

Algorithms for Parallel Simulation of Large-Scale DEVS and Cell-DEVS Models

By

Qi Liu, B. Eng., M. A. Sc.

A thesis submitted to the Faculty of Graduate Studies and Research

In partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Electrical and Computer Engineering

Ottawa-Carleton Institute for Electrical and Computer Engineering (OCIECE)

Department of Systems and Computer Engineering

Carleton University

Ottawa, Ontario, Canada

September 2010

© Copyright 2010, Qi Liu

The undersigned hereby recommends to the Faculty of Graduate Studies and Research
acceptance of the thesis

Algorithms for Parallel Simulation of Large-Scale DEVS and Cell-DEVS Models

Submitted by Qi Liu

In partial fulfillment of the requirements for the degree of
Doctor of Philosophy in Electrical and Computer Engineering

Thesis Supervisor
Dr. Gabriel A. Wainer

External Examiner
Dr. Richard M. Fujimoto

Chair, Department of Systems and Computer Engineering
Dr. Howard M. Schwartz

Carleton University

September 2010

Dedicated with all my love

to

my loving wife, Sara,

and to

my little son, Jed.

“Let us ... restrict ourselves to simple structures whenever possible and to avoid in all intellectual modesty ‘clever constructions’ like the plague.”

Edsger W. Dijkstra,
Notes on Structured Programming,
In Pursuit of Simplicity,
1969 - 1970.

Abstract

The **Discrete Event System Specification (DEVS)** provides a general methodology for hierarchical construction of reusable models in a modular way and has been used to simulate sophisticated systems in a variety of domains. This dissertation addresses software design and performance issues that arise in parallel simulation of large-scale DEVS-based models on both multiprocessor clusters and chip-multiprocessor architectures.

The **Time Warp (TW)** mechanism is the most well-known optimistic synchronization protocol for **Parallel Discrete-Event Simulations (PDES)**. With the increasing scale and complexity, TW simulations face new challenges in terms of excessive memory consumption and operational overhead. In an effort to alleviate these problems, a novel **Lightweight Time Warp (LTW)** protocol is proposed for efficient optimistic parallel DEVS simulation on multiprocessor clusters. By exploring the intrinsic computational properties of DEVS-based simulations, the LTW protocol allows purely optimistic parallel simulation to be driven by only a few full-fledged TW **Logical Processes (LPs)**, while most of the LPs are set free from the burden of TW execution. The experimental results indicate that simulation performance can be improved significantly in various aspects, including shortened execution time, reduced memory footprint, lowered operational overhead, accelerated event queue operations, facilitated process migration, and enhanced system stability and scalability.

To address the limitations of microprocessor performance, the industry is moving towards multicore chip-multiprocessor designs. As a latest example of this trend, the IBM Cell processor has attracted a growing interest from the modeling and simulation community. However, general-purpose PDES on such platform requires innovative redesign of existing algorithms in return for better simulation performance. To this end, a new computing technique called **Multicore Acceleration of DEVS Systems (MADS)** is developed for high-performance parallel DEVS simulation on the Cell processor, combining multi-grained parallelism and various optimizations to overcome the major performance bottlenecks, while hiding, to a great extent, the technical details of multicore programming from general users. Through the concept of LP virtualization, the MADS technique explicitly exploits the massive data- and event-level parallelism inherent in the simulation, making the achievable

performance gain more deterministic and predictable than the traditional LP-oriented approaches. Promising results have been produced in the experiments, demonstrating that the MADS technique can be used to accelerate both memory-bound and compute-bound computational kernels in demanding parallel DEVS simulations. The proposed technique not only allows a broad community of DEVS users to tap the potential of the Cell processor with a minimal knowledge of the multicore execution environment, but also makes it possible to integrate cluster-based parallel simulation with multicore-accelerated parallel simulation on hybrid supercomputers.

Acknowledgements

This work would not have been possible without the ceaseless inspiration, continuous support, and unending encouragement of my supervisor Dr. Gabriel Wainer. I am especially grateful for his commitment to providing a stimulating learning and research environment and his investment of time and effort in my professional and personal development. I also greatly appreciate Dr. Richard Fujimoto, Dr. Emil Petriu, Dr. Frank Dehne, and Dr. James Green for their willingness to be on my defense committee and for their scientific insight and expertise.

This research was supported in part by Ontario Graduate Scholarship Program (OGS), Ontario Graduate Scholarship in Science and Technology (OGSST), MITACS Accelerate Ontario Program, Canada Foundation for Innovation (CFI), Koningstein Scholarship for Excellence in Science and Engineering, Carleton University, and by IBM T. J. Watson Research Center. I would like to express my gratitude to Dr. Michael Perrone, Dr. Ligang Lu, Dr. Lurng-Kou Liu, and Dr. Daniele Scarpazza from the IBM Watson Research Center for their helpful suggestions and constructive feedback.

I would also like to thank the members of the ARS Laboratory and the Department of Systems and Computer Engineering, both past and present, as well as all my friends and colleagues at Carleton University. Special thanks go to Narendra Mehta for all kinds of timely technical assistance and Tao Yue for always being an instant message away.

Finally, I am forever indebted to my dear parents and beloved wife for their endless love, enduring patience, and kind understanding throughout the course of my studies.

Table of Contents

Abstract	v
Acknowledgements	vii
Table of Contents	viii
List of Tables	xii
List of Figures	xiii
List of Acronyms	xv
Chapter 1. Introduction	1
1.1. Research Motivations and Objectives	3
1.1.1. DEVS Simulation with Time Warp	3
1.1.2. DEVS Simulation on Cell Processor	5
1.2. Contributions	7
1.2.1. Lightweight Time Warp	7
1.2.2. Multicore Acceleration of DEVS Systems	9
1.3. Research Publications	10
1.4. Organization	13
Chapter 2. DEVS Framework and Its Implementation	14
2.1. Conceptual Modeling and Simulation Framework	14
2.2. Classical DEVS Formalism	16
2.3. Parallel DEVS Formalism	20
2.4. Timed Cell-DEVS Formalism	21
2.5. Parallel Cell-DEVS Formalism	24
2.6. Parallel DEVS and Cell-DEVS Simulation in CD++	25
2.6.1. Event-Processing Algorithms	27
2.6.2. Structural Representation	32
2.6.3. Computational Properties	34
Chapter 3. Literature Review	36
3.1. Parallel Discrete-Event Simulation	36
3.1.1. Conservative Synchronization Algorithms	37

3.1.2. Optimistic Synchronization Algorithms	39
3.2. Challenges of Optimistic PDES with Time Warp.....	41
3.2.1. Memory Management	41
3.2.2. Cascaded Rollback.....	45
3.2.3. Event Management.....	49
3.2.4. Dynamic Process Migration.....	52
3.2.5. DEVS-based Time Warp Simulation	54
3.3. Challenges of Efficient PDES on Cell Processor.....	56
3.3.1. Asymmetric Architecture with Explicit Memory Control.....	56
3.3.2. Multi-Grained Parallelization Strategies	59
3.3.3. Abstract Programming Models	60
3.3.4. Automated Compilation Techniques	62
3.3.5. Middleware Frameworks.....	63
3.3.6. Application Development on Cell.....	64
Chapter 4. The Lightweight Time Warp Protocol.....	67
4.1. Problem Statement and Design Methodologies	67
4.2. Concepts and Assumptions	69
4.3. Rule-Based Dual-Queue Event Management.....	71
4.3.1. Introducing a Volatile Input Queue.....	72
4.3.2. A Rule-Based Event-Scheduling Algorithm.....	75
4.4. Aggregate State Management.....	78
4.4.1. Introducing an aggregate state manager	78
4.4.2. An Enhanced Risk-Free Infrequent State-Saving Strategy	79
4.5. Lightweight Rollback Mechanism.....	82
4.5.1. Full-Fledged, Interface, and Lightweight LPs	82
4.5.2. A Lightweight Rollback Algorithm.....	83
4.6. Implications of the LTW Protocol.....	85
Chapter 5. Multicore Acceleration of DEVS Systems	89
5.1. Problem Statement and Design Methodologies	89
5.2. Workload Analysis and Computational Kernels	91
5.2.1. Large-Scale Simulations over a Long Period of Virtual Time	92

5.2.2. Highly-Active Simulation of Complex Model Behavior	94
5.3. FC Synchronization Kernel	96
5.3.1. Optimizing FC Synchronization Task	97
5.3.2. Flattening Simulator Timing Data	98
5.3.3. Processing Simulator Timing Data on SPE.....	100
5.3.4. FSK Orchestration Algorithms	103
5.4. Simulator Event-Processing Kernel.....	105
5.4.1. Event Level Parallelism.....	106
5.4.2. LP Virtualization	108
5.4.3. Virtual LP State Management.....	109
5.4.4. Decentralized Event Management	110
5.4.5. Evaluating Local Transition Functions on SPE.....	112
5.4.6. Processing SEK Jobs on SPE.....	116
5.4.7. An SEK Memory Control and Notification Algorithm.....	119
5.4.8. SEK Orchestration Algorithms.....	121
5.5. Parallel DEVS Simulation on the Cell Processor.....	125
5.6. Implications of the MADS Technique.....	127
Chapter 6. Performance Analysis.....	131
6.1. Introduction to the Benchmark Models	131
6.1.1. Definition of a Wildfire Model.....	131
6.1.2. Definition of a Watershed Model	133
6.2. Experimental Configurations and Performance Metrics	134
6.2.1. Configuration and Metrics for the LTW Protocol	135
6.2.2. Configuration and Metrics for the MADS Technique	137
6.3. Evaluation of the LTW Protocol	138
6.4. Evaluation of the MADS Technique	148
6.4.1. Performance of the FSK algorithms	148
6.4.2. Performance of the SEK algorithms.....	152
Chapter 7. Conclusion and Future Work.....	156
7.1. Summary of the Dissertation	156
7.2. Review of Key Contributions.....	158

7.2.1. Lightweight Time Warp Protocol	158
7.2.2. Multicore Acceleration of DEVS Systems	159
7.3. Suggestions for Future Research.....	160
7.3.1. Future Research on the LTW protocol.....	160
7.3.2. Future Research on the MADS technique.....	161
References	163

List of Tables

Table 1. Wildfire Simulation Profile on PPE	93
Table 2. FC Synchronization Task in the Wildfire Simulation	93
Table 3. Event Counts in Wildfire Simulation	94
Table 4. Watershed Simulation Profile on PPE.....	95
Table 5. FC Synchronization Task in the Watershed Simulation.....	95
Table 6. Event Counts in Watershed Simulation.....	96
Table 7. Wildfire Simulation Profile on PPE (Synchronization Optimized).....	98
Table 8. CD++ Syntax Node Representation (Partial)	114
Table 9. Performance Metrics Defined for the LTW Protocol.....	136
Table 10. Total Execution Time and Maximum Memory Consumption for <i>Fire1</i>	139
Table 11. Comparison for 100×100 <i>Fire1</i> on 14 Nodes.....	141
Table 12. Total Execution Time and Maximum Memory Consumption for <i>Fire2</i>	143
Table 13. Total Execution Time and Maximum Memory Consumption for <i>Watershed</i>	145
Table 14. Comparison for 100×100 <i>Fire2</i> on 20 Nodes.....	147
Table 15. Comparison for 20×20×2 <i>Watershed</i> on 18 Nodes	147
Table 16. 1024×1024 <i>Fire1</i> Simulation Profile on PPE (Data Flattened).....	149
Table 17. 320×320×2 <i>Watershed</i> Simulation Profile on PPE (Data Flattened)	153

List of Figures

Figure 1. Entities and Relationships of a System M&S Framework [Zei00].....	14
Figure 2. Layered Software Architecture of the M&S Framework.....	15
Figure 3. Informal Illustration of a DEVS Atomic Model [Zei00]	17
Figure 4. Informal Illustration of a Timed Cell-DEVS Atomic Model [Wai02a].....	23
Figure 5. Flat LP Structure in PCD++	26
Figure 6. Simulator Event-Processing Algorithms.....	28
Figure 7. FC Event-Processing Algorithms.....	29
Figure 8. NC Event-Processing Algorithms	31
Figure 9. PCD++ Message-Passing Scenario	32
Figure 10. Multi-Phased Simulation Process on a Node	33
Figure 11. Cell Processor Block Diagram	57
Figure 12. Division of Simulation Domains in the LTW Protocol	69
Figure 13. A Message-Passing Scenario with LTW Event Classification	72
Figure 14. A Message-Passing Scenario from the TW Perspective	73
Figure 15. Dual-Queue Event Scheduling	75
Figure 16. LTW Event-Scheduling Algorithm.....	76
Figure 17. Structure of FC Aggregate State Manager	78
Figure 18. Introducing a State-Saving Phase for Each Virtual Time	80
Figure 19. LTW State-Saving Algorithm	81
Figure 20. LTW Rollback Algorithm for FC.....	84
Figure 21. LTW Rollback Algorithm for Event Scheduler	85
Figure 22. A Flat Data Layout for the FSK.....	99
Figure 23. Parallel Data Processing on the SPEs	101
Figure 24. Function <code>findMinTime</code> Definition.....	101
Figure 25. Function <code>findImminents</code> Definition.....	102
Figure 26. FSK Main Loop and Function <code>terminateFSK</code>	103
Figure 27. In-Place Invocation of Function <code>findMinTime</code> at the FC.....	104
Figure 28. In-Advance Invocation of Function <code>findImminents</code> at the NC.....	104
Figure 29. Retrieving Imminent IDs at the FC.....	105

Figure 30. Terminating FSKs at the NC.....	105
Figure 31. Event Level Parallelism in DEVS-based Simulation Process.....	107
Figure 32. Virtual Simulator State Management.....	110
Figure 33. Virtual Simulator Event Management.....	111
Figure 34. Evaluation of a Local Transition Function.....	113
Figure 35. Transforming the Syntax Trees of a State Transition Rule.....	114
Figure 36. Double-Buffered Rule Evaluation on the SPEs	116
Figure 37. SEK Job-Processing Algorithms.....	118
Figure 38. Pending Job Queue for an SEK.....	119
Figure 39. SEK Memory Control and Notification Algorithm.....	120
Figure 40. SEK Main Loop and Function <code>terminateSEK</code>	122
Figure 41. SEK Control Message Bit Pattern.....	122
Figure 42. SEK Orchestration Algorithm on PPE (Part I).....	123
Figure 43. SEK Orchestration Algorithm on PPE (Part II)	124
Figure 44. Terminating SEKs at the NC.....	125
Figure 45. Architectural Overview of the MADS Technique	126
Figure 46. Predetermined Spread Rates for the <i>Fire1</i> Model [Wai06]	132
Figure 47. A Skeleton of the <i>Fire1</i> Model Definition in CD++ [Wai06]	132
Figure 48. A Skeleton of the <i>Fire2</i> Model Definition in CD++.....	133
Figure 49. A Skeleton of the <i>Watershed</i> Model Definition in CD++ [Wai06]	134
Figure 50. <i>Fire1</i> Overall Speedups (Test Cases with Sufficient Memory)	140
Figure 51. <i>Fire2</i> Overall Speedups (Test Cases with Sufficient Memory)	144
Figure 52. <i>Watershed</i> Overall Speedups (Test Cases with Sufficient Memory).....	146
Figure 53. Total Execution Time in the 1024×1024 <i>Fire1</i> Simulation	148
Figure 54. FSK Scale-Ups in the 1024×1024 <i>Fire1</i> Simulation	150
Figure 55. Total Execution Time in <i>Fire1</i> Simulations with Varied Sizes	150
Figure 56. Overall Simulation Scale-Ups in <i>Fire1</i> Simulations.....	151
Figure 57. Total Execution Time in the 320×320×2 <i>Watershed</i> Simulation.....	152
Figure 58. Total Execution Time in <i>Watershed</i> Simulations with Varied Sizes	154
Figure 59. Overall Simulation Scale-Ups in <i>Watershed</i> Simulations.....	155

List of Acronyms

ALF	Accelerated Library Framework
API	Application Programming Interface
BIF	Broadband Interface
CA	Cellular Automata
CMP	Chip Multiprocessor
CT	Precondition Syntax Tree
DEDS	Discrete Event Dynamic System
DEVS	Discrete Event System Specification
DMA	Direct Memory Access
DT	Delay Syntax Tree
EIB	Element Interconnect Bus
EIC	External Input Coupling
EOC	External Output Coupling
FC	Flat Coordinator
FEL	Future Event List
FIFO	First In, First Out
FSK	FC Synchronization Kernel
GPU	Graphics Processing Units
GVT	Global Virtual Time
IA	Imminent ID Array
IC	Internal Coupling
ISA	Instruction Set Architecture
LBTS	Lower Bound on Time Stamp
LCT	Latest state Change Time
LP	Logical Process
LS	Local Storage
LTSF	Least Time Stamp First
LTW	Lightweight Time Warp

LVT	Local Virtual Time
M&S	Modeling and Simulation
MADS	Multicore Acceleration of DEVS Systems
MIC	Memory Interface Controller
MIMD	Multiple Instruction, Multiple Data
MPI	Message Passing Interface
MTSS	Message Type-based State Saving
NC	Node Coordinator
PDES	Parallel Discrete Event Simulation
P-DEVS	Parallel DEVS
PPE	Power Processor Element
PT	Postcondition Syntax Tree
RPC	Remote Procedure Call
SDK	Software Development Kit
SEK	Simulator Event-processing Kernel
SIMD	Single Instruction, Multiple Data
SMP	Symmetric Multiprocessing
SMT	Simultaneous Multithreading
SPE	Synergistic Processing Elements
STL	Standard Template Library
TA	Time Array
TW	Time Warp
TWLP	Time Warp Logical Process