_o$. If π ends with $\mathbb{E}_f(\sigma_o)$, then it is trivial that $q_0 \in I'(\sigma_o)$. When π ends with an unobservable label, we assume that it can be represented as $\pi = (q_0, u^{j_1}e_1, q_1)\cdots(q_{n-1}, u^{j_n}e_n, q_n)(q_n, u, q_{n+1})$, where $j_1, \cdots, j_n \in \mathbb{N}$. Let $\pi' = (q_0, u^{j_1}e_1, q_1)\cdots(q_{n-1}, u^{j_n}e_n, q_n)$. Then π' satisfies Eq. (5.2). Hence, q_0 belongs to $I'(\sigma_o)$. □

In a logical DES, a possible trajectory of the system is represented by an infinite sequence of labels that the system may generate. The set of all possible trajectories of the system is denoted by the ω-language [Thistle and Wonham, 1994]. Similarly, in the framework of WAs, let a possible trajectory of a WA G also be represented by an infinite sequence of (label, weight) pairs that G may generate. The set of all possible trajectories of G defines the ω-language $L^{\omega}(G)$. For any finite or infinite sequence σ, we denote by $Pr(\sigma)$ the set of all its prefixes.

Inspired by [Shu and Lin, 2013] where I-detectability is investigated for NFAs, we define strong I-detectability and weak I-detectability for UWAs as follows.

Definition 5.2 (Strong I-Detectability)**.** A UWA G is strongly I-detectable with respect to projection P if, for all trajectories, the set of possible initial states shrinks to a singleton after a finite number of observations, i.e.,

$$(\exists n \in \mathbb{N})(\forall \sigma \in L^{\omega}(G))(\forall \sigma' \in Pr(\sigma))|P(\sigma')| > n \Rightarrow |I(P(\sigma'))| = 1.$$

Definition 5.3 (Weak I-Detectability)**.** A UWA G is weakly I-detectable with respect to projection P if, for some trajectories, the set of possible initial states shrinks to a singleton after a finite number of observations, i.e.,

$$(\exists n \in \mathbb{N})(\exists \sigma \in L^{\omega}(G))(\forall \sigma' \in Pr(\sigma))|P(\sigma')| > n \Rightarrow |I(P(\sigma'))| = 1.$$

From the above definitions, we know that if a UWA is strongly I-detectable, then it is weakly I-detectable.

Problem 5.1. Check, in a systematic way, if a given UWA $G = (Q, E, \alpha, \mu)$ is strongly I-detectable (resp. weekly I-detectable).

5.2.2 I-Opacity

The generated weighted language of a UWA $G = (Q, E, \alpha, \mu)$ from an initial state $q_i \in Q_i$ is defined as

$$L(G, q_i) = \{\sigma \in (E \times \mathcal{D})^* \mid \exists q \in Q, \exists \omega \in E^*, \exists \pi \in q_i \overset{\omega}{\rightsquigarrow} q : \sigma(\pi) = \sigma\}. \tag{5.3}$$

In addition, given a subset of the initial state set $X \subseteq Q_i$, we define the weighted language generated by X as

$$L(G, X) = \bigcup_{q_i \in X} L(G, q_i).$$

Due to unambiguity, if Q_i is divided into two disjoint subsets: Q_i^1 and Q_i^2, then the generated weighted language of G can be represented by

$$L(G) = \bigcup_{q_i \in Q_i} L(G, q_i) = L(G, Q_i^1) \cup L(G, Q_i^2),$$

and we have

$$P(L(G)) = P(L(G, Q_i^1)) \cup P(L(G, Q_i^2)).$$

Given a UWA $G = (Q, E, \alpha, \mu)$ with generated weighted language $L(G)$, the intruder can only observe the projected weighted sequences, i.e., $P(L(G))$. Assume that the intruder has full knowledge of the structure of G, and that the secret is described by an arbitrary subset of Q. A system is said to be initial-state opaque with respect to the secret if the intruder can never infer that the initial state of the system is within the secret. The I-opacity property of a UWA is formally defined as follows.

Definition 5.4 (Initial-state opacity)**.** Given a UWA $G = (Q, E, \alpha, \mu)$, projection $P : E^* \to E_o^*$, and a set of secret states $Q_s \subseteq Q$, G is initial-state opaque with respect to Q_s and P, if for all $q \in Q_i \cap Q_s$ and for all $\sigma \in L(G, q)$, there exist $q' \in Q_i \setminus Q_s$ and $\sigma' \in L(G, q')$ such that $P(\sigma') = P(\sigma)$, that is, $P(L(G, Q_i \cap Q_s)) \subseteq P(L(G, Q_i \setminus Q_s))$, or equivalently, $P(L(G)) = P(L(G, Q_i \setminus Q_s))$.

Problem 5.2. Check, in a systematic way, if a given UWA $G = (Q, E, \alpha, \mu)$ is initial-state opaque with respect to a secret $Q_s \subseteq Q$.

The following example illustrates the notion of I-detectability and I-opacity.

Example 5.1. Consider the UWA G in Fig. 5.1 with the set of initial states $Q_i = \{1, 4\}$ and $\otimes = +$. We assume that the set of secret states is $Q_s = \{3, 4\}$, and u is the only unobservable label.

Detectability analysis: Initially, the system can be in state 1 and/or state 4. When weighted sequence $(b, 7)$ has been observed, there are two paths that are consistent with this observed weighted sequence, that is, $(1, u, 2)(2, b, 3)$ and $(4, u, 5)(5, u, 6)(6, b, 3)$. Both states 1 and 4 are then in $I((b, 7))$, and the same goes for the infinite weighted sequence $(b, 7)(c, 9)(c, 11) \cdots$. Therefore, G is not strongly I-detectable. However, we can observe that for weighted sequence $(d, 4)$ and infinite weighted sequence $(d, 4)(c, 6)(c, 8) \cdots$, the set of possible initial states, i.e., $I((d, 4)(c, 6)(c, 8) \cdots)$ contains only state 4 and we can conclude that G is weakly I-detectable.

Opacity analysis: Consider the secret initial state 4 and $\sigma = (u, 2)(d, 4) \in L(G, 4)$. No weighted sequence can be generated by G from non-secret initial state 1 such that its projection is equal to $P(\sigma)$. Therefore G is not initial-state opaque with respective to secret S. □

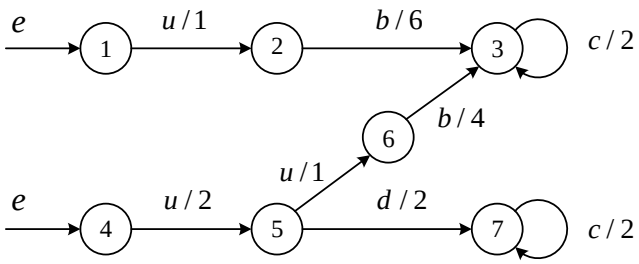

Figure 5.1 – An unambiguous weighted automaton G.

5.3 Verification of I-Detectability and I-Opacity

Inspired by the work in [Shu and Lin, 2013; Saboori and Hadjicostis, 2013b], we propose a formal procedure to solve Problems 5.1 and 5.2, that is, to check whether a UWA G is strongly (or weakly) I-detectable, and whether G is initial-state opaque. A two-step approach is first proposed to construct the initial-state estimator for G, called I-observer. Then, based on the constructed I-observer, necessary and sufficient conditions are proposed to check I-detectability and I-opacity of system G.

5.3.1 Construction of the I-Observer

First, we build the augmented UWA for G by extending its states to state pairs. Then, we construct the current-state estimator of the augmented automaton, which can be used as the initial-state estimator of G.

Construction of the Augmented Automaton

We extend UWA $G = (Q, E, \alpha, \mu)$ into its augmented version

$$G^{aug} = \left(Q^{aug}, E, t^{aug}, Q_i^{aug}, \varrho^{aug}\right)$$

where $Q^{aug} \subseteq Q \times Q_i$ is the set of states; $t^{aug} : Q^{aug} \times E \times Q^{aug} \to \mathcal{D}$ is the transition function; $Q_i^{aug} \subseteq Q^{aug}$ is the set of initial states; $\varrho^{aug} : Q_i^{aug} \to \mathcal{D}$ is the function of initial weights. It should be noted that each state $q^{aug} \in Q^{aug}$ is a pair (q_c, q_i), implying that state q_c can be reached in G from initial state q_i. In particular, the set of initial states of G^{aug} is defined as:

$$Q_i^{aug} = \{(q_i, q_i) \mid q_i \in Q_i\}.$$

For automaton $G = (Q, E, \alpha, \mu)$, we denote by $R(q, a)$ the set of reachable states for label $a \in E$ from state q, i.e., $R(q, a) = \left\{q' \in Q \mid \mu(a)_{qq'} \neq \varepsilon\right\}$. From a given state (q_c, q_i) in G^{aug}, the occurrence of a label $a \in E$ from state q_c in G is modeled in G^{aug} by transitions to states (q_c', q_i) with $q_c' \in R(q_c, a)$. That is, the set of reachable states for $a \in E$ from state (q_c, q_i) in G^{aug} is defined as:

$$R^{aug}((q_c, q_i), a) = \left\{(q_c', q_i) \mid q_c' \in R(q_c, a)\right\}.$$

Algorithm 5.1 details the process of constructing such an augmented automaton for G.

Steps 2–5 calculate the set of initial states Q_i^{aug} and specify the function of initial weights for G^{aug}. These initial states are first stored in the stack $Queue$. Then, for each pair (q_c, q_i)

Algorithm 5.1: Construction of augmented automaton G^{aug}

Require: A UWA $G = (Q, E, \alpha, \mu)$.
Ensure: An augmented automaton G^{aug} for G.
1: Let $Q_i^{aug} = \emptyset$, $Queue = \emptyset$
2: **for** each $q \in Q_i$ **do**
3: $\quad Q_i^{aug} = Q_i^{aug} \cup \{(q,q)\}$, $\varrho^{aug}((q,q)) = \alpha_q$
4: $\quad$ Enqueue (q,q) in $Queue$
5: **end for**
6: Let $Q^{aug} = Q_i^{aug}$
7: **while** $Queue \neq \emptyset$ **do**
8: $\quad$ Dequeue (q_c, q_i) from $Queue$
9: $\quad$ **for** each $a \in E$ **do**
10: $\quad\quad R^{aug}((q_c,q_i),a) = \{(q_c', q_i) \mid q_c' \in R(q_c,a)\}$
11: $\quad\quad$ **for** each $(q_c', q_i) \in R^{aug}((q_c,q_i),a)$ **do**
12: $\quad\quad\quad$ **if** $(q_c', q_i) \notin Q^{aug}$ **then**
13: $\quad\quad\quad\quad Q^{aug} = Q^{aug} \cup \{(q_c', q_i)\}$
14: $\quad\quad\quad\quad$ Enqueue (q_c', q_i) in $Queue$
15: $\quad\quad\quad$ **end if**
16: $\quad\quad\quad t^{aug}((q_c,q_i),a,(q_c',q_i)) = \mu(a)_{q_c q_c'}$
17: $\quad\quad$ **end for**
18: $\quad$ **end for**
19: **end while**
20: Return $G^{aug} = (Q^{aug}, E, t^{aug}, Q_i^{aug}, \varrho^{aug})$

belonging to $Queue$, we calculate all the reachable states for the occurrence of any label, i.e., $R^{aug}((q_c,q_i),a)$, for all $a \in E$. If an element $(q_c', q_i) \in R^{aug}((q_c,q_i),a)$ has not been defined, then it is added in Q^{aug} and in the stack. Besides, an arc labeled by $a/\mu(a)_{q_c q_c'}$ from node (q_c, q_i) to node (q_c', q_i) is also defined. Repeat this procedure until stack $Queue$ becomes empty, then the augmented automaton G^{aug} is constructed.

Remark 5.1. In the worst case, the number of states of G^{aug} is $|Q^{aug}| = |Q|^2$. As a result, the computational complexity of constructing the augmented automaton G^{aug} using Algorithm 5.1 is polynomial with respect to the number of states of G. □

Example 5.2. Consider again the automaton G in Fig. 5.1 with $\otimes = +$. After applying Algorithm 5.1, we obtain the augmented automaton of G shown in Fig. 5.2. □

The following lemma indicates that the augmented automaton retains the unambiguity property.

Lemma 5.2. The augmented automaton G^{aug} of G constructed by Algorithm 5.1 is unambiguous.

Proof. Assume that G^{aug} is not unambiguous. Then, there is more than one path labeled by a string, e.g. $e_1 e_2 \cdots e_i$, from the initial states leading to a state (q_c, q_i) in G^{aug} where

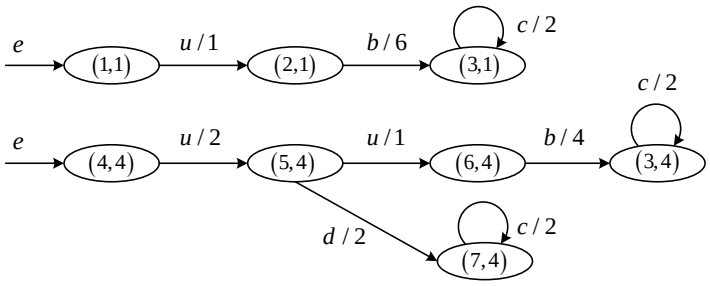

Figure 5.2 – Augmented automaton G^{aug} of G in Fig. 5.1.

$q_i \in Q_i$ and q_c is reachable from q_i. Using Algorithm 5.1, this situation occurs if there exists more than one path from the initial states to state q_c labeled by string $e_1 e_2 \cdots e_i$ in G. This contradicts the unambiguity of G. Therefore, G^{aug} is unambiguous. □

Lemma 5.3. The weighted language generated by the augmented automaton G^{aug} is equal to the weighted language generated by UWA G, i.e., $L(G^{aug}) = L(G)$.

Proof. We have to prove that for any weighted sequence $\sigma \in (E \times \mathbb{R})^*$, $\sigma \in L(G^{aug})$ if and only if $\sigma \in L(G)$.

First, consider a weighted sequence $\sigma = (e_1, \tau_1)(e_2, \tau_2) \cdots (e_n, \tau_n) \in L(G)$. Suppose that $\pi = (q_0, e_1, q_1)(q_1, e_2, q_2) \cdots (q_{k-1}, e_k, q_k)$ is a path from initial state q_0 such that the weighted sequence it generates is equal to σ. Since G is unambiguous and has null initial weights, we have $\tau_1 = \mu(e_1)_{q_0 q_1}$, $\tau_k = \tau_{k-1} \otimes \mu(e_k)_{q_{k-1} q_k}$ for $k = 2, 3, \cdots, n$. According to Algorithm 5.1, there must be the following path in G^{aug}:

$$\pi' = ((q_0, q_0), e_1, (q_1, q_0))((q_1, q_0), e_2, (q_2, q_0)) \cdots ((q_{k-1}, q_0), e_k, (q_k, q_0))$$

along which the weights of transitions are equal to the weights of the corresponding transitions in π. Since G^{aug} is unambiguous, $\sigma(\pi') = (e_1, \mu(e_1))(e_2, \mu(e_1) \otimes \mu(e_2)) \cdots (e_n, \mu(e_1) \otimes \cdots \otimes \mu(e_n)) = (e_1, \tau_1)(e_2, \tau_2) \cdots (e_n, \tau_n)$. That is, σ is generated by π', i.e., $\sigma \in L(G^{aug})$. Hence, $\forall \sigma \in L(G), \sigma \in L(G^{aug})$.

Second, for a sequence $\sigma = (e_1, \tau_1)(e_2, \tau_2) \cdots (e_n, \tau_n) \in L(G^{aug})$, we assume that it is generated by the following path

$$\pi = ((q_0, q_0), e_1, (q_1, q_0))((q_1, q_0), e_2, (q_2, q_0)) \cdots ((q_{n-1}, q_0), e_n, (q_n, q_0))$$

with (q_0, q_0) being an initial state of G^{aug}, and $\tau_1 = \mu(e_1)_{(q_0,q_0)(q_1,q_0)}, \ldots, \tau_k = \tau_{k-1} \otimes \mu(e_k)_{(q_{k-1},q_0)(q_k,q_0)}$ for $k = 2, 3, \cdots, n$. According to Algorithm 5.1, there must exist the path $\pi' = (q_0, e_1, q_1)(q_1, e_2, q_2) \cdots (q_{k-1}, e_k, q_k)$ in G with $q_0 \in Q_i$. Besides, the weights

of transitions in π' are $\mu(e_i)_{q_{i-1}q_i} = \mu(e_i)_{(q_{i-1},q_0)(q_i,q_0)}$, for $i = 1,2,\cdots,n$. Since G is unambiguous, we know that $\sigma(\pi') = (e_1,\tau_1)(e_2,\tau_2)\cdots(e_n,\tau_n)$. That is, σ is generated by π', i.e., $\sigma \in L(G)$. Therefore, for all $\sigma \in L(G^{aug})$, we have $\sigma \in L(G)$. □

I-observer

Let $G^{aug} = (Q^{aug}, E, \alpha^{aug}, \mu^{aug})$ be the equivalent representation of augmented automaton $G^{aug} = \left(Q^{aug}, E, t^{aug}, Q_i^{aug}, \varrho^{aug}\right)$ as defined in Def. 2.16.

We first adopt Algorithm 4.1 to construct the observer of $G^{aug} = (Q^{aug}, E, \alpha^{aug}, \mu^{aug})$, denoted by $G_{obs}^{aug} = (Q_{obs}^{aug}, E_{obs}^{aug}, \delta_{obs}^{aug}, q_{i,obs}^{aug})$. Then we prove that G_{obs}^{aug} can serve as the initial-state estimator, called I-observer, of the original system G.

Remark 5.2. Note that Algorithm 4.1 is presented in Subsection 4.3.1 for constructing the current-state estimator $G_{obs} = (Q_{obs}, E_{obs}, \delta_{obs}, q_{i,obs})$ of a given WA $G = (Q, E, \alpha, \mu)$ to deal with the verification of current-state detectability. □

The I-observer G_{obs}^{aug} is a DFA over a weighted alphabet $E_{obs}^{aug} \subseteq E_o \times \mathcal{D}$. That is, G_{obs}^{aug} has only one initial state, and from a given state no two transitions of G_{obs}^{aug} are labeled by the same weighted label $(a, t_a) \in E_{obs}^{aug}$. It should be noted that from a state in G_{obs}^{aug} there may exist several output transitions labeled by the same label $a \in E_o$ but with different weights t_a. In this case, the external agent can distinguish the transitions from the different associated weights.

Example 5.3. Consider the UWA G in Fig. 5.1 whose augmented automaton G^{aug} is visualized in Fig. 5.2. Its I-observer G_{obs}^{aug} computed by applying Algorithm 4.1 to G^{aug} is shown in Fig. 5.3. □

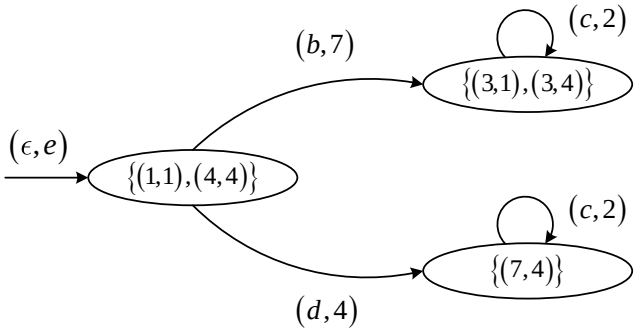

Figure 5.3 – I-observer G_{obs}^{aug} of the automaton G in Fig. 5.1.

Let $(E_{obs}^{aug})^*$ be the set of all the finite strings over weighted alphabet E_{obs}^{aug} including (ϵ, e) corresponding to empty string ϵ and identity weight. The language generated by the I-observer $G_{obs}^{aug} = (Q_{obs}^{aug}, E_{obs}^{aug}, \delta_{obs}^{aug}, q_{i,obs}^{aug})$ is defined as:

$$L(G_{obs}^{aug}) = \{\omega \in (E_{obs}^{aug})^* \mid \exists q_{obs}^{aug} \in Q_{obs}^{aug} : \delta_{obs}^{aug}(q_{i,obs}^{aug}, \omega) = q_{obs}^{aug}\}.$$

Note that the language generated by G_{obs}^{aug} is a subset of $(E_{obs}^{aug})^*$, i.e., $L(G_{obs}^{aug}) \subseteq (E_{obs}^{aug})^*$. While the language generated by $G = (Q, E, \alpha, \mu)$ is a subset of $(E \times \mathcal{D})^*$, i.e., $L(G) \subseteq (E \times \mathcal{D})^*$. For any sequence $\omega = (a_1, \tau_1)(a_2, \tau_2) \cdots (a_n, \tau_n) \in L(G_{obs}^{aug})$, its equivalent notation in $(E \times \mathcal{D})^*$ can be defined by $\omega^{cum} = (a_1, \tau_1)(a_2, \tau_2') \cdots (a_n, \tau_n')$ where $\tau_k' = \tau_1 \otimes \cdots \otimes \tau_k$, $k = 2, 3, \cdots, n$, represents the cumulated weight for sequence $a_1 a_2 \cdots a_k$. We denote by $L^{cum}(G_{obs}^{aug})$ the equivalent notation of $L(G_{obs}^{aug})$, that is,

$$L^{cum}(G_{obs}^{aug}) = \{\sigma \in (E \times \mathcal{D})^* \mid \exists \omega \in L(G_{obs}^{aug}) : \omega^{cum} = \sigma\}.$$

The following lemma states the relationship between the observed language generated by G and the language generated by I-observer G_{obs}^{aug}.

Lemma 5.4. The projection of language $L(G)$ generated by G coincides with $L^{cum}(G_{obs}^{aug})$, that is, $P(L(G)) = L^{cum}(G_{obs}^{aug})$.

Proof. Lemma 5.3 states that $L(G^{aug}) = L(G)$. Besides, by Lemma 4.1 [1], we have $P(L(G^{aug})) = L^{cum}(G_{obs}^{aug})$. Therefore $P(L(G)) = L^{cum}(G_{obs}^{aug})$ holds. □

Lemma 5.5. The possible initial states of UWA G after observing weighted sequence $\sigma_o = (a_1, \tau_1)(a_2, \tau_2) \cdots (a_k, \tau_k) \in P(L(G))$ can be calculated by

$$I(\sigma_o) = \{q_i \in Q_i \mid \exists q_c \in Q : (q_c, q_i) \in \delta_{obs}^{aug}(q_{i,obs}^{aug}, (a_1, \tau_1)(a_2, \tau_2 \otimes^{-1} \tau_1) \cdots (a_k, \tau_k \otimes^{-1} \tau_{k-1}))\}$$

where $\tau_k \otimes^{-1} \tau_{k-1}$ is defined as the value $x \in \mathcal{D}$ such that $x \otimes \tau_{k-1} = \tau_k$, with τ_k and $\tau_{k-1} \in \mathcal{D} \setminus \{\varepsilon\}$.

Proof. According to Algorithm 5.1, for all $q_i \in Q_i$, $q_c \in Q$ and $\sigma \in L(G)$, we have

$$q_i \overset{\sigma}{\rightsquigarrow} q_c \Leftrightarrow (q_i, q_i) \overset{\sigma}{\rightsquigarrow} (q_c, q_i).$$

1. The projection of language $L(G)$ generated by G coincides with $L^{cum}(G_{obs})$, i.e., $P(L(G)) = L^{cum}(G_{obs})$, where $L^{cum}(G_{obs}) = \{\sigma \in (E \times \mathcal{D})^* \mid \exists \omega \in L(G_{obs}) : \omega^{cum} = \sigma\}$.

Therefore, for any $\sigma_o = (a_1, \tau_1)(a_2, \tau_2)\cdots(a_k, \tau_k) \in P(L(G))$, we have

$$
\begin{aligned}
&I(\sigma_o)\\
&= \{q_i \in Q_i \mid \exists q_c \in Q, \exists \sigma \in L(G) : P(\sigma) = \sigma_o, q_i \overset{\sigma}{\rightsquigarrow} q_c\}\\
&\quad \textit{(by the definition of } I(\sigma_o) \textit{ in Eq. (5.1))}\\
&= \{q_i \in Q_i \mid \exists q_c \in Q, \exists \sigma \in L(G) : P(\sigma) = \sigma_o, q_i \overset{\sigma}{\rightsquigarrow} q_c,\\
&\qquad\qquad \mathbb{E}_f(\sigma) = \mathbb{E}_f(\sigma_o)\}\\
&\quad \textit{(by Lemma 5.1)}\\
&= \{q_i \in Q_i \mid \exists q_c \in Q, \exists \sigma \in L(G^{aug}) :\\
&\qquad P(\sigma) = \sigma_o, (q_i, q_i) \overset{\sigma}{\rightsquigarrow} (q_c, q_i), \mathbb{E}_f(\sigma) = \mathbb{E}_f(\sigma_o)\}\\
&\quad \textit{(by Algorithm 5.1)}\\
&= \{q_i \in Q_i \mid \exists q_c \in Q : (q_c, q_i) \in\\
&\delta^{aug}_{obs}(q^{aug}_{i,obs}, (a_1, \tau_1)(a_2, \tau_2 \otimes^{-1} \tau_1)\cdots(a_k, \tau_k \otimes^{-1} \tau_{k-1}))\}\\
&\quad \textit{(by Proposition 4.1 [2])}
\end{aligned}
$$

Therefore, Lemma 5.5 is established. □

5.3.2 Criteria for Verifying I-Detectability

This subsection focuses on Problem 5.1, that is, the verification of I-detectability. I-detectability problem can be rephrased as follows. If no observable label has been observed, we consider that all the states in Q_i are possible initial states. Afterwards, from the successive observations, we may be able to restrict the set of possible initial states. The I-detectability problem is to determine if this set can shrink to a singleton after a finite number of observations, and then, to identify the only possible initial state. Necessary and sufficient conditions for checking strong I-detectability and weak I-detectability for UWA G based on its I-observer G^{aug}_{obs} are presented.

We denote by S_{ci} the set of all elementary circuits of G^{aug}_{obs} as:

$$
\begin{aligned}
S_{ci} = \{(q^{aug}_{obs}, s) \in Q^{aug}_{obs} \times (E^{aug}_{obs})^* \mid \delta^{aug}_{obs}(q^{aug}_{obs}, s) = q^{aug}_{obs}\\
\wedge |s| \geq 1 \wedge (\forall s' \in Pr(s) \text{ s.t. } s' \neq s : \delta^{aug}_{obs}(q^{aug}_{obs}, s') \neq q^{aug}_{obs})\}.
\end{aligned}
$$

2. The set of possible current states of G after observing the weighted sequence $(a_1, \tau_1)(a_2, \tau_2)\cdots(a_k, \tau_k)$ is given by $\delta_{obs}(q_{i,obs}, (a_1, \tau_1)(a_2, \tau_2 \otimes^{-1} \tau_1)\cdots(a_k, \tau_k \otimes^{-1} \tau_{k-1}))$, where δ_{obs} is the transition function of the current-state estimator $G_{obs} = (Q_{obs}, E_{obs}, \delta_{obs}, q_{i,obs})$ of UWA $G = (Q, E, \alpha, \mu)$.

Let $Q^{aug}_{I,obs}$ be a subset of Q^{aug}_{obs} composed of states with the same second element (initial state of G), that is,

$$Q^{aug}_{I,obs} = \{q^{aug}_{obs} \in Q^{aug}_{obs} \mid \exists q \in Q_i, \forall (q_c, q_i) \in q^{aug}_{obs} : q_i = q\}.$$

Remark 5.3. Since G is deadlock free and I-observer G^{aug}_{obs} is finite, there must exist at least one circuit in G^{aug}_{obs}. □

Remark 5.4. If an observed weighted sequence from G leads to a state in $Q^{aug}_{I,obs}$, then the set of possible initial states is a singleton and this remains true for any continuation of the observed weighted sequence. Therefore, if there are circuits in G^{aug}_{obs}, then all nodes of a circuit belong to either $Q^{aug}_{I,obs}$ or $Q^{aug}_{obs} \setminus Q^{aug}_{I,obs}$. □

Theorem 5.1 (Criterion for Checking Strong I-Detectability)**.** A UWA G is strongly I-detectable with respect to projection P iff $(\forall (q^{aug}_{obs}, s) \in S_{ci})(\forall \omega \in Pr(s))\delta^{aug}_{obs}(q^{aug}_{obs}, \omega) \in Q^{aug}_{I,obs}$, that is, all the elementary circuits in G^{aug}_{obs} have their nodes in $Q^{aug}_{I,obs}$.

Proof. (If) If $(\forall (q^{aug}_{obs}, s) \in S_{ci})(\forall \omega \in Pr(s))\delta^{aug}_{obs}(q^{aug}_{obs}, \omega) \in Q^{aug}_{I,obs}$, then the system will eventually reach a state in $Q^{aug}_{I,obs}$ after a finite number of observations for all possible trajectories of the system. According to Remark 5.4, the set of possible initial states is a singleton for all possible continuations, that is, the system is strongly I-detectable.

(Only If) Assume $(\exists (q^{aug}_{obs}, s) \in S_{ci})(\forall \omega \in Pr(s))\delta^{aug}_{obs}(q^{aug}_{obs}, \omega) \in Q^{aug}_{obs} \setminus Q^{aug}_{I,obs}$, i.e., there exist elementary circuits in G^{aug}_{obs} whose nodes all belong to $Q^{aug}_{obs} \setminus Q^{aug}_{I,obs}$. Then a possible evolution of the system can iterate indefinitely such a circuit. That is, we cannot determine the initial state of G for some trajectories forever. Hence, the system is not strongly I-detectable. □

Theorem 5.2 (Criterion for Checking Weak I-Detectability)**.** A UWA G is weakly I-detectable with respect to P iff $Q^{aug}_{I,obs} \neq \emptyset$.

Proof. (If) If $Q^{aug}_{I,obs}$ is not empty, then the system can reach a state in $Q^{aug}_{I,obs}$ after a finite number of observations for some trajectories of the system. As pointed out in Remark 5.4, we can determine the only possible initial state of the system G for trajectories corresponding to continuations of these observations. Hence, the system is weakly I-detectable.

(Only If) If $Q^{aug}_{I,obs} = \emptyset$, then the set of possible initial states does not shrink to a singleton whatever is the trajectory. Hence, the system is not weakly I-detectable. □

Example 5.4. Consider the I-observer G^{aug}_{obs} in Fig. 5.3 of UWA G in Fig. 5.1. There are three states in G^{aug}_{obs}, i.e., $\{(1,1),(4,4)\}$, $\{(3,1),(3,4)\}$, $\{(7,4)\}$, and we have $Q^{aug}_{I,obs} = \{\{(7,4)\}\}$.

By Theorem 5.1, G is weakly I-detectable since $Q^{aug}_{I,obs}$ is not empty. On the other hand, G is not strongly I-detectable since not all elementary circuits have all their nodes within $Q^{aug}_{I,obs}$. For instance, elementary circuit $\{(3,1),(3,4)\} \xrightarrow{(c,2)} \{(3,1),(3,4)\}$ has its node belongs to $Q^{aug}_{obs} \setminus Q^{aug}_{I,obs}$. □

Example 5.5. Consider the UWA G in Fig. 5.4 with $\otimes = +$. Its augmented automaton G^{aug} computed by Algorithm 5.1 is shown in Fig. 5.5. After applying Algorithm 4.1 to G^{aug}, we obtain the I-observer G^{aug}_{obs} shown in Fig. 5.6. It can be checked that all elementary circuits have all their nodes in $Q^{aug}_{I,obs}$. Therefore, by Theorem 5.1, G is strongly I-detectable (hence weakly I-detectable). □

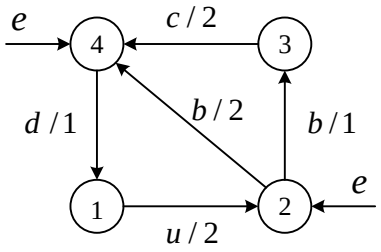

Figure 5.4 – An unambiguous automaton G

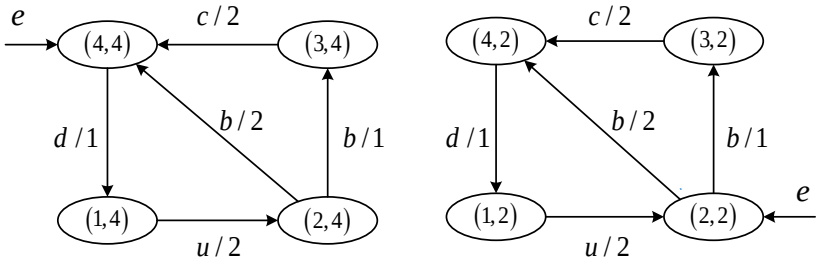

Figure 5.5 – Augmented automaton G^{aug} of G in Fig. 5.4.

5.3.3 Criteria for Verifying I-Opacity

In this section, we focus on Problem 5.2, that is, the verification of I-opacity for a UWA $G = (Q, E, \alpha, \mu)$. The secret, denoted by Q_s, is defined as an arbitrary subset of Q, i.e., $Q_s \subseteq Q$. The set of initial states Q_i of G is divided into two disjoint parts by the secret Q_s: $Q_i \cap Q_s$ and $Q_i \setminus Q_s$.

Lemma 5.6. Given a UWA $G = (Q, E, \alpha, \mu)$, projection $P : E^* \to E_o^*$, and a set of secret states $Q_s \subseteq Q$, G is initial-state opaque with respect to Q_s and P iff for all $\sigma_o \in P(L(G))$, $I(\sigma_o) \nsubseteq Q_s$ holds, where $I(\sigma_o)$ is the set of possible initial states for the observation σ_o.

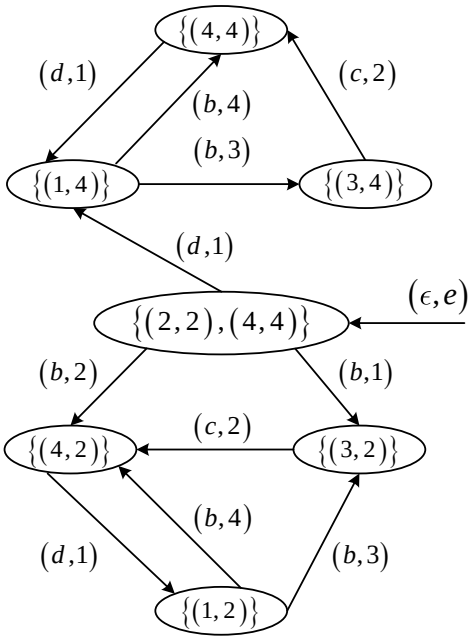

Figure 5.6 – I-observer G_{obs}^{aug} of G in Fig. 5.4.

Proof.

$$
\begin{aligned}
&(\forall \sigma_o \in P(L(G)))I(\sigma_o) \not\subseteq Q_s \\
&\Leftrightarrow (\forall \sigma_o \in P(L(G)))(\exists q' \in I(\sigma_o))q' \notin Q_s \\
&\Leftrightarrow (\forall \sigma_o \in P(L(G)))(\exists q' \in Q_i \setminus Q_s)(\exists q \in Q) \\
&\qquad\qquad (\exists \sigma' \in L(G))P(\sigma') = \sigma_o \wedge q' \overset{\sigma'}{\rightsquigarrow} q \\
&\quad \textit{(by the definition of } I(\sigma_o) \textit{ in Eq. (5.1))} \\
&\Leftrightarrow (\forall \sigma_o \in P(L(G)))(\exists q' \in Q_i \setminus Q_s)(\exists \sigma' \in L(G, q')) \\
&\qquad\qquad P(\sigma') = \sigma_o \\
&\quad \textit{(according to Eq. (5.3))} \\
&\Leftrightarrow (\forall \sigma_o \in P(L(G, Q_i \cap Q_s)))(\exists q' \in Q_i \setminus Q_s) \\
&\qquad\qquad (\exists \sigma' \in L(G, q'))P(\sigma') = \sigma_o \\
&(\textit{by } P(L(G)) = P(L(G, Q_i \cap Q_s)) \cup P(L(G, Q_i \setminus Q_s))\ ^{3}) \\
&\Leftrightarrow (\forall q \in Q_i \cap Q_s)(\forall \sigma \in L(G, q))(\exists q' \in Q_i \setminus Q_s) \\
&\qquad\qquad (\exists \sigma' \in L(G, q'))P(\sigma') = P(\sigma)
\end{aligned}
$$

which is consistent with Definition 5.4 of I-opacity. Hence the proof is completed. □

3. More precisely, ($\Rightarrow$) holds true simply because $P(L(G, Q_i \cap Q_s)) \subseteq P(L(G))$. In order to prove the correctness of ($\Leftarrow$), we have to prove that $\forall \sigma_o \in P(L(G, Q_i \setminus Q_s)), \exists q' \in Q_i \setminus Q_s, \exists \sigma' \in L(G, q') : P(\sigma') = \sigma_o$, which is self-evident.

For any $q_{obs}^{aug} \in Q_{obs}^{aug}$, we define $I(q_{obs}^{aug}) = \{q_i \in Q_i \mid \exists (q_c, q_i) \in q_{obs}^{aug}\}$ as the initial states of automaton G that it contains. The following theorem illustrates that the I-observer G_{obs}^{aug} can be used to verify the I-opacity for UWA G.

Theorem 5.3. Given a UWA $G = (Q, E, \alpha, \mu)$, projection $P : E^* \to E_o^*$, and a set of secret states $Q_s \subseteq Q$, G is initial-state opaque with respect to Q_s and P iff

$$(\forall q_{obs}^{aug} \in Q_{obs}^{aug}) I(q_{obs}^{aug}) \nsubseteq Q_s$$

where Q_{obs}^{aug} is the set of states in I-observer G_{obs}^{aug} of G.

Proof. It follows from Lemmas 5.5 and 5.6. □

Example 5.6. Consider the I-observer G_{obs}^{aug} in Fig. 5.3 of UWA G in Fig. 5.1. There are three states in G_{obs}^{aug}, i.e., $\{(1,1),(4,4)\}$, $\{(3,1),(3,4)\}$ and $\{(7,4)\}$. Assume that the secret is $Q_s = \{3,4\}$. Since $I(\{(7,4)\}) = 4 \subseteq Q_s$, according to Theorem 5.3, G is not initial-state opaque with respect to Q_s and P. If now the secret is $Q_s = \{1,3\}$, then G is initial-state opaque since $I(\{(1,1),(4,4)\}) = \{1,4\} \nsubseteq Q_s$, $I(\{(3,1),(3,4)\}) = \{1,4\} \nsubseteq Q_s$ and $I(\{(7,4)\}) = 4 \nsubseteq Q_s$. □

Example 5.7. Consider the I-observer G_{obs}^{aug} in Fig. 5.6 of UWA G in Fig. 5.4 where $Q_i = \{2,4\}$. It can be checked that G is always non-opaque with respect to P and an arbitrary secret Q_s so long as $Q_s \cap Q_i \neq \emptyset$. In fact, among the states of G_{obs}^{aug}, we have $I(\{1,4\}) = I(\{4,4\}) = I(\{3,4\}) = \{4\}$ and $I(\{4,2\}) = I(\{3,2\}) = I(\{1,2\}) = \{2\}$. Therefore, for a secret Q_s containing initial state 2 and/or 4, there must exist one state $q_{obs}^{aug} \in Q_{obs}^{aug}$ such that $I(q_{obs}^{aug}) \subseteq Q_s$. □

Remark 5.5. Note that the UWA whose transition weights are associated with a time interpretation are strongly related to real-time automata. Unlike in [Wang *et al.*, 2018] where the verification of I-opacity in real-time automata has been investigated by solving the inclusion problem of regular languages, in this work, the I-opacity problem of a UWA is solved by the construction of its initial-state estimator. □

5.3.4 Computational Complexity Analysis

Given a UWA $G = (Q, E, \alpha, \mu)$, in the worst case, the number of states of its augmented automaton G^{aug} obtained by using Algorithm 5.1 is $|Q^{aug}| = |Q|^2$. As a result, the computational complexity of constructing G^{aug} using Algorithm 5.1 is polynomial with respect to the number of states of G.

Checking I-detectability and I-opacity for UWA G using Theorems 1-3 requires the construction of an I-observer. Since the augmented automaton has at most $|Q|^2$ states, where $|Q|$ is the number of states of G, in the worst case, I-observer G_{obs}^{aug} has $2^{|Q|^2} - 1$ states. Therefore, verifying the I-detectability and I-opacity using G_{obs}^{aug} is of exponential complexity.

5.4 Conclusion

This paper addresses the verification of I-detectability and I-opacity for UWAs under partial event observation. Given a UWA G, an algorithm is first proposed to construct its augmented version G^{aug}. It should be noted that each state q^{aug} in G^{aug} is a pair (q_c, q_i) of states from G, implying that state q_c can be reached from initial state q_i in system G. Then the initial-state estimator, I-observer, of G is obtained by constructing the current state estimator of the augmented automaton G^{aug}. Finally, necessary and sufficient conditions are derived to verify strong I-detectability, weak I-detectability and I-opacity based on the constructed I-observer.

The work in this chapter has been submitted as follows:

A. Lai, S. Lahaye and Z. Li. "Initial-state detectability and initial-state opacity of unambiguous weighted automata", *Automatica* (submitted), 2019 [Lai *et al.*, 2019d].

6

Conclusion and Future Work

In this chapter, we make some conclusions on the main work of this thesis and discuss the possible directions for our future research.

6.1 Conclusion

State estimation and property verification are important problems in systems and control theory. These problems have been widely investigated for classical automata and Petri nets (PNs), two important models of discrete event systems (DESs). In addition to automata and PNs, weighted automata (WAs) can be considered as another class of DES models, which are a quantitative extension of nondeterministic finite automata (NFAs). WAs found much interest in Computer Science due to their importance both in theory as well as in current practical applications, e.g., natural language processing, speech recognition and digital image compression. From a different perspective, in this thesis, we use WAs as DES models to deal with state estimation, fault diagnosis, detectability and opacity verification problems, which are usually studied by researchers from the control community. The main work of this thesis is briefly described below.

- We formally define the state estimation problem for WAs. Given a WA where only partial labels are observable, state estimation aims at calculating all possible states the system can be in according to the observation. Algorithms are proposed in Chapter

3 for tackling this problem for WAs. The proposed approach is based on the analysis of state vector, and all paths consistent with the observation are enumerated so as to capture the possible current states.

- We deal with the fault diagnosis problem of WAs where failures are associated with partial unobservable labels. For a given WA, we first construct its augmented automaton with respect to a fault class, in which states are partitioned as fault states and non-fault states. Then the state estimation technique can be used to determine whether some fault labels have occurred or not given based on the observation.

- The verification of current-state detectability is investigated for unambiguous weighted automata (UWAs). A WA is said to be unambiguous if for any state and any string, there is at most one path labeled by this string leading from an initial state to that state. We define strong detectabilities (i.e., strong detectability and strong periodic detectability) and weak detectabilities (i.e., weak detectability and weak periodic detectability) for UWAs. Current-state detectability requires that the current state of the system can always be determined uniquely/unambiguously within a finite delay. An observer-based approach whose computational complexity is exponential with respect to the number of states of the studied UWA is proposed to verify the strong detectabilities and weak detectabilities. Besides, we find that the complexity of verification of strong detectabilities can be reduced by introducing the tool of detector whose size is polynomial with respect to the state cardinality of the original UWA.

- We study the verification of initial-state detectability (I-detectability) and initial-state opacity (I-opacity) of UWAs. Strong I-detectability and weak I-detectability, I-opacity are defined for UWAs. I-detectability requires that the initial state of the system can always be determined uniquely/unambiguously within a finite delay. I-opacity requires that the intruder can never be sure that the system is initially from a secret state for all trajectories of the system. Note that the secret in this thesis is modeled as a set of secret states. Algorithms are proposed to build the I-observer of a UWA so as to derive necessary and sufficient conditions for verifying strong I-detectability, weak I-detectability and I-opacity.

6.2 Future Work

In this section, we point out several possible directions for future research related to the work in this thesis.

- In this thesis, a fault is defined as "the occurrence of a faulty label", the same as in logical models. Maybe we can consider different types of faults for WAs because they are no longer logical models. For example, if we assume that a fault is "the occurrence of a transition with a delay", can we detect this type of fault? On the other hand, we will consider the problem of diagnosability of fault using WAs. The diagnosability characterizes whether one can always determine the occurrence of a fault within a limited number of delays, e.g., a finite number of label observations and a finite number of time units.
- All work on verifying current-state detectability and I-detectability as well as I-opacity of WAs is done with the *unambiguity* assumption. This assumption greatly limits the contribution of our research. As a future work, we plan to relax this assumption when dealing with these problems.
- In addition to state-based opacity, language-based opacity is another important class of opacity properties, where the secret is defined as a language, i.e., a set of label sequences. A system is said to be language opaque if the intruder can never infer if the evolution of the system belongs to the secret. Our future work will focus on current-state opacity and language opacity in the framework of WAs.
- The opacity verification problem is to determine whether a system is opaque with respect to a given secret or not. Another issue related to opacity verification is the problem of opacity enforcement. Given a system that is not opaque, the opacity enforcement problem is to make the system opaque. Our future work is of exploring techniques to deal with I-opacity enforcement and language-based opacity enforcement for UWAs. As in automata and PNs, such techniques may rely on supervisory control, dynamically changing the observability of labels, and inserting additional labels into the output behavior of the system.

References

[Allauzen *et al.*, 2011] Cyril Allauzen, Mehryar Mohri, and Ashish Rastogi. General algorithms for testing the ambiguity of finite automata and the double-tape ambiguity of finite-state transducers. *International Journal of Foundations of Computer Science*, 22(04):883–904, 2011. 39

[Alur and Dill, 1994] R. Alur and D. L. Dill. A theory of timed automata. *Theoretical computer science*, 126(2):183–235, 1994.

[Athanasopoulou and Hadjicostis, 2005] Eleftheria Athanasopoulou and Christoforos N Hadjicostis. Probabilistic approaches to fault detection in networked discrete event systems. *IEEE transactions on neural networks*, 16(5):1042–1052, 2005. 22

[Baccelli *et al.*, Wiley 1992] François Baccelli, Guy Cohen, Geert Jan Olsder, and Jean-Pierre Quadrat. Synchronization and linearity: an algebra for discrete event systems. Wiley, 1992. 34

[Badouel *et al.*, 2007] Eric Badouel, Marek Bednarczyk, Andrzej Borzyszkowski, Benoît Caillaud, and Philippe Darondeau. Concurrent secrets. *Discrete Event Dynamic Systems*, 17(4):425–446, 2007. 25

[Basile *et al.*, 2009] Francesco Basile, Pasquale Chiacchio, and Gianmaria De Tommasi. An efficient approach for online diagnosis of discrete event systems. *IEEE Transactions on Automatic Control*, 54(4):748–759, 2009. 22, 57

[Basile *et al.*, 2012] Francesco Basile, Pasquale Chiacchio, and Gianmaria De Tommasi. On k-diagnosability of Petri nets via integer linear programming. *Automatica*, 48(9):2047–2058, 2012. 22

[Basile *et al.*, 2013] F. Basile, M. P. Cabasino, and C. Seatzu. Marking estimation of time Petri nets with unobservable transitions. In *Proceedings of 18th IEEE Conference on Emerging Technologies & Factory Automation, Cagliari, Italy*, pages 1–7, 2013. 18

[Basile *et al.*, 2015] Francesco Basile, Maria Paola Cabasino, and Carla Seatzu. State estimation and fault diagnosis of labeled time petri net systems with unobservable transitions. *IEEE Transactions on Automatic Control*, 60(4):997–1009, 2015. 18, 22, 58, 59

[Béal *et al.*, 2008] Marie-Pierre Béal, Eugen Czeizler, Jarkko Kari, and Dominique Perrin. Unambiguous automata. *Mathematics in Computer Science*, 1(4):625–638, 2008. 38

[Benveniste *et al.*, 2003] Albert Benveniste, Eric Fabre, Stefan Haar, and Claude Jard. Diagnosis of asynchronous discrete-event systems: a net unfolding approach. *IEEE Transactions on Automatic Control*, 48(5):714–727, 2003. 23

[Bérard *et al.*, 2015] Béatrice Bérard, Krishnendu Chatterjee, and Nathalie Sznajder. Probabilistic opacity for markov decision processes. *Information Processing Letters*, 115(1):52–59, 2015. 25

[Bonhomme, 2013] Patrice Bonhomme. State observer synthesis of real-time systems modeled by P-time petri nets. In *Proceedings of 18th IEEE Conference on Emerging Technologies & Factory Automation, Cagliari, Italy*, pages 1–8, 2013. 18

[Bonhomme, 2015] P. Bonhomme. Marking estimation of P-time Petri nets with unobservable transitions. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 45:508–518, 2015. 18

[Boukra *et al.*, 2015] Rabah Boukra, Sébastien Lahaye, and Jean-Louis Boimond. New representations for (max,+) automata with applications to performance evaluation and control of discrete event systems. *Discrete Event Dynamic Systems*, 25(1-2):295–322, 2015.

[Bouyer *et al.*, 2005] Patricia Bouyer, Fabrice Chevalier, and Deepak DSouza. Fault diagnosis using timed automata. In *International Conference on Foundations of Software Science and Computation Structures*, pages 219–233. Springer, 2005. 21, 58

[Bryans *et al.*, 2005] Jeremy W Bryans, Maciej Koutny, and Peter YA Ryan. Modelling opacity using Petri nets. *Electronic Notes in Theoretical Computer Science*, 121:101–115, 2005. 24

[Bryans *et al.*, 2008] Jeremy W Bryans, Maciej Koutny, Laurent Mazaré, and Peter YA Ryan. Opacity generalised to transition systems. *International Journal of Information Security*, 7(6):421–435, 2008. 24

[Buchholz, 2008] Peter Buchholz. Bisimulation relations for weighted automata. *Theoretical Computer Science*, 393(1-3):109–123, 2008. 45

[Cabasino *et al.*, 2010] Maria Paola Cabasino, Alessandro Giua, and Carla Seatzu. Fault detection for discrete event systems using Petri nets with unobservable transitions. *Automatica*, 46(9):1531–1539, 2010. 22, 58, 59

[Cabasino *et al.*, 2011] M. P. Cabasino, A. Giua, M. Pocci, and C. Seatzu. Discrete event diagnosis using labeled Petri nets. An application to manufacturing systems. *Control Engineering Practice*, 19:989–1001, 2011. 18, 22, 51

[Cabasino *et al.*, 2014] Maria Paola Cabasino, Alessandro Giua, and Carla Seatzu. Diagnosability of discrete-event systems using labeled Petri nets. *IEEE Transactions on Automation Science and Engineering*, 11(1):144–153, 2014. 23

[Caines and Wang, 1989] Peter E Caines and Suning Wang. Classical and logic based regulator design and its complexity for partially observed automata. In *Proceedings of the 28th IEEE Conference on Decision and Control,*, pages 132–137. IEEE, 1989. 17

[Cassandras and Lafortune, 2009] Christos G Cassandras and Stephane Lafortune. *Introduction to discrete event systems*. Springer Science & Business Media, 2009. 29, 30

[Cassez, 2009a] Franck Cassez. The dark side of timed opacity. *Advances in Information Security and Assurance*, pages 21–30, 2009. 24

[Cassez, 2009b] Franck Cassez. A note on fault diagnosis algorithms. In *Proceedings of 48th IEEE Conference on Decision and Control and 28th Chinese Control Conference, Shanghai, China*, pages 6941–6946, 2009.

[Cassez, 2010] Franck Cassez. Dynamic observers for fault diagnosis of timed systems. In *Proceedings of 49th IEEE Conference on Decision and Control, Atlanta, Georgia USA*, pages 4359–4364. IEEE, 2010.

[Chen and Provan, 1997] Yi-Liang Chen and Gregory Provan. Modeling and diagnosis of timed discrete event systems–A factory automation example. In *Proceedings of American Control Conference, Albuquerque, NM*, volume 1, pages 31–36, 1997. 21

[Cong *et al.*, 2017] Xuya Cong, Maria Pia Fanti, Agostino Marcello Mangini, and Zhiwu Li. Decentralized diagnosis by Petri nets and integer linear programming. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 48(10):1689–1700, 2017. 23

[Cong *et al.*, 2018] Xuya Cong, Maria Pia Fanti, Agostino Marcello Mangini, and Zhiwu Li. On-line verification of current-state opacity by petri nets and integer linear programming. *Automatica*, 94:205–213, 2018. 23

[Corona *et al.*, 2007] D. Corona, A. Giua, and C. Seatzu. Marking estimation of Petri nets with silent transitions. *IEEE Transactions on Automatic Control*, 52:1695–1699, 2007. 18

[Declerck and Bonhomme, 2014] P. Declerck and P. Bonhomme. State estimation of timed labeled Petri nets with unobservable transitions. *IEEE Transactions on Automation Science and Engineering*, 11:103–110, 2014. 18

[Dotoli *et al.*, 2009] Mariagrazia Dotoli, Maria Pia Fanti, Agostino Marcello Mangini, and Walter Ukovich. On-line fault detection in discrete event systems by petri nets and integer linear programming. *Automatica*, 45(11):2665–2672, 2009. 23, 57

[Droste and Gastin, 2007] Manfred Droste and Paul Gastin. Weighted automata and weighted logics. *Theoretical Computer Science*, 380(1-2):69–86, 2007. 45

[Droste *et al.*, 2009] Manfred Droste, Werner Kuich, and Heiko Vogler. *Handbook of weighted automata*. Springer Science & Business Media, 2009. 25, 29, 34, 36, 45

[Fabre and Jezequel, 2010] Eric Fabre and Loïg Jezequel. On the construction of probabilistic diagnosers. *IFAC Proceedings Volumes*, 43(12):229–234, 2010. 22

[Gaubert and Mairesse, 1999] S. Gaubert and J. Mairesse. Modeling and analysis of timed petri nets using heaps of pieces. *IEEE Transactions on Automatic Control*, 44:683–697, 1999.

[Gaubert, 1995] S. Gaubert. Performance evaluation of (max,+) automata. *IEEE transactions on automatic Control*, 40:2014–2025, 1995.

[Ghazel *et al.*, 2009] Mohamed Ghazel, Armand Toguyéni, and Pascal Yim. State observer for DES under partial observation with time Petri nets. *Discrete Event Dynamic Systems*, 19(2):137–165, 2009. 20

[Giua and Seatzu, 2002] A. Giua and C. Seatzu. Observability of place/transition nets. *IEEE Transactions on Automatic Control*, 47:1424–1437, 2002. 20

[Giua and Seatzu, 2014] A. Giua and C. Seatzu. A survey on state estimation using Petri nets. In *Proceedings of 2014 European Control Conference, Strasbourg, France*, pages 2630–2635, 2014. 18

[Giua *et al.*, 2005] A. Giua, D. Corona, and C. Seatzu. State estimation of λ-free labeled Petri nets with contact-free nondeterministic transitions. *Discrete Event Dynamic Systems*, 15:85–108, 2005. 18

[Giua, 2011] A. Giua. State estimation and fault detection using Petri nets. In *Proceedings of 32nd International Conference on Application and Theory of Petri Nets, Newcastle, UK*, pages 38–48, 2011. 17

[Gondran and Minoux, 2008] Michel Gondran and Michel Minoux. *Graphs, dioids and semirings: new models and algorithms*, volume 41. Springer Science & Business Media, 2008. 34

[Hadj-Alouane *et al.*, 2005a] N Ben Hadj-Alouane, Stéphane Lafrance, Feng Lin, John Mullins, and Mohamed Moez Yeddes. On the verification of intransitive noninterference in mulitlevel security. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 35(5):948–958, 2005. 25

[Hadj-Alouane *et al.*, 2005b] Nejib Ben Hadj-Alouane, Stéphane Lafrance, Feng Lin, John Mullins, and Moez Yeddes. Characterizing intransitive noninterference for 3-domain security policies with observability. *IEEE transactions on automatic control*, 50(6):920–925, 2005. 25

[Hardouin *et al.*, 2010] L. Hardouin, C. A. Maia, B. Cottenceau, and M. Lhommeau. Observer design for (max,+) linear systems. *IEEE Transactions on Automatic Control*, 55:538–543, 2010. 18

[Hardouin *et al.*, 2017] L. Hardouin, Y. Shang, C. A. Maia, and B. Cottenceau. Observer-based controllers for max-plus linear systems. *IEEE Transactions on Automatic Control*, 62:2153–2165, 2017.

[Hardouin *et al.*, 2018] Laurent Hardouin, Bertrand Cottenceau, Ying Shang, Jörg Raisch, et al. Control and state estimation for max-plus linear systems. *Foundations and Trends® in Systems and Control*, 6(1):1–116, 2018. 34

[Heidergott *et al.*, 2014] Bernd Heidergott, Geert Jan Olsder, and Jacob Van Der Woude. *Max Plus at work: modeling and analysis of synchronized systems: a course on Max-Plus algebra and its applications*, volume 48. Princeton University Press, 2014. 34

[Hernandez-Flores *et al.*, 2011] E Hernandez-Flores, E Lopez-Mellado, and A Ramirez-Trevino. Diagnosability analysis of partially observable deadlock-free Petri nets. In *Proceedings of 3rd International Workshop on Dependable Control of Discrete Systems, Saarbrücken, Germany*, pages 174–179, 2011. 23

[Jacob *et al.*, 2016] Romain Jacob, Jean-Jacques Lesage, and Jean-Marc Faure. Overview of discrete event systems opacity: Models, validation, and quantification. *Annual Reviews in Control*, 41:135–146, 2016. 25

[Jiang and Kumar, 2006] Shengbing Jiang and Ratnesh 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.

[Jiang *et al.*, 2001] Shengbing Jiang, Zhongdong Huang, Vigyan Chandra, and Ratnesh Kumar. A polynomial algorithm for testing diagnosability of discrete-event systems. *IEEE Transactions on Automatic Control*, 46(8):1318–1321, 2001. 57

[Jiroveanu and Boel, 2010] George Jiroveanu and René K Boel. The diagnosability of Petri net models using minimal explanations. *IEEE Transactions on Automatic Control*, 55(7):1663–1668, 2010. 23

[Johnson, 1975] Donald B Johnson. Finding all the elementary circuits of a directed graph. *SIAM Journal on Computing*, 4(1):77–84, 1975. 80

[Keroglou and Hadjicostis, 2013] Christoforos Keroglou and Christoforos N Hadjicostis. Initial state opacity in stochastic DES. In *Proc. ETFA*, pages 1–8, 2013. 23, 24

[Keroglou and Hadjicostis, 2015] C. Keroglou and C. N. Hadjicostis. Detectability in stochastic discrete event systems. *Systems & Control Letters*, 84:21–26, 2015. 19, 20

[Keroglou and Hadjicostis, 2017] C. Keroglou and C. N. Hadjicostis. Verification of detectability in probabilistic finite automata. *Automatica*, 86:192–198, 2017. 19, 20

[Kirsten and Lombardy, 2009] Daniel Kirsten and Sylvain Lombardy. Deciding unambiguity and sequentiality of polynomially ambiguous min-plus automata. In *LIPIcs-Leibniz International Proceedings in Informatics*, volume 3. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2009. 39

[Klimann *et al.*, 2004] Ines Klimann, Sylvain Lombardy, Jean Mairesse, and Christophe Prieur. Deciding unambiguity and sequentiality from a finitely ambiguous max-plus automaton. *Theoretical Computer Science*, 327(3):349–373, 2004. 39

[Komenda *et al.*, 2017] Jan Komenda, Sbastien Lahaye, Jean-Louis Boimond, and T van den Boom. Max-plus algebra and discrete event systems. In *Proceedings of 20th IFAC World Congress, Toulouse, France*, volume 50, pages 1784–1790, 2017. 18

[Komenda *et al.*, 2018] J. Komenda, S. Lahaye, J.-L. Boimond, and T. van den Boom. Max-plus algebra in the history of discrete event systems. *Annual Reviews in Control*, 45:240–249, 2018. 18

[Lahaye *et al.*, 2015] S. Lahaye, J. Komenda, and J.-L. Boimond. Compositions of (max,+) automata. *Discrete Event Dynamic Systems*, 25:323–344, 2015.

[Lai *et al.*, 2019a] Aiwen Lai, Sébastien Lahaye, and Alessandro Giua. State estimation of max-plus automata with unobservable events. *Automatica*, 105:36–42, 2019. 68

[Lai *et al.*, 2019b] Aiwen Lai, Sébastien Lahaye, and Alessandro Giua. A two-step approach for fault diagnosis of max-plus automata. In *proceedings 6th International Conference on Control, Decision and Information Technologies (CoDIT)*, 2019. 68

[Lai *et al.*, 2019c] Aiwen Lai, Sébastien Lahaye, and Alessandro Giua. Verification of detectability for unambiguous weighted automata. *IEEE Transactions on Automatic Control (submitted)*, 2019. Author version downloadable from *http://tiny.cc/h5hn4y*. 93

[Lai *et al.*, 2019d] Aiwen Lai, Sébastien Lahaye, and Zhiwu Li. Initial-state detectability and initial-state opacity of unambiguous weighted automata. *Automatica (submitted)*, 2019. Author version downloadable from *http://tiny.cc/z6hn4y*. 110

[Lefebvre and Delherm, 2007] Dimitri Lefebvre and Catherine Delherm. Diagnosis of DES with Petri net models. *IEEE Transactions on Automation Science and Engineering*, 4(1):114–118, 2007. 23

[Li and Hadjicostis, 2013] Lingxi Li and Christoforos N Hadjicostis. Minimum initial marking estimation in labeled Petri nets. *IEEE Transactions on Automatic Control*, 58(1):198–203, 2013. 18

[Lin, 1994] F. Lin. Diagnosability of discrete event systems and its applications. *Discrete Event Dynamic Systems*, 4:197–212, 1994. 22

[Lunze and Schröder, 2001] Jan Lunze and Jochen Schröder. State observation and diagnosis of discrete-event systems described by stochastic automata. *Discrete Event Dynamic Systems*, 11(4):319–369, 2001. 22

[Ma *et al.*, 2017] Ziyue Ma, Yin Tong, Zhiwu Li, and Alessandro Giua. Basis marking representation of Petri net reachability spaces and its application to the reachability problem. *IEEE Transactions on Automatic Control*, 62(3):1078–1093, 2017. 20

[Masopust and Yin, 2019a] Tomáš Masopust and Xiang Yin. Complexity of detectability, opacity and a-diagnosability for modular discrete event systems. *Automatica*, 101:290–295, 2019. 20

[Masopust and Yin, 2019b] Tomáš Masopust and Xiang Yin. Deciding detectability for labeled petri nets. *Automatica*, 2019. 20

[Masopust, 2018] Tomáš Masopust. Complexity of deciding detectability in discrete event systems. *Automatica*, 93:257–261, 2018. 20

[Mazaré, 2004] Laurent Mazaré. Using unification for opacity properties. *Proceedings of the 4th IFIP WG1*, 7:165–176, 2004.

[Ozveren and Willsky, 1990] C. M. Ozveren and A. S. Willsky. Observability of discrete event dynamic systems. *IEEE Transactions on Automatic Control*, 35:797–806, 1990. 17

[Ramadge, 1986] P. J. Ramadge. Observability of discrete event systems. In *Proceedings of 25th IEEE Conference on Decision and Control, Athens, Greece*, pages 1108–1112, 1986. 17

[Ramírez-Treviño *et al.*, 2003] Antonio Ramírez-Treviño, Israel Rivera-Rangel, and Ernesto López-Mellado. Observability of discrete event systems modeled by interpreted Petri nets. *IEEE Transactions on Robotics and Automation*, 19(4):557–565, 2003. 20

[Ramírez-Treviño *et al.*, 2007] Antonio Ramírez-Treviño, Elvia Ruiz-Beltrán, Israel Rivera-Rangel, and Ernesto Lopez-Mellado. Online fault diagnosis of discrete event systems. a Petri net-based approach. *IEEE Transactions on Automation Science and Engineering*, 4(1):31–39, 2007. 23

[Ru and Hadjicostis, 2010] Yu Ru and Christoforos N Hadjicostis. Sensor selection for structural observability in discrete event systems modeled by Petri nets. *IEEE Transactions on Automatic Control*, 55(8):1751–1764, 2010.

[Saboori and Hadjicostis, 2007] Anooshiravan Saboori and Christoforos N Hadjicostis. Notions of security and opacity in discrete event systems. In *Decision and Control, 2007 46th IEEE Conference on*, pages 5056–5061. IEEE, 2007. 23

[Saboori and Hadjicostis, 2008] Anooshiravan Saboori and Christoforos N Hadjicostis. Verification of initial-state opacity in security applications of des. In *2008 9th International Workshop on Discrete Event Systems*, pages 328–333. IEEE, 2008. 25

[Saboori and Hadjicostis, 2011] Anooshiravan Saboori and Christoforos N Hadjicostis. Verification of k-step opacity and analysis of its complexity. *IEEE Transactions on Automation Science and Engineering*, 8(3):549–559, 2011. 23, 25

[Saboori and Hadjicostis, 2013a] Anooshiravan Saboori and Christoforos N Hadjicostis. Current-state opacity formulations in probabilistic finite automata. *IEEE Transactions on automatic control*, 59(1):120–133, 2013. 23

[Saboori and Hadjicostis, 2013b] Anooshiravan Saboori and Christoforos N Hadjicostis. Verification of initial-state opacity in security applications of discrete event systems. *Information Sciences*, 246:115–132, 2013. 23, 24, 100

[Sampath *et al.*, 1995] M. Sampath, R. Sengupta, S. Lafortune, K. Sinnamohideen, and D. Teneketzis. Diagnosability of discrete-event systems. *IEEE Transactions on Automatic Control*, 40:1555–1575, 1995. 21, 59

[Sasi and Lin, 2018] Yazeed Sasi and Feng Lin. Detectability of networked discrete event systems. *Discrete Event Dynamic Systems*, 28(3):449–470, 2018. 20

[Schützenberger, 1961] Marcel Paul Schützenberger. On the definition of a family of automata. *Information and control*, 4(2-3):245–270, 1961. 36

[Shu and Lin, 2011] Shaolong Shu and Feng Lin. Generalized detectability for discrete event systems. *Systems & control letters*, 60(5):310–317, 2011. 19, 26, 46, 57, 59, 72, 78, 83, 92

[Shu and Lin, 2013] Shaolong Shu and Feng Lin. I-detectability of discrete-event systems. *IEEE Transactions on Automation Science and Engineering*, 10(1):187–196, 2013. 19, 20, 97, 100

[Shu *et al.*, 2007] S. Shu, F. Lin, and H. Ying. Detectability of discrete event systems. *IEEE Transactions on Automatic Control*, 52:2356–2359, 2007. 19, 26

[Shu *et al.*, 2008] S. Shu, F. Lin, H. Ying, and X. Chen. State estimation and detectability of probabilistic discrete event systems. *Automatica*, 44:3054–3060, 2008. 19

[Sweeney, 2002] Latanya Sweeney. k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(05):557–570, 2002. 24

[Takai and Kumar, 2009] Shigemasa Takai and Ratnesh Kumar. Verification and synthesis for secrecy in discrete-event systems. In *2009 American Control Conference*, pages 4741–4746. IEEE, 2009. 25

[Thistle and Wonham, 1994] John G Thistle and W Murray Wonham. Control of infinite behavior of finite automata. *SIAM Journal on Control and Optimization*, 32(4):1075–1097, 1994. 46, 71, 97

[Thorsley *et al.*, 2008] David Thorsley, Tae-Sic Yoo, and Humberto E Garcia. Diagnosability of stochastic discrete-event systems under unreliable observations. In *Proceedings of American Control Conference, Seattle, USA*, pages 1158–1165, 2008. 22

[Tong *et al.*, 2017] Yin Tong, Zhiwu Li, Carla Seatzu, and Alessandro Giua. Verification of state-based opacity using Petri nets. *IEEE Transactions on Automatic Control*, 62(6):2823–2837, 2017. 23, 24

[Tripakis, 2002] S. Tripakis. Fault diagnosis for timed automata. In *International symposium on formal techniques in real-time and fault-tolerant systems*, pages 205–221. Springer, 2002. 21, 58

[Wang *et al.*, 2011] X. Wang, C. Mahulea, J.e Júlvez, and M. Silva. On state estimation of timed choice-free Petri nets. In *Proceedings of 18th IFAC World Congress, Milano, Italy*, volume 44, pages 8687–8692, 2011. 18

[Wang *et al.*, 2018] Lingtai Wang, Naijun Zhan, and Jie An. The opacity of real-time automata. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 37(11):2845–2856, 2018. 23, 24, 109

[Weber and Seidl, 1991] Andreas Weber and Helmut Seidl. On the degree of ambiguity of finite automata. *Theoretical Computer Science*, 88(2):325–349, 1991. 39

[Wen *et al.*, 2005] YuanLin Wen, ChunHsi Li, and MuDer Jeng. A polynomial algorithm for checking diagnosability of Petri nets. In *Proceedings of IEEE International Conference on Systems, Man and Cybernetics, Waikoloa, Hawaii, USA*, volume 3, pages 2542–2547, 2005. 23

[Wonham, 1976] W. M. Wonham. Towards an abstract internal model principle. *IEEE Transactions on Systems, Man, and Cybernetics*, 6:735–740, 1976. 17

[Wu and Hadjicostis, 2005] Yingquan Wu and Christoforos N Hadjicostis. Algebraic approaches for fault identification in discrete-event systems. *IEEE Transactions on Automatic Control*, 50(12):2048–2055, 2005. 22

[Wu and Lafortune, 2013] Yi-Chin Wu and Stéphane Lafortune. Comparative analysis of related notions of opacity in centralized and coordinated architectures. *Discrete Event Dynamic Systems*, 23(3):307–339, 2013. 23, 24

[Yin and Lafortune, 2015] Xiang Yin and Stéphane Lafortune. Codiagnosability and coobservability under dynamic observations: Transformation and verification. *Automatica*, 61:241–252, 2015.

[Yin and Lafortune, 2017a] Xiang Yin and Stéphane Lafortune. A new approach for the verification of infinite-step and k-step opacity using two-way observers. *Automatica*, 80:162–171, 2017. 25

[Yin and Lafortune, 2017b] Xiang Yin and Stéphane Lafortune. On the decidability and complexity of diagnosability for labeled Petri nets. *IEEE Transactions on Automatic Control*, 62(11):5931–5938, 2017. 23

[Yin and Lafortune, 2017c] Xiang Yin and Stéphane Lafortune. Verification complexity of a class of observational properties for modular discrete events systems. *Automatica*, 83:199–205, 2017. 20

[Yin *et al.*, 2019] Xiang Yin, Zhaojian Li, Weilin Wang, and Shaoyuan Li. Infinite-step opacity and k-step opacity of stochastic discrete-event systems. *Automatica*, 99:266–274, 2019. 25

[Yin, 2017] X. Yin. Initial-state detectability of stochastic discrete-event systems with probabilistic sensor failures. *Automatica*, 80:127–134, 2017. 19, 20

[Yin, 2019] Xiang Yin. Estimation and verification of partially-observed discrete-event systems. *arXiv preprint arXiv:1903.11413*, 2019. 20, 25

[Yoo and Lafortune, 2002] Tae-Sic Yoo and Stéphane Lafortune. Polynomial-time verification of diagnosability of partially observed discrete-event systems. *IEEE Transactions on automatic control*, 47(9):1491–1495, 2002. 57

[Zad *et al.*, 2003] S. H. Zad, R. H. Kwong, and W. M. Wonham. Fault diagnosis in discrete-event systems: Framework and model reduction. *IEEE Transactions on Automatic Control*, 48:1199–1212, 2003. 22, 61

[Zad *et al.*, 2005] S Hashtrudi Zad, Raymond H Kwong, and W Murray Wonham. Fault diagnosis in discrete-event systems: Incorporating timing information. *IEEE Transactions on Automatic Control*, 50(7):1010–1015, 2005. 22

[Zaytoon and Lafortune, 2013] Janan Zaytoon and Stéphane Lafortune. Overview of fault diagnosis methods for discrete event systems. *Annual Reviews in Control*, 37(2):308–320, 2013. 21

[Zhang, 2017] Kuize Zhang. The problem of determining the weak (periodic) detectability of discrete event systems is pspace-complete. *Automatica*, 81:217–220, 2017. 20

[Zhao *et al.*, 2019] Pei Zhao, Shaolong Shu, Feng Lin, and Bo Zhang. Detectability measure for state estimation of discrete event systems. *IEEE Transactions on Automatic Control*, 64(1):433–439, 2019. 20

UNIVERSITE BRETAGNE LOIRE

université angers

Thèse de Doctorat

Aiwen LAI

Estimation d'état et Vérification de la Détectabilité et de l'Opacité dans les Automates Pondérés

State Estimation and Verification of Detectability and Opacity in Weighted Automata

Résumé

Cette thèse porte sur l'estimation détat, le diagnostic d'erreur et la vérification de la détectabilité de létat actuel, de la détectabilité de létat initial et de l'opacité de l'état initial dans le cadre des automates pondérés.

Abstract

This thesis focuses on the state estimation, fault diagnosis, and verification of current-state detectability, initial-state detectability and initial-state opacity in the framework of weighted automata.

Mots clés

Système événements discrets, automates pondérés, estimation d'état, diagnostic d'erreur, détectabilité, opacité.

Key Words

Discrete event system, weighted automata, state estimation, fault diagnosis, detectability, opacity.