Efficient diagnosability assessment via ILP optimization: a railway benchmark

Francesco Basile
DIEM
Università di Salerno
Fisciano, Italy
fbasile@unisa.it

Gianmaria De Tommasi , Claudio Sterle
DIETI
Università di Napoli Federico II
Naples, Italy
{detommas,claudio.sterle}@unina.it

Abderraouf Boussif , Mohamed Ghazel
Univ. Lille Nord de France, IFSTTAR
Cosys/Estas
Villeneuve d'Ascq, France
{abderraouf.boussif,mohamed.ghazel}@ifsttar.fr

***Abstract*—Diagnosability of faults in discrete event systems modeled with Petri nets can be assessed either via *graph-based* techniques (also called diagnoser, verifier/twin-plant based techniques), or via the solution of optimization problems. The approaches that belong to the former class are based on the analysis of the net reachability or coverability graphs (or of a more compact version of them). The latter approach exploits the mathematical representation of the net itself to specify and solve optimization problems, which are usually expressed as integer linear programming (ILP) problems.**

In this paper we exploit the railway Petri net model originally proposed in [16], and extended in [14] to be used as a benchmark for diagnosability analysis, to assess the efficiency of the approach based on the solution of ILP problems proposed in [3]. In order to show the effectiveness of the proposed technique, a comparison with a graph-based approach for analyzing diagnosability is also presented.

***Index Terms*—Discrete event systems, Petri nets, diagnosability, ILP problems, benchmarking.**

I. Introduction

In the context of discrete event systems (DES), fault diagnosis is the task related to the detection (and identification) of the occurrences of unobservable faults on the basis of the words of observed events. Since the seminal works [19] and [20] by Sampath *et al.*, many approaches have been proposed in the literature to perform this task, both in the context of finite state automata and Petri net models[1].

However, online fault detection can be performed only if the system is *diagnosable*. Roughly speaking, a DES is said to be diagnosable if all the fault occurrences can be detected when the system evolution is observed for a sufficiently long *time*, i.e. after the occurrence of a sufficiently long word of observable events. It follows that diagnosability is the precondition that is needed in order to perform fault diagnosis. Moreover, diagnosability assessment is a task that can be performed offline, and different approaches have been proposed in literature: from the so-called *diagnoser* approach, proposed in the context of finite state automata (see [19]), to the case of approaches based on model-checking problem, in the context of Petri net (see [21]).

[1]The interested reader can refer to [22] for an overview of the literature in the DES field, and to [6] for a more comprehensive overview of fault detection and fault-tolerant control systems.

When DES are modeled using Petri nets, the approaches that have been proposed to assess diagnosability can be classified in two different categories. On one hand there are those ones that rely on a graph representation of the system state space. These *graph-based* techniques require to analyze the coverability graph or a compact version of it (in the case of bounded net systems), or of the coverability graph (in the case of unbounded systems). Examples of diagnosability approaches that belong to this class can be found in [8], [11], [12], [15], [17]. On the other hand, there are *optimization-based* approaches that exploit the mathematical representation of Petri nets to assess diagnosability by solving optimization problems, which usually are in the form of Integer Linear Programming (ILP) problems. Examples of this technique are [2], [3], [5], [13] in the diagnosability context; a similar approach can be also used for supervisory control, e.g. [4].

Given the complexity of ILP problems, it follows that the diagnosability conditions that belong to this latter class require the solution of NP-hard problems. However, since ILP programming is a standard optimization tool, these techniques can rely on efficient software suites that are available *off-the-shelf*, such as CPLEX® or FICO™ Xpress [1]. It turns out that, despite their computational complexity, the *optimization-based* approaches for diagnosability can be practically more convenient when compared with the *graph-based* ones, which usually requires *ad hoc* algorithms.

In this paper we exploit the railway benchmark model used in [14] to show the effectiveness of the *optimization-based* approach proposed in [3], by comparing it with the *graph-based* one proposed in [7], [8]. This latter approach makes use of the *semi-symbolic diagnoser* (SSD), and turned out to be the most efficient one in comparison to two other reference *graph-based* techniques when applied to the considered railway benchmark (more details can be found in [10]).

For the sake of comparison, it should be noticed that, although computationally efficient, the diagnosability approach proposed in [3] presents several limitations compared to others, such as [8], [17], [21]. In particular, it provides necessary and sufficient condition to check $\mathcal{K}$-diagnosability, i.e. diagnosability after the occurrence of at most $\mathcal{K}$ events after the fault, where the integer $\mathcal{K}$ is fixed. For this reason the technique in [3] cannot be used to assess non-diagnosability and it requires to solve an ILP problem for each value of $\mathcal{K}$ to be tested. Furthermore, this approach works only when the net is bounded.

Despite these limitations, in the case of bounded Petri net model of the plant (which is the case of the railway benchmark in this paper), and when the interest is in *practical* diagnosability, i.e. diagnosability in at most $\mathcal{K}$ steps, the considered *optimization-based* technique turns out to be particularly efficient from the computational point of view. It should be also noticed that, $\mathcal{K}$-diagnosability permits to verify if the fault can be detected within a specified maximum time delay (steps / transition firings). Indeed, if the maximum interleaving between two event occurrences is given, and if it is required to detect the fault within a maximum delay, that implies the fault detection to be performed within a maximum number of firings, which is the parameter $\mathcal{K}$.

This paper is structured as follows. Some backgrounds on labeled Petri nets and diagnosability are given in Section II. Section III briefly introduces the two considered diagnosability approaches for Petri net models: the *optimization-based* one originally proposed in [3], and the *graph-based* one described in [8] and [7]. The railway Petri net model used as a benchmark for the comparison is presented in Section IV, while the results of the numerical experiments aimed at showing the effectiveness of the considered *optimization-based* approach are described in Section V. Eventually, a conclusive discussion is given.

II. Backgrounds

The Petri net basics, together with the definition of diagnosability in the field of Petri nets are introduced in what follows.

A Petri net is a 4-tuple $N = (P, T, \mathbf{Pre}, \mathbf{Post})$, where P is a set of m places (represented by circles), T is a set of n transitions (represented by empty boxes and each one associated to an event), $\mathbf{Pre} : P \times T \mapsto \mathbb{N}$ ($\mathbf{Post} : P \times T \mapsto \mathbb{N}$) is the *pre-* (*post-*) *incidence* matrix. $\mathbf{Pre}(p,t) = w$ ($\mathbf{Post}(p,t) = w$) means that there is an arc with weight w from p to t (from t to p); $\boldsymbol{C} = \mathbf{Post} - \mathbf{Pre}$ is the incidence matrix.

A *marking* is a function $\boldsymbol{m} : P \mapsto \mathbb{N}$ that assigns to each place of a net a nonnegative integer number of tokens, drawn as black dots. It is useful to represent the marking of a net with a vector $\boldsymbol{m} \in \mathbb{N}^m$. A *net system* $S = \langle N, \boldsymbol{m}_0 \rangle$ is a net N with an initial marking $\boldsymbol{m}_0$. A transition t is enabled at $\boldsymbol{m}$ if and only if $\boldsymbol{m} \geq \mathbf{Pre}(\cdot, t)$ and this is denoted as $\boldsymbol{m}[t\rangle$. An enabled transition t may fire, yielding the marking $\boldsymbol{m}' = \boldsymbol{m} + \boldsymbol{C}(\cdot, t)$, and this is denoted as $\boldsymbol{m}[t\rangle \boldsymbol{m}'$.

A *firing sequence* from $\boldsymbol{m}$ is a sequence of transitions $\sigma = t_1 t_2 \ldots t_k$ such that $\boldsymbol{m}\big[t_1\rangle \boldsymbol{m}_1 \big[t_2\rangle \boldsymbol{m}_2 \ldots \big[t_k\rangle \boldsymbol{m}_k$, and this is denoted as $\boldsymbol{m}[\sigma\rangle \boldsymbol{m}_k$. The notations $\boldsymbol{m}\big[\sigma\rangle$ and $\boldsymbol{m}\neg\big[\sigma\rangle$ denote an enabled and a disabled sequence under a marking $\boldsymbol{m}$, respectively. Furthermore, $t_i \in \sigma$ denotes that the transition t_i belongs to the sequence σ. The length of a sequence σ is denoted with $|\sigma|$.

A marking $\boldsymbol{m}'$ is said to be *reachable* from $\boldsymbol{m}_0$ if and only if there exists a sequence σ such that $\boldsymbol{m}_0[\sigma\rangle \boldsymbol{m}'$. $R(N, \boldsymbol{m}_0)$ denotes the set of reachable markings of the net system $\langle N, \boldsymbol{m}_0 \rangle$.

The function $\boldsymbol{\sigma} : T \mapsto \mathbb{N}$, where $\boldsymbol{\sigma}(t)$ represents the number of occurrences of t in σ, is called *firing count vector* of the firing sequence σ. As it has been done for the marking of a net, the firing count vector is often denoted as a vector $\boldsymbol{\sigma} \in \mathbb{N}^n$. The notation $\boldsymbol{\sigma} = \pi(\sigma)$ is used to denote that $\boldsymbol{\sigma}$ is the firing count vector of σ. Given a sequence σ the 1-norm of the related firing count vector[2] $\boldsymbol{\sigma} = \pi(\sigma)$ is equal to the length of the sequence, i.e., $\|\boldsymbol{\sigma}\|_1 = |\sigma|$.

If $\boldsymbol{m}_0[\sigma\rangle \boldsymbol{m}$, then it is possible to write the vector equation

$$\boldsymbol{m} = \boldsymbol{m}_0 + \boldsymbol{C} \cdot \boldsymbol{\sigma}\,, \qquad (1)$$

which is called the *state equation* of the net system.

The set T can be partitioned into the disjoint sets of *observable* and *unobservable* transitions, named respectively T_o and T_{uo} with $\mathrm{card}(T_{uo}) = n_{uo} \leq n$, where $\mathrm{card}(T_{uo})$ denotes the cardinality of T_{uo}. In this paper the fault events $t \in T_f$ are supposed to be unobservable, i.e., $T_f \subseteq T_{uo}$, with $\mathrm{card}(T_f) = n_f \leq n_{uo}$.

We now introduce the notion of *labeled* Petri net (LPN), which allows us to associate events to the net transitions.

Definition 1 (Labeled Petri net system): A *labeled* Petri net (LPN) system is the 3-ple $G = \langle N, \boldsymbol{m}_0, \lambda \rangle$, where N is a standard P/T net, $\boldsymbol{m}_0$ is the initial marking, and $\lambda : T \mapsto E \cup \{\varepsilon\}$ is the *labeling function* which assigns to each transition $t \in T$ either an event from the set E or the *silent event* ε. In particular, it is $\lambda(t) = \varepsilon$ if $t \in T_{uo}$, while $\lambda(t) \neq \varepsilon$ otherwise ◇

In the following it is $\mathrm{card}(E) = e$, and we denote with $T^{\alpha_i} = \big\{t \in T \mid \lambda(t) = \alpha_i\big\}$ the set of transitions associated with the same event $\alpha_i \in E$. Moreover, we denote as w the word of events associated with a sequence σ such that $w = \lambda(\sigma)$, assuming the usual extension of the labeling function[3] $\lambda : T^* \mapsto E^*$. Given a word w we will denote with $|w|$ its length, and with $|w|_{\alpha_i}$ the number of occurrences of the event α_i in w.

Given a firing count vector $\boldsymbol{\sigma} \in \mathbb{N}^n$, in this paper we are often interested to consider only the firings of either the observable or the unobservable transitions. For this reason we introduce the following notations:

$$\boldsymbol{\sigma}_{|T_o} \in \mathbb{N}^n\,,\ \text{with } \boldsymbol{\sigma}_{|T_o}(t) = \begin{cases} \boldsymbol{\sigma}(t) & \text{if } t \in T_o \\ 0 & \text{if} t \notin T_o \end{cases}$$

$$\boldsymbol{\sigma}_{|T_{uo}} \in \mathbb{N}^n\,,\ \text{with } \boldsymbol{\sigma}_{|T_{uo}}(t) = \begin{cases} \boldsymbol{\sigma}(t) & \text{if } t \in T_{uo} \\ 0 & \text{if } t \notin T_{uo} \end{cases}$$

Given a firing count vector $\boldsymbol{\sigma}$ it is $\boldsymbol{\sigma} = \boldsymbol{\sigma}_{|T_o} + \boldsymbol{\sigma}_{|T_{uo}}$.

[2]Given a vector $\boldsymbol{v}$, the 1-norm $\|\boldsymbol{v}\|_1$ is equal to the sum of the absolute values of the vector elements.

[3]The superscript S^* denotes the Kleene closure of the given set S.

Consider a net system $\langle N, \boldsymbol{m}_0\rangle$ with $T = T_{uo} \cup T_o$, and $T_f \subseteq T_{uo}$. Let L be the live and prefix-close language generated by $\langle N, \boldsymbol{m}_0\rangle$. We denote by $L/u = \{v \in T^* \text{ s.t. } uv \in L\}$. the post-language of L after the sequence of transitions u.

Let $Pr : T^* \mapsto T_o^*$ be the usual projection (see [18]), which "erases" the unobservable transitions in a sequence u. The inverse projection operator Pr_L^{-1} is defined as $Pr_L^{-1}(r) = \{\sigma \in L \text{ s.t. } Pr(u) = r\}$.

Let $\dot{u}$ be the final transition of sequence u and define

$$\Psi(\hat{t}) = \left\{u \in L \text{ s.t. } \dot{u} = \hat{t}\right\} .$$

Given a fault t_f and a positive integer $\mathcal{K}$, it is now possible to give the following definition of $\mathcal{K}$-diagnosable fault t_f.

Definition 2 ($\mathcal{K}$-diagnosable fault): Given $t_f \in T_f$ and $\mathcal{K} \in \mathbb{N}$, t_f is said to be $\mathcal{K}$-diagnosable if

$$\forall\ u \in \Psi(t_f) \text{ and } \forall\ v \in L/u \text{ such that } |v| \geq \mathcal{K} ,$$

then it is

$$r \in Pr_L^{-1}\big(Pr(uv)\big) \Rightarrow t_f \in r .$$

$\diamond$

If u is any sequence generated by the system that ends in a failure event t_f, and v is a continuation of u which holds at least $\mathcal{K}$ transitions, then $\mathcal{K}$-diagnosability implies that it is possible to detect the occurrence of t_f within a finite delay, specifically after the firing of at most $\mathcal{K}$ transitions after its occurrence.

It should be noticed that diagnosability requires the existence of an upper bound for the continuation of u (see also [19]), while $\mathcal{K}$-diagnosability specifies a *quantitative* bound for the number of events in the continuation of u, i.e., it specifies an upper bound for the number of events that are needed to detect a fault. In this sense we claim that $\mathcal{K}$-diagnosability has more sense from a *practical* point of view. Moreover, when labeled nets are considered, two or more transitions can share the same event α. This source of nondeterminism affects the diagnosability.

III. Diagnosability of Petri net models

In this section the two diagnosability approaches that are compared in this paper are briefly introduced, and some details about their computational complexity are given. For more details, the interested reader should refer to [3], [8] and [7].

A. $\mathcal{K}$-diagnosability analysis via resolving ILP problems

The following theorem represents the main result presented in [3] that gives a necessary and sufficient condition to check $\mathcal{K}$-diagnosability in bounded and live labeled net systems.

Theorem 1 ([3]): Consider a labeled bounded and live net system $G = \langle N, \boldsymbol{m}_0, \lambda\rangle$ and a fault transition t_f, and let $\mathcal{J}$ be a positive integer such that the constraints (3), in the following denoted with $\mathcal{F}(\boldsymbol{m}_0, \hat{t}, \mathcal{J}, \mathcal{K})$, fully describe the set

$$\mathcal{M}(t_f) = \left\{\boldsymbol{m} \in \mathbb{N}^m \mid \Big(\boldsymbol{m}_0[u\rangle\boldsymbol{m}\Big) \bigwedge \Big(t_f \notin u\Big) \bigwedge \Big(\boldsymbol{m}[t_f\rangle\Big)\right\} . \qquad (2)$$

where $\bigwedge$ denotes the logical *AND* operator. Given a positive integer $\mathcal{K}$, t_f is $\mathcal{K}$-diagnosable *if and only if* there exist $3\big(\mathcal{J} + \mathcal{K}\big)$ vectors $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_{\mathcal{J}}, \boldsymbol{v}_1, \ldots, \boldsymbol{v}_{\mathcal{K}}, \boldsymbol{\epsilon}_1, \ldots, \boldsymbol{\epsilon}_{\mathcal{J}+\mathcal{K}}, \boldsymbol{s}_1, \ldots, \boldsymbol{s}_{\mathcal{J}+\mathcal{K}} \in \mathbb{N}^n$ such that:

$$\min_{\text{s.t. } \mathcal{LD}(\boldsymbol{m}_0, t_f, \mathcal{J}, \mathcal{K})} \sum_{r=1}^{\mathcal{J}+\mathcal{K}} \boldsymbol{\epsilon}_r(t_f) \neq 0 ,$$

where the two sets of constraints $\mathcal{LD}(\boldsymbol{m}_0, t_f, \mathcal{J}, \mathcal{K})$ and $\mathcal{LE}(\boldsymbol{m}_0, |w|_{\alpha_1}, \ldots, |w|_{\alpha_e})$ are specified in (4) and (5), respectively.

$$\begin{cases}
\boldsymbol{m}_0 \geq \mathbf{Pre} \cdot \boldsymbol{u}_1 \\
\boldsymbol{m}_0 + \boldsymbol{C} \cdot \boldsymbol{u}_1 \geq \mathbf{Pre} \cdot \boldsymbol{u}_2 \\
\ldots \qquad \text{(3a)} \\
\boldsymbol{m}_0 + \boldsymbol{C} \cdot \sum_{i=1}^{\mathcal{J}-1} \boldsymbol{u}_i \geq \mathbf{Pre} \cdot \boldsymbol{u}_{\mathcal{J}} \\
\boldsymbol{m}_0 + \boldsymbol{C} \cdot \sum_{i=1}^{\mathcal{J}} \boldsymbol{u}_i \geq \mathbf{Pre}(\cdot, \hat{t}) \qquad \text{(3b)} \\
\boldsymbol{m}_0 + \boldsymbol{C} \cdot \sum_{i=1}^{\mathcal{J}} \boldsymbol{u}_i + \boldsymbol{C}(\cdot, \hat{t}) \geq \mathbf{Pre} \cdot \boldsymbol{v}_1 \\
\boldsymbol{m}_0 + \boldsymbol{C} \cdot \sum_{i=1}^{\mathcal{J}} \boldsymbol{u}_i + \boldsymbol{C}(\cdot, \hat{t}) + \boldsymbol{C} \cdot \boldsymbol{v}_1 \geq \mathbf{Pre} \cdot \boldsymbol{v}_2 \\
\ldots \qquad \text{(3c)} \\
\boldsymbol{m}_0 + \boldsymbol{C} \cdot \sum_{i=1}^{\mathcal{J}} \boldsymbol{u}_i + \boldsymbol{C}(\cdot, \hat{t}) + \boldsymbol{C} \cdot \sum_{j=1}^{\mathcal{K}-1} \boldsymbol{v}_j \geq \mathbf{Pre} \cdot \boldsymbol{v}_{\mathcal{K}} \\
\sum_{i=1}^{\mathcal{J}} \boldsymbol{u}(\hat{t}) = 0 \qquad \text{(3d)} \\
\left\| \sum_{j=1}^{\mathcal{K}} \boldsymbol{v}_j \right\|_1 \geq \mathcal{K} \qquad \text{(3e)}
\end{cases}$$

It is worth to notice that, since we are dealing with bounded net systems, there exists an integer $\mathcal{J}_{\min}$ such that all the markings in the set (2) can be characterized by the set of constraints (3). Although in the worst case $\mathcal{J}_{\min}$ may be equal to $\operatorname{card}(R(N, \boldsymbol{m}_0)) - 1$, in many cases $\mathcal{J}_{\min} \ll \operatorname{card}(R(N, \boldsymbol{m}_0)) - 1$. For a more comprehensive discussion on this issue, the interested reader can refer to [3, Sec. 3], where some results to estimate $\mathcal{J}_{\min}$ for bounded and live systems are given.

$$\begin{cases}
\boldsymbol{m}_0 + \boldsymbol{C} \cdot \boldsymbol{\epsilon}_{1|T_{uo}} \geq \mathbf{Pre} \cdot \boldsymbol{s}_{1|T_o} \\
\boldsymbol{m}_0 + \boldsymbol{C} \cdot \sum_{i=1}^{2} \boldsymbol{\epsilon}_{i|T_{uo}} + \boldsymbol{C} \cdot \boldsymbol{s}_{1|T_o} \geq \mathbf{Pre} \cdot \boldsymbol{s}_{2|T_o} \\
\ldots \qquad \text{(5a)} \\
\boldsymbol{m}_0 + \boldsymbol{C} \cdot \sum_{i=1}^{\rho} \boldsymbol{\epsilon}_{i|T_{uo}} + \boldsymbol{C} \cdot \sum_{i=1}^{\rho-1} \boldsymbol{s}_{j|T_o} \geq \mathbf{Pre} \cdot \boldsymbol{s}_{\rho|T_o} \\
\boldsymbol{m}_0 \geq \mathbf{Pre} \cdot \boldsymbol{\epsilon}_{1|T_{uo}} \\
\boldsymbol{m}_0 + \boldsymbol{C} \cdot \big(\boldsymbol{\epsilon}_{1|T_{uo}} + \boldsymbol{s}_{1|T_o}\big) \geq \mathbf{Pre} \cdot \boldsymbol{\epsilon}_{2|T_{uo}} \\
\ldots \qquad \text{(5b)} \\
\boldsymbol{m}_0 + \boldsymbol{C} \cdot \sum_{i=1}^{\rho-1} \Big(\boldsymbol{\epsilon}_{i|T_{uo}} + \boldsymbol{s}_{i|T_o}\Big) \geq \mathbf{Pre} \cdot \boldsymbol{\epsilon}_{\rho|T_{uo}} \\
\sum_{t_j \in T^{\alpha_l}} \sum_{i=1}^{\rho} \boldsymbol{s}_i(t_j) = |w|_{\alpha_l} , \quad l = 1, \ldots, e \qquad \text{(5c)}
\end{cases}$$

$$\begin{cases} \mathcal{F}\big(\boldsymbol{m}_0\,, t_f\,, \mathcal{J}\,, \mathcal{K}\big) & \text{(4a)} \\ \mathcal{LE}\Big(\boldsymbol{m}_0\,, \sum_{t_l\in T^{\alpha_1}} \Big(\sum_{i=1}^{\mathcal{J}} \boldsymbol{u}_i(t_l) + \sum_{j=1}^{\mathcal{K}} \boldsymbol{v}_j(t_l)\Big), \ldots, \sum_{t_l\in T^{\alpha_e}} \Big(\sum_{i=1}^{\mathcal{J}} \boldsymbol{u}_i(t_l) + \sum_{j=1}^{\mathcal{K}} \boldsymbol{v}_j(t_l)\Big)\Big) & \text{(4b)} \\ \sum_{t_j\in T^{\alpha_l}} \boldsymbol{s}_1(t_j) = \sum_{t_j\in T^{\alpha_l}} \boldsymbol{u}_1(t_j)\,, \quad l = 1, \ldots, e & \\ \ldots & \\ \sum_{t_j\in T^{\alpha_l}} \boldsymbol{s}_{\mathcal{J}}(t_j) = \sum_{t_j\in T^{\alpha_l}} \boldsymbol{u}_{\mathcal{J}}(t_j)\,, \quad l = 1, \ldots, e & \\ \sum_{t_j\in T^{\alpha_l}} \boldsymbol{s}_{\mathcal{J}+1}(t_j) = \sum_{t_j\in T^{\alpha_l}} \boldsymbol{v}_1(t_j)\,, \quad l = 1, \ldots, e & \text{(4c)} \\ \ldots & \\ \sum_{t_j\in T^{\alpha_l}} \boldsymbol{s}_{\mathcal{J}+\mathcal{K}}(t_j) = \sum_{t_j\in T^{\alpha_l}} \boldsymbol{v}_{\mathcal{K}}(t_j)\,, \quad l = 1, \ldots, e & \end{cases}$$

We now briefly discuss the complexity of the proposed *optimization based* approach to check $\mathcal{K}$-diagnosability. Although it is well known that the ILP optimization problems are NP-hard, their complexity depends on the number of unknowns and constraints that are involved.

Assuming that, for a given net the integer $\mathcal{J}$ is given and does not depend on the net size and on the initial marking (as it will be in many practical cases, including the benchmark considered in this paper), it readily follows that the number of unknowns in $\mathcal{F}\left(\boldsymbol{m}_0\,, t_f\,, \mathcal{J}\,, \mathcal{K}\right)$ is equal to:

$$\#\text{unknowns}_{\mathcal{F}} = (\mathcal{J} + \mathcal{K}) \cdot n\,,$$

while the correspondent number of constraints is:

$$\#\text{constraints}_{\mathcal{F}} = (\mathcal{J} + \mathcal{K} + 2) \cdot m + 1\,.$$

When dealing with the set of constraints $\mathcal{E}(\cdot\,,\cdot\,,\cdot)$ it should be noticed that the unknowns in each $\boldsymbol{\epsilon}_i$ vector are the n_{uo} components related to the unobservable transitions, while in each $\boldsymbol{s}_i$ vector the unknowns are the $n - n_{uo}$ observable components. Moreover, thanks to the constraints (4c), the latest unknowns are fictitious, thus

$$\#\text{unknowns}_{\mathcal{E}} = (\mathcal{J} + \mathcal{K}) \cdot n_{uo}\,,$$

and

$$\#\text{constraints}_{\mathcal{E}} = [2 \cdot (\mathcal{J} + \mathcal{K}) + 1] \cdot m\,.$$

It turns out that the overall number of unknowns is:

$$\#\text{unknowns} = (\mathcal{J} + \mathcal{K}) \cdot (n + n_{uo}) < 2n\mathcal{J} + 2n\mathcal{K}\,,$$

while the total number of constraints is:

$$\#\text{constraints} = 3m\mathcal{J} + 3m\mathcal{K} + 3m + 1\,,$$

Hence, if $\mathcal{J}$ is given, the numbers of unknowns and of constraints grow linearly with respect to $\mathcal{K}$. Furthermore, if $\mathcal{J}$ is given, the numbers of constraints and unknowns increase linearly with the net size, and are independent from the initial marking. However, in the more general case where the integer $\mathcal{J}$ cannot be considered constant for a given net system, its value may be changed as a function of the initial marking. In [3] it was proved that it is:

$$\mathcal{J}_{\min} \leq 2 \cdot \|\boldsymbol{m}_0\|_1 \cdot \left\| \sum_{\boldsymbol{y}\in\mathcal{T}(N)} \boldsymbol{y} \right\|_1\,,$$

where $\mathcal{T}(N)$ is the set of minimal support T-invariants of N. Hence, in the worst case, the complexity grows exponentially with respect to the net size.

B. Diagnosability via graph-based approach

In order to formally present the considered *graph-based* approach, we introduce the following notations:

- Given a subset of transitions $T' \subseteq T$, we define $Enable_{T'}(\boldsymbol{m}) = \{t \in T' \mid \boldsymbol{m}\,[\,t\rangle\}$, as the set of transitions in T' that are enabled at marking $\boldsymbol{m}$. The extension to a subset of markings $M' \subseteq R(N, \boldsymbol{m}_0)$, is $Enable_{T'}(M') = \bigcup_{\boldsymbol{m}\in M'} Enable_{T'}(\boldsymbol{m})$.
- Given a subset of markings $M \subseteq R(N, \boldsymbol{m}_0)$ and a transition $t \in T$, we define $Img(M, t) = \{\boldsymbol{m}' \in R(N, \boldsymbol{m}_0) \,|\, \exists \boldsymbol{m} \in M : \mathbf{m}[\mathbf{t}\rangle \boldsymbol{m}'\}$ as the set of markings that are reachable from the markings in M by firing transition t. The generalization to a subset of transitions $T' \subseteq T$ is $Img(M, T') = \bigcup_{t\in T'} Img(M, t)$.
- Given a marking $\boldsymbol{m} \in R(N, \boldsymbol{m}_0)$ and a subset of transitions $T' \subseteq T$, we define $Reach_{T'}(\boldsymbol{m}) = \{\boldsymbol{m}\} \cup \{\boldsymbol{m}' \in R(N, \boldsymbol{m}_0) | (\exists\ \sigma \in T'^*) : \boldsymbol{m}\,[\,\sigma\,\rangle\,\boldsymbol{m}'\}$ as the set of markings that can be reached by firing a sequence of transitions in T' from marking $\boldsymbol{m}$. The generalization to a subset of markings $M \subseteq R(N, \boldsymbol{m}_0)$ is $Reach_{T'}(M) = \bigcup_{\boldsymbol{m}\in M} Reach_{T'}(\boldsymbol{m})$.

Generally, all *graph-based* approaches use a deterministic graph (called *diagnoser*) whose nodes contain a set of reachable (normal and/or faulty) markings and whose arcs are the observed labels (events). In fact, the diagnoser is used in order to verify the diagnosability property and perform the online diagnosis (in the case of diagnosable systems).

The *graph-based* technique considered in this paper for the comparison study is the SSD approach, introduced in [7],

[8], which is based on the computation of a *semi-symbolic diagnoser* (from which the SSD acronym is derived). The SSD technique shows three interesting features compared to the classic approaches, namely: i) a new structure which explicitly separates between the normal and the faulty markings in each node, (ii) a compact representation of the node markings using *binary decision diagrams* (BDD), and (iii) an on-the-fly algorithm to synthesize the diagnoser and analyze diagnosability simultaneously.

In fact, in the SSD approach each node a of the diagnoser is partitioned into two distinct subsets of markings, each of them is encoded using a BDD:

1) *the set of normal markings* (denoted by $a.\mathcal{M}_N$), which is the subset of markings in node a that are reachable upon the of firing faulty-free sequences.
2) *the set of faulty markings* (denoted by $a.\mathcal{M}_F$), which is the subset of markings in node a that are reachable upon the firing faulty sequences.

It should be noticed that, in a given node a, there may exist some faulty transitions that link some markings in $a.\mathcal{M}_N$ to some others in $a.\mathcal{M}_F$. The existence of such transitions is also encoded within each node using a boolean variable denoted $\mathcal{B}_a$ (*true* if such transitions exist and *false* if not). As illustrated in [7], such a node structure can be advantageously explored for rendering diagnosability analysis more efficiently than using the classic diagnoser approaches.

Definition 3: The SSD associated with a LPN $G = \langle N, \boldsymbol{m}_0, \lambda\rangle$ is a deterministic graph $\mathcal{D} = \langle \mathcal{A}, E, \delta_{\mathcal{D}}, a_0\rangle$, where:

1) $\mathcal{A}$ is a finite set of diagnoser nodes;
2) E is a finite set of events (labels);
3) a_0 is the initial diagnoser node with:
   a) $a_0.\mathcal{M}_N = Reach_{T_{reg}}(\boldsymbol{m}_0)$;
   b) $a_0.\mathcal{M}_F = Reach_{T_{uo}}(Img(a_0.\mathcal{M}_N, T_f))$.
   c)
$$\mathcal{B}_{a_0}\begin{cases} true & \text{if } Img(a_0.\mathcal{M}_N, T_f) \neq \emptyset \\ false & \text{if } Img(a_0.\mathcal{M}_N, T_f) = \emptyset \end{cases}$$
4) $\delta_{\mathcal{D}} : \mathcal{A} \times E \to \mathcal{A}$ is the transition relation, defined as follows:
   $\forall\ a, a' \in \mathcal{A},\ \alpha \in \Sigma_o$: $a' = \delta_{\mathcal{D}}(a, \alpha) \Leftrightarrow a'.\mathcal{M}_N = Reach_{T_{reg}}(Img(a.\mathcal{M}_N, T_\sigma)) \wedge a'.\mathcal{M}_F = Reach_{T_{uo}}(Img(a'.\mathcal{M}_N, T_f) \cup Img(a.\mathcal{M}_F, T^\alpha))$
$$\mathcal{B}_{a'}\begin{cases} true & \text{if } Img(a'.\mathcal{M}_N, \Sigma_f) \neq \emptyset \\ false & \text{if } Img(a'.\mathcal{M}_N, \Sigma_f) = \emptyset \end{cases}$$
   with $T_{reg} = T_{uo} \backslash T_f$.
5) In each node a_i, sets $a_i.\mathcal{M}_N$ and $a_i.\mathcal{M}_F$ are encoded as BDDs, this aspect is not discussed in the present paper; the reader should refer to [8] for more details. $\diamond$

According to this definition, one can differentiate between three types of diagnoser nodes:

- *N-certain node* a: is a diagnoser node where the set of faulty markings is empty ($a.\mathcal{M}_F = \emptyset$);
- *F-certain node* a: is a diagnoser node where the set of normal marking is empty ($a.\mathcal{M}_N = \emptyset$);
- *F-uncertain node* a: is a diagnoser node of which neither the normal set nor the faulty set of markings is empty, i.e., $a.\mathcal{M}_N \neq \emptyset$ and $a.\mathcal{M}_F \neq \emptyset$.

In the same way as in [19], we define F-uncertain cycle as follows:

Definition 4: (F-uncertain cycle)
A cycle $c\ell = a_1, a_2, \ldots, a_n$, with $\delta_{\mathcal{D}}(a_i, \alpha_i) = a_{(i+1)mod_n}$[4] for $1 \leq i \leq n$, in diagnoser $\mathcal{D}$, is said to be an F-uncertain cycle, if $\forall i :\ 1 \leq i \leq n : a_i$ is an F-uncertain node.

Diagnosability verification using the SSD is based on a simplified necessary and sufficient established using the notion of "indicating sequence", which is associated with the F-uncertain cycles[5].

Definition 5: ($c\ell$-indicating sequence)
Let $c\ell = a_1, a_2, \ldots, a_n$ be an F-*uncertain* cycle in $\mathcal{D}$ (the starting node a_1 can be arbitrarily chosen in the cycle), with $\delta_{\mathcal{D}}(a_i, \alpha_i) = a_{(i+1)mod_n}$ for $1 \leq i \leq n$. The associated $c\ell$-indicating sequence $\rho^{c\ell} = \mathcal{S}_1, \mathcal{S}_2, \ldots$, is an infinite sequence of sets of markings, such that:
– $\mathcal{S}_1 = a_1.\mathcal{M}_F$;
– $\forall\, i > 1 :\ \mathcal{S}_i = Reach_{T_{uo}}(Img(\mathcal{S}_{i-1}, T_{\alpha_{(i-1)_{mod_n}}}))$. $\diamond$

In other terms, the $c\ell$-indicating sequence tracks the subsets of faulty markings in each node of $c\ell$ without considering the faulty markings generated through the occurrence of some faulty transitions outgoing from the normal set of markings in the traversed nodes (except for $\mathcal{S}_1$ which holds all the faulty states of node a_1, i.e., $\mathcal{S}_1 = a_1.\mathcal{M}_F$).

In fact, the $c\ell$-indicating sequence is introduced with the aim of elucidating the actual faulty cycles corresponding to a given F-*uncertain* cycle, if such cycles exist in the original model.

Based on this notion, a simplified necessary and sufficient condition for diagnosability of LPNs can be reformulated as follows:

Theorem 2: ([7]) *An LPN is said to be diagnosable, w.r.t. P and T_f, if and only if for each $F-$uncertain cycle $c\ell$ in its diagnoser $\mathcal{D}$ and $\rho^{c\ell} = \mathcal{S}_1, \mathcal{S}_2, \ldots$ is its indicating sequence; then $\exists\ i \in \mathbb{N}^*\ :\ \mathcal{S}_i = \emptyset$.* $\diamond$

It is worth recalling that series $\rho^{c\ell}$ necessarily reaches a repetitive cycle and that i in the above theorem is most equal to the number of markings in the reachability graph of the net. For the actual verification of diagnosability, and based on this result, *a systematic procedure* is derived directly from Theorem 2. This procedure is repeated on each F-uncertain cycle generated on-the-fly in $\mathcal{D}$, and is performed as follows.

When an F-uncertain cycle $c\ell$ is built in SSD $\mathcal{D}$, then:

1) generate the successive elements of $c\ell$-indicating sequence $\rho^{c\ell}$ (starting from $\mathcal{S}_1$), and for each element $\mathcal{S}_i$ check the following conditions:

[4]The notation $\delta_{\mathcal{D}}(a_i, \alpha_i) = a_{(i+1)mod_n}$ is used to denote that $\forall i < n$ it is $\delta_{\mathcal{D}}(a_i, \alpha_i) = a_{i+1}$ and $\delta_{\mathcal{D}}(a_n, \alpha_n) = a_1$.

[5]A similar notion called "refined sequence" has been introduced independently by A. Giua in the lecture available at http://www.diee.unica.it/giua/ARP/.

a) if $\mathcal{S}_i = \emptyset$, then cycle $c\ell$ satisfies the condition of Theorem 2 and therefore the procedure is stopped;

b) else, if $\mathcal{S}_i \neq \emptyset$ and $\exists k \in \mathbb{N} : i = 1 + kn$ (with $n = |c\ell|$), then:

   i) if $\mathcal{S}_i = \mathcal{S}_{(i-n)}$, then cycle $c\ell$ violates the condition of Theorem 2 and stop the procedure;

   ii) else continue.

The SSD-based algorithm for analyzing diagnosability has been implemented in a tool called DPN-SOG [9], which is a command-line software tool developed in C++ programming language. DPN-SOG builds the SSD on the fly and simultaneously analyzes the diagnosability. When the LPN system is non-diagnosable, DPN-SOG outputs the generated part of the diagnoser as well as a witnessed event-trace that violates the diagnosability property (the first encountered event-trace that violates the diagnosability feature, based on Theorem 2). If the LPN model is found to be diagnosable, DPN-SOG generates the part of the diagnoser that is sufficient to perform the online diagnosis, which is does not necessarily requires to build the whole reachability graph, as explained in [7]. Further pieces of information that help assessing the efficiency of the approach are output by the tool, namely, the required CPU time, the size of the diagnoser, the number of used BDD nodes and memory required to store the SSD (in kilobytes).

IV. The Level Crossing Benchmark Model

In this section we briefly introduce the benchmark model for diagnosability analysis that has been proposed in [14], and which has been derived from the railway model originally proposed in [16].

The considered benchmark is a railway level crossing system (LC) which considers one or more railway tracks. The LC system is composed of three main subsystems: i) sensing module to detect trains' position relative to the LC along each track; ii) barriers module to stop the road traffic; and iii) a local control module to activate/deactivate the barriers, flashing lights and sound alarm. The operational logic of a multi-track LC considers the railway traffic on each track:

- the LC is closed to road traffic if for at least one track, a train is in the crossing zone;
- the LC is reopened to road traffic only if, for all the tracks no train is in the crossing zone.

The LC benchmark was proposed in [14] in order to analyze various diagnosis issues. An interesting feature of this benchmark is that it can be extended to n railway tracks. Hence, while the size of the model grows linearly, its state-space grows exponentially, which is suitable for evaluating the efficiency of an approach.

For diagnosis purposes, all the benchmark models include two classes of faults, and the fault transitions are represented in red in the LPN model shown in Fig. 1. The first one, named T_{f_1}, related to a train-sensing defect is modeled by unobservable transition $(t_{i,4}, ig)$ and indicates that the train may enter the LC zone before the barriers are lowered. The second failure, named T_{f_2}, modeled by unobservable transition (t_6, bf) indicates a defect of the barriers that results in a premature rising. Either of these two faults can induce incorrect operation of the LC control and possibly train-car collisions.

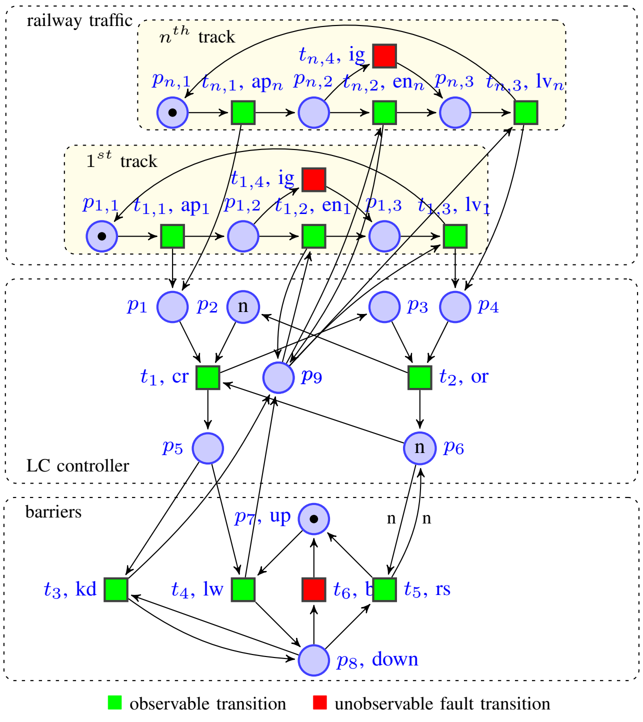

Fig. 1. The multi-track level crossing benchmark

V. Numerical Experiments

In this section we use the railway LC benchmark previously introduced to compare the performances of the *ILP-based* approach for diagnosability analysis of DES modeled with LPN systems. In particular:

- in order to apply the *optimization-based* approach described in Section III-A, a Matlab® script has been used, which calls the FICO™ Xpress [1] API to solve the ILP problem in Theorem 1;
- the DPN-SOG tool [9] has been used as a reference as it implements the SSD-based technique that was already compared to two reference *graph-based* approaches, namely the classic diagnoser technique and the MBRG\BRD based one ([11], [12]), on the basis on the level-crossing benchmark that we consider in the present paper.

Moreover, before discussing the numerical results of the experiments that have been performed, it is important to notice that:

- the current implementation of *graph-based* SSD approach within DPN-SOG permits to assess diagnosability but not $\mathcal{K}$-diagnosability. However, the tool will be extended in

TABLE I

COMPARATIVE EXPERIMENTAL RESULTS

n	Petri net features				Diagnosability via SSD				$\mathcal{K}$-diagnosability via ILP				
	$\|P\|$	$\|T\|$	$\|\mathcal{N}\|$	$\|\mathcal{A}\|$	$\|\mathcal{D}_S\|$	$\|\mathcal{D}_T\|$	$\mathcal{D}_e$ (s)	$\mathcal{D}_m$ (kB)	$\mathcal{K}$	Last_ILP$_e$ (s)	Total_ILP$_e$ (s)	#constr. (origin / Xpress)	#unkow. (origin / Xpress)
1	12	10	20	43	10	14	**0**	44	7	0.3	**9**	721 / 225	228 / 180
2	15	14	142	500	83	205	**0**	1056	13	0.6	**26**	1171 / 467	425 / 380
3	18	18	832	4085	483	1745	**1**	8696	19	0.7	**56**	1729 / 798	682 / 639
4	21	22	4314	27142	2434	11774	**2**	80400	25	1.1	**108**	2395 / 1237	999 / 923
5	24	26	20556	157551	11304	69112	**30**	430456	31	4	**194**	3169 / 1764	1376 / 1294
6	27	30	92070	831384	56136	414299	**458**	2155100	37	2.7	**326**	4051 / 2386	1813 / 1725
7	30	34	393336	4086585	261262	2282890	**7836**	10167015	43	5.6	**507**	5041 / 3110	2310 / 2197
8	33	38	1618866	19013130	*	*	o.t.	*	49	6.4	**767**	6139 / 3940	2867 / 2743
9	36	42	*	*	*	*	o.t.	*	55	8.8	**1079**	7345 / 5006	3484 / 3351
10	39	46	*	*	*	*	o.t.	*	61	11.8	**1514**	8659 / 6013	5822 / 4017
11	42	50	*	*	*	*	o.t.	*	64	20	**1874**	12519 / 6686	11400 / 4555
12	45	54	*	*	*	*	o.t.	*	67	32	**3630**	13962 / 7688	12798 / 5125

*: No result obtained in 4 hours. o.t.: Out of time (more than 4 hours).

- n: the number of tracks;
- $|P|$ and $|T|$: the number of places and transitions in the PN models, respectively;
- $|\mathcal{N}|$ and $|\mathcal{A}|$: the number of nodes and arcs in the reachability graph, respectively;
- $|\mathcal{D}_S|$ and $|\mathcal{D}_T|$: the numbers of nodes and arcs in the SSD, respectively;
- $\mathcal{D}_e$ and $\mathcal{D}_m$: the required time and memory to generate perform the verification respectively;
- $\mathcal{K}$: number of events needed to detect the fault;
- Last_ILP$_e$: the time taken by Xpress to solve the ILP problem that satisfies Theorem 1;
- Total_ILP$_e$: the time taken by Xpress to solve the $\mathcal{K}$ ILP problems needed to assess $\mathcal{K}$-diagnosability;
- #const.: the number of constraints in the ILP problem that satisfies Theorem 1 before and after Xpress presolver, respectively;
- #unkow.: the number of unknowns in the ILP problem that satisfies Theorem 1 before and after Xpress presolver, respectively.

the future in order to make it possible to also tackle $\mathcal{K}$-diagnosability investigation.

- on the other hand, the considered *ILP-based* approach cannot be used to assess non-diagnosability, and hence it cannot be used to elucidate non-diagnosable faults. For this reason, in this comparison we have considered only fault $(t_6\,, bf)$, since the faults belonging to class T_{f_1} are not diagnosable as soon as the benchmark has more than one railway track (see the results presented in [10]).
- the considered *optimization-based* also requires to solve an ILP problem for each value of $\mathcal{K}$ that needs to be tested. For this reason in Table I reports both the time to solve the ILP problem that satisfies Theorem 1 (*Last_ILP$_e$*), and the total time needed to solve the $\mathcal{K}$ ILP problems (*Total_ILP$_e$*).

The numerical experiments carried out to compare the two considered approaches have been run on the same platform; in particular a 64-bit PC equipped with CPU Intel® Core™ i3-6100U, at 2.30 GHz with 4GB of RAM has been used.

The results of the numerical experiments for the benchmark models with a number of tracks that ranges from 1 to 12 are summarized in Table I. Taking into account that a time out of 4 hours has been considered, several observations can be made, as discussed below:

- although the proposed *optimization-based* approach requires to solve a number of ILP problems equal to $\mathcal{K}$ to assess $\mathcal{K}$-diagnosability, as soon as the size of the model becomes relatively large (in our case, as soon as $n > 6$), the time needed to perform the analysis becomes way lower than the one required by the *graph-based* SSD approach[6].
- Given the exponential explosion of the state space, the *graph-based* approach becomes practically unfeasible for $n > 7$, not terminating within the 4 hours timeout on the considered platform. Instead, the *ILP-based* approach does not need to explicitly compute the state space of the system, and is capable to give the result for the model with 10 tracks in less then half an hour. In particular, on the considered platform, the proposed *ILP-based* approach it is possible to assess the diagnosability of the $(t_6\,, bf)$ fault up to $n = 12$ track in about one hour, then for $n > 12$ the memory limit is reached without reaching the timeout.
- The algorithm underlying the SSD-based technique implements an on-the-fly procedure which is, in general, particularly efficient in the case of non-diagnosable models. Indeed, in this case the algorithm stops as soon as an indeterminate cycle is built within the SSD and a negative verdict regarding diagnosability is issued. Hence, in general, a small part of the reachability graph, and likewise of the SSD, is built in this case. However, it is worth recalling that, in the experiments carried out in the present paper, only diagnosable faults were considered.
- Since it does not require the explicit computation of the reachability set of the model, the *ILP-based* approach takes advantage of the parallelism of the proposed

[6]While the SSD algorithm has been directly implemented in C++, the *ILP-based* $\mathcal{K}$-diagnosability approach has been deployed in the Matlab® environment and relies on the FICO™ Xpress API. It follows that, from the implementation point-of-view, there is a time overhead for the latter approach that is bigger than for the former, and this fact may have a non negligible impact when the size of the problem is relatively small.

benchmark. Indeed, while adding an additional track has a significant impact on the size of the model state space, it does not affect too much the efficiency of *ILP-based* approach. This result is achieved thanks to the fact that the algebraic formulation of the constraints of Theorem 1 enables to exploit the parallelism in the dynamic evolution of each track, and that the tracks evolve in parallel. It turns out that the proposed *ILP-based* is particularly well suited for LPN models with a high level of parallelism.
- As already claimed in the introduction, the *ILP-based* approach exploits commercial tools such as FICO™ Xpress for the solution of the ILP problems. Such an approach not only enables the use of efficient optimization codes, but takes also advantage of all the preprocessing processes of these commercial tools. Indeed, in the considered case, the number of constraints and unknowns after the run of the Xpress presolver is always smaller than the one of the original ILP problem in Theorem 1, and this has a positive impact on the time needed to solve the problem.

CONCLUSIONS

In this paper, two approaches to assess diagnosability in DES modeled with LPN have been compared. In particular, the *graph-based* technique based on SSD [8]–[7], and the approach based on the solution of ILP problems presented in [3] have been considered. The comparison was carried out using the railway benchmark presented in [14].

From the numerical experiments that have been performed, it can be concluded that, despite its limitation (i.e. it cannot assess non-diagnosability), the proposed *ILP-based* approach for diagnosability analysis represents a valuable alternative to the *graph-based* approaches, since

- it does not require to develop *ad hoc* software, since it relies on commercial optimization tools that are available off-the-shelf;
- compared with the SSD approach, which proved to be among the more efficient *graph-based* approaches proposed in the literature, given a limit in time for the diagnosability assessment, it allows for dealing with problems whose size is larger.

Acknowledgements: the research leading to these results has been partially supported by Department of Electrical Engineering and Information Technology (DIETI) of University of Naples Federico II under the project MODAL (*MOdel-Driven AnaLysis of Critical Industrial Systems*).

REFERENCES

[1] Xpress-Optimizer - Reference manual, release 31.01. FICO™ Xpress Optimization Suite, April 2017.

[2] F. Basile, P. Chiacchio, and G. De Tommasi. Decentralized $\mathcal{K}$-diagnosability of Petri nets. In *Proc. of the* 11th *International Workshop on Discrete Event Systems (WODES'12)*, pages 214–220, Guadaljara, Mexico, October 2012.

[3] F. Basile, P. Chiacchio, and G. De Tommasi. On $\mathcal{K}$-diagnosability of Petri nets via integer linear programming. *Automatica*, 48(9):2047–2058, 2012.

[4] F. Basile, R. Cordone, and L. Piroddi. A branch and bound approach for the design of decentralized supervisors in Petri net models. *Automatica*, 52:322–333, 2015.

[5] F. Basile, G. De Tommasi, and C. Sterle. Sensors selection for K-diagnosability of Petri nets via Integer Linear Programming. In *Proc. of the* 23rd *Mediterranean Conference on Control and Automation (MED'15)*, pages 168–175, Torremolinos, Spain, June 2015.

[6] M. Blanke, M. Kinnaert, J. Lunze, M. Staroswiecki, and J. Schröder. *Diagnosis and fault-tolerant control*. Springer, 2nd edition, 2006.

[7] A. Boussif. *Contributions to fault diagnosis of discrete-event systems*. PhD thesis, Université Lille Nord de France, University of Lille 1, 2016.

[8] A. Boussif, M. Ghazel, and K. Klai. Combining enumerative and symbolic techniques for diagnosis of discrete-event systems. In 9th *Workshop on Verification and Evaluation of Computer and Communication Systems*, pages 23–33, Bucharest, Romania, September 2015.

[9] A. Boussif, M. Ghazel, and K. Klai. DPN-SOG: A software tool for fault diagnosis of labeled petri nets using the semi-symbolic diagnoser. In *11ème Colloque sur la Modélisation des Systèmes Réactifs (MSR 2017)*, 2017.

[10] A. Boussif, B. Liu, and M. Ghazel. An experimental comparison of three diagnosis techniques for discrete event systems. In *28th International Workshop on Principles of Diagnosis (DX'17)*, 2017.

[11] M. P. Cabasino, A. Giua, S. Lafortune, and C. Seatzu. Diagnosability analysis of unbounded Petri nets. *Proc. of the* 48th *IEEE Conf. on Decision and Control, Shangai, China*, pages 1267–1272, Dec. 2009.

[12] M. P. Cabasino, A. Giua, S. Lafortune, and C. Seatzu. A new approach for diagnosability analysis of Petri nets using verifier nets. *IEEE Trans. Aut. Contr.*, 57(12):3104–3117, 2012.

[13] X. Cong, M. P. Fanti, A. Mangini, and Z. Li. Decentralized Diagnosis by Petri Nets and Integer Linear Programming. *IEEE Trans. on Systems, Man, and Cybernetics: Systems*, In press 2017.

[14] M. Ghazel and B. Liu. A customizable railway benchmark to deal with fault diagnosis issues in DES. In *Proc. of the* 13th *International Workshop on Discrete Event Systems (WODES'16)*, pages 177–182, Xi'an, People's Republic of China, June 2016.

[15] G. Jiroveanu and R. K. Boel. The diagnosability of petri net models using minimal explanations. *IEEE Trans. Aut. Contr.*, 55(7):1663–1668, Jul. 2010.

[16] N. G. Leveson and J. L. Stolzy. Safety analysis using petri nets. *IEEE Trans. on Software Eng.*, (3):386–397, 1987.

[17] B. Liu, M. Ghazel, and A. Toguyéni. On-the-Fly and Incremental Technique for Fault Diagnosis of Discrete Event Systems Modeled by Labeled Petri Nets. *Asian J. Contr.*, 19(5):1659–1671, 2017.

[18] P. J. Ramadge and W. M. Wonham. The control of vector discrete-event systems. *Proc. of IEEE*, 77(1):81–98, Jan. 1989.

[19] M. Sampath, R. Sengupta, S. Lafortune, K. Sinnamohideen, and D. Teneketzis. Diagnosability of Discrete Event Systems. *IEEE Trans. Aut. Contr.*, 40(9):1555–1575, Sep. 1995.

[20] M. Sampath, R. Sengupta, S. Lafortune, K. Sinnamohideen, and D. C. Teneketzis. Failure Diagnosis Using Discrete-Event Models. *IEEE Trans. Contr. Sys. Technol.*, 4(2):105–124, Mar. 1996.

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

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