Scenario analysis of a network of traffic signals designed with Petri nets

M. dos Santos Soares & J. Vrancken
Faculty of Technology, Policy and Management,
Delft University of Technology, Delft, The Netherlands

Abstract

The Petri net formalism is a powerful tool to model and analyze discrete event systems. The dynamic behavior of traffic signals is a discrete model and is one of the most important and effective methods of controlling traffic at intersections. In this paper, Petri nets are applied to model a network of intersections and the formal proof is based on the sequent calculus of Linear Logic. The approach can prove that unsafe states are not reached, and that desirable states are reached. This is done using the equivalence between Petri nets reachability and the proof of a set of sequents in Linear Logic. The approach is illustrated by an example of a network of intersections controlled by traffic signals.

1 Introduction

Traffic control in urban roads is a major area in which Intelligent Transportation Systems approaches can be applied. Traffic signals are one of the main approaches to control an intersection. They act regulating, warning and guiding transportation with the purpose of improving safety and efficiency for pedestrians and vehicles. Among the main advantages of traffic signals are the flexibility of the signaling scheme, the ability to provide priority treatment and the feasibility of coordinated control along streets. But when not well-designed, traffic signals may lead to excessive delays when cycle lengths are too long and increase the frequency of collisions.

Petri nets [1] are a common tool applied to design, simulate and analyze discrete event systems, such as traffic signals. Some of the advantages are the graphical representation, the possibility of analyzing properties and the ability to represent aspects such as concurrency, synchronization and shared resources.

There are several examples in the literature of the application of Petri nets to design and analyze traffic signals. In [2], it is described an approach via programmable logic controller and Petri nets to control urban traffic signals. In [3], the purpose is to evaluate the performance of different traffic signals by means of Petri nets. In [4], continuous Petri nets are applied to obtain realistic and compact models for traffic systems. In [5], Petri nets are used to model the control of signalized intersections, and the good properties of the system are evaluated by means of invariant analysis and simulation. In [6], timed Petri nets are applied with the aim of minimizing congestion situations via a traffic-responsive signaling control procedure.

In this paper, a subnetwork of traffic signals is modeled by Petri nets with time associated to transitions. From the model, scenarios are extracted and analyzed with Linear Logic, which is an analysis tool for Petri nets. The approach allows to use symbolic dates instead of numeric ones. As a result, some important properties are formally proved.

2 Rules for traffic signals design

A road intersection can be defined as the general area where two or more roads join or cross, including the roadway and roadside facilities for traffic management [7]. Traffic signals are an important mechanism applied to solve intersections conflicts and regulate traffic flow. To correctly control intersections, traffic signals design must take care of efficiency and speed, but also safety and security. Figure 1 shows a small network of roads with two intersections: I1 and I2. The main road has a great flow from B to D, and A and C are non-priority roads. There are sensors in the main road (from B to D) to detect the first of a platoon of vehicles after a green for B in intersection I1 and allow a green in intersection I2. This approach is used to try to improve the offset in such a way that as the first vehicle just arrives at the next intersection in the network, the signal controlling of this intersection turns green. This ideal offset is difficult to achieve when pre-determined offsets are defined. The risk is that if the platoon of vehicles goes slower or faster than it is expected, then the result will inevitably be sub-optimal. When considering sensors to detect the first vehicle of a platoon, although risks still exists, it is considerable lower.

Some important rules (non-exhaustive) for the proper functioning of traffic signals are [8]:

- the traffic signal must not allow the green state to two conflicting road sections simultaneously;
- each traffic signal must follow a defined sequence of active color lights, normally from green to yellow and red, and then backing to green;
- the right to use an intersection has to be given to all sections.

When reasoning on a network of intersections, three other rules are important:

- any user in the intersection should not wait for more than a maximum service delay, otherwise the user may presume that the traffic signal is not functioning, which can lead to non-secure decisions by the users, or the formation of big queues.
- the offset is ideally designed in such a way that as the first vehicle just arrives at the next intersection in the network, the signal controlling this intersection turns green.
- the length of green time for each road section can be different, depending on section priority, for instance.

The UML use case in figure 2 shows a context diagram for traffic signals controlling a network of intersections and considering the rules above. The actors Sensor and Traffic Signal are responsible to realize the use case Control Phases. In order to correctly and efficiently control the network, the safety and performance rules must be applied.

Each use case is a set of actions performed by the system, which yields an observable result for the involved actors. The dynamic behavior of the use cases are designed with Petri nets in the approach presented in this paper.

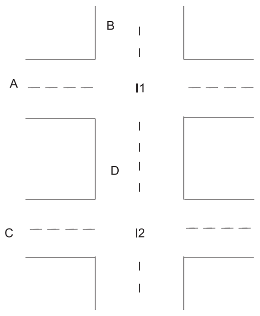

Figure 1: Subnetwork example.

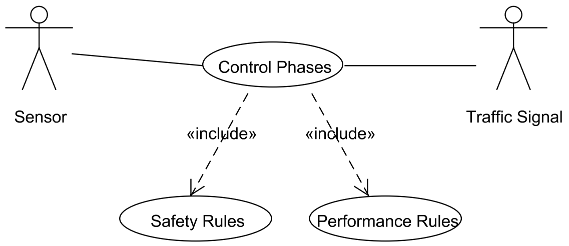

Figure 2: System Traffic Signal Use case.

3 Petri nets modeling

Petri nets [1] are a graphical and mathematical tool applicable to a large variety of systems in which concurrency, dynamic behavior, synchronous and asynchronous communication, and resource sharing have to be modeled. As a graphical tool, with simple but powerful primitives, Petri nets are useful as a visual communication aid. As a mathematical tool, Petri nets can formally prove a set of desirable properties of systems and evaluate performance aspects.

Formally, a Petri net is a 5-tuple $N = (P, T, I, O, M_0)$, where P is a finite set of places; T is a finite set of transitions; I is an input function that defines directed arcs from places to transitions; O is an output function that defines directed arcs from transitions to places; and M_0 is the initial marking. A marking is an assignment of tokens to the places of a Petri net. For the analysis of Petri net models, the concept of marking is fundamental.

There are several advantages for using Petri nets in the design of complex discrete event systems. The graphical representation is composed of few basic elements, very simple to understand. As a formal tool, Petri nets allow to perform a formal check of desirable properties. The formal analysis of a Petri net model can reveal whether the model is reliable or not. For instance, with reachability analysis it is possible to find out whether an unsafe state that could cause an accident can be reached. Petri nets have mechanisms to provide important aspects for the design of large and complex systems, such as abstraction, hierarchical design and modularity. Transitions and places can represent sub-nets, allowing the modeling in several levels of detail. In addition, it is feasible to construct large models relating smaller Petri nets to each other. For instance, the fusion of common places, representing the same resource in two different Petri nets. Finally, some common characteristics of discrete event systems that are present in a network of intersections controlled by traffic signals are easily represented by Petri nets, such as shared resources, synchronization, time aspects and sequence of events.

The road intersections of the previous section are modeled as the Petri net in figure 3. Intersections I1 and I2 are shared resources for the road sections, and can be well represented in the Petri net model as the I1 and I2 places. The states are represented as places and the transitions separate one state from another. For instance, GA represents a green for section A, YA a yellow and RA a red. The firing of a transition allows state changing. A token in a place represents the current state of the traffic signal (TLS).

The proposed Petri net has a deterministic duration time associated to each transition [9]. This duration, in the model, corresponds to a time delay before the firing of a transition, indicating the duration associated to a specific state. For instance, in figure 3, associating a duration of 20 seconds to transition t2 means that the transition can only be fired after 20 seconds. As a matter of fact, a green-time of 20 seconds is given to section D.

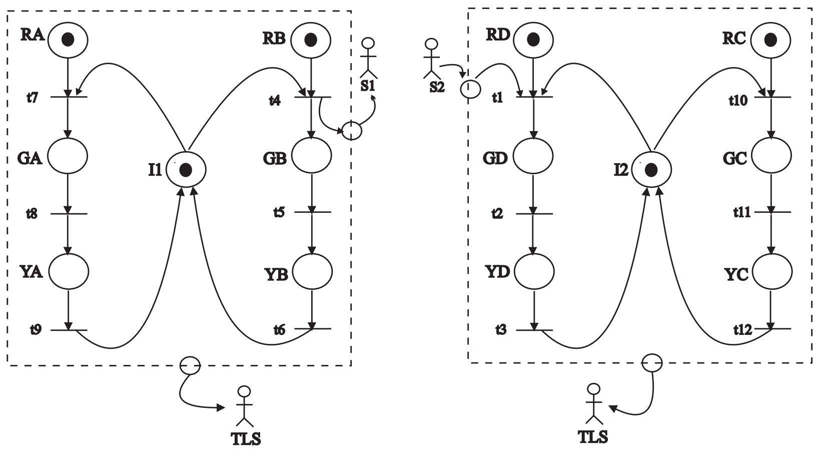

Figure 3: Global Petri net model.

4 Analysis with Linear Logic

Classical logic deals with eternal truths: once a fact is used to prove another fact, the first fact is still available. When dealing with resources that can be consumed and produced, this concept of eternal truth is not possible any-more. It's natural that resources can be counted, but this is not possible in Classical Logic. Linear Logic was then proposed by Girard [10] as a restriction of Classical Logic in order to deal with resources. The propositions in Linear Logic are resources that can be produced or consumed. A deduction represents a state changing in which resources are consumed in order to produce other resources.

As Petri nets deal correctly with the notion of resources and state changing, some first results appeared on combining Petri nets and Linear Logic [11,12]. Within Linear Logic, the places of a Petri net are considered resources. The production and consumption of instances of a resource corresponds to producing/consuming tokens in the associated place when a transition is fired.

In this paper, the translation from Petri nets to formulas of Linear Logic is done as in [13] and is given as follows. An atomic proposition P is associated with each place of the Petri net. A marking M is represented only with the connective $\otimes$, ie., a marking is represented by $M = P_1 \otimes P_2 \otimes \cdots \otimes P_k$ where P_i are places with tokens. A transition is an expression of the form $M_1 \multimap M_2$ where M_1 and M_2 are markings. In fact, for the Petri net these are the equivalent *Pre* and *Post* functions of the transition. The connective $\multimap$ is the linear implication.

A sequent $M_i, t_i \vdash M_f$ represents a scenario where M_i and M_f are respectively the initial and final markings, and t_i is a set of transitions. The $\vdash$ is a

Linear Logic primitive symbol. It divides the sequent into the left part (premisses), and the right part (conclusions). The sequent is proved by applying the rules of the sequent calculus. To prove a sequent is to show that it is syntactically correct. The proof is constructed bottom up, and stops when all the leaves of the tree are identity sequents, such as $P \vdash P$.

Within Linear Logic, the reachability problem is in fact the problem of sequent proving. This is possible because there is equivalence between the proof of Linear Logic sequents and the reachability tree in a Petri net [14]. Linear Logic has been chosen because it allows to accurately characterize state changes and production/comsumption of resources. It is also possible to evaluate scenarios with time associated to places or transitions in the Petri net. This allows the possibility of working with symbolic durations instead of real numeric ones, which can improve simulation. Another advantage is that it is possible to derive specific scenarios directly from the Petri net without constructing complete reachability trees [15]. As a direct consequence, Linear Logic can be considered as an analysis tool for Petri nets. Linear Logic rules applied to construct the proof tree are explained in the next section.

4.1 Linear sequent calculus and proof tree

In this paper, the fragment Multiplicative Intuitionistic Linear Logic is used. This fragment contains the multiplicative connective $\otimes$, representing accumulative resources, and the linear implication $\multimap$, representing causal dependency. There's no negation and the meta connective "," is commutative. The rules applied are:

$$\frac{}{F \vdash F}\text{id} \qquad \frac{\Gamma, F, G \vdash \Delta}{\Gamma, F \otimes G \vdash \Delta}\otimes_{\mathrm{L}}$$

$$\frac{\Gamma \vdash F \qquad \Delta \vdash G}{\Gamma, \Delta \vdash F \otimes G}\otimes_{\mathrm{R}} \qquad \frac{\Gamma \vdash F \qquad G \vdash \Delta}{\Gamma, F \multimap G \vdash \Delta}\multimap_{\mathrm{L}}$$

In this paper, the proof tree is build as follows [13]:

- the rule $\otimes_L$ is applied repetitively to separate a marking into atoms, which allows the application of the $\multimap_L$ rule.
- the $\multimap_L$ rule expresses a transition firing. It generates two sequents. The left sequent represents the tokens that were consumed by the transition firing. The right sequent represents the new tokens produced by the transition firing.
- the rule $\otimes_R$ transforms sequents of the form $P1, P2 \vdash P1 \otimes P2$ into two identity ones $P1 \vdash P1$ and $P2 \vdash P2$, which are proved leaves.

4.2 Scenarios

As an analysis tool for Petri nets, Linear Logic can be applied to give some important results about the models. The purpose in this paper is not to consider all evolutions of the Petri net, but reason on specific scenarios. The same approach can be easily applied to other scenarios.

4.2.1 Reachability of an unsafe state

The considered scenario is the one in which an unsafe state is reached, such as the reachability of simultaneous greens for two conflicting road sections, as for instance, A and B in figure 3. By design, places GA and GB, meaning respectively green for section A and B, are in mutual exclusion. From the Linear Logic point of view, this is equivalent to prove that the state $GA \otimes GB$ is not reachable from another state. What is to be proved is that from the initial state $(RA \otimes RB \otimes I1)$ the final state $GA \otimes GB$ is not reached. The equivalent sequent is $RA \otimes RB \otimes I1, t7, t4 \multimap GA \otimes GB$. This sequent states that from red states for both sections, it will be allowed green for section A and B simultaneously.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{RA \vdash RA \qquad I1 \vdash I1}{RA, I1 \vdash RA \otimes I1}\otimes_R \qquad GA, RB, (I1 \otimes RB \multimap GB) \vdash GA \otimes GB}{RA, RB, I1, (I1 \otimes RA \multimap GA), t_4 \vdash GA \otimes GB}\multimap_L}{RA, RB, I1, t_7, t_4 \vdash GA \otimes GB}\otimes_L}{RA \otimes RB, I1, t_7, t_4 \vdash GA \otimes GB}\otimes_L}{RA \otimes RB \otimes I1, t_7, t_4, \vdash GA \otimes GB}$$

From the proof tree generated, it is clear that the final state is not reached, as there's no further Linear Logic rules that can be applied to transform the branch $GA, RB, (I1 \otimes RB \multimap GB) \vdash GA \otimes GB$ of the proof tree into identity sequents. The same reasoning can be done firing transition t4 before transition t7. In this case, a similar tree is generated with the same final result.

4.2.2 Reachability of a desirable state

An important scenario is to know if it is possible to have simultaneous green for sections B and D, and for how long this state will remain. Knowing this information, more accurate off-line plans can be generated and can be evaluated if this approach can really improve traffic flow. From red state for sections B and D, it has to be proved that a situation where there are greens for both sections is reached.

The communication places and the sensors S1 and S2 in the global Petri net of figure 3 acts in practice as a delay, indicated by the date D. A simultaneous green for sections B and D is possible after the firing of transition t1, which leads to green for section D, and before the firing of transitions t5 or t2, which leads to yellow states.

Using symbolic dates generated after the construction of a proof tree, the first yellow will be allowed after date $d_4 + d_5$ (production of a token in YB) or after date $d_4 + d_1 + d_2 + D$ (production of a token in YD). The green for section D starts at date $D + d_4 + d_1$. The simultaneous green SG starts after the green is allowed to section D and ends before the first yellow is produced for section B or D. The equation with symbolic dates is:

$$SG = minimum(d_4 + d_5, d_4 + d_1 + d_2 + D) - (D + d_4 + d_1)$$

For instance, considering $d_4 = d_1 = D = 5$, $d_5 = 20$ and $d_2 = 40$, then $SG = 10$, which mean that there will be simultaneous green for 10 seconds. With other values, such as $d_4 = d_1 = 5$, $D = 10$, $d_5 = 45$ and $d_2 = 35$, $SG = 30$. Several other values can be simulated using these symbolic dates. Due to the possibility of modularity design with Petri nets, the same evaluations can be done for several sub-networks, and other Linear Logic expressions generated. The real numeric values can then simply be substituted into the symbolic ones. With this approach, it is not necessary to construct new reachability trees for every different set of transition times.

5 Conclusion

The paper presented a technique to model traffic signals with Petri nets. A subnetwork of traffic signals is modeled with Petri nets, and important scenarios are extracted and analyzed with Linear Logic. This is possible due to the equivalence between Linear Logic proofs and the reachability of Petri nets. With the symbolic dates, several simulations can be done just by changing real numeric values into the symbolic ones. As an analysis tool for Petri nets, some results about properties of the model can be done with Linear Logic proof trees.

Acknowledgements

This work was supported by the Next Generation Infrastructures Foundation and the Research Center Next Generation Infrastructures, both situated in Delft, The Netherlands.

References

[1] Murata, T., Petri nets: Properties, analysis and applications. *Proceedings of the IEEE*, **77(4)**, pp. 541–580, 1989.

[2] Lin, L., Nan, T., Xiangyang, M. & Fubing, S., Implementation of traffic lights control based on petri nets. *Proc. of the 6th IEEE International Conference on Intelligent Transportation Systems*, Toronto, 2003.

[3] Tolba, C., Thomas, P. & ElMoudni, A., Performances evaluation of the traffic control in a single crossroad by Petri nets. *Proceedings of the ETFA - Emerging Technologies and Factory Automation*, Lisbon, volume 2, pp. 157–160, 2003.

[4] Julvez, J. & Boel, R., Modeling and controlling traffic behavior with continuous petri nets. *Proceedings of the 16th triennial world congress of the International Federation of Automatic Control (IFAC2005)*, Prague, 2005.

[5] List, G. & Cetin, M., Modeling traffic signal control using petri nets. *IEEE Transactions on Intelligent Transportation Systems*, **5**, pp. 177–187, 2004.

[6] Febbraro, A., Giglio, D. & Sacco, N., Urban traffic control structure based on hybrid petri nets. *IEEE Transactions on Intelligent Transportation Systems*, **5(4)**, pp. 224–237, 2004.

[7] Khisty, C., & Lall, B., *Transportation Engineering: An Introduction.* Prentice Hall, 2003.

[8] Gallego, J., Farges, J. & Henry, J., Design by petri nets of an intersection signal controller. *Transportation Research Part C: Emerging Technologies*, **4(4)**, pp. 231–248, 1996.

[9] Wang, J., *Deterministic Timed Petri nets (Chapter 3).* Kluwer Academic Publishers, pp. 37–62, 1998.

[10] Girard, J., Linear logic. *Theoretical Computer Science*, **50**, pp. 1–102, 1987.

[11] Brown, C., Relating petri nets to formulas of linear logic. *Edinburgh Tech. Report ECS-LFCS-89-87*, 1989.

[12] Engberg, U. & Winskel, G., Petri nets as models of linear logic. *Proceedings of Colloquium on Trees in Algebra and Programming*, Springer-Verlag LNCS 389, pp. 147–161, 2003.

[13] Riviere, N., Pradin-Chezalviel, B. & Valette, R., Reachability and temporal conflicts in t-time petri nets. *Proceedings of the 9th International Workshop on Petri nets and Performance Models (PNPM'01)*, Aachen, pp. 229–238, 2001.

[14] Girault, F., Pradin-Chezalviel, B. & Valette, R., A logic for petri nets. *JESA: Journal Europeen des Systemes Automatises*, **31(3)**, pp. 525–542, 1997.

[15] Demmou, D., Khalfaoui, S., Guilhem, E. & Valette, R., Critical scenarios derivation methodology for mechatronic systems. *Reliability Engineering & System Safety*, **84(1)**, pp. 525–542, 2004.