Parallel Programming

Thomas Rauber · Gudula Rünger

Parallel Programming

For Multicore and Cluster Systems

Springer

Thomas Rauber
Universität Bayreuth
Computer Science Department
95440 Bayreuth
Germany
rauber@uni-bayreuth.de

Gudula Rünger
Technische Universität Chemnitz
Computer Science Department
09107 Chemnitz
Germany
ruenger@informatik.tu-chemnitz.de

ISBN 978-3-642-04817-3 e-ISBN 978-3-642-04818-0
DOI 10.1007/978-3-642-04818-0
Springer Heidelberg Dordrecht London New York

ACM Computing Classification (1998): D.1, C.1, C.2, C.4

Library of Congress Control Number: 2009941473

This is an extended English language translation of the German language edition:
Parallele Programmierung (2nd edn.) by T. Rauber and G. Rünger
Published in the book series: Springer-Lehrbuch

Cover design: KuenkelLopka GmbH, Heidelberg

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

Innovations in hardware architecture, like hyperthreading or multicore processors, make parallel computing resources available for inexpensive desktop computers. However, the use of these innovations requires parallel programming techniques. In a few years, many standard software products will be based on concepts of parallel programming to use the hardware resources of future multicore processors efficiently. Thus, the need for parallel programming will extend to all areas of software development. The application area will be much larger than the area of scientific computing, which used to be the main area for parallel computing for many years. The expansion of the application area for parallel computing will lead to an enormous need for software developers with parallel programming skills. Some chip manufacturers already demand to include parallel programming as a standard course in computer science curricula.

This book takes up the new development in processor architecture by giving a detailed description of important parallel programming techniques that are necessary for developing efficient programs for multicore processors as well as for parallel cluster systems or supercomputers. Both shared and distributed address space architectures are covered. The main goal of the book is to present parallel programming techniques that can be used in many situations for many application areas and to enable the reader to develop correct and efficient parallel programs. Many example programs and exercises are provided to support this goal and to show how the techniques can be applied to further applications. The book can be used as both a textbook for students and a reference book for professionals. The material of the book has been used for courses in parallel programming at different universities for many years.

This is the third version of the book on parallel programming. The first two versions have been published in German in the years 2000 and 2007, respectively. This new English version is an updated and revised version of the newest German edition of the book. The update especially covers new developments in the area of multicore processors as well as a more detailed description of OpenMP and Java threads.

The content of the book consists of three main parts, covering all areas of parallel computing: the architecture of parallel systems, parallel programming models and environments, and the implementation of efficient application algorithms. The

emphasis lies on parallel programming techniques needed for different architectures.

The first part contains an overview of the architecture of parallel systems, including cache and memory organization, interconnection networks, routing and switching techniques, as well as technologies that are relevant for modern and future multicore processors.

The second part presents parallel programming models, performance models, and parallel programming environments for message passing and shared memory models, including MPI, Pthreads, Java threads, and OpenMP. For each of these parallel programming environments, the book gives basic concepts as well as more advanced programming methods and enables the reader to write and run semantically correct and efficient parallel programs. Parallel design patterns like pipelining, client–server, and task pools are presented for different environments to illustrate parallel programming techniques and to facilitate the implementation of efficient parallel programs for a wide variety of application areas. Performance models and techniques for runtime analysis are described in detail, as they are a prerequisite for achieving efficiency and high performance.

The third part applies the programming techniques from the second part to representative algorithms from scientific computing. The emphasis lies on basic methods for solving linear equation systems, which play an important role in many scientific simulations. The focus of the presentation lies on the analysis of the algorithmic structure of the different algorithms, which is the basis for parallelization, and not on the mathematical properties of the solution methods. For each algorithm, the book discusses different parallelization variants, using different methods and strategies.

Many colleagues and students have helped to improve the quality of this book. We would like to thank all of them for their help and constructive criticisms. For numerous corrections and suggestions we would like to thank Jörg Dümmler, Marvin Ferber, Michael Hofmann, Ralf Hoffmann, Sascha Hunold, Matthias Korch, Raphael Kunis, Jens Lang, John O'Donnell, Andreas Prell, Carsten Scholtes, and Michael Schwind. Many thanks to Matthias Korch, Carsten Scholtes, and Michael Schwind for help with the exercises. We thank Monika Glaser for her help and support with the LaTeX typesetting of the book. We also thank all the people who have been involved in the writing of the first two German versions of this book. It has been a pleasure working with the Springer Verlag in the development of this book. We especially thank Ralf Gerstner for his support and patience.

Bayreuth — Thomas Rauber
Chemnitz — Gudula Rünger
August 2009

Contents

1 Introduction 1
1.1 Classical Use of Parallelism 1
1.2 Parallelism in Today's Hardware 2
1.3 Basic Concepts 3
1.4 Overview of the Book 5

2 Parallel Computer Architecture 7
2.1 Processor Architecture and Technology Trends 7
2.2 Flynn's Taxonomy of Parallel Architectures 10
2.3 Memory Organization of Parallel Computers 12
2.3.1 Computers with Distributed Memory Organization 12
2.3.2 Computers with Shared Memory Organization 15
2.3.3 Reducing Memory Access Times 17
2.4 Thread-Level Parallelism 20
2.4.1 Simultaneous Multithreading 21
2.4.2 Multicore Processors 22
2.4.3 Architecture of Multicore Processors 24
2.5 Interconnection Networks 28
2.5.1 Properties of Interconnection Networks 29
2.5.2 Direct Interconnection Networks 32
2.5.3 Embeddings 37
2.5.4 Dynamic Interconnection Networks 40
2.6 Routing and Switching 46
2.6.1 Routing Algorithms 46
2.6.2 Routing in the Omega Network 53
2.6.3 Switching 56
2.6.4 Flow Control Mechanisms 63
2.7 Caches and Memory Hierarchy 64
2.7.1 Characteristics of Caches 65
2.7.2 Write Policy 73
2.7.3 Cache Coherency 75
2.7.4 Memory Consistency 82
2.8 Exercises for Chap. 2 88

3 Parallel Programming Models 93
3.1 Models for Parallel Systems 93
3.2 Parallelization of Programs 96
3.3 Levels of Parallelism 98
3.3.1 Parallelism at Instruction Level 98
3.3.2 Data Parallelism 100
3.3.3 Loop Parallelism 102
3.3.4 Functional Parallelism 104
3.3.5 Explicit and Implicit Representation of Parallelism 105
3.3.6 Parallel Programming Patterns 108
3.4 Data Distributions for Arrays 113
3.4.1 Data Distribution for One-Dimensional Arrays 113
3.4.2 Data Distribution for Two-Dimensional Arrays 114
3.4.3 Parameterized Data Distribution 116
3.5 Information Exchange 117
3.5.1 Shared Variables 117
3.5.2 Communication Operations 118
3.6 Parallel Matrix–Vector Product 125
3.6.1 Parallel Computation of Scalar Products 126
3.6.2 Parallel Computation of the Linear Combinations 129
3.7 Processes and Threads 130
3.7.1 Processes 130
3.7.2 Threads 132
3.7.3 Synchronization Mechanisms 136
3.7.4 Developing Efficient and Correct Thread Programs 139
3.8 Further Parallel Programming Approaches 141
3.8.1 Approaches for New Parallel Languages 142
3.8.2 Transactional Memory 144
3.9 Exercises for Chap. 3 147

4 Performance Analysis of Parallel Programs 151
4.1 Performance Evaluation of Computer Systems 152
4.1.1 Evaluation of CPU Performance 152
4.1.2 MIPS and MFLOPS 154
4.1.3 Performance of Processors with a Memory Hierarchy 155
4.1.4 Benchmark Programs 158
4.2 Performance Metrics for Parallel Programs 161
4.2.1 Speedup and Efficiency 162
4.2.2 Scalability of Parallel Programs 165
4.3 Asymptotic Times for Global Communication 166
4.3.1 Implementing Global Communication Operations 167
4.3.2 Communications Operations on a Hypercube 173
4.4 Analysis of Parallel Execution Times 181
4.4.1 Parallel Scalar Product 181

4.4.2 Parallel Matrix–Vector Product . . . 183
4.5 Parallel Computational Models . . . 186
4.5.1 PRAM Model . . . 186
4.5.2 BSP Model . . . 189
4.5.3 LogP Model . . . 191
4.6 Exercises for Chap. 4 . . . 193

5 Message-Passing Programming . . . 197
5.1 Introduction to MPI . . . 198
5.1.1 MPI Point-to-Point Communication . . . 199
5.1.2 Deadlocks with Point-to-Point Communications . . . 204
5.1.3 Non-blocking Operations and Communication Modes . . . 208
5.1.4 Communication Mode . . . 212
5.2 Collective Communication Operations . . . 213
5.2.1 Collective Communication in MPI . . . 214
5.2.2 Deadlocks with Collective Communication . . . 227
5.3 Process Groups and Communicators . . . 229
5.3.1 Process Groups in MPI . . . 229
5.3.2 Process Topologies . . . 234
5.3.3 Timings and Aborting Processes . . . 239
5.4 Introduction to MPI-2 . . . 240
5.4.1 Dynamic Process Generation and Management . . . 240
5.4.2 One-Sided Communication . . . 243
5.5 Exercises for Chap. 5 . . . 252

6 Thread Programming . . . 257
6.1 Programming with Pthreads . . . 257
6.1.1 Creating and Merging Threads . . . 259
6.1.2 Thread Coordination with Pthreads . . . 263
6.1.3 Condition Variables . . . 270
6.1.4 Extended Lock Mechanism . . . 274
6.1.5 One-Time Initialization . . . 276
6.1.6 Implementation of a Task Pool . . . 276
6.1.7 Parallelism by Pipelining . . . 280
6.1.8 Implementation of a Client–Server Model . . . 286
6.1.9 Thread Attributes and Cancellation . . . 290
6.1.10 Thread Scheduling with Pthreads . . . 299
6.1.11 Priority Inversion . . . 303
6.1.12 Thread-Specific Data . . . 306
6.2 Java Threads . . . 308
6.2.1 Thread Generation in Java . . . 308
6.2.2 Synchronization of Java Threads . . . 312
6.2.3 Wait and Notify . . . 320
6.2.4 Extended Synchronization Patterns . . . 326

6.2.5 Thread Scheduling in Java . . . 331
6.2.6 Package `java.util.concurrent` . . . 332
6.3 OpenMP . . . 339
6.3.1 Compiler Directives . . . 340
6.3.2 Execution Environment Routines . . . 348
6.3.3 Coordination and Synchronization of Threads . . . 349
6.4 Exercises for Chap. 6 . . . 353

7 Algorithms for Systems of Linear Equations . . . 359
7.1 Gaussian Elimination . . . 360
7.1.1 Gaussian Elimination and LU Decomposition . . . 360
7.1.2 Parallel Row-Cyclic Implementation . . . 363
7.1.3 Parallel Implementation with Checkerboard Distribution . . . 367
7.1.4 Analysis of the Parallel Execution Time . . . 373
7.2 Direct Methods for Linear Systems with Banded Structure . . . 378
7.2.1 Discretization of the Poisson Equation . . . 378
7.2.2 Tridiagonal Systems . . . 383
7.2.3 Generalization to Banded Matrices . . . 395
7.2.4 Solving the Discretized Poisson Equation . . . 397
7.3 Iterative Methods for Linear Systems . . . 399
7.3.1 Standard Iteration Methods . . . 400
7.3.2 Parallel Implementation of the Jacobi Iteration . . . 404
7.3.3 Parallel Implementation of the Gauss–Seidel Iteration . . . 405
7.3.4 Gauss–Seidel Iteration for Sparse Systems . . . 407
7.3.5 Red–Black Ordering . . . 411
7.4 Conjugate Gradient Method . . . 417
7.4.1 Sequential CG Method . . . 418
7.4.2 Parallel CG Method . . . 420
7.5 Cholesky Factorization for Sparse Matrices . . . 424
7.5.1 Sequential Algorithm . . . 424
7.5.2 Storage Scheme for Sparse Matrices . . . 430
7.5.3 Implementation for Shared Variables . . . 432
7.6 Exercises for Chap. 7 . . . 437

References . . . 441

Index . . . 449