



Discrete-event simulation of railway systems with hybrid models

G. Decknatel & E. Schnieder

Institut für Regelungs- und Automatisierungstechnik, Technische Universität Braunschweig, Braunschweig, Germany.

Abstract

The paper describes a multi-train simulator where the emphasis lies on integrating the continuous and discrete aspects of railway systems. The language we used for building the simulator is based on high-level Petri nets. In addition to the usual features of Petri nets, we introduced continuous transitions which are able to manipulate real-valued attributes of tokens. We outline two methods for simulating the hybrid models with a standard Petri net tool, one of which uses discrete events. Some examples of models and simulation results are given.

1 Introduction

The purpose of the project described here is to study the possibilities of modelling railway systems under the premise that they possess both continuous and discrete processes such that they belong to the class of mixed continuous-discrete systems, also called hybrid systems. More specifically, we want to explore a form of integrated modelling using a language which can be used to describe continuous as well as discrete processes and their interaction.

We decided to use a variant of hybrid Petri nets, as described in [2]. The modelling, simulation and analysis of these hybrid nets is supported by a tool for high-level Petri nets, which we extended with



some functions for the additional continuous transitions. We used two different approaches for the simulation, one of which is based on a simple numerical integration. The advantage of this approach is that it can be hidden in a lower layer of the net model, and that there are no restrictions on the formulation of continuous processes. The disadvantages are slow simulation speed and the inaccuracy of the numerical integration.

The second simulation approach uses discrete events. The advantage here is much faster simulation speed and accuracy, a limitation is imposed by the need to be able to calculate event times.

The simulation model represents the movement of trains across a given topology including points and signals. It includes not only the exact position and length of each train at any given time but also the length of the track ahead of the train which is required to come to a halt. The trains push their own braking curve, which is changing according to speed and acceleration of the train. As an example, the train may start to brake as soon as the far end of the braking curve reaches a stop signal. This allows to model different types of train control.

2 Language

Petri nets in their original, discrete form are already being employed successfully in the field of railway modelling. There are several ways to add continuous processes to the various styles of Petri nets. The discrete Petri net models of railway systems often use high-level nets, which can be characterized by the fact that the tokens are distinguishable, and can thus carry arbitrary data structures [5]. The main reason to use high-level nets in contrast to nets with anonymous tokens is that they allow more condensed and manageable models of complex systems.

The tokens in high-level nets can carry data structures such as tuples. If we allow the real numbers as a domain of the tuple elements, we are already rather close to being able to model continuous processes. We then only have to give the transitions the additional ability to manipulate the values of these tuple elements continuously [6]. This is achieved by introducing continuous transitions. It is not necessary to define a new kind of places.

With respect to the semantics of these continuous transitions, we



use two basic forms of continuous processes: integrators and algebraic relationships. Accordingly, we distinguish two forms of continuous transitions. Integrating continuous always have one input place and one output place. They manipulate attributes within the tokens on their output place such that their value is the derivative of the value of a given attribute of tokens on their input place. The algebraic transitions correspond to operators such as addition or multiplication and thus have two input places and one output place. Here, the value of an attribute of a token on the output place depends on the values of attributes from the input places.

There are functions associated to the arcs around continuous transitions that indicate which specific attribute the transition is referring to. Similar to the ordinary discrete transitions, the continuous transitions have logical guards which specify when a transition may fire and which tokens it may use.

Among the different styles of high-level Petri nets, we chose the Coloured Petri Nets by K. Jensen [4] as the basis for our hybrid nets. A more formal presentation of how we extend the definition of CPN to include the two forms of continuous transitions can be found in [1].

For practical experimentation, we use the tool Design/CPN [3], which is based on K. Jensen's definitions of Coloured Petri Nets. It has a programmer's interface which allowed us to introduce the continuous transitions graphically and syntactically as well as semantically. In order to visibly distinguish them from the ordinary transitions, which are typically drawn as squares or rectangles, and the places, which are circles or ovals, we use triangles for the integrating continuous transitions and diamond-shaped algebraic transitions.

3 Models

In this section we will describe some details of a railway simulator which was constructed with the tool Design/CPN using the hybrid nets outlined above.

CPN models are structured in pages. These are separate (non-hierarchical) nets which can be connected by transition refinement and fusion sets. Transition refinement means that one transition on a given page serves as a placeholder for another page, which is then called the subpage of the transition. A fusion set is a collection of

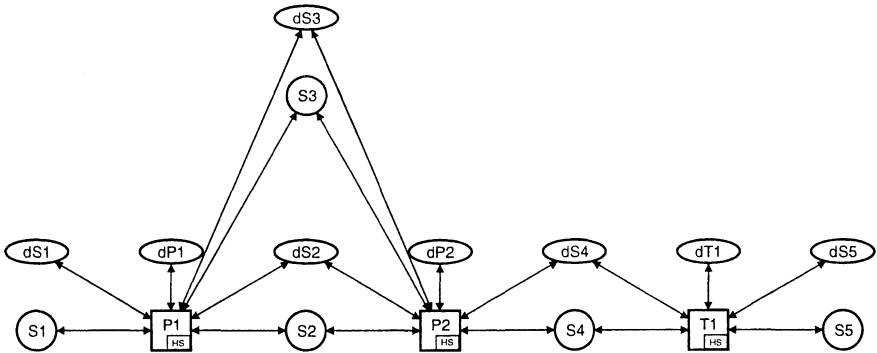


Figure 1: Model of the track topology:
 Sections, borders and points.

separate places, which means that all places within the set always carry the same tokens, such that they actually behave as one single place. This can be used in order to simplify the graphical layout of a net by creating duplicates of a place which is connected to several transitions, and it can be used to connect separate pages.

Our model consists of about thirty pages. There is one central page which contains the track topology. This allows to quickly adapt the model to different scenarios by simply replacing the topology page. Figure 1 shows a part from a model of a single line with a small station.

The places S1 to S5 represent track sections, the transitions P1 and P2 stand for points (switches), and the transition T1 is the border between the sections S3 and S4, which can e.g. be used as the position of a signal or the end of a block. The smaller ovals are places which carry specific attributes of their associated place or transition: the place dP1 for example contains data about the point P1 such as signals or counter jumps. All transitions have subpages, which is indicated by the little rectangles marked HS. There are two subpages in total, one for section borders and one for points, such that all transitions within such a net are refined by one of those two subpages.

The actual net in Design/CPN contains some further annotations which are omitted here, such as the type (color set) of the places, the mapping between the places around a transition and the places on its subpage, initial markings and arc inscriptions.

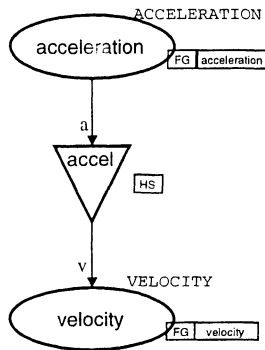


Figure 2: Integrating transition linking acceleration and velocity.

As an example for an integrating transition, Figure 2 shows the page which connects acceleration and velocity, with the integrating transition *accel*. It has a subpage which is responsible for the realization of the (pseudo-)continuous behaviour. The two places belong to fusion sets, as indicated by the boxes marked FG. These fusion sets connect this page to the other pages in the model. The colour sets (types) of the places are indicated next to the places (ACCELERATION and VELOCITY).

In order to control the movement of the trains, we include their braking distance into the model. The far end of the braking curve, i.e. the point at which the train would come to a halt if a given deceleration was applied, is moving ahead of the train. The velocity of that point depends on the current velocity and acceleration of the train, and its braking ability (assumed deceleration). Since acceleration and velocity of the train change dynamically, the velocity of the front end of the braking curve is calculated using continuous algebraic transitions as shown in Figure 3.

4 Simulation

In order to be able to use the model described above for simulation, we have to find a way to simulate the behaviour of the continuous processes as described by the continuous transitions. One possibility would be to build an interface to a tool for the continuous simulation.

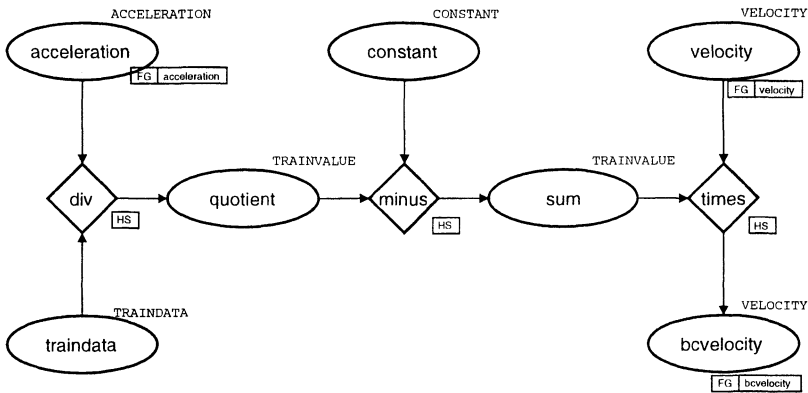


Figure 3: Continuous calculation of the velocity of the front of the braking curve.

Instead, we prefer to stay within the language of Petri nets and use the simulator which is part of the tool Design/CPN. This requires a translation of the continuous behaviour into discrete terms.

4.1 Numerical integration

The first solution is based on the idea to hide the discrete approximation of the continuous processes as a subpage below the continuous transitions. We added a function to the tool which constructs such a subpage automatically on the basis of a few parameters which are supplied by the user as comments to the continuous transition. One advantage of this approach is that it is flexible and that the modeller does not have to deal with the discretization in detail. A disadvantage is the inaccuracy of the approximation and the reduction in performance as soon as several continuous transitions with small steps sizes are active.

4.2 Discrete-event simulation

In order to overcome these disadvantages, we added a second simulation approach using discrete events. The main difference is that, instead of continuously accessing the tokens, the (pseudo-) continuous transitions fire only at event times, setting the continuous evolution of the attribute values by parameters. The tokens thus not only con-

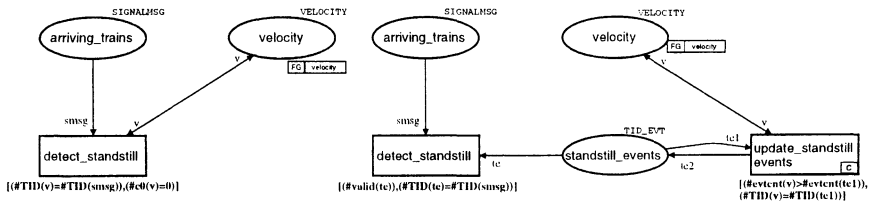


Figure 4: Detection of standstill:
Hybrid and discrete-event model.

tain the current value of such an attribute, but also parameters which describe the continuous behaviour of that value. In our example, we used quadratic polynomials, such that the function describing the behaviour of an attribute value is given by three parameters (coefficients).

For each transition which has a firing condition which depends on the value of such an attribute, such as the crossing of a threshold, there is one auxiliary transition which calculates the event time when the condition with respect to that attribute will become true. This auxiliary transition fires when the parameters of the attribute change. Figure 4 shows an example of this construction.

On the left of Figure 4, the transition **detect_standstill** is supposed to fire as soon as the velocity of a train which belongs to the **arriving trains** has reached zero. This is expressed by the firing condition (guard) below the transition, which contains two clauses. The first clause requires that the TIDs (train identifier) of the tokens from both places are identical, and the second is the condition that the velocity is zero, where $\#c0(v)$ is the value of the attribute $c0$ of the token v , in this case the train velocity.

The net on the right is the discrete-event version of this net, where **update_standstill events** is the auxiliary transition which sets the event times. It has a code region (not shown) which calls a function to calculate those times. The transition is triggered by an increase in the event counter **evtcnt**.

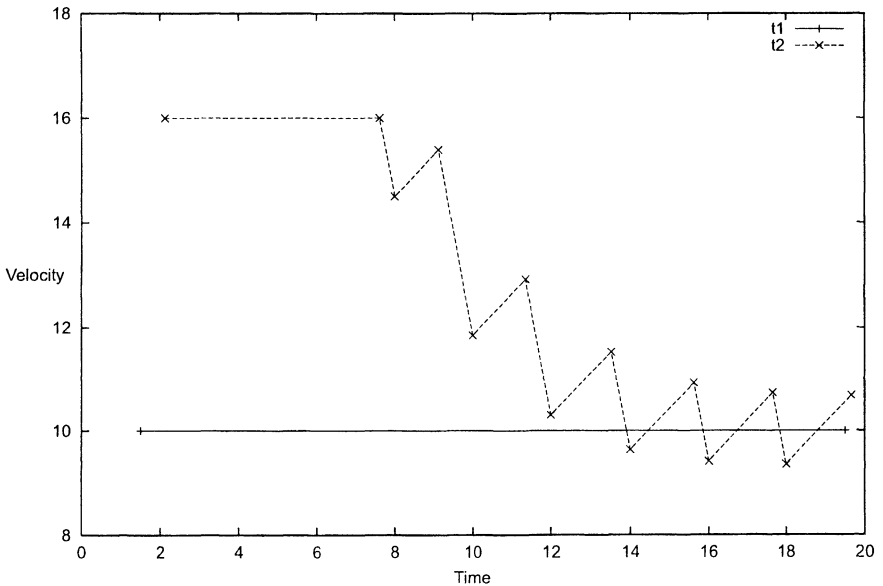


Figure 5: Chart of a simulation run: t2 follows t1.

4.3 Results

Figure 5 is a graph of a simulation run using the discrete-event simulation. It shows the velocity of two trains, t1 (straight line) and t2 (dashed line) over time. Both trains are running on the same track in the same direction, t2 is following t1. Since t2 starts with a higher velocity, it catches up and eventually has to brake down to the speed of t1. The distance between the trains is controlled by blocks, such that t2 can only enter a block after it has been cleared by t1. Since t2 uses maximum acceleration whenever possible, it alternates between acceleration and braking before the next signal.

Figure 6 shows another scenario with a simulated encounter of two trains in a station next to a single track. The graphs represent the movement of the front and rear ends of both trains. Train t1 is moving out of the single track into the station at constant speed. Train t2, travelling from left to right, first starts to brake since it must wait for t1 and then accelerates back to normal speed as soon as the track is cleared.

In addition to this graphical representation of single simulation

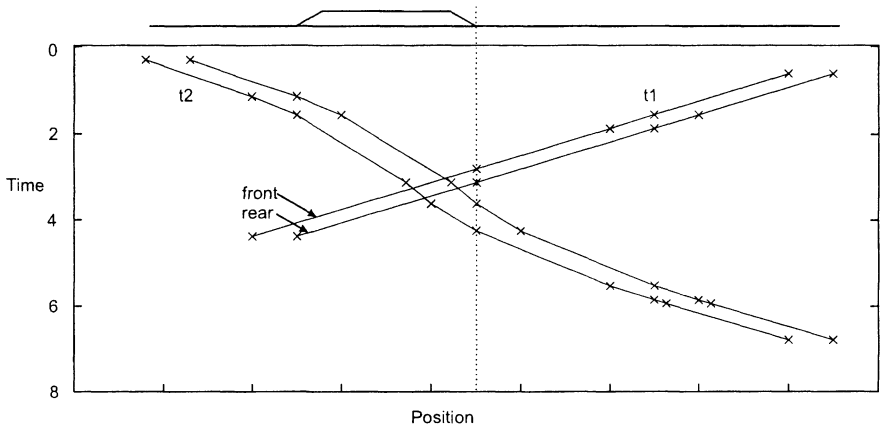


Figure 6: Chart of a simulation run: Single line.

runs, the discrete-event version of the simulation model allows the calculation of the reachability graph, which reveals all possible sequences of actions and all reachable states starting from a given initial state.

5 Conclusion

We showed how a particular variant of hybrid high-level nets on the basis of Coloured Petri Nets can be used to model railway systems as hybrid systems. This allows to combine models of complex (discrete) control systems with continuous processes such as the train motion. Using a discrete-event approach we were able to achieve a reasonably efficient simulation of typical scenarios, details of which were presented above. In addition, it is possible to automatically construct the reachability graph for any given scenario.

Acknowledgement The work presented here is part of the project MASKoBa which is supported by the Deutsche Forschungsgemeinschaft (DFG) within the program KONDISK.

References

- [1] Decknatel, G. & Schnieder, E. An Extension of Coloured Petri Nets for the Simulation of Mixed Continuous-Discrete Systems.



- Proc. of the 11th European Simulation Symposium*, eds. G. Horton, D. Möller & U. Rüde, Society for Computer Simulation International, pp. 53-60, 1999.
- [2] Decknatel, G. & Schnieder, E. Hybrid Petri nets as a new formalism for modelling railway systems. *Computers in Railways VI*, Computational Mechanics Publications/WIT Press, pp. 773-782, 1998.
- [3] University of Aarhus, Department of Computer Science, CPN Group. *Design/CPN Version 4.0.1*,
<http://www.daimi.au.dk/designCPN>.
- [4] Jensen, K. *Coloured Petri Nets - Basic Concepts, Analysis Methods and Practical Use*, 3 Volumes, Monographs in Theoretical Computer Science, Springer-Verlag, 1997.
- [5] Jensen, K. & Rozenberg. G. (eds). *High-level Petri Nets: theory and application*, Springer-Verlag, 1991.
- [6] Wieting, R. Hybrid High-Level Nets. *Proc. of the 1996 Winter Simulation Conf., Coronado, CA, USA*, eds. J.M. Charnes, D.J. Morrice, D.T. Brunner & J.J. Swain, pp. 848-855, 1996.