Distributed Computing
Principles, Algorithms, and Systems

Ajay D. Kshemkalyani
University of Illinois at Chicago, Chicago

and

Mukesh Singhal
University of Kentucky, Lexington

CAMBRIDGE UNIVERSITY PRESS
Contents

Preface

1 Introduction
1.1 Definition
1.2 Relation to computer system components
1.3 Motivation
1.4 Relation to parallel multiprocessor/multicomputer systems
1.5 Message-passing systems versus shared memory systems
1.6 Primitives for distributed communication
1.7 Synchronous versus asynchronous executions
1.8 Design issues and challenges
1.9 Selection and coverage of topics
1.10 Chapter summary
1.11 Exercises
1.12 Notes on references
References

2 A model of distributed computations
2.1 A distributed program
2.2 A model of distributed executions
2.3 Models of communication networks
2.4 Global state of a distributed system
2.5 Cuts of a distributed computation
2.6 Past and future cones of an event
2.7 Models of process communications
2.8 Chapter summary
2.9 Exercises
2.10 Notes on references
References
### 3 Logical time  
3.1 Introduction 50  
3.2 A framework for a system of logical clocks 52  
3.3 Scalar time 53  
3.4 Vector time 55  
3.5 Efficient implementations of vector clocks 59  
3.6 Jard–Jourdán’s adaptive technique 65  
3.7 Matrix time 68  
3.8 Virtual time 69  
3.9 Physical clock synchronization: NTP 78  
3.10 Chapter summary 81  
3.11 Exercises 84  
3.12 Notes on references 84  
References 84

### 4 Global state and snapshot recording algorithms  
4.1 Introduction 87  
4.2 System model and definitions 90  
4.3 Snapshot algorithms for FIFO channels 93  
4.4 Variations of the Chandy–Lamport algorithm 97  
4.5 Snapshot algorithms for non-FIFO channels 101  
4.6 Snapshots in a causal delivery system 106  
4.7 Monitoring global state 109  
4.8 Necessary and sufficient conditions for consistent global snapshots 110  
4.9 Finding consistent global snapshots in a distributed computation 114  
4.10 Chapter summary 121  
4.11 Exercises 122  
4.12 Notes on references 122  
References 123

### 5 Terminology and basic algorithms  
5.1 Topology abstraction and overlays 126  
5.2 Classifications and basic concepts 128  
5.3 Complexity measures and metrics 135  
5.4 Program structure 137  
5.5 Elementary graph algorithms 138  
5.6 Synchronizers 163  
5.7 Maximal independent set (MIS) 169  
5.8 Connected dominating set 171  
5.9 Compact routing tables 172  
5.10 Leader election 174
### 6 Message ordering and group communication

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.1 Message ordering paradigms</td>
<td>190</td>
</tr>
<tr>
<td>6.2 Asynchronous execution with synchronous communication</td>
<td>195</td>
</tr>
<tr>
<td>6.3 Synchronous program order on an asynchronous system</td>
<td>200</td>
</tr>
<tr>
<td>6.4 Group communication</td>
<td>205</td>
</tr>
<tr>
<td>6.5 Causal order (CO)</td>
<td>206</td>
</tr>
<tr>
<td>6.6 Total order</td>
<td>215</td>
</tr>
<tr>
<td>6.7 A nomenclature for multicast</td>
<td>220</td>
</tr>
<tr>
<td>6.8 Propagation trees for multicast</td>
<td>221</td>
</tr>
<tr>
<td>6.9 Classification of application-level multicast algorithms</td>
<td>225</td>
</tr>
<tr>
<td>6.10 Semantics of fault-tolerant group communication</td>
<td>228</td>
</tr>
<tr>
<td>6.11 Distributed multicast algorithms at the network layer</td>
<td>230</td>
</tr>
<tr>
<td>6.12 Chapter summary</td>
<td>236</td>
</tr>
<tr>
<td>6.13 Exercises</td>
<td>236</td>
</tr>
<tr>
<td>6.14 Notes on references</td>
<td>238</td>
</tr>
</tbody>
</table>

### 7 Termination detection

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.1 Introduction</td>
<td>241</td>
</tr>
<tr>
<td>7.2 System model of a distributed computation</td>
<td>242</td>
</tr>
<tr>
<td>7.3 Termination detection using distributed snapshots</td>
<td>243</td>
</tr>
<tr>
<td>7.4 Termination detection by weight throwing</td>
<td>245</td>
</tr>
<tr>
<td>7.5 A spanning-tree-based termination detection algorithm</td>
<td>247</td>
</tr>
<tr>
<td>7.6 Message-optimal termination detection</td>
<td>253</td>
</tr>
<tr>
<td>7.7 Termination detection in a very general distributed computing model</td>
<td>257</td>
</tr>
<tr>
<td>7.8 Termination detection in the atomic computation model</td>
<td>263</td>
</tr>
<tr>
<td>7.9 Termination detection in a faulty distributed system</td>
<td>272</td>
</tr>
<tr>
<td>7.10 Chapter summary</td>
<td>279</td>
</tr>
<tr>
<td>7.11 Exercises</td>
<td>279</td>
</tr>
<tr>
<td>7.12 Notes on references</td>
<td>280</td>
</tr>
</tbody>
</table>

### 8 Reasoning with knowledge

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>8.1 The muddy children puzzle</td>
<td>282</td>
</tr>
<tr>
<td>8.2 Logic of knowledge</td>
<td>283</td>
</tr>
<tr>
<td>Section</td>
<td>Title</td>
</tr>
<tr>
<td>---------</td>
<td>-------</td>
</tr>
<tr>
<td>8.3</td>
<td>Knowledge in synchronous systems</td>
</tr>
<tr>
<td>8.4</td>
<td>Knowledge in asynchronous systems</td>
</tr>
<tr>
<td>8.5</td>
<td>Knowledge transfer</td>
</tr>
<tr>
<td>8.6</td>
<td>Knowledge and clocks</td>
</tr>
<tr>
<td>8.7</td>
<td>Chapter summary</td>
</tr>
<tr>
<td>8.8</td>
<td>Exercises</td>
</tr>
<tr>
<td>8.9</td>
<td>Notes on references</td>
</tr>
<tr>
<td>8.10</td>
<td>References</td>
</tr>
<tr>
<td>9</td>
<td>Distributed mutual exclusion algorithms</td>
</tr>
<tr>
<td>9.1</td>
<td>Introduction</td>
</tr>
<tr>
<td>9.2</td>
<td>Preliminaries</td>
</tr>
<tr>
<td>9.3</td>
<td>Lamport’s algorithm</td>
</tr>
<tr>
<td>9.4</td>
<td>Ricart–Agrawala algorithm</td>
</tr>
<tr>
<td>9.5</td>
<td>Singhal’s dynamic information-structure algorithm</td>
</tr>
<tr>
<td>9.6</td>
<td>Lodha and Kshemkalyani’s fair mutual exclusion algorithm</td>
</tr>
<tr>
<td>9.7</td>
<td>Quorum-based mutual exclusion algorithms</td>
</tr>
<tr>
<td>9.8</td>
<td>Maekawa’s algorithm</td>
</tr>
<tr>
<td>9.9</td>
<td>Agarwal–El Abbadi quorum-based algorithm</td>
</tr>
<tr>
<td>9.10</td>
<td>Token-based algorithms</td>
</tr>
<tr>
<td>9.11</td>
<td>Suzuki–Kasami’s broadcast algorithm</td>
</tr>
<tr>
<td>9.12</td>
<td>Raymond’s tree-based algorithm</td>
</tr>
<tr>
<td>9.13</td>
<td>Chapter summary</td>
</tr>
<tr>
<td>9.14</td>
<td>Exercises</td>
</tr>
<tr>
<td>9.15</td>
<td>Notes on references</td>
</tr>
<tr>
<td>9.16</td>
<td>References</td>
</tr>
<tr>
<td>10</td>
<td>Deadlock detection in distributed systems</td>
</tr>
<tr>
<td>10.1</td>
<td>Introduction</td>
</tr>
<tr>
<td>10.2</td>
<td>System model</td>
</tr>
<tr>
<td>10.3</td>
<td>Preliminaries</td>
</tr>
<tr>
<td>10.4</td>
<td>Models of deadlocks</td>
</tr>
<tr>
<td>10.5</td>
<td>Knapp’s classification of distributed deadlock detection algorithms</td>
</tr>
<tr>
<td>10.6</td>
<td>Mitchell and Merritt’s algorithm for the single-resource model</td>
</tr>
<tr>
<td>10.7</td>
<td>Chandy–Misra–Haas algorithm for the AND model</td>
</tr>
<tr>
<td>10.8</td>
<td>Chandy–Misra–Haas algorithm for the OR model</td>
</tr>
<tr>
<td>10.9</td>
<td>Kshemkalyani–Singhal algorithm for the P-out-of-Q model</td>
</tr>
<tr>
<td>10.10</td>
<td>Chapter summary</td>
</tr>
<tr>
<td>10.11</td>
<td>Exercises</td>
</tr>
<tr>
<td>10.12</td>
<td>Notes on references</td>
</tr>
<tr>
<td>10.13</td>
<td>References</td>
</tr>
</tbody>
</table>
Contents

11 Global predicate detection
11.1 Stable and unstable predicates
11.2 Modalities on predicates
11.3 Centralized algorithm for relational predicates
11.4 Conjunctive predicates
11.5 Distributed algorithms for conjunctive predicates
11.6 Further classification of predicates
11.7 Chapter summary
11.8 Exercises
11.9 Notes on references
References

12 Distributed shared memory
12.1 Abstraction and advantages
12.2 Memory consistency modes
12.3 Shared memory mutual exclusion
12.4 Wait-freedom
12.5 Register hierarchy and wait-free simulations
12.6 Wait-free atomic snapshots of shared objects
12.7 Chapter summary
12.8 Exercises
12.9 Notes on references
References

13 Checkpointing and rollback recovery
13.1 Introduction
13.2 Background and definitions
13.3 Issues in failure recovery
13.4 Checkpoint-based recovery
13.5 Log-based rollback recovery
13.6 Koo-Toueg coordinated checkpointing algorithm
13.7 Juang–Venkatesan algorithm for asynchronous checkpointing and recovery
13.8 Manivannan–Singhal quasi-synchronous checkpointing algorithm
13.9 Peterson–Kearns algorithm based on vector time
13.10 Helary–Mostefaoui–Netzer–Raynal communication-induced protocol
13.11 Chapter summary
13.12 Exercises
13.13 Notes on references
References
## Contents

### 14 Consensus and agreement algorithms
- 14.1 Problem definition 510
- 14.2 Overview of results 514
- 14.3 Agreement in a failure-free system (synchronous or asynchronous) 515
- 14.4 Agreement in (message-passing) synchronous systems with failures 516
- 14.5 Agreement in asynchronous message-passing systems with failures 529
- 14.6 Wait-free shared memory consensus in asynchronous systems 544
- 14.7 Chapter summary 562
- 14.8 Exercises 563
- 14.9 Notes on references 564
- References 565

### 15 Failure detectors
- 15.1 Introduction 567
- 15.2 Unreliable failure detectors 568
- 15.3 The consensus problem 577
- 15.4 Atomic broadcast 583
- 15.5 A solution to atomic broadcast 584
- 15.6 The weakest failure detectors to solve fundamental agreement problems 585
- 15.7 An implementation of a failure detector 589
- 15.8 An adaptive failure detection protocol 591
- 15.9 Exercises 596
- 15.10 Notes on references 596
- References 596

### 16 Authentication in distributed systems
- 16.1 Introduction 598
- 16.2 Background and definitions 599
- 16.3 Protocols based on symmetric cryptosystems 602
- 16.4 Protocols based on asymmetric cryptosystems 615
- 16.5 Password-based authentication 622
- 16.6 Authentication protocol failures 625
- 16.7 Chapter summary 626
- 16.8 Exercises 627
- 16.9 Notes on references 627
- References 628

### 17 Self-stabilization
- 17.1 Introduction 631
- 17.2 System model 632
17.3 Definition of self-stabilization 634
17.4 Issues in the design of self-stabilization algorithms 636
17.5 Methodologies for designing self-stabilizing systems 647
17.6 Communication protocols 649
17.7 Self-stabilizing distributed spanning trees 650
17.8 Self-stabilizing algorithms for spanning-tree construction 652
17.9 An anonymous self-stabilizing algorithm for 1-maximal independent set in trees 657
17.10 A probabilistic self-stabilizing leader election algorithm 660
17.11 The role of compilers in self-stabilization 662
17.12 Self-stabilization as a solution to fault tolerance 665
17.13 Factors preventing self-stabilization 667
17.14 Limitations of self-stabilization 668
17.15 Chapter summary 670
17.16 Exercises 670
17.17 Notes on references 671
References 671

18 Peer-to-peer computing and overlay graphs 677
18.1 Introduction 677
18.2 Data indexing and overlays 679
18.3 Unstructured overlays 681
18.4 Chord distributed hash table 688
18.5 Content addressable networks (CAN) 695
18.6 Tapestry 701
18.7 Some other challenges in P2P system design 708
18.8 Tradeoffs between table storage and route lengths 710
18.9 Graph structures of complex networks 712
18.10 Internet graphs 714
18.11 Generalized random graph networks 720
18.12 Small-world networks 720
18.13 Scale-free networks 721
18.14 Evolving networks 723
18.15 Chapter summary 727
18.16 Exercises 727
18.17 Notes on references 728
References 729

Index 731