A unified concurrent-composition method to state/event inference and concealment in discrete-event systems

Kuize Zhang

Control Systems Group, Technical University of Berlin, 10587 Berlin, Germany

kuize.zhang@campus.tu-berlin.de

Abstract Discrete-event systems usually consist of discrete states and transitions between them caused by spontaneous occurrences of labelled (aka partially-observed) events. Due to the partially-observed feature, fundamental properties therein could be classified into two categories: state/event-inference-based properties (e.g., strong detectability, diagnosability, and predictability) and state-concealment-based properties (e.g., opacity). Intuitively, the former category describes whether one can use observed output sequences to infer the current and subsequent states, past occurrences of faulty events, or future certain occurrences of faulty events; while the latter describes whether one cannot use observed output sequences to infer whether some secret states have been visited (that is, whether the DES can conceal the status that its secret states have been visited). Over the past two decades these properties were studied separately using different methods. In this review article, for labeled finite-state automata, a unified concurrent-composition method is shown to verify all above inference-based properties and concealment-based properties, resulting in a unified mathematical framework for the two categories of properties. In addition, compared with the previous methods in the literature, the concurrent-composition method does not depend on assumptions and is more efficient.

Keywords discrete-event system, labeled finite-state automaton, inference, concealment, concurrent composition

1 Introduction

Discrete-event systems (DESs) usually consist of discrete states and transitions between them caused by spontaneous occurrences of labelled (aka partially-observed) events. Hence DESs are autonomous (they are not driven by external factors) and nonlinear [1]. Due to the partially-observed feature of DESs, observation-based fundamental properties could be classified into two categories: *inference*-based and *concealment*-based. The former means whether one can infer further information of a DES from the observations to the DES, so that these information could be used to do further study on the DES, e.g., synthesizing a controller to change several properties of the DES. While the latter refers to whether the DES can forbid several information from being leaked to external intruders, even though the intruders can see outputs/labels generated by the DES. Roughly speaking, the former is dual to the latter. The former category contain detectability, diagnosability, predictability, etc.; the latter contain opacity, etc. Detectability, diagnosability, and predictability mean whether one can use observed output sequences generated by a DES to determine its current states, past occurrences of faulty events, and future certain occurrences of faulty events; opacity refers to whether one cannot use observed output sequences to determine whether secret states have been visited.

Over the two past decades, these properties were studied separately, but no intrinsic relations between them were revealed. In this paper, for DESs modeled by labeled finite-state automata (LFSAs), a unified *concurrent-composition* method is given to verify all of them, thus a unified mathematical framework is given to include many inference-based and concealment-based properties. Roughly speaking, a concurrent composition collects all pairs of trajectories of two systems producing the same output sequence, in which observable transitions with the same output are synchronized but unobservable transitions interleave. For strong detectability, diagnosability, and predictability (which are inference-based properties), the concurrent compositions of trivial variants of an LFSA and trivial variants of itself are constructed to do verification in polynomial time; for variants of opacity (which are concealment-based properties), the concurrent compositions of trivial variants of an LFSA and the observers[1] of trivial variants of itself are constructed to do verification in exponential time. In a large extent, *from a complexity point of view, it is easy to infer something, but it is hard to conceal something.*

The concurrent-composition method shows advantage in verifying inference-based properties of DESs since it does not rely on any assumption. Existing results in employing the detector method [3] to verify strong detectability depends on two fundamental assumptions, i.e., deadlock-freeness which assumes an LFSA always running, and divergence-freeness which requires the running of an LFSA will always be eventually observed. These two assumptions are also adopted in the twin-plant method [4] and the verifier method in [5, 6] for verifying diagnosability and predictability.

When being applied to verify concealment-based properties, the concurrent-composition method is more efficient than almost all methods in the literature, e.g., the initial-state-estimator method [7], the two-way-observer method [8], the K-delay-trajectory-estimator-method [9], and the K/Inf-step-recognizer method [10]. The only known exception lies in the fact that only two variants of opacity, current-state opacity and initial-state opacity, could be verified by directly using the notions of observer [11, 12] and reverse observer [13] (see Section 5 for details).

Notation Symbol $\mathbb{N}$ denotes the set of nonnegative integers. For an (finite) alphabet Σ (i.e., every sequence of elements of Σ is a unique sequence of elements of Σ, e.g., $\{0, 00\}$ is not an alphabet since $000 = 0\,00 = 00\,0$), Σ^* and Σ^ω are used to denote the set of *words* (i.e., finite-length sequences of elements of Σ) over Σ including the empty word ϵ and the set of *configurations* (i.e., infinite-length sequences of elements of Σ) over Σ, respectively. $\Sigma^+ := \Sigma^* \setminus \{\epsilon\}$. For a word $s \in \Sigma^*$, $|s|$ stands for its length, and we set $|s'| = +\infty$ for all $s' \in \Sigma^\omega$. For $s \in \Sigma^+$ and $k \in \mathbb{N}$, s^k and s^ω denote the concatenations of k copies of s and infinitely many copies of s, respectively. Analogously, $L_1L_2 := \{e_1e_2 | e_1 \in L_1, e_2 \in L_2\}$, where $L_1, L_2 \subset \Sigma^*$. For a word (configuration) $s \in \Sigma^*(\Sigma^\omega)$, a word $s' \in \Sigma^*$ is called a *prefix* of s, denoted as $s' \sqsubset s$, if there exists another word (configuration) $s'' \in \Sigma^*(\Sigma^\omega)$ such that $s = s's''$. For a set S, $|S|$ denotes its cardinality and 2^S its power set. Symbols $\subset$ and $\subsetneq$ denote the subset and strict subset relations, respectively.

[1]i.e., the standard powerset construction used for determinizing nondeterministic finite automata with ϵ-transitions [2]

Definition 1 *A* labeled finite-state automaton *is a sextuple $\mathcal{S} = (Q, E, \delta, Q_0, \Sigma, \ell)$, where*

1. *Q is a finite set of* states,
2. *E (which is an alphabet) is a finite set of* events,
3. *$\delta : Q \times E \to 2^Q$ is the* transition function *(equivalently represented by the* transition relation *$\delta \subset Q \times E \times Q$ such that $q' \in \delta(q, e)$ if and only if $(q, e, q') \in \delta$),*
4. *$Q_0 \subset Q$ is a set of* initial states,
5. *Σ (also an alphabet) is a finite set of* outputs/labels*, and*
6. *$\ell : E \to \Sigma \cup \{\epsilon\}$ is the* labeling function.

A transition $(q, e, q') \in \delta$ is interpreted as when $\mathcal{S}$ is in state q and event e occurs $\mathcal{S}$ transitions to state q'. The event set E can been rewritten as disjoint union of *observable* event set $E_o = \{e \in E | \ell(e) \in \Sigma\}$ and *unobservable* event set $E_{uo} = \{e \in E | \ell(e) = \epsilon\}$. When an observable event occurs, its label can be observed; when an unobservable event occurs, nothing can be observed. Transition function $\delta : Q \times E \to 2^Q$ is recursively extended to $\delta : Q \times E^* \to 2^Q$ as follows: for all $q \in Q$, $u \in E^*$, and $a \in \Sigma$, one has $\delta(q, \epsilon) = \{q\}$ and $\delta(q, ua) = \bigcup_{p \in \delta(q,u)} \delta(p, a)$; equivalently, transition relation $\delta \subset Q \times E \times Q$ is recursively extended to $\delta \subset Q \times E^* \times Q$ as follows: (1) for all $q, q' \in Q$, $(q, \epsilon, q') \in \delta$ if and only if $q = q'$; (2) for all $q, q' \in Q$, $s \in E^*$, and $e \in E$, one has $(q, se, q') \in \delta$, also denoted by $q \xrightarrow{se} q'$, called *transition sequence* or *run*, if and only if, $(q, s, q''), (q'', e, q') \in \delta$ for some $q'' \in Q$.

Labeling function $\ell : E \to \Sigma \cup \{\epsilon\}$ is recursively extended to $\ell : E^* \cup E^\omega \to \Sigma^* \cup \Sigma^\omega$ as $\ell(e_1 e_2 \dots) = \ell(e_1)\ell(e_2)\dots$ and $\ell(\epsilon) = \epsilon$. For all $E' \subset E$, $\ell(E') := \{\ell(e) | e \in E'\}$. Transitions $x \xrightarrow{e} x'$ with $\ell(e) = \epsilon$ (resp., $\ell(e) \neq \epsilon$) are called *unobservable* (resp., *observable*). For $q \in Q$ and $s \in E^+$, (q, s, q) is called a *transition cycle* if $(q, s, q) \in \delta$. An *observable transition cycle* is defined by a transition cycle with at least one observable transition. Analogously an *unobservable transition cycle* is defined by a transition cycle with no observable transition. An LFSA is called *deterministic* if $|Q_0| = 1$ and for all $q, q', q'' \in Q$ and $e \in E$, if $(q, e, q'), (q, e, q'') \in \delta$ then $q' = q''$.

A state $q \in Q$ is called *live* if $(q, e, q') \in \delta$ for some $e \in E$ and $q' \in Q$. $\mathcal{S}$ is called *live/deadlock-free* if each of its reachable states is live. *A state $q' \in Q$ is reachable from a state* $q \in Q$ if there exists $s \in E^+$ such that $q \xrightarrow{s} q'$. *A subset Q' of Q is reachable from a state* $q \in Q$ if some state of Q' is reachable from q. Similarly *a state $q \in Q$ is reachable from a subset Q' of Q* if q is reachable from some state of Q'. A state $q \in Q$ is called *reachable* (*in $\mathcal{S}$*) if either $q \in Q_0$ or it is reachable from some initial state. For a transition $(q, e, q') \in \delta$, a transition $(q'', e', q''') \in \delta$ is called a *predecessor* of (q, e, q') if either $q = q'''$ or q is reachable from q'''; a transition (q'', e', q''') is called a *successor* of (q, e, q') if either $q'' = q'$ or q'' is reachable from q'.

The symbol $L(\mathcal{S}) := \{s \in E^* | (\exists q_0 \in Q_0)(\exists q \in Q)[q_0 \xrightarrow{s} q]\}$ will be used to denote the set of finite-length event sequences generated by $\mathcal{S}$, $L^\omega(\mathcal{S}) = \{e_1 e_2 \dots \in E^\omega | (\exists q_0 \in Q_0)(\exists q_1, q_2, \dots \in Q)[q_0 \xrightarrow{e_1} q_1 \xrightarrow{e_2} \cdots]\}$ will denote the set of infinite-length event sequences generated by $\mathcal{S}$. For each $\sigma \in \Sigma^*$, $\mathcal{M}(\mathcal{S}, \sigma)$ denotes the *current-state estimate*, i.e., the set of states that the system can be in when σ has just been generated, i.e., $\mathcal{M}(\mathcal{S}, \sigma) := \{q \in Q | (\exists q_0 \in Q_0)(\exists s \in E^*)[(\ell(s) = \sigma) \wedge (q_0 \xrightarrow{s}$

$q)]\}$. $\mathcal{L}(\mathcal{S})$ denotes the *language generated* by $\mathcal{S}$, i.e., $\mathcal{L}(\mathcal{S}) := \{\sigma \in \Sigma^* | \mathcal{M}(\mathcal{S},\sigma) \neq \emptyset\}$. $\mathcal{L}^\omega(\mathcal{S})$ denotes the *ω-language generated* by $\mathcal{S}$, i.e., $\mathcal{L}^\omega(\mathcal{S}) := \{\sigma \in \Sigma^\omega | (\exists s \in L^\omega(\mathcal{S}))[\ell(s) = \sigma]\}$. For a subset $x \subset Q$ of states, its *unobservable reach* $\mathrm{UR}(x)$ is defined by $\bigcup_{q\in x}\bigcup_{s\in(E_{uo})^*}\delta(q,s)$.

Example 1 *Consider the following LFSA $\mathcal{S}_1$. It is deterministic but not live (q_1 is not live). One sees $L(\mathcal{S}_1) = \{(e_1)^n, (e_1)^n e_2 | n \geq 0\}$, $L^\omega(\mathcal{S}_1) = \{(e_1)^\omega\}$, $\mathcal{L}(\mathcal{S}_1) = \{a^n, a^n b | n \geq 0\}$, $\mathcal{L}^\omega(\mathcal{S}_1) = \{a^\omega\}$, and $\mathcal{M}(\mathcal{S}_1, b) = \{q_1\}$.*

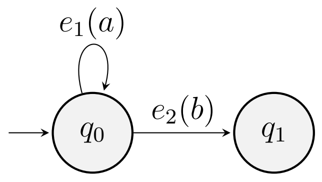


Figure 1: LFSA $\mathcal{S}_1$, where q_0 is the initial state (with an input arrow from nowhere), $\ell(e_1) = a$, $\ell(e_2) = b$.

Next we introduce the main tool — concurrent composition. The concurrent-composition structure exactly arose from characterizing *negation* of a strong version of detectability called *eventual strong detectability* in [14], where the eventual strong detectability is essentially different from and strictly weaker than the notion of *strong detectability* proposed in [15]. In the concurrent composition of two automata, observable transitions with the same label are synchronized, while unobservable transitions interleave.

Definition 2 ([16, 14]) *Consider two LFSAs $\mathcal{S}^i = (Q_i, E, \delta_i, Q_{0i}, \Sigma, \ell)$, $i = 1, 2$, we define the* concurrent composition $\mathrm{CC_A}(\mathcal{S}^1, \mathcal{S}^2)$ *of $\mathcal{S}^1$ and $\mathcal{S}^2$ by*

$$\mathrm{CC_A}(\mathcal{S}^1, \mathcal{S}^2) = (Q', E', \delta', Q'_0, \Sigma, \ell'), \tag{1}$$

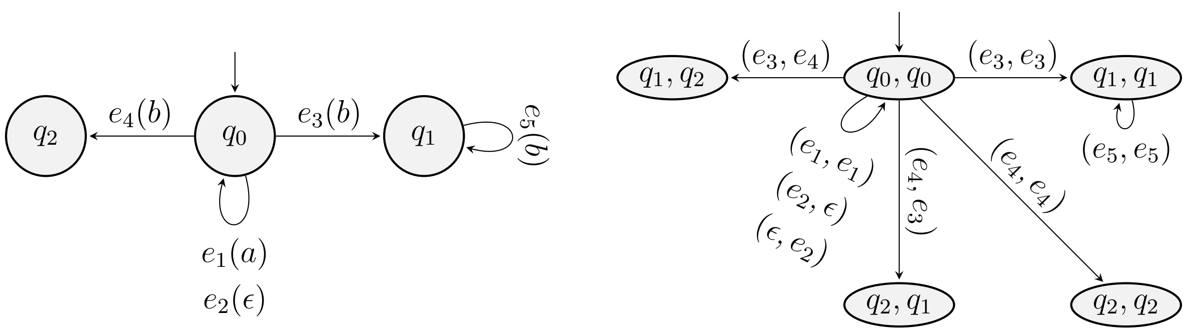
where

1. *$Q' = Q_1 \times Q_2$;*
2. *$E' = E'_o \cup E'_{uo}$, where $E'_o = \{(\breve{e}, \breve{e}') | \breve{e}, \breve{e}' \in E_o, \ell(\breve{e}) = \ell(\breve{e}')\}$, $E'_{uo} = \{(\breve{e}, \epsilon) | \breve{e} \in E_{uo}\} \cup \{(\epsilon, \breve{e}) | \breve{e} \in E_{uo}\}$;*
3. *for all $(\breve{q}_1, \breve{q}'_1), (\breve{q}_2, \breve{q}'_2) \in Q'$, $(\breve{e}, \breve{e}') \in E'_o$, $(\breve{e}'', \epsilon) \in E'_{uo}$, and $(\epsilon, \breve{e}''') \in E'_{uo}$,*
   - *$((\breve{q}_1, \breve{q}'_1), (\breve{e}, \breve{e}'), (\breve{q}_2, \breve{q}'_2)) \in \delta'$ if and only if $(\breve{q}_1, \breve{e}, \breve{q}_2) \in \delta_1$, $(\breve{q}'_1, \breve{e}', \breve{q}'_2) \in \delta_2$,*
   - *$((\breve{q}_1, \breve{q}'_1), (\breve{e}'', \epsilon), (\breve{q}_2, \breve{q}'_2)) \in \delta'$ if and only if $(\breve{q}_1, \breve{e}'', \breve{q}_2) \in \delta_1$, $\breve{q}'_1 = \breve{q}'_2$,*
   - *$((\breve{q}_1, \breve{q}'_1), (\epsilon, \breve{e}'''), (\breve{q}_2, \breve{q}'_2)) \in \delta'$ if and only if $\breve{q}_1 = \breve{q}_2$, $(\breve{q}'_1, \breve{e}''', \breve{q}'_2) \in \delta_2$;*
4. *$Q'_0 = Q_{01} \times Q_{02}$;*
5. *for all $(\breve{e}, \breve{e}') \in E'$, $\ell'((\breve{e}, \breve{e}')) := \ell(\breve{e}) = \ell(\breve{e}')$.*

Particularly if $\mathcal{S}^1 = \mathcal{S}^2$, then $\mathrm{CC_A}(\mathcal{S}^1, \mathcal{S}^2) =: \mathrm{CC_A}(\mathcal{S}^1)$ is called the self-composition *of $\mathcal{S}^1$.*

For an event sequence $s' \in (E')^*$, we use $s'(L)$ and $s'(R)$ to denote its left and right components, respectively. Similar notation is applied to states of Q'. In addition, for every $s' \in (E')^*$, we use $\ell(s')$ to denote $\ell(s'(L))$ or $\ell(s'(R))$, since $\ell(s'(L)) = \ell(s'(R))$. In the above construction, $\mathrm{CC_A}(\mathcal{S}^1, \mathcal{S}^2)$ aggregates all pairs of runs of $\mathcal{S}_1$ and runs of $\mathcal{S}_2$ that produce the same label sequence.

Example 2 *An LFSA $\mathcal{S}_2$ and its self-composition $\mathrm{CC_A}(\mathcal{S}_2)$ are shown in Figure 2.*



(a) $\mathcal{S}_2$. (b) $\mathrm{CC_A}(\mathcal{S}_2)$.

Figure 2: LFSA $\mathcal{S}_2$ (left) and its self-composition (right, only reachable states illustrated).

2 Detectability

In this section, we show how to use the concurrent-composition method to verify strong detectability.

The study of the state detection problem dates back to the 1950s [17] in computer science and the 1960s [18] in control science, respectively. In the former, Moore studied initial-state detection (called Gedanken-experiment) of finite-state machines which were called Moore machines later; in the latter, Kalman studied initial-state detection (called observability) of linear differential equations. The two seminal papers induces many research branches in computer science and control, e.g., model-based testing of all kinds of reactive systems [19] in computer science and observability studies of all kinds of control systems, e.g., arranging from linear systems [18, 20], to nonlinear systems [21, 22, 23], to switched systems [24], and also to networked systems [25, 26].

The state detection problem in DESs dates back to the 1980s [27, 28], and two widely accepted fundamental notions are *strong detectability* and *weak detectability* proposed in 2007 by Shu, Lin, and Ying [15], where the former implies that there is a delay k such that for *each* event sequence generated by an LFSA, each prefix of its output sequence of length greater than k allows reconstructing the current state. The latter relaxes the former by changing *each* to *some*. When long-term behavior is considered, we let the above conditions apply to all infinite-length generated event sequences (in this case we call the notions ω-detectability); when short-term behavior is considered, we let them apply to all finite-length generated event sequences (in this case we call the notions $*$-detectability).

Definition 3 **(ω-SD [15])** *An LFSA $\mathcal{S}$ is called* ω-strongly detectable *if there exists a positive integer k such that for each infinite-length event sequence $s \in L^\omega(\mathcal{S})$, $|\mathcal{M}(\mathcal{S}, \sigma)| = 1$ for every prefix σ of $\ell(s)$ satisfying $|\sigma| > k$.*

Definition 4 (ω-WD [15]) *An LFSA $\mathcal{S}$ is called* ω-weakly detectable *if $L^\omega(\mathcal{S}) \neq \emptyset$ implies there exists an infinite-length event sequence $s \in L^\omega(\mathcal{S})$ such that for some positive integer k, $|\mathcal{M}(\mathcal{S},\sigma)| = 1$ for every prefix σ of $\ell(s)$ satisfying $|\sigma| > k$.*

Definition 5 ($*$-SD) *An LFSA $\mathcal{S}$ is called* $*$-strongly detectable *if there exists a positive integer k such that for each finite-length event sequence $s \in L(\mathcal{S})$, $|\mathcal{M}(\mathcal{S},\sigma)| = 1$ for every prefix σ of $\ell(s)$ satisfying $|\sigma| > k$.*

Definition 6 ($*$-WD) *An LFSA $\mathcal{S}$ is called* $*$-weakly detectable *if there exists a finite-length event sequence $s \in L(\mathcal{S})$ such that for some positive integer k, $|\mathcal{M}(\mathcal{S},\sigma)| = 1$ for every prefix σ of $\ell(s)$ satisfying $|\sigma| > k$.*

An exponential-time verification algorithm based on the notion of observer for weak detectability was given in 2007 [15]. Recently, verifying weak detectability was proven to be PSPACE-complete [29, 30]. In [3], a detector method was used to verify strong detectability in polynomial time, under the two assumptions of deadlock-freeness and divergence-freeness as mentioned above, where the detector is a simplified version of the observer by splitting the states of the observer into subsets of cardinality 2. We refer the reader to [14, Remark 2] for why the detector method depends on the two assumptions and without the two assumptions the detector method does not work generally. In order to verify strong detectability, we choose to characterize its *negation* (which is essentially different from the way of directly verifying strong detectability adopted in [15, 3]). By definition, the following proposition holds.

Proposition 2.1 ([31]) *An LFSA $\mathcal{S}$ is not ω-strongly detectable (resp., $*$-strongly detectable) if and only if for every positive integer k there exists an infinite-length (resp., finite-length) event sequence $s \in L^\omega(\mathcal{S})$ (resp., $s \in L(\mathcal{S})$) such that $|\mathcal{M}(\mathcal{S},\sigma)| > 1$ for some prefix σ of $\ell(s)$ satisfying $|\sigma| > k$.*

With the notion of self-composition of an LFSA $\mathcal{S}$, we give sufficient and necessary conditions for negation of two versions of strong detectability, without any assumption.

Theorem 2.2 ([16, 31]) *An LFSA $\mathcal{S}$ is not $*$-strongly detectable if and only if in $\mathrm{CC_A}(\mathcal{S})$, there exists a run*

$$q_0' \xrightarrow{s_1'} q_1' \xrightarrow{s_2'} q_1' \xrightarrow{s_3'} q_2' \tag{2}$$

satisfying

$$q_0' \in Q_0';\ q_1', q_2' \in Q';\ s_1', s_2', s_3' \in (E')^*;\ \ell(s_2') \in \Sigma^+;\ q_2'(L) \neq q_2'(R). \tag{3}$$

An LFSA $\mathcal{S}$ is not ω-strongly detectable if and only if in $\mathrm{CC_A}(\mathcal{S})$, there exists a run (2) satisfying (3) and in $\mathcal{S}$, there exists a transition cycle reachable from $q_2'(L)$.

Proof We use Proposition 2.1 to prove this theorem. We first consider $*$-strong detectability.

"only if": Assume $\mathcal{S}$ is not $*$-strongly detectable. Then by Proposition 2.1, choose $k = |Q|^2$, there exists $s_k \in L(\mathcal{S})$ and $\sigma \in \Sigma^+$ such that $\mathcal{M}(\mathcal{S},\sigma) > 1$, $\sigma \sqsubset \ell(s)$, and $|\sigma| > k$. Then in $\mathrm{CC_A}(\mathcal{S})$, there exists a run $q_0' \xrightarrow{s'} q'$ such that $q_0' \in Q_0'$, $\ell(s') = \sigma$ and $q'(L) \neq q'(R)$. Since $|\sigma| > |Q|^2$ and there exist at most $|Q|^2$ distinct states in $\mathrm{CC_A}(\mathcal{S})$, by the pigeonhole principle[2], the run $q_0' \xrightarrow{s'} q'$ can be rewritten as $q_0' \xrightarrow{s_1'} q_1' \xrightarrow{s_2'} q_1' \xrightarrow{s_3'} q'$, where $\ell(s_2') \in \Sigma^+$.

"if": Assume in $\mathrm{CC_A}(\mathcal{S})$ there exists a run (2) satisfying (3). We choose event sequence $s := s_1'(L)(s_2'(L))^{k+1}s_3'(L) \in L(\mathcal{S})$, then $|\ell(s)| > k$ and $|\mathcal{M}(\mathcal{S}, \ell(s))| > 1$. By Proposition 2.1, $\mathcal{S}$ is not $*$-strongly detectable.

We second consider ω-strong detectability. Because a transition cycle reachable from $q_2'(L)$ can be repeated arbitrarily often, resulting in an infinite-length run starting from $q_2'(L)$. Then based on the above argument for $*$-strong detectability, the sufficient and necessary condition for ω-strong detectability also holds. □

Example 3 *Reconsider the LFSA $\mathcal{S}_2$ in Figure 2 (left) and its self-composition $\mathrm{CC_A}(\mathcal{S}_2)$ in Figure 2 (right). In $\mathrm{CC_A}(\mathcal{S}_2)$, one sees a run*

$$(q_0, q_0) \xrightarrow{(e_1,e_1)} (q_0, q_0) \xrightarrow{(e_3,e_4)} (q_1, q_2)$$

such that $\ell((e_1, e_1)) = a$ is of positive length and $q_1 \neq q_2$. Then by Theorem 2.2, $\mathcal{S}_2$ is not $*$-strongly detectable. In addition, in $\mathcal{S}_2$, there is a self-loop on q_1, hence also by Theorem 2.2, $\mathcal{S}_2$ is not ω-strongly detectable.

3 Diagnosability

In order to define *diagnosability* for an LFSA $\mathcal{S} = (Q, E, \delta, Q_0, \Sigma, \ell)$, we specify a subset $E_f \subset E$ of faulty events. Diagnosability describes whether one can use an observed output sequence to determine whether some faulty event has occurred. For an event sequence $s \in E^*$, $E_f \in s$ denotes that some element of E_f appears in s.

Definition 7 (Diag [32]) *Consider an LFSA $\mathcal{S} = (Q, E, \delta, Q_0, \Sigma, \ell)$ and a subset $E_f \subset E$ of faulty events. $\mathcal{S}$ is called E_f-*diagnosable *if*

$$\begin{aligned}&(\exists k \in \mathbb{N})(\forall s \in L(\mathcal{S}) \cap E^* E_f)(\forall s' : ss' \in L(\mathcal{S}))\\ &[(|s'| > k) \implies \mathbf{D}],\end{aligned}$$

where $\mathbf{D} = (\forall s'' \in L(\mathcal{S}))[(\ell(s'') = \ell(ss')) \implies (E_f \in s'')]$.

Intuitively, if $\mathcal{S}$ is E_f-diagnosable, then once a faulty event (e.g., the last event in s) occurs, one can make sure that some faulty event has occurred after at least k subsequent events (e.g., s') occur by observing output sequences.

In 1995, Sampath et al. [32] proposed a diagnoser method to verify diagnosability. The diagnoser of an LFSA records state estimates along observed output sequences and also records fault propagation along transitions of states of the LFSA. The same as the observer mentioned above, the diagnoser

[2] If n items are put into m containers, with $n > m$, then at least one container must contain more than one item.

also has exponential complexity, and diagnosability is verifiable by a relatively simple cycle condition on the diagnoser. Hence diagnosability can be verified in exponential time. Also the same as the case that the observer method depends on the two assumptions of deadlock-freeness and divergence-freeness when being applied to verify detectability [15], the diagnoser method also depends on the two assumptions when being applied to verify diagnosability. Later in 2001, a twin-plant method with polynomial complexity was proposed by Jiang et al. [4] to verify diagnosability in polynomial time. Because in a twin plant, only observable transitions are synchronized, the method also depends on the two assumptions. One year later, Yoo and Lafortune [5] proposed a verifier method to verify diagnosability in polynomial time, where in a verifier, observable transitions are synchronized, unobservable transitions are also considered but their events' positions (left or right) are neglected, so that the method also depends on the two assumptions. From then on, in many papers, verification of all kinds of variants of diagnosability depends on the two assumptions. The two assumptions were removed by Cassez and Tripakis [33] in 2008 by using a generalized version of the twin-plant structure to verify *negation* of diagnosability in polynomial time, where in the generalized version of the twin plant, observable transitions are synchronized, unobservable transitions are also considered but their events' positions (left or right) are also considered. The generalized version of the twin-plant structure and the concurrent-composition structure [14] were proposed in a similar way: they were proposed by characterizing *negation* of inference-based properties.

In order to verify E_f-diagnosability of $\mathcal{S}$, we use the concurrent composition $\mathrm{CC_A}(\mathcal{S}_\mathsf{f}, \mathcal{S}_\mathsf{n})$ (similar to but simpler than the generalized version of the twin plant proposed in [33]) of the faulty subautomaton $\mathcal{S}_\mathsf{f}$ and the normal subautomaton $\mathcal{S}_\mathsf{n}$, where $\mathcal{S}_\mathsf{f}$ is obtained from $\mathcal{S}$ by only keeping faulty transitions and all their predecessors and successors, $\mathcal{S}_\mathsf{n}$ is obtained from $\mathcal{S}$ by removing all faulty transitions. $\mathrm{CC_A}(\mathcal{S}_\mathsf{f}, \mathcal{S}_\mathsf{n})$ is computed similarly as in Definition 2.

Theorem 3.1 ([34]) *Consider an LFSA $\mathcal{S} = (Q, E, \delta, Q_0, \Sigma, \ell)$ and a subset $E_f \subset E$ of faulty events. $\mathcal{S}$ is not E_f-diagnosable if and only if in $\mathrm{CC_A}(\mathcal{S}_\mathsf{f}, \mathcal{S}_\mathsf{n})$, there exists a run*

$$q_0' \xrightarrow{s_1'} q_1' \xrightarrow{e'} q_2' \xrightarrow{s_2'} q_3' \xrightarrow{s_3'} q_3' \tag{4}$$

satisfying

$$q_0' \in Q_0';\ e'(L) \in E_f;\ |s_3'(L)| > 0. \tag{5}$$

Proof By definition, $\mathcal{S}$ is not E_f-diagnosable if and only if

$$(\forall k \in \mathbb{N})(\exists s_k \in L(\mathcal{S}) \cap E^* E_f)(\exists s_k' : s_k s_k' \in L(\mathcal{S}))(\exists s_k'' \in L(\mathcal{S}))$$
$$[(|s_k'| > k) \wedge (\ell(s_k'') = \ell(s_k s_k')) \wedge (E_f \notin s_k'')].$$

Choose sufficiently large k, by the finiteness of the number of states of $\mathcal{S}$ and the pigeonhole principle, $\mathcal{S}$ is not E_f-diagnosable if and only if in $\mathrm{CC_A}(\mathcal{S}_\mathsf{f}, \mathcal{S}_\mathsf{n})$, there exists a run (4) satisfying (5). $\square$

Example 4 *Consider the LFSA $\mathcal{S}_3$ in Figure 3. We compute part of the concurrent composition $\mathrm{CC_A}(\mathcal{S}_{3\mathsf{f}}, \mathcal{S}_{3\mathsf{n}})$ as in Figure 4. In $\mathrm{CC_A}(\mathcal{S}_{3\mathsf{f}}, \mathcal{S}_{3\mathsf{n}})$, one sees a run $(q_0, q_0) \xrightarrow{(e_1,e_1)} (q_1, q_2) \xrightarrow{(e_2,e_2)}$*

$(q_3,q_4) \xrightarrow{(f,\epsilon)} (q_5,q_4) \xrightarrow{(u,\epsilon)} (q_5,q_4)$, which satisfies (5). Then by Theorem 3.1, $\mathcal{S}_3$ is not $\{f\}$-diagnosable.

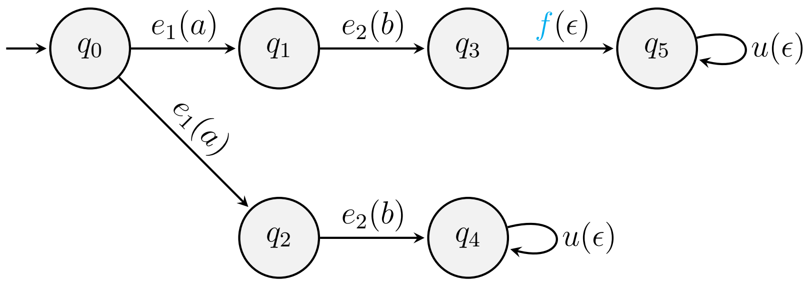


Figure 3: LFSA $\mathcal{S}_3$, where only event f is faulty.

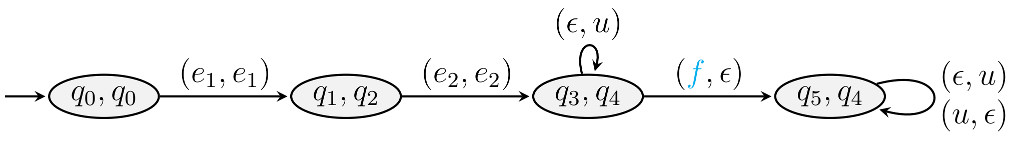


Figure 4: Part of $\mathrm{CC}_\mathrm{A}(\mathcal{S}_{3\mathsf{f}},\mathcal{S}_{3\mathsf{n}})$, where $\mathcal{S}_3$ is shown in Figure 3.

4 Predictability

Differently from diagnosability, *predictability* describes whether one can use an observed output sequence to make sure some faulty event will be certain to occur.

Definition 8 (Pred [6]) *Consider an LFSA $\mathcal{S}=(Q,E,\delta,Q_0,\Sigma,\ell)$ and a subset $E_f \subset E$ of faulty events. $\mathcal{S}$ is called E_f-*predictable *if*

$$(\exists k\in\mathbb{N})(\forall s\in L(\mathcal{S})\cap E^*E_f)(\exists s'\sqsubset s: E_f\notin s')(\forall uv\in L(\mathcal{S}))$$
$$[((\ell(s')=\ell(u))\wedge(E_f\notin u)\wedge(|v|>k))\implies(E_f\in v)].$$

Intuitively, if $\mathcal{S}$ is E_f-predictable, then once a faulty event will definitely occur, then before any faulty event occurs, one can make sure that after a common time delay (representing the number of occurrences of events, e.g., k), all generated event sequences with the same observation without any faulty event must be continued by an event sequence containing a faulty event, so as to raise an alarm to definite occurrence of some faulty event.

In order to verify E_f-predictability of $\mathcal{S}$, we use the self-composition $\mathrm{CC}_\mathrm{A}(\mathcal{S}_\mathsf{n},\mathcal{S}_\mathsf{n})$ of the normal subautomaton $\mathcal{S}_\mathsf{n}$. $\mathrm{CC}_\mathrm{A}(\mathcal{S}_\mathsf{n},\mathcal{S}_\mathsf{n})$ is also computed similarly as in Definition 2.

Theorem 4.1 ([34]) *Consider an LFSA $\mathcal{S}=(Q,E,\delta,Q_0,\Sigma,\ell)$ and a subset $E_f\subset E$ of faulty events. $\mathcal{S}$ is not E_f-predictable if and only if in $\mathrm{CC}_\mathrm{A}(\mathcal{S}_\mathsf{n},\mathcal{S}_\mathsf{n})$, there exists a run*

$$q_0' \xrightarrow{s_1'} q_1' \tag{6}$$

such that

$$q_0' \in Q_0'; \tag{7a}$$
$$(q_1'(L), e_f, q) \in \delta \text{ for some } e_f \in E_f \text{ and } q \in Q; \tag{7b}$$
$$\text{in } \mathcal{S}_{\mathsf{n}},\ \text{there is a transition cycle reachable from } q_1'(R). \tag{7c}$$

Proof By definition, $\mathcal{S}$ is not E_f-predictable if and only if

$$(\forall k \in \mathbb{N})(\exists s_k \in L(\mathcal{S}) \cap E^* E_f)(\forall s_k' \sqsubset s_k : E_f \notin s_k')(\exists u_k v_k \in L(\mathcal{S}))$$
$$[(\ell(s_k') = \ell(u_k)) \wedge (E_f \notin u_k v_k) \wedge (|v_k| > k)].$$

Choose sufficiently large k, by the finiteness of the number of states of $\mathcal{S}$ and the pigeonhole principle, $\mathcal{S}$ is not E_f-predictable if and only if in $\mathrm{CC_A}(\mathcal{S}_{\mathsf{n}}, \mathcal{S}_{\mathsf{n}})$, there exists a run (6) satisfying (7). □

Example 5 *Reconsider the LFSA $\mathcal{S}_3$ in Figure 3. Part of the concurrent composition $\mathrm{CC_A}(\mathcal{S}_{3\mathsf{n}}, \mathcal{S}_{3\mathsf{n}})$ can be obtained from Figure 4 by removing the transition with event (f, ϵ). Then in $\mathrm{CC_A}(\mathcal{S}_{3\mathsf{n}}, \mathcal{S}_{3\mathsf{n}})$, one sees a run $(q_0, q_0) \xrightarrow{(e_1,e_1)} (q_1, q_2) \xrightarrow{(e_2,e_2)} (q_3, q_4)$, which satisfies (7). Then by Theorem 4.1, $\mathcal{S}_3$ is not $\{f\}$-predictable.*

5 Standard opacity

Opacity is a concealment-based (confidentiality) property which was first proposed by Mazaré [35] in 2004. Opacity describes whether the visit of *secrets* in a system could be forbidden from being leaked to an external *intruder*, given that the intruder knows complete knowledge of the system's structure but can only see generated outputs. It has been widely used to describe all kinds of scenarios in cyber security/privacy problems such as the dinning cryptographers problem [36], encryption using pseudo random generators and tracking of mobile agents in sensor networks [37], ensuring privacy in location-based services [38], the indoor location privacy problem using obfuscation [39, 40, 41].

In [42], a general run-based opacity framework was proposed for labeled transition systems (which contain LFSAs, labeled Petri nets, etc., as subclasses), where such a system is opaque if for every secret run, there exists a non-secret run such that the two runs produce the same observation. Later on, two special types of secrets are studied: subsets of event sequences (aka traces) and subsets of states. According to the two types of secrets, opacity is classified into language-based opacity and state-based opacity. The former refers to for every secret generated trace, there is a non-secret generated trace such that they produce the same observation; the latter refers to whenever a run passes through a secret state at some instant, there exists another run that does not pass any secret state at the same instant such that the two runs produce the same observation. Language-based opacity is more involved, because it is already undecidable for LFSAs which contain no observable events [42]; particularly, when secret languages and non-secret languages are regular, language-based opacity is decidable in exponential time [43]. State-based opacity is relatively simpler. When the time instant of visiting secret states is specified as the initial time, the current time, any past time, and at most K steps prior to the current time, the notions of state-based opacity can be formulated as *initial-state* *opacity* (ISO) [44], *current-state opacity* (CSO) [11], *infinite-step opacity* (InfSO) [7], and K*-step opacity* (KSO) [45], respectively. The problems of verifying the four types of state-based opacity are PSPACE-complete in LFSAs [44, 11, 7], the four properties and the special case of language-based opacity studied in [43] are polynomially reducible to each other [13, 46].

Next, we show a concurrent-composition method to verify the four properties of state-based opacity. One can directly use an observer to verify CSO [11, 12] and directly use a reverse observer to verify ISO [13]. The verification methods in [11, 12, 13] are currently the most efficient methods for verifying CSO and ISO. However, verifying InfSO and KSO are more difficult, currently one cannot see any possibility of directly using an observer and a reverse observer to do their verification. The concurrent-composition method to be shown to verify InfSO and KSO is more efficient than the initial-state-estimator method [7] and the two-way-observer method (i.e., the alternating product of an observer and a reverse observer) [8].

Definition 9 (ISO [44]) *Consider an LFSA $\mathcal{S} = (Q, E, \delta, Q_0, \Sigma, \ell)$ and a subset $Q_S \subset Q$ of secret states. $\mathcal{S}$ is called* initial-state opaque *(ISO)* with respect to *Q_S if for every run $q_0 \xrightarrow{s} q$ with $q_0 \in Q_0 \cap Q_S$, there exists a run $q_0' \xrightarrow{s'} q'$ such that $q_0' \in Q_0 \setminus Q_S$ and $\ell(s) = \ell(s')$.*

From now on, ISO is short for “initial-state opacity” or “initial-state opaque” adapted to the context. Analogous for CSO, InfSO, and KSO.

Intuitively, if an LFSA is ISO, then an external intruder cannot make sure whether the initial state is secret by observing generated label sequences.

Definition 10 (CSO [11]) *Consider an LFSA $\mathcal{S} = (Q, E, \delta, Q_0, \Sigma, \ell)$ and a subset $Q_S \subset Q$ of secret states. $\mathcal{S}$ is called* current-state opaque *(CSO)* with respect to *Q_S if for every run $q_0 \xrightarrow{s} q$ with $q_0 \in Q_0$ and $q \in Q_S$, there exists a run $q_0' \xrightarrow{s'} q'$ such that $q_0' \in Q_0$, $q' \in Q \setminus Q_S$, and $\ell(s) = \ell(s')$.*

If an LFSA is CSO, then an external intruder cannot make sure whether the current state is secret by observing generated label sequences.

Definition 11 (InfSO [7]) *Consider an LFSA $\mathcal{S} = (Q, E, \delta, Q_0, \Sigma, \ell)$ and a subset $Q_S \subset Q$ of secret states. $\mathcal{S}$ is called* infinite-step opaque *(InfSO)* with respect to *Q_S if for every run $q_0 \xrightarrow{s_1} q_1 \xrightarrow{s_2} q_2$ with $q_0 \in Q_0$ and $q_1 \in Q_S$, there exists a run $q_0' \xrightarrow{s_1'} q_1' \xrightarrow{s_2'} q_2'$ such that $q_0' \in Q_0$, $q_1' \in Q \setminus Q_S$, $\ell(s_1) = \ell(s_1')$, and $\ell(s_2) = \ell(s_2')$.*

Definition 12 (KSO [45]) *Consider an LFSA $\mathcal{S} = (Q, E, \delta, Q_0, \Sigma, \ell)$, a subset $Q_S \subset Q$ of secret states, and a positive integer K. $\mathcal{S}$ is called* K-step opaque *(KSO)* with respect to *Q_S if for every run $q_0 \xrightarrow{s_1} q_1 \xrightarrow{s_2} q_2$ with $q_0 \in Q_0$, $q_1 \in Q_S$, and $|\ell(s_2)| \le K$, there exists a run $q_0' \xrightarrow{s_1'} q_1' \xrightarrow{s_2'} q_2'$ such that $q_0' \in Q_0$, $q_1' \in Q \setminus Q_S$, $\ell(s_1) = \ell(s_1')$, and $\ell(s_2) = \ell(s_2')$.*

If an LFSA is InfSO (KSO), then an external intruder cannot make sure whether any past state (at most K steps prior to the current time) is secret by observing generated label sequences.

In order to verify CSO, the notion of observer is enough. Observer is the classical powerset construction used for determinizing nondeterministic finite automata with ϵ-transitions.

Definition 13 ([2]) *Consider an LFSA $\mathcal{S} = (Q, E, \delta, Q_0, \Sigma, \ell)$. Its* observer $\mathcal{S}_{\text{obs}}$ *(the term "observer" was used in [15] and hereafter) is defined by a deterministic finite automaton*

$$(Q_{\text{obs}}, \ell(E_o), \delta_{\text{obs}}, q_{0\,\text{obs}}),$$

where

1. $Q_{\text{obs}} = 2^Q$,
2. $\ell(E_o) = \ell(E) \setminus \{\epsilon\}$,
3. *for all* $x \in Q_{\text{obs}}$ *and* $a \in \ell(E_o)$, $\delta_{\text{obs}}(x, a) = \bigcup_{q \in x} \bigcup_{\substack{e_a \in E_o \\ \ell(e_a)=a \\ s \in (E_{uo})^*}} \delta(q, e_a s)$,
4. $q_{0\,\text{obs}} = \bigcup_{q_0 \in Q_0} \bigcup_{s \in (E_{uo})^*} \delta(q_0, s)$ *(i.e.,* $\text{UR}(Q_0)$*).*

By definition, for all $a \in \ell(E_o)$, one has $\delta_{\text{obs}}(\emptyset, a) = \emptyset$. The size of $\mathcal{S}_{\text{obs}}$ is $O(2^{|Q|}|\ell(E_o)|)$, the time consumption of computing $\mathcal{S}_{\text{obs}}$ is $O(2^{|Q|}|Q|^2|\ell(E_o)||E|)$: for every subset $x \subset Q$ and every label $a \in \ell(E_o)$, the time consumption of computing $\delta(x, a)$ is bounded (from above) by $|Q|^2|E_o| + |Q|^2|E_{uo}| = |Q|^2|E|$.

Example 6 *Part of the observer $\mathcal{S}_{2\,\text{obs}}$ of the LFSA $\mathcal{S}_2$ in Figure 2 is shown in Figure 5.*

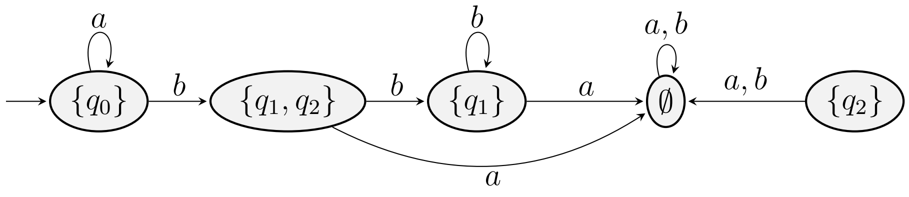


Figure 5: Part of observer $\mathcal{S}_{2\,\text{obs}}$ of the LFSA $\mathcal{S}_2$ in Figure 2.

Theorem 5.1 ([11, 12]) *An LFSA $\mathcal{S} = (Q, E, \delta, Q_0, \Sigma, \ell)$ is CSO with respect to $Q_S \subset Q$ if and only if for every nonempty state x reachable in $\mathcal{S}_{\text{obs}}$, $x \not\subset Q_S$.*

Example 7 *Reconsider the LFSA $\mathcal{S}_2$ in Figure 2 and its observer $\mathcal{S}_{2\,\text{obs}}$ in Figure 5. By Theorem 5.1, $\mathcal{S}_2$ is CSO with respect to $\{q_2\}$.*

We use the concurrent composition $\text{CC}_\text{A}(\mathcal{S}_\varepsilon, \mathcal{S}_{\text{obs}}^\varepsilon)$ to verify the other three notions of opacity, where $\mathcal{S}_\varepsilon$ is obtained from $\mathcal{S}$ by changing each transition $q \xrightarrow{e} q'$ to $q \xrightarrow{\ell(e)} q'$ if $e \in E_o$, to $q \xrightarrow{\varepsilon} q'$ if $e \in E_{uo}$, and replacing the labeling function of $\mathcal{S}$ by the map ℓ' on $\ell(E_o) \cup \{\varepsilon\}$ satisfying that $\ell'|_{\ell(E_o)}$ (the restriction of ℓ' to $\ell(E_o)$) is the identity map and $\ell'(\varepsilon) = \epsilon$; $\mathcal{S}_{\text{obs}}^\varepsilon$ is obtained from $\mathcal{S}_{\text{obs}}$ by adding an additional event ε and the labeling function ℓ' of $\mathcal{S}_\varepsilon$. In this particular case, $\text{CC}_\text{A}(\mathcal{S}_\varepsilon, \mathcal{S}_{\text{obs}}^\varepsilon)$ is almost the same as the parallel composition of $\mathcal{S}_\varepsilon$ and $\mathcal{S}_{\text{obs}}^\varepsilon$ in [47, Page 80].

The size of $\text{CC}_\text{A}(\mathcal{S}_\varepsilon, \mathcal{S}_{\text{obs}}^\varepsilon)$ is $O(|Q|2^{|Q|}(|\ell(E_o)||Q| + |E_{uo}||Q|)) = O(|Q|^2 2^{|Q|}|E|)$. The time consumption of computing $\text{CC}_\text{A}(\mathcal{S}_\varepsilon, \mathcal{S}_{\text{obs}}^\varepsilon)$ is also $O(|Q|^2 2^{|Q|}|E|)$ after $\mathcal{S}_\varepsilon$ and $\mathcal{S}_{\text{obs}}^\varepsilon$ have been computed.

Theorem 5.2 *An LFSA $\mathcal{S} = (Q, E, \delta, Q_0, \Sigma, \ell)$ is ISO with respect to $Q_S \subset Q$ if and only if $Q_0 \neq \emptyset \implies Q_0 \not\subset Q_S$ and for every $q_0 \in Q_0 \cap Q_S$, in concurrent composition $\mathrm{CC_A}(\mathcal{S}_\varepsilon, \mathcal{S}_{\mathrm{obs}}^\varepsilon)$, all states reachable from $(q_0, \mathrm{UR}(Q_0 \setminus Q_S))$ are of the form $(-, x)$ with $x \neq \emptyset$.*

Theorem 5.3 ([48]) *An LFSA $\mathcal{S} = (Q, E, \delta, Q_0, \Sigma, \ell)$ is InfSO with respect to $Q_S \subset Q$ if and only if for every nonempty state x reachable in $\mathcal{S}_{\mathrm{obs}}$, one has $x \not\subset Q_S$ and for every $q \in x \cap Q_S$, in concurrent composition $\mathrm{CC_A}(\mathcal{S}_\varepsilon, \mathcal{S}_{\mathrm{obs}}^\varepsilon)$, all states reachable from $(q, x \setminus Q_S)$ are of the form $(-, x')$ with $x' \neq \emptyset$.*

Theorem 5.4 ([48]) *An LFSA $\mathcal{S} = (Q, E, \delta, Q_0, \Sigma, \ell)$ is KSO with respect to $Q_S \subset Q$ if and only if for every nonempty state x reachable in $\mathcal{S}_{\mathrm{obs}}$, one has $x \not\subset Q_S$ and for every $q \in x \cap Q_S$, in concurrent composition $\mathrm{CC_A}(\mathcal{S}_\varepsilon, \mathcal{S}_{\mathrm{obs}}^\varepsilon)$, for every run $(q, x \setminus Q_S) \xrightarrow{s'} (q', x')$ with $|\ell(s')| \leq K$, $x' \neq \emptyset$.*

Theorems 5.1, 5.2, 5.3, and 5.4 directly follow from definition. By definition, one directly sees the following corollaries. If an LFSA $\mathcal{S}$ is KSO (with respect to $Q_S \subset Q$), then it is K'SO for any $K' < K$. Conversely, if $\mathcal{S}$ is not KSO with $K > 2^{|Q|} - 2$, then it is not K'SO for some $K' \leq 2^{|Q|} - 2$, because $\mathcal{S}_{\mathrm{obs}}$ has at most $2^{|Q|} - 1$ nonempty states; then it is not $(2^{|Q|} - 2)$SO. Hence the verification of KSO based on Theorem 5.4 does not depend on K if $K > 2^{|Q|} - 2$. The verification algorithms shown in Theorems 5.1, 5.2, 5.3, and 5.4 all run in time $O(2^{|Q|}|Q|^2|\ell(E_o)||E|)$. The upper bound $2^{|Q|} - 2$ for K was obtained in [8]. The upper bound for K obtained in [45] is $2^{|Q|^2} - 2$. Compared with the concurrent-composition method, the relative inefficiency of the two-way observer method [8] comes from computing a reverse observer (with the same complexity as computing an observer) and the alternating product (i.e., the so-called two-way observer) of the observer and the reverse observer. The verification algorithms obtained in [7, 45] have even higher complexity.

Corollary 5.5 ([48]) *An LFSA $\mathcal{S} = (Q, E, \delta, Q_0, \Sigma, \ell)$ is KSO with respect to $Q_S \subset Q$ if and only if it is $\min\{K, |2^{|Q|} - 2\}$SO with respect to Q_S.*

Corollary 5.6 *An LFSA $\mathcal{S} = (Q, E, \delta, Q_0, \Sigma, \ell)$ is InfSO with respect to $Q_S \subset Q$ if and only if it is KSO with respect to Q_S with $K > 2^{|Q|} - 2$.*

Example 8 *Reconsider the LFSA $\mathcal{S}_2$ in Figure 2 and its observer $\mathcal{S}_{2\,\mathrm{obs}}$ in Figure 5. The corresponding $\mathcal{S}_{2\varepsilon}$ is shown in Figure 6. The concurrent composition $\mathrm{CC_A}(\mathcal{S}_{2\varepsilon}, \mathcal{S}_{2\,\mathrm{obs}}^\varepsilon)$ is shown in Figure 7.*

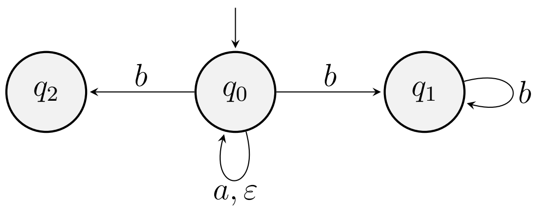


Figure 6: LFSA $\mathcal{S}_{2\varepsilon}$ corresponding to the LFSA $\mathcal{S}_2$ in Figure 2.

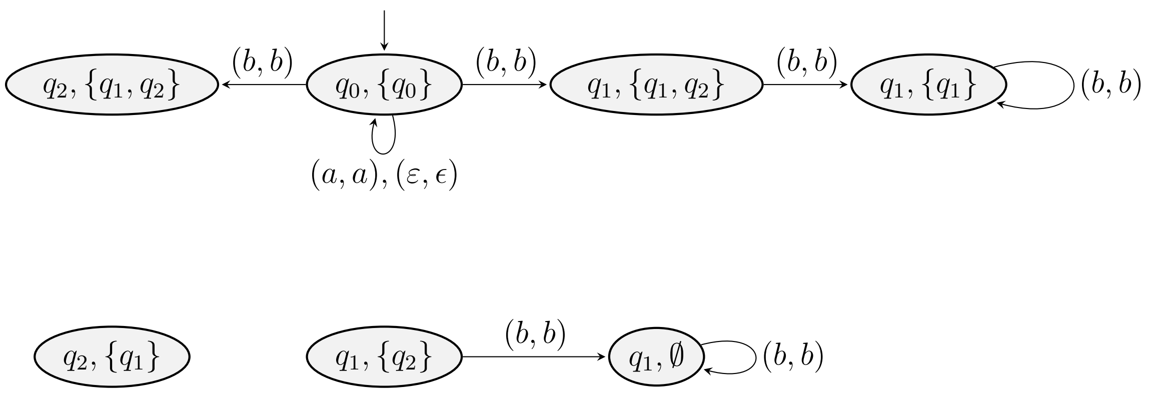


Figure 7: Part of $\mathrm{CC_A}(\mathcal{S}_{2\varepsilon}, \mathcal{S}_{2\,\mathrm{obs}}^{\varepsilon})$ corresponding to the LFSA $\mathcal{S}_2$ in Figure 2.

By Theorem 5.2, $\mathcal{S}_2$ is not ISO with respect to $\{q_0\}$, because q_0 is the unique initial state. In addition, one has $\mathcal{S}_2$ is InfSO with respect to $\{q_2\}$ by Theorem 5.3, because the unique reachable state of $\mathcal{S}_{2\,\mathrm{obs}}$ containing q_2 is $\{q_1, q_2\}$ and in $\mathrm{CC_A}(\mathcal{S}_{2\varepsilon}, \mathcal{S}_{2\,\mathrm{obs}}^{\varepsilon})$, there is no state reachable from $(q_2, \{q_1\})$ of the form $(-, \emptyset)$. By the reachable state $\{q_1\}$ of $\mathcal{S}_{2\,\mathrm{obs}}$, one sees $\mathcal{S}_2$ is not InfSO with respect to $\{q_1\}$, which can also be seen from the fact that $\{q_1, q_2\}$ is reachable in $\mathcal{S}_{2\,\mathrm{obs}}$ and in $\mathrm{CC_A}(\mathcal{S}_{2\varepsilon}, \mathcal{S}_{2\,\mathrm{obs}}^{\varepsilon})$, the state $(q_1, \emptyset)$ is reachable from $(q_1, \{q_2\})$.

6 Strong opacity

In Section 5, variants of notions of opacity were shown to describe the ability of an LFSA to forbid its visit of secret states from being leaked to an external intruder. Sometimes, such "standard" opacity is not sufficiently strong, e.g., in some CSO LFSA, when observing a generated label sequence, one can make sure that some secret state must have been visited, although cannot make sure of the exact visit instant of time. Consider the following LFSA $\mathcal{S}_4$:

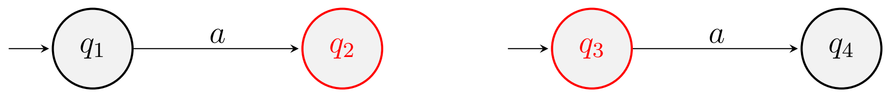


Figure 8: LSFA $\mathcal{S}_4$, where $\ell(a) = a$, q_2 and q_3 are secret, q_1 and q_4 are not.

Automaton $\mathcal{S}_4$ is CSO with respect to $\{q_2, q_3\}$. When observing a, one can make sure that at least one secret state has been visited, in detail, if $q_1 \xrightarrow{a} q_2$ was generated then q_2 was visited, if $q_3 \xrightarrow{a} q_4$ was generated then q_3 was visited. This leads to a "strong version" of CSO which guarantees that an intruder cannot make sure whether the current state is secret, and can also guarantee that the intruder cannot make sure whether some secret state has been visited. Analogously, the other three standard versions of opacity studied in Section 5 could also be reformulated as their strong versions.

In order to define strong versions of state-based opacity, we define a *non-secret* run of an LFSA by a run containing no secret states.

Definition 14 (SISO [49]) *Consider an LFSA $\mathcal{S} = (Q, E, \delta, Q_0, \Sigma, \ell)$ and a subset $Q_S \subset Q$ of secret states. $\mathcal{S}$ is called* strongly initial-state opaque *(SISO)* with respect to Q_S *if for every run $q_0 \xrightarrow{s} q$ with $q_0 \in Q_0 \cap Q_S$, there exists a* non-secret *run $q_0' \xrightarrow{s'} q'$ such that $q_0' \in Q_0 \setminus Q_S$ and $\ell(s) = \ell(s')$.*

If an LFSA is SISO, then an external intruder cannot make sure whether the initial state is secret and cannot make sure whether some secret state has been visited either, by observing generated label sequences.

Definition 15 (SCSO [49]) *Consider an LFSA $\mathcal{S} = (Q, E, \delta, Q_0, \Sigma, \ell)$ and a subset $Q_S \subset Q$ of secret states. $\mathcal{S}$ is called* strongly current-state opaque *(SCSO)* with respect to Q_S *if for every run $q_0 \xrightarrow{s} q$ with $q_0 \in Q_0$ and $q \in Q_S$, there exists a* non-secret *run $q_0' \xrightarrow{s'} q'$ such that $q_0' \in Q_0$ and $\ell(s) = \ell(s')$.*

If an LFSA is SCSO, then an external intruder cannot make sure whether the current state is secret and cannot make sure whether some secret state has been visited either, by observing generated label sequences.

Definition 16 (SInfSO [10]) *Consider an LFSA $\mathcal{S} = (Q, E, \delta, Q_0, \Sigma, \ell)$ and a subset $Q_S \subset Q$ of secret states. $\mathcal{S}$ is called* strongly infinite-step opaque *(SInfSO)* with respect to Q_S *if for every run $q_0 \xrightarrow{s_1} q_1 \xrightarrow{s_2} q_2$ with $q_0 \in Q_0$ and $q_1 \in Q_S$, there exists a* non-secret *run $q_0' \xrightarrow{s_1'} q_1' \xrightarrow{s_2'} q_2'$ such that $q_0' \in Q_0$, $\ell(s_1) = \ell(s_1')$, and $\ell(s_2) = \ell(s_2')$.*

Definition 17 (SKSO) *Consider an LFSA $\mathcal{S} = (Q, E, \delta, Q_0, \Sigma, \ell)$, a subset $Q_S \subset Q$ of secret states, and a positive integer K. $\mathcal{S}$ is called* strongly K-step opaque[3] *(SKSO)* with respect to Q_S *if for every run $q_0 \xrightarrow{s_1} q_1 \xrightarrow{s_2} q_2$ with $q_0 \in Q_0$, $q_1 \in Q_S$, and $|\ell(s_2)| \le K$, there exists a* non-secret *run $q_0' \xrightarrow{s_1'} q_1' \xrightarrow{s_2'} q_2'$ such that $q_0' \in Q_0$, $\ell(s_1) = \ell(s_1')$, and $\ell(s_2) = \ell(s_2')$.*

From now on, SISO is short for "strong initial-state opacity" or "strongly initial-state opaque" adapted to the context. Analogous for SCSO, SInfSO, and SKSO.

If an LFSA is SInfSO (SKSO), then an external intruder cannot make sure whether any past state (at most K steps prior to the current time) is secret and cannot make sure whether some secret state has been visited either, by observing generated label sequences.

Next we use the concurrent-composition structure to do verification for the four strong versions of state-based opacity, where the derived verification algorithms are more efficient than the K/Inf-step recognizer method proposed in [10].

Consider an LFSA $\mathcal{S} = (Q, E, \delta, Q_0, \Sigma, \ell)$ and a subset $Q_S \subset Q$ of secret states, let

$$\mathcal{S}_{\text{dss}} = (Q_{\text{dss}}, E_{\text{dss}}, \delta_{\text{dss}}, Q_{0\,\text{dss}}, \Sigma, \ell_{\text{dss}})$$

be the accessible part of the remainder of $\mathcal{S}$ by *deleting secret states* (dss) of $\mathcal{S}$. Let

$$\mathcal{S}_{\text{dss obs}} = (Q_{\text{dss obs}}, \ell_{\text{dss}}(E_{\text{dss}}) \setminus \{\epsilon\}, \delta_{\text{dss obs}}, q_{0\,\text{dss obs}})$$

[3]Note that the current SKSO is slightly stronger than the K-step strong opacity proposed in [9], where in the latter, $q_0' \xrightarrow{s_1'} q_1' \xrightarrow{s_2'} q_2'$ is not necessarily non-secret, but only $q_1' \xrightarrow{s_2'} q_2'$ is necessarily non-secret.

be the observer of $\mathcal{S}_{\text{dss}}$.

Similarly to $\mathcal{S}_{\text{obs}}$, the size of $\mathcal{S}_{\text{dss obs}}$ is $O(2^{|Q|}|\ell(E_o)|)$, the time consumption of computing $\mathcal{S}_{\text{dss obs}}$ is $O(2^{|Q|}|Q|^2|\ell(E_o)||E|)$. The size of $\mathcal{S}_{\text{dss obs}}$ is slightly smaller than that of $\mathcal{S}_{\text{obs}}$.

We will use the concurrent composition $\text{CC}_{\text{A}}(\mathcal{S}_\varepsilon, \mathcal{S}^\varepsilon_{\text{dss obs}})$ to verify the four strong versions of opacity. Unlike CSO and ISO, SCSO and SISO cannot be verified by directly using the notions of observer and reverse observer.

Similarly to $\text{CC}_{\text{A}}(\mathcal{S}_\varepsilon, \mathcal{S}^\varepsilon_{\text{obs}})$, the size of $\text{CC}_{\text{A}}(\mathcal{S}_\varepsilon, \mathcal{S}^\varepsilon_{\text{dss obs}})$ is $O(|Q|^2 2^{|Q|}|E|)$. The time consumption of computing $\text{CC}_{\text{A}}(\mathcal{S}_\varepsilon, \mathcal{S}^\varepsilon_{\text{dss obs}})$ is also $O(|Q|^2 2^{|Q|}|E|)$ after $\mathcal{S}_\varepsilon$ and $\mathcal{S}^\varepsilon_{\text{dss obs}}$ have been computed.

Theorem 6.1 ([49]) *An LFSA $\mathcal{S} = (Q, E, \delta, Q_0, \Sigma, \ell)$ is SCSO with respect to $Q_S \subset Q$ if and only if for every state (q, x) reachable in $\text{CC}_{\text{A}}(\mathcal{S}_\varepsilon, \mathcal{S}^\varepsilon_{\text{dss obs}})$, if $q \in Q_S$ then $x \neq \emptyset$.*

Theorem 6.2 ([49]) *An LFSA $\mathcal{S} = (Q, E, \delta, Q_0, \Sigma, \ell)$ is SISO with respect to $Q_S \subset Q$ if and only if $Q_0 \neq \emptyset \implies Q_0 \not\subset Q_S$ and for every $q_0 \in Q_0 \cap Q_S$, in concurrent composition $\text{CC}_{\text{A}}(\mathcal{S}_\varepsilon, \mathcal{S}^\varepsilon_{\text{dss obs}})$, all states reachable from $(q_0, q_{0\,\text{dss obs}})$ are of the form $(-, x)$ with $x \neq \emptyset$.*

Theorem 6.3 ([49]) *An LFSA $\mathcal{S} = (Q, E, \delta, Q_0, \Sigma, \ell)$ is SInfSO with respect to $Q_S \subset Q$, if and only if, (i) for every state (q, x) reachable in concurrent composition $\text{CC}_{\text{A}}(\mathcal{S}_\varepsilon, \mathcal{S}^\varepsilon_{\text{dss obs}})$, if $q \in Q_S$ then $x \neq \emptyset$ and all states reachable from (q, x) are of the form $(-, x')$ with $x' \neq \emptyset$, if and only if, (ii) all states (q'', x'') reachable in $\text{CC}_{\text{A}}(\mathcal{S}_\varepsilon, \mathcal{S}^\varepsilon_{\text{dss obs}})$ satisfy $x'' \neq \emptyset$.*

Proof By definition, (i) is equivalent for $\mathcal{S}$ to be SInfSO with respect to Q_S.

(ii) $\implies$ (i): This trivially holds.

(i) $\implies$ (ii): Consider a state (q'', x'') reachable in $\text{CC}_{\text{A}}(\mathcal{S}_\varepsilon, \mathcal{S}^\varepsilon_{\text{dss obs}})$. If there is a run from some initial state of $\text{CC}_{\text{A}}(\mathcal{S}_\varepsilon, \mathcal{S}^\varepsilon_{\text{dss obs}})$ to (q'', x'') containing a state (q, x) with $q \in Q_S$, then by (i), one has $x'' \neq \emptyset$; otherwise one also has $x'' \neq \emptyset$ because $q'' \in x''$ by definition of $\text{CC}_{\text{A}}(\mathcal{S}_\varepsilon, \mathcal{S}^\varepsilon_{\text{dss obs}})$. $\square$

Theorem 6.4 *An LFSA $\mathcal{S} = (Q, E, \delta, Q_0, \Sigma, \ell)$ is SKSO with respect to $Q_S \subset Q$ if and only if for every state (q, x) reachable in concurrent composition $\text{CC}_{\text{A}}(\mathcal{S}_\varepsilon, \mathcal{S}^\varepsilon_{\text{dss obs}})$, if $q \in Q_S$ then $x \neq \emptyset$ and for every run $(q, x) \xrightarrow{s'} (q', x')$ with $|\ell(s')| \le K$, $x' \neq \emptyset$.*

Similarly to the standard versions of opacity, by definition, one also directly sees the following corollaries, because $\mathcal{S}_{\text{dss obs}}$ has at most $2^{|Q \setminus Q_S|} - 1$ nonempty states. Hence the verification of SKSO based on Theorem 6.4 does not depend on K if $K > 2^{|Q\setminus Q_S|} - 2$. The verification algorithms shown in Theorems 6.1, 6.2, 6.3, and 6.4 all run in time $O(2^{|Q|}|Q|^2|\ell(E_o)||E|)$.

Corollary 6.5 *An LFSA $\mathcal{S} = (Q, E, \delta, Q_0, \Sigma, \ell)$ is SKSO with respect to $Q_S \subset Q$ if and only if it is S$\min\{K, 2^{|Q\setminus Q_S|} - 2\}$SO with respect to Q_S.*

Corollary 6.6 *An LFSA $\mathcal{S} = (Q, E, \delta, Q_0, \Sigma, \ell)$ is SInfSO with respect to $Q_S \subset Q$ if and only if it is SKSO with respect to Q_S and positive integer K with $K > 2^{|Q\setminus Q_S|} - 2$.*

Remark 1 *The main time consumption in the verification algorithms shown in Theorems 5.1, 5.2, 5.3, 5.4, 6.1, 6.2, 6.3, and 6.4 comes from computing the corresponding observer. If the observer is not explicitly computed, then by using nondeterministic search, verification can be done in* PSPACE.

Example 9 *Consider the following LFSA $\mathcal{S}_5$:*

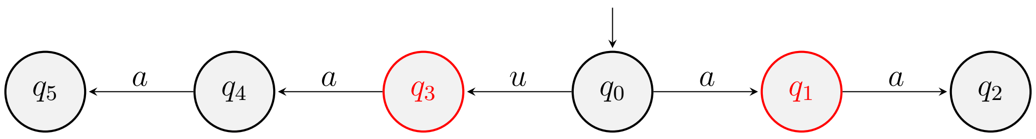


Figure 9: LSFA $\mathcal{S}_5$, where $\ell(u) = \epsilon$, $\ell(a) = a$, q_1 and q_3 are secret, the other states are not.

We verify whether $\mathcal{S}_5$ is InfSO or SInfSO with respect to $\{q_1, q_3\}$ by Theorem 5.3 and Theorem 6.3. By Theorem 5.3, we compute $\mathcal{S}_{5\varepsilon}$, $\mathcal{S}_{5\,\mathrm{obs}}$, and $\mathrm{CC_A}(\mathcal{S}_{5\varepsilon}, \mathcal{S}^{\varepsilon}_{5\,\mathrm{obs}})$ as follows:

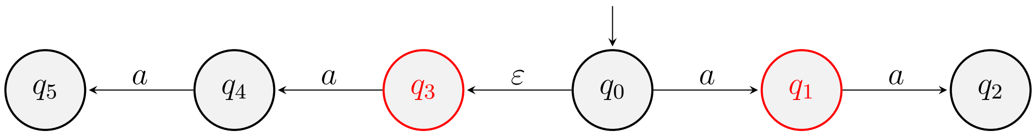


Figure 10: $\mathcal{S}_{5\varepsilon}$ corresponding to the LFSA $\mathcal{S}_5$ in Figure 9.

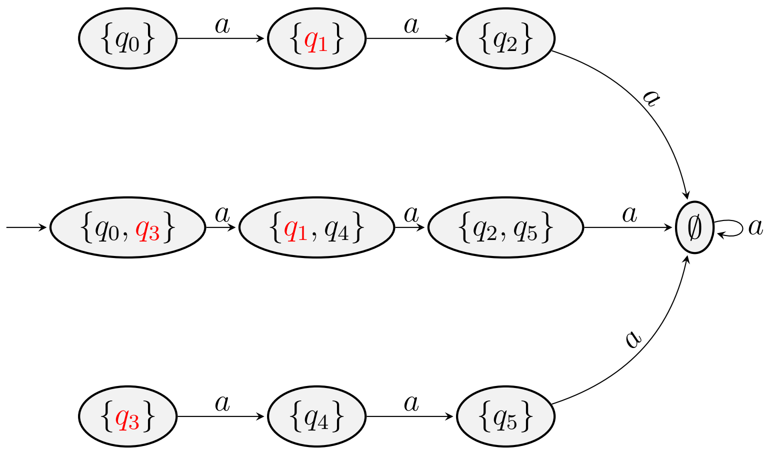


Figure 11: Part of $\mathcal{S}_{5\,\mathrm{obs}}$ corresponding to the LFSA $\mathcal{S}_5$ in Figure 9.

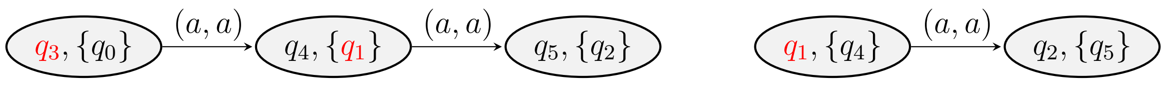


Figure 12: Part of $\mathrm{CC_A}(\mathcal{S}_{5\varepsilon}, \mathcal{S}^{\varepsilon}_{5\,\mathrm{obs}})$ corresponding to the LFSA $\mathcal{S}_5$ in Figure 9.

In observer $\mathcal{S}_{5\,\mathrm{obs}}$, the reachable states containing secret states are $\{q_0, q_3\}$ and $\{q_1, q_4\}$. In $\mathrm{CC_A}(\mathcal{S}_{5\varepsilon}, \mathcal{S}^{\varepsilon}_{5\,\mathrm{obs}})$, the states reachable from $(q_3, \{q_0\})$ and $(q_1, \{q_4\})$ all satisfy that their right components are nonempty. Then by Theorem 5.3, $\mathcal{S}_5$ is InfSO with respect to $\{q_1, q_3\}$.

By Theorem 6.3, we compute $\mathcal{S}_{5\,\mathrm{dss}}$, $\mathcal{S}_{5\,\mathrm{dss\,obs}}$, and $\mathrm{CC_A}(\mathcal{S}_{5\varepsilon}, \mathcal{S}^{\varepsilon}_{5\,\mathrm{dss\,obs}})$ as follows:

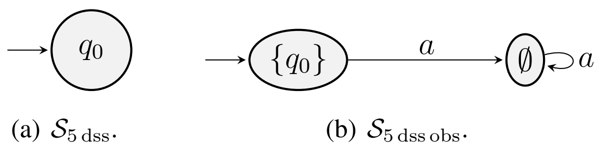

q_0
$\{q_0\}$
a
$\emptyset$
a
(a) $\mathcal{S}_{5\,\mathrm{dss}}$.
(b) $\mathcal{S}_{5\,\mathrm{dss\,obs}}$.

Figure 13: $\mathcal{S}_{5\,\mathrm{dss}}$ and $\mathcal{S}_{5\,\mathrm{dss\,obs}}$ corresponding to the LFSA $\mathcal{S}_5$ in Figure 9.

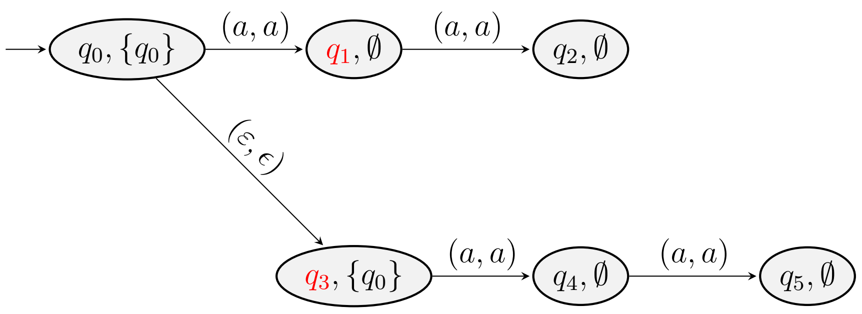


Figure 14: Part of $\mathrm{CC_A}(\mathcal{S}_{5\varepsilon}, \mathcal{S}^{\varepsilon}_{5\,\mathrm{dss\,obs}})$ (all reachable states illustrated) corresponding to the LFSA $\mathcal{S}_5$ in Figure 9.

In $\mathrm{CC_A}(\mathcal{S}_{5\varepsilon}, \mathcal{S}^{\varepsilon}_{5\,\mathrm{dss\,obs}})$*, there exist reachable states whose right components are equal to* $\emptyset$*, then by Theorem 6.3,* $\mathcal{S}_5$ *is not SInfSO with respect to* $\{q_1, q_3\}$.

7 Conclusion

In this paper, a unified concurrent-composition method was given to verify inference-based properties and concealment-based properties in labeled finite-state automata. Compared with the previous verification algorithms in the literature, the concurrent-composition method does not depend on assumptions and is more efficient. These results for the first time showed that many inference-based properties and concealment-based properties can be unified into one mathematical framework, although the two categories of properties look quite different. This similarity between the two categories has never been revealed before. It is interesting to explore other usages of the concurrent-composition method, e.g., what other properties could be verified by the method, what other kinds of models in discrete-event systems could be dealt with by the method, and what other problems (e.g., enforcement) can be solved by the method.

References

[1] W.M. Wonham and K. Cai. *Supervisory Control of Discrete-Event Systems*. Springer International Publishing, 2019.

[2] M. Sipser. *Introduction to the Theory of Computation*. International Thomson Publishing, 1st edition, 1996.

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

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

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

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

[7] A. Saboori and C. N. Hadjicostis. Verification of infinite-step opacity and complexity considerations. *IEEE Transactions on Automatic Control*, 57(5):1265–1269, May 2012.

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

[9] Y. Falcone and H. Marchand. Enforcement and validation (at runtime) of various notions of opacity. *Discrete Event Dyn. Sys.: Theory & Apl.*, 25:531–570, 2015.

[10] Z. Ma, X. Yin, and Z. Li. Verification and enforcement of strong infinite- and k-step opacity using state recognizers. *Automatica*, 133:109838, 2021.

[11] F. Cassez, J. Dubreil, and H. Marchand. Dynamic observers for the synthesis of opaque systems. In *Automated Technology for Verification and Analysis*, pages 352–367, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.

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

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

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

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

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

[17] E.F. Moore. Gedanken-experiments on sequential machines. *Automata Studies, Annals of Math. Studies*, 34:129–153, 1956.

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

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

[20] W.M. Wonham. *Linear Multivariable Control: a Geometric Approach, 3rd Ed.* Springer-Verlag New York, 1985.

[21] E.D. Sontag. On the observability of polynomial systems, I: Finite-time problems. *SIAM Journal on Control and Optimization*, 17:139–151, 1979.

[22] G. Conte, C.H. Moog, and A.M. Perdon. *Algebraic Methods for Nonlinear Control Systems, 2nd Ed.* Springer-Verlag London, 2007.

[23] A. Isidori. *Nonlinear Control Systems*. Communications and Control Engineering. Springer-Verlag London, 1995.

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

[25] A. Y. Kibangou, F. Garin, and S. Gracy. Input and state observability of network systems with a single unknown Input. *IFAC-PapersOnLine*, 49(22):37–42, 2016. 6th IFAC Workshop on Distributed Estimation and Control in Networked Systems NECSYS 2016.

[26] M. T. Angulo, A. Aparicio, and C. H. Moog. Structural accessibility and structural observability of nonlinear networked systems. *IEEE Transactions on Network Science and Engineering*, page online, 2019.

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

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

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

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

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

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

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

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

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

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

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

[38] Y. Wu. *Verification and Enforcement of Opacity Security Properties in Discrete Event Systems*. PhD thesis, University of Michigan, 2014.

[39] Y. Wu, V. Raman, B.C. Rawlings, S. Lafortune, and S.A. Seshia. Synthesis of obfuscation policies to ensure privacy and utility. *Journal of Automated Reasoning*, 60(1):107–131, 2018.

[40] Y. Wu, V. Raman, S. Lafortune, and S.A. Seshia. Obfuscator synthesis for privacy and utility. In Sanjai Rayadurgam and Oksana Tkachuk, editors, *NASA Formal Methods*, pages 133–149, Cham, 2016. Springer International Publishing.

[41] R.M. Góes, B.C. Rawlings, N. Recker, G. Willett, and S. Lafortune. Demonstration of indoor location privacy enforcement using obfuscation. *IFAC-PapersOnLine*, 51(7):145–151, 2018. 14th IFAC Workshop on Discrete Event Systems WODES 2018.

[42] J. W. Bryans, M. Koutny, L. Mazaré, and P. Y. A. Ryan. Opacity generalised to transition systems. *International Journal of Information Security*, 7(6):421–435, Nov 2008.

[43] F. Lin. Opacity of discrete event systems and its applications. *Automatica*, 47(3):496–503, March 2011.

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

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

[46] J. Balun and T. Masopust. Comparing the notions of opacity for discrete-event systems. *Discrete Event Dynamic Systems*, 2021.

[47] C.G. Cassandras and S. Lafortune. *Introduction to Discrete Event Systems*. Springer Publishing Company, Incorporated, 2nd edition, 2010.

[48] J. Balun and T. Masopust. K-step opacity in discrete event systems: Verification, complexity, and relations. https://arxiv.org/abs/2109.02158.

[49] X. Han, K. Zhang, J. Zhang, Z. Li, and Z. Chen. Strong current-state and initial-state opacity of discrete-event systems. https://arxiv.org/abs/2109.05475.

Appendix

We briefly review the twin-plant method proposed in [4] and the verifier method proposed in [5] used for verifying diagnosability and show that they usually do not work without the two assumptions of liveness/deadlock-freeness and divergence-freeness. We also briefly show the coincident similarity between the concurrent composition and the generalized version of the twin plant proposed in [33].

For brevity, we consider an LFSA $\mathcal{S} = (Q, E, \delta, Q_0, \Sigma, \ell)$ in which $\ell|_{E_o}$ is the identity mapping and Q_0 is a singleton and denoted by $\{q_0\}$. Recall that E_o is the set of observable events, and $E_{uo} = E \setminus E_o$ is the set of unobservable events. Consider a single faulty event $f \in E_{uo}$.

The twin plant $\mathrm{TwPl}_{\mathcal{S}}$ of $\mathcal{S}$ proposed in [4] is constructed as follows:

(1) Construct the automaton $\mathcal{S}_\phi = (Q_\phi, E_o, (x_0, \phi), \delta_\phi)$, where the initial state is $(x_0, \phi) \in Q_\phi$, $Q_\phi = Q \times \{\phi, F\}$; for all $(x_1, \phi), (x_2, l_2) \in Q_\phi$ and $t \in E_o$, $((x_1, \phi), t, (x_2, l_2)) \in \delta_\phi$ if and only if there is a run $x_1 \xrightarrow{st} x_2$ in $\mathcal{S}$ such that $s \in (E_{uo})^*$, and $l_2 = F$ if and only if f appears in at least one such s; for all $(x_1, F), (x_2, l_2) \in Q_\phi$ and $t \in E_o$, $((x_1, F), t, (x_2, l_2)) \in \delta_\phi$ if and only if there is a run $x_1 \xrightarrow{st} x_2$ in $\mathcal{S}$ such that $s \in E_{uo}^*$ and $l_2 = F$[4].

(2) The twin plant $\mathrm{TwPl}_{\mathcal{S}}$ is the parallel composition $\mathcal{S}_\phi || \mathcal{S}_\phi$ of $\mathcal{S}_\phi$ with itself, where the parallel composition is as in [47, Page 80]. In this special case, $\mathcal{S}_\phi || \mathcal{S}_\phi$ is almost the same as the self-composition $\mathrm{CC_A}(\mathcal{S}_\phi)$ because $\mathcal{S}_\phi$ contains no unobservable events. After replacing each event e_o in $\mathcal{S}_\phi || \mathcal{S}_\phi$ by (e_o, e_o), $\mathrm{CC_A}(\mathcal{S}_\phi)$ is obtained.

Proposition 7.1 ([4]) *A live and divergence-free $\mathcal{S}$ is $\{f\}$-diagnosable if and only if in $\mathrm{TwPl}_{\mathcal{S}}$ all states of all cycles are of the form $((-, l_1), (-, l_2))$ with $l_1 = l_2$.*

Example 10 *Proposition 7.1 does not generally hold for $\mathcal{S}$ that is not live or divergence-free. Consider the following LFSA $\mathcal{S}_6$:*

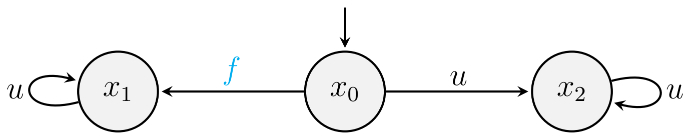


Figure 15: LFSA $\mathcal{S}_6$, where $\ell(f) = \ell(u) = \epsilon$.

By definition, $\mathcal{S}_{6\phi}$ consists of only the initial state (x_0, ϕ). Hence by Proposition 7.1, $\mathcal{S}_6$ is $\{f\}$-diagnosable vacuously. However, by definition, $\mathcal{S}_6$ is not $\{f\}$-diagnosable.

[4]Here F denotes propagation of f, i.e., along every run of $\mathcal{S}_\phi$, once a state has its right component equal to F, then all subsequent states have their right components equal to F.

The verifier $\mathrm{Ver}_{\mathcal{S}} = (Q_{\mathrm{Ver}}, E, (x_0, N, x_0, N), \delta_{\mathrm{Ver}})$ of $\mathcal{S}$ proposed in [5] is constructed as follows: $Q_{\mathrm{Ver}} = Q \times \{N, F\} \times Q \times \{N, F\}$, for all $(x_1, l_1, x_2, l_2) \in Q_{\mathrm{Ver}}$, $\sigma_o \in E_o$, and $\sigma_{uo} \in E_{uo} \setminus \{f\}$,

(i) $((x_1, l_1, x_2, l_2), f, (x_1', F, x_2, l_2)) \in \delta_{\mathrm{Ver}}$ if and only if $(x_1, f, x_1') \in \delta$,

(ii) $((x_1, l_1, x_2, l_2), f, (x_1, l_1, x_2', F)) \in \delta_{\mathrm{Ver}}$ if and only if $(x_2, f, x_2') \in \delta$,

(iii) $((x_1, l_1, x_2, l_2), f, (x_1', F, x_2', F)) \in \delta_{\mathrm{Ver}}$ if and only if $(x_1, f, x_1'), (x_2, f, x_2') \in \delta$,

(iv) $((x_1, l_1, x_2, l_2), \sigma_{uo}, (x_1', l_1, x_2, l_2)) \in \delta_{\mathrm{Ver}}$ if and only if $(x_1, \sigma_{uo}, x_1') \in \delta$,

(v) $((x_1, l_1, x_2, l_2), \sigma_{uo}, (x_1, l_1, x_2', l_2)) \in \delta_{\mathrm{Ver}}$ if and only if $(x_2, \sigma_{uo}, x_2') \in \delta$,

(vi) $((x_1, l_1, x_2, l_2), \sigma_{uo}, (x_1', l_1, x_2', l_2)) \in \delta_{\mathrm{Ver}}$ if and only if $(x_1, \sigma_{uo}, x_1'), (x_2, \sigma_{uo}, x_2') \in \delta$,

(vii) $((x_1, l_1, x_2, l_2), \sigma_o, (x_1', l_1, x_2', l_2)) \in \delta_{\mathrm{Ver}}$ if and only if $(x_1, \sigma_o, x_1'), (x_2, \sigma_o, x_2') \in \delta$.

Proposition 7.2 ([5]) *A live and divergence-free $\mathcal{S}$ is $\{f\}$-diagnosable if and only if in $\mathrm{Ver}_{\mathcal{S}}$ all states of all cycles are of the form $(-, l_1, -, l_2)$ with $l_1 = l_2$*[5].

Example 11 *Proposition 7.2 does not generally hold for $\mathcal{S}$ that is not live or divergence-free. Consider the following LFSA $\mathcal{S}_7$:*

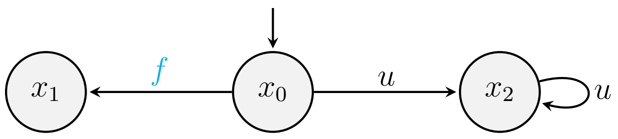


Figure 16: LFSA $\mathcal{S}_7$, where $\ell(f) = \ell(u) = \epsilon$.

Part of $\mathrm{Ver}_{\mathcal{S}_7}$ is shown as follows:

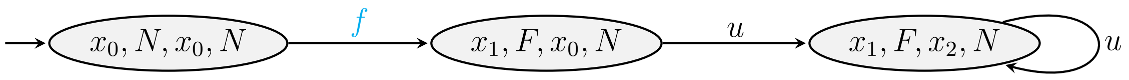


Figure 17: Part of $\mathrm{Ver}_{\mathcal{S}_7}$ corresponding to the LFSA $\mathcal{S}_7$ in Figure 16.

The cycle $(x_1, F, x_2, N) \xrightarrow{u} (x_1, F, x_2, N)$ in $\mathrm{Ver}_{\mathcal{S}_7}$ contradicts the condition in Proposition 7.2, hence by Proposition 7.2 $\mathcal{S}_7$ is not $\{f\}$-diagnsoable. However, by definition, $\mathcal{S}_7$ is $\{f\}$-diagnosable vacuously.

The generalized twin plant

$$\overline{\mathrm{TwPl}}_{\mathcal{S}} = (Q_{\mathrm{Ver}}, \{(\sigma_o, \sigma_o) | \sigma_o \in E_o\} \cup (T_{uo} \times \{\epsilon\}) \cup (\{\epsilon\} \times (E_{uo} \setminus \{f\})), (x_0, N, x_0, N), \delta_{\overline{\mathrm{TwPl}}})$$

of $\mathcal{S}$ proposed in [33] is constructed as follows: for all $(x_1, l_1, x_2, l_2) \in Q_{\mathrm{Ver}}$, (σ_o, σ_o) with $\sigma_o \in E_o$, and $\sigma_{uo} \in E_{uo} \setminus \{f\}$,

[5]In $\mathrm{Ver}_{\mathcal{S}}$, if there is a cycle containing a state of the form $(-, l_1, -, l_2)$ with $l_1 \neq l_2$, then either (1) all states in the cycle are of the form $(-, F, -, N)$ or (2) all states in the cycle are of the form $(-, N, -, F)$.

(a) $((x_1, l_1, x_2, l_2), (f, \epsilon), (x_1', F, x_2, l_2)) \in \delta_{\text{Ver}}$ if and only if $(x_1, f, x_1') \in \delta$,

(b) $((x_1, l_1, x_2, l_2), (\sigma_{uo}, \epsilon), (x_1', l_1, x_2, l_2)) \in \delta_{\text{Ver}}$ if and only if $(x_1, \sigma_{uo}, x_1') \in \delta$,

(c) $((x_1, l_1, x_2, l_2), (\epsilon, \sigma_{uo}), (x_1, l_1, x_2', l_2)) \in \delta_{\text{Ver}}$ if and only if $(x_2, \sigma_{uo}, x_2') \in \delta$,

(d) $((x_1, l_1, x_2, l_2), (\sigma_o, \sigma_o), (x_1', l_1, x_2', l_2)) \in \delta_{\text{Ver}}$ if and only if $(x_1, \sigma_o, x_1'), (x_2, \sigma_o, x_2') \in \delta$.

There is no state of the form $(-, -, -, F)$ reachable in $\overline{\text{TwPl}}_{\mathcal{S}}$.

Proposition 7.3 ([33]) *An $\mathcal{S}$ is not $\{f\}$-diagnosable if and only if in $\overline{\text{TwPl}}_{\mathcal{S}}$ there is a reachable cycle in which all states are of the form $(-, F, -, N)$ and there is at least one event of the form $(\sigma, -)$ with $\sigma \in E$.*

Example 12 *Consider $\mathcal{S}_7$ in Figure 16. $\overline{\text{TwPl}}_{\mathcal{S}_7}$ is shown as follows:*

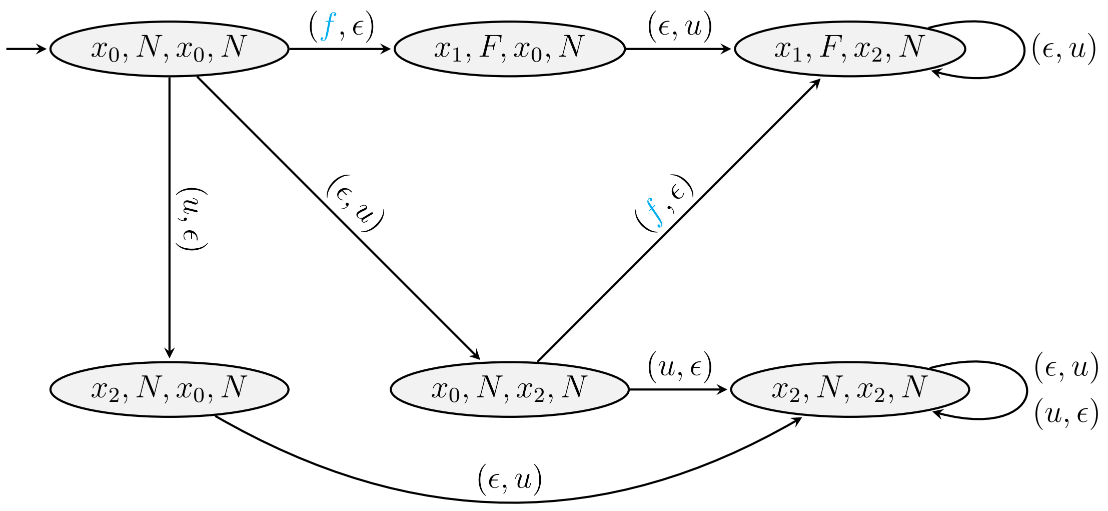


Figure 18: Part of $\overline{\text{TwPl}}_{\mathcal{S}_7}$ corresponding to the LFSA $\mathcal{S}_7$ in Figure 16.

In Figure 18, there is no reachable cycle as in Proposition 7.3. In the cycle $(x_1, F, x_2, N) \xrightarrow{(\epsilon,u)} (x_1, F, x_2, N)$, in the unique event (ϵ, u), the left component is ϵ. Hence by Proposition 7.3 $\mathcal{S}_7$ is $\{f\}$-diagnosable.

It is easy to see that the concurrent composition $\text{CC}_\text{A}(\mathcal{S}_\mathsf{f}, \mathcal{S}_\mathsf{n})$ in the current paper used for verifying diagnosability is coincidently similar to the generalized version of twin plant $\overline{\text{TwPl}}_{\mathcal{S}}$ proposed in [33]. After removing all F's and N's from $\overline{\text{TwPl}}_{\mathcal{S}}$, $\text{CC}_\text{A}(\mathcal{S}_\mathsf{f}, \mathcal{S}_\mathsf{n})$ is obtained.