

Asset Analytics

Performance and Safety Management

Series Editors: Ajit Kumar Verma · P. K. Kapur · Uday Kumar

Reggie Davidrajuh

Petri Nets for Modeling of Large Discrete Systems



Springer

Asset Analytics

Performance and Safety Management

Series Editors

Ajit Kumar Verma, Western Norway University of Applied Sciences, Haugesund, Rogaland Fylke, Norway

P. K. Kapur, Centre for Interdisciplinary Research, Amity University, Noida, India

Uday Kumar, Division of Operation and Maintenance Engineering, Luleå University of Technology, Luleå, Sweden

The main aim of this book series is to provide a floor for researchers, industries, asset managers, government policy makers and infrastructure operators to cooperate and collaborate among themselves to improve the performance and safety of the assets with maximum return on assets and improved utilization for the benefit of society and the environment.

Assets can be defined as any resource that will create value to the business. Assets include physical (railway, road, buildings, industrial etc.), human, and intangible assets (software, data etc.). The scope of the book series will be but not limited to:

- Optimization, modelling and analysis of assets
- Application of RAMS to the system of systems
- Interdisciplinary and multidisciplinary research to deal with sustainability issues
- Application of advanced analytics for improvement of systems
- Application of computational intelligence, IT and software systems for decisions
- Interdisciplinary approach to performance management
- Integrated approach to system efficiency and effectiveness
- Life cycle management of the assets
- Integrated risk, hazard, vulnerability analysis and assurance management
- Adaptability of the systems to the usage and environment
- Integration of data-information-knowledge for decision support
- Production rate enhancement with best practices
- Optimization of renewable and non-renewable energy resources

More information about this series at <http://www.springer.com/series/15776>

Reggie Davidrajuh

Petri Nets for Modeling of Large Discrete Systems

Reggie Davidrajuh
Department of Electrical Engineering
and Computer Science
University of Stavanger
Stavanger, Norway

ISSN 2522-5162

Asset Analytics

ISBN 978-981-16-5202-8

<https://doi.org/10.1007/978-981-16-5203-5>

ISSN 2522-5170 (electronic)

ISBN 978-981-16-5203-5 (eBook)

© The Editor(s) (if applicable) and The Author(s), under exclusive license to Springer Nature Singapore Pte Ltd. 2021

This work is subject to copyright. All rights are solely and exclusively licensed by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, expressed or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

This Springer imprint is published by the registered company Springer Nature Singapore Pte Ltd.
The registered company address is: 152 Beach Road, #21-01/04 Gateway East, Singapore 189721, Singapore

*This book is dedicated to my family:
my wife Ruglin
and
my daughter Ada*

Foreword

It gives me great pleasure to write a “foreword” for this wonderful book on Petri nets and an innovative approach to handling huge Petri net models using distributed computing so as to enable them to be fast enough for real-time control applications. Petri nets are handy for modeling discrete event systems. However, they suffer from some weaknesses, e.g., massive size, huge state space, and slow simulation. Due to the enormous state space, model checking a Petri net is difficult. Also, Petri nets are difficult to be used for real-time applications due to the slowness in simulation. Thus, this book suggests modular Petri nets as a way of overcoming these difficulties.

In modular Petri nets, modules are designed, developed, and run independently. Also, the modules communicate with each other via inter-modular connectors. Hence, the bottom-up approach is suggested (starting with the modules and then combining them after thorough testing) for developing newer Petri net models. However, many Petri net models of real-life systems exist (legacy models), which are enormous and non-modular. These legacy models cannot be discarded as large amounts of time, effort, and money were spent developing these legacy models. In this case, the top-down approach is the only solution, which starts with the existing legacy models, decomposing them into modules, and then joining them together after testing them individually. This book presents both approaches centering around modules (known as “Petri modules”). Petri modules are well defined for inter-modular collaboration.

This book aims to introduce a methodology in which Petri nets are moved to a new level. In this new level, large Petri net models are made of Petri modules that are independent and run on different computers. Also, Petri modules communicate with each other using the inter-modular components (e.g., TCP/IP sockets). Thereby, the compact Petri modules run faster and, thus, become suitable for supervisory control of real-time systems.

This book focuses on Petri modules. This book presents the literature study on modular Petri nets and definitions for the newer Petri modules. Also, algorithms for extracting Petri modules and algorithms for connecting Petri modules and applications are given in this book. The ideas and algorithms proposed in this book are

implemented in the software General-purpose Petri net Simulator (GPenSIM, developed by the author of this book). Hence, real-life discrete event systems could be modeled, analyzed, and performance-optimized with GPenSIM.

I congratulate Prof. Davidrajuh for bringing out this highly technical book in a very lucid manner and welcome this book as a valuable addition to the Springer series on “Asset Analytics”.

Ajit Kumar Verma
Professor (Technical Safety) and Series Editor,
Asset Analytics-Performance and Safety
Management, Western Norway University
of Applied Sciences, Haugesund, Norway

Preface

Cyber-physical systems involve hardware and software that are highly interconnected and complex. Unexpected failures in these systems can cause material damages, cost a lot of money and reputation, and can risk human lives too. The correct way of avoiding unexpected failures is to make a mathematical model of the system and to perform exhaustive analysis on it. Also, mathematical models are inevitable for performance analysis of the systems (e.g., finding throughput and bottlenecks) and to plan for extensions (e.g., measuring cost over performance factor).

Petri net, since its inception in 1962, has been used for modeling and performance analysis of discrete event systems. In the 1970s and 80s, Petri net was tried out on modeling smaller discrete manufacturing systems like flexible manufacturing systems with a handful of CNC machines and robots. Very soon, it became apparent that even for smaller systems, the Petri net models become huge. Also, the state spaces generated from the models become too big to handle (the problem of state explosion). Of course, for a large Petri net model of a real-life discrete system, the state space is enormous, and analysis of it is not a possibility.

Also, simulations take too much time, as the simulation of a Petri net means tracking the movement of tokens as they pass through places and transitions. For a large Petri net with hundreds of places and transitions, monitoring the movements of all those tokens makes the simulations slower. The slow simulations also make Petri net not suitable for real-time control. If faster execution of Petri net is possible, then Petri net could become an ideal candidate for supervisory control of real-time systems.

Finally, researchers and engineers agree that model checking is the only systematic way of finding and fixing errors in industrial systems before they go into production. For Petri net models of discrete event system, model checking verifies whether the state space satisfies a given property specification. Here too, due to the enormous size of the state space, model checking is either difficult or not possible at all.

Some Petri net slicing methods are proposed in the literature to make the Petri net smaller so that the resulting state space also becomes smaller. However, for real-life discrete event systems (especially for manufacturing systems), the proposed Petri net slicing methods fail to produce any slice at all, as the systems are strongly connected.

In some cases, the slicing methods provide slices; however, the resulting state spaces remain as huge as before.

For all these problems mentioned above (massive Petri net models resulting in enormous state spaces, slowness in simulation, and unsuitability for real-time control), this book presents the modular Petri net as the solution. Although in the literature, modular Petri net has been proposed as a solution, this is the first time modular Petri net is given a full focus. In this book, starting with a historical perspective (literature study), a clear and well-defined Petri module is designed and analyzed. Also, the toolbox of the General-purpose Petri net Simulator (GPenSIM) is extended to support modules, so that modular Petri net models of real-life discrete event systems can be developed and tested.

It must be emphasized that the modular Petri net presented in this book is designed and developed as a general solution to many problems that are discussed above. In particular, the theory, design, and implementation of modular Petri net presented in this book provide four benefits:

1. **Fast and Flexible model development:** Modelers can independently design, develop, and test modules that make up a modular Petri net model.
2. **Faster simulation:** The modules can be hosted on different computers so that they can run faster, and these modules can communicate through TCP/IP sockets, exchanging tokens (messages, status, and control policies as tokens).
3. **Real-time control:** A module can receive data from sensors that are attached to the computer that is hosting the module. Also, a module can control actuators that are also attached to the computer.
4. **Departmental cost calculation:** Since GPenSIM supports cost calculation, activity-based departmental cost calculation is possible as modules represent different departments of a manufacturing system and the transitions inside a module represent different activities of a department.

The Structure of This Book

This book consists of four parts. Part-I “Petri nets and GPenSIM” is the introductory part consisting of four chapters:

- Chapter 1 “Introduction to Petri nets” starts with formal definitions of the key terms used in this book.
- Chapter 2 “Introduction to GPenSIM” introduces the software known as the General-purpose Petri net Simulator (GPenSIM). Only the basics of GPenSIM are given in this chapter. GPenSIM defines a Petri net language on the MATLAB platform. GPenSIM is also a simulator with which Petri net models can be developed, simulated, and analyzed. GPenSIM is developed by the author of this book.
- Chapter 3 “Models of Real-Life Systems” presents some Petri net models of typical manufacturing systems. The reason behind this chapter is to clearly show

the major difficulties in using Petri nets for modeling of real-life discrete event systems generally and manufacturing systems especially.

- Chapter 4 “GPenSIM for Monolithic Petri nets” presents the software GPenSIM, focusing on the transitions and their visibility. Chapter 4 is essential as the knowledge of how GPenSIM supported the development of monolithic Petri nets will help understand the extensions that are added to GPenSIM to support modules.

Part-II “Design of Modular Petri Nets” consists of the following four chapters:

- Chapter 5 “Literature Review on Modular Petri Nets” presents an extensive literature study on modular Petri nets. The conclusions drawn from the literature study are the basis for the new design of Petri net modules presented in this book.
- With the conclusion from the literature study, Chap. 6 “Toward Developing a New Modular Petri net” takes the first attempt to design a modular Petri net. This chapter introduces the main elements of a modular Petri net (such as Petri modules and Inter-Modular Connectors (IMC)) and also the input and out ports (IO ports) of a module.
- Chapter 7 “Design of a New Modular Petri Nets” presents complete details about modular Petri nets, providing the formal definitions to all the elements. An application example on modular Petri net is also given.
- Chapter 8 “GPenSIM support for Petri Modules” presents the extension to GPenSIM to support developing modular Petri nets. This chapter discusses the programming constructs provided by GPenSIM for the development.

Part-III “Legacy Petri Nets” is about transforming legacy models into Modular Petri nets. Part-III consists of the following two chapters:

- Chapter 9 “Module Extraction”: Modularization of a monolithic model starts with identifying the modules (or segments or groups of elements) of the model that can decompose the model into a set of connected independent modules. Once the modules are identified, how can the modules be carved out of the model so that these modules individually adhere to the formal definition of modules? The module extraction algorithm is described in this chapter. Also, this chapter presents an analysis of the preservation of structural properties by modularization.
- Chapter 10 “Activity-Oriented Petri nets (AOPN)”: Sometimes, it may not be possible to modularize a legacy model, due to its crisscrossing connections. For example, the modularization techniques described in Chap. 9 may not be successful. In this case, AOPN could be a remedy.

The final part (Part-IV) is about “Collaborating Modules”, allowing the modules to collaborate to perform larger jobs. Part-IV consists of the following five chapters:

- Chapter 11 “Discrete Systems as Petri Modules” is about putting the modules together in a modular Petri net. This chapter consists of three groups of sections. In the first group of sections, nine blocks are introduced as the basic building blocks of a discrete system. The second group of sections introduces five matrices, such as adjacency, reachability, Rader’s, connection, and component matrices. The third group of sections introduces the spanning tree and connected components

of a graph. All the material introduced in this chapter will be used in the newer algorithms developed in Chap. 12.

- Chapter 12 “Algorithms for Modular Connectivity” presents five new algorithms for analyzing the network connectivity in the modular Petri net models. The algorithms are designed with Industry 4.0 in mind; this means the modules are intelligent enough to choose with whom they may communicate, in case some of the modules are not collaborating.
- Chapter 13 “Model Checking for Collaborativeness” focuses on the property of collaborativeness. Collaborativeness is a property of a module, indicating whether the module is willing or be able to participate in collaboration with the other modules. This chapter applies the theories and techniques developed in this book to solve the problem of model checking for collaborativeness. The literature study suggests the use of external tools, off-line, for model checking of collaborativeness. However, Chap. 13 proposes using a novel online technique that is tailor-made for the GPenSIM environment.
- Chapter 14 “Generic Petri Module” presents a generic three-layered architecture for Petri modules that can participate in collaboration while performing some useful functions on their own. This chapter starts with a crude structure for Petri modules and discusses the necessity of using colored Petri net. Finally, the generic architecture is presented.

Companion Website

Companion Website for this book is

<http://www.davidrajuh.net/gpensim/book-II/>.

This Companion website presents the GPenSIM code for some of the examples given in this book.

Acknowledgement

The book is a result of my teaching and research at two institutions. I work as a Professor of informatics at the Department of Electrical Engineering and Computer Science (IDE), University of Stavanger. I give two courses for master students in Computer Science, “DAT530 Discrete Simulations” and “DAT600 Algorithm Theory”. My experiences from these two courses are instrumental in writing this book. I am also a Visiting Professor at the Department of Engineering Processes Automation and Integrated Manufacturing Systems (RMT2), Silesian University of Technology, Gliwice, Poland. In Gliwice, I give a variety of short courses to master students in Mechanical Engineering. A part of the book is written during my stays in Gliwice, benefiting from my discussions with my colleagues there. My sincere

thanks go to these two institutions (IDE and RMT2) for offering me ample time and research facilities to complete this book.

Dr. Damian Krenczyk (Silesian University of Technology, Gliwice, Poland), has tirelessly read the full draft version of the book and gave very useful suggestions for improvement. I am indebted to Dr. Krenczyk for all the help and advice I received from him.

From the publisher Springer's side, I would like to thank the "Asset Analytics" series editor Prof. Ajit Verma Kumar, for encouraging me to submit the book to the series and for the speedy and qualitative review process. Prof. Ajit Verma was also kind enough to write the foreword of this book. My thanks also go to Ms. Kamiya Khatter—the receiving editor at Springer Nature India. Ms. Kamiya patiently answered my numerous emails to her.

I am also thankful to my wife Ruglin and my daughter Ada for tolerating my frequent absence in their daily lives.

Stavanger, Norway
May 2021

Reggie Davidrajuh

Contents

Part I Petri Nets and GPenSIM

1	Introduction to Petri Nets	3
1.1	Petri Nets	3
1.2	Formal Definitions	4
1.2.1	P/T Petri Nets	4
1.2.2	State Space (<i>aka</i> Reachability Graph)	6
1.2.3	State Equation and the Incidence Matrix	6
1.2.4	Structural Property: P-Invariant	8
1.2.5	Structural Property: T-Invariant	9
1.2.6	Timed Petri Nets	9
1.2.7	Strongly Connected Event Graphs (SCEGs)	10
	References	12
2	Introduction to GPenSIM	15
2.1	Time in GPenSIM	15
2.2	Automaticity and Virtual Tokens	16
2.3	Layered Architecture	17
2.4	Modeling with GPenSIM: The Basics	18
2.4.1	Separating the Static and Dynamic Details	19
2.4.2	Processor Files	20
2.4.3	Global Info	21
2.4.4	Integrating with MATLAB Environment	21
2.5	Creating a Simple Petri Net Model with GPenSIM	22
2.5.1	Creating a Simple Petri Net Model: An Example	22
2.6	GPenSIM and Its Applications	25
	References	26
3	Models of Real-Life Systems	29
3.1	Model-1: Flexible Manufacturing System	29
3.1.1	The Petri Net Model	30
3.1.2	Analysis of the Petri Net Model: P-Invariants	32
3.1.3	Analysis of the Petri Net Model: T-Invariants	33

3.1.4	Analysis of the Petri Net Model: State Space	33
3.2	Model-2: Circular AGVs	36
3.2.1	The Petri Net Model of Circular AVG Transport System	37
3.2.2	Analysis of the Petri Net Model: State Space	38
3.2.3	Analysis of the Petri Net Model: P-Invariants	41
3.2.4	Analysis of the Petri Net Model: T-Invariants	41
3.3	Model-3: Cyclic Processes	42
3.3.1	Petri Net Model of the Cyclic Processes	43
3.3.2	Analysis of the Petri Net Model: State Space	43
3.3.3	Analysis of the Petri Net Model: P-Invariants	47
3.3.4	Analysis of the Petri Net Model: T-Invariants	47
3.4	The Problem with Petri Net Models	47
3.5	Research Goals of This Book	48
	References	49
4	GPenSIM for Monolithic Petri Nets	51
4.1	Revisiting the Files in GPenSIM	51
4.2	Transitions and Their Visibility	53
4.3	Visibility in Monolithic Petri Net	53
4.3.1	Global Visibility	53
4.3.2	Private Visibility	54
4.4	GPenSIM Application-I: Model Checking	59
4.4.1	The Soda Vending Machine	59
4.4.2	The Petri Net Model	60
4.4.3	GPenSIM Implementation	60
4.4.4	Model Checking the Petri Net	63
4.5	GPenSIM Application-II: Performance Evaluation	65
4.5.1	Performance Evaluation of an FMS	66
4.5.2	GPenSIM Files for Simulation	66
4.5.3	The Simulation Results	68
4.5.4	GPenSIM for Performance Evaluation	71
4.6	Colored Petri Nets with GPenSIM	72
	References	73
 Part II Design of Modular Petri Nets		
5	Literature Review on Modular Petri Nets	77
5.1	First-Generation Works: Ease of Modeling	77
5.2	Second-Generation Works: Analysis	78
5.3	Third-Generation Works: Applications and Tools	80
5.4	Fourth-Generation Works: Independent Modules for Modeling Smart Manufacturing	82
5.5	Research Gap: Conclusions of Literature Study	83
	References	84

6	Toward Developing a New Modular Petri Net	87
6.1	Toward Independent Development of Modules	87
6.2	Design of Modular Petri Nets	90
6.2.1	Composition of Modular Petri Nets	91
6.2.2	Transitions in Modular Petri Nets	91
6.2.3	Visibility of Transitions in a Modular Petri Net	92
	References	93
7	Design of a New Modular Petri Nets	95
7.1	Petri Module	95
7.2	Inter-Modular Connectors	96
7.3	Formal Definitions for the New Entities	97
7.3.1	Formal Definition of Petri Module	97
7.3.2	Formal Definition of Inter-Modular Connector	98
7.3.3	Formal Definition of Modular Petri Net	98
7.4	Application Example: A Modular Petri Net	99
7.4.1	The Problem: Computing a Quadratic Function	99
7.4.2	Petri Module of a Communicating Agent	99
7.4.3	Modular Petri Net Model	101
7.5	Chapter Summary	103
7.5.1	Technical Notes on TCP/IP Based Communication	105
	References	105
8	GPenSIM Support for Petri Modules	107
8.1	Modular Model Building	107
8.1.1	Declaring Modules with the IO Ports	108
8.2	Modular Processor Files	108
8.3	Modular Visibility of Transitions	110
8.3.1	Modular Visibility: An Example	111
8.3.2	Modular Visibility: Summary	112
8.4	Developing Modules with GPenSIM	113
8.4.1	The Adder Module	113
8.4.2	The Multiplier Module	115
8.4.3	The Client Module	116
8.4.4	Putting All Together	116
8.4.5	Technical Note on Independent Module Development	117
	References	119

Part III Legacy Petri Nets

9	Module Extraction	123
9.1	Rules of Interfacing	123
9.2	Algorithm: Module Extraction	124
9.2.1	Fixing Violations in Pass-1	126

9.2.2	Fixing Violations in Pass-2	128
9.3	Application of Module Extraction Algorithm	128
9.3.1	Input Modules	129
9.3.2	Machining Modules	131
9.3.3	Finishing and Output Module	131
9.3.4	Inter-Modular Connector	131
9.3.5	The Complete Modular Petri Net	133
9.4	Analyzing the Fixes	134
9.4.1	General Analysis of the Fixes	134
9.4.2	P-Invariants	135
9.4.3	T-Invariants	138
	References	140
10	Activity-Oriented Petri Nets (AOPN)	141
10.1	The Background of AOPN	141
10.1.1	Formal Definition of AOPN	142
10.2	Two Phases of the AOPN Approach	142
10.3	Application Example on AOPN	143
10.3.1	Phase-1: Creating the Static Petri Net Graph	145
10.3.2	Phase-2: Adding the Run-Time Dynamics	145
10.3.3	GPenSIM Functions for AOPN Realization	146
10.3.4	Simulation of the AOPN Model	148
10.3.5	Discussion: Advantage of AOPN Approach	152
10.4	When Not to Use AOPN	154
	References	155
 Part IV Collaborating Modules		
11	Discrete Systems as Petri Modules	159
11.1	Module Abstraction	160
11.1.1	Collaborativeness and Timing	160
11.1.2	Collaborative Module	161
11.2	Loosely Connected Modules	164
11.2.1	Serial Block	164
11.2.2	Parallel Block	165
11.2.3	Cyclic (Iterating) Block	166
11.2.4	Alternating (Mutual-Exclusion) Block	167
11.2.5	Independent Block	168
11.2.6	Assembly Block	168
11.2.7	Disassembly Block	169
11.2.8	Source Block and Sink Block	169
11.3	Analysis: Connectivity of Modules	170
11.4	Adjacency Matrix and Laplacian Matrix	170
11.5	Spanning Tree and Connectedness	172
11.5.1	Spanning Tree and Connected Components	173
11.6	The Four Matrices	174

11.6.1	Reachability Matrix	174
11.6.2	Rader's Matrix	175
11.6.3	Connection Matrix	176
11.6.4	Component Matrix	177
11.7	Algorithms for Modular Connectivity	178
	References	178
12	Algorithms for Modular Connectivity	179
12.1	Algorithm: Existence of Spanning Tree	180
12.2	Algorithm: Source of a Spanning Tree	180
12.3	Algorithm: Functional Spanning Tree	181
12.4	Algorithm: Existence of Connected Components	182
12.5	Algorithm: Steiner Spanning Tree	183
12.5.1	Steiner Spanning Tree: The Basics	183
12.5.2	Steiner Spanning Tree: The Algorithm	185
12.6	Applications of the Algorithms	187
12.6.1	Scenario for Creation of Networks	187
12.7	Leader-Followers Networking (LFN)	187
12.7.1	Algorithm for Creation of LFN	188
12.7.2	Example for Creation of LFN	189
12.8	Steiner-Mode Networking (SMN)	189
12.8.1	Algorithm for Creation of SMN	190
12.8.2	Example for Creation of SMN	190
12.9	Negotiating-Peers Networking (NPN)	191
12.9.1	Algorithm for Creating NPN	192
12.9.2	Example for Creation of NPN	193
12.10	Chapter Summary	193
	References	194
13	Model Checking for Collaborativeness	197
13.1	The Problem	197
13.2	Towards a New Approach	198
13.3	Step-1: Identifying the Group of Modules	198
13.4	Step-2: Inviting a Module	199
13.5	Step-3: Generating the State Space	200
13.5.1	Terminal State and Terminal Transition	201
13.5.2	Possibility for Deadlocks	201
13.5.3	Possibility for Livelocks	203
13.5.4	Free from Livelocks and Deadlocks	205
13.6	Step-4: Analyzing the State Space	205
13.6.1	Making a Decision on Collaborativeness	206
13.7	Step-5: Establishing the Connectivity	207
13.8	The New Approach	208
13.9	Application Example	209
13.10	Chapter Summary	210
	References	210

14	Generic Petri Module	213
14.1	Structure of Petri Module for Collaboration	213
14.2	Colored Petri Module	214
14.3	Generic Petri Module	215
	Reference	217
Appendix A: GPenSIM Webpage		219
Index		221

About the Author

Reggie Davidrajuh has a Bachelor's study in Physics, a Master's degree in Control Systems, and a Ph.D. in Industrial Engineering (awarded by the Norwegian University of Science and Technology). Also, he has a D.Sc. (habilitation) degree in Information Science (AGH University of Science and Technology) and one more Ph.D. in Mechanical Engineering (Silesian University of Technology). He is presently a professor of Informatics at the University of Stavanger, Norway, and holds a visiting professor position at the Silesian University of Technology, Poland.

Dr. Davidrajuh is an editor of the journal "International Journal of Business and Systems Research". Also, he serves on the editorial committees of many journals that include "Expert Systems with Applications" (Elsevier) and "Archives of Control Sciences" (a Quarterly of Polish Academy of Sciences). Dr. Davidrajuh has published over 150 publications in diverse areas such as supply chain, e-commerce, e-government, modeling and simulation, discrete event systems, green power generation, etc. His three papers won the best paper awards: "Modeling humanoid robot as a discrete event system" at the IEEE Third International Conference on Artificial Intelligence, Modeling, and Simulation (AIMS2015), in December 2015, in Kuala Lumpur; "GPenSIM for Performance Evaluation of Event Graphs" at the International Scientific and Technical Conference on Manufacturing (Manufacturing 2017), in October 2017, in Poznan, Poland; "Measuring Network Centrality in Petri Nets" at the 2018 IEEE International Conference on Advanced Manufacturing (ICAM), in November 2018, in Yunlin, Taiwan.

He has organized over 60 international conferences and has given keynote speeches in four conferences. His current research interests are "Modeling, simulation, and performance analysis of discrete-event systems", Algorithms, and Graph Theory. He is a senior member of IEEE and a Fellow of the British Computer Society. He is also a member of the Norwegian Academy of Technical Sciences.

Nomenclature

λ_2	Algebraic Connectivity: the second eigenvalue of a Laplacian matrix
β	Self-loop strength (arc weight of an arc that is directed from a node to itself)
$\bullet p$	Input transitions of place p
$\bullet t$	Input places of transitions t
δ	Value for step-wise reduction of the self-loop strength
$\xrightarrow{t_i}$	Firing transition t_i
Φ	State formula (in CTL)
ϕ	Path formula (in CTL)
Φ_i	Petri module
Ψ_i	Inter-Modular Connector
A	Adjacency Matrix (N, N) of a graph, consisting of N nodes
A	CTL path quantifier for “all paths”
A	Incidence Matrix (m, n) of a Petri Net, m places and n transitions
A	Set of arcs (directed connections) of a Petri Net
$AOPN$	Activity-Oriented Petri Nets
a or a_i	Arc of a Petri Net
a or a_{ij}	Element of an adjacency matrix or incidence matrix
C	Connection Matrix
C_{m_i}	Collaborativeness of a Petri module
CTL	Computation Tree Logic
D	Rader’s Matrix, Diagonal matrix
E	(Number of) edges of a graph $G = (V, E)$
E	CTL path quantifier for “there exist some paths”
F	CTL temporal modality for “in future” or “eventually”
FMS	Flexible Manufacturing System
G	CTL temporal modality for “always” or “globally”
$GPenSIM$	General-purpose Petri Net Simulator
IMC	Inter-Modular Connector
L	Laplacian Matrix
LFN	Leader-Followers Networking
LTS	Labeled Transition System (<i>aka</i> the Kripke Structure)

M	Marking (state) of a Petri Net
$M(p)$	Marking of (the number of tokens in) place p
$M[t_i\rangle$	Enabled transition t_i in the marking M
M_0	Initial marking (initial state) of a Petri Net
$M_{\perp}$	Terminal state of a state space
$M_0[\}$	All the markings that are reachable from M_0
$M_0[t_1]M_1$	M_1 is directly reachable from M_0 due to the firing of t_1
MPN	Modular Petri Net, consisting of one or more Petri modules and zero or more Inter-Modular Connectors
MSF	Main Simulation File (in GPenSIM)
N	CTL temporal operator for “Next”
N	Natural numbers
N^+	Natural numbers, excluding zero
NPN	Negotiating-Peers Networking
P	Set of Places of a Petri Net
p	Atomic property (in CTL)
$p\bullet$	Output transitions of place p
PDF	Petri net Definition File (in GPenSIM)
PTN	Place–Transition (P/T) Petri Net
p or p_i	Place of a Petri Net
R	Reachability Matrix
R	Real numbers
R^+	Positive real numbers, greater than zero
RG	Reachability Graph (State Space)
$SCEG$	Strongly Connected Event Graph
SMN	Steiner-Mode Networking
T	Set of Transitions of a Petri Net
$t\bullet$	Output places of transitions t
$t_{\perp}$	Terminal transition of a state space
$t_i\rangle$	Enabled transition t_i
T_{m_i}	Timing of a Petri module
tI	Input port(s) of a Petri module
$tINV$	Transition for inviting the other modules for collaboration
$tMCC$	Transition for model checking for collaborativeness
tO	Output port(s) of a Petri module
TPN	Timed Petri Net
t or t_i	Transition of a Petri Net
U	CTL temporal operator for “Until”
V	(Number of) vertices of a graph $G = (V, E)$
$W(p_i, t_j)$	Weight of the arc between the place p_i and the transition t_j
Z	Integers

List of Figures

Fig. 1.1	A simple P-T Petri net	5
Fig. 1.2	State space of the marked Petri net shown in Fig. 1.1	7
Fig. 1.3	A Strongly Connected Event Graph (SCEG)	11
Fig. 1.4	A graph that is neither strongly connected not an event graph	11
Fig. 2.1	Composition of a non-primitive transition	17
Fig. 2.2	Maintaining the “atomicity” property during the firing of a non-primitive transition	18
Fig. 2.3	Layered architecture of GPenSIM	19
Fig. 2.4	Third-party toolboxes built on top of GPenSIM	19
Fig. 2.5	Separating the static and the dynamic details	20
Fig. 2.6	Integrating with MATLAB environment	21
Fig. 2.7	Petri net model of a production facility	22
Fig. 2.8	The Petri net Definition File (PDF)	23
Fig. 2.9	The pre-processor of tRobot_1 (“tRobot_1_pre.m”)	24
Fig. 2.10	The pre-processors of tRobot_2 and tRobot_3 (“tRobot_2_pre.m” and “tRobot_3_pre.m”)	24
Fig. 2.11	The main simulation file (MSF, “production.m”)	25
Fig. 2.12	Simulation results, showing how the tokens in places changed during the simulation	25
Fig. 3.1	A simple Flexible Manufacturing System (FMS)	30
Fig. 3.2	The Petri net model of a Flexible Manufacturing System	31
Fig. 3.3	The State Space of the Flexible Manufacturing System. <i>The state space is incomprehensible due to overlapping of the 45 states</i>	34
Fig. 3.4	Number of states in the state space versus number of initial tokens in the input buffers pIB1 and pIB2	35
Fig. 3.5	Factory circular transportation System involving AGVs	36
Fig. 3.6	Factory circular transportation system involving AGVs	37
Fig. 3.7	The state space of the circular transportation system	39
Fig. 3.8	Number of states in the state space versus number of initial tokens in the input buffer	40

Fig. 3.9	Production of cyclic processes	42
Fig. 3.10	Petri net model of the cyclic processes	44
Fig. 3.11	Summary of the resource usage	45
Fig. 3.12	Scheduling of the resources	45
Fig. 3.13	State space of cyclic processes, when $n = m = k = 1$	46
Fig. 4.1	The four GPenSIM files	52
Fig. 4.2	A monolithic P-T Petri net	54
Fig. 4.3	Alternating robots	55
Fig. 4.4	The specific processor files	56
Fig. 4.5	PDF for alternating robots	57
Fig. 4.6	MSF for alternating robots	57
Fig. 4.7	Simulation results	58
Fig. 4.8	The common pre-processor COMMON_PRE	58
Fig. 4.9	The common post-processor COMMON_POST	59
Fig. 4.10	The Petri net model of a simple soda vending machine	60
Fig. 4.11	PDF for the vending machine	62
Fig. 4.12	MSF for the vending machine	62
Fig. 4.13	The reachability graph for the vending machine	63
Fig. 4.14	The Petri net model satisfies the properties A and B	64
Fig. 4.15	The Petri net model satisfies the properties C and D	65
Fig. 4.16	A simple Flexible Manufacturing System (FMS)	66
Fig. 4.17	The Petri net model of a Flexible Manufacturing System	67
Fig. 4.18	The PDF	69
Fig. 4.19	The MSF	70
Fig. 4.20	Simulation result identifying the eight elementary cycles	70
Fig. 4.21	Simulation result showing minimum cycle and token flow rate	71
Fig. 6.1	Resource allocation system	88
Fig. 6.2	Modularization using fusion places	89
Fig. 6.3	Modular Petri net model of RAS: a proposal	90
Fig. 6.4	A modular Petri net with two modules and two IMCs	91
Fig. 7.1	The messages and interactions between the agents	100
Fig. 7.2	Generic Petri module of a communicating agent	100
Fig. 7.3	The modular Petri model	102
Fig. 7.4	Parallel execution of modules	104
Fig. 8.1	A Petri net model with three Petri modules	109
Fig. 8.2	PDF for the module Adder	110
Fig. 8.3	PDF for the module Multiplier	110
Fig. 8.4	PDF for the module Client	110
Fig. 8.5	Transitions of the module Adder and their visibility	112
Fig. 8.6	Transitions of the module Multiplier and their visibility	112
Fig. 8.7	Transitions of the module Client and their visibility	112
Fig. 8.8	Three communicating agents: the Adder module	114
Fig. 8.9	The MOD_Adder_PRE file	114
Fig. 8.10	Three communicating agents: the Multiplier module	115