A General-Purpose Proxel Simulator for an Industrial Software Tool

Fabian Wickborn, Graham Horton*
Otto-von-Guericke Universität Magdeburg
{fabian,graham}@sim-md.de

Stefan Heller, Felix Engelhard†
DaimlerChrysler AG
{stefan.heller,felix.engelhard}@daimlerchrysler.com

Abstract

At DaimlerChrysler, the reliability and safety of automobiles are analysed using simulation. While discrete-event simulation can be used for these models, this approach may not deliver accurate results in acceptable time. Markov chain modelling for a state-space-based analysis is too complex for the user. Proxel-based simulation has shown to combine the best of both worlds and even to be more efficient for small models. In this paper we present a general-purpose proxel-based simulator that has been implemented into an analysis tool of DaimlerChrysler. Furthermore, a caching strategy is presented which reduces the computational effort of proxel-based simulation.

1 Introduction

In recent years, simulation has become a widespread analysis instrument for various system properties, such as cost, performance, reliability, and safety. Simulation users can choose between two different paradigms for the analysis of his model. On the one hand, there is event-based simulation, often called Discrete Event Simulation (DES). Advantages of DES are, that it is an well-understood approach and can make use of general random distribution for activity times. The major drawback is that since its results are random variables, DES may not be satisfactory for simulation models concerning reliability and safety. In this cases, rare events and their consequences are not guaranteed to be represented well enough in the simulation results to accurately capture the behaviour of the model.
On the other hand, a user can choose to use state-space based analysis with the paradigm of Markov Chains (MC). While these numerical solution methods of MC are known to be very accurate, the user is restricted to the use of the exponential distribution for the modelling of activity time. purposes. In most cased, this restriction is unacceptable for the user's purposes

*Fakultät für Informatik, Institut für Simulation and Graphik, D-39016 Magdeburg, Germany
†DaimlerChrysler AG, REI/AA, D-70546 Stuttgart, Germany

At DaimlerChrysler, modelling and analysis of reliability and safety models is currently done with the software tool EXPECT which was developed at DaimlerChrysler Research Center for corporate internal use [Gre04]. The software emphasises the modelling paradigm of stochastic Petri nets (SPN) with general firing time distributions. DES is the main analysis method for the SPN models. Some of these models contain events, which occur very seldom within to the simulated time interval. To gain confidence in the simulation results, a large number of DES runs has to be performed. That leads to computation times of many hours or even days. This is unacceptable for industrial purposes. To speed up the computation, the simulation runs have been by distributed to multiple workstations. However, by doing so, the accumulated computational effort is not reduced.
Proxel-based simulation was introduced by Horton as a state-space-based analysis method for Stochastic Petri Nets, which are analysed as Discrete-Time Markov Chains (DTMC) [Hor02]. It computes the probability of all potential state changes in a deterministic way just like Markov chains. However, it is not restricted to the use of exponential distributions. Supplementary variables are used for the storage of the age information of enabled state transitions. The probability of state changes is computed by means of the instantaneous rate functions of the associated distributions. The age information is stored together with the discrete state and the probability, forming a Probability Element (short: proxel). The proxel method generates and tracks all possible developments of the system behaviour of the system for discrete steps over the simulation time. Rare events and the thereby reached system states are guaranteed to be considered. The method has shown to be applicable for the numerical simulation of SPNs. Furthermore, it has successfully been applied for the numerical solution of small warranty models at DaimlerChrysler. The method has dramatically reduced the analysis time of these models from several hours to a few seconds [LMH04a].

2 General-purpose Proxel Algorithm

EXPECT's primary modelling paradigm are SPNs with general firing times and immediate transitions. Research has already shown that the proxel method is able to handle these models [LMH04b, LMH04b]. While it is possible to create proxel-based special-purpose solvers for most of the analysis problems at DaimlerChrysler, this would require additional programming effort for each individual model. For the industrial environment, a general-purpose simulator based on proxels is more valuable in the long term.
For this reason, a general-purpose proxel simulator for SPNs has been implemented into the industrial software tool EXPECT. It is an integral part of the tool and enables the user to perform a proxel-based simulation of a SPN with no specific knowledge about the method itself. The input for the new simulator consists of the same model that would otherwise be fed into the discrete-event simulator. Necessary parameters for the analysis are the simulated mission time of the system, and the initial time step size. The proxel-based simulator is designed to compute the same result statistics as the DES (e.g., the firing rate of a transitions) and to present them in a similar fashion. As a result, a user who is able to use the DES analysis functionality of EXPECT is able to use the proxel-based simulator as well

without special teaching.
Proxel-based simulation operates on the reachability graph (RG) of a model. To build such an RG, a SPN model has to be generated to determine its discrete states and the relationships between them. Until now, this was done explicitly as a preprocessing step for each model of interest. Since such preprocessing would not terminate for an SPN model with an infinite number of states, we construct the reachability graph on-the-fly during the simulation. Thus, the new proxel-based simulator can handle models with an infinite reachability graph.
The results of proxel-based simulation is improved by Richardson's Extrapolation technique [Hor02], in which the error of the approximation is considered to be a function of the time step size. This is possible because the proxel method resembles an integration method for the probability state equations of the analysed model. Our proxel-based simulator automatically performs extrapolation in an iterative way: After each completed proxel analysis, the time step size is decreased by a user-provided factor and the simulation is restarted with this smaller step size. The results obtained from each iteration are used to extrapolate a result for a theoretically infinitesimal time step size. Therefore, the quality of the results is improved with each iteration without any interaction needed. This approach is considered to be most convenient for the user, since he or she is able to gain a rough approximation quickly or to invest more time for a more precise computation.
The model paradigm of EXPECT augments the SPN paradigm with the ability to specify output measures as reward-based functions, which can be impulse rewards (Re_{imp}), accumulated rate-rewards (Re_{acc}), or non-accumulated rate-rewards (Re_{nonAcc}). The values of this rewards are computed on-the-fly from the probability values of the proxels and the instantaneous reward values

$$Re_{imp} = \frac{1}{T_{max}} \sum_{P|\, Tr \in P.enabled} P.p \cdot h_{Tr}(k\tau) \cdot f_{Re} \tag{1}$$

$$Re_{acc} = \sum_{P} \sum_{Tr \in P.enabled} P.p \cdot h_{Tr}(k\tau) \cdot f_{Re} \tag{2}$$

$$Re_{nonAcc} = \frac{\tau}{T_{max}} \sum_{P} P.p \cdot f_{Re} \tag{3}$$

In the above equations, k is the current time step, τ is the time step size, T_{max} is the simulated mission time, f_{Re} is the instantaneous reward value, P is a proxel, $P.p$ is the probability of the P, $P.enabled$ is the set of enabled transition in the state described by P, Tr is a transition, and h_{Tr} is the instantaneous rate function of Tr.
A essential feature for the modelling requirements of DaimlerChrysler is the server multiplicity of SPN transitions, whereby one transition represents multiple equivalent activities (if the model has such). This feature makes creating the model easier and reduces the size of the created SPN models. If it is used, the number of active servers within a transition depends on the number of tokens in the places connected to the transition via the input arcs. Because of the change in that number, it is possible that fewer servers are pre-empted than there were enabled ones. Analogously, with transitions with Race Age policy, fewer

servers may become re-enabled than there were previously pre-empted servers. With proxel-based simulation, the age information of all active or pre-empted servers is stored. Every different combination of pre-empted servers can lead to a different model state, since the newly reached states have different age information. Therefore, all possible combinations and their individual probabilities have to be considered. This problem can be reduced to the problem of generating all k out of n combinations as solved by Knuth [Knu04]. By doing so, server multiplicities can be mapped to proxel-based simulation. This also is considered to be convenient for the users at DaimlerChrysler, since thereby models can be analysed without modification.

3 Caching of Proxel Adjacency

The ability to handle general-purpose models causes an algorithmic overhead that does not occur in otherwise. Therefore, the performance of a general-purpose solver would be worse than the performance of the special-purpose solver. However, the performance of the general-purpose solver is improved by a new caching strategy for the proxels.

One part of the information contained in a proxel is the discrete state of the model, that is, the marking of the SPN. Let T be a transition and M_0 and M_1 be two consecutive markings of the SPN, so that the firing of T in the marking M_0 leads to the marking M_1. Basically, this is the information one would store in a reachability graph. Now, any proxel whose discrete state is M_0 would create a successor proxel with the discrete state M_1, as T fires. proxels are stored in a way, so that all proxels belonging to a specific marking, are accessible as once. If a look-up has to be made for a given model state, one has to look only in the set of proxels belonging to the same marking. For the future, two problems exist. The first one is how to store the markings in the most efficient way. The second one is how to store the set of proxels belonging to each marking. Both problems are similar to the problems encountered with any reachability graph generator.

In previous publications, each proxel was the state of the model (consisting of discrete marking and age information) which was only valid for one point in time [Hor02]. During the simulation, multiple proxels describing the same state for different points in time were created. With EXPECT, each unique model state is stored in a single proxel. Each proxel stores the probability of its model state for multiple points in time. The successors proxels originating from each proxel are stored as a mapping of the transitions via which the succeeding stats are reached. Thereby, the SPN model is implicitly transformed into a DTMC in which supplementary variables are used to store the age information. The algorithm is analogous to the reachability graph creation for the SPN, but the DTMC is solved at the same time as it is created.

With EXPECT, the parameters of the SPN can be functionally dependent on various values, e.g., the simulated time. Hence, it is possible that the succeeding state reached from a proxel via a specific transition changes over time. For most models, however, these succeeding stats are unique. Then, the succeeding states of every proxel do not need to be computed more than once and the flow of probability can be computed instantaneously. Succeeding proxels can be accessed in $O(1)$ after they have been computed initially. If the succeeding

states are not unique, they still can be cached. Proxels succeeding from the same origin state and event are stored together in a mapping with the transition as a key. Thus, the effort for accessing a specific successor is $O(s(P,T))$, where s is the number of successors for the origin proxel P and the firing transition T. In most of the cases, $s(P,T)$ is considered to negligible small compared to the number of all proxels during the simulation.
Figure 1 shows a schematic of the proxel data structure EXPECT, which resembles the above explanations. A proxel holds a reference to its marking (which itself references to all associated proxels) and the age information for the enabled or pre-empted transitions. In addition, multiple probability values are stored with their associated time points. Finally, the succeeding proxels are accessible in a mapping in which the enabled transitions act as keys.

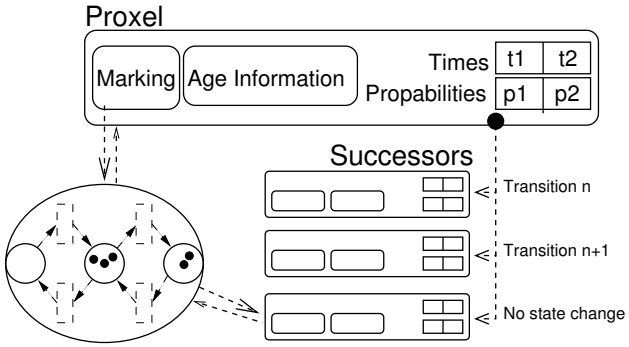

Figure 1: Caching of proxel adjacency

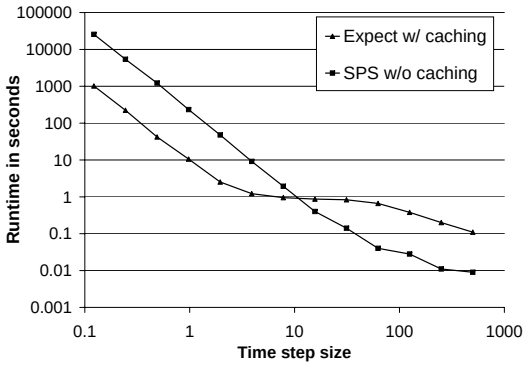

Figure 2: Runtime with and without caching of proxel adjacency

Figure 2 shows measured run times of the proxel simulator of EXPECT and a special-purpose solver (SPS). All measurements were made on a Pentium IV with 3.06 GHz. The special-purpose solver does not use the above caching strategy. Both solvers consider the

automobile warranty model from [LMH04a] for a mission time of 5000 time units. In case of a large time step size, the run time of both solvers is less than one second and the special-purpose solver performs better. This is due to the overhead of the general-purpose solver. The computational effort of both solvers increases as the step size is reduced. The general-purpose solver has the same or better performance when the step size is equal to or less than ten time units. For this model, a step size of ten time units is the threshold, from which on the advantage of the caching over-weighs the overhead. The performance of the simulator with caching is increased by the factor of 50 compared to the simulator with no caching. So, caching of proxel adjacency improves the performance of the proxel-based simulation method in the long term.

4 Conclusions

In this paper, we presented a general-purpose proxel simulator which is integrated into EXPECT, an industrial tool for reliability, safety and cost analyses which is developed and used by DaimlerChrysler. Our implementation of the proxel method offers proxel-based simulation of the general SPN models at DaimlerChrysler and is easy to use for users with no knowledge about state-space based analysis or proxels. Furthermore, we invented a caching strategy which increases the performance of the proxel-based method. We presented an example in which the performance could be improved by a factor of 50.
Our current research is aimed towards further enhancing the proxel algorithm and the data structures in cooperation with DaimlerChrysler. The resulting methods will be tested on industrial reliability and safety models and tuned to the industrial accuracy requirements.

References

[Gre04] Stefan Greiner. Using simulation to predict quality and cost in the automotive business. In *Proceedings of the 18th European Simulation Multiconference*, Magdeburg, Germany, 2004. SCS European Publishing House.

[Hor02] Graham Horton. A new paradigm for the numerical simulation of stochastic petri nets with general firing times. In *Proceedings of the European Simulation Symposium 2002*, Dresden, October 2002. SCS European Publishing House.

[Knu04] Donald E. Knuth. *The Art of Computer Programming, Pre-Fascicle 3A, A Draft of Section 7.2.1.3: Generating All Combinations*. Zeroth printing (revision 7) edition, 2004.

[LMH04a] Sanja Lazarova-Molnar and Graham Horton. Proxel-based simulation of a warranty model. In *European Simulation multiconference 2004*. SCS European Publishing House, 2004.

[LMH04b] Sanja Lazarova-Molnar and Graham Horton. Proxel-based simulation of stochastic petri nets. In *Simulation und Visualisierung 2004*. SCS European Publishing House, 2004.