Diagnosability Verification for Automata and Petri Nets: Can We Do Better? ⋆

Shaowen Miao [a,b], Petr Jančar [c], Jan Komenda [d], Tomáš Masopust [c], Aiwen Lai [a,*]

[a] *Department of Automation, Xiamen University, Xiamen 361102, China*

[b] *Robotics and Autonomous Systems Thrust, The Hong Kong University of Science and Technology (Guangzhou), China*

[c] *Department of Computer Science, Faculty of Science, Palacky University Olomouc, Czechia*

[d] *Institute of Mathematics of the Czech Academy of Sciences, Prague, Czechia*

Abstract

Diagnosability determines if the occurrence of faults can be inferred from a limited number of observations. The fundamental question asks whether a proposed verification algorithm is the fastest possible. Since no techniques are known how to prove that a faster algorithm does not exist, we consider well-established conjectures from the complexity theory to justify that the algorithms are the best possible in the sense that a significant improvement in the time complexity would refute the conjectures. For strong diagnosability, we show that the quadratic algorithm for verifying diagnosability of monolithic systems is optimal (in the above sense), and that there is no algorithm verifying diagnosability of modular systems faster than the quadratic algorithm applied to the constructed monolithic system. For weak diagnosability, we propose a new exponential algorithm for monolithic systems and show that it is optimal. Finally, we extend weak diagnosability from automata to labeled Petri nets and prove that weak diagnosability is undecidable for unbounded nets.

Key words: Automata, Petri nets, diagnosability, complexity, decidability.

1 Introduction

Diagnosability of discrete-event systems has attracted attention of the research community for several decades (Basilio et al., 2021; Cassandras and Lafortune, 2021). The goal of diagnosability verification is to determine whether every fault can be detected within a finite number of observation steps. While *strong diagnosability* requires every fault to be detected on every path within a finite delay, *weak diagnosability* requires that there is always a possibility to detect the fault.[1] We discuss the complexity of both strong diagnosability (or simply, diagnosability) of Sampath et al. (1995) and weak diagnosability of Cao et al. (2021). Note that weak diagnosability is equivalent to the logical version of A-diagnosability of Masopust and Yin (2019).

For systems modeled by automata, Sampath et al. (1995) proposed an exponential algorithm to verify diagnosability of a monolithic system based on a *diagnoser*, and Jiang et al. (2001) designed a polynomial algorithm based on the parallel composition of two copies of the system, nowadays commonly known as the *twin-plant construction*. This idea eventually led to a quadratic algorithm (Yoo and Lafortune, 2002; Moreira et al., 2011). If the system is modular, the approach to compute a monolithic system results in an algorithm running in time $O(n^{2k})$, where n is the maximum number of states of k modules. Yin and Lafortune (2017b) showed that diagnosability of a modular system cannot be verified in polynomial-time unless P = PSPACE.

While the verification of diagnosability for monolithic systems is solvable in polynomial time, the verification of weak diagnosability is not, unless P = PSPACE (Bertrand et al., 2014; Chen et al., 2016). The algorithm of Cao et al. (2021) verifies weak diagnosability of monolithic systems in time

⋆ This work was supported by Fujian Provincial Natural Science Foundation of China under Grant 2025J01054; by the Natural Science Foundation of Xiamen, China under Grant 3502Z202573035; by the Fundamental Research Funds for the Central Universities under Grant 20720250160; and by RVO 67985840.

* Corresponding author.

Email addresses: smiao585@connect.hkust-gz.edu.cn (Shaowen Miao), petr.jancar@upol.cz (Petr Jančar), komenda@ipm.cz (Jan Komenda), tomas.masopust@upol.cz (Tomáš Masopust), aiwenlai@xmu.edu.cn (Aiwen Lai).

[1] A different notion of weak diagnosability, related to weak fairness, has beed discussed, e.g., by Germanos et al. (2015).

$O(n^2 2^{4n})$, where n is the number of states. For modular systems, Masopust and Yin (2019) showed that there is no polynomial algorithm verifying weak diagnosability. They further showed that, unless P = PSPACE, there is no polynomial algorithm verifying diagnosability of a modular system even in a special case where all shared events are observable.

A challenging question asks whether there are faster algorithms. Is the quadratic algorithm to verify diagnosability for monolithic systems optimal or are there sub-quadratic or even linear algorithms? And how about the exponential algorithms? Indeed, under the assumption that P ≠ PSPACE, we can exclude the existence of polynomial algorithms. However, there could still be significantly faster, sub-exponential algorithms.

In this paper, we address these questions. Since no techniques are known how to prove that an algorithm is the fastest possible, we make use of several well-established conjectures from the complexity theory to justify optimality in the sense that it is very unlikely that there are faster algorithms. We show that:

(1) the quadratic algorithm to verify diagnosability of a monolithic system is optimal;
(2) the algorithm to verify diagnosability of modular systems, running in time $O(n^{2k})$ where n is the maximum number of states among k modules, is optimal in the sense of the existence of an $n^{o(k)}$-time algorithm—it is an open problem whether there is an $O(n^{ck})$-time algorithm for a constant $c < 2$; and
(3) we propose a new, optimal algorithm to verify weak diagnosability for monolithic systems.

Recall that optimality refers to time complexity. In particular, various approaches to diagnosability leverage the modular structure of systems and focus on different aspects; see, e.g., Basilio and Toguyéni (2023); Contant et al. (2006); Debouk et al. (2002); Fabre et al. (2005); García et al. (2005); Li et al. (2017); Pencolé (2004); Ramírez-Treviño et al. (2007); Ricker and Fabre (2000); Schmidt (2013); Ye and Dague (2010); Zhou et al. (2008).

For systems modeled by labeled Petri nets, Cabasino et al. (2012) introduced a verifier net and leveraged its reachability and coverability tree to verify diagnosability of bounded and unbounded nets. The diagnosability verification for certain classes of Petri nets has been discussed, e.g., by Ran et al. (2017); Yin and Lafortune (2017a); Bérard et al. (2018). To the best of our knowledge, weak diagnosability has not been investigated in the context of Petri nets. To bridge the gap, we extend the definition to labeled Petri nets and show that verifying weak diagnosability for labeled Petri nets is undecidable.

2 Preliminaries and definitions for automata

We assume that the reader is familiar with the basic concepts of automata and complexity theory. For all unexplained notions, we refer the reader to Sipser (2012).

Let $\mathbb{N}$ denote the set of non-negative integers. For a set A, $|A|$ denotes the cardinality of A. An alphabet Σ is a finite nonempty set of events. A word over Σ is a finite sequence of events. Let Σ^* denote the set of all words over Σ, including the empty word ε. For a word $w \in \Sigma^*$, $|w|$ denotes the length of w. For two words $w_1, w_2 \in \Sigma^*$, we denote by $w_1 w_2$ the *concatenation* of w_1 and w_2. A language over Σ is a subset $L \subseteq \Sigma^*$. The *post-language* (or left quotient) of L after a word w_1 is the set $L/w_1 = \{w_2 \in \Sigma^* \mid w_1 w_2 \in L\}$.

A *decision problem* is a yes-no question formulated for instances of the problem. A problem is *decidable* if there is an algorithm that for every instance decides whether it is positive or negative. Decidable problems are classified into classes based on the time or space required by an algorithm to solve them. In particular, the classes P and PSPACE consist of problems solvable in polynomial time and polynomial space, respectively. A problem is PSPACE-complete if (i) it belongs to PSPACE and (ii) every problem from PSPACE can be reduced to it in polynomial time. Condition (i) is known as *membership* and (ii) as *hardness*. The question whether P = PSPACE is a long-standing open problem. It is widely believed that P ≠ PSPACE, that is, no polynomial-time algorithms exist for PSPACE-complete problems.

A *nondeterministic finite automaton* (NFA) is a tuple $G = (Q, \Sigma, \delta, I, F)$, where Q is a finite nonempty set of states, $I \subseteq Q$ is a nonempty set of initial states, $F \subseteq Q$ is a set of accepting states, and $\delta\colon Q \times \Sigma \to 2^Q$ is a transition function extendable to $\delta\colon 2^Q \times \Sigma^* \to 2^Q$ by induction. The language *generated* by G is $L(G) = \{w \in \Sigma^* \mid \delta(I, w) \neq \emptyset\}$ and the language *accepted* by G is $L_m(G) = \{w \in \Sigma^* \mid \delta(I, w) \cap F \neq \emptyset\}$. An NFA G is *deterministic* (DFA) if it has a unique initial state ($|I| = 1$) and no nondeterministic transitions ($|\delta(q, a)| \leq 1$ for every $q \in Q$ and $a \in \Sigma$). An NFA is *total* if its transition function δ is total, i.e., $\delta(q, a) \neq \emptyset$ for every $q \in Q$ and $a \in \Sigma$.

The alphabet Σ of an NFA G is partitioned into *observable* events Σ_o and *unobservable* events Σ_{uo}. The *projection* $P\colon \Sigma^* \to \Sigma_o^*$ is a morphism for concatenation defined by $P(a) = a$ for $a \in \Sigma_o$, and $P(a) = \varepsilon$ for $a \in \Sigma_{uo}$, that is, $P(a_1 \cdots a_n) = P(a_1) \cdots P(a_n)$ for every word $a_1 \cdots a_n \in \Sigma^*$. The definition can naturally be extended to languages.

We denote by $\mathrm{Obs}(G)$ the *observer* of G computed by the standard subset construction (Cassandras and Lafortune, 2021). In this paper, however, the transition function of the observe is total, i.e., $\mathrm{Obs}(G)$ includes the empty set as a state. This decision is useful in our algorithmic design. We call the states of $\mathrm{Obs}(G)$ the *configurations* of G.

A modular NFA G is a set $\{G_1, \ldots, G_n\}$ of $n \geq 2$ NFAs, the behavior of which coincides with the parallel composition $\|_{i=1}^n G_i$; see Cassandras and Lafortune (2021) for details.

An NFA is *live* (*deadlock-free*) if there is an outgoing transition from every reachable state, and it is *convergent* if there is no infinite sequence of unobservable events from any reachable state. For NFAs, liveness can be verified in linear time using the classical breadth-first search algorithm on the underlying graph of the automaton, and convergence can be tested by looking for nontrivial strongly connected components in the underlying graph of the automaton restricted to unobservable events. For modular NFAs, deciding both properties is PSPACE-complete; see the appendix.

We use O^* and $\tilde{O}$ to hide polynomial factors $n^{O(1)}$ and subpolynomial factors $n^{o(1)}$, respectively, in the classical big-O notation. Recall that $o(g(n))$ is the class of functions $f(n)$ such that $f(n)/g(n)$ tends to 0 for n going to infinity. For example, $O(n2^{4n})$ can be written as $O^*(2^{4n})$, and $O(n^2 \log n)$ can be written as $\tilde{O}(n^2)$.

3 Conjectures from the complexity theory

A well-established conjecture in the complexity theory is the *orthogonal vectors conjecture* (OVC). The *orthogonal vectors* (OV) problem asks whether, given two sets $\{u_1, \ldots, u_n\}$ and $\{v_1, \ldots, v_n\}$ of binary vectors of dimension d, i.e., vectors in $\{0,1\}^d$, there are vectors u_i and v_j that are orthogonal, i.e., their scalar product is equal to 0. The OVC states that there is no algorithm solving OV in time $O(n^{2-\varepsilon}d^c)$ for any $\varepsilon, c > 0$ (Williams, 2005).

Hypothesis 1 (OVC)**.** *For every $\varepsilon > 0$, there is $c \geq 1$ such that OV cannot be solved in time $\tilde{O}(n^{2-\varepsilon})$ on instances with $d = c \log n$.*

Other well-established hypotheses are the exponential time hypotheses. These hypotheses are based on the fact that so far no one has been able to solve the *Boolean satisfiability problem* (SAT) significantly faster than by trying all possible truth assignments.[2] Namely, the *exponential time hypothesis* (ETH) states that 3-SAT cannot be solved in the subexponential time $2^{o(n)}$, where n is the number of variables.

Hypothesis 2 (ETH)**.** *There is some $\varepsilon > 0$ such that 3-SAT cannot be solved in time $O(2^{\varepsilon n})$, where n is the number of variables in the formula.*

However, ETH admits algorithms running in time $O(c^n)$ for $c < 2$. The fastest 3-SAT algorithm of Paturi et al. (2005) runs in time $O^*(1.30704^n)$. For increasing k, the k-SAT algorithms are getting slower. The best 4-SAT algorithm of Hertli (2014) runs in time $O^*(1.46899^n)$. This observation motivated the formulation of the *strong exponential time hypothesis* (SETH) stating that, for every constant $c < 2$, there is a sufficiently large k such that k-SAT cannot be solved in time $O(c^n)$, see Impagliazzo and Paturi (2001).

Hypothesis 3 (SETH)**.** *For every $\varepsilon > 0$, there is $k \geq 3$ such that k-SAT cannot be solved in time $O(2^{(1-\varepsilon)n})$.*

Finally, we consider the *k-DFA non-emptiness intersection problem* (k-DFA-NEI) that asks, given k DFAs $A_1, \ldots, A_k$, whether the intersection $L_m(A_1) \cap \cdots \cap L_m(A_k) \neq \emptyset$. An obvious algorithm runs in time $O(n^k)$, where n is the maximum number of states of $A_1, \ldots, A_k$. It is conjectured that there is no faster algorithm. In particular, de Oliveira Oliveira and Wehar (2020) discuss the implications of an algorithm for k-DFA-NEI running in time $n^{o(k)}$ on open problems in circuit and computation complexity.

Hypothesis 4 (k-DFA-NEI conjecture)**.** *There is no algorithm solving k-DFA-NEI in time $n^{o(k)}$, where n is the maximum number of states among the k DFAs of the instance.*

4 Diagnosability for automata

As is common in the literature, we consider a single type of fault. A generalization to multiple types of faults is done type by type. We denote by $\Sigma_f \subseteq \Sigma$ the set of fault events (of a given type), call the words from $\Sigma^*\Sigma_f\Sigma^*$ *faulty*, and the other words *non-faulty*. Without loss of generality, we assume that Σ_f is a singleton, i.e., $\Sigma_f = \{\sigma_f\}$, and with a slight abuse of notation, we write $\Sigma_f \in w$ to indicate that a fault event occurs in w.

Definition 1 (Diagnosability for NFAs)**.** A live and convergent NFA G over Σ is *diagnosable* with respect to projection $P\colon \Sigma^* \to \Sigma_o^*$ and fault events $\Sigma_f \subseteq \Sigma_{uo}$ if $(\exists k \in \mathbb{N})(\forall w_1 \in L(G) \cap \Sigma^*\Sigma_f\Sigma^*)(\forall w_2 \in L(G)/w_1)\big[|w_2| \geq k \Rightarrow (\forall w \in L(G))[P(w) = P(w_1w_2) \Rightarrow (\Sigma_f \in w)]\big]$.

The fastest algorithm deciding diagnosability for NFAs runs in time quadratic in the number of states of the automaton (Yoo and Lafortune, 2002; Moreira et al., 2011). We show that this algorithm is optimal in the sense that any subquadratic algorithm would significantly improve the complexity of OV.

Theorem 1. There is no algorithm deciding diagnosability of NFAs in time $\tilde{O}(n^{2-\varepsilon})$, for any $\varepsilon > 0$, unless OVC fails.

Proof. Let $U = \{u_1, \ldots, u_n\}$ and $V = \{v_1, \ldots, v_n\}$ be two sets of binary vectors of dimension d. We can see the vectors as binary words over $\{0,1\}$. Let $\sigma_f \notin \{0,1\}$ be a fault event, and let the events $0, 1$ be observable. For every vector u_i, we construct a $(d+2)$-state automaton A_{u_i} accepting the language $u_i\sigma_f\{0,1\}^*$, and for every v_i, we construct a $(d+1)$-state automaton A_{v_i} accepting the language $v_i\{0,1\}^*$,

[2] A boolean formula consists of variables, logical connectives $\wedge, \vee, \neg$, and parentheses. A literal is a variable or its negation. A clause is a disjunction of literals. A formula is in conjunctive normal form (CNF) if it is a conjunction of clauses. If each clause has at most k literals, the formula is in k-CNF. A formula is satisfiable if there is an assignment of 1 and 0 to the variables evaluating the formula to 1. The (k-)SAT problem is to decide whether a given formula in (k-)CNF is satisfiable.

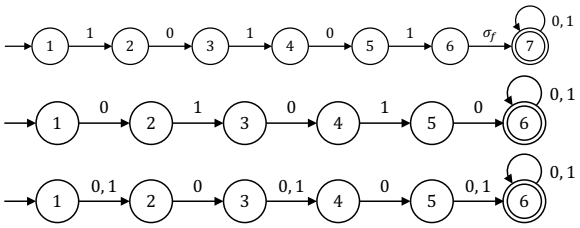

Figure 1. Automata A_{u_i}, A_{v_j}, B_{v_j} for $u_i = 10101$ and $v_j = 01010$.

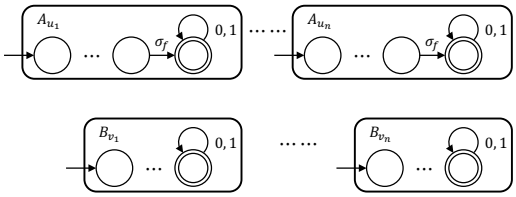

Figure 2. The automaton C.

see Figure 1 for an illustration. From A_{v_i}, we construct the automaton B_{v_i} by replacing events of non-self-loop transitions as follows: event 1 is replaced by 0, and event 0 is replaced by 0,1, see Figure 1. Then B_{v_i} accepts all words of the form $w_i\{0,1\}^*$, where w_i is orthogonal to v_i.

Let C denote the automaton consisting of automata A_{u_i} and B_{v_j} for $i, j = 1, \ldots, n$, where the initial states of these automata form the set of initial states of C, see Figure 2. Then C has $O(nd)$ states and transitions, and we show that C is diagnosable if and only if there is no pair of orthogonal vectors $(u, v) \in U \times V$.

Assume first that there is no pair of orthogonal vectors, and consider a faulty word w. By the construction, the word w is of the form $u_i\sigma_f\{0,1\}^*$, for some $u_i \in U$. Since there is no v_j orthogonal to u_i, there is no word different from w with the same projection as w, and hence C is diagnosable.

Now, let u_i and v_j be orthogonal. For the sake of contradiction, assume that C is diagnosable for a number k. Consider the faulty word $u_i\sigma_f$ and its extension $w = 1^k$. Then the non-faulty word $v_j w$ has the same projection as the faulty word $u_i\sigma_f w$, which contradicts the assumption that C is diagnosable.

Thus, if there were an algorithm solving diagnosability of NFAs in time $O((nd)^{2-\varepsilon})$, for any $\varepsilon > 0$, then we would have an algorithm solving OV in the same time, which contradicts OVC. □

Considering a modular system $\{G_1, \ldots, G_k\}$, where each G_i is an NFA. The simplest algorithm to decide whether the modular system is diagnosable is to construct the monolithic system $G = \|_{i=1}^k G_i$ and apply the quadratic diagnosability algorithm. This algorithm runs in time $O(n^{2k})$, where n is the maximum number of states across the k modules. We show that any substantially faster algorithm, running in time $n^{o(k)}$,

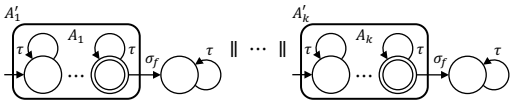

Figure 3. Sketch of the reduction in the proof of Theorem 2.

would imply an $n^{o(k)}$-time algorithm solving the k-DFA-NEI problem, which is considered unlikely. The existence of an $O(n^{ck})$-time algorithm, for a constant $c < 2$, is open.

Theorem 2. There is no algorithm deciding diagnosability of a modular system $\{G_1, \ldots, G_k\}$ running in time $n^{o(k)}$, where n is the maximum number of states among the modules G_i, unless the k-DFA-NEI conjecture fails.

Proof. We make use of the reduction of k-DFA-NEI to the diagnosability of a modular system consisting of k modules of Yin and Lafortune (2017b). Namely, given k DFAs $A_1, \ldots, A_n$, from every A_i, we create an NFA A'_i as a copy of A_i with an additional state f and transitions to f under a fault event σ_f from all accepting states of A_i, and by adding self-loops under a new observable event τ to every state of A'_i, see Figure 3. (Yin and Lafortune, 2017b, Theorem 1) shows that $A_1, \ldots, A_k$ have empty intersection if and only if the modular system $A' = \|_{i=1}^k A'_i$ is diagnosable. Since every A'_i has one state more than A_i, if we could decide diagnosability of A' in time $n^{o(k)}$, where n is the maximum number of states among $A'_1, \ldots, A'_k$, we could solve the k-DFA-NEI problem in time $n^{o(k)}$, which contradicts the k-DFA-NEI conjecture. □

5 Weak diagnosability for automata

We now focus on the verification of weak diagnosability for NFAs. We propose a new verification algorithm with lower complexity, and show that our algorithm is optimal, unless (S)ETH fails.

Definition 2 (Weak Diagnosability for NFAs)**.** A live and convergent NFA G over Σ is *weakly diagnosable* with respect to projection $P\colon \Sigma^* \to \Sigma_o^*$ and fault events $\Sigma_f \subseteq \Sigma_{uo}$ if $(\forall w_1 \in L(G) \cap \Sigma^*\Sigma_f\Sigma^*)(\exists w_2 \in L(G)/w_1)(\forall w \in L(G))[P(w) = P(w_1w_2) \Rightarrow (\Sigma_f \in w)]$.

The definition of weak diagnosability is equivalent to the definition of A-diagnosability of Masopust and Yin (2019) for logical systems. Consequently, the results of Masopust and Yin (2019) apply to weak diagnosablity.

To verify weak diagnosability, Cao et al. (2021) utilize the test automaton of Viana and Basilio (2019), resulting in an algorithm running in time $O(n^2 2^{4n})$, where n is the number of states.

We now propose Algorithm 1 that verifies weak diagnosability in time $O^*(2^n)$. To this end, we make use of the labeling

Algorithm 1 Verification of weak diagnosability

Require: A live and convergent NFA G over Σ, projection P, and fault events $\Sigma_f \subseteq \Sigma$.

Ensure: True if and only if G is weakly diagnosable with respect to P and Σ_f.

1: Compute $G_\ell = G \parallel A_\ell$.
▷ We denote the state (q, ℓ) of G_ℓ by q_ℓ.
2: Let G' denote G with the Σ_f-transitions removed.
3: Compute G_N as the accessible part of G'.
4: Compute $H = G_\ell \parallel \mathrm{Obs}(G_N)$.
5: **if** from every reachable state $(q_F, \cdot)$ of H, a state of the form $(\cdot, \emptyset)$ is reachable **then return** True
6: **else return** False

automaton $A_\ell = (\{N, F\}, \Sigma_f, \delta_\ell, \{N\})$, where $\delta_\ell(N, \sigma_f) = \delta_\ell(F, \sigma_f) = F$. Intuitively, $G_\ell = G \parallel A_\ell$ labels all states of G before (resp. after) the fault occurrence with N (resp. F), and the automaton G_N generates all non-faulty words of G.

Lemma 1. Algorithm 1 is correct.

Proof. Let G be a live and convergent NFA. We show that G is weakly diagnosable with respect to P and Σ_f if and only if from every reachable state of the form $(q_F, \cdot)$ in H, a state of the form $(\cdot, \emptyset)$ is reachable in H.

Let (q_F, X) be a reachable state of H. Then, by the construction of $H = G_\ell \parallel \mathrm{Obs}(G_N)$, there are a faulty word $w_1 \in L(G) \cap \Sigma^*\Sigma_f\Sigma^*$ and a non-faulty word $w_1' \in L(G)$, with the same observations, leading the automaton G from an initial state to state q and the observer $\mathrm{Obs}(G_N)$ from the initial state to state X, respectively.

If there is no state $(p_\ell, \emptyset)$ reachable from state (q_F, X) in H, then for every extension $w_2 \in L(G)/w_1$ of w_1 leading G from state q to state p, there is a non-faulty extension $w_2' \in L(G)/w_1'$ of w_1' with the same observation as w_2 leading the observer $\mathrm{Obs}(G_N)$ from state X to a state $Y \neq \emptyset$. Thus, there is a faulty word w_1 such that for every extension w_2 of w_1, there is a non-faulty word $w_1'w_2'$ with the same observation as w_1w_2 that violates weak diagnosability.

On the other hand, if there is a state $(p_\ell, \emptyset)$ reachable from state (q_F, X) in H, then there is an extension $w_2 \in L(G)/w_1$ of w_1 leading G from state q to state p, and an extension of w_1' leading $\mathrm{Obs}(G_N)$ from state X to state $\emptyset$. However, the state $\emptyset$ of the observer $\mathrm{Obs}(G_N)$ means that there is no non-faulty word generated by G with the same observation as w_1w_2. Therefore, if from every reachable state of the form $(q_F, \cdot)$ in H, a state of the form $(\cdot, \emptyset)$ is reachable in H, then G is weakly diagnosable. □

Lemma 2. The complexity of Algorithm 1 is $O^*(2^n)$, where n is the number of states of the input automaton.

Proof. The complexity of Algorithm 1 is linear in the size of the automaton $H = G_\ell \parallel \mathrm{Obs}(G_N)$, which is $O((n + m)2^n)$, where n is the number of states and m is the number of transitions of the input automaton G. In particular, the verification in the **if** statement on line 5 is done by coloring all states that are backward reachable from states of the form $(\cdot, \emptyset)$. The classical breadth-first search can be used for this coloring in linear time (Cormen et al., 2022). To verify weak diagnosability then means to check whether there is an uncolored state in H of the form $(q_F, \cdot)$. □

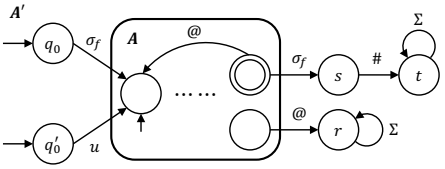

Figure 4. Sketch of the reduction in the proof of Theorem 3.

To highlight the improvement achieved by Algorithm 1 over the approach of Cao et al. (2021), reducing the complexity from $O^*(2^{4n})$ to $O^*(2^n)$, compare, for instance, $2^6 = 64$ with $2^{4 \cdot 6} = 16,777,216$.

We now show that our algorithm is the most efficient algorithm for checking weak diagnosability, unless (S)ETH fails.

Theorem 3. There is no algorithm deciding whether an n-state NFA is weakly diagnosable running in time (i) $2^{o(n)}$, unless ETH fails (this result holds even if the alphabet is binary), (ii) $O^*(2^{n/c})$, for any $c > 2$, unless SETH fails.

Proof. To prove the result, we reduce the universality problem to weak diagnosability. Let $A = (Q, \Sigma, \delta, I, F)$ be an NFA. The idea is to create an NFA A' with two initial states, one that always starts with a fault, σ_f, and the other with a non-faulty unobservable transition, u. The automaton then simulates A. If every configuration of A is accepting, then A' can always generate a particular word ensuring that the fault has to eventually happen and be thus detected. However, if there is a non-accepting configuration of A, then A' can enter a state where no fault ever happens.

Formally, the state set of A' consists of states of A and of five new states, q_0, q_0', r, s, t. The alphabet of A' is $\Sigma \cup \{@, \#, \sigma_f, u\}$, where u is a new unobservable event, σ_f is the unique fault event, and $@, \#$ are new observable events. The events of Σ are observable. The initial states of A' are the two new states q_0 and q_0'.

All transitions of A are transitions of A'. In addition, for every $i \in I$, we add the transitions (q_0, σ_f, i) and (q_0', u, i). For every state $q \in F$ and every state $i \in I$, we add the transition $(q, @, i)$. For every state $q \in Q - F$, we add the transition $(q, @, r)$. From every state $q \in F$, we allow the word $\sigma_f\#$, that is, we define (q, σ_f, s) and $(s, \#, t)$. Finally, we add self-loops under all events of Σ to states r and t, see Figure 4.

If A is universal, then the configuration of A' after reading a faulty word w contains an accepting state of A or state s or t. In addition, w either contains # or not. If it does not, then it has an extension $w' \in \{\#, \sigma_f\#\}$. Since every word that looks the same as a word containing # is faulty, the automaton A' is weakly diagnosable.

If A is not universal, then there is a word w that is not accepted by A. The configuration of A, after generating w, consists of non-accepting states only, and hence the words $\sigma_f w@$ and $uw@$ lead the automaton A' to the configuration containing only state r. In state r, A' can generate any word w' over Σ, which means that no w' can distinguish the faulty word $\sigma_f w@w'$ from the non-faulty word $uw@w'$. Therefore, A' is not weakly diagnosable.

If A has n states, then the automaton A' has $N = n + 5$ states and can be constructed from A in polynomial time. Therefore, if we could decide weak diagnosability of A' in time $2^{o(N)}$, then we could also decide universality of A in time $2^{o(n)}$. However, Fernau and Krebs (2017) have shown that no such algorithm deciding universality exists, unless ETH fails.

The reader may notice that the standard binary encoding of all events of the alphabet proves the result for automata over a binary alphabet.

To prove the second claim of the theorem, we may notice that if we could solve weak diagnosability of the automaton A' in time $O^*(2^{(n+5)/c})$, for any $c > 2$, then we could also solve universality of A in time $O^*(2^{n/c})$. However, this contradicts the result shown in Balun et al. (2023) that no such algorithm exists, unless SETH fails. □

6 Weak diagnosability for labeled Petri nets

A *Petri net* (PN) is a tuple $N = (P, T, \mathcal{F})$, where P and T are finite disjoint sets of *places* and *transitions*, respectively, and $\mathcal{F} \subseteq (P \times T) \cup (T \times P)$ is the *flow relation* specifying the arcs from places to transitions and from transitions to places. It is also viewed as a function $\mathcal{F} \colon (P \times T) \cup (T \times P) \to \{0, 1\}$. Moreover, for $x \in P \cup T$ we put $\mathcal{F}(x) = \{y \mid (x, y) \in \mathcal{F}\}$ and $\mathcal{F}^{-1}(x) = \{y \mid (y, x) \in \mathcal{F}\}$. A *marking* is a map $M \colon P \to \mathbb{N}$ assigning to each place a number of *tokens*. A *transition* t is *enabled* in M, denoted by $M \xrightarrow{t}$, if $M(p) \geq \mathcal{F}(p, t)$ for every place $p \in P$. *Firing*, or *performing*, an enabled transition t in M leads to the marking M', where $M'(p) = M(p) - \mathcal{F}(p, t) + \mathcal{F}(t, p)$ for every $p \in P$; denoted by $M \xrightarrow{t} M'$. For finite transition sequences $\sigma \in T^*$, we write $M \xrightarrow{\sigma}$ to denote that σ is enabled in M, and $M \xrightarrow{\sigma} M'$ to denote that M' is the marking resulting from the firing of σ from M. A marking M is *reachable from* M_0 if $M_0 \xrightarrow{\sigma} M$ for some $\sigma \in T^*$.

A *labeled Petri net* (LPN) is a tuple $G = (P, T, \mathcal{F}, \Sigma, \lambda)$, where $(P, T, \mathcal{F})$ is a Petri net, Σ is a finite alphabet, containing *labels*, and $\lambda \colon T \to \Sigma \cup \{\varepsilon\}$ is a *labeling function*, assigning labels to transitions. The function λ is also viewed as homomorphism $\lambda \colon T^* \to \Sigma^*$ (satisfying $\lambda(\varepsilon) = \varepsilon$ and $\lambda(uv) = \lambda(u)\lambda(v)$). If $\lambda(t) \in \Sigma$, then t is *observable*, and if $\lambda(t) = \varepsilon$, then t is *unobservable*; see, e.g., Peterson (1981) for more details.

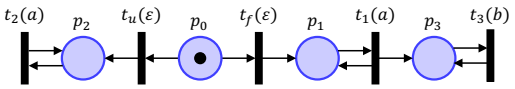

Figure 5. An LPN with alphabet $\{a, b\}$ and the labeling of transitions $\lambda(t_1) = \lambda(t_2) = a$, $\lambda(t_3) = b$, and $\lambda(t_u) = \lambda(t_f) = \varepsilon$.

Given an LPN G, a *marking* M_0 is *live* (i.e., *deadlock-free*) if there is an enabled transition in each marking reachable from M_0, and it is *convergent* if no marking reachable from M_0 enables an infinite sequence of unobservable transitions. The liveness problem is tightly related to the reachability problem for Petri nets, and has a huge complexity, being Ackermann-complete (Czerwinski and Orlikowski, 2021; Leroux, 2021), while the convergence problem is equivalent to promptness, and thus EXPSPACE-complete (Atig and Habermehl, 2011).

Definition 3 (Weak Diagnosability for LPNs)**.** Let $G = (P, T, \mathcal{F}, \Sigma, \lambda)$ be an LPN with a specified set $T_f \subseteq \{t \in T \mid \lambda(t) = \varepsilon\}$ of *faulty transitions*, and with an *initial marking* M_0. A *transition sequence* $\sigma \in T^*$ is *faulty* if some $t \in T_f$ occurs in σ. We say that (G, T_f, M_0) is *weakly diagnosable* if for every faulty sequence σ such that $M_0 \xrightarrow{\sigma}$ there is $\sigma' \in T^*$, where $M_0 \xrightarrow{\sigma\sigma'}$ and each sequence $\rho \in T^*$ satisfying $M_0 \xrightarrow{\rho}$ and $\lambda(\rho) = \lambda(\sigma\sigma')$ is faulty.

Figure 5 depicts a live and convergent LPN $(G, \{t_f\}, M_0)$. It is clear that once b is observed, we know that a fault has occurred. However, the system is not diagnosable since after the occurrence of the fault transition t_f it might be performing just the transition t_1 indefinitely. But it is weakly diagnosable, since any faulty sequence has an extension with the observable content ab, which yields the overall observation that only faulty sequences can match.

We now state the main theorem.

Theorem 4. Given a live and convergent LPN (G, T_f, M_0), it is undecidable if it is weakly diagnosable.

We prove the theorem in the rest of this section. Our proof is partly inspired by a construction from Jančar (1995) that shows the undecidability of action-based equivalences for labeled Petri nets. This construction does not yield our result directly, and thus subtle enhancements needed to be found. We get the undecidability of weak diagnosability by a reduction from a variant of the (undecidable) *halting problem* (HP) for the counter machines introduced by Minsky (1967), i.e., for simple programs that are equivalent to Turing machines but operate on nonnegative integer counters. Specifically, we use a reduction from the problem in Proposition 1

below.

A *Minsky machine* $\mathbf{M}$ *with two counters* c_1 and c_2 is a sequence of numbered instructions

$$0 : ins_0 \,;\, 1 : ins_1 \,;\, 2 : ins_2 \ldots ;\, k : ins_k \qquad (1)$$

where $ins_k = \text{HALT}$, and each ins_i, $i \in \{0, 1, \ldots, k-1\}$, is of one of the following two types:

a) $(c_j := c_j + 1; \text{GOTO } \ell)$
   where $j \in \{1, 2\}$ and $\ell \in \{0, 1, \ldots, k\}$,
b) IF $c_j = 0$ THEN GOTO ℓ_1 ELSE $(c_j := c_j - 1; \text{GOTO } \ell_2)$
   where $j \in \{1, 2\}$ and $\ell_1, \ell_2 \in \{0, 1, \ldots, k\}$.

Given initial counter values $x_1, x_2 \in \mathbb{N}$ of c_1 and c_2, respectively, the computation of $\mathbf{M}$ for x_1 and x_2 executes the program, starting with the initial instruction ins_0; it clearly keeps the counter values nonnegative, and it halts if it reaches the instruction HALT (i.e., ins_k), while it is infinite otherwise. For our aims it is useful to introduce the (nondeterministic) *relaxed machine* $\mathbf{M}_{\text{REL}}$ related to $\mathbf{M}$: we replace the ELSE-part of instruction b) with the *nondeterministic choice* performing either GOTO ℓ_1 or $(c_j := c_j - 1; \text{GOTO } \ell_2)$; $\mathbf{M}_{\text{REL}}$ can thus *cheat* by choosing to perform GOTO ℓ_1 even when $c_j > 0$.

We say that $\mathbf{M}$ is *eligible* if $\mathbf{M}_{\text{REL}}$ has a halting computation for some $x_1, x_2 \in \mathbb{N}$; this property of $\mathbf{M}$ is easily decidable (it amounts to checking if there is a path from 0 to k in the naturally defined graph with the nodes $\{0, 1, \ldots, k\}$ and the edges induced by the instructions ins_i).

Proposition 1. The following problem HP is undecidable:
Instance: an eligible Minsky machine $\mathbf{M}$ with two counters;
Question: are there $x_1, x_2 \in \mathbb{N}$ for which $\mathbf{M}$ halts?

Now we fix an instance $\mathbf{M}$ of HP, of the form (1), and construct an LPN $G = (P, T, \mathcal{F}, \Sigma, \lambda)$, with one faulty transition t_{err}, and a marking m_0. It will be clear that $(G, \{t_{err}\}, m_0)$ is live and convergent, and we will show that $\mathbf{M}$ is a positive instance of HP iff $(G, \{t_{err}\}, m_0)$ is weakly diagnosable.

The construction of G is illustrated using a simple program in Figure 6; we sketch the idea before giving a formal description. We can consider two disjoint copies $\mathbf{M}_1$, $\mathbf{M}_2$ of $\mathbf{M}_{\text{REL}}$ working on the same counters c_1, c_2; in Figure 6, a token being in b_1 or b_2 determines which of $\mathbf{M}_1$ and $\mathbf{M}_2$ is currently simulated. If a computation halts (a token is in q_2 in Figure 6), then a faulty transition t_{err} is enabled if $\mathbf{M}_1$ is active, while a non-faulty transition t_{nerr} is enabled if $\mathbf{M}_2$ is active. After any of t_{err}, t_{nerr}, there is a possibility to restart $\mathbf{M}_1$ with any chosen counter values $x_1, x_2 \in \mathbb{N}$. Moreover, there is an adjustment entailing that each cheating move GOTO ℓ_1 of $\mathbf{M}_1$ has an analogue that switches the control to $\mathbf{M}_2$. This idea is implemented by the LPN G defined as follows. The set of places is

$$P = \{c_1, c_2\} \cup \{q_0, q_1, \ldots, q_k\} \cup \{r\} \cup \{b_1, b_2\},$$

and the initial marking m_0 satisfies $m_0(q_0) = m_0(b_1) = 1$ and $m_0(p) = 0$ for all $p \in P \setminus \{q_0, b_1\}$. The set Σ of observable actions is

$$\{\text{INC}, \text{DEC}, \text{ZER}, \text{RESET}, \text{INC}1, \text{DEC}1, \text{INC}2, \text{DEC}2, \text{RESTART}\}.$$

The transition set T, the flow relation $\mathcal{F} \subseteq (P \times T) \cup (T \times P)$, and the labeling function $\lambda : T \to \Sigma \cup \{\varepsilon\}$ are defined by the points (1)–(6) below, where we use a transparent notation: e.g., the expression $q_i \xrightarrow{\text{INC}} \{q_\ell, c_j\}$ denotes a transition t with $\mathcal{F}^{-1}(t) = \{q_i\}$, $\mathcal{F}(t) = \{q_\ell, c_j\}$, and $\lambda(t) = \text{INC}$.

(1) For $i \in \{0, 1, \ldots, k-1\}$, if ins_i is $(c_j := c_j + 1; \text{GOTO } \ell)$, then it gives rise to the transition $q_i \xrightarrow{\text{INC}} \{q_\ell, c_j\}$;
(2) for $i \in \{0, 1, \ldots, k-1\}$, if ins_i is
IF $c_j = 0$ THEN GOTO ℓ_1 ELSE $\{c_j := c_j - 1; \text{GOTO } \ell_2\}$,
then it gives rise to two ZER-transitions
$q_i \xrightarrow{\text{ZER}} q_{\ell_1}$ and $\{q_i, c_j, b_1\} \xrightarrow{\text{ZER}} \{q_{\ell_1}, c_j, b_2\}$,
and one DEC-transition $\{q_i, c_j\} \xrightarrow{\text{DEC}} q_{\ell_2}$;
(3) the single faulty transition t_{err} is
$\{q_k, b_1\} \xrightarrow{\varepsilon} \{r, b_1\}$,
and there is also a non-faulty t_{nerr} transition
$\{q_k, b_2\} \xrightarrow{\varepsilon} \{r, b_1\}$;
(4) for each $i \in \{0, 1, \ldots, k-1\}$, there is $q_i \xrightarrow{\text{RESET}} r$;
(5) there are counter-setting transitions
$r \xrightarrow{\text{INC}1} \{r, c_1\}$, $\{r, c_1\} \xrightarrow{\text{DEC}1} r$,
$r \xrightarrow{\text{INC}2} \{r, c_2\}$, $\{r, c_2\} \xrightarrow{\text{DEC}2} r$;
(6) finally, there are restarting transitions
$\{r, b_1\} \xrightarrow{\text{RESTART}} \{q_0, b_1\}$, $\{r, b_2\} \xrightarrow{\text{RESTART}} \{q_0, b_1\}$.

The places c_1, c_2 naturally represent the current values of the two counters of $\mathbf{M}$, while $q_0, q_1, \ldots, q_k, r, b_1, b_2$ are "control places"; we observe that each marking $m \in \mathcal{R}(m_0) = \{m \mid m \text{ is reachable from } m_0\}$ satisfies:

$$m(b_1) + m(b_2) = 1, \text{ and}$$
$$m(q_0) + m(q_1) + \cdots + m(q_k) + m(r) = 1.$$

For any $m \in \mathcal{R}(m_0)$ we use the notation (q, b, x_1, x_2) where

$$q \in \{q_0, q_1, \ldots, q_k\} \cup \{r\}, \text{ and } m(q) = 1,$$
$$b \in \{b_1, b_2\}, \text{ and } m(b) = 1,$$
$$m(c_1) = x_1 \text{ and } m(c_2) = x_2.$$

The marking (q_i, b, x_1, x_2) is meant to correspond to the configuration of $\mathbf{M}$, where the instruction $i : ins_i$ is about to be executed. The place r is viewed as a "reset place", and $b \in \{b_1, b_2\}$ serves as a "binary flag". Since Petri nets cannot test places for zero, G can simulate $\mathbf{M}$ unfaithfully; but we can make the following observations, for all $m \in \mathcal{R}(m_0)$:

a) for every (q_i, b, x_1, x_2) where $i \in \{0, 1, \ldots, k-1\}$ we have $(q_i, b, x_1, x_2) \xrightarrow{t} (r, b, x_1, x_2)$ for a RESET-transition t from (4); moreover, $(q_k, b, x_1, x_2) \xrightarrow{t} (r, b_1, x_1, x_2)$

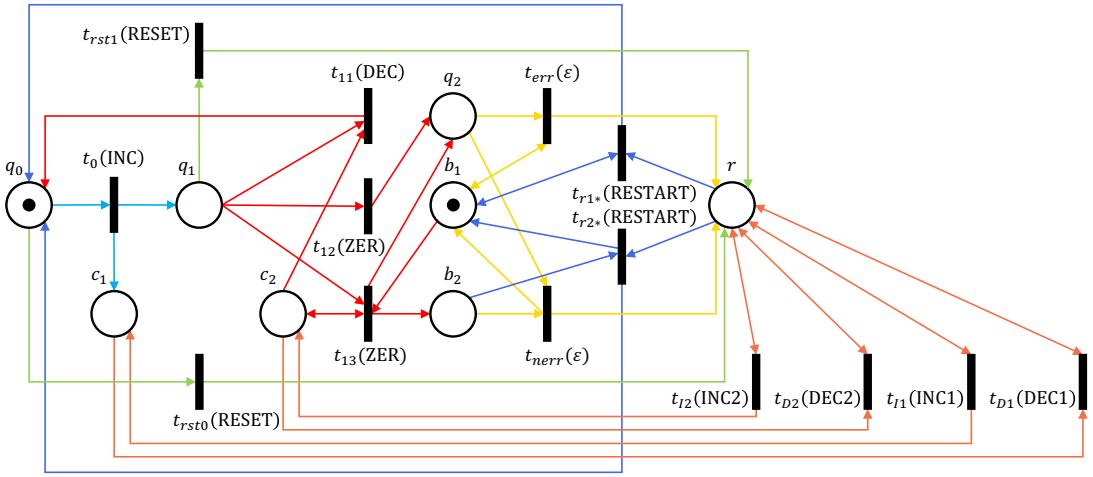

Figure 6. LPN G for the Minsky machine "0 : ($c_1 := c_1 + 1$; GOTO 1); 1 : IF $c_2 = 0$ THEN GOTO 2 ELSE ($c_2 := c_2 - 1$; GOTO 0); 2 : HALT" where the transitions corresponding to (1)–(6) of the construction are in cyan, red, yellow, green, orange, and blue, respectively.

where t is from (3), namely the faulty transition t_{err} if $b = b_1$, and the non-faulty transition t_{nerr} if $b = b_2$;

b) for every $b \in \{b_1, b_2\}$ and $x_1', x_2', x_1, x_2 \in \mathbf{N}$ we have
$$(r, b, x_1', x_2') \xrightarrow{\sigma} (r, b, x_1, x_2) \xrightarrow{t} (q_0, b_1, x_1, x_2)$$
for σ consisting of transitions (5) and t belonging to (6);

c) the only possible case of nondeterminism with respect to labeling is the case of ZER-transitions in (2) at markings (q_i, b_1, x_1, x_2) where $x_j \geq 1$; in this situation, the instruction ins_i of $\mathbf{M}$ prescribes decreasing of c_j, while G can "cheat" and perform one of the two enabled ZER-transitions instead, the second one moving the token from b_1 to b_2;

d) for every two runs $(q_0, b_1, 0, 0) \xrightarrow{\sigma_1} (q, b, x_1, x_2)$ and $(q_0, b_1, 0, 0) \xrightarrow{\sigma_2} (q', b', x_1', x_2')$ with $\lambda(\sigma_1) = \lambda(\sigma_2)$ we have $(x_1, x_2) = (x_1', x_2')$, we might have $b \neq b'$, and if $q \neq q'$, then $\{q, q'\} = \{q_k, r\}$, and thus precisely one of σ_1, σ_2 can be prolonged by an ε-transition (from (3)), by which the first component q_k changes to r as well.

We recall that we aim to show that $\mathbf{M}$ is a positive instance of HP iff $(G, \{t_{err}\}, m_0)$ is weakly diagnosable, where $m_0 = (q_0, b_1, 0, 0)$; we show this by the implications "$\Rightarrow$" and "$\Leftarrow$" below.

"$\Rightarrow$". Let $\mathbf{M}$ be a positive instance of HP, and let us fix some counter values $\bar{x}_1, \bar{x}_2 \in \mathbb{N}$ for which $\mathbf{M}$ halts. Let $(q_0, b_1, 0, 0) \xrightarrow{\sigma} (q, b, x_1, x_2)$ be an arbitrary faulty run of G, that is, $\sigma \in T^* t_{err} T^*$. Using the observations a) and b), we get a run $(q, b, x_1, x_2) \xrightarrow{\pi} (q_0, b_1, \bar{x}_1, \bar{x}_2)$. Now, we consider the run $(q_0, b_1, \bar{x}_1, \bar{x}_2) \xrightarrow{\rho} (q_k, b_1, x_1', x_2')$ that faithfully simulates the computation of $\mathbf{M}$, and is followed by

$$(q_k, b_1, x_1', x_2') \xrightarrow{t_{err}} (r, b_1, x_1', x_2') \xrightarrow{t} (q_0, b_1, x_1', x_2'),$$

where t is the first RESTART-transition from (6). By d) and c), any run $(q_0, b_1, 0, 0) \xrightarrow{\tau} m$, where $\lambda(\tau) = \lambda(\sigma\pi\rho t_{err} t)$, must be of the form

$$(q_0, b_1, 0, 0) \xrightarrow{\tau'} (q_0, b_1, \bar{x}_1, \bar{x}_2) \xrightarrow{\rho\, t_{err}\, t} (q_0, b_1, x_1', x_2')$$

(since there is no cheating in the run $(q_0, b_1, \bar{x}_1, \bar{x}_2) \xrightarrow{\rho}$). Hence $(G, \{t_{err}\}, m_0)$ is weakly diagnosable.

"$\Leftarrow$." Let $\mathbf{M}$ be a negative instance of HP, that is, there are no values $x_1, x_2 \in \mathbb{N}$ for which $\mathbf{M}$ halts. Since $\mathbf{M}$ is eligible, $\mathbf{M}_{\text{REL}}$ has some halting computations (though all of them are cheating), which entails that there is a faulty run $m_0 \xrightarrow{\sigma} m$ of G, that is, $\sigma \in T^* t_{err} T^*$. We now fix an arbitrary faulty run $m_0 \xrightarrow{\sigma} m$ of G, and show that there is a run $m_0 \xrightarrow{\sigma'} m'$ where $\lambda(\sigma') = \lambda(\sigma)$ and the faulty transition t_{err} does not occur in σ'; this will entail that $(G, \{t_{err}\}, m_0)$ is not weakly diagnosable. (Indeed: If we prolong $m_0 \xrightarrow{\sigma} m$ with any continuation, we still get a faulty run, and for this run there is a non-faulty run with the same observable content.)

To this end, let us consider one particular occurrence of the faulty transition t_{err}, i.e.$\{q_k, b_1\} \xrightarrow{\varepsilon} \{r, b_1\}$, in the run $m_0 \xrightarrow{\sigma} m$. It must be the final transition in the respective shortest subrun of the form

$$(q_0, b_1, x_1, x_2) \xrightarrow{\rho} (q_k, b_1, x_1', x_2') \xrightarrow{t_{err}} (r, b_1, x_1', x_2').$$

The subrun $(q_0, b_1, x_1, x_2) \xrightarrow{\rho} (q_k, b_1, x_1', x_2')$ cannot mimic the computation of $\mathbf{M}$ for x_1, x_2 (since $\mathbf{M}$ does not halt for x_1, x_2), and hence at least once a ZER-transition was used instead of the correct DEC-transition. Let us replace one occurrence of such a cheating transition $q_i \xrightarrow{\text{ZER}} q_\ell$ by the transition $\{q_i, c_j, b_1\} \xrightarrow{\text{ZER}} \{q_\ell, c_j, b_2\}$ for the respective $j \in \{1, 2\}$; the occurrence of t_{err} at the end of the subrun is then replaced with the non-faulty transition

$\{q_k, b_2\} \xrightarrow{\varepsilon} \{r, b_1\}$. This replacement of two transition occurrences in σ leads to another enabled sequence that has the same observable content as σ and one occurrence of t_{err} less than σ. Continuing in this way, we create the required $m_0 \xrightarrow{\sigma'} m'$ with no occurrences of t_{err}.

The proof of Theorem 4 is thus finished.

7 Conclusion

We investigated the complexity of verifying strong and weak diagnosability for discrete-event systems modeled by automata and Petri nets, offering new insights into the computational limits of these verification problems. Our results highlight the inherent computational challenges in verifying diagnosability in discrete-event systems, suggesting that any future progress in this area will likely require overcoming fundamental complexity barriers. For an overview of the results, see Table 1. Future work will consider extensions to timed discrete-event systems Miao et al. (2025b,a) and stochastic discree-event systems Bertrand et al. (2014); Lefebvre and Hadjicostis (2022), as well as to fault pattern diagnosis Lefebvre et al. (2023); Ma et al. (2023), which may require seeking additional hypotheses.

A Deciding liveness and convergence

Theorem 5. Deciding liveness and convergence of modular systems modeled by automata is PSPACE-complete.

Proof. To sketch membership, a nondeterministic algorithm guessing a path from an initial state to a state violating the property state by state needs polynomial space to store an n-tuple of states of the modular system consists of n components; hence, liveness (resp. convergence) can be verified in (nondeterministic) polynomial space.

For hardness, we reduce the DFA-intersection problem (Kozen, 1977): Given DFAs $B_1, \ldots, B_n$ over a common alphabet Σ, is $\bigcap_{i=1}^n L_m(B_i) = \emptyset$? Without loss of generality, we assume that B_i are complete, i.e., $L(B_i) = \Sigma^*$. Let $a, x, y_1, \ldots, y_n \notin \Sigma$ be new events. From B_i, we construct A_i by adding a self-loop under a to every state of B_i, and by adding a new state q_i with a self-loop under y_i, to which we have a transition under x from every accepting state of B_i. The only accepting state of A_i is the new state q_i. See Figure A.1 for an illustration. The events of $\Sigma \cup \{a, x\}$ are observable; the status of y_i is discussed below. Then the automaton A_i is live but not convergent when y_i is unobservable; notice that a composition of convergent automata is convergent.

We have $L(A_i) = (\Sigma \cup \{a\})^* \cup L_m(B_i)xy_i^*$, and therefore $L(\|_{i=1}^n A_i) = (\Sigma \cup \{a\})^* \cup \bigcap_{i=1}^n L_m(B_i)xy_i^*$. We show that $\|_{i=1}^n A_i$ is live/convergent if and only if $\bigcap_{i=1}^n L_m(B_i) = \emptyset$.

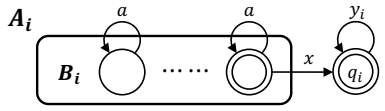

Figure A.1. Modification of each DFA B_i.

If $\bigcap_{i=1}^n L_m(B_i) = \emptyset$, then $L(\|_{i=1}^n A_i) = (\Sigma \cup \{a\})^*$, which is live, because every A_i is complete over $\Sigma \cup \{a\}$, and convergent, because it contains no unobservable events.

If $\bigcap_{i=1}^n L_m(B_i) \neq \emptyset$, then $L(\|_{i=1}^n A_i) = (\Sigma \cup \{a\})^* \cup \bigcap_{i=1}^n L_m(B_i)xy_i^*$, and therefore there is a word w such that $wx \in \bigcap_{i=1}^n L_m(B_i)xy_i^*$. However, after reading wx, the automaton $\|_{i=1}^n A_i$ is in state $(q_1, \ldots, q_n)$, which is blocking if (at least one of) $y_1, \ldots, y_n$ is shared, showing that the composition is not live, or contains a cycle of unobservable events if $y_1, \ldots, y_n$ are local unobservable events, showing that the composition is not convergent. □

References

Atig, M.F., Habermehl, P., 2011. On Yen's path logic for Petri nets. International Journal of Foundations of Computer Science 22, 783–799.

Balun, J., Masopust, T., Osička, P., 2023. Speed me up if you can: Conditional lower bounds on opacity verification, in: International Symposium on Mathematical Foundations of Computer Science, pp. 16:1–16:15.

Basilio, J.C., Hadjicostis, C.N., Su, R., 2021. Analysis and control for resilience of discrete event systems: Fault diagnosis, opacity and cyber security. Foundations and Trends® in Systems and Control 8, 285–443.

Basilio, J.C., Toguyéni, A., 2023. Modular diagnosability of discrete event systems synchronized by observable or unobservable events. IFAC-PapersOnLine 56, 4570–4575.

Bérard, B., Haar, S., Schmitz, S., Schwoon, S., 2018. The complexity of diagnosability and opacity verification for Petri nets. Fundamenta Informaticae 161, 317–349.

Bertrand, N., Haddad, S., Lefaucheux, E., 2014. Foundation of diagnosis and predictability in probabilistic systems, in: IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, pp. 417–429.

Cabasino, M.P., Giua, A., Lafortune, S., Seatzu, C., 2012. A new approach for diagnosability analysis of Petri nets using verifier nets. IEEE Transactions on Automatic Control 57, 3104–3117.

Cao, L., Shu, S., Lin, F., Chen, Q., Liu, C., 2021. Weak diagnosability of discrete-event systems. IEEE Transactions on Control of Network Systems 9, 184–196.

Cassandras, C.G., Lafortune, S., 2021. Introduction to discrete event systems. Springer.

Chen, J., Keroglou, C., Hadjicostis, C.N., Kumar, R., 2016. Revised test for stochastic diagnosability of discrete-event systems. IEEE Transactions on Automation Science and Engineering 15, 404–408.

Table 1
An overview of our and known results; n denotes the maximum number of states among k modules, and ? denotes an open problem; [1] = Yoo and Lafortune (2002), [2] = Bérard et al. (2018), [3] = Yin and Lafortune (2017b), [4] = Bertrand et al. (2014), [5] = Masopust and Yin (2019).

Property	Model	Lower bound		Upper bound		Complexity	
Diag.	NFA	$\tilde{O}(n^2)$	[Theorem 1]	$\tilde{O}(n^2)$	[1]	NL-complete	[2]
Weak Diag.	NFA	$O^*(2^{n/c})$, for any $c > 2$	[Theorem 3]	$O^*(2^n)$	[Theorem 1]	PSPACE-complete	[4]
Diag.	modular	$n^{O(k)}$	[Theorem 2]	$\tilde{O}(n^{2k})$		PSPACE-complete	[3]
Weak Diag.	modular	?		$O^*(2^{n^k})$		EXPSPACE-complete	[5]
Diag.	LPN	?		?		EXPSPACE-complete	[2]
Weak Diag.	LPN	–		–		undecidable	[Theorem 4]

Contant, O., Lafortune, S., Teneketzis, D., 2006. Diagnosability of discrete event systems with modular structure. Discrete Event Dynamic Systems 16, 9–37.

Cormen, T.H., Leiserson, C.E., Rivest, R.L., Stein, C., 2022. Introduction to algorithms. MIT press.

Czerwinski, W., Orlikowski, L., 2021. Reachability in vector addition systems is Ackermann-complete, in: IEEE Symposium on Foundations of Computer Science, pp. 1229–1240.

Debouk, R., Malik, R., Brandin, B.A., 2002. A modular architecture for diagnosis of discrete event systems, in: IEEE Conference on Decision and Control, pp. 417–422.

Fabre, E., Benveniste, A., Haar, S., Jard, C., 2005. Distributed monitoring of concurrent and asynchronous systems. Discrete Event Dynamic Systems 15, 33–84.

Fernau, H., Krebs, A., 2017. Problems on finite automata and the exponential time hypothesis. Algorithms 10, 24.

García, E., Correcher, A., Morant, F., Quiles, E., Blasco, R., 2005. Modular fault diagnosis based on discrete event systems. Discrete Event Dynamic Systems 15, 237–256.

Germanos, V., Haar, S., Khomenko, V., Schwoon, S., 2015. Diagnosability under weak fairness. ACM Transactions on Embedded Computing Systems 14, 1–19.

Hertli, T., 2014. 3-SAT faster and simpler – unique-SAT bounds for PPSZ hold in general. SIAM Journal on Computing 43, 718–729.

Impagliazzo, R., Paturi, R., 2001. On the complexity of k-SAT. Journal of Computer and System Sciences 62, 367–375.

Jančar, P., 1995. Undecidability of bisimilarity for Petri nets and some related problems. Theoretical Computer Science 148, 281–301.

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

Kozen, D., 1977. Lower bounds for natural proof systems, in: IEEE Annual Symposium on Foundations of Computer Science, pp. 254–266.

Lefebvre, D., Hadjicostis, C.N., 2022. Diagnosability of fault patterns with labeled stochastic petri nets. Information Sciences 593, 341–363.

Lefebvre, D., Li, Z., Liang, Y., 2023. Diagnosis of timed patterns for discrete event systems by means of state isolation. Automatica 153, 111045.

Leroux, J., 2021. The reachability problem for Petri nets is not primitive recursive, in: IEEE Symposium on Foundations of Computer Science, pp. 1241–1252.

Li, B., Basilio, J.C., Khlif-Bouassida, M., Toguyéni, A., 2017. Polynomial time verification of modular diagnosability of discrete event systems. IFAC-PapersOnLine 50, 13618–13623.

Ma, Z., Tong, Y., Seatzu, C., 2023. Verification of pattern–pattern diagnosability in partially observed discrete event systems. IEEE Transactions on Automatic Control 69, 2044–2051.

Masopust, T., Yin, X., 2019. Complexity of detectability, opacity and A-diagnosability for modular discrete event systems. Automatica 101, 290–295.

Miao, S., Komenda, J., Lai, A., 2025a. Active diagnosis of time-interval automata: Time perspectives. IEEE Transactions on Automation Science and Engineering 22, 11239–11249.

Miao, S., Lai, A., Komenda, J., Lahaye, S., 2025b. Decentralized fault diagnosis for constant-time automata. IEEE Control Systems Letters 8, 3392–3397.

Minsky, M.L., 1967. Computation: Finite and Infinite Machines. Prentice-Hall, Inc., USA.

Moreira, M.V., Jesus, T.C., Basilio, J.C., 2011. Polynomial time verification of decentralized diagnosability of discrete event systems. IEEE Transactions on Automatic Control 56, 1679–1684.

de Oliveira Oliveira, M., Wehar, M., 2020. On the fine grained complexity of finite automata non-emptiness of intersection, in: International Conference Developments in Language Theory, pp. 69–82.

Paturi, R., Pudlák, P., Saks, M.E., Zane, F., 2005. An improved exponential-time algorithm for k-SAT. Journal of the ACM 52, 337–364.

Pencolé, Y., 2004. Diagnosability analysis of distributed discrete event systems, in: Eureopean Conference on Artificial Intelligence, pp. 43–47.

Peterson, J.L., 1981. Petri net theory and the modeling of systems. Prentice Hall PTR.

Ramírez-Treviño, A., Ruiz-Beltrán, E., Rivera-Rangel, I., López-Mellado, E., 2007. Online fault diagnosis of discrete event systems. A petri net-based approach. IEEE Transactions on Automation Science and Engineering 4, 31–39.

Ran, N., Su, H., Wang, S., 2017. An improved approach

to test diagnosability of bounded Petri nets. IEEE/CAA Journal of Automatica Sinica 4, 297–303.

Ricker, S.L., Fabre, E., 2000. On the construction of modular observers and diagnosers for discrete-event systems, in: IEEE Conference on Decision and Control, pp. 2240–2244.

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

Schmidt, K.W., 2013. Verification of modular diagnosability with local specifications for discrete-event systems. IEEE Transactions on Systems, Man, and Cybernetics: Systems 43, 1130–1140.

Sipser, M., 2012. Introduction to the Theory of Computation. Cengage Learning.

Viana, G.S., Basilio, J.C., 2019. Codiagnosability of discrete event systems revisited: A new necessary and sufficient condition and its applications. Automatica 101, 354–364.

Williams, R., 2005. A new algorithm for optimal 2-constraint satisfaction and its implications. Theoretical Computer Science 348, 357–365.

Ye, L., Dague, P., 2010. Diagnosability analysis of discrete event systems with autonomous components, in: European Conference on Artificial Intelligence, pp. 105–110.

Yin, X., Lafortune, S., 2017a. On the decidability and complexity of diagnosability for labeled Petri nets. IEEE Transactions on Automatic Control 62, 5931–5938.

Yin, X., Lafortune, S., 2017b. Verification complexity of a class of observational properties for modular discrete events systems. Automatica 83, 199–205.

Yoo, T.S., Lafortune, S., 2002. Polynomial-time verification of diagnosability of partially observed discrete-event systems. IEEE Transactions on Automatic Control 47, 1491–1495.

Zhou, C., Kumar, R., Sreenivas, R.S., 2008. Decentralized modular diagnosis of concurrent discrete event systems, in: International Workshop on Discrete Event Systems, p. 388–393.