

Contents

| | | |
|----------|--|-----------|
| 1 | Basic Concepts in Parallel Algorithms and Parallel Programming . . | 1 |
| 1.1 | Parallel Machines | 1 |
| 1.2 | Basic Concepts of Parallel Programs | 7 |
| 1.2.1 | Partial Orders as Basic Parallel Programs | 9 |
| 1.3 | Time Diagrams | 14 |
| 1.4 | Execution Times of Parallel Programs | 17 |
| 1.5 | Moving to a General Parallel Programming Language | 22 |
| 1.6 | Developing Practical Parallel Algorithms | 29 |
| 1.6.1 | Some Basic Properties of Efficient Parallel Algorithms . . . | 30 |
| 1.6.2 | Reducing Synchronization in Practical Parallel Algorithms | 33 |
| 1.6.3 | Some General Methods for Implementing the Scalability of Parallel Algorithms | 36 |
| 1.7 | Exercises | 39 |
| 1.8 | Bibliographic Notes | 42 |
| | References | 43 |
| 2 | Principles of Shared Memory Parallel Programming Using <i>ParC</i> . . | 45 |
| 2.1 | Introduction | 45 |
| 2.1.1 | System Structure | 47 |
| 2.2 | Parallel Constructs | 48 |
| 2.3 | Scoping Rules | 55 |
| 2.3.1 | Using Static Variables | 60 |
| 2.3.2 | Bypassing the Scoping Rules | 61 |
| 2.4 | Effect of Execution Order and Atomicity | 62 |
| 2.5 | Parallel Algorithms for Finding the Minimum | 63 |
| 2.5.1 | Divide and Conquer Min | 64 |
| 2.5.2 | Crash Min | 64 |
| 2.5.3 | Random Min | 66 |
| 2.5.4 | Divide and Crush | 67 |
| 2.5.5 | Asynchronous Min | 71 |

| | | |
|----------|--|------------|
| 2.6 | Recursion and Parallel Loops: The Parallel Prefix Example | 72 |
| 2.7 | Minimum Spanning Tree | 77 |
| 2.8 | Exercises | 82 |
| 2.8.1 | Questions About Summing | 84 |
| 2.8.2 | Questions About Min Algorithms | 86 |
| 2.8.3 | Questions About Snake Sorting | 86 |
| 2.9 | Bibliographic Notes | 88 |
| | References | 90 |
| 3 | Locality and Synchronization | 93 |
| 3.1 | Locality of Memory Operations via the Scoping Rules | 94 |
| 3.2 | Locality of Memory Operations Through Partitioned Arrays | 96 |
| 3.3 | Synchronization | 102 |
| 3.4 | Synchronization Through Shared Variables | 103 |
| 3.5 | Fetch and Add Synchronization | 105 |
| 3.6 | The Sync Instruction | 110 |
| 3.7 | Controlling the Execution by Using the Yield and Switch Instructions | 114 |
| 3.8 | Fairness Related Issues | 117 |
| 3.9 | Scheduling Policy | 118 |
| 3.10 | Forced Termination | 118 |
| 3.11 | Exercises | 122 |
| 3.11.1 | Replacing Sync with f&a() | 122 |
| 3.11.2 | Draw Program | 123 |
| 3.12 | Bibliographic Notes | 124 |
| | References | 126 |
| 4 | Multicore Machines | 129 |
| 4.1 | Caches | 130 |
| 4.2 | Basic Mode of Operation of Multicore Machines | 132 |
| 4.3 | The MESI Protocol for Memory Coherence in Multicore Machines | 134 |
| 4.4 | Counting MESI's Transitions and Overhead | 138 |
| 4.5 | The MESI Protocol Preserves Sequential Consistency | 140 |
| 4.6 | The MOESI Extension | 142 |
| 4.7 | Preserving the Locality Principle of ParC | 143 |
| 4.8 | False Sharing of Cache Lines | 147 |
| 4.9 | Controlling Cache Line Sharing at Different Times | 149 |
| 4.10 | Prefetching Data | 151 |
| 4.11 | Exercises | 154 |
| 4.12 | Bibliographic Notes | 156 |
| | References | 157 |
| 5 | Improving the Performance of Parallel Programs: The Analytical Approach | 159 |
| 5.1 | Introduction | 159 |

| | | |
|----------|---|------------|
| 5.2 | A Simple Model of a Virtual Machine | 160 |
| 5.3 | The <i>ParC</i> Virtual Machine Model | 161 |
| 5.4 | Speedup Notion for Programs | 167 |
| 5.5 | Using the Speedup Factors | 170 |
| 5.6 | The Effect of Scheduling on $T(R)$ | 175 |
| 5.7 | Accounting for Memory References for Multicore Machines | 183 |
| 5.8 | On the Usage of a Simulator | 185 |
| 5.9 | Conclusions | 187 |
| 5.10 | Exercises | 188 |
| 5.10.1 | Optimizing Speedup Factors | 188 |
| 5.10.2 | Amdhal's Law | 189 |
| 5.10.3 | Scheduling | 190 |
| 5.10.4 | Two-Processor Machines | 191 |
| 5.10.5 | Sync-Based Exercises | 192 |
| 5.10.6 | The Effect of Sync on the Execution Time | 193 |
| 5.11 | Bibliographic Notes | 195 |
| | References | 196 |
| 6 | Compilation Techniques | 197 |
| 6.1 | Introduction | 197 |
| 6.2 | Inserting Explicit Context Switch Instructions | 200 |
| 6.3 | Using Function Calls to Spawn Threads | 202 |
| 6.4 | Scheduling and Supporting Context-Switch | 207 |
| 6.5 | Memory Management of Stacks | 211 |
| 6.6 | Bibliographical Notes | 215 |
| | References | 216 |
| 7 | Working with Sequential Versions | 219 |
| 7.1 | The Transformation Set | 222 |
| 7.1.1 | parfor Transformation | 223 |
| 7.1.2 | parblock Transformation | 223 |
| 7.1.3 | Privatization Transformation | 223 |
| 7.1.4 | Chunk Transformations | 224 |
| 7.1.5 | Loop Reorder Transformation | 225 |
| 7.1.6 | Asynchronous Transformation | 226 |
| 7.1.7 | Localization Transformation | 227 |
| 7.2 | Loop Transformations | 228 |
| 7.2.1 | Loop Interchange | 228 |
| 7.2.2 | Loop Distribution and Loop Fusion | 228 |
| 7.3 | Case Study: Transitive Closure | 230 |
| 7.4 | Case Study: Matrix Multiplication | 236 |
| 7.5 | Case Study: PDE—Partial Differential Equations | 238 |
| 7.6 | Case Study: Dynamic Chunking | 241 |
| 7.7 | Case Study: Using Pointer Arrays | 244 |
| 7.8 | Parallelization of Sequential Loops | 245 |
| 7.9 | Lamport's Diagonalization | 246 |

| | |
|--|------------|
| 7.10 Exercises | 252 |
| 7.10.1 Q1 | 252 |
| 7.10.2 Q2 | 255 |
| 7.11 Bibliographic Notes | 256 |
| References | 257 |
| 8 Performance and Overhead Measurements | 259 |
| 8.1 Introduction | 259 |
| 8.2 Processing Rates | 260 |
| 8.3 The Effect of MESI/MOESI Bus Transactions on the Processing Rates | 261 |
| 8.4 The Effect of Hardware Threads | 263 |
| 8.5 Initial Measurements of MESI | 266 |
| 8.6 Remote Access Penalty | 267 |
| 8.7 Wrapping MESI Overhead by Local Accesses | 269 |
| 8.8 Experimenting with the MESI's Protocol States | 272 |
| 8.9 Spawning New Threads | 273 |
| 8.10 Bibliographic Notes | 275 |
| References | 276 |

<http://www.springer.com/978-1-4471-2163-3>

Multicore Programming Using the ParC Language

Ben-Asher, Y.

2012, XIV, 277 p. 67 illus., Softcover

ISBN: 978-1-4471-2163-3