

# Contents

## *Preface*

*page xv*

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Definition	1
1.2	Relation to computer system components	2
1.3	Motivation	3
1.4	Relation to parallel multiprocessor/multicomputer systems	5
1.5	Message-passing systems versus shared memory systems	13
1.6	Primitives for distributed communication	14
1.7	Synchronous versus asynchronous executions	19
1.8	Design issues and challenges	22
1.9	Selection and coverage of topics	33
1.10	Chapter summary	34
1.11	Exercises	35
1.12	Notes on references	36
	References	37
<b>2</b>	<b>A model of distributed computations</b>	<b>39</b>
2.1	A distributed program	39
2.2	A model of distributed executions	40
2.3	Models of communication networks	42
2.4	Global state of a distributed system	43
2.5	Cuts of a distributed computation	45
2.6	Past and future cones of an event	46
2.7	Models of process communications	47
2.8	Chapter summary	48
2.9	Exercises	48
2.10	Notes on references	48
	References	49

<b>3</b>	<b>Logical time</b>	<b>50</b>
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-Jourdan'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
<b>4</b>	<b>Global state and snapshot recording algorithms</b>	<b>87</b>
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
<b>5</b>	<b>Terminology and basic algorithms</b>	<b>126</b>
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

5.11	Challenges in designing distributed graph algorithms	17
5.12	Object replication problems	17
5.13	Chapter summary	18
5.14	Exercises	18
5.15	Notes on references	18
	References	18
<b>6</b>	<b>Message ordering and group communication</b>	<b>18</b>
6.1	Message ordering paradigms	19
6.2	Asynchronous execution with synchronous communication	19
6.3	Synchronous program order on an asynchronous system	20
6.4	Group communication	20
6.5	Causal order (CO)	20
6.6	Total order	21
6.7	A nomenclature for multicast	22
6.8	Propagation trees for multicast	22
6.9	Classification of application-level multicast algorithms	22
6.10	Semantics of fault-tolerant group communication	22
6.11	Distributed multicast algorithms at the network layer	23
6.12	Chapter summary	23
6.13	Exercises	23
6.14	Notes on references	23
	References	23
<b>7</b>	<b>Termination detection</b>	<b>24</b>
7.1	Introduction	24
7.2	System model of a distributed computation	24
7.3	Termination detection using distributed snapshots	24
7.4	Termination detection by weight throwing	24
7.5	A spanning-tree-based termination detection algorithm	24
7.6	Message-optimal termination detection	25
7.7	Termination detection in a very general distributed computing model	25
7.8	Termination detection in the atomic computation model	26
7.9	Termination detection in a faulty distributed system	27
7.10	Chapter summary	27
7.11	Exercises	27
7.12	Notes on references	28
	References	28
<b>8</b>	<b>Reasoning with knowledge</b>	<b>28</b>
8.1	The muddy children puzzle	28
8.2	Logic of knowledge	28

3.3	Knowledge in synchronous systems	289
3.4	Knowledge in asynchronous systems	290
3.5	Knowledge transfer	298
3.6	Knowledge and clocks	300
3.7	Chapter summary	301
3.8	Exercises	302
3.9	Notes on references	303
	References	303
<b>9</b>	<b>Distributed mutual exclusion algorithms</b>	<b>305</b>
9.1	Introduction	305
9.2	Preliminaries	306
9.3	Lamport's algorithm	309
9.4	Ricart-Agrawala algorithm	312
9.5	Singhal's dynamic information-structure algorithm	315
9.6	Lodha and Kshemkalyani's fair mutual exclusion algorithm	321
9.7	Quorum-based mutual exclusion algorithms	327
9.8	Maekawa's algorithm	328
9.9	Agarwal-El Abbadi quorum-based algorithm	331
9.10	Token-based algorithms	336
9.11	Suzuki-Kasami's broadcast algorithm	336
9.12	Raymond's tree-based algorithm	339
9.13	Chapter summary	348
9.14	Exercises	348
9.15	Notes on references	349
	References	350
<b>10</b>	<b>Deadlock detection in distributed systems</b>	<b>352</b>
10.1	Introduction	352
10.2	System model	352
10.3	Preliminaries	353
10.4	Models of deadlocks	355
10.5	Knapp's classification of distributed deadlock detection algorithms	358
10.6	Mitchell and Merritt's algorithm for the single-resource model	360
10.7	Chandy-Misra-Haas algorithm for the AND model	362
10.8	Chandy-Misra-Haas algorithm for the OR model	364
10.9	Kshemkalyani-Singhal algorithm for the $P$ -out-of- $Q$ model	365
10.10	Chapter summary	374
10.11	Exercises	375
10.12	Notes on references	375
	References	376

<b>11</b>	<b>Global predicate detection</b>	<b>379</b>
11.1	Stable and unstable predicates	379
11.2	Modalities on predicates	382
11.3	Centralized algorithm for relational predicates	384
11.4	Conjunctive predicates	388
11.5	Distributed algorithms for conjunctive predicates	395
11.6	Further classification of predicates	404
11.7	Chapter summary	405
11.8	Exercises	406
11.9	Notes on references	407
	References	408
<b>12</b>	<b>Distributed shared memory</b>	<b>410</b>
12.1	Abstraction and advantages	410
12.2	Memory consistency models	413
12.3	Shared memory mutual exclusion	427
12.4	Wait-freedom	434
12.5	Register hierarchy and wait-free simulations	434
12.6	Wait-free atomic snapshots of shared objects	447
12.7	Chapter summary	451
12.8	Exercises	452
12.9	Notes on references	453
	References	454
<b>13</b>	<b>Checkpointing and rollback recovery</b>	<b>456</b>
13.1	Introduction	456
13.2	Background and definitions	457
13.3	Issues in failure recovery	462
13.4	Checkpoint-based recovery	464
13.5	Log-based rollback recovery	470
13.6	Koo-Toueg coordinated checkpointing algorithm	476
13.7	Juang-Venkatesan algorithm for asynchronous checkpointing and recovery	478
13.8	Manivannan-Singhal quasi-synchronous checkpointing algorithm	483
13.9	Peterson-Kearns algorithm based on vector time	492
13.10	Helary-Mostefaoui-Netzer-Raynal communication-induced protocol	499
13.11	Chapter summary	505
13.12	Exercises	506
13.13	Notes on references	506
	References	507

<b>14</b>	<b>Consensus and agreement algorithms</b>	<b>510</b>
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
<b>15</b>	<b>Failure detectors</b>	<b>567</b>
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
<b>16</b>	<b>Authentication in distributed systems</b>	<b>598</b>
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
<b>17</b>	<b>Self-stabilization</b>	<b>631</b>
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
<b>18</b>	<b>Peer-to-peer computing and overlay graphs</b>	<b>677</b>
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
	<i>Index</i>	731