TECHNISCHE UNIVERSITÄT
ILMENAU

Department of Computer Science and Automation
Systems and Software Engineering Group

Master Thesis

Thesis Title

Registration Date:	XX. January 20XX
Submission Date:	XX. June 20XX
Supervisor:	Prof. Responsible
Submitted by:	Author Name Matriculation Number XXXXX author-email@tu-ilmenau.de

Acknowledgments

I would like to thank ...

Abstract

The English Abstract...

Zusammenfassung

Eine deutsche Zusammenfassung.

Contents

1. Über das Template

Der Hauptteil kann aus den Kapiteln Aufgabenstellung, Konzept, Implementierung, Test, usw. je nach Arbeitsthema bestehen. innerhalb der `thesis.tex` wird der Grobaufbau der Arbeit definiert. Die Datei `header.tex` konfiguriert die Packages und alle Dokumenteigenschaften. Beide Dateien sollten bekannt sein und müssen bei Bedarf abgeändert bzw. erweitert werden. Die LaTeX-Definitionen und die eingebundenen Packages stellen einige Möglichkeiten bereit, diese sollen hier kurz vorgestellt werden.

1.1. Sprache

Dieses Dokument kann zur Erstellung von deutschen wie englischen Texten verwendet werden. Die Standardeinstellung ist Englisch, um zu Deutsch zu wechseln müssen in den Dateien `thesis.tex` und `header.tex` alle mit "‘LANGUAGE"’ markierten Zeilen entsprechend angepasst werden.

Ein bekanntes Problem ist die nicht immer fehlerfreie Unterstützung von deutschen Umlauten. Neben einigen möglichen Ursachen ist ein oft auftretendes Problem die nicht identische Kodierung der Textdateien innerhalb des Projekts. Treten Probleme nur im Bereich einer `*.tex` Datei auf muss die Kodierung dieser einen Datei an die der anderen angeglichen werden (z.B. mit Notepad++).

1.2. Abbildungen

Das Einbinden von Grafiken soll an Abbildung 1.1 gezeigt werden. Die Grafik selbst liegt entsprechend der Definition in der `header.tex` im Projekt-Unterordner "‘images"’ und sollte zur guten Darstellung als Vektorgrafik oder png-Bilddatei vorliegen.

Die Positionsdefinition "‘htb"’ beschreibt die automatische Positionierung in LaTeX in der bevorzugten Reihenfolge "‘here, top, buttom"’. Wenn man weiß was man tut und die Grafik

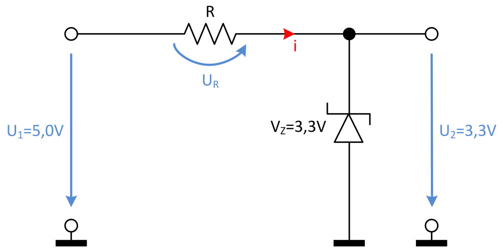


Figure 1.1. – Ein Bild als Beispiel

en

in jedem Fall genau an der gewählten Stelle eingebunden werden soll ist "‘h!"’ zu nutzen. Ein Warning wird generiert wenn diese Anforderung nicht erfüllt werden kann.

1.3. Verschiedenes

Eine Tabelle unter Nutzung des `booktabs` Package kann beispielsweise so aussehen:

Device	Teiler	Baudrate [Baud]
Modul	BR=4	125 000
Standard	BR=5	100 000
		Diff=25 000
Modul	Step=0x327	125 033
ExtendedLong	Step=0x326	124 878
		Diff=155

Table 1.1. – Beispiel Tabelle

asss hier
ng we-
u langen
hen er-

Quelltext kann mit Hilfe des `listings` Package direkt oder über eine Datei eingebunden werden:

```
    #define SCIBDH        SCI1BDH
    #define SCIBDL        SCI1BDL
    #define SCIC1         SCI1C1
    #define SCIC2         SCI1C2
     ...                   ...
```

Source Code 1.1 – `SCI.c` - SCI-Definitionen

```
if (gu8SCIDataFlag == 1)
{
    if (gu8SCIRx_i == 0)
    {   //Empfang des Start-Bytes
        if (gu8SCIData[0] > 0x0F)
        {   //ungültiges Start-Byte erhalten
            gu8SCIDataFlag = 0;
```

Source Code 1.2 – `wireless_uart.c` - Auszug

Ein Beispiel für eine nummerierte Tabelle mit Großbuchstaben und einer speziellen Formatierung:

(A) - Das Kapitel 1 trägt den Titel Über das Template[1] und beginnt auf Seite 1

(B) - 3 V liegt im Bereich zwischen 2,5 to 3,5 V. Die Stromstärke wird meist in der Einheit mA angegeben und auch eine einfache Zahl wie bspw. 65 536,128 lässt sich mit dem Package `siunitx` sicher darstellen.

[1] Die Nutzung von Nameref ist in vielen Situationen nützlich

2. Introduction

A correct design for a system is a crucial step to reach a satisfactory result. Building a model is essential to analyze and evaluate the different possibilities in an early stage, or to prove the correctness of the solution in diverse scenarios. For this purpose, there are several models [ZPK00] that can be used, with different expressiveness and properties. If the model is complex enough, an analytical approach may not be possible due to the need of a complete state space exploration and the state explosion limitation [Val98]. To overcome this problem, simulation methods [FJMRM10] can provide an acceptable solution but they introduce a new issue: rare-events [GLFH01, Hee97]. If the simulation needs to evaluate a measure [Haa02] based on an event that is very unlikely to happen (e.g. 10^{-6}), unless special considerations are taken the simulation will rarely go through this event, therefore a good confidence level will be highly time-consuming to achieve. The goal of the current work is to design, implement and evaluate a simulation algorithm, which will be able to overcome the rare-event limitation without requiring a deep understanding of it, nor a high level analysis of the model. Moreover, it introduces a new concept of how to deal with the simulation execution, avoiding random decisions to achieve an execution as close as possible to an analysis...

3. Background Research on Rare-Event Simulation

In this chapter background research on rare-event simulation is reviewed, including the models and the simulation algorithms. First, a descriptive presentation of the different models is conducted, where both Markovian and non-Markovian examples are given...

3.1. Models, Markovian and Non-Markovian

In the current work different types of models are considered, where one of the main classifications is the requirement of memory or not. As introduced, a model without memory requirement is considered Markovian, and by opposition one that requires memory is non-Markovian. A Markovian model is a stochastic model that changes based on a random factor, and the future state depends only on the current one, not on previous states or events leading to it. A non-Markovian model does not fulfill this property, therefore only knowing the current state is not enough to completely describe the current situation. In a non-Markovian model the followed path of states and the events that lead to them are relevant as well.

3.1.1. Markov Chain

A Markov chain model is a static Markovian model, composed of states and transitions, where each state has incoming and outgoing transitions. Each of the outgoing transitions has a probability of being followed, and by definition, this probability only depends on the current state, not on previous visited states. Therefore, if a simulation is run over a Markov chain and at two different points in time the same state is reached, there is no difference between the execution status at those points, and all the following possible paths have the same probability in both scenarios. A simple Markov chain with three states and their transitions is presented in Figure 3.1. The values shown over the transitions correspond to

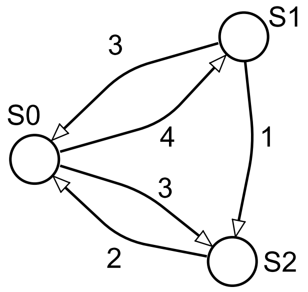


Figure 3.1. – Simple Markov Chain Model

their rates, and the probability is calculated with this value over the sum of all outgoing transitions of the state.

The distribution of the time remaining in a state is memoryless, which is modeled with an exponential distribution: $f(x) = \lambda e^{-\lambda x}$, $F(x) = 1 - e^{-\lambda x}$ $(x \geq 0)$ [Haa02]. The average firing time for a transition with rate λ is λ^{-1}. Moreover, in a state with t outgoing transitions the average time spent is:

$$Average\ Time = \frac{1}{\sum\limits_{i=0}^{t}(rate_i)}$$

In the example presented in Figure 3.1 the transition matrix is:

$$Q = \begin{pmatrix} -7 & 4 & 3 \\ 3 & -4 & 1 \\ 2 & 0 & -2 \end{pmatrix}$$

3.2. Monte Carlo Simulation Algorithm

A Monte Carlo simulation algorithm follows a standard Monte Carlo method [DPK11]. Whenever a decision needs to be made, a random result is generated based on the probability of each of the possible choices. Considering a Markov chain as the underlying model for a simulation executing a Monte Carlo algorithm, given the current state there are some outgoing transitions, each of them with a probability of being executed next...

```
// Initialization, set current state to the initial state
state *current_state = initial_state;

// Main simulation loop
while ( ! stop_condition_reached() )
{
  int tran, selected_tran;
  // Array with all the outgoing transitions rates from the current state
  double trans_rates[] = get_out_transitions_rates( current_state );
  int trans_count = get_out_transitions_count( current_state );

  // Calculate the sum of all the outgoing transitions rates
  double total_rates = 0;
  for( tran = 0; tran < trans_count; tran++)
    total_rates += trans_rates[ tran ];

  // Calculate the average time spent in the state
  double time_spent = 1.0 / total_rates;
  // Evaluate measures defined over the current state
  evaluate_measures( current_state, time_spent );

  // Randomly select one outgoing transitions, based on their rates
  double random_value = generate_random_value( 0, total_rates );
  for( tran = 0; tran < trans_count; tran++)
  {
    if ( random_value < trans_rates[ tran ] )
    {
      selected_tran = tran;
      break;
    }
    random_value -= trans_rates[ tran ];
  }

  // Execute the selected transition, and update the current state
  current_state = fire_transition( current_state, selected_tran );
}

```

Source Code 3.1 – `Abstract Representation` - Monte Carlo Algorithm

4. Design of a Multi-Trajectory Simulation Algorithm

This chapter introduces a design for a multi-trajectory simulation algorithm and provides explanations and examples of its main steps. The strength of this algorithm lies in its ability to run simulations within complex models, either with or without the presence of rare-events, and its adaptability to reach a wide coverage of the state space with efficiency in different types of models...

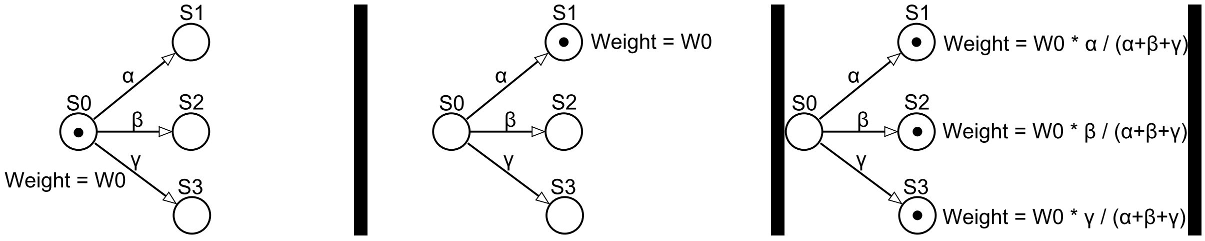
4.1. Introduction to Multi-Trajectory

Multi-trajectory simulation algorithm is based on particles, where a particle represents one possible trajectory of the simulation, and carries a weight that indicates the probability of that trajectory to happen. At any given time there is a collection of particles ready to be executed. Each particle is associated with one state of the model, and the execution of it implies a change in that state by following one or more of its possible transitions.

The core idea of the algorithm can be described as a loop of three steps:

1. Selection of the next particle...
2. Execution of the particle, which consists in choosing between:
   a) Move: Randomly select...
   b) Split: Create one new...
3. Insertion of the particle...

Against expectations, after the implementation of these strategies the results obtained by the executions were incorrect. Different attempts were made to overcome this error without success. An analysis of it with a numerical example is presented in Section 3.1.1. Due to this limitation a solution with a fair execution over the particles is chosen, using the definition


S1
α
S0
β
S2
γ
Weight = W0
S3
S1
Weight = W0
α
S0
β
S2
γ
S3
S1
Weight = W0 * α / (α+β+γ)
α
S0
β
S2
Weight = W0 * β / (α+β+γ)
γ
S3
Weight = W0 * γ / (α+β+γ)

Initial

Move, with probability $\alpha/(\alpha+\beta+\gamma)$

Split

Figure 4.1. – Move or Split Example

of a cycle as the execution of all particles in the pool and the restriction of not execute a particle twice in a cycle.

Only Exponential Transitions Enabled

If there are only exponential transitions enabled, ... and the average time spent in the state is equal to the inverse of the sum of all those weighting factors.

$$Time\ Spent = \frac{1}{\sum_{i=0}^{n}(\frac{enabled_i}{delay_i})}$$

$$Weighting\ Factor[i] = \begin{cases} Immediate\ Transition & = 0 \\ Exponential\ Transition & = \dfrac{enabled_i}{delay_i} \\ Deterministic\ Transition & = 0 \end{cases}$$

5. Implementation

In this chapter the implementation of a multi-trajectory simulation algorithm is presented...

6. Results and Comparative Analysis

In this chapter an analysis and evaluation of the TimeNET implementation of the multi-trajectory algorithm is presented. First, an analysis of its parameters is conducted and for this purpose a model is designed. Execution of this model with all possible values of different multi-trajectory parameters are shown, and the best combination of parameters is found. Afterwards, a comparative analysis of the implementation with the other three TimeNET implemented simulation algorithms is presented. This analysis is based on a previous research conducted by the author of the current work, *Rare-Event Algorithms Analysis and Simulation* [Can14]. Its goals were to evaluate RESTART TimeNET implementation and the implemented heuristic for its importance function. The multi-trajectory simulation results are compared together with the results obtained from the earlier work, followed by an analysis of how variations of the model properties exert different impact on the execution time of the algorithms. Finally, a non-Markovian model is analyzed and different simulations executed on it are presented.

6.1. Multi-Trajectory Results

In this section the results from the multi-trajectory simulations are compared with the results from the previous work. The multi-trajectory algorithm has been executed with the default parameters over all variations of the models, but only the most significant results are presented here.

Model Variations					**Standard Simulation Results**			
Mod	Pla	Tok	Del	Math Res	CPU Time	Overall T.	Value	Error
1	5	1	10	9.00E-05	0:00:13	0:00:03	9.20E-05	2.22%
1	5	1	100	9.90E-09	2:41:08	0:28:59	9.19E-09	7.17%
1	5	1	1000	9.99E-13		» 24 hs		
1	5	3	10	2.70E-04	0:00:11	0:00:03	2.71E-04	0.37%
1	5	3	100	2.97E-08	2:45:31	0:29:44	2.83E-08	4.71%
1	5	3	1000	3.00E-12		» 24 hs		
1	5	10	10	9.00E-04	0:00:29	0:00:06	9.23E-04	2.56%
1	5	10	100	9.90E-08	2:48:56	0:32:47	9.67E-08	2.32%
1	5	10	1000	9.99E-12		» 24 hs		
2	5	1	10	6.83E-05	0:00:30	0:00:06	6.92E-05	1.32%
2	5	1	100	9.61E-09	2:41:24	0:28:34	8.95E-09	6.87%
2	5	1	1000	9.96E-13		» 24 hs		
2	5	3	10	2.05E-04	0:00:11	0:00:03	2.04E-04	0.49%
2	5	3	100	2.88E-08	2:49:14	0:29:59	2.63E-08	8.68%
2	5	3	1000	2.99E-12		» 24 hs		
2	5	10	10	6.83E-04	0:00:17	0:00:04	6.79E-04	0.59%
2	5	10	100	9.61E-08	3:09:25	0:34:05	9.32E-08	3.02%
2	5	10	1000	9.96E-12		» 24 hs		

Table 6.1. – Standard Simulation Results [Can14]

Model Variations					**RESTART Simulation Results**			
Mod	Pla	Tok	Del	Math Res	CPU Time	Overall T.	Value	Error
1	5	1	10	9.00E-05	0:00:15	0:00:04	8.87E-05	1.44%
1	5	1	100	9.90E-09	0:02:20	0:00:25	9.74E-09	1.62%
1	5	1	1000	9.99E-13	0:47:54	0:08:01	1.10E-12	10.11%
1	5	3	10	2.70E-04	0:00:17	0:00:05	3.45E-04	27.78%
1	5	3	100	2.97E-08	0:03:27	0:00:35	3.06E-08	3.03%
1	5	3	1000	3.00E-12	0:59:34	0:11:24	3.00E-12	0.00%
1	5	10	10	9.00E-04	0:00:22	0:00:04	1.06E-03	17.78%
1	5	10	100	9.90E-08	0:13:53	0:02:21	1.17E-07	18.18%
1	5	10	1000	9.99E-12	2:20:50	0:23:54	9.84E-12	1.50%
2	5	1	10	6.83E-05	0:00:17	0:00:05	6.74E-05	1.32%
2	5	1	100	9.61E-09	0:00:35	0:00:07	9.75E-09	1.46%
2	5	1	1000	9.96E-13	0:21:25	0:03:38	9.52E-13	4.42%
2	5	3	10	2.05E-04	0:00:17	0:00:04	2.14E-04	4.39%
2	5	3	100	2.88E-08	0:00:47	0:00:10	3.05E-08	5.90%
2	5	3	1000	2.99E-12	0:36:54	0:06:19	3.49E-12	16.72%
2	5	10	10	6.83E-04	0:00:17	0:00:04	7.18E-04	5.12%
2	5	10	100	9.61E-08	0:01:15	0:00:13	9.51E-08	1.04%
2	5	10	1000	9.96E-12	2:16:35	0:23:21	9.69E-12	2.71%

Table 6.2. – RESTART Simulation Results [Can14]

7. Conclusions and Further Work

The goal of the current work was to design, implement and evaluate a new simulation algorithm capable of ...

A. Appendix

This appendix presents the details of the stationary analysis and multi-trajectory simulation executions of the model introduced in Figure 4.1 (Section 4.1, page 12).

Interconnected Model Analysis Output

Interconnected Model Analysis Output

```
Analysis Output:

STEADY STATE SOLUTION OF NET Interconnected_Model


STRUCTURAL ANALYSIS ...


GENERATING THE REDUCED REACHABILITY GRAPH USING TIME EFFICIENT ALGORITHM ...
The reduced reachability graph of this eDSPN
cannot be generated because of lacking main memory.

ERROR occurred while derive_SMC execution.
SOLUTION OF MODEL Interconnected_Model FAILED.

Removing temporary files.


-----------------------------------------------------------------------------------------
                               Analysis error, see above.
-----------------------------------------------------------------------------------------
```

CD Structure

A soft version of the current work is submitted. Following, the CD structure is presented.

```
Root
├── Latex_Project                          The latex project of the current work
│   │
│   └── ...
│
├── Models                               All the models used in the current work
│   │
│   ├── Model_1_variations
│   │   │
│   │   └── ...                            All 9 variations for model in Fig 5.2
│   │
│   ├── Model_2_variations
│   │   │
│   │   └── ...                            All 9 variations for model in Fig 5.3
│   │
│   ├── Interconnected_Model.xml                                Model in Fig 5.1
│   │
│   ├── Interconnected_Model__one_token_variation.xml
│   │
│   ├── Non_Markovian_Model_GSM_R.xml                           Model in Fig 5.4
│   │
│   └── Non_Markovian_Model_GSM_R__two_tokens_variation.xml
│
├── Multi-Trajectory_Kernel_Source            Core algorithm implementation
│   │
│   ├── sim_mt_kernel.c                               Main loop and evaluation
│   │
│   └── sim_mt_particle.c                 Particle definition and structures
│
├── TimeNET                                           TimeNET source projects
│   │
│   └── ...                               including all files added by this work
│
└── Multi-Trajectory Rare-Event Simulation of Stochastic Petri Nets.pdf
```

List of Tables

List of Figures

List of Source Code

Bibliography

[Can14] Canabal, Andres: *Rare Event Algorithms Analysis and Simulation.* 2014 6, 6.1, 6.2, 7

[CS92] Campos, J. ; Silva, M.: Structural Techniques and Performance Bounds of Stochastic Petri Net models. In: Rozenberg, G. (Hrsg.): *Advances in Petri Nets 1992* Bd. 609. Springer Verlag, 1992, S. 352–391

[DPK11] Dirk P. Kroese, Zdravko I. B. Thomas Taimre T. Thomas Taimre: *Handbook of Monte Carlo Methods.* John Wiley and Sons, 2011 3.2

[FJMRM10] Faulin, Javier (Hrsg.) ; Juan, Angel A. (Hrsg.) ; Martorell, Sebastián (Hrsg.) ; Ramírez-Márquez, José-Emmanuel (Hrsg.): *Simulation methods for reliability and availability of complex systems.* Springer, 2010. – xvii + 315 S. 2

[GLFH01] Görg, Carmelita ; Lamers, Eugen ; Fuss, Oliver ; Heegaard, Poul: Rare Event Simulation / Computer Systems and Telematics, Norwegian Institute of Technology. 2001 (COST 257). – Forschungsbericht 2

[GZH94] German, Reinhard ; Zimmermann, Christian Kellingand A. ; Hommel, Günter: TimeNET - A Toolkit for Evaluating Non-Markovian Stochastic Petri Nets / Technische Universität Berlin. 1994 (19). – Technischer Bericht des Fachbereichs Informatik

[Haa02] Haas, Peter J.: *Stochastic Petri Nets: Modelling, Stability, Simulation.* Springer Verlag, 2002 (Springer Series in Operations Research) 2, 3.1.1

[Hee97] Heegaard, Poul E.: Speedup Simulation Techniques. In: *Proc. Workshop on Rare Event Simulation.* Aachen, Germany, 1997, S. 28–29 2

[Hel09] Hellfritsch, Christoph: TimeNet-Examples of Extended Deterministic and Stochastic Petri Nets. (2009)

[Kel95] Kelling, Christian: Rare Event Simulation with RESTART in a Petri Net Modeling Environment. In: *Proc. of the European Simulation Symposium*. Erlangen, 1995, S. 370–374

[RR98] *Kapitel* The state explosion problem. In: Reisig, Wolfgang ; Rozenberg, Grzegorz: *Lectures on petri nets i: basic models*. Bd. 1491. Springer Science & Business Media, 1998, S. 429–528

[Val98] Valmari, Antti: The state explosion problem. In: *Lectures on Petri nets I: Basic models*. Springer, 1998, S. 429–528 2

[VAVA94] Villén-Altamirano, Manuel ; Villén-Altamirano, Jose: RESTART: A Straightforward Method for Fast Simulation of Rare Events. In: *Proc. Winter Simulation Conference*, 1994, S. 282–289

[WR11] White, K P. ; Robinson, Stewart: Initial Transient Period in Steady-State Systems. In: *Wiley Encyclopedia of Operations Research and Management Science* (2011)

[Zim08] Zimmermann, Armin: *Stochastic Discrete Event Systems: Modeling, Evaluation, Applications*. Springer, 2008

[ZK07] Zimmermann, Armin ; Knoke, Michael: TimeNET 4.0 User Manual / Technische Universität Berlin. 2007 (13). – Technical Report

[ZKHH06] Zimmermann, A. ; Knoke, M. ; Huck, A. ; Hommel, G.: Towards Version 4.0 of TimeNET. In: *13th GI/ITG Conference on Measurement, Modeling, and Evaluation of Computer and Communication Systems (MMB 2006)*, 2006, S. 477–480

[ZM12] Zimmermann, Armin ; Maciel, Paulo: Importance Function Derivation for RESTART Simulations of Petri Nets. In: *9th Int. Workshop on Rare Event Simulation (RESIM 2012)*. Trondheim, Norway, Juni 2012, S. 8–15

[ZPK00] Zeigler, Bernhard P. ; Praehofer, Herbert ; Kim, Tag G.: *Theory of Modeling and Simulation*. 2nd. London : Academic Press, 2000 2

Todo list

Declaration Of Authorship

I, Author Name, Matriculation Number XXXXX, hereby declare that my Master Thesis with the Title

Thesis Title

was produced by me, independently and without using any other references than those mentioned. This work is submitted in completion of the requirements for the Master Degree Research in Computer and Systems Engineering in Ilmenau University of Technology, and I did not publish nor submit this thesis to another university examination board.

Ilmenau, XX.XX.20XX

Author Name