Proceedings of the 40th IEEE
Conference on Decision and Control
Orlando, Florida USA, December 2001

Supervisory control of railway networks with Petri nets

Alessandro Giua, Carla Seatzu
Department of Electrical and Electronic Engineering, University of Cagliari, Piazza d'Armi — 09123 Cagliari, Italy
Tel: +39 (70) 675-5892. Fax: +39 (70) 675-5900. Email: {giua,seatzu}@diee.unica.it.

Abstract

In this paper we deal with the problem of designing control logic for railway networks using Petri nets. We first use the framework of supervisory control theory, taking into account the presence of uncontrollable and unobservable transitions, to derive a maximally permissive control policy that ensures safeness. The corresponding controller takes the form of monitor places, possibly with self-loops. In a second step, we investigate the liveness problem and present an heuristic technique based on structural analysis that, whenever applicable, leads to live models. As an example, we consider a segment of the railway network in Sardinia, Italy.

1 Introduction

The specification, analysis and implementation of railway control logic has ever been an important activity since trains and railways were invented centuries ago, and failure of control logic can lead to railway accidents and loss of human life. At present time, this activity is even more important because railway networks are often large, the speed of trains and traffic density is increasing, and activities within networks are taking place concurrently and at geographically different locations. As a result, the overall complexity of railway systems increases, and hence greater demands are placed on the control logic of these systems [13]. Note that the control of a railway network can be divided into two distinct phases. The first one, at a lower level, imposes the satisfaction of a series of safeness constraints (collision avoidance) and liveness constraints (deadlock freeness). The second one, at a higher level, is concerned with the problem of scheduling both the departures and the stops, so as to optimize the efficiency of the net. In this paper the attention is uniquely devoted to the first phase.

We focus our attention on the modeling and control of railway networks with Petri nets [17], that provide a powerful framework for the analysis and control of distributed and concurrent systems. Some of the advantages of Petri nets as models for discrete event control include [10]: graphical representation, solid foundations based in mathematics, the existence of simulation and formal analysis techniques, and the existence of computer tools for simulation, analysis and control. The literature on modelling and analyzing railway systems using Petri nets is not extensive and a good survey is given by Janczura in [13]. The idea of applying Petri net theory goes back to Genrich [7], then it was revisited in [2, 14] and in [12] where coloured Petri nets have been used. Significant contributions in this field are also due to Decknatel and Schnieder [4] and Di Febbraro *et al.* [6] who used hybrid Petri nets to model transportation systems.

The original contribution of our paper with respect to the above mentioned approaches, concerns three aspects.

Modeling. We show how it is possible to model a railway network using Petri nets with (un)controllable and/or (un)observable transitions, following the paradigm of supervisory control [18]. As an example, a controllable and observable transition is associated to the crossing of a section controlled by a semaphore, where a traffic signal that may stop a train and a sensor that detects the passing of the train is placed. The possibility offered by supervisory control to handle such primitives as uncontrollable and unobservable events leads to a very simple model that can be directly exploited in the subsequent phase of control synthesis. The use of Petri nets allows a modular representation of railway networks where each of the composed subnets corresponds to a station or a track.

Control. There exist several techniques for automatically designing controllers for P/T nets with uncontrollable and/or unobservable transitions [10]. In particular, we show how collision avoidance constraints can be expressed as Generalized Mutual Exclusion Constraints (GMECs) [8] and how the corresponding controller takes the form of a set of monitor places that can be computed using Moody's parametrization [16]. However, it is well known that in general a monitor-based solution to a GMEC may not be maximally permissive. In a previous work [5], we showed that this is the case for constraints related to the arrival and departure of a train from a station, where the designed monitor controller is too restrictive and leads to a *local deadlock*. In [5] we also showed that — although the maximally permissive control policy corresponds to a set of legal markings that is not convex and thus cannot be enforced by a monitor place — the corresponding control structure is still very simple and takes the form of a "monitor with self-loops". A nice feature of this approach is that the whole control problem can be divided into a certain number of sub–problems, thus making the proposed control procedure suitable even for large dimensions cases.

Deadlock avoidance. While in our previous paper [5] the focus was on the modeling and safeness-enforcing control, in this paper we address the problem of *global deadlock* avoidance and *simulation*. In fact, when all the previous modules are put together, it may well be the case that the net reaches a deadlock marking, i.e., a blocking state from which no further evolution is possible. We provide a solution to this problem applying siphon analysis to a simplified net (that we call skeleton) and adding new monitors that, controlling the net siphons to prevent them from becoming empty, ensure liveness for this system. To compute the liveness-enforcing monitors, we use a very efficient linear algebraic technique that does not require the exhaustive enumeration of all siphons, whose number may be too large even for small nets such as the one we consider. This technique is applied to the Petri net model of a short segment of the railway in Sardinia, Italy.

Finally, we show how it is possible to determine the maximum number of trains a given railway network can manage effectively through numerical simulations.

As the scope of this paper is to show how several Petri nets techniques, from monitor based supervisory control to siphon analysis can be successfully applied to a real problem, we prefer to keep the discussion at an informal level. References are given, however, to more technical papers where all the used results are formally proven.

0-7803-7061-9/01/$10.00 © 2001 IEEE

2 Background

2.1 Generalities on Petri nets

In this section we recall the formalism used in the paper. For more details on Petri nets we address to [17].

A *Place/Transition net* (P/T net) is a structure $N = (P, T, \boldsymbol{Pre}, \boldsymbol{Post})$, where P is a set of m places; T is a set of n transitions; $\boldsymbol{Pre} : P \times T \to \mathbb{N}$ and $\boldsymbol{Post} : P \times T \to \mathbb{N}$ are the *pre*– and *post*– incidence functions that specify the arcs; $\boldsymbol{C} = \boldsymbol{Post} - \boldsymbol{Pre}$ is the incidence matrix.

A *marking* is a vector $\boldsymbol{m} : P \to \mathbb{N}$ that assigns to each place of a P/T net a non–negative integer number of tokens, represented by black dots. In the following we denote as m_i the marking of place p_i. A P/T *system* or *net system* $\langle N, \boldsymbol{m}_0 \rangle$ is a net N with an initial marking $\boldsymbol{m}_0$ and its set of reachable markings is denoted $R(N, \boldsymbol{m}_0)$.

A non-null vector $\boldsymbol{x} \in \mathbb{N}^m$ such that $\boldsymbol{x}^T \boldsymbol{C} = \boldsymbol{0}$ is called a *P–semiflow* (or *P–invariant*) of the net N. The *support* of a P–semiflow is the set of places p_i such that $x_i > 0$. Let $\boldsymbol{X}$ be a matrix where each column is a P–semiflow of N, and denote $\mathcal{I}_X(N, \boldsymbol{m}_0) = \{\boldsymbol{m} \in \mathbb{N}^m \mid \boldsymbol{X}^T \boldsymbol{m} = \boldsymbol{X}^T \boldsymbol{m}_0\}$. Then $R(N, \boldsymbol{m}_0) \subseteq \mathcal{I}_X(N, \boldsymbol{m}_0)$.

A P/T net is called *ordinary* when all of its arc weights are 1's. A *siphon* of an ordinary net is a set of places $S \subseteq P$ such that: $\bigcup_{p \in S} {}^\bullet p \subseteq \bigcup_{p \in S} p^\bullet$. A siphon is *minimal* if it is not the superset of any other siphon. The number of tokens assigned to the siphon S by a marking $\boldsymbol{m}$ is $\boldsymbol{m}(S) = \sum_{p_i \in S} m_i$. A siphon can also be described by its *characteristic vector* $\boldsymbol{s} \in \{0,1\}^m$ such that $s_i = 1$ if $p_i \in S$, else $s_i = 0$; thus $\boldsymbol{m}(S) = \boldsymbol{s}^T \boldsymbol{m}$.

2.2 GMECs, monitors and controllability

The development of this subsection is kept very concise for sake of brevity. Please, refer to [16] for a more complete discussion of this topic.

Assume we are given a set of legal markings $\mathcal{L} \subseteq \mathbb{N}^m$, and consider the basic control problem of designing a supervisor that restricts the reachability set of the plant in closed loop to $\mathcal{L} \cap R(N, \boldsymbol{m}_0)$. Of particular interest are those PN state–based control problems where the set of legal markings $\mathcal{L}$ is expressed by a set of n_c linear inequality constraints called *Generalized Mutual Exclusion Constraints* (GMECs).

Each GMEC is a couple $(\boldsymbol{w}, k)$ where $\boldsymbol{w} : P \to \mathbb{Z}$ is a $m \times 1$ weight vector and $k \in \mathbb{Z}$. Given the net system $\langle N, \boldsymbol{m}_0 \rangle$, a GMEC defines a set of markings that will be called *legal markings*: $\mathcal{M}(\boldsymbol{w}, k) = \{\boldsymbol{m} \in \mathbb{N}^m \mid \boldsymbol{w}^T \boldsymbol{m} \le k\}$. The markings that are not legal are called *forbidden markings*. A controlling agent, called supervisor, must ensure that the forbidden markings will be not reached. So the set of legal markings under control is $\mathcal{M}_c(\boldsymbol{w}, k) = \mathcal{M}(\boldsymbol{w}, k) \cap R(N, \boldsymbol{m}_0)$.

In the presence of multiple constraints, all constraints can be grouped and written in matrix form as

$$\boldsymbol{W}^T \boldsymbol{m} \le \boldsymbol{k} \tag{1}$$

where $\boldsymbol{W} \in \mathbb{Z}^{m \times n_c}$ and $\boldsymbol{k} \in \mathbb{Z}^{n_c}$. The set of legal markings is $\mathcal{M}(\boldsymbol{W}, \boldsymbol{k}) = \{\boldsymbol{m} \in \mathbb{N}^m \mid \boldsymbol{W}^T \boldsymbol{m} \le \boldsymbol{k}\}$.

Each constraint requires the introduction of a new place (denoted as *monitor place*). To each monitor place, it corresponds an additional row in the incidence matrix of the closed loop system. In particular, let $\boldsymbol{C}_c$ be the matrix that contains the arcs connecting the monitor places to the transitions of the plant, and $(\boldsymbol{m}_{c0})$ $\boldsymbol{m}_c$ the (initial) marking of the monitors. The incidence matrix $\boldsymbol{C} \in \mathbb{Z}^{(m+n_c) \times n}$ of the closed loop system is

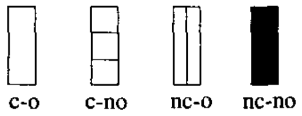

Figure 1: *Convention on transitions:* c *controllable,* o *observable,* nc *uncontrollable,* no *unobservable.*

$$\boldsymbol{C} = \begin{bmatrix} \boldsymbol{C}_p \\ \boldsymbol{C}_c \end{bmatrix} \tag{2}$$

and the marking vector $\boldsymbol{m} \in \mathbb{Z}^{m+n_c}$ and initial marking $\boldsymbol{m}_0$ are

$$\boldsymbol{m} = \begin{bmatrix} \boldsymbol{m}_p \\ \boldsymbol{m}_c \end{bmatrix}, \quad \boldsymbol{m}_0 = \begin{bmatrix} \boldsymbol{m}_{p0} \\ \boldsymbol{m}_{c0} \end{bmatrix}, \tag{3}$$

where the subscript p has been used to denote the variables of the plant.

In the case of controllable and observable transitions, Giua *et al.* provided the following theorem.

Theorem 1 ([8]) *If* $\boldsymbol{k} - \boldsymbol{W}^T \boldsymbol{m}_0 \ge \boldsymbol{0}$ *then a Petri net controller with incidence matrix* $\boldsymbol{C}_c = -\boldsymbol{W}^T \boldsymbol{C}_p$ *and initial marking* $\boldsymbol{m}_{c0} = \boldsymbol{k} - \boldsymbol{W}^T \boldsymbol{m}_{p0}$ *enforces constraint (1) when included in the closed loop system (2) with marking (3).*

The controller so constructed is maximally permissive, i.e. it prevents only transitions firings that yield forbidden markings. The controller net has n_c monitor places and no transition is added.

It often occurs that certain transitions can not be disabled by any control action (*uncontrollable transitions*) or their firing can not be directly detected or measured (*unobservable transitions*). We adopt the convention reported in figure 1 to distinguish among controllable and/or uncontrollable, observable and/or unobservable transitions.

An admissible monitor must satisfy two structural conditions [15, 16] when uncontrollable or unobservable transitions exist. No arcs is allowed from a monitor to an uncontrollable transition t, so that t can never be disabled by the controller. An unobservable transition must have the same number of input and output arcs to/from a monitor — i.e. its only admissible connection to a monitor is given by self–loops — so that its firing does not change the state of the controller and thus can never be detected.

If the monitor constructed applying the previous theorem does not satisfy these structural conditions, an appropriate set of transformed constraints (more restrictive than the original ones) needs to be determined so as to construct a Petri net controller. A general technique to do this with little more than the integer triangularization of a suitable matrix was presented in [15, 16]. An example of constraint transformation is given in section 4.

2.3 Constraints involving the firing vector

Certain control goals may involve the firing vector of a Petri net as well as the tokens content of places [16]. A constraint of this kind takes the form:

$$\boldsymbol{w}^T \boldsymbol{m} + v_j q_j \le k \tag{4}$$

where $v_j \in \mathbb{N}$, and $q_j \in \{0, 1\}$ is such that $q_j = 1$ if t_j is enabled, otherwise $q_j = 0$. Thus, constraint (4) implies

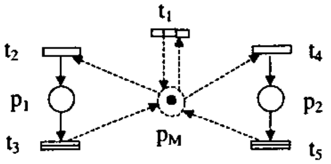

Figure 2: *A monitor with self-loop.*

that $w^T m \leq k$ and that transition t_j should be enabled if $k - w^T m \geq v_j$.

The corresponding control structure takes the form of a monitor place with a self-loop. As an example, in figure 2 we have shown the monitor with self-loop p_M that enforces the constraint $m_1 + m_2 + q_1 \leq 1$. Note that transition t_1 must be controllable, transitions t_2 and t_4 must be controllable and observable, transitions t_3 and t_3 must be observable.

3 Modeling railway networks with Petri nets

In this section we show how Petri nets can be efficiently used as a modeling tool for railway networks. In particular, we show that the whole network can be seen as the composition of a certain number of elementary modules, namely tracks and stations.

3.1 The track model

An example of Petri net modeling a track is shown in figure 3. It consists of two series of places ($p_1, \cdots, p_5$ and $p'_1, \cdots, p'_5$) and transitions ($t_1, \cdots, t_4$ and $t'_1, \cdots, t'_4$), each one representative of the flow of trains in a certain direction. Each couple of places p_i, p'_i represents a segment of the track, i.e. the marking of either p_i or p'_i denotes the presence of a train in the segment. Note that, in the case of a double track, the two lines are independent and places p_i and p'_i correspond to parallel segments and can be marked simultaneously. On the contrary, in the case of a single track two places, p_i and p'_i, are used to represent the same segment of the track that can be crossed in both directions, but places can not be marked at the same time. Note that during simulation a release delay is associated to each transition, to represent the time a train requires to run along that segment.

Transitions may be (un)controllable and/or (un)observable. In this setting, a transition that is both controllable and observable represents a semaphore (see transitions t_3 and t'_2 in figure 3), i.e., in that point of the net the presence of a train can be detected and its transit can be forbidden. In all real situations a semaphore is placed at the exit of a track, or equivalently at the entrance of a station.

A transition that is observable but not controllable (see transitions t_1, t_4, t'_1 and t'_4), represents a sensor counting the number of axles of the train, i.e., the number of cars passing through that point.

The number of places used to represent the track depends on the required precision. On one hand, we assume that the Petri net is safe (such a condition will be imposed by the addition of appropriate monitor places), thus the number of places is mainly limited by the required safeness distance, i.e., we assume that the length of each segment is such that no more than one train can be contained within it at any given time instant. On the other hand, we take into account the presence of sensors and semaphores that are modeled by appropriate transitions as discussed above. Note that, even if these elements are only associated to one direction of flow, an equal number of uncontrollable and unobservable transitions should be added in the other direction so as to keep the structure shown in figure 3.

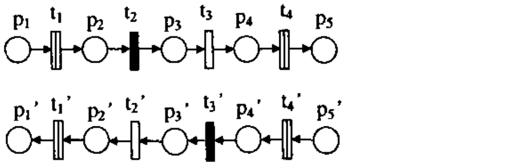

Figure 3: *The Petri net model of a track.*

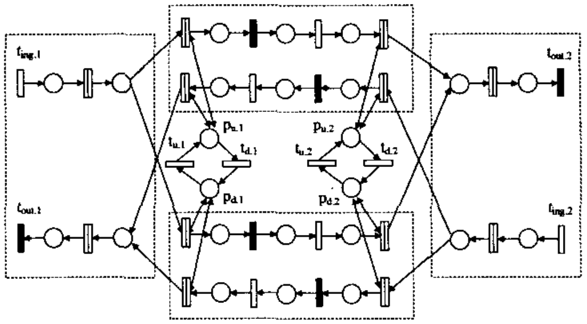

Figure 4: *The Petri net model of a two–tracks station.*

3.2 The railway station model

The Petri net model of a two–tracks railway station is sketched in figure 4, where double arrows have been used to denote self–loops. The station is composed of two stretches, whose models are analogous to that already presented in the previous subsection.

The firing of controllable and observable transitions $t_{ing,1}$ and $t_{ing,2}$ represents the input of a train in the station, while the firing of uncontrollable and unobservable transitions $t_{out,1}$ and $t_{out,2}$ represents the output of a train from the station. Note that, as in the case of the track model, a controllable and observable transition is used to model a semaphore, while an observable but uncontrollable transition is used to model an axles counter.

The two cycles $p_{u,1}, t_{d,1}, p_{d,1}, t_{u,1}$ and $p_{u,2}, t_{d,2}, p_{d,2}, t_{u,2}$ model the points, i.e., when places $p_{u,1}$ and $p_{u,2}$ are marked, trains are directed to the up–track or may leave the up–track; on the contrary, when places $p_{d,1}$ and $p_{d,2}$ are marked, trains are directed to the down–track or may leave the down–track.

Note that this model can be easily extended to an arbitrary number of tracks. An example of a three–tracks railway station may be found in [5].

4 The controller design for tracks and stations

In this paper we shall deal with the problem of designing a Petri net supervisor for a railway network so as to ensure safeness. In other words, the goal of the supervisor is that of guaranteeing that trains may flow through the net without colliding.

Safeness constraints can be written as GMECs that ensure that each couple of places corresponding to the same segment of a single–track (that may also belong to a station) are not marked simultaneously, and that each place never contains more than one token at a time. As an example, a constraint of the form

$$m_i + m'_i \leq 1 \qquad (5)$$

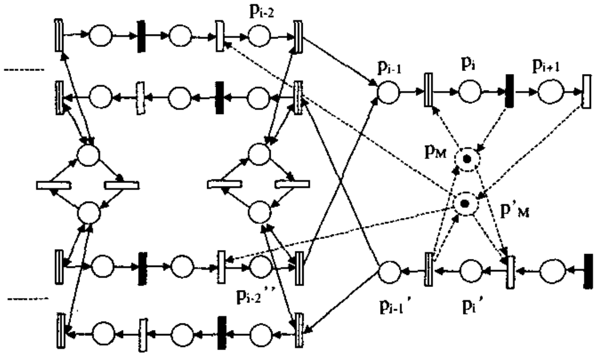

Figure 5: *Monitor place relative to the constraint $m'_i + m''_i \leq 1$ assuming that all transitions are controllable and observable (p_M) and taking into account the uncontrollability and unobservability of transitions (p'_M).*

relative to a given segment of a track adjacent to a station (see figure 5) ensures that places p_i and p'_i are not marked at the same time and each place never contains more than one token.

In accordance to the supervisory control theory briefly summarized in section 2.2 each constraint requires the introduction of a monitor place. If all transitions were controllable and observable, the monitor ensuring the satisfaction of (5) would have been place p_M in figure 5.

In the case of uncontrollable and/or unobservable transitions, constraints need to be appropriately transformed. In fact, place p_M is not an admissible supervisor because it has arcs going to uncontrollable transitions and arcs coming from unobservable ones. If we move "upwards" the arcs going to uncontrollable transitions until we reach controllable ones, and "downwards" the arcs coming from unobservable transitions until we reach observable ones, we obtain the monitor p'_M in figure 5, that corresponds to the more restrictive constraint

$$m_{i-2} + m''_{i-2} + m_{i-1} + m_i + m_{i+1} + m'_i \leq 1. \qquad (6)$$

A formal algorithm for constraint transformation is given in [15, 16].

Although we need to add a number of constraints equal to the number of track segments (this number is about half the number of places in the net), once all constraints have been transformed one finds out that many of them are redundant and can be discarded.

Constraints of this kind ensure safeness. Nevertheless, it can be proved [5] that they reveal to be too restrictive when applied to places relative to tracks within the stations, while they ensure a satisfactory behaviour of the net when imposed to places modeling the intermediate tracks. A better solution to this problem consists in the introduction a new set of constraints also involving the firing vector that regulate the input of trains in the stations, and the points within them.

Let us consider a two–tracks railway station. The detailed Petri net model is reported in figure 4. Nevertheless, when imposing logical constraints, we do not need such a detailed model, and it is enough to consider the Petri net in figure 6 that is obtained from the previous one by simply grouping together some places.

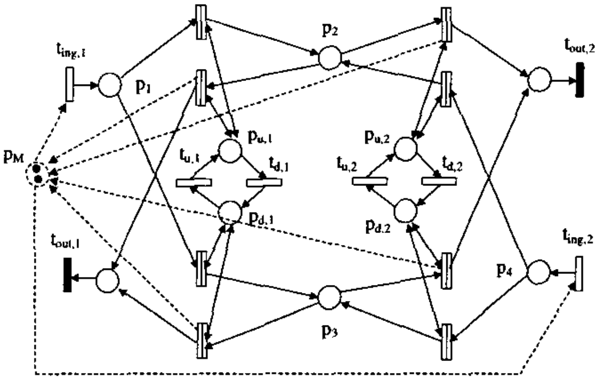

Figure 6: *The reduced Petri net model of a two–tracks railway station and the monitor place relative to constraint 7.a.*

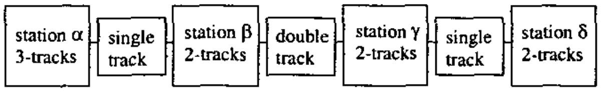

Figure 7: *Scheme of the railway network.*

The set of constraints, most of which also involve the firing vector, is:

$$\left\{ \begin{array}{lr} m_1 + m_2 + m_3 + m_4 \leq 2 & (a) \\ q_{ing,2} + m_{u,2} + m_2 \leq 2 & (b) \\ q_{ing,2} + m_{d,2} + m_3 \leq 2 & (c) \\ q_{u,2} + m_2 + m_4 \leq 2 & (d) \\ q_{d,2} + m_3 + m_4 \leq 2 & (e) \\ q_{ing,1} + m_{u,1} + m_2 \leq 2 & (f) \\ q_{ing,1} + m_{d,1} + m_3 \leq 2 & (g) \\ q_{u,1} + m_1 + m_2 \leq 2 & (h) \\ q_{d,1} + m_1 + m_3 \leq 2 & (i) \end{array} \right. \qquad (7)$$

where (a)–(c) and (f)–(g) regulate the input of trains in the station, while (d), (e), (h), (i) regulate the points.

These constraints can be forced by simply introducing appropriate monitor places as illustrated in section 2.3. As an example, in figure 6 we have reported the monitor place relative to constraint 7.a.

5 Liveness constraints in a real network

We now consider the railway system sketched in figure 7, that represents a short segment between the stations of Chilivani and Olbia, in Sardinia, Italy. It consists of four stations, where the first one is a three–tracks station while the others are two–tracks stations. All intermediate tracks are single tracks, apart from the second one where two trains may travel in opposite directions simultaneously.

A skeleton Petri net model of the network (at this level of abstraction all transitions can be considered as controllable and observable) is shown in figure 8; here the monitors inside rectangles limit the number of trains within stations and tracks according to each station or track capacity. The monitor place p_0 contains the maximum number B of trains that may be allowed into the network.

It is easy to verify using this skeleton model that when different modules are put together several blocking conditions may occur. Consider the case in which two trains are in the station β directed towards station α (place p_9 contains two tokens) and one train has already left station α and is moving towards station β (place p_4 contains one token). When such a marking is reached places p_5 and p_8 are empty and the net reaches a partial deadlock.

We present a technique, based on the analysis of the skeleton net, to determine a maximally permissive liveness enforcing control policy.

We first observe that the skeleton net belongs to the class of ES^3PR nets (a subclass of ordinary PN's) defined in [19]. To ensure liveness of the model, we use a result presented in [19].

Proposition 2 *Let $\langle N, \boldsymbol{m}\rangle$ be a marked ES^3PR net. If a transition $t \in T$ is dead for a reachable marking $\boldsymbol{m}$, then there exists a reachable marking $\boldsymbol{m}'$ and siphon $S \neq \emptyset$ such that $\boldsymbol{m}(S) = 0$, i.e., all places in the siphon S are empty.*

We then adopt a standard technique [19] to enforce liveness: determine if there are siphons in the net that can become empty and if so add a monitor to control them and prevent this. There are two problems with this technique: first of all the new monitors may create new siphons that may need to be controlled as well; secondly, the addition of new monitors may lead to a net that is not ordinary any more. However, for this particular example, the procedure could be successfully applied to all cases in which place p_0 is initially marked with $B \leq 7$ tokens, in the sense that by adding new monitors the net always remains ordinary and after a finite number of steps the net converges to a structure where no siphon may become empty.

Note that the methodology presented in [11], that allows one to transform non-ordinary nets into ordinary ones, may also be used to continue deadlock analysis for a larger number of trains.

We sketch how we proceeded to compute the liveness-enforcing monitors, using a linear algebraic technique based on integer programming that does not require the exhaustive enumeration of all siphons, whose number is too large even for a small net such as the one we consider. Although solving a linear integer optmization problem is still an NP complete problem (as is siphon enumeration) we observed that in practice the integer programming approach is much more efficient. This technique is inspired by other linear algebraic approaches appeared in the literature, in particular by the results of [3].

First of all we observe that the net in figure 8 has 9 semiflows corresponding to the monitors places $p_0, p_2, p_5, \cdots, p_{21}$ shown as dashed circles; the places in the support of each semiflow are shown within a rectangle, except for the semiflow corresponding to place p_0 whose support contains all places in the net. Thus the reachable set of the net can be approximated as

$$R(N, \boldsymbol{m}_0) \subseteq \mathcal{I}_X(N, \boldsymbol{m}_0) = \{\boldsymbol{m} \in \mathbb{N}^m \mid \boldsymbol{X}^T\boldsymbol{m} = \boldsymbol{k}\}$$

where each column of the the 23×9 matrix $\boldsymbol{X}$ contains a P-semiflow and $\boldsymbol{k} = \boldsymbol{X}^T\boldsymbol{m}_0$ is a 9×1 vector whose components represent the token content of each semiflow. Although we cannot formally prove that $R(N, \boldsymbol{m}_0) = \mathcal{I}_X(N, \boldsymbol{m}_0)$ if we can enforce that no deadlock marking $\boldsymbol{m} \in \mathcal{I}_X(N, \boldsymbol{m}_0)$ is reachable, then no reachable marking may be a deadlock. Thus in the following we use the previous equation as a characterization of marking rechability.

To determine if there are siphons that need to be controlled one may use the following non-linear integer program:

$$\left\{\begin{array}{ll} \min & \boldsymbol{s}^T\boldsymbol{m} \\ s.t. & \mathrm{sgn}(\boldsymbol{Pre}^T\boldsymbol{s}) \geq \mathrm{sgn}(\boldsymbol{Post}^T\boldsymbol{s}) \\ & \boldsymbol{X}^T\boldsymbol{m} = \boldsymbol{k} \\ & \boldsymbol{1}^T\boldsymbol{s} \geq 1 \end{array}\right. \tag{8}$$

where $\boldsymbol{s} \in \{0,1\}^m$ and $\boldsymbol{m} \in \mathbb{N}^m$ are the unknowns, and $\mathrm{sgn}(\boldsymbol{x})$ is a vector whose i-th component is 1 (resp., 0, -1) if the i-th component of $\boldsymbol{x}$ is positive (resp., null, negative). The equation $\mathrm{sgn}(\boldsymbol{Pre})^T\boldsymbol{s}) \geq \mathrm{sgn}(\boldsymbol{Post}^T\boldsymbol{s})$ ensures that $\boldsymbol{s}$ is the characteristic vector of a siphon S, the second equation ensures that $\boldsymbol{m}$ is reachable, and the equation $\boldsymbol{1}^T\boldsymbol{s} \geq 1$ ensures that S is not the empty set. Thus a solution $(\boldsymbol{m}, \boldsymbol{s})$ of this program with optimal value $\boldsymbol{s}^T\boldsymbol{m} = 0$ corresponds to a reachable marking $\boldsymbol{m}$ such that the siphon S with characteristic vector $\boldsymbol{s}$ is empty.

The non-linearity of the previous program is an undesirable feature, that makes solving it a hard task. We convert it to an equivalent (linear) integer program

$$\left\{\begin{array}{ll} \min & \boldsymbol{1}^T\boldsymbol{s} \\ s.t. & K_1\boldsymbol{Pre}^T\boldsymbol{s} \geq \boldsymbol{Post}^T\boldsymbol{s} \\ & \boldsymbol{X}^T\boldsymbol{m} = \boldsymbol{k} \\ & K_2\boldsymbol{s} + \boldsymbol{m} \leq K_2\boldsymbol{1} \\ & \boldsymbol{1}^T\boldsymbol{s} \geq 1 \end{array}\right. \tag{9}$$

where $K_1 = \max\{\boldsymbol{1}^T\boldsymbol{Post}(\cdot, t) \mid t \in T\}$ and $K_2 = \max\{m(p) \mid p \in P, \boldsymbol{m} \in R(N, \boldsymbol{m}_0)\}$ (for the net in figure 8 $K_1 = 2$ and $K_2 = B$). We claim (a formal proof can be found in [1]) that the program (8) has an optimal solution $(\boldsymbol{m}, \boldsymbol{s})$ such that $\boldsymbol{s}^T\boldsymbol{m} = 0$ if and only if the program (9) has an admissible solution. In fact, the first constraint in (9) is perfectly equivalent to the first constraint in (8), while the new constraint in (9) (the third one) ensures that for all $p_i \in P$, $K_2s_i + m_i \leq K_2$ holds, i.e., either $s_i = 1$ and $m_i = 0$ or (*exclusive or*) $s_i = 0$ and $m_i > 0$. The objective function chosen for the program (9) ensures that only minimal siphons are computed.

We started with a value of $B = 3$ and applied the previously described approach to determine siphons to be controlled. As such a siphon is found, we add a new monitor to the net to prevent the siphon from becoming empty. After a few steps the procedure converges to a live net. We increase the value of B of one token and continue the procedure.

These are the GMEC's corresponding to the liveness enforcing monitors determined with the previous procedure as B goes from 1 to 7.

$$\left\{\begin{array}{l} m_4 + m_9 \leq 2; \quad m_{14} + m_{19} \leq 2; \quad m_{17} + m_{22} \leq 2; \\ m_1 + m_6 \leq 3; \quad m_{14} + m_{22} \leq 3; \quad m_1 + m_9 \leq 4; \\ m_7 + m_{10} + m_{12} + m_{16} \leq 5; \\ m_7 + m_{10} + m_{12} + m_{14} + m_{16} + m_{19} \leq 6; \\ m_4 + m_7 + m_9 + m_{10} + m_{12} + m_{16} \leq 6; \end{array}\right. \tag{10}$$

When $B = 8$, the procedure finds empty siphons that cannot be controlled by ordinary monitors; thus we have to stop.

Although we are able to ensure liveness of the model for a number of trains up to 7, we will show in the following subsection that it is desirable to allow no more than 5 trains in the network to bound the time it takes a train to go from one end station to the other one.

5.1 Numerical simulations

In this subsection we present the results of some numerical simulation performed via the software SIRPHYCO [9]. During numerical simulations we associate a time delay to each transition, corresponding to the time a train requires to cross over a given segment, or equivalently, in the case of the points models, it denotes the time required to change the enabled track. More precisely, we assume stochastic transitions, with an ex-

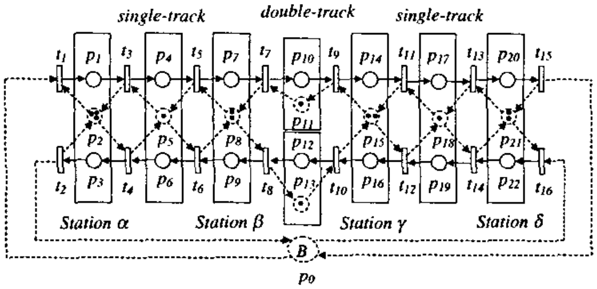

Figure 8: *The skeleton Petri net model of the railway network in figure 7.*

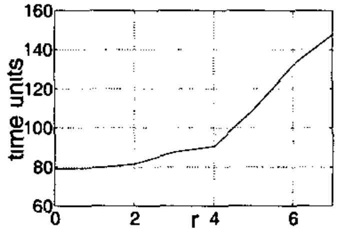

Figure 9: *The results of the numerical simulation presented in subsection 5.1 .*

ponentially distribute law, thus the chosen time delays represent average values.

We assume that one train starts moving from station α to station δ while r trains (with $r \geq 0$) are moving from station δ to α. We compute the traversal time of first train, i.e., the time it spends within the net before reaching station δ and leaving the net. This traversal time grows with r, as shown in figure 9. In particular, we observe that the traversal time significantly increases for $r > 4$, i.e., we may conclude that $B = r + 1 = 5$ is the maximum number of trains the considered net can effectively manage.

6 Conclusions and future works

In this paper we have investigated the problem of modelling railway networks with Petri nets so as to apply supervisory control to automatically design a controller that both ensures safeness and liveness. Transitions have been assumed either (un)controllable or (un)observable so as to represent sensors and semaphores.

The procedure we have proposed is based on a modular representation of the net, thus making it easily extensible to even large dimensions problems. We used both generalized mutual exclusion constraints and constraints involving the firing vector, and the corresponding control structures take the form of monitor places.

Our future work will be that of scheduling both the departures and the stops, so as to optimize the efficiency of the net.

Acknowledgements. The authors would like to thank Asma Ghaffari and Xiaolan Xie for their useful comments and valuable discussions.

References

[1] F. Basile, P. Chiacchio, A. Giua, C. Seatzu "Deadlock recovery of controlled Petri net models using observers," *8th IEEE Int. Conf. on Emerging Technologies and Factory Automation*, Antibes, France, October 2001.

[2] J. Billington, "Many-sorted high-level nets," *Proc. of the 3rd Int. Work. on Petri Nets and Performance Models*, Kyoto, Japan, pp. 166–179, 1989.

[3] F. Chu, X. Xie, "Deadlock Analysis of Petri Nets Using Siphons and Mathematical Programming," *IEEE Trans. on Robotics and Automation*, Vol. 13, No. 6, pp. 793–804, 1997.

[4] G. Decknatel, and E. Schnieder, "Modelling Railway Systems with Hybrid Petri Nets," *Proc. 3rd Int. Conf. on Automation of Mixed Processes*, Reims, France, March 1998.

[5] F. Diana, A. Giua, C. Seatzu "Safeness-Enforcing Supervisory Control for Railway Networks," *2001 IEEE/ASME Int. Conf. on Advanced Intelligent Mechatronics*, Como, Italy, July 2001.

[6] A. Di Febbraro, and A. Ferrara, "A New Two-Level Model for Multiclass Freeway Traffic," *IEEE Trans. on Vehicular Technology*, pp. 189–200, 1996.

[7] H.J. Genrich, "Predicate/Transition nets," In W. Brauer, W. Reisig, and G. Rozenberg (eds), *Advances in Petri nets*, Lecture Notes in Computer Science, Vols. 254 and 255, Springer Verlag, 1987.

[8] A. Giua, F. DiCesare, M. Silva, "Generalized Mutual Exclusion Constraints for Nets with Uncontrollable Transitions", *Proc. IEEE Int. Conf. on Systems, Man & Cybernetics*, Chicago, USA, pp. 974–979, October 1992.

[9] G. Guerre-Chaley, "Conception et réalisation d'un simulateur de réseaux de Petri continus et hybrides pour l'evaluation de performances des systmes discrets et/ou continus," *CNAM dissertation, Conservatoire National des Arts et Métiers*, Grenoble, France, April 1997.

[10] L. E. Holloway, B. H. Krogh, A. Giua, "A Survey of Petri Net Method for Controlled Discrete Event Systems", *Discrete Event Systems*, Vol. 7, pp. 151-190, 1997.

[11] M. V. Iordache, J. O. Moody and P. J. Antsaklis, "Automated Synthesis of Deadlock Prevention Supervisors Using Petri Nets", *ISIS Techinal Report ISIS 2000-003*, May 2000.

[12] K. Jensen, *Coloured Petri Nets. Basic concepts, analysis methods and practical use. Volume 2: Analysis methods*, EATCS Monographs on Theoretical Computer Science, Springer Verlag, 1994.

[13] C.W. Janczura, "Modelling and analysis of railway network control logic using coloured Petri nets," *Ph.D. Thesis*, University of South Australia, August 1998.

[14] N.G. Levson and J.L. Stolzy, "Safety analysis using Petri nets," *IEEE Trans. on Software Engineering*, Vol. 13, N. 3, pp. 386–397, 1987.

[15] J.O. Moody, K. Yamalidou, M.D. Lemmon and P.J. Antsaklis, "Feedback control of Petri nets based on Place Invariants," *Automatica*, Vol. 32, N. 1, pp. 15–28, January 1996.

[16] J.O. Moody, P.J. Antsaklis, "Supervisory control of discrete event systems using Petri nets," Kluwer Academic Publishers, 1998.

[17] T. Murata, "Petri nets: properties, analysis and applications," *Proc. of the IEEE*, Vol. 77, N. 4, pp. 541–580, April 1989.

[18] P.J. Ramadge and W.M. Wonham, "Control of Discrete-Event Systems", *Proc. of the IEEE*, Vol. 77, N. 1, pp. 81-98, January 1989.

[19] F. Tricas, F. García-Vallés, J.M. Colom, and J. Ezpeleta, "A Structural Approach to the Problem of Deadlock Prevention in Process with Resources," *Proc. WODES98: 4th Int. Work. on Discrete Event Systems* , (Cagliari, Italy), pp. 273–278, August, 1998.