

Preface to the Second Edition

Distributed systems have witnessed phenomenal growth in the past few years. The declining cost of hardware, the advancements in communication technology, the explosive growth of the Internet and our ever-increasing dependence on networks for a wide range of applications ranging from social communication to financial transactions have contributed to this growth. The breakthroughs in embedded systems, nanotechnology, and wireless communication have opened up new frontiers of applications like sensor networks and wearable computers. The rapid growth of cloud computing and the growing importance of big data have changed the landscape of distributed computing.

Most applications in distributed computing center around a set of core subproblems. A proper understanding of these subproblems requires a background of the underlying theory and algorithmic issues. This book provides a balanced coverage of the foundational topics, and their relationship to real-life applications. The language has been kept as unobfuscated as possible – clarity has been given priority over formalism. The second edition fixes many of the problems in the first edition, adds new topics, and significantly upgrades the contents. The twenty-one chapters have been divided into five sections:

Section I (Chapters 1-2) deals with *background materials* that include various cloud computing platforms.

Section II (Chapter 3-6) presents *foundational topics*, which address system models, correctness criteria, and proof techniques.

Section III (Chapter 7-11) presents the core paradigms in distributed systems – these include logical clocks, distributed snapshots and debugging, deadlock and termination detection, election, and distributed graph algorithms.

Section IV (Chapters 12-17) addresses failures and fault-tolerance techniques in various applications – it covers consensus, transactions, group communication, replicated data management, and self-stabilization. Group communication and consensus have been included in this section since they are two of the primary beneficiaries of fault-tolerant designs. Finally,

Section V (Chapters 18-21) addresses a few real-world issues: these include distributed discrete-event simulation, security, sensor networks, social and peer-to-peer networks. Each chapter has a list of exercises that will challenge the readers (those tagged with * are the more challenging ones). A small number of these are programming exercises. Some exercises will encourage the readers to learn about outside materials.

The book is intended for use in a one-semester course at the senior undergraduate or the first-year graduate level. About 75% of the materials can be covered in one semester. Accordingly, the chapters can be picked and packaged in several different ways. A theory oriented offering is possible using Chapters 1-17. For a more practical flavor, use chapters 1-2, selected topics from chapters 3-16, and chapters 18-21, supplemented by a semester-long project chosen from replicated data management, sensor networks, group communication, discrete-event simulation, social or P2P networks. Here is a disclaimer: this book is *not* about programming distributed systems. Chapter 2 is only a high level description that we expect everyone to know, but is *not* an introduction to programming. If programming is the goal, then I encourage readers to look for other materials. There are several good books available for this.

Table of Contents
SECTION I. BACKGROUND MATERIALS

1	Introduction
1.1.	What is a distributed system
1.2.	Why distributed systems
1.3.	Examples of distributed systems
1.4.	Important issues in distributed systems
1.5.	Common subproblems
1.6.	Implementing a distributed system
1.7.	Parallel vs. distributed Systems
1.8.	Bibliographic Notes
2	Interprocess Communication: An Overview
2.1	Introduction
2.1.1	Processes and threads
2.1.2	Client Server Model
2.1.3	Middleware
2.2	Network Protocols
2.2.1	The Ethernet
2.2.2	Wireless networks
2.2.3	The OSI model
2.2.4	Internet protocol (IP)
2.2.5	Transport layer protocols
2.2.6	Interprocess communication using sockets
2.3	Naming
2.3.1	Domain Name Service
2.3.2	Naming Service for Mobile Clients
2.4	Remote Procedure call
2.4.1	Implementing RPC
2.4.2	SUN ONC/RPC
2.5	Remote Method Invocation
2.6	Messages
2.6.1	Transient and Persistent Messages
2.6.2	Streams
2.7	Web services
2.8	Event Notification
2.9	Virtualization: Cloud Computing
2.9.1	Classification of Cloud Services
2.9.2	MapReduce
2.9.3	Hadoop
2.10	Mobile Agents
2.11	Basic Group Communication Services
2.12	Concluding Remarks
2.13	Bibliographic Notes

SECTION II. FOUNDATIONAL TOPICS

3	Models for Communication
3.1.	The Need for a Model
3.2.	A Message-Passing Model for Interprocess Communication

3.2.1	Process actions
3.2.2	Channels
3.2.3	Synchronous vs. Asynchronous Systems
3.2.4	Real-time Systems
3.3	Shared Variables
3.3.1	Linda
3.4	Modeling mobile agents
3.5	Relationship Among Models
3.5.1	Strong and Weak Models
3.5.2	Implementing a FIFO channel using a non-FIFO channel
3.5.3	Implementing Message-passing using Shared Memory
3.5.4	Implementing Shared Memory using Message-passing
3.5.5	An impossibility result with channels
3.6	Classification Based on Special Properties
3.6.1	Reactive vs. Transformational Systems
3.6.2	Named vs. Anonymous Systems
3.7	Complexity Measures
3.8	Concluding Remarks
3.9	Bibliographic Notes
4.	Representing Distributed Algorithms: Syntax and Semantics
4.1	Introduction
4.2	Guarded Actions
4.3	Non-determinism
4.4	Atomic Operations
4.5	Fairness
4.6	Central and Distributed Schedulers
4.7	Concluding remarks
4.8	Bibliographic Notes
5	Program Correctness
5.1	Introduction
5.2	Correctness Criteria
5.2.1	Safety Properties
5.2.2	Liveness Properties
5.3	Correctness Proofs
5.3.1	A Review of Propositional Logic
5.3.2	A Overview of Predicate Logic
5.4	Assertional Reasoning: Proving safety Properties
5.5	Proving Liveness Properties using Well-founded Sets
5.6	Programming Logic
5.7	Predicate Transformers
5.8	Concluding Remarks
5.9	Bibliographic Notes
6	Time in a Distributed System
6.1	Introduction
6.1.1	The Physical time
6.1.2	Sequential and Concurrent Events
6.2	Logical clocks
6.3	Vector Clocks
6.4	Physical Clock Synchronization

6.4.1	Preliminary Definitions
6.4.2	Clock Reading Error
6.4.3	Algorithms for Internal Synchronization
6.4.4	Algorithms for External Synchronization
6.5	Concluding Remarks
6.6	Bibliographic Notes

SECTION III. IMPORTANT PARADIGMS

7	Mutual Exclusion
7.1.	Introduction
7.2.	Solution on Message-passing Systems
7.2.1	Lamport's Algorithm
7.2.2	Ricart-Agrawala's Algorithm
7.2.3	Maekawa's Algorithm
7.3	Token-passing algorithms
7.3.1	Suzuki-Kasami's Algorithm
7.3.2	Raymond's Algorithm
7.4	Solutions on the Shared Memory Model
7.4.1	Peterson's algorithm
7.5.	Mutual Exclusion using Special Instructions
7.5.1	Solutions using Test-and-Set
7.5.2	Solutions using Load-linked and Store-conditional
7.6	Group Mutual Exclusion
7.7	Concluding Remarks
7.8	Bibliographic Notes
8	Distributed Snapshot
8.1	Introduction
8.2	Properties of Consistent Snapshots
8.3	Chandy-Lamport's Algorithm
8.3.1	Two Examples
8.4	Lai-Yang's Algorithm
8.5	Distributed Debugging
8.6	Concluding Remarks
8.7	Bibliographic Notes
9	Global State Collection
9.1	Introduction
9.2	An Elementary Broadcasting Algorithm
9.3	Termination Detection Algorithms
9.3.1	The Dijkstra-Scholten Algorithm
9.3.2	Termination Detection on a Unidirectional Ring
9.3.3	Credit-Recovery Algorithm for Termination Detection
9.4	Wave Algorithms
9.4.1	Propagation of Information with Feedback
9.5	Distributed Deadlock Detection
9.5.1	Resource Deadlock and Communication Deadlock
9.5.2	Detection of Resource Deadlock
9.5.3	Detection of Communication Deadlock
9.6	Concluding Remarks

9.7	Bibliographic Notes
10	Graph Algorithms
10.1	Introduction
10.2	Routing Algorithms
10.2.1	Computation of Shortest Path
10.2.2	Distance Vector Routing
10.2.3	Link State Routing
10.2.4	Interval Routing
10.2.5	Prefix Routing
10.3	Graph Traversal
10.3.1	Spanning Tree Construction
10.3.2	Tarry's Traversal Algorithm
10.3.3	Minimum Spanning Tree Construction
10.4	Graph Coloring
10.4.1	(D+1)-coloring Algorithm
10.4.2	6-coloring of Planar Graphs
10.5	Cole-Vishkin Reduction Algorithm for Tree-coloring
10.6	Maximal Independent Set: Luby's Algorithms
10.7	Concluding Remarks
10.8	Bibliographic Notes
11	Coordination Algorithms
11.1.	Introduction
11.2.	Leader Election
11.2.1.	The Bully Algorithm
11.2.2.	Maxima Finding on a Ring
11.2.2.1.	Chang-Robert's Algorithm
11.2.2.2.	Franklin's Algorithm
11.2.2.3.	Peterson's Algorithm
11.2.3	Election in Arbitrary Networks
11.2.4	Election in Anonymous Networks
11.3	Synchronizers
11.3.1	The ABD Synchronizer
11.3.2	Awerbuch's Synchronizers
11.3.2.1	The α Synchronizer
11.3.2.2	The β Synchronizer
11.3.2.3	The γ Synchronizer
11.3.2.4	The Performance of Synchronizer-based Algorithms
11.4	Concluding remarks
11.5	Bibliographic Notes

SECTION IV. FAULTS AND FAULT-TOLERANT SYSTEMS

12	Fault Tolerant Systems
12.1	Introduction
12.2	Classification of Faults
12.3	Specification of Faults
12.4	Fault-tolerant Systems
12.4.1	Masking Tolerance

12.4.2	Non-masking Tolerance
12.4.3	Fail-safe Tolerance
12.4.4	Graceful Degradation
12.4.5	Detection of Failures in Synchronous Systems
12.5	Tolerating Crash failures
12.5.1	Double and Triple Modular Redundancy
12.6	Tolerating Omission Failures
12.6.1	Stenning's Protocol
12.6.2	The Sliding Window Protocol
12.6.3	The Alternating Bit Protocol
12.6.4	How TCP works
12.7	Concluding remarks
12.8	Bibliographic Notes
13	Distributed Consensus
13.1	Introduction
13.2	Consensus in Asynchronous Systems
13.3	Consensus in Synchronous Systems: Byzantine Generals Problem
13.3.1	The Solution with no traitor
13.3.2	Solution with traitors: Interactive Consistency Criteria
13.3.3	Consensus with Oral Messages
13.3.3.1	An impossibility result
13.3.3.2	The OM(m) Algorithms
13.3.4	Consensus with Signed Messages
13.4	The Paxos Algorithm
13.5	Failure Detectors
13.5.1	Solving consensus using failure detectors
13.6	Concluding Remarks
13.7	Bibliographic Notes
14	Distributed Transactions
14.1	Introduction
14.2	Classification of Transactions
14.3	Implementing Transactions
14.4	Concurrency Control and Serializability
14.4.1	Testing for Serializability
14.4.2	Two-phase Locking
14.4.3	Concurrency Control via Timestamp Ordering
14.5	Atomic Commit Protocols
14.5.1	One-phase commit
14.5.2	Two-phase commit
14.5.3	Three-phase commit
14.6	Recovery from Failures
14.6.1	Stable Storage
14.6.2	Checkpointing and Rollback Recovery
14.6.3	Message logging
14.7	Concluding Remarks
14.8	Bibliographic Notes
15	Group Communication
15.1	Introduction
15.2	Atomic Multicast

15.3	IP Multicast	15.3.1 Reverse Path Forwarding
15.4	Application Layer Multicast	
15.5	Ordered Multicast	15.5.1 Implementing Total Order Multicast 15.5.2 Implementing Causal Order Broadcast
15.6	Reliable Multicast	15.6.1 Scalable Reliable Multicast 15.6.2 Reliable Ordered Multicast
15.7	Open Groups	15.7.1 View-synchronous Group Communication
15.8	An Overview of Transis	
15.9	Concluding Remarks	
15.10	Bibliographic Notes	
16	Replicated Data Management	
16.1	Introduction	16.1.1 Reliability vs. Availability
16.2	The Architecture of Replicated Data Management	16.2.1 Passive vs. Active Replication 16.2.2 Fault-tolerant State Machines
16.3	Data-centric Consistency Models	16.3.1 Strict Consistency 16.3.2 Linearizability 16.3.3 Sequential Consistency 16.3.4 Causal Consistency 16.3.5 FIFO Consistency
16.4	Client-centric Consistency Models	16.4.1 Eventual consistency 16.4.2 Consistency models for mobile clients
16.5	Implementation of Data-centric Consistency Models	
16.6	Quorum-based protocols	
16.7	Replica placement	
16.8	Brewer's CAP Theorem	
16.9	Case studies	16.9.1 Replication Management in Coda 16.9.2 Replication Management in Bayou 16.9.3 Amazon Dynamo
16.10	Concluding Remarks	
16.11	Bibliographic Notes	
17	Self-stabilizing Distributed Systems	
17.1	Introduction	
17.2	Theoretical Foundations	
17.3	Stabilizing Mutual Exclusion	17.3.1 Mutual Exclusion on a Unidirectional Ring 17.3.2 Mutual exclusion on a Bidirectional Array
17.4	Stabilizing Graph Coloring	
17.5	Stabilizing Spanning Tree Protocol	
17.6	Stabilizing Maximal Matching	
17.7	Distributed Reset	
17.8	Stabilizing Clock Phase Synchronization	

- 17.9 Concluding Remarks
- 17.10 Bibliographic Notes

SECTION V. REAL WORLD ISSUES

- 18 Distributed Discrete-event Simulation
 - 18.1 Introduction
 - 18.1.1 Event-driven Simulation
 - 18.2 Distribution Simulation
 - 18.2.1 The Challenges
 - 18.2.2 Correctness Issues
 - 18.3 Conservative Simulation
 - 18.4 Optimistic Simulation and Time Warp
 - 18.4.1 Global Virtual Time
 - 18.5 Concluding Remarks
 - 18.6 Bibliographic Notes
- 19 Security in Distributed Systems
 - 19.1 Introduction
 - 19.2 Security Mechanisms
 - 19.3 Common Security Attacks
 - 19.4 Encryption
 - 19.5 Secret-key Cryptosystem
 - 19.5.1 Confusion and Diffusion
 - 19.5.2 DES
 - 19.5.3 3DES
 - 19.5.4 AES
 - 19.5.5 One-time pad
 - 19.5.6 Stream Ciphers
 - 19.5.7 Steganography
 - 19.6 Public-key Cryptosystems
 - 19.6.1 The RSA method
 - 19.6.2 ElGamal Cryptosystem
 - 19.7 Digital Signatures
 - 19.7.1 Signatures in Secret-key Cryptosystems
 - 19.7.2 Signatures in Public-key Cryptosystems
 - 19.8 Hashing Algorithms
 - 19.8.1 Birthday attack
 - 19.9 Elliptic curve cryptography
 - 19.10 Authentication Server
 - 19.10.1 Authentication Server for Secret-key Cryptosystems
 - 19.10.2 Authentication Server for Public-key Cryptosystems
 - 19.11 Digital Certificates
 - 19.12 Case Studies
 - 19.12.1 Kerberos
 - 19.12.2 Pretty Good Privacy
 - 19.12.3 Secure Socket Layer
 - 19.13 Virtual Private Networks and Firewalls
 - 19.13.1 Virtual Private network
 - 19.13.2 Firewall
 - 19.14 Sharing a secret
 - 19.15 Concluding Remarks

	19.16 Bibliographic Notes
20	Sensor Networks
20.1	The vision
20.2	The architecture of a sensor node
20.2.1	MICA mote
20.2.2	ZigBee enabled sensor nodes
20.2.3	TinyOS operating system
20.3	The challenges in wireless sensor networks
20.3.1	Energy conservation
20.3.2	Fault tolerance
20.3.3	Routing
20.3.4	Time synchronization
20.3.5	Location management
20.3.6	Middleware design
20.3.7	Security
20.4	Routing algorithms for sensor networks
20.4.1	Directed diffusion
20.4.2	Cluster-based routing
20.4.2.1	LEACH
20.4.2.2	PEGASIS
20.4.3	Meta-data based routing: SPIN
20.5	Time synchronization using reference broadcast
20.6	Localization algorithms
20.6.1	RSSI-base ranging
20.6.2	Ranging using Time-difference of Arrival
20.6.3	Anchor-based ranging
20.7	Security in sensor networks
20.7.1	SPIN for data security
20.7.2	Attacks on routing
20.8	Applications
20.8.1	Healthcare Applications
20.8.2	Environment Monitoring and Control
20.8.3	Citizen Sensing
20.8.4	Pursuer-evader game
20.9	Concluding remarks
20.10	Bibliographic Notes
21	Social and Peer-to-peer Networks
21.1	Introduction to Social Networks
21.2	Metrics of Social Networks
21.2.1	Clustering Coefficient
21.2.2	Diameter
21.3	Modeling Social Networks
21.3.1	Erdős-Rényi Model
21.3.2	The Small World Model
21.3.3	Power Law Graphs
21.4	Centrality Measures in Social Networks
21.4.1	Degree Centrality
21.4.2	Closeness Centrality
21.4.3	Betweenness Centrality
21.5	Community Detection

- 21.5.1 Girvan-Newman Algorithm
- 21.6 Introduction to Peer-to-peer Networks
- 21.7 First generation P2P networks
 - 21.7.1 Napster
 - 21.7.2 Gnutella
- 21.8 Second generation P2P networks
 - 21.8.1 KaZaA
 - 21.8.2 Chord
 - 21.8.3 Content Addressable Network (CAN)
 - 21.8.4 Pastry
- 21.9 Koorde and De Bruijn Graph
- 21.10 Skip Graph
- 21.11 Replication management
- 21.12 BitTorrent and Free-riding
- 21.13 Censorship resistance, anonymity
- 21.14 Concluding Remarks
- 21.15 Bibliographic Notes

22 Bibliography