SOLVING HIDDEN NON-MARKOVIAN MODELS: HOW TO COMPUTE CONDITIONAL STATE CHANGE PROBABILITIES

Claudia Krull[(a)], Graham Horton[(b)]

[(a)(b)]University of Magdeburg, Department of Simulation and Graphics

[(a)]claudia@sim-md.de, [(b)]graham@sim-md.de

ABSTRACT

Current production systems produce a wealth of information in the form of protocols, which is hardly exploited, often due to a lack of appropriate analysis methods. Hidden non-Markovian Models attempt to relieve this problem. They enable the analysis of not observable systems only based on recorded output. The paper implements and tests solution algorithms based on the original Hidden Markov Model algorithms. The problem of computing the conditional state change probabilities at specific points in time is solved. The implemented algorithms compute the probability of a given output sequence and the most probable generating path for Markov regenerative models. This enables the analysis of protocols of not observable systems and the solution of problems that cannot be solved using existing methods. This opens up new application areas, including a different approach to mining the wealth of data already collected today with huge monetary effort.

Keywords: hidden non-Markovian models, Hidden Markov Models, discrete stochastic models.

1. INTRODUCTION

Today machines and electronic components are getting more and more complex. As a result of that growing complexity they can produce a wealth of information on the system behavior in the form of protocols. These protocols are most of the time discarded or simply archived and not analyzed properly, often due to a lack of appropriate tools. Another problem is, that often the protocol entries are ambiguous. An error for example could have been produced by several internal causes and each cause could also result in different errors. If that is the case, finding the exact or even a probable system behavior that produced the given protocol is almost impossible using existing methods.

The recently developed paradigm of Hidden non-Markovian Models (HnMM) strives to relieve this problem. HnMM are an extension of Hidden Markov Models to more general discrete stochastic hidden models. HMM can analyze hidden behavior based only on recorded system output. However, the hidden model is a discrete-time Markov chain and therefore severely limited. The extension to discrete stochastic hidden models enables more realistic models and the analysis of real life systems. These HnMM can in theory analyze the recorded protocols of not observed systems and determine the protocol probability or the most likely generating behavior.

Solving HnMM is still an issue of current research. HnMM can be analyzed using Proxel-based simulation (Lazarova-Molnar 2005, Wickborn et al. 2006, Krull and Horton 2008), which is however quite expensive, because it computes the complete reachable model state space and consequently all possible generating paths. An alternative solution method was proposed in (Krull and Horton 2009), where the theory for HnMM was described as well as a theoretical transfer of the original HMM solution algorithms to the new paradigm.

This paper shows an implementation of the adapted HMM algorithms. It also solves the problem of how to compute the conditional state change probabilities at a given point in time in continuous time. This is not trivial, since in theory the probability of a state change via a continuous distribution function at a specific point in time is 0. Furthermore the performance of the implemented algorithms is tested under growing model size and growing protocol length. A comparison with the current Proxel-based solution algorithm will show the differences of the two approaches. The adapted HMM algorithms are much faster and enable the analysis of longer sequences and larger models. However, the algorithms only compute the cumulative state probability or the one most likely generating path. Therefore both solution methods have different application areas and should be developed further to be applicable to real world problems.

1.1. Possible Application Areas of HnMM

Application areas encompass all engineering problems where the true system or machine state is not detectable or has not been stored and only protocols containing ambiguous records are available. A possible application area outside of engineering could be the analysis of the symptoms of a patient, which can be interpreted as the output of the unobservable true state of his or her disease.

Today in some parts of the industry a huge effort is invested in collecting data on existing production lines by installing internal sensors and logging the observed processes. The resulting amount of data is often huge

and too often not consistent, since the calibration and coordination of different sensors can be very difficult. Consequently the analysis of the data can be tedious. Often the effort that would need to be invested to gain reliable sensor equipment and data analysis is larger than the expected benefit of such a step.

A concrete industry application of HnMM could therefore be the following. A small production line is being observed over a limited period of time through external sensors such as cameras or photoelectric barriers. These sensors can only detect the visible part of the production process and might not be able to observe the individual machines' or products' status or a hidden buffer content. The analysis of these logs can help determine the working status of the machines and their current performance indices, helping the owner of the production line in his decisions regarding the modernization or replacement of machines. Hidden models are necessary in this context, since only a portion of the production process can be observed. However, HMM are not sufficient, since a production process is most likely time dependent and includes several non-Markovian transitions. Only the extension to Hidden non-Markovian Models can provide a time dependent model for the hidden process.

The application of the proposed method and HnMM analysis in this example could shift and considerably reduce the necessary investment for sensor equipment and resulting effort for data collection, enabling also smaller firms to conduct such studies.

2. STATE OF THE ART

2.1. Hidden Markov Models

Hidden Markov Models (HMM) (Rabiner 1989, Fink 2008) are a well known paradigm in the realm of speech and pattern recognition. They are used to determine the hidden meaning of an observed output in the form of recorded speech or scanned pictures or letters. HMM consist of a discrete-time Markov chain (DTMC) (Bolch et. al 2006) as hidden model and a second stochastic process emitting symbols with given probabilities based on the current system state. Given a sequence of output symbols, one can solve the following three tasks:

- Evaluation: Determine the probability of that sequence for a given model.
- Decoding: Determine the most likely generating state sequence for a given model.
- Training: Train an initialized model to produce that sequence most likely.

Algorithms exist to solve all of these problems. The evaluation and decoding problem are solved through iterative algorithms. Training is by far the most complex task of the three and is usually solved through an optimization-like method.

The applicability of HMM to our area of interest is limited due to DTMCs as hidden model. These can only accurately represent memoryless processes and are therefore not very realistic for most applications. Current extensions of HMM focus on increasing the speech recognition capability (Fink 2008). Most of them change the output process from discrete to continuous. The few attempts to generalize the hidden process of an HMM were abandoned due to the minimal increase in speech recognition capability, compared to the large increase in computational complexity of the solution algorithms.

2.2. Hidden non-Markovian Models

The extension of HMM to more general models leads to so-called Hidden non-Markovian Models (HnMM). These have been formalized in (Krull and Horton 2009) and the difference to HMM is twofold. HnMM use the state space of a discrete stochastic model (DSM) as hidden model, enabling non-Markovian state transitions. Furthermore, they associate the symbol emissions with the state changes, since these are most often the objects of interest in a DSM.

The Proxel-based simulation algorithm was proposed as HnMM solution method even before formalization (Wickborn et al. 2006, Krull and Horton 2008). The Proxel algorithm is a state space-based simulation method that discovers all possible system developments in discrete time steps (Horton 2002, Lazarova-Molnar 2005). These possible development paths can correspond to generating paths of HnMM output sequences. The solution of HnMM via Proxels is however quite expensive. The method explores all possible generating paths and has therefore a large memory and runtime requirement and is limited to very small models.

The goal of this paper is to implement the adapted HMM algorithms presented by Krull and Horton (2009) that promise to be more efficient. One major challenge is the computation of conditional state change probabilities at given points in time in continuous time. This would enable the analysis of larger and therefore more realistic models in order to increase the practical applicability of HnMM.

3. ADAPTED HMM ALGORITHMS

3.1. Existing HnMM Theory

This section gives a brief definition of HnMM and their components. A more detailed formalization can be found in Krull and Horton (2009). A Hidden non-Markovian Model contains the state space of a discrete stochastic system as hidden model. The symbol outputs are associated to the transitions.

$$(S, C, V, A, B, \pi) \tag{1}$$

An HnMM is represented by a 6-tuple as seen in Equation (1). The first three elements are the set of discrete system states *S*, the set of state changes *C*, the set of output symbols *V*. The transition behavior is described by matrix *A*, which contains the instantaneous

rate functions (IRF) (Horton 2002) of the state transitions. The symbol output process is described by B, where each state transition is associated with output symbols, each with its own emission probability. The initial probability vector π contains the probability to be in each of the discrete system states at the start of the analysis.

For HnMM analysis one also needs a representation for the observed symbol sequence O and a sequence of state changes Q. Both sequences contain elements associated with a time stamp (see Equations (2) and (3)).

$$O = \{(o_1, t_1), (o_2, t_2), \dots (o_T, t_T)\} \quad (2)$$
$$Q = \{(c_1, t_1), (c_2, t_2), \dots (c_T, t_T)\} \quad (3)$$

In the paper Krull and Horton (2009) also formalized the adaption of the original HMM solution algorithms to HnMM. However, this was only possible for Markov regenerative processes that regenerate at every firing of a transition. Nevertheless we hope that for that class of models the algorithms will show much better performance than Proxel-based HnMM analysis. In the current paper we also concentrate on models where all transitions emit symbols, and where therefore each state change is represented by a symbol emission. This model configuration poses the easiest one and should therefore be implemented first.

3.2. Computing the Conditional State Change Probability

The key challenge when implementing the adapted HMM algorithms is the computation of the conditional state change probability. Equation (4) describes this probability of making the state change from state s_i to state s_j at time t_n under the condition of emitting symbol o_n. The computation of this quantity is not straight forward, since the point in time of the state change is predetermined by the time stamp in the symbol sequence. In theory, the probability of a state change at a specific point in time via a continuous distribution function is 0. This can however not be the solution for the given case, because we know that the state change will happen at exactly that point in time.

$$P(c_{ij} \text{ at } t_n | o_n) = p \quad (4)$$

The solution we propose is to compute the probability to remain in state s_i until time t_n and then assume a total state change probability of 1, meaning that all of the remaining probability is that of the state change. The current rate of a state change that has not occurred yet can be described by the instantaneous rate function (IRF) (Horton 2002). This can also be used to compute the probability to not perform a state change within a certain time period. The exact remaining probability can be computed using a parallel ODE45 (Buchholz 2008) integration scheme for the transitions IRF (or the in Markov regenerative case the transitions cumulative distribution function (CDF)). If at the time of the symbol emission there are multiple state changes possible, the remaining probability is weighted with the normalized IRF of each transition (see Equation (5)).

$$p = P(stay\ in\ s_i\ until\ t_n) * \frac{\mu_{ij}(t_n - t_{n-1})}{\sum_k \mu_{ik}(t_n - t_{n-1})} \quad (5)$$

This measure should be a good estimate of the conditional state change probability at a specific point in time. The experiments will show its practical applicability.

3.3. Implementation of the Adapted HMM Algorithms

The HnMM solution algorithms based on the original HMM algorithms were described in Krull and Horton (2009). The implementation of these algorithms was straight forward, once the problem of computing the conditional state change probability had been solved.

The Forward algorithm and the Backward algorithm (Rabiner 1989) solve the evaluation problem for HMM. Their general structure could be transferred directly to HnMM (see Krull and Horton (2009) for further details). The conditional α and β measures were computed for every time stamp and the final trace probability derived.

The Viterbi algorithm solves the decoding problem for HMM. It is a greedy algorithm that computes the most likely generating path for a symbol sequence. The conditional γ probabilities were also computed for every time stamp. The backtracking yielded the most likely generating sequence in terms of state changes that happened. This is actually the most likely sequence, since we are dealing with a Markov regenerative model that regenerates at every firing of a transitions. When lifting that restriction, the optimality of the path discovered can no longer be guaranteed.

The training algorithm was not attempted in this implementation. Already in the formalization paper it was stated, that it is unclear how to train non-Markovian state transitions. Therefore this HMM problem needs to be tackled otherwise. One method was already described in Isensee et al. (2006). There, discrete phase-type distributions were used to turn a non-Markovian model into a DTMC, which was then trained using the original Baum-Welch algorithm. We will therefore not investigate the problem of model training further in the current paper.

The current implementation now enables the analysis of some classes of HnMM through the adapted Forward and Viterbi algorithms.

4. EXPERIMENTS

This section describes some experiments conducted to show the correct functionality of the adapted HMM algorithms, to compare it with the Proxel-based solution algorithms and to test its performance.

The model topology used for the experiments is a machine maintenance model with three states. The hidden model part in the form of a stochastic Petri net is

shown in Figure 1. The machine can be in working condition (*OK*), under maintenance (*Maint*) or failed (*Failed*). The transitions between these states are non-Markovian except for the failure distribution and every transition can emit symbols.

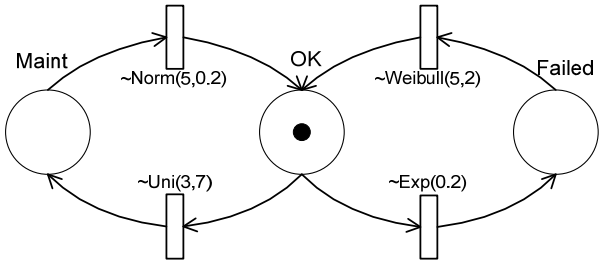

Figure 1 : Example 3-State Model Hidden Part

The complete HnMM consists of the following elements $(S, C, V, A(t), B, \pi)$. A full description of the notation can be found in Krull and Horton (2009).

$$S = \{OK, Maint, Faíled\}$$
$$C = \{MaintenanceInterval, Maintenance, Failure, Repair\} = \{MI, M, F, R\}$$
$$V = \{A, B, C, D, E\}$$
$$A(t) = \begin{bmatrix} 0 & \mu_{MI}(t) & \mu_F(t) \\ \mu_M(t) & 0 & 0 \\ \mu_R(t) & 0 & 0 \end{bmatrix}$$
$$B = \begin{bmatrix} 0.1 & 0.2 & 0.2 & 0.4 & 0.1 \\ 0.3 & 0.1 & 0.2 & 0.1 & 0.3 \\ 0.1 & 0.3 & 0.1 & 0.2 & 0.3 \\ 0.4 & 0.2 & 0.1 & 0.1 & 0.2 \end{bmatrix}$$
$$\pi = (1,0,0)$$

The output sequences consisted of random series of the five output symbol letters of length 10 to 100,000. The time stamps were equally spaced at intervals of 5 time units.

4.1. Validation Experiments

The first validation experiment compared the resulting trace probability and Viterbi path of the adapted HnMM algorithms with the Proxel-based HnMM analysis. Both algorithms can handle non-Markovian distributions and symbol outputs associated with the transitions. Therefore the same model was used for both algorithms. A symbol sequence of length 20 was used as benchmark.

Both the adapted HMM algorithm and the Proxel algorithm determined the same most likely path of state changes to have produced this sequence. This is the most important fact, showing the equivalence of both approaches. The total probability of the trace was in the same order of magnitude for both algorithms. An exact match could not be obtained due to the Proxel algorithm using a discretization time step of 1 time unit, leading to an approximate result. Furthermore, the absolute probability values are not as important as the difference in values obtained for different traces or different generating paths, using the same tool. These help discriminate among different traces and paths to choose the most likely one.

The second validation experiment compared the results of the adapted HMM algorithms to the results of the original HMM algorithms. For that experiment an HnMM had to be adapted to mimic the behavior of an HMM emitting a symbol in every step. This included using only transitions with exponential distributions. Since the resulting HMM and HnMM model only correspond approximately to each other, the best match that could be achieved here was also the same most likely generating state change sequence (HnMM) corresponding to the Viterbi path (HMM).

These two experiments showed that the adaption of the HMM algorithms did not affect the main functionality in a serious way, so that they still compute valid results at least in terms of the most likely generating path.

4.2. Performance Experiments

The second set of experiments was conducted to obtain information on the parameters influencing the performance of the adapted HMM algorithms. The machine maintenance model and symbol traces of a maximum length of 100,000 were used. Furthermore was the model size varied by increasing the number of states. This was done by replicating the machine maintenance model and concatenating the states. The measure of interest was the combined *Forward* and *Viterbi* algorithm runtime in seconds on a laptop with an Intel® Core™2 Duo CPU with 2 GHz and 2GB RAM running Windows XP.

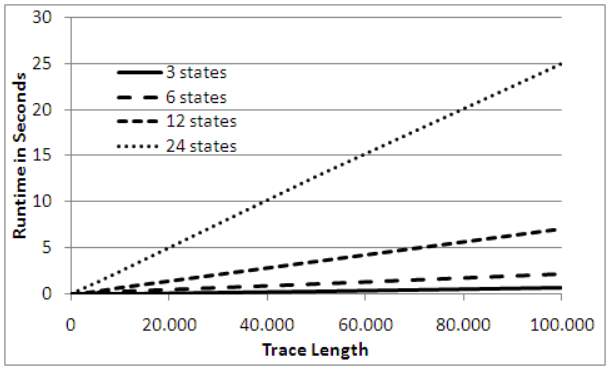

Figure 2 : Runtime over Trace Length for Different Model Sizes

Figure 2 shows the development of the algorithm runtime with increasing trace length for different model sizes. All four graphs show a linear increase in runtime with increasing trace length. This behavior is as expected. It corresponds to the linear relationship of runtime to trace length for the original HMM algorithms. The memory requirement was not tested and compared here, since due to the iterative nature of the algorithms the amount of data that needs to be stored in working memory is constant for every step and only marginally increases with growing model size.

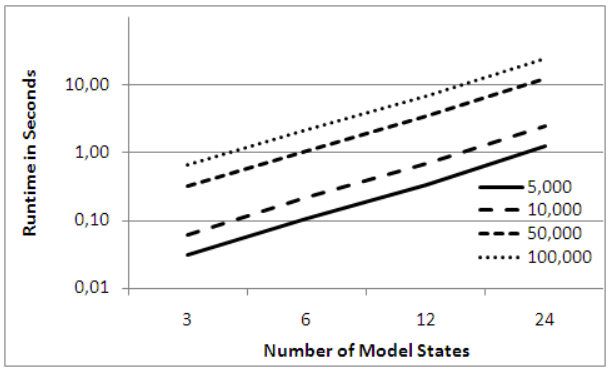

Figure 3 : Algorithm Runtime (logarithmic scaling) over Model Size for Different Trace Lengths

Figure 3 shows the same data from a different perspective. Here the runtime of the algorithms is shown in relation to the size of the model for four traces of different length. The runtime is depicted in logarithmic scaling. One can see that with growing model size, the runtime grows faster than linearly but somewhat exponentially. This is due to the increased complexity of the model that increases the amount of computation time needed at every step.

Another experiment was conducted testing the influence of the number of output symbols on the algorithm performance. Since increasing the number of output symbols did not affect algorithm performance, the results are not included here.

Overall the runtime behavior of the adapted HMM algorithms is as expected very similar to that of the original HMM algorithms. However, the absolute speed of the computation is very fast, even with growing model sizes. This promises a good practical applicability.

4.3. Comparison Experiments

The last set of experiments compared the results and performance of the adapted HMM algorithm with that of the Proxel HnMM analysis. As seen above, the runtime cost of the adapted HMM increases linearly with the sequence length. The runtime cost and memory requirement of the Proxel method on the other hand increases exponentially with increasing sequence length, mostly due to a growing number of possible generating paths that need to be handled in every step.

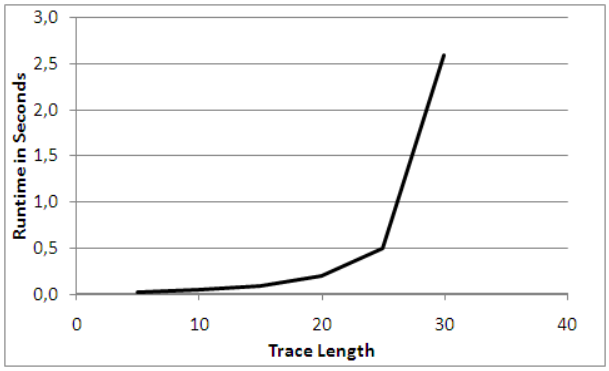

Figure 4 : Proxel HnMM Analysis Algorithm Runtime over Trace Length

The performance of the Proxel method is directly influenced by the discretization time step, which has no equivalent in the adapted HMM algorithms. Therefore the Proxel algorithm is inherently slower and more memory consuming than the adapted HMM algorithms. This results for example in a runtime of several seconds for the above three state model with a sequence length of 30 (see Figure 4), which is already several orders of magnitude slower than adapted HMM. Using significantly longer traces, the Proxel algorithm aborted without a result, because of too many recursions. Adapting the Proxel algorithm such that it can handle longer traces is one area if future research.

The results of the Proxel method on the other hand are more rich than those of the adapted HMM algorithms. The Proxel method computes in one run the total trace probability as well as all possible generating paths above a certain cutoff probability. The adapted Viterbi algorithm only returns the one most likely generating path. A ranking of all or at least of the top X paths enables their comparison to gain valuable insights. A situation where the top paths have almost equal probability, making them equally likely, would not be apparent using the adapted Viterbi algorithm.

The last point is the applicability of the methods. The adapted HMM algorithms are right now only applicable to Markov regenerative models that regenerate at every firing of a transition, and where all transitions emit symbols. This limits the model class that can be analyzed using the proposed algorithms significantly. The Proxel method on the other hand does not have these restrictions, it can handle transitions that do not emit symbols and time dependent transitions whose age is not affected by the firing of other transitions. This makes the Proxel method in general the method of choice for a given model. However, if the model is of the type that can be analyzed using adapted HMM, then these algorithms should be preferred, due to the superior runtime behavior.

5. CONCLUSION AND OUTLOOK

The paper documents the implementation of solution algorithms for Hidden non-Markovian Models. The algorithms implemented are an adaption of the original Hidden Markov Model solution algorithms. One crucial issue was the computation of the conditional state change probability in continuous time, which was solved in this paper. The implemented algorithms are however limited to Markov regenerative models that regenerate at every state change.

The algorithms are much faster than the Proxel solution proposed in earlier papers. However, the results are far less general. It is now possible to find the probability and generating path for a given system protocol. Using the current algorithm, significantly larger models and longer traces can be analyzed then using Proxels, and therefore more realistic problems are possible.

Future work includes the extension of the current implementation to further model classes. That will involve models where not all possible state changes generate output symbols, which will probably result in a mix of the adapted HMM algorithms and the Proxel solution method for computing the path probability between two consecutive symbol emissions.

Competitive algorithms for the analysis of HnMM will enable the solution of questions which cannot be solved using existing tools today. Protocols can be used to determine possible system behaviors that could likely have produced them. This can in turn help to determine whether a given system is functioning within the given specifications or, in the medical case, in what state of a disease a patient is in most likely. Eventually, HnMM can contribute to mining the wealth of data which is already being collected in current production systems and elsewhere, decreasing the necessary effort for obtaining valuable results.

REFERENCES

Bolch, G., Greiner, S., de Meer, H., Trivedi, K.S., 2006. *Queueing Networks and Markov Chains*. John Wiley & Sons, Hoboken, New York, 2nd edition.

Buchholz, R., 2008. *Improving the Efficiency of the Proxel Method by using Variable Time Steps*. Master's thesis, Otto-von-Guericke-University Magdeburg.

Fink, G.A., 2008. *Markov Models for Pattern recognition*. Springer-Verlag Berlin Heidelberg.

Horton, G., 2002. A new paradigm for the numerical simulation of stochastic petri nets with general firing times. *Proceedings of the European Simulation Symposium 2002*. SCS European Publishing House.

Isensee, C., Wickborn, F., Horton, G., 2006. Training Hidden Non-Markov Models. *Proceedings of 13th International Conference on ANALYTICAL and STOCHASTIC MODELLING TECHNIQUES and APPLICATIONS*, Bonn, Germany, pp. 105-110.

Krull, C., Horton, G., 2008. The Effect of Rare Events on the Evaluation and Decoding of Hidden non-Markovian Models. *Proceedings of the 7th International Workshop on Rare Event Simulation*, pp. 153-164.

Krull, C., Horton, G., 2009. HIDDEN NON-MARKOVIAN MODELS: FORMALIZATION AND SOLUTION APPROACHES, *Proceedings of 6th Vienna Conference on Mathematical Modelling*, Vienna, Austria, pp. 682-693.

Lazarova-Molnar, S. 2005. *The Proxel-Based Method: Formalisation, Analysis and Applications*. PhD thesis, Otto-von-Guericke-University Magdeburg. Germany.

Rabiner, L. R., 1989. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, vol. 77, pp. 257-286.

Wickborn, F., Isensee, C., Simon, T., Lazarova-Molnar, S., Horton, G., 2006. A New Approach for Computing Conditional Probabilities of General Stochastic Processes. *Proceedings of 39th Annual Simulation Symposium 2006*, Huntsville, USA, pp. 152-159.

AUTHORS BIOGRAPHY

CLAUDIA KRULL studied Computer Science at the Otto-von-Guericke-University Magdeburg, obtaining her Diploma in 2003. She spent an exchange year at the University of Wisconsin, Stevens Point, where she graduated in 2002. In April 2008 she successfully defended her PhD thesis at the Otto-von-Guericke-University Magdeburg.

GRAHAM HORTON studied Computer Science at the University of Erlangen, obtaining his Masters degree ("Diplom") in 1989. He obtained his PhD in Computer Science in 1991 and his "Habilitation" in 1998 at the same university, in the field of simulation. Since 2001, he is Professor for Simulation and Modelling at the Computer Science department of the University of Magdeburg.