

# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Classical Use of Parallelism	1
1.2	Parallelism in Today's Hardware	2
1.3	Basic Concepts	4
1.4	Overview of the Book	5
<b>2</b>	<b>Parallel Computer Architecture</b>	<b>9</b>
2.1	Processor Architecture and Technology Trends	9
2.2	Flynn's Taxonomy of Parallel Architectures	13
2.3	Memory Organization of Parallel Computers	14
2.3.1	Computers with Distributed Memory Organization	15
2.3.2	Computers with Shared Memory Organization	18
2.3.3	Reducing memory access times	20
2.4	Thread-Level Parallelism	24
2.4.1	Simultaneous Multithreading	24
2.4.2	Energy Consumption of Processors	25
2.4.3	Multicore Processors	26
2.4.4	Architecture of Multicore Processors	28
2.4.5	Example: Architecture of the Intel Core i7	32
2.5	Interconnection Networks	35
2.5.1	Properties of Interconnection Networks	36
2.5.2	Direct Interconnection Networks	38
2.5.3	Embeddings	44
2.5.4	Dynamic Interconnection Networks	47
2.6	Routing and Switching	52
2.6.1	Routing Algorithms	53
2.6.2	Routing in the Omega Network	61
2.6.3	Switching	63
2.6.4	Flow control mechanisms	71
2.7	Caches and Memory Hierarchy	72
2.7.1	Characteristics of Caches	73
2.7.2	Write Policy	81
2.7.3	Cache coherency	83

2.7.4	Memory consistency . . . . .	91
2.8	Example: IBM Blue Gene supercomputer . . . . .	97
2.9	Exercises for Chapter 2 . . . . .	100
<b>3</b>	<b>Parallel Programming Models . . . . .</b>	<b>105</b>
3.1	Models for parallel systems . . . . .	105
3.2	Parallelization of programs . . . . .	108
3.3	Levels of parallelism . . . . .	110
3.3.1	Parallelism at instruction level . . . . .	110
3.3.2	Data parallelism . . . . .	112
3.3.3	Loop parallelism . . . . .	114
3.3.4	Functional parallelism . . . . .	116
3.3.5	Explicit and implicit representation of parallelism . . . . .	117
3.3.6	Parallel programming patterns . . . . .	120
3.4	SIMD Computations . . . . .	125
3.4.1	Execution of vector operations . . . . .	125
3.4.2	SIMD instructions . . . . .	127
3.5	Data distributions for arrays . . . . .	128
3.5.1	Data distribution for one-dimensional arrays . . . . .	129
3.5.2	Data distribution for two-dimensional arrays . . . . .	130
3.5.3	Parameterized data distribution . . . . .	132
3.6	Information exchange . . . . .	133
3.6.1	Shared variables . . . . .	133
3.6.2	Communication operations . . . . .	134
3.7	Parallel matrix-vector product . . . . .	141
3.7.1	Parallel computation of scalar products . . . . .	142
3.7.2	Parallel computation of the linear combinations . . . . .	145
3.8	Processes and Threads . . . . .	146
3.8.1	Processes . . . . .	148
3.8.2	Threads . . . . .	149
3.8.3	Synchronization mechanisms . . . . .	152
3.8.4	Developing efficient and correct thread programs . . . . .	156
3.9	Further parallel programming approaches . . . . .	158
3.9.1	Approaches for new parallel languages . . . . .	159
3.9.2	Transactional memory . . . . .	161
3.10	Exercises for Chapter 3 . . . . .	164
<b>4</b>	<b>Performance Analysis of Parallel Programs . . . . .</b>	<b>169</b>
4.1	Performance Evaluation of Computer Systems . . . . .	170
4.1.1	Evaluation of CPU Performance . . . . .	170
4.1.2	MIPS and MFLOPS . . . . .	172
4.1.3	Performance of Processors with a Memory Hierarchy . . . . .	174
4.1.4	Benchmark Programs . . . . .	176

4.2	Performance Metrics for Parallel Programs . . . . .	179
4.2.1	Speedup and Efficiency . . . . .	180
4.2.2	Scalability of Parallel Programs . . . . .	183
4.3	Asymptotic Times for Global Communication . . . . .	184
4.3.1	Implementing Global Communication Operations . . . .	186
4.3.2	Communications Operations on a Hypercube. . . . .	191
4.4	Analysis of Parallel Execution Times . . . . .	199
4.4.1	Parallel Scalar Product . . . . .	199
4.4.2	Parallel Matrix-vector Product . . . . .	201
4.5	Parallel Computational Models . . . . .	203
4.5.1	PRAM Model . . . . .	204
4.5.2	BSP Model . . . . .	207
4.5.3	LogP Model . . . . .	209
4.6	Loop Scheduling and Loop Tiling . . . . .	211
4.6.1	Loop Scheduling . . . . .	212
4.6.2	Loop Tiling . . . . .	220
4.7	Exercises for Chapter 4 . . . . .	222
<b>5</b>	<b>Message-Passing Programming.</b> . . . .	227
5.1	Introduction to MPI . . . . .	228
5.1.1	MPI point-to-point communication . . . . .	230
5.1.2	Deadlocks with Point-to-point Communications . . . . .	234
5.1.3	Nonblocking Operations and Communication Modes . . . . .	237
5.1.4	Communication mode . . . . .	241
5.2	Collective Communication Operations . . . . .	243
5.2.1	Collective Communication in MPI . . . . .	243
5.2.2	Deadlocks with Collective Communication . . . . .	256
5.3	Process Groups and Communicators . . . . .	258
5.3.1	Process Groups in MPI . . . . .	259
5.3.2	Process Topologies . . . . .	264
5.3.3	Timings and aborting processes . . . . .	268
5.4	Introduction to MPI-2 . . . . .	269
5.4.1	Dynamic Process Generation and Management . . . . .	269
5.4.2	One-sided communication . . . . .	272
5.5	Exercises for Chapter 5 . . . . .	281
<b>6</b>	<b>Thread Programming.</b> . . . .	287
6.1	Programming with Pthreads . . . . .	287
6.1.1	Creating and Merging Threads . . . . .	289
6.1.2	Thread Coordination with Pthreads . . . . .	293
6.1.3	Condition Variables . . . . .	298
6.1.4	Extended Lock Mechanism . . . . .	304
6.1.5	One-Time Initialization . . . . .	306
6.1.6	Implementation of a Task Pool . . . . .	307

6.1.7	Parallelism by Pipelining . . . . .	310
6.1.8	Implementation of a Client-Server Model . . . . .	316
6.1.9	Thread Attributes and Cancellation . . . . .	321
6.1.10	Thread Scheduling with Pthreads . . . . .	327
6.1.11	Priority Inversion . . . . .	333
6.1.12	Thread-specific Data . . . . .	335
6.2	Java Threads . . . . .	337
6.2.1	Thread Generation in Java . . . . .	337
6.2.2	Synchronization of Java Threads . . . . .	341
6.2.3	Wait and Notify . . . . .	349
6.2.4	Extended Synchronization Patterns . . . . .	355
6.2.5	Thread Scheduling in Java . . . . .	359
6.2.6	Package <code>java.util.concurrent</code> . . . . .	361
6.3	OpenMP . . . . .	367
6.3.1	Compiler directives . . . . .	369
6.3.2	Execution environment routines . . . . .	377
6.3.3	Coordination and synchronization of threads . . . . .	377
6.4	Exercises for Chapter 6 . . . . .	383
<b>7</b>	<b>General Purpose GPU Programming . . . . .</b>	<b>387</b>
7.1	The Architecture of GPUs . . . . .	387
7.2	Introduction to CUDA Programming . . . . .	393
7.3	Synchronization and Shared Memory . . . . .	399
7.4	CUDA Thread Scheduling . . . . .	404
7.5	Efficient Memory Access and Tiling Technique . . . . .	406
7.6	Introduction to OpenCL . . . . .	412
7.7	Exercises for Chapter 7 . . . . .	414
<b>8</b>	<b>Algorithms for Systems of Linear Equations . . . . .</b>	<b>417</b>
8.1	Gaussian Elimination . . . . .	418
8.1.1	Gaussian Elimination and LU Decomposition . . . . .	418
8.1.2	Parallel Row-Cyclic Implementation . . . . .	422
8.1.3	Parallel Implementation with Checkerboard Distribution . . . . .	425
8.1.4	Analysis of the Parallel Execution Time . . . . .	431
8.2	Direct Methods for Linear Systems with Banded Structure . . . . .	436
8.2.1	Discretization of the Poisson Equation . . . . .	436
8.2.2	Tridiagonal Systems . . . . .	441
8.2.3	Generalization to Banded Matrices . . . . .	454
8.2.4	Solving the Discretized Poisson Equation . . . . .	456
8.3	Iterative Methods for Linear Systems . . . . .	458
8.3.1	Standard Iteration Methods . . . . .	459
8.3.2	Parallel implementation of the Jacobi Iteration . . . . .	463
8.3.3	Parallel Implementation of the Gauss-Seidel Iteration . . . . .	465

8.3.4	Gauss-Seidel Iteration for Sparse Systems. . . . .	467
8.3.5	Red-black Ordering . . . . .	470
8.4	Conjugate Gradient Method. . . . .	476
8.4.1	Sequential CG method . . . . .	476
8.4.2	Parallel CG Method . . . . .	478
8.5	Cholesky Factorization for Sparse Matrices. . . . .	483
8.5.1	Sequential Algorithm . . . . .	483
8.5.2	Storage Scheme for Sparse Matrices. . . . .	489
8.5.3	Implementation for Shared Variables . . . . .	491
8.6	Exercises for Chapter 8 . . . . .	496
<b>References . . . . .</b>		<b>501</b>
<b>Index . . . . .</b>		<b>509</b>



<http://www.springer.com/978-3-642-37800-3>

Parallel Programming  
for Multicore and Cluster Systems

Rauber, Th.; Rünger, G.

2013, XIII, 516 p., Hardcover

ISBN: 978-3-642-37800-3