
DESIGN AND ANALYSIS OF DISTRIBUTED ALGORITHMS

Nicola Santoro

Carleton University, Ottawa, Canada



WILEY-INTERSCIENCE
A JOHN WILEY & SONS, INC., PUBLICATION

CONTENTS

| | |
|--|------------|
| Preface | xiv |
| 1. Distributed Computing Environments | 1 |
| 1.1 Entities | 1 |
| 1.2 Communication | 4 |
| 1.3 Axioms and Restrictions | 4 |
| 1.3.1 Axioms | 5 |
| 1.3.2 Restrictions | 6 |
| 1.4 Cost and Complexity | 9 |
| 1.4.1 Amount of Communication Activities | 9 |
| 1.4.2 Time | 10 |
| 1.5 An Example: Broadcasting | 10 |
| 1.6 States and Events | 14 |
| 1.6.1 Time and Events | 14 |
| 1.6.2 States and Configurations | 16 |
| 1.7 Problems and Solutions (★) | 17 |
| 1.8 Knowledge | 19 |
| 1.8.1 Levels of Knowledge | 19 |
| 1.8.2 Types of Knowledge | 21 |
| 1.9 Technical Considerations | 22 |
| 1.9.1 Messages | 22 |
| 1.9.2 Protocol | 23 |
| 1.9.3 Communication Mechanism | 24 |
| 1.10 Summary of Definitions | 25 |
| 1.11 Bibliographical Notes | 25 |
| 1.12 Exercises, Problems, and Answers | 26 |
| 1.12.1 Exercises and Problems | 26 |
| 1.12.2 Answers to Exercises | 27 |
| 2. Basic Problems And Protocols | 29 |
| 2.1 Broadcast | 29 |
| 2.1.1 The Problem | 29 |
| 2.1.2 Cost of Broadcasting | 30 |
| 2.1.3 Broadcasting in Special Networks | 32 |

| | | |
|-----------|---|-----------|
| 2.2 | Wake-Up | 36 |
| 2.2.1 | Generic Wake-Up | 36 |
| 2.2.2 | Wake-Up in Special Networks | 37 |
| 2.3 | Traversal | 41 |
| 2.3.1 | Depth-First Traversal | 42 |
| 2.3.2 | Hacking (★) | 44 |
| 2.3.3 | Traversal in Special Networks | 49 |
| 2.3.4 | Considerations on Traversal | 50 |
| 2.4 | Practical Implications: Use a Subnet | 51 |
| 2.5 | Constructing a Spanning Tree | 52 |
| 2.5.1 | SPT Construction with a Single Initiator: Shout | 53 |
| 2.5.2 | Other SPT Constructions with Single Initiator | 58 |
| 2.5.3 | Considerations on the Constructed Tree | 60 |
| 2.5.4 | Application: Better Traversal | 62 |
| 2.5.5 | Spanning-Tree Construction with Multiple Initiators | 62 |
| 2.5.6 | Impossibility Result | 63 |
| 2.5.7 | SPT with Initial Distinct Values | 65 |
| 2.6 | Computations in Trees | 70 |
| 2.6.1 | Saturation: A Basic Technique | 71 |
| 2.6.2 | Minimum Finding | 74 |
| 2.6.3 | Distributed Function Evaluation | 76 |
| 2.6.4 | Finding Eccentricities | 78 |
| 2.6.5 | Center Finding | 81 |
| 2.6.6 | Other Computations | 84 |
| 2.6.7 | Computing in Rooted Trees | 85 |
| 2.7 | Summary | 89 |
| 2.7.1 | Summary of Problems | 89 |
| 2.7.2 | Summary of Techniques | 90 |
| 2.8 | Bibliographical Notes | 90 |
| 2.9 | Exercises, Problems, and Answers | 91 |
| 2.9.1 | Exercises | 91 |
| 2.9.2 | Problems | 95 |
| 2.9.3 | Answers to Exercises | 95 |
| 3. | Election | 99 |
| 3.1 | Introduction | 99 |
| 3.1.1 | Impossibility Result | 99 |
| 3.1.2 | Additional Restrictions | 100 |
| 3.1.3 | Solution Strategies | 101 |
| 3.2 | Election in Trees | 102 |
| 3.3 | Election in Rings | 104 |
| 3.3.1 | All the Way | 105 |

| | |
|--|------------|
| 3.3.2 As Far As It Can | 109 |
| 3.3.3 Controlled Distance | 115 |
| 3.3.4 Electoral Stages | 122 |
| 3.3.5 Stages with Feedback | 127 |
| 3.3.6 Alternating Steps | 130 |
| 3.3.7 Unidirectional Protocols | 134 |
| 3.3.8 Limits to Improvements (★) | 150 |
| 3.3.9 Summary and Lessons | 157 |
| 3.4 Election in Mesh Networks | 158 |
| 3.4.1 Meshes | 158 |
| 3.4.2 Tori | 161 |
| 3.5 Election in Cube Networks | 166 |
| 3.5.1 Oriented Hypercubes | 166 |
| 3.5.2 Unoriented Hypercubes | 174 |
| 3.6 Election in Complete Networks | 174 |
| 3.6.1 Stages and Territory | 174 |
| 3.6.2 Surprising Limitation | 177 |
| 3.6.3 Harvesting the Communication Power | 180 |
| 3.7 Election in Chordal Rings (★) | 183 |
| 3.7.1 Chordal Rings | 183 |
| 3.7.2 Lower Bounds | 184 |
| 3.8 Universal Election Protocols | 185 |
| 3.8.1 Mega-Merger | 185 |
| 3.8.2 Analysis of Mega-Merger | 193 |
| 3.8.3 YO-YO | 199 |
| 3.8.4 Lower Bounds and Equivalences | 209 |
| 3.9 Bibliographical Notes | 212 |
| 3.10 Exercises, Problems, and Answers | 214 |
| 3.10.1 Exercises | 214 |
| 3.10.2 Problems | 220 |
| 3.10.3 Answers to Exercises | 222 |
| 4. Message Routing and Shortest Paths | 225 |
| 4.1 Introduction | 225 |
| 4.2 Shortest Path Routing | 226 |
| 4.2.1 Gossiping the Network Maps | 226 |
| 4.2.2 Iterative Construction of Routing Tables | 228 |
| 4.2.3 Constructing Shortest-Path Spanning Tree | 230 |
| 4.2.4 Constructing All-Pairs Shortest Paths | 237 |
| 4.2.5 Min-Hop Routing | 240 |
| 4.2.6 Suboptimal Solutions: Routing Trees | 250 |
| 4.3 Coping with Changes | 253 |
| 4.3.1 Adaptive Routing | 253 |

| | | |
|-----------|---|------------|
| 4.3.2 | Fault-Tolerant Tables | 255 |
| 4.3.3 | On Correctness and Guarantees | 259 |
| 4.4 | Routing in Static Systems: Compact Tables | 261 |
| 4.4.1 | The Size of Routing Tables | 261 |
| 4.4.2 | Interval Routing | 262 |
| 4.5 | Bibliographical Notes | 267 |
| 4.6 | Exercises, Problems, and Answers | 269 |
| 4.6.1 | Exercises | 269 |
| 4.6.2 | Problems | 274 |
| 4.6.3 | Answers to Exercises | 274 |
| 5. | Distributed Set Operations | 277 |
| 5.1 | Introduction | 277 |
| 5.2 | Distributed Selection | 279 |
| 5.2.1 | Order Statistics | 279 |
| 5.2.2 | Selection in a Small Data Set | 280 |
| 5.2.3 | Simple Case: Selection Among Two Sites | 282 |
| 5.2.4 | General Selection Strategy: RankSelect | 287 |
| 5.2.5 | Reducing the Worst Case: ReduceSelect | 292 |
| 5.3 | Sorting a Distributed Set | 297 |
| 5.3.1 | Distributed Sorting | 297 |
| 5.3.2 | Special Case: Sorting on a Ordered Line | 299 |
| 5.3.3 | Removing the Topological Constraints: Complete Graph | 303 |
| 5.3.4 | Basic Limitations | 306 |
| 5.3.5 | Efficient Sorting: SelectSort | 309 |
| 5.3.6 | Unrestricted Sorting | 312 |
| 5.4 | Distributed Sets Operations | 315 |
| 5.4.1 | Operations on Distributed Sets | 315 |
| 5.4.2 | Local Structure | 317 |
| 5.4.3 | Local Evaluation (*) | 319 |
| 5.4.4 | Global Evaluation | 322 |
| 5.4.5 | Operational Costs | 323 |
| 5.5 | Bibliographical Notes | 323 |
| 5.6 | Exercises, Problems, and Answers | 324 |
| 5.6.1 | Exercises | 324 |
| 5.6.2 | Problems | 329 |
| 5.6.3 | Answers to Exercises | 329 |
| 6. | Synchronous Computations | 333 |
| 6.1 | Synchronous Distributed Computing | 333 |
| 6.1.1 | Fully Synchronous Systems | 333 |

| | | |
|-----------|---|------------|
| 6.1.2 | Clocks and Unit of Time | 334 |
| 6.1.3 | Communication Delays and Size of Messages | 336 |
| 6.1.4 | On the Unique Nature of Synchronous Computations | 336 |
| 6.1.5 | The Cost of Synchronous Protocols | 342 |
| 6.2 | Communicators, Pipeline, and Transformers | 343 |
| 6.2.1 | Two-Party Communication | 344 |
| 6.2.2 | Pipeline | 353 |
| 6.2.3 | Transformers | 357 |
| 6.3 | Min-Finding and Election: Waiting and Guessing | 360 |
| 6.3.1 | Waiting | 360 |
| 6.3.2 | Guessing | 370 |
| 6.3.3 | Double Wait: Integrating Waiting and Guessing | 378 |
| 6.4 | Synchronization Problems: Reset, Unison, and Firing Squad | 385 |
| 6.4.1 | Reset/Wake-up | 386 |
| 6.4.2 | Unison | 387 |
| 6.4.3 | Firing Squad | 389 |
| 6.5 | Bibliographical Notes | 391 |
| 6.6 | Exercises, Problems, and Answers | 392 |
| 6.6.1 | Exercises | 392 |
| 6.6.2 | Problems | 398 |
| 6.6.3 | Answers to Exercises | 400 |
| 7. | Computing in Presence of Faults | 408 |
| 7.1 | Introduction | 408 |
| 7.1.1 | Faults and Failures | 408 |
| 7.1.2 | Modeling Faults | 410 |
| 7.1.3 | Topological Factors | 413 |
| 7.1.4 | Fault Tolerance, Agreement, and Common Knowledge | 415 |
| 7.2 | The Crushing Impact of Failures | 417 |
| 7.2.1 | Node Failures: Single-Fault Disaster | 417 |
| 7.2.2 | Consequences of the Single-Fault Disaster | 424 |
| 7.3 | Localized Entity Failures: Using Synchrony | 425 |
| 7.3.1 | Synchronous Consensus with Crash Failures | 426 |
| 7.3.2 | Synchronous Consensus with Byzantine Failures | 430 |
| 7.3.3 | Limit to Number of Byzantine Entities for Agreement | 435 |
| 7.3.4 | From Boolean to General Byzantine Agreement | 438 |
| 7.3.5 | Byzantine Agreement in Arbitrary Graphs | 440 |
| 7.4 | Localized Entity Failures: Using Randomization | 443 |
| 7.4.1 | Random Actions and Coin Flips | 443 |
| 7.4.2 | Randomized Asynchronous Consensus: Crash Failures | 444 |
| 7.4.3 | Concluding Remarks | 449 |

| | | |
|-----------|--|------------|
| 7.5 | Localized Entity Failures: Using Fault Detection | 449 |
| 7.5.1 | Failure Detectors and Their Properties | 450 |
| 7.5.2 | The Weakest Failure Detector | 452 |
| 7.6 | Localized Entity Failures: Preexecution Failures | 454 |
| 7.6.1 | Partial Reliability | 454 |
| 7.6.2 | Example: Election in Complete Network | 455 |
| 7.7 | Localized Link Failures | 457 |
| 7.7.1 | A Tale of Two Synchronous Generals | 458 |
| 7.7.2 | Computing With Faulty Links | 461 |
| 7.7.3 | Concluding Remarks | 466 |
| 7.7.4 | Considerations on Localized Entity Failures | 466 |
| 7.8 | Ubiquitous Faults | 467 |
| 7.8.1 | Communication Faults and Agreement | 467 |
| 7.8.2 | Limits to Number of Ubiquitous Faults for Majority | 468 |
| 7.8.3 | Unanimity in Spite of Ubiquitous Faults | 475 |
| 7.8.4 | Tightness | 485 |
| 7.9 | Bibliographical Notes | 486 |
| 7.10 | Exercises, Problems, and Answers | 488 |
| 7.10.1 | Exercises | 488 |
| 7.10.2 | Problems | 492 |
| 7.10.3 | Answers to Exercises | 493 |
| 8. | Detecting Stable Properties | 500 |
| 8.1 | Introduction | 500 |
| 8.2 | Deadlock Detection | 500 |
| 8.2.1 | Deadlock | 500 |
| 8.2.2 | Detecting Deadlock: Wait-for Graph | 501 |
| 8.2.3 | Single-Request Systems | 503 |
| 8.2.4 | Multiple-Requests Systems | 505 |
| 8.2.5 | Dynamic Wait-for Graphs | 512 |
| 8.2.6 | Other Requests Systems | 516 |
| 8.3 | Global Termination Detection | 518 |
| 8.3.1 | A Simple Solution: Repeated Termination Queries | 519 |
| 8.3.2 | Improved Protocols: Shrink | 523 |
| 8.3.3 | Concluding Remarks | 525 |
| 8.4 | Global Stable Property Detection | 526 |
| 8.4.1 | General Strategy | 526 |
| 8.4.2 | Time Cuts and Consistent Snapshots | 527 |
| 8.4.3 | Computing A Consistent Snapshot | 530 |
| 8.4.4 | Summary: Putting All Together | 531 |
| 8.5 | Bibliographical Notes | 532 |

| | |
|---|------------|
| 8.6 Exercises, Problems, and Answers | 534 |
| 8.6.1 Exercises | 534 |
| 8.6.2 Problems | 536 |
| 8.6.3 Answers to Exercises | 538 |
| 9. Continuous Computations | 541 |
| 9.1 Introduction | 541 |
| 9.2 Keeping Virtual Time | 542 |
| 9.2.1 Virtual Time and Causal Order | 542 |
| 9.2.2 Causal Order: Counter Clocks | 544 |
| 9.2.3 Complete Causal Order: Vector Clocks | 545 |
| 9.2.4 Concluding Remarks | 548 |
| 9.3 Distributed Mutual Exclusion | 549 |
| 9.3.1 The Problem | 549 |
| 9.3.2 A Simple and Efficient Solution | 550 |
| 9.3.3 Traversing the Network | 551 |
| 9.3.4 Managing a Distributed Queue | 554 |
| 9.3.5 Decentralized Permissions | 559 |
| 9.3.6 Mutual Exclusion in Complete Graphs: Quorum | 561 |
| 9.3.7 Concluding Remarks | 564 |
| 9.4 Deadlock: System Detection and Resolution | 566 |
| 9.4.1 System Detection and Resolution | 566 |
| 9.4.2 Detection and Resolution in Single-Request Systems | 567 |
| 9.4.3 Detection and Resolution in Multiple-Requests Systems | 568 |
| 9.5 Bibliographical Notes | 569 |
| 9.6 Exercises, Problems, and Answers | 570 |
| 9.6.1 Exercises | 570 |
| 9.6.2 Problems | 572 |
| 9.6.3 Answers to Exercises | 573 |
| Index | 577 |