Consistent Distributed Storage

Synthesis Lectures on Distributed Computing Theory

Editor
Michel Raynal, ***University of Rennes, France and Hong Kong Polytechnic University***

Founding Editor
Nancy Lynch, ***Massachusetts Institute of Technology***

Synthesis Lectures on Distributed Computing Theory was founded by Nancy Lynch of the Massachusetts Institute of Technology, and is now edited by Michel Raynal of the University of Rennes, France and Hong Kong Polytechnic University. The series publishes 50- to 150-page publications on topics pertaining to distributed computing theory. The scope largely follows the purview of premier information and computer science conferences, such as ACM PODC, DISC, SPAA, OPODIS, CONCUR, DialM-POMC, ICDCS, SODA, Sirocco, SSS, and related conferences. Potential topics include, but not are limited to: distributed algorithms and lower bounds, algorithm design methods, formal modeling and verification of distributed algorithms, and concurrent data structures.

Consistent Distributed Storage
Vincent Gramoli, Nicolas Nicolaou, and Alexander A. Schwarzmann
2021

Network Topology and Fault-Tolerant Consensus
Dimitris Sakavalas and Lewis Tseng
2019

Introduction to Distributed Self-Stabilizing Algorithms
Karine Altisen, Stéphane Devismes, Swan Dubois, and Franck Petit
2019

Distributed Computing Perls
Gadi Taubenfeld
2018

Decidability of Parameterized Verification
Roderick Bloem, Swen Jacobs, Ayrat Khalimov, Igor Konnov, Sasha Rubin, Helmut Veith, and Josef Widder
2015

Impossibility Results for Distributed Computing
Hagit Attiya and Faith Ellen
2014

Distributed Graph Coloring: Fundamentals and Recent Developments
Leonid Barenboim and Michael Elkin
2013

Distributed Computing by Oblivious Mobile Robots
Paola Flocchini, Giuseppe Prencipe, and Nicola Santoro
2012

Quorum Systems: With Applications to Storage and Consensus
Marko Vukolić
2012

Link Reversal Algorithms
Jennifer L. Welch and Jennifer E. Walter
2011

Cooperative Task-Oriented Computing: Algorithms and Complexity
Chryssis Georgiou and Alexander A. Shvartsman
2011

New Models for Population Protocols
Othon Michail, Ioannis Chatzigiannakis, and Paul G. Spirakis
2011

The Theory of Timed I/O Automata, Second Edition
Dilsun K. Kaynar, Nancy Lynch, Roberto Segala, and Frits Vaandrager
2010

Principles of Transactional Memory
Rachid Guerraoui and Michał Kapałka
2010

Fault-tolerant Agreement in Synchronous Message-passing Systems
Michel Raynal
2010

Communication and Agreement Abstractions for Fault-Tolerant Asynchronous Distributed Systems
Michel Raynal
2010

The Mobile Agent Rendezvous Problem in the Ring
Evangelos Kranakis, Danny Krizanc, and Euripides Markou
2010

© Springer Nature Switzerland AG 2022
Reprint of original edition © Morgan & Claypool 2021

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means—electronic, mechanical, photocopy, recording, or any other except for brief quotations in printed reviews, without the prior permission of the publisher.

Consistent Distributed Storage
Vincent Gramoli, Nicolas Nicolaou, and Alexander A. Schwarzmann

ISBN: 978-3-031-00887-0 paperback
ISBN: 978-3-031-02015-5 eBook
ISBN: 978-3-031-00133-8 hardcover

DOI 10.1007/978-3-031-02015-5

A Publication in the Springer series
SYNTHESIS LECTURES ON DISTRIBUTED COMPUTING THEORY

Lecture #17
Series Editor: Michel Raynal, *University of Rennes, France and Hong Kong Polytechnic University*
Founding Editor: Nancy Lynch, *Massachusetts Institute of Technology*
Series ISSN
Print 2155-1626 Electronic 2155-1634

Consistent Distributed Storage

Vincent Gramoli
University of Sydney, Australia
EFPL, Switzerland

Nicolas Nicolaou
Algolysis Ltd., Cyprus

Alexander A. Schwarzmann
Augusta University, Georgia, USA

SYNTHESIS LECTURES ON DISTRIBUTED COMPUTING THEORY #17

ABSTRACT

Providing a shared memory abstraction in distributed systems is a powerful tool that can simplify the design and implementation of software systems for networked platforms. This enables the system designers to work with abstract readable and writable objects without the need to deal with the complexity and dynamism of the underlying platform. The key property of shared memory implementations is the consistency guarantee that it provides under concurrent access to the shared objects. The most intuitive memory consistency model is *atomicity* because of its equivalence with a memory system where accesses occur serially, one at a time. Emulations of shared atomic memory in distributed systems is an active area of research and development. The problem proves to be challenging, and especially so in distributed message passing settings with unreliable components, as is often the case in networked systems. We present several approaches to implementing shared memory services with the help of replication on top of message-passing distributed platforms subject to a variety of perturbations in the computing medium.

KEYWORDS

distributed shared memory, atomicity, read/write objects, quorum systems, reconfiguration, consensus, fault-tolerance, dynamism

Contents

Acknowledgments . . . . . **xi**

Outline . . . . . **xiii**

1 Introduction **1**

1.1 Shared Storage: A Landscape 1

1.2 Distribution and Consistency 3

2 Model of Computation **7**

2.1 Distributed System 7

2.1.1 Input/Output Automata and Executions 7

2.1.2 Failures 9

2.1.3 Communication 10

2.1.4 Quorum Systems 11

2.2 Consistency: Atomic Object Semantics 13

2.3 Notation Summary 16

2.4 Bibliographic Notes 17

3 The Static Environment **19**

3.1 Replication 19

3.2 Communication and Rounds 19

3.3 Efficiency Measures 20

3.4 List of Symbols 21

4 The Single-Writer Setting **23**

4.1 SWMR Algorithm ABD: Basic Techniques 23

4.1.1 Algorithm ABD Specification 24

4.1.2 Algorithm ABD Correctness 25

4.1.3 Efficiency of Algorithm ABD 29

4.2 Algorithm Fast: Expediting Read Operations 30

4.2.1 Algorithm Fast Specification 30

4.2.2 Algorithm Fast Correctness 34

4.3 Lower Bound: Limitations of Fast Implementations 37
4.4 Algorithm Sliq: Introducing Quorum Views 38
4.4.1 Algorithmic Technique: Quorum Views 39
4.4.2 Algorithm Sliq Specification 40
4.4.3 Algorithm Sliq Correctness 44
4.5 Bibliographic Notes 45

5 The Multiple-Writer Setting 47
5.1 Algorithm mwABD: Multi-Writer ABD 47
5.1.1 Algorithm mwABD Specification 47
5.1.2 Algorithm mwABD Correctness 49
5.2 Algorithm CwFr: Quorum View Generalization 51
5.2.1 Algorithmic Technique: MW Quorum Views 51
5.2.2 Algorithm CwFr Specification 52
5.2.3 Algorithm CwFr Correctness 57
5.3 Lower Bound: Inherent Limitations of mwmr on Fast Operations 60
5.4 Algorithm SfW: Expediting Write Operations 66
5.4.1 Algorithmic Technique: Server Side Ordering (SSO) 67
5.4.2 Algorithmic Technique: Enhanced Tagging 68
5.4.3 Algorithm SfW Specification 69
5.4.4 Algorithm SfW Correctness 73
5.5 Bibliographic Notes 77

6 The Dynamic Environment 79
6.1 Consensus 80
6.2 Group Communication Services 81
6.3 Using Reconfiguration for Direct Implementations. 82

7 RAMBO: Reconfigurable Dynamic Memory 83
7.1 Models and Definitions 84
7.2 Rambo Service Specifications 86
7.2.1 Rambo Service Specification 86
7.2.2 Recon Service Specification 88
7.3 Implementation of Rambo 91
7.3.1 Overall Architecture 92
7.3.2 Implementation of Joiner 92
7.3.3 Implementation of Reader-Writer 92

7.3.4 Implementation of the Recon Service 100
7.3.5 The Complete Rambo Algorithm 102
7.4 Atomicity of Rambo 102
7.5 Conditional Performance Analysis: Latency Bounds 105
7.6 Extensions of Rambo 108
7.7 GeoQuorums—Adaptation of Rambo for Mobile Settings 111
7.8 Bibliographic Notes 113

8 RDS: Integrated Reconfigurations 115

8.1 Preliminaries 115
8.2 The Paxos Consensus Algorithm 116
8.2.1 The Part-Time Parliament 116
8.2.2 Fast Paxos 117
8.2.3 Deciding Upon a New Configuration 118
8.3 Reconfigurable Distributed Storage 119
8.3.1 Signature and State 119
8.3.2 Read and Write Operations 121
8.3.3 Communication and Independent Transitions 122
8.3.4 Reconfiguration 125
8.4 Consensus of RDS 128
8.5 Conditional Performance Analysis 129
8.6 Bibliographic Notes 131

9 DynaStore: Incremental Reconfigurations 135

9.1 Preliminaries 136
9.1.1 Incremental Reconfiguration 137
9.1.2 Activity of Processes 137
9.1.3 Majority Assumption 138
9.2 The Weak Snapshot Automaton 138
9.3 The Reader-Writer-Recon Automaton 142
9.3.1 Partially Ordered Configurations 144
9.3.2 Adding New Configurations 145
9.3.3 Contacting Quorums 146
9.4 Correctness 147
9.5 Implementation and Performance Considerations 152
9.6 Bibliographic Notes 154

10 Concluding Remarks and Looking Ahead 155

Bibliography 159

Authors' Biographies 171

Index 173

Acknowledgments

The work includes results obtained in collaboration with several colleagues and by other researchers. The bibliographic notes throughout provide details on prior work.

The authors thank Nancy Lynch, the Series Founding Editor, for supporting this project and for her encouragement, and Michel Raynal, the Series Editor, for shepherding this project to completion. We thank the reviewers, Christian Cachin, Rachid Guerraoui, and Marko Vukolić, whose meaningful and insightful feedback helped improve the quality of this presentation. Special thanks are due to Diane Cerra, Editor at Morgan & Claypool Publishers, for her active role, encouragement, patience, and perseverance.

The work of the first author was partially supported by Australian Research Council Discovery Projects funding scheme (project number 180104030) and Future Fellowship funding scheme (project number 180100496). The work of the second author was in part supported by the European Commission Marie Skłodowska-Curie Action (project number 629088) and through the Cyprus Research and Innovation Foundation (POST-DOC/0916/0090) . This work of the third author was in part supported by the National Science Foundation (NSF) Grant 1017232 and Augusta University Research Foundation.

Finally, we would like to thank our families.

The first author wishes to thank his wife, Klara, his daughter, Zoé, and his two sons, Hugo and Roméo, for their patience during the writing of this monograph. The writing has taken longer than expected, but he notes in retrospect that the writing of this monograph is the perfect illustration of the French idiom "se bonnifier avec l'âge" because the result ended up exceeding his initial expectation.

The second author would like to express his gratitude to his wife, Andri, for her steady support, love, and patience during the journey of writing this book. It's been an unexpectedly long journey, and within this time she endured traveling by his side along with their three children to two continents for fulfilling personal goals. It would have been a major omission not to thank his three heroes, his children, Raphaela, Ioannis, and Chrysanthos, for they sat patiently waiting (most of the time) for some quality time with their daddy.

The third author is grateful to his wife, Sana, for her love, encouragement and support. This project did take longer than planned, but he notes with pride that his children had plenty of time to give him three grandsons. Finally he dedicates his work on this monograph to the Memory of his Father who would say, "It is about time you were done!"

Vincent Gramoli
Nicolas Nicolaou
Alexander A. Schwarzmann

Outline

This book contains ten chapters. Figure 1 shows some dependencies among chapters and suggests some reading orders.

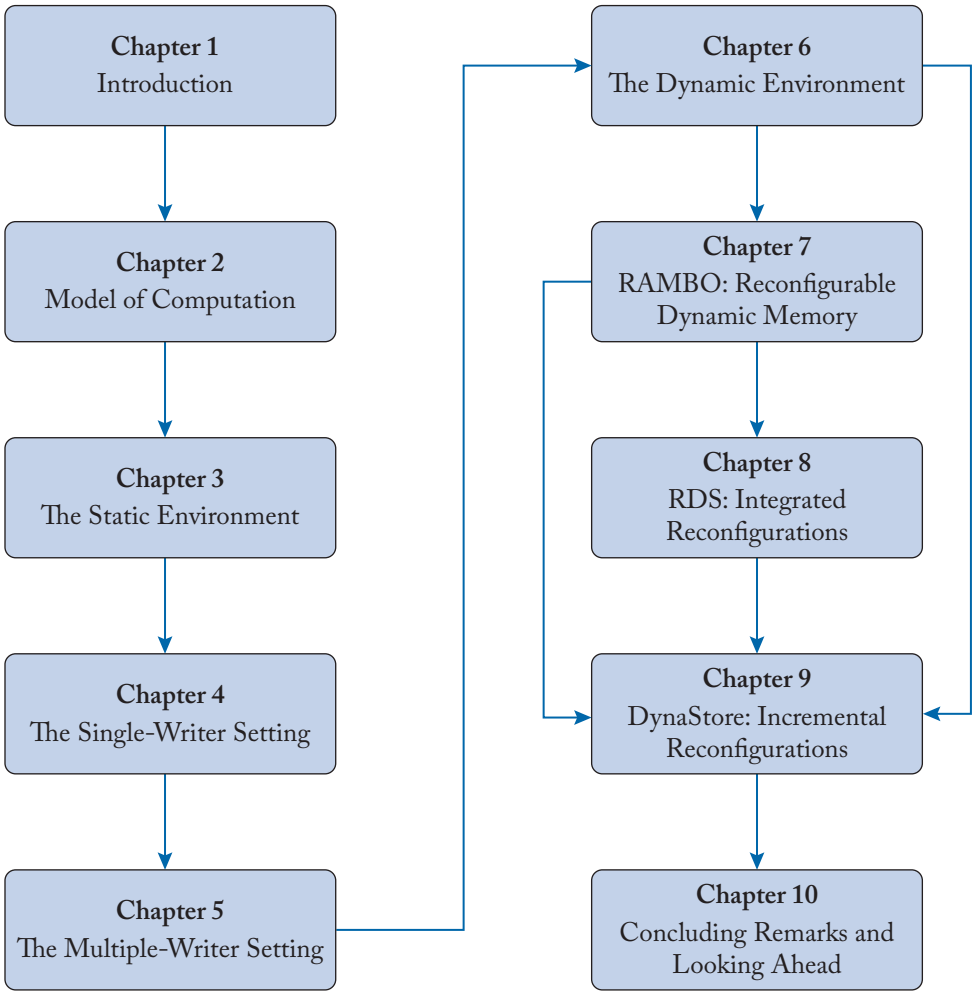

Figure 1: Suggested order of reading.

We begin the presentation in Chapter 1 where we discuss the general landscape of distributed shared storage systems, and introduce the challenges associated with combining replication and consistency in distributed settings.

In Chapter 2, we describe the model of computation, define consistency, and assumptions about communication and failure models. When a particular algorithm assumes a specialized model, we present model refinements and additional assumptions in the relevant section before presenting the algorithm.

The rest of the book is divided into two logical parts. In the first part we deal with static systems where the collection of failure-prone object replica servers is known a-priori. In the second part we deal with dynamic systems, where the set of participating servers may change over time as existing participants depart or fail, and as new participants join.

The first part, dealing with static systems, consists of Chapters 3, 4, and 5. Chapter 3 discusses additional modeling parameters. In Chapter 4, we deal with the algorithms for the single writer, multiple readers model that guarantee consistency in all executions, and where read and write operations terminate provided that a majority of servers do not fail. In Chapter 5, we present algorithms for the multiple writers, multiple readers model, and we exemplify techniques that can be used to speed up read and write operations.

The second part, dealing with dynamic systems, consists of Chapters 6, 7, 8, and 9. Chapter 6 discusses approaches to dynamic atomic memory implementation using general purpose distributed building blocks. Chapter 7 presents a framework to reconfigure a dynamic atomic memory by replacing a configuration of servers by another while offering availability of reads and writes. This reconfiguration relies on an external building block for servers to agree on configurations. The chapter concludes with a survey of several algorithms that optimize and specialize the framework for particular target environments. Chapter 8 presents algorithms that integrate the reconfiguration procedures within the atomic memory implementation to reduce time and message complexities. This integration improves the fault tolerance of the previous framework by minimizing the number of configurations needed at any point of the execution. Chapter 9 presents algorithms that obviate the need for including agreement within an explicit reconfiguration procedure. These algorithms accept incremental changes to the configuration in order to avoid the need for a consensus-based reconfiguration.

We conclude in Chapter 10 by discussing practical consistent memory services, performance considerations, and challenges associated with replication and reconfiguration.

Lastly, in addition to the reading orders proposed in Figure 1, the reader may benefit from the content of particular sections to fast-track to later chapters in the book. In particular, after reading Sections 4.1 and 5.1 (in this order) the reader may either choose to continue with the corresponding chapter sequels dealing with the static system, or may proceed to Chapter 6 and start reading about the dynamic systems.

Vincent Gramoli
Nicolas Nicolaou
Alexander A. Schwarzmann
June 2021