A continuous Petri net approach for model predictive control of traffic systems

Jorge Júlvez and René Boel

***Abstract*—Traffic systems are often highly populated discrete event systems that exhibit several modes of behavior such as free flow traffic, traffic jams, stop-and-go waves, etc. An appropriate closed loop control of the congested system is crucial in order to avoid undesirable behavior. This paper proposes a macroscopic model based on continuous Petri nets as a tool for designing control laws that improve the behavior of traffic systems. The main reason to use a continuous model is to avoid the state explosion problem inherent to large discrete event systems. The obtained model captures the different operation modes of a traffic system and is highly compositional. In order to handle the variability of the traffic conditions, a model predictive control strategy is proposed and validated.**

I. Introduction

The behavior of a traffic system greatly depends on the density of vehicles in the traffic network and on the rules governing the flow of traffic, such as the switching control of traffic lights. Traffic models should cope with different modes of operation depending on the state and traffic conditions of the system. The use of traffic models gives one the chance to analyze, to simulate and to predict the future behavior of traffic systems. Thus these models enable the model based design of feedback control strategies, the application of which improves important traffic performance measures such as throughput, delay and fuel consumption.

The state of a traffic system is usually given by the discrete values counting the number of vehicles present in the different sections of the traffic network. Hence, in principle discrete event models (see [1], [2], [3] and references therein) are appropriate to accurately describe the behavior of traffic systems. Unfortunately, highly populated discrete systems suffer from the state explosion problem that makes the analysis of the system performance extremely difficult. Moreover the control strategies require accurate predictions exactly in those cases where traffic is congested, i.e., those cases where the state space explosion is most acute. One way of overcoming this problem is to relax the original model. Macroscopic models of traffic systems disregard the individual vehicles and consider only three real valued variables describing the local behavior (local both in space and in time) of the traffic flow: its density, its average speed, and the flow rate, which is the product of the density and the average speed. Examples of macroscopic traffic models can be found in [4], [5], [6], [7], [8].

Petri nets represent a powerful modeling formalism that has been successfully used in different application domains such as manufacturing and logistics. This paper deals with continuous Petri nets instead of 'classical' discrete Petri nets. Continuous Petri nets are the result of relaxing discrete nets by removing the integrality constraint in the firing of transitions. In contrast to discrete nets, the state of a continuous net is a vector of nonnegative real numbers and the firing of the transitions are real valued flows of material/cars that pass from the input places to the output places. This paper has two main goals:

- Obtain a macroscopic traffic model based on continuous Petri nets.
- Design a control strategy using such a model taking into account the changing traffic conditions.

An interesting feature of the proposed model is that the trade-off between accuracy and simplicity of the model can be easily achieved by modifying the Petri net structure. Moreover, given that road sections are modeled as independent subnets, each subnet being a timed continuous Petri net, the resulting model is highly compositional.

Macroscopic traffic models describe the behavior of a traffic network by interconnecting many road sections, and by describing the traffic variables density, average speed, and flow rate, in this particular road section at a given point in time. This model should represent faithfully the *fundamental traffic diagram* [9] which relates the local flow rate and the car density. To achieve this goal using continuous Petri nets, some time extensions to the existing continuous Petri net paradigm will be proposed. The model for the whole network is obtained by joining together the nets for the individual sections. Traffic lights are modelled by adding discrete places and discrete transitions to the system. Thus, the aggregate model is a hybrid Petri net (see [10] for preliminary results).

The behavior of the traffic system can be modified and controlled through the switching of traffic lights. The control goal that will be considered is to minimize the total delay of vehicles in the system. It is desirable to use a control strategy that is able to minimize the given objective function while taking into account the stochastic fluctuations in the inflow of cars into the system. A reasonable approach for this purpose is to adopt a model predictive control (MPC) policy [11], [12]. In comparison to previous work [7], [13], [14] that uses hybrid Petri nets, the results in the present paper:

The research is supported by a European Community Marie Curie Fellowship, CTS, contract number: HPMT-CT-2001-00278; by the Spanish Ministry of Education and Science (Juan de la Cierva fellowship) and project TIN2007-66523; by the European Community's Seventh Framework Programme under project DISC (Grant Agreement n. INFSO-ICT-224498); and by the Belgian Programme on Inter-University Poles of Attraction initiated by the Belgian State, Prime Minister's Office of Science, Technology and Culture. The scientific responsibility for the results presented here remains with the authors.

Jorge Júlvez is with the Universidad de Zaragoza, Spain. (julvez@unizar.es)

René Boel is with the Universiteit Gent, Belgium. (Rene.Boel@UGent.be)

- allows to approximate the fundamental traffic diagram by means of the network structure (this way, no new firing semantics are necessary),
- and applies MPC to handle varying traffic conditions.

The paper is organized as follows: Section II introduces the continuous Petri net formalism. In Section III some extensions are added to this model to represent more realistically the dynamics of traffic systems. Section IV presents the timed continuous Petri net model for a traffic section. Such a model is the key structure to assemble larger models. In Section V the control problem and the model predictive control strategy are presented. Two control scenarios are reported in Section VI. The main conclusions are drawn in Section VII.

Some comment on the notation: Square brackets are used to access the value of a place or transition in a given vector, e.g., $\mathbf{m}[p]$ denotes the marking of $p \in P$, while, e.g., $C[P,t]$ denotes the column of size $\sharp(P)$ corresponding to transition t. Parenthesis are used to get the value of a variable at a given time, e.g., $\mathbf{m}(\tau)$ is the vector of markings at time τ, and $\mathbf{m}[p](\tau)$ is the marking of place p at time τ.

II. Continuous Petri nets

The reader is assumed to be familiar with Petri nets (PNs) (see [15], [16] for an introduction), a formalism with many domains of application (see [17], [18] for recent references). The Petri net systems that will be considered here are *continuous* [19], [20]. Unlike discrete PN, the marking and the arc weights of the net are non-negative real values, not necessarily integer-valued.

Definition 1: A continuous PN is a tuple $\mathcal{N} = \langle P, T, \mathbf{Pre}, \mathbf{Post} \rangle$ where P and T denote sets of places, resp. transitions, and $\mathbf{Post} \in \mathbb{R}_{0+}^{P\times T}$ and $\mathbf{Pre} \in \mathbb{R}_{0+}^{P\times T}$ are the arc weight matrices.

A continuous PN system is a pair $\langle \mathcal{N}, \mathbf{m_0} \rangle$, where $\mathcal{N}$ specifies the net structure, and $\mathbf{m_0} \in \mathbb{R}_{0+}^{P}$ is the initial marking. The set of input (resp. output) places of a given set V of transitions is denoted as ${}^\bullet V$ (resp. $V^\bullet$). Correspondingly, the set of input (resp. output) transitions of a given set W of places is denoted as ${}^\bullet W$ (resp. $W^\bullet$).

Continuous PNs are obtained as a relaxation of *discrete* ones. Unlike the "usual" discrete PN systems, the amount in which a transition can be fired in a continuous PN is a nonnegative real number. Graphically, a continuous place is represented as a double circle and a continuous transition as a white box.

A transition t in a continuous PN is *enabled* at $\mathbf{m}$ if for every $p \in {}^\bullet t$, $\mathbf{m}[p] > 0$. As in discrete PNs, the *enabling degree* at $\mathbf{m}$ of a transition measures the maximal amount in which the transition can be fired in a single occurrence:

$$\mathrm{enab}(t, \mathbf{m}) = \min_{p \in {}^\bullet t} \left\{ \frac{\mathbf{m}[p]}{\mathbf{Pre}[p,t]} \right\} \tag{1}$$

The firing of t in a certain amount $\alpha \leq \mathrm{enab}(t, \mathbf{m})$ leads to a new marking $\mathbf{m}'$, and it is denoted as $\mathbf{m} \xrightarrow{\alpha t} \mathbf{m}'$. Generalizing the equations for discrete Petri nets $\mathbf{m}' = \mathbf{m} + \mathbf{C}[P,t] \cdot \alpha$. Thus, if $\mathbf{m}$ is the initial marking, the marking $\mathbf{m}'$ reached after several transition firings (with firing count vector $\boldsymbol{\sigma}$, i.e. the sum of the amounts by which each transitions has fired) is given by the fundamental state equation: $\mathbf{m}' = \mathbf{m} + \mathbf{C} \cdot \boldsymbol{\sigma}$.

The evolution of the marking over time can also be expressed in terms of the state equation:

$$\mathbf{m}(\tau) = \mathbf{m_0} + \mathbf{C} \cdot \boldsymbol{\sigma}(\tau) \tag{2}$$

where τ represents time. Differentiating with respect to time $\dot{\mathbf{m}}(\tau) = \mathbf{C} \cdot \dot{\boldsymbol{\sigma}}(\tau)$ is obtained. Let us denote $\mathbf{f} = \dot{\boldsymbol{\sigma}}$, since it represents the *flow* through the transitions. In this paper the flow through transitions is variable (similar to [21]), more specifically, infinite server semantics [22] is used. It will be shown that infinite server semantics allows one to model in a natural way the rising edge of the *fundamental traffic diagram*.

Infinite server semantics is obtained from a first order or deterministic approximation of the discrete case. Thus, the flow through a transition t at instant τ is defined as:

$$\mathbf{f}[t](\tau) = \boldsymbol{\lambda}[t] \cdot \mathrm{enab}(t, \mathbf{m}(\tau)) \tag{3}$$

where $\boldsymbol{\lambda}[t] > 0$ is a constant parameter representing the internal speed of the transition. This way, the flow of a transition is proportional to the marking of the input place determining the enabling degree. The overall behavior of a time continuous PN is similar to that of a piecewise linear system. In PNs a switch between linear dynamics is triggered by a change in the marking of the input place determining the enabling degree of a transition.

III. Timed Petri nets for traffic systems

This section first analyzes the capabilities of continuous Petri nets to model the fundamental traffic diagram representing the behavior of a traffic system. Then, it proposes two modifications to the timed continuous Petri net formalism that are useful for obtaining more realistic, and yet compact, models for traffic systems.

A. Ratio marking vs. flow

Infinite server semantics is used in system models in which the processing speed, i.e., the flow of transitions, is proportional to the number of customers in the upstream place, i.e., proportional to the enabling degree. The following examples show how the flow of transitions and the rate of change of the marking of places can be affected by the arc weights.

Consider transition t_1 (see Figure 1(a)) that has one input place p_1. Its flow is $\mathbf{f}[t_1] = \boldsymbol{\lambda}[t_1] \cdot \mathbf{m}[p_1]/z$ where $z > 0$ is the weight of the arc. As shown in Section II, under infinite server semantics the marking changes according to $\dot{\mathbf{m}}(\tau) = \mathbf{C} \cdot \mathbf{f}$. So, in this case $\dot{\mathbf{m}}[p_1] = -z \cdot \mathbf{f}[t_1] = -\boldsymbol{\lambda}[t_1] \cdot \mathbf{m}[p_1]$. Thus, the evolution of the marking of p_1 does not depend on z, i.e., on the weight of the arc.

By slightly manipulating the system in Figure 1(a), it is possible to obtain a system in which the evolution of p_1 depends on the weight of its input (output) arc. Consider the system in Figure 1(b) with $q > 0$, and $q - a > 0$, since arc weights must be positive. Place p_2 is said to be a self-loop. The flow of transition t_2 is $\mathbf{f}[t_2] = \boldsymbol{\lambda}[t_2] \cdot \mathbf{m}[p_2]/q$, and the marking of p_2 evolves according to $\dot{\mathbf{m}}[p_2] = (q - a - q) \cdot \mathbf{f}[t_2] = -a/q \cdot \boldsymbol{\lambda}[t_2] \cdot \mathbf{m}[p_2]$,

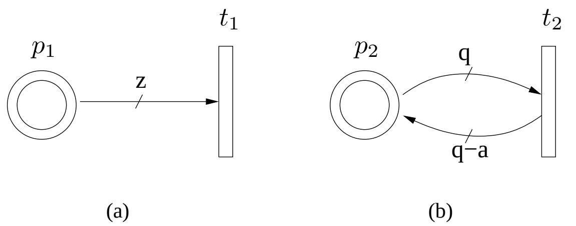


Fig. 1. In a) the marking evolution does not depend on the arc weight. In b) the marking evolution depends on the arc weights.

that is, it depends on the parameter values q and a. If $a > 0$ the marking of p_2 decreases (the condition $q > a$ guarantees that the maximum rate of decrease is bounded by $\dot{\mathbf{m}}[p_2] = -\boldsymbol{\lambda}[t_2] \cdot \mathbf{m}[p_2]$). If $a = 0$ then $\mathbf{m}[p_2]$ is constant and so is the flow of t_2. If $a < 0$ then $\mathbf{m}[p_2]$ increases (the rate of increase is not bounded).

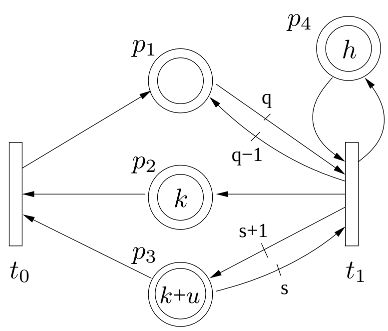


Fig. 2. A continuous Petri net with several self-loops.

Following these ideas, the flow of a transition can be modeled as a piecewise linear function of the marking of a given place. Let us consider the system in Figure 2. Let the internal speed of t_1 be $\lambda[t_1]$, while the initial markings are given by $\mathbf{m_0}[p_1] = 0$, $\mathbf{m_0}[p_2] = k$, $\mathbf{m_0}[p_3] = k + u$ and $\mathbf{m_0}[p_4] = h$ where k, u and h are positive real values. From the net structure, the following marking invariants (or P-semiflows) can be deduced: $\mathbf{m}[p_1] + \mathbf{m}[p_2] = k$, $\mathbf{m}[p_1] + \mathbf{m}[p_3] = k + u$, and $\mathbf{m}[p_4] = h$. The existence of P-semiflows greatly helps to synthesize the Petri net structure and to choose the arc weights that realize a given piecewise linear relationship between the marking $\mathbf{m}[p_1]$ and the flow $\mathbf{f}[t_1]$. Later on, these piecewise linear relationships will be used to approximate the fundamental diagram expressing the relationship between density of cars and flow of cars in a given section of the road.

The thick line in Figure 3 plots the piecewise linear relationship between $\mathbf{f}[t_1]$ and $\mathbf{m}[p_1]$. When the marking of p_1 is smaller than $h \cdot q$ it constrains the firing of t_1, and the flow of t_1 is proportional to $\mathbf{m}[p_1]$. As soon as $\mathbf{m}[p_1]$ satisfies $\mathbf{m}[p_1]/q > h$, the flow of t_1 is constrained by p_4. Given that the $\mathbf{m}[p_4]$ is constant the flow will also remain constant. Assume that $\mathbf{m}[p_1]$ keeps increasing. This fact involves a decrease in $\mathbf{m}[p_2]$ and $\mathbf{m}[p_3]$ since $\mathbf{m}[p_1] + \mathbf{m}[p_2] = k$ and $\mathbf{m}[p_1] + \mathbf{m}[p_3] = k + u$. Given that p_3 is also an input place of t_1, it will constrain the flow of t_1 if $\mathbf{m}[p_3]/s < h$ what is equivalent to $\mathbf{m}[p_1] > k + u - h \cdot s$. Since all markings are positive and $\mathbf{m}[p_1] + \mathbf{m}[p_2] = k$, the maximum value that

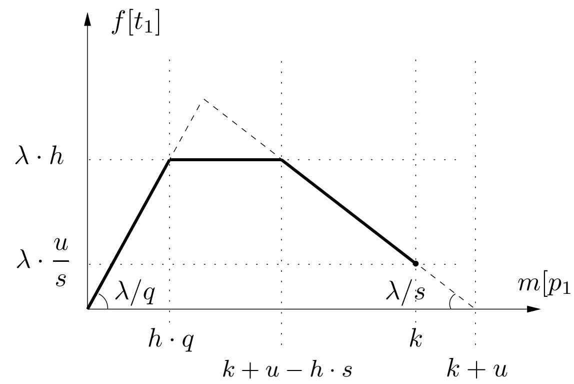


Fig. 3. The flow of transition t_1 (see Figure 2) is a piecewise linear function of the marking of p_1.

$\mathbf{m}[p_1]$ can get is k. Summing up, the flow of t_1 is given by:

$$\mathbf{f}[t_1] = \begin{cases} \lambda[t_1] \cdot \dfrac{\mathbf{m}[p_1]}{q} & \text{if } \mathbf{m}[p_1] < h \cdot q \\ \lambda[t_1] \cdot h & \text{if } h \cdot q \leq \mathbf{m}[p_1] \leq k + u - h \cdot s \\ \lambda[t_1] \cdot \dfrac{k + u - \mathbf{m}[p_1]}{s} & \text{if } k + u - h \cdot s < \mathbf{m}[p_1] \end{cases}$$

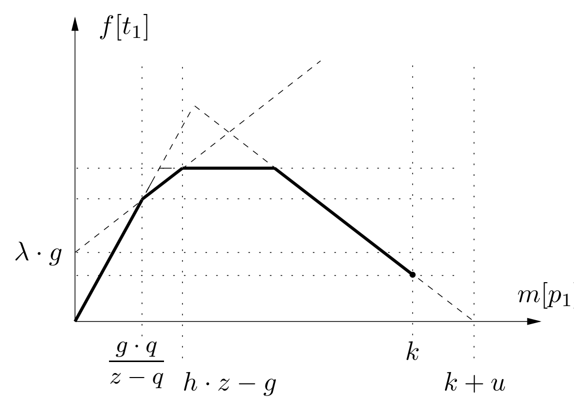


Fig. 4. The addition of a new self-loop can be used to slightly modify the piecewise relationship.

Interestingly, the plot in Figure 3 can be softened by adding more self-loops. For instance, assume that a new place p_5 is added to the net in Figure 2 such that p_5 has the same arcs as p_1 but that arc weight q is substituted by z, and $z > q$. Let $\mathbf{m_0}[p_5] = \mathbf{m_0}[p_1] + g$ be the initial marking of p_5 with $g > 0$. Clearly, at any time it holds $\mathbf{m}[p_5] = \mathbf{m}[p_1] + g$. Given that $z > q$, when $\mathbf{m}[p_1]$ is high enough, the firing of t_1 will be constrained by p_5. The new relationship between $\mathbf{f}[t_1]$ and $\mathbf{m}[p_1]$ is represented by the thick line in Figure 4.

This way, an appropriate choice of self-loop places and arc weights allows one to approximate (arbitrarily closely) any bell-shaped function (recall that traffic diagrams are usually bell-shaped).

B. Discrete time model

The standard continuous Petri net model with infinite server semantics has instantaneous flow of material (or vehicles in the traffic model) from one place, e.g., representing a section in a traffic network, to the next place. Let us consider the system in Figure 5. It represents a machine, t_1, working at constant speed, $\mathbf{f}[t_1] = \boldsymbol{\lambda}[t_1] \cdot \mathbf{m}[p_1]$, that places its production on a conveyor belt represented by p_2. One can imagine that machine t_1 places pieces of finished material at uniformly distributed locations on the conveyor belt p_2; the conveyor belt then moves those pieces to the second machine t_2 which removes the pieces from the conveyor belt. Machine t_2 then processes this input material and stores it in the warehouse p_3. The initial marking of the system is $\mathbf{m_0} = (1\ 0\ 0)$, i.e., the conveyor belt and the warehouse are initially empty.

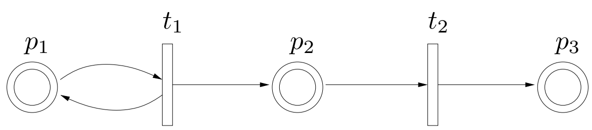


Fig. 5. A continuous Petri net modeling a conveyor.

According to the usual continuous time model the initial flow of t_1 is $\mathbf{f}[t_1](\tau = 0) = \boldsymbol{\lambda}[t_1]$. This implies that material is placed on the conveyor belt p_2 from the initial instant $\tau = 0$ ($\mathbf{m}[p_2](\tau) > 0$ for every $\tau > 0$). This entails $\mathbf{f}[t_2](\tau) > 0$ for every $\tau > 0$. This behavior cannot be a faithful representation of the real system behavior since it implies that an infinitesimal amount of the material spends zero units of time to reach t_2, i.e., the conveyor is infinitely fast (or infinitely short).

One way of avoiding an infinitely fast movement of material going from one transition to the next one is to use a discrete time model (alternatively [19] models this behavior by means of discrete transitions and 0^+ weighted arcs). According to our approach, time is discretized in steps (intervals) of length $\Delta > 0$. At the beginning of each step, the flow of the transitions is computed with the usual expression for infinite server semantics: $\mathbf{f}[t](k) = \boldsymbol{\lambda}[t] \cdot \min_{p \in {}^\bullet t}\{\mathbf{m}[p](k)/\mathbf{Pre}[p,t]\}$ for the k^{th} step. The marking at the next step is defined by $\mathbf{m}(k+1) = \mathbf{m}(k) + \mathbf{C} \cdot \mathbf{f}(k) \cdot \Delta$. This way, the flow of a transition during Δ units of time depends only on the marking of its input places at the beginning of the interval. The interval Δ can be seen as the minimal travelling time (delay) of the material between two transitions. In Figure 5, Δ is the time the conveyor takes to move a piece from t_1 to t_2. Notice that the flow of t_2 is zero during the first interval ($\mathbf{f}[t_2](\tau = 0..\Delta) = 0$). This discrete time continuous PN model makes it possible to represent delays; moreover the fact that the flow of the system is constant during each interval allows one to carry out fast simulations.

C. Maximum time period

In the discrete time model, Δ is a design parameter modeling the minimal time required to travel from the beginning of a section to the end of the section. According to the semantics defined in the previous subsection, the marking changes linearly during an interval. Thus, if Δ is set too high, it might lead to a negative marking. Fortunately, it is possible to compute an upper bound Δ_{max} such that for any $\Delta \leq \Delta_{max}$ the marking, calculated according to the semantics of the discrete time model, is guaranteed to always remain nonnegative. This upper bound depends only on the structure of the net (not on the marking), and can thus be calculated independently of the initial marking. In order to compute Δ_{max}, each place will be considered separately. Without loss of generality one can assume that no input flow is coming into the place (since input flow is always positive and can only make the marking larger). For each place it will be calculated how fast it can become empty, given its maximal outflow rate. If p is the place of the net that can become empty in the shortest time, let us say after γ units of time, then Δ must be less than or equal to γ to avoid negative markings.

Let us compute how fast the place p_1 of the system in Figure 6(a) can become empty. Clearly, the marking of p_1 decreases iff $r > s$, hence only this case is considered. Let us first compute how long it takes to empty p_1 if $\frac{\mathbf{m}[p_1]}{r} \leq \frac{\mathbf{m}[p_2]}{q}$ ($\mathbf{m}[p_1]$ defines the enabling degree of t_1, i.e., p_1 is the constraining place for t_1). In that case $\mathbf{f}[t_1](k) = \boldsymbol{\lambda}[t_1] \cdot \frac{\mathbf{m}[p_1](k)}{r}$ and

$$\mathbf{m}[p_1](k+1) = \mathbf{m}[p_1](k) + (s - r) \cdot \boldsymbol{\lambda}[t_1] \cdot \frac{\mathbf{m}[p_1](k)}{r} \cdot \delta$$

It follows that $\mathbf{m}[p_1](k+1) = 0$ when $\delta = \dfrac{r}{\boldsymbol{\lambda}[t_1] \cdot (r - s)}$. Notice that in the case that $\frac{\mathbf{m}[p_1]}{r} > \frac{\mathbf{m}[p_2]}{q}$ ($\mathbf{m}[p_2]$ defines the enabling degree of t_1) the flow through t_1 would be less than in the previous case and therefore it would take longer to empty p_1. Thus, for the system in Figure 6(a), place p_1 cannot get empty (whatever the marking $\mathbf{m}[p_1](k)$ is) in less time units than:

$$\frac{r}{\boldsymbol{\lambda}[t_1] \cdot (r - s)} \tag{4}$$

Selecting a value of Δ smaller than the value given by (4) prevents p_1 from becoming negative.

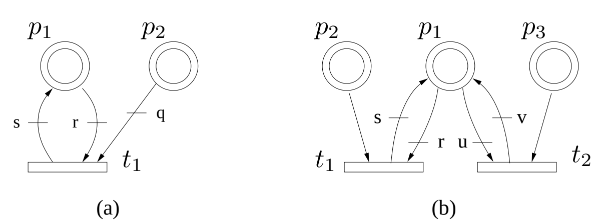


Fig. 6. The bound of Δ that ensures non-negative markings does not depend on the marking.

A similar approach can be taken to compute a bound Δ_{max} for a system having places with several output transitions (see Figure 6(b)). As in the previous example, in order to compute the shortest emptying time of p_1 only the output transitions that decrease the marking are considered, i.e., t_1 (resp. t_2) is considered iff $r > s$ (resp. $u > v$). Similarly to the previous example, the shortest emptying time occurs when p_1 is determining the flow of both output transitions, that is, $\frac{\mathbf{m}[p_1](k)}{r} \leq \mathbf{m}[p_2](k)$ and $\frac{\mathbf{m}[p_1](k)}{u} \leq \mathbf{m}[p_3](k)$. So, if we assume that these inequalities hold the marking in the next step is:

$$\mathbf{m}[p_1](k+1) =$$

$$\mathbf{m}[p_1](k) + ((s-r)\cdot \mathbf{f}[t_1](k) + (v-u)\cdot \mathbf{f}[t_2](k))\cdot \delta =$$

$$\mathbf{m}[p_1](k) + \left((s-r)\cdot \boldsymbol{\lambda}[t_1]\cdot \frac{\mathbf{m}[p_1](k)}{r} + (v-u)\cdot \boldsymbol{\lambda}[t_2]\cdot \frac{\mathbf{m}[p_1](k)}{u}\right)\cdot \delta \tag{5}$$

Then, the time interval δ required to empty p_1, i.e., $\mathbf{m}[p_1](k+1) = 0$, is:

$$\frac{1}{\dfrac{\boldsymbol{\lambda}[t_1]\cdot(r-s)}{r} + \dfrac{\boldsymbol{\lambda}[t_2]\cdot(u-v)}{u}} \tag{6}$$

Equation (6) can be generalized in order to compute Δ_{max} for more complex systems. It suffices to apply (6) for every place of the system, and to compute the minimum of all these values. This Δ_{max} for a general system can be expressed as:

$$\Delta_{max} = \min_{p,\exists t\in\ p^\bullet, \mathbf{Pre}[p,t]>\mathbf{Post}[p,t]} \left\{\frac{1}{g[p,t]}\right\} \tag{7}$$

where $g[p,t]$ is given by:

$$g[p,t] = \sum_{t\in\ p^\bullet,\ \mathbf{Pre}[p,t]>\mathbf{Post}[p,t]} \frac{\boldsymbol{\lambda}[t]\cdot(\mathbf{Pre}[p,t]-\mathbf{Post}[p,t])}{\mathbf{Pre}[p,t]}$$

Notice that this expression depends only on the structure of the net and not on the marking.

D. Emptying places

Let us consider the discrete time evolution of the system in Figure 7. Let Δ be the length of the time interval of the discrete time model (according to the previous Subsection, $\Delta \leq \min\{\frac{1}{\boldsymbol{\lambda}[t_1]}, \frac{1}{\boldsymbol{\lambda}[t_2]}\}$). After the first time step of size Δ, the marking of p_1 is

$$\mathbf{m}[p_1](1) = \mathbf{m}[p_1](0) + \mathbf{C}\cdot \mathbf{f}[t_1](0)\cdot \Delta =$$
$$\mathbf{m}[p_1](0) - \boldsymbol{\lambda}[t_1]\cdot \mathbf{m}[p_1](0)\cdot \Delta = (1-\boldsymbol{\lambda}[t_1]\cdot\Delta)\cdot \mathbf{m}[p_1](0)$$

After the second time step

$$\mathbf{m}[p_1](2) = (1-\boldsymbol{\lambda}[t_1]\cdot\Delta)\cdot \mathbf{m}[p_1](1) =$$
$$(1-\boldsymbol{\lambda}[t_1]\cdot\Delta)^2\cdot \mathbf{m}[p_1](0)$$

and after the k^{th} time step

$$\mathbf{m}[p_1](k) = (1-\boldsymbol{\lambda}[t_1]\cdot\Delta)^k\cdot \mathbf{m}[p_1](0)$$

This way, if $\Delta = \frac{1}{\boldsymbol{\lambda}[t_1]}$, p_1 becomes empty after the first step and remains empty indefinitely. However, if $\Delta < \frac{1}{\boldsymbol{\lambda}[t_1]}$ the evolution of $\mathbf{m}[p_1]$ follows a geometric progression and never gets completely empty.

From a modeling point of view the emptying of a place at a geometric rate can be useful, for example, in order to model how a capacitor discharges exponentially. Nevertheless, for other modeling purposes this feature is not desirable. Suppose that the marking of p_1 is the number of pieces in a conveyor. Then, the flow of t_1 expresses the number of pieces leaving the conveyor per unit of time. If from a given instant no new pieces enter the conveyor, the flow of t_1 should remain constant until the conveyor empties. It would not be realistic that the flow of t_1 decreases exponentially as the conveyor empties.

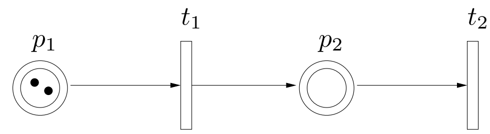


Fig. 7. Place p_1 is emptied in finite time iff $\Delta = 1/\boldsymbol{\lambda}[t_1]$.

By slightly modifying the described firing semantics it is possible to avoid falling in a geometric progression when emptying a place: For a given transition t and at a given step k it will be checked whether its input place determining the enabling degree had input flow (new customers) during the previous step $k-1$. If there was no input flow to that place the flow of t is kept the same, $\mathbf{f}[t](k) = \mathbf{f}[t](k-1)$, otherwise the usual firing semantics is applied, $\mathbf{f}[t](k) = \boldsymbol{\lambda}[t]\cdot \mathrm{enab}(t, \mathbf{m}, k)$. Keeping the same flow of a transition allows one to empty a place in finite time. Anyway, one should be aware that this modification in the model leads to *non pure* discrete time infinite server semantics and could cause negative markings even if the bound for Δ is considered. In order to avoid negative markings, the flow of the transitions will be forced to be the minimum between the value just described and the flow that would empty one of the input places at the end of the time interval. This way, places become empty exactly at the end of time intervals.

Observe that many properties that make Petri nets so useful for modeling remain valid after the modifications introduced in this section. For example our discrete time continuous PNs will satisfy place invariants and transition invariants, markings are still states for the dynamic evolution, structural analysis is applicable, etc.

IV. A MODEL FOR TRAFFIC SYSTEMS

This section proposes a model for traffic systems based on the concepts presented in the previous sections. The model assumes that the road can be virtually divided into several road sections. In Subsection IV-A a continuous PN model for one single road section is presented. Subsection IV-B uses this model of a single road section as a building block for assembling large traffic networks. Traffic lights are modeled in Subsection IV-C as discrete places and discrete transitions connected to the continuous PN model.

A. A road section

The traffic model to be presented requires a spatial discretization of the road to be modelled, i.e., the road is divided into several sections. In this subsection, a continuous PN model for one single road section is presented.

The state of a section of a road network is described by three macroscopic variables: the density $d(\tau)$ of cars at time τ, their average speed $v(\tau)$ and the flow $f(\tau)$. The marking $m(\tau)$ of a place will represent the number of cars in the section, these cars being uniformly distributed along the length of the section, and having average speed $v(\tau)$. Note that $m(\tau)$ is proportional to the density $d(\tau)$ of cars along the section. The flow $f(\tau)$ of cars leaving the section is then $f(\tau) = d(\tau)\cdot v(\tau)$.

In a traffic system the cars in a section with low density travel at a given free speed, this is called *free flow traffic*. In this case the flow out of the section increases proportionally to the density. When the density of the section is higher, the average speed decreases and the flow out of the section *ideally remains constant*. If the density is much higher, the traffic becomes heavy and the flow out of the section decreases due to congestion. This (bell shaped) relationship between the flow and the density is known as the *fundamental traffic diagram* [9]. In the proposed model the fundamental traffic diagram will be approximated by a piecewise linear function following the ideas in Subsection III-A. First, a net that models free flow traffic and constant flow traffic is presented. Later on, it will be shown how the decrease in flow due to congestion is modeled when two sections are joined.

Figure 8(a) is the first step to model a given road section i. The number of cars in section i is represented by the marking of place p_1^i, the flow of cars leaving the section is the flow of transition t_i, and the flow of cars entering the section is the flow of transition t_{i-1}. If p_3^i is ignored, the use of infinite server semantics establishes $\mathbf{f}[t_i] = \boldsymbol{\lambda}[t_i] \cdot \mathbf{m}[p_1^i]$, i.e., the outflow is proportional to the density. Hence, the subnet p_1^i, t_i with an appropriate $\boldsymbol{\lambda}[t_i]$ models free flow traffic. Notice that this relationship between the flow and the marking, $\mathbf{f}[t_i] = \boldsymbol{\lambda}[t_i] \cdot \mathbf{m}[p_1^i]$, cannot be represented with finite server semantics where the flow of a transition is independent of the marking of its positively marked input places [23].

For simplicity, it will be assumed that the system mode changes from free flow to constant flow traffic without intermediate modes. Nonetheless, for a better approximation of the fundamental diagram, such intermediate modes can be easily modeled by adding more self-loop places (as in Subsection III-A). Constant flow traffic can be modeled by adding p_3^i. The marking of p_3^i is always constant and imposes an upper bound on the flow of t_i, $\mathbf{f}[t_i] = \boldsymbol{\lambda}[t_i] \cdot \min\{\mathbf{m}[p_1^i], \mathbf{m}[p_3^i]\}$. Therefore, when $\mathbf{m}[p_1^i] > \mathbf{m}[p_3^i]$ the flow of t_i is constant, $\mathbf{f}[t_i] = \boldsymbol{\lambda}[t_i] \cdot \mathbf{m}[p_3^i] = \boldsymbol{\lambda}[t_i] \cdot h^i$.

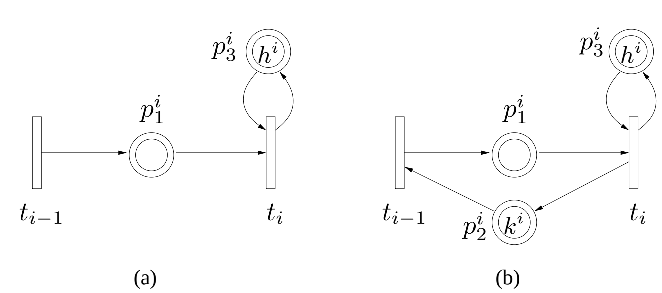


Fig. 8. Petri net model of a road section.

Obviously, the number of cars that can be in a road section is finite. This means that the model of a section must impose an upper bound on the marking of the place representing the number of cars. This can be easily achieved by adding a new place to the section model, viz. p_2^i in Figure 8(b). At all times the model in Figure 8(b) ensures that $\mathbf{m}[p_1^i] + \mathbf{m}[p_2^i] = k^i$ where k^i represents the capacity of the section and $\mathbf{m}[p_2^i]$ represents the number of free gaps in the section.

The model of a road section as proposed above describes the behavior of the system before the onset of congestion. Subsection IV-B shows that the behavior of a congested section is captured in a natural way as a result of the interaction with downstream sections.

B. *Joining sections*

In a PN model with several sections, two adjacent sections, i, j, share a transition, t_i, whose flow represents the flow rate of cars passing the boundary between section i and section j (measured in cars per time unit). This transition t_i has three input places: p_1^i representing the number of cars in section i, p_3^i with constant marking bounding the flow of t_i and p_2^{i+1} representing the number of gaps in section j. Therefore, the flow of cars from section i to section $i+1$ also depends on the number of gaps in the downstream section $i+1$, $\mathbf{f}[t_i] = \boldsymbol{\lambda}[t_i]\cdot\min\{\mathbf{m}[p_1^i], \mathbf{m}[p_3^i], \mathbf{m}[p_2^{i+1}]\}$. This model closely represents the physical reality of the upstream propagation of a traffic jam. Indeed, if not enough free gaps are available downstream, i.e. if the downstream section is congested, then the outflow from the upstream section will decrease. The outflow from t_i is thus, proportional to the minimum of the number of cars desiring to leave the upstream section, i.e., the number of tokens in the upstream place p_i^1, and the number of cars allowed to enter the downstream section, i.e., the number of tokens in downstream place p_2^{i+1}. This is analogous to the sending and receiving functions described in [24].

The outflow from a low density section i (a section in free flow condition with $d_i(\tau) \leq h \cdot q$) is proportional to the number of cars ($\mathbf{f}[t_i] = \boldsymbol{\lambda}[t_i] \cdot \min\{\mathbf{m}[p_1^i]\}$) with proportionality constant $\boldsymbol{\lambda}[t_i]$. If the downstream section becomes full, the outflow is proportional to the number of gaps of the downstream section ($\mathbf{f}[t_i] = \boldsymbol{\lambda}[t_i] \cdot \min\{\mathbf{m}[p_2^j]\}$), with $\boldsymbol{\lambda}[t_i]$ as the proportionality constant. This model implies that the proportionality constant $\boldsymbol{\lambda}[t_i]$ takes the same value under both situations. This is not in agreement with real traffic data. One way to avoid this fact is to use arc loops as shown in Figure 1. The use of such arc loops allows one to have different proportionality constants for the density of cars and the number of gaps. Notice that the constant flow traffic is modeled thanks to a place, p_3^i, with a constant marking. Hence, for any $\boldsymbol{\lambda}[t_i]$ its marking can be chosen to correctly upper bound the flow of t_i without introducing weights in its input/output arcs.

Figure 9 shows a traffic model consisting of three sections with arc loops controlling the proportionality constants. With an appropriate $\boldsymbol{\lambda}$, that system can be reduced to an equivalent one with only one arc loop for each transition (since $\boldsymbol{\lambda}[t_i]$ is already the proportionality constant either for the density or for the number of gaps).

Notice that the special features of the model described previously are useful for traffic modeling. By using a discrete time model it is possible to represent the minimal delay of the cars coming from the input transition of a given section to the output transition of the same section. This time interval,

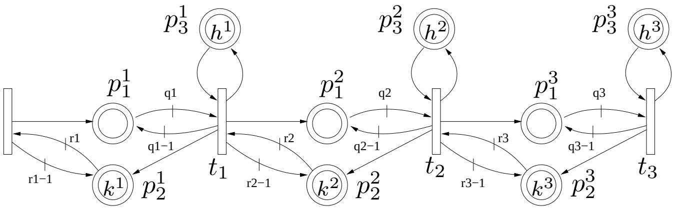
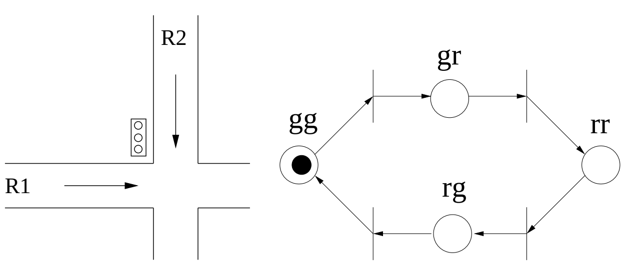


Fig. 9. Traffic system with three sections.

Δ, can be seen as the minimal time required for a car to travel from the beginning of a section to the beginning of the next one. A continuous time model could be possible, but then it would require an infinite dimensional state space, corresponding to infinitely many infinitesimally short sections (a partial differential equation is obtained by letting Δ go to 0). Besides, the extensions presented in this paper allow sections to become empty in finite time by keeping the outflow constant as long as no inflow exists. These extensions represent more faithfully the behavior of a real traffic system than the original continuous PN formalism, and they will be used in the sequel for simulations.

C. *Traffic lights*

Traffic lights are the most common way to control real traffic systems. Traffic lights can be seen as discrete event systems whose state can be either red, amber or green. This is why we propose to model traffic lights with simple discrete Petri nets. See Figure 10 for traffic lights ruling an intersection of two one-way streets $R1$ and $R2$.



(a) (b)

Fig. 10. A discrete Petri net modeling traffic lights in an intersection.

The system that models traffic lights has four phases, each phase being represented by a place in the net. A given phase is active when its corresponding place is marked. Since only one phase can be active at a time, the number of tokens in the net is 1. The actions associated to each phase are:

- *gg*: Cars from $R1$ crossing. Traffic lights state: green lights for $R1$; red lights for $R2$.
- *gr*: Cars from $R1$ stop, cars from $R2$ start crossing. Traffic lights state: first amber, and then red light for $R1$; red, amber and finally green for $R2$.
- *rr*: Cars from $R2$ crossing. Traffic lights state: red lights for $R1$; green lights for $R2$.
- *rg*: Cars from $R2$ stop, cars from $R1$ start crossing. Traffic lights state: red, amber and finally green for $R1$; first amber, and then red light for $R2$.

Figure 11 sketches how the flow of cars coming from $R1$ and $R2$ is regulated by the traffic lights. The flow of cars crossing the intersection from $R1$ ($R2$) at a given time is obtained by multiplying the flow of the continuous transition (see Section III) associated to $R1$ ($R2$) by the value $v(R1)$ ($v(R2)$) at the corresponding time, as plotted in Figure 11. For instance, the flow of transition t_1 in Figure 12 during phase *gg* is the same as if there were no traffic lights since $v(R1) = 1$; during phase *gr* the flow decreases linearly and becomes zero after α time units; all along phase *rr* the flow remains equal to zero; finally, β time units before the end of phase *rg* the flow increases linearly from zero to the value it would take if there were no traffic lights.

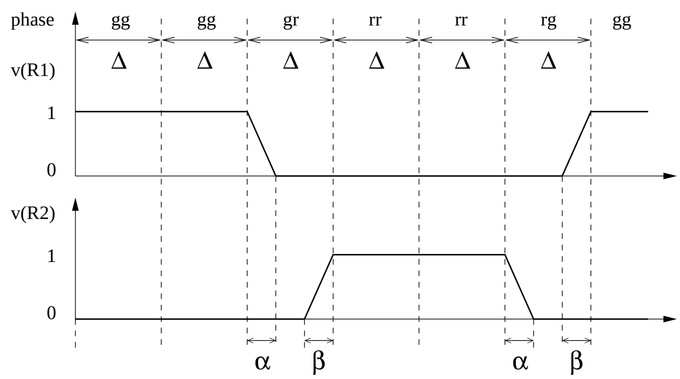


Fig. 11. Scaling factors, $v(R1)$ and $v(R2)$, for the flows of the four traffic lights phases.

Phases *gr* and *rg* allow one to model how the flow of cars evolves smoothly from maximal flow to zero flow and vice versa, and so to obtain a more realistic model of the traffic system. The positive real values α and β are modeling parameters that have to fulfill $\alpha + \beta < \Delta$ (where Δ is the value used in the discrete time model proposed in subsection IV-B). The safety time interval during which no cars cross the intersection is $\Delta - \alpha - \beta$.

V. CONTROL STRATEGY

This section illustrates how the model of road traffic as developed above can be used for designing a model predictive feedback controller, approximately minimizing a given objective function. The first subsections introduce the objective function and the control constraints that have to be considered. Then, the model predictive control scheme is presented.

A. *Objective function*

Many different control goals can be pursued for traffic systems. In this paper we focus on the minimization of the total delay (waiting time) of the cars in the system. In other words the control strategy used by the traffic lights must minimize the sum of the time delays spent by all cars during the control horizon in which the control is applied.

Let us consider that the marking of place p_1^i represents the number of cars in section i. Then, in the continuous time domain the delay of all the cars passing through section i during the time interval from 0 to ρ is given by the integral

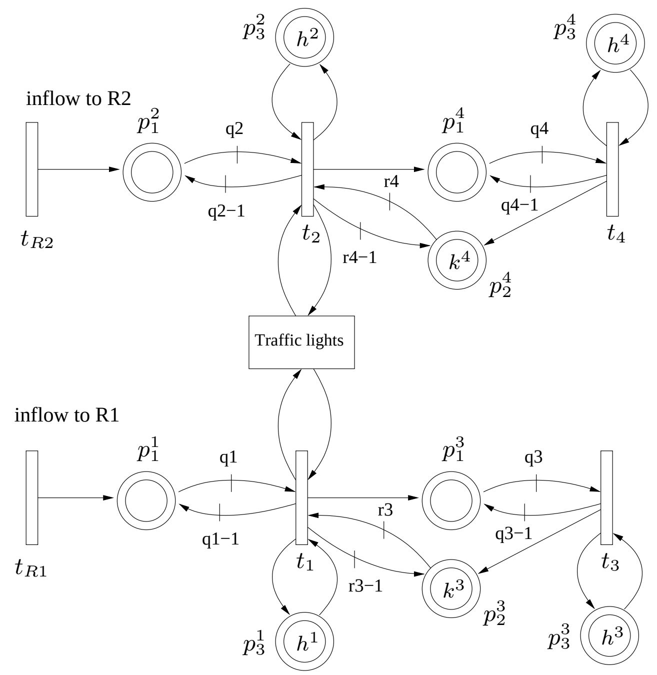


Fig. 12. An intersection modeled by a continuous Petri net.

$\int_0^\rho \mathbf{m}[p_1^i](\tau)d\tau$. The total delay of all the cars passing through the system is obtained by summing over all the places p_1^i representing all the sections of the network:

$$\sum_{p_1^i} \int_0^\rho \mathbf{m}[p_1^i](\tau)d\tau \tag{8}$$

Since infinite server semantics is used, the following equation:

$$\dot{\mathbf{m}}[p_1^i](\xi) = \mathbf{f}[{}^\bullet p_1^i](\xi) - \mathbf{f}[p_1^{i\bullet}](\xi)$$

holds for each place p_1^i, and therefore:

$$\mathbf{m}[p_1^i](\tau) = \mathbf{m}[p_1^i](0) + \int_0^\tau \mathbf{f}[{}^\bullet p_1^i](\xi)d\xi - \int_0^\tau \mathbf{f}[p_1^{i\bullet}](\xi)d\xi$$

Notice that in the traffic model the output transition of a given section is the input transition of the downstream section. Thus, many terms of Equation (8) cancel and so the total delay can be expressed as:

$$\begin{aligned}\sum_{p_1^i} \int_0^\rho \mathbf{m}[p_1^i](0)d\tau + \sum_{t\in T_{in}} \int_0^\rho \int_0^\tau \mathbf{f}[t](\xi)d\xi d\tau \; - \\ \sum_{t\in T_{out}} \int_0^\rho \int_0^\tau \mathbf{f}[t](\xi)d\xi d\tau\end{aligned} \tag{9}$$

where $T_{in}(T_{out})$ stands for the set of input(output) transitions to(from) the system. For example, for the system in Figure 12, $T_{in} = \{t_{R1}, t_{R2}\}$ and $T_{out} = \{t_3, t_4\}$. Clearly the first term of Equation (9) does not depend on the control policy. Let us assume that the incoming flow of cars to the system is modeled by a stochastic function. Thus the second term of Equation (9) does not depend on the control policy either. This way, it turns out that minimizing the total delay is equivalent to maximizing the outflow from the system, given by:

$$\sum_{t\in T_{out}} \int_0^\rho \int_0^\tau \mathbf{f}[t](\xi)d\xi d\tau \tag{10}$$

Hence only the flows of the output transitions of the system are required. Assume for the sake of simplicity that the output transitions of the system are not regulated by traffic lights. Assume that the discrete time domain described in Subsection III-B is implemented by the simulator of the network. Then the flow of any output transition t_i is piecewise constant, all periods being of length Δ. At the end of the period H, where $H = \frac{\rho}{\Delta}$:

$$\int_0^\rho \int_0^\tau \mathbf{f}[t](\xi)d\xi d\tau = \frac{\Delta^2}{2} \sum_{i=1}^{H} \Big(\big(2\cdot(H-i)+1\big) \cdot \mathbf{f}^i[t] \Big) \tag{11}$$

where $\mathbf{f}^i[t]$ is the flow of transition t at the beginning of period i. Since Δ is constant it can be removed from the objective function. The final expression for the objective function is obtained by applying Equation (11) to every output transition of the system and summing the obtained values:

$$\max \sum_{i=1}^{H} \Big(\big(2\cdot(H-i)+1\big) \cdot \sum_{t\in T_{out}} \mathbf{f}^i[t] \Big) \tag{12}$$

B. Control constraints

In order to avoid excessively long waiting times for individual cars, a maximum time interval $Mred$ for red lights will be established at each intersection. When the traffic light has remained red for a given direction for an interval of time equal to $Mred$ then a transition is forced to fire in the PN representing the traffic lights, causing the light to turn green for the given direction. Similarly, the proposed control offers the chance of establishing minimum time intervals $mgreen$ for green lights to avoid having excessively short green light cycles. Notice that establishing maximum red (minimum green) time intervals for a given road crossing an intersection implies establishing maximum green (minimum red) time intervals for the other road crossing the same intersection.

C. Model predictive control

This subsection proposes a model predictive control (MPC) policy [11], [12] for a traffic network modeled by connecting several components of a road network as suggested in Section IV, approximately minimizing the cost function defined in Subsection V-A, and taking the constraints of Subsection V-B. An MPC approach allows one to minimize the given objective function (12). Since an MPC controller acts as a closed-loop structure it generates a robust feedback controller that achieves good performance under a reasonably broad class of perturbations of the model, e.g., uncertainty about the inflow of cars in the system.

Basically, the MPC computes in each iteration the switching sequence of the traffic lights that minimizes the total delay of cars in the system. Then, during one step, i.e., Δ time units, it applies the control action specified by the optimum switching sequence on the traffic lights and continues looping.

The input data of the MPC are: the structure of the model (roads, intersections, traffic lights,...), the initial state of the system (number of cars in each section and state of the traffic lights), the time interval (Δ), the control horizon (H), the maximum red ($Mred$) and minimum green ($mgreen$) intervals allowed for each intersection, and the inflows of cars at the entrance transitions. The following algorithm sketches the MPC structure for the traffic model:

Algorithm 1:

Input: System structure, $Initial\_state$, Δ, H, $Mred$, $mgreen$, $Inflow\_of\_cars$

1) $Current\_state$:= $Initial\_state$
2) *loop*
3) *Compute the potential switching sequences of traffic lights over control horizon 'H' satisfying the '*$Mred$*' and '*$mgreen$*' constraints*
4) *Take the switching sequence '*s*' that minimizes the total delay of the system from* $Current\_state$ *over period 'h'*
5) *Apply the first control action specified by '*s*' on the traffic lights*
6) *Get the '*New_state*' of the system after* Δ *time units*
7) $Current\_state$:= New_state
8) *end loop*

The commands in steps 3 and 4 compute the control action for the next step according to the current state. The commands in steps 5, 6 and 7 apply the computed control action during one time period and update the system state. Given that the number of switching sequences is exponential with respect to the number of traffic lights, the computation time of step 4 might become too high if one must check every single sequence to find the optimal one. Fortunately, only the sequences satisfying the '$Mred$' and '$mgreen$' constraints must be checked. The optimal sequence is obtained by simulation: after the simulation of each feasible sequence, the one yielding the minimum total delay is selected.

VI. Control Scenarios

This section shows two traffic scenarios modeled by timed continuous Petri nets and controlled by the MPC feedback controller proposed in Subsection V-C.

A. An intersection

This traffic scenario, modeled as in Figure 12 (see Figure 13 for a sketch), consists of two one-way streets $R1$ and $R2$ that cross at an intersection. Each road $R1$ and $R2$ is divided into two sections: $R1$ consisting of $S1$ and $S3$, and $R2$ consisting of $S2$ and $S4$. Traffic lights regulate the flow of cars at the intersection, i.e., at the end of sections $S1$ and $S2$. For the sake of simplicity all sections are assumed to have the same parameters. Each section has two lanes with a total capacity of 60 cars. The model parameters are the following: $q_1 = q_2 = q_3 = q_4 = 100$, $r_3 = r_4 = 80$, $\boldsymbol{\lambda}[t_1] = \boldsymbol{\lambda}[t_2] = \boldsymbol{\lambda}[t_5] = \boldsymbol{\lambda}[t_6] = 4$, $\boldsymbol{\lambda}[t_3] = \boldsymbol{\lambda}[t_4] = 5$ and $\mathbf{m}[p_3^1] = \mathbf{m}[p_3^2] = \mathbf{m}[p_3^3] = \mathbf{m}[p_3^4] = 0.4$. The initial distribution of cars in the system is: $\mathbf{m_0}[p_1^1] = 15$, $\mathbf{m_0}[p_1^2] = 20$, $\mathbf{m_0}[p_1^3] = 35$, $\mathbf{m_0}[p_1^4] = 15$. Since the capacity of the sections is 60, the initial values of the complementary places are $\mathbf{m_0}[p_2^3] = 25$, $\mathbf{m_0}[p_2^4] = 45$. The flow rate cars/second entering $R1$ (resp. $R2$) is a random variable uniformly distributed in the interval $[0.2, 0.3]$ (resp. $[0.4, 0.6]$).

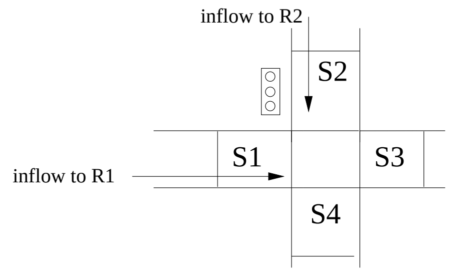


Fig. 13. Sketch of the traffic system in Figure 12.

Time is discretized in periods of 8 seconds, i.e., $\Delta = 8$. The parameters α and β for traffic light phases rg and gr, see Subsection IV-C, are $\alpha = 3$ seconds and $\beta = 2$ seconds. The maximum interval of red lights is specified as $6 \cdot \Delta = 48$ seconds.

The goal of the MPC is to minimize the total delay of cars in the system. The control horizon is 6 periods, i.e., 48 seconds, which is sufficient for a car to cross the whole system provided there are no traffic jams. Figure 14 shows the evolution of the number of cars in each section under MPC, where m1(m2, m3, m4) stands for $\mathbf{m}[p_1^1]$($\mathbf{m}[p_1^2]$, $\mathbf{m}[p_1^3]$, $\mathbf{m}[p_1^4]$), i.e., the number of cars in S1(S2, S3, S4). Green lights for $R1$, so red lights for $R2$, are represented by stars at 1. Red lights for $R1$, so green lights for $R2$, are represented by stars at 3. Stars at 2 represent switching from red to green and vice versa. Since the input flow to $R2$ is greater than the input flow to $R1$, the result of the MPC is that green lights for $R2$ last longer than for $R1$. Given that the incoming flow to $R2$ is stochastic, the rate green/red of the traffic lights is not constant.

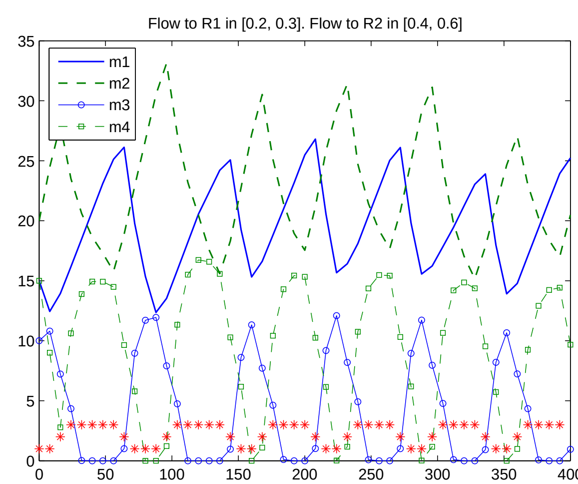


Fig. 14. Evolution of the system in Figure 12 under MPC.

Figure 15 presents the evolution under a "blind" (non MPC) control that disregards the state of the system and that simply applies a constant switching interval to the traffic lights: 32 seconds for red lights and 32 seconds for green lights (the value 32 seconds was chosen after some experimentation as optimal for an open loop control of the traffic lights). The result is a more congested traffic than with MPC control. In particular, section 2 starts to saturate due to its high incoming flow.

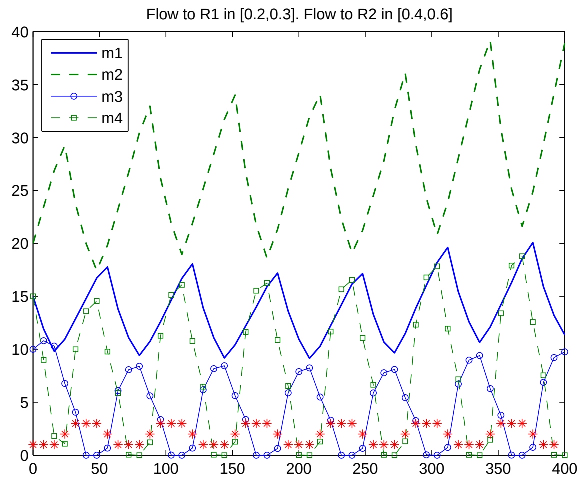


Fig. 15. Evolution of the system in Figure 12 under "blind" control.

Let us illustrate how the MPC "reacts" when the traffic conditions change. Assume that at time $\tau = 200$ the flow of cars entering $R2$ changes from a random variable in $[0.4, 0.6]$ to a constant rate of 0.3 cars per second. This reduction in flow can be due to traffic works, accidents, etc. Figure 16 shows how the MPC automatically adjusts the green ratio after the flow change.

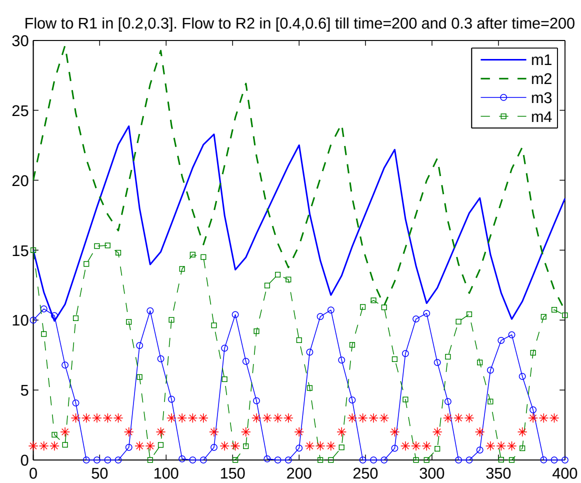


Fig. 16. MPC control of the system in Figure 12 after a flow change.

B. Main road and several intersections

Let us now consider the traffic scenario depicted in Figure 17. It consists of a main road $R1$ that is crossed by three roads $R2$, $R3$ and $R4$ with traffic moving only in the direction indicated by the arrows. Each intersection is regulated by traffic lights: intersection $R1R2$ by traffic lights 1(tl1), intersection $R1R3$ by traffic lights 2(tl2), and intersection $R1R4$ by traffic lights 3(tl3). Road $R1$ is composed of sections $S11$, $S12$, $S13$, $S14$, $S15$ and $S16$; road $R2$ is composed of sections $S21$ and $S22$; road $R3$ is composed of sections $S31$ and $S32$; and road $R4$ is composed of sections $S41$ and $S42$.

The sections in $R1$ have 3 lanes and capacity for 90 cars. The $\boldsymbol{\lambda}$ associated to the transitions in $R1$ is equal to 5 and the marking of the place in the self-loop (like m_3^i in the previous scenario) is equal to 0.45. The sections in roads $R2$, $R3$ and $R4$ have two lanes and a capacity of 60 cars, the associated $\boldsymbol{\lambda}$ is equal to 4 and the marking of the places in the self-loops is equal to 0.4. The initial car loads of sections $S11$, $S12$, $S13$, $S14$, $S15$, $S16$, $S21$, $S22$, $S31$, $S32$, $S41$ and $S42$ are 20, 40, 45, 50, 20, 35, 20, 25, 20, 35, 40 and 35 respectively. The weights of the arcs are $q = 100$ and $r = 80$ for all sections. It is assumed that the incoming flow of cars varies stochastically. For $R1$ the incoming flow yields in the interval $[0.4, 0.7]$ cars/second, for $R2$ in the interval $[0.2, 0.5]$, for $R3$ in $[0.2, 0.4]$, and for $R4$ in $[0.3, 0.5]$.

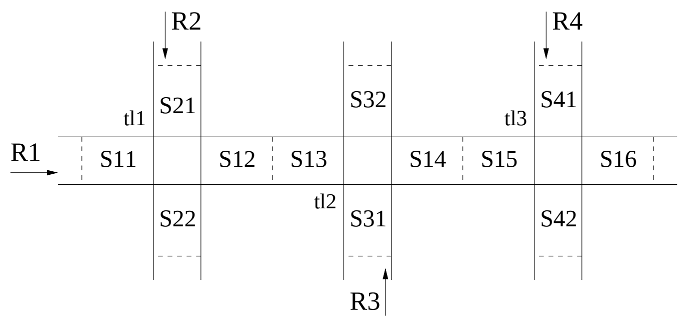


Fig. 17. A main road crossed by three roads.

Time has been discretized in periods of $\Delta = 8$ seconds. The values α and β for soft switching from red to green are $\alpha = 3$ and $\beta = 2$ seconds. The MPC algorithm 1 has been applied to this traffic scenario. The goal of the control is minimizing the total delay of cars. As detailed in Subsection V-A, this is equivalent to maximizing a function (12) that depends on the flow of the output transitions of the system. The output transitions of the system in Figure 17 are the ones corresponding to sections $S16$, $S22$, $S32$ and $S41$. Notice, however, that if only those output transitions are considered the cars entering $R1$ are not in a fair situation: They have to cross 6 sections to leave the system while the rest of the cars only have to cross 2 sections ($R2$, $R3$ and $R4$ have been modeled just by 2 sections). This way, a controller that considers only $S16$, $S22$, $S32$ and $S41$ as output sections will give less priority to $R1$ at intersections $R1R2$ and $R1R3$ (given that it takes longer to flush out the cars in $R1$). An easy way to make the situation fair is to consider the transitions after the intersections as output transitions, i.e., transition between $S12$ and $S13$, and transition between $S14$ and $S15$ are taken as output transitions.

Figure 18 shows the result of applying the MPC scheme during 50 periods (400 seconds). The control horizon for the

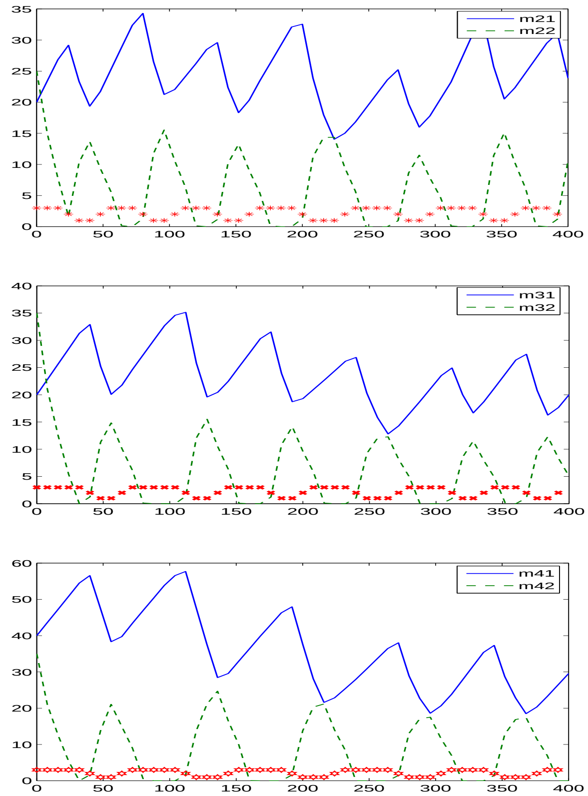


Fig. 18. Marking evolution of sections S21 and S22 (upper), S31 and S32 (middle), S41 and S42 (lower).

MPC was 6 periods, i.e., 48 seconds. Maximum intervals of 40 seconds for red lights and minimum intervals of 16 seconds for green lights were established. Figure 18 upper (middle, lower) shows the evolution of the number of cars in sections $S21$ and $S22$($S31$ and $S32$, $S41$ and $S42$) as well as the state of the traffic lights regulating the intersection $R1R2$($R1R3$, $R1R4$): 1 means green lights for $R2$($R3$,$R4$), 3 means red lights for $R2$($R3$,$R4$) and 2 means traffic lights switching. Given that $R1$ is more loaded than the other roads and its incoming flow is higher, the controller gives priority to $R1$. The rate green/red in each intersection adapts dynamically to the stochastic changes in the incoming flows. This control scenario was run under Matlab 6.5 on a Pentium Centrino 1.5 GHz. The step computation time was 1.1 seconds, implying that a real time application of this strategy may be feasible for traffic systems of reasonable size.

VII. Conclusions

A dynamical model based on continuous Petri nets has been introduced to model the macroscopic behavior of traffic systems. In such a model, the marking of a place represents the number of cars in a given section, and the firing speed of its output transitions stands for the flow of cars leaving that section. By properly selecting the weights of the arcs of the Petri net one can adjust the flow of cars to approximate a given traffic diagram. Some of the main advantages of a continuous Petri net model are: a) it enjoys all structural properties of classical Petri nets; b) it can approximate arbitrarily well a traffic diagram; c) it is highly compositional, i.e., the different parts of the system can be designed separately, and then assembled together easily.

The described traffic model provides a basis to apply a control strategy on traffic systems. Given that the traffic conditions in a traffic road may vary rapidly, a model predictive control approach is a good choice to handle such changes. It has been shown how this control approach can be used to minimize the total delay of cars in the traffic network.

References

[1] J.L. Gallego, J.L. Farges, and J.J. Henry. Design by Petri nets of an intersection signal controller. In *Transportation Research Part C: Emerging Technologies*, volume 4, pages 231–248, 1996.

[2] A. Tzes, Seongho Kim, and W.R. McShane. Applications of Petri networks to transportation network modeling. *Vehicular Technology, IEEE Transactions on*, 45(2):391–400, May 1996.

[3] M. Dotoli and M. P. Fanti. An urban traffic network model via coloured timed Petri nets. *Control Engineering Practice*, 14(10):1213 – 1229, 2006.

[4] S. Hoogendoorn and P. Bovy. State-of-the-art of Vehicular Traffic Flow Modelling. *Special Issue on Road Traffic Modelling and Control of the Journal of Systems and Control Eng. Proc. of the IME I*, 2001.

[5] A. Kotsialos, M. Papageorgiou, C. Diakaki, Y. Pavis, and F. Middelham. Traffic Flow Modelling of Large-Scale Motorway Using the Macroscopic Modeling Tool METANET. *IEEE Transactions on Intelligent Transportation Systems*, 3(4):282–292, 2002.

[6] D. Helbing. Traffic Data and Their Implications for Consistent Traffic Flow Modelling. In M. Papageorgiou and A. Pouliezos, editors, *Transportation Systems (IFAC, Chania, Greece)*, volume 2, pages 809–814, 1997.

[7] Di Febbraro A., Giglio D., and Sacco N. Urban Traffic Control Structure Based on Hybrid Petri Nets. *IEEE Transactions on Intelligent Transportation Systems*, 5 (4):224– 237, 2004.

[8] T. Kato, Y. Kim, S. Okuma, and T. Narikiyo. Large-scale traffic network control based on mixed integer non-linear system formulation. In *IEEE Industrial Electronics, IECON 2006 - 32nd Annual Conference on*, 2006.

[9] M. Papageorgiou. *Applications of Automatic Control Concepts to Traffic Flow Modeling and Control*. Berlin; New York: Springer-Verlag, 1983.

[10] J. Júlvez and R. Boel. Modelling and controlling traffic behaviour with continuous Petri nets. In 16^{th} *triennial world congress of the International Federation of Automatic Control (IFAC)*, 2005.

[11] J.M.Maciejowski. *Predictive control with constraints*. Prentice Hall, 2001.

[12] S. Joe Qin and Thomas A. Badgwell. A survey of industrial model predictive control technology. *Control Engineering Practice*, 11 (7):733–764, 2003.

[13] M. Dotoli, M.P. Fanti, and G. Iacobellis. An Urban Traffic Network Model by First Order Hybrid Petri Nets. In *IEEE International Conference on Systems, Man and Cybernetics (SMC)*, Singapore, October 2008.

[14] YoungWoo Kim, Tatsuya Kato, Shigeru Okuma, and T. Narikiyo. Traffic network control based on hybrid dynamical system modeling and mixed integer nonlinear programming with convexity analysis. *IEEE Transactions on Systems, Man, and Cybernetics, Part A*, 38(2):346–357, 2008.

[15] M. Silva. Introducing Petri Nets. In *Practice of Petri Nets in Manufacturing*, pages 1–62. Chapman & Hall, 1993.

[16] T. Murata. Petri Nets: Properties, Analysis and Applications. *Proceedings of the IEEE*, 77(4):541–580, 1989.

[17] A.L. Feller, T. Wu, D.L. Shunk, and J. Fowler. Petri net translation patterns for the analysis of ebusiness collaboration messaging protocols. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 39(5):1022–1034, Sept. 2009.

[18] Jiliang Luo, Weimin Wu, Hongye Su, and Jian Chu. Supervisor synthesis for enforcing a class of generalized mutual exclusion constraints on Petri nets. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 39(6):1237–1246, Nov. 2009.

[19] R. David and H. Alla. *Discrete, Continuous, and Hybrid Petri Nets*. Springer, 2004.

[20] M. Silva and L. Recalde. On fluidification of Petri net models: from discrete to hybrid and continuous models. *Annual Reviews in Control*, 28:253–266, 2004.

[21] P. Thomas C. Tolba, D. Lefebvre and A. El Moudni. Continuous and timed Petri nets for the macroscopic and microscopic traffic flow modelling. *Simulation Modelling Practice and Theory*, 13 (5):407–436, 2005.

[22] L. Recalde and M. Silva. Petri Nets Fluidification revisited: Semantics and Steady state. *APII-JESA*, 35(4):435–449, 2001.

[23] F. Balduzzi, A. Giua, and G. Menga. First-Order Hybrid Petri Nets: a Model for Optimization and Control. *IEEE Trans. on Robotics and Automation*, 16(4):382–399, 2000.

[24] C. Daganzo. A Finite Difference Approximation of the Kinematic Wave Model of Traffic Flow. *Transportation Research B*, 29B(4):261–276, 1995.