

Peer-to-Peer Computing

Quang Hieu Vu • Mihai Lupu • Beng Chin Ooi

Peer-to-Peer Computing

Principles and Applications

 Springer

Quang Hieu Vu
Institute for Infocomm Research (I²R)
1 Fusionopolis Way
#21-01 Connexis, South Tower
Singapore 138632
Singapore
qhvu@i2r.a-star.edu.sg

Mihai Lupu
Information Retrieval Facility (IRF)
Operngasse 20b
1040 Wien
Austria
mihailupu@gmail.com

Beng Chin Ooi
Department Computer Science
School of Computing
National University of Singapore
Computing Drive
Computing 1
Singapore 117590
Singapore
ooibc@comp.nus.edu.sg

ISBN 978-3-642-03513-5
DOI 10.1007/978-3-642-03514-2
Springer Heidelberg Dordrecht London New York

e-ISBN 978-3-642-03514-2

Library of Congress Control Number: 2009938717

ACM Computing Classification (1998) C.2, D.4, H.3, K.6

© Springer-Verlag Berlin Heidelberg 2010

This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilm or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the German Copyright Law of September 9, 1965, in its current version, and permission for use must always be obtained from Springer. Violations are liable to prosecution under the German Copyright Law.

The use of general descriptive names, registered names, trademarks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

Cover design: KuenkelLopka, GmbH

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

Peer-to-peer (P2P) technology, or peer computing, is an emerging paradigm that is now viewed as a potential technology to redesign distributed architectures (e.g., the Internet) and, consequently, distributed processing. In a classical P2P network, all participating computers (or nodes) have equivalent capabilities and responsibilities. The nodes can directly exchange resources and services between each other without the need for centralized servers. They can collaborate to perform tasks by aggregating the pool of resources (e.g., storage, CPU cycles) available in the P2P network. The distributed nature of such a design provides exciting opportunities for new killer applications to be developed.

P2P computing distinguishes itself from traditional distributed computing in three main aspects. First, the scalability of P2P systems goes far beyond that of traditional distributed systems. In particular, since P2P systems are able to scale to thousands of nodes, they can harness the power of computers over the Internet. Second, P2P, in its most uncompromising definition, requires everything to be completely decentralized. Ideally, no centralized structures should exist in P2P systems. Finally, and also the most important one, P2P applications often work in highly dynamic environments. Specifically, in terms of network topology, since P2P nodes can join and leave the system anytime, P2P systems do not have a fixed topology. Instead, their topology changes according to nodes in the system. Furthermore, the system's content and load are distributed in real time according to the actual demand and resource capability of nodes. For example, if a sharing file becomes "hot", i.e., it is repeatedly requested by several users, the file can be duplicated and deployed in many parts of the system.

The scale and dynamism that characterize P2P systems require traditional distributed technologies to be reexamined. A paradigm shift that includes self-reorganization, adaptation and resilience is called for. In recent years, there has been a proliferation of research efforts to design P2P systems and applications. This book attempts to present the *technical* challenges offered by P2P systems, and the efforts that have been proposed to address them. The purpose of this book is to provide a thorough and comprehensive review of recent advances on routing and discovery methods; load balancing and replication techniques; security, accountability and

anonymity, as well as trust and reputation schemes; programming models and P2P systems and projects. Besides surveying existing methods and systems, the book also compares and evaluates some of the more promising schemes.

The need for such a book is evident. It provides a single source for practitioners, researchers and newcomers on the state-of-the-art in the field. For practitioners, this book explains best practice, guiding selection of appropriate techniques for each application. For researchers, this book provides a foundation for development of new and more effective methods. For newcomers, this book is an overview of the wide range of advanced techniques for realizing effective P2P systems. This book can also be used as a text for an advanced course on Peer-to-Peer Computing and Technologies, or as a companion text for a variety of courses, including courses on distributed systems, grid, and cluster computing.

Organization of the Book

This book consists of ten chapters. Besides the first chapter that sets up the context and the last chapter that concludes with directions on the future of P2P, each of the other eight chapters is essentially self-contained and focuses on one aspect of P2P computing. These eight chapters can thus be read and used on their own, independently of the others.

- In Chap. 1, we provide background on P2P computing in general. We discuss the characteristics of P2P systems that distinguish them from other distributed systems. This chapter also looks at the benefits and promises of P2P, and some of the applications that will benefit from P2P computing. It examines the issues in designing P2P systems and sets the stage for subsequent chapters.
- Chapter 2 presents the various architectures of P2P systems. At one extreme, we have P2P systems that are supported by centralized servers. At the other extreme, pure P2P systems are completely decentralized. Between these two extremes are hybrid systems where nodes are organized into two layers: the upper tier “super” nodes act as servers for lower tier nodes. We compare these different architectures. In parallel to the static architectural considerations, we also look at how peers are defined—statically or dynamically. Support for dynamic reorganization of peers allows communities to be formed based on some common interests among nodes. For hybrid systems, we examine how nodes that are more powerful can be exploited to shoulder more responsibilities. Issues on incentives and fairness are also addressed.
- In Chap. 3, we focus on the issue of searching. There are several modes in which searching can be performed. First, a query node can broadcast queries to all nodes. Second, the query can be directed to nodes that are more likely to contain useful information first. This requires nodes to organize their peers based on some optimization criterion. Third, hashing techniques can be applied. We also look at how load-balancing can be realized in the hash-based category. Each of these techniques call for different metadata to be maintained.

- Chapter 4 presents techniques to perform complex queries. Besides simple keyword search, there is an increasing need to support more semantic-based queries for database and multimedia applications. These include partial match queries, range and join queries, and queries involving high-dimension vectors. We also look at how distributed queries are optimized and processed in the P2P context.
- Replication and caching are very effective mechanisms that can bring the data/results closer to the users to improve performance. However, in the P2P environment, it becomes much harder to control the optimal degree of replication, as well as to maintain the consistency between replicas. Chapter 5 presents the issues that need to be addressed and examines some of the existing solutions. In particular, we look at techniques that manage replicas/cache dynamically.
- Before P2P can be widely accepted by users, there are several other issues that need to be addressed: trust, privacy, anonymity, accountability, reliability, and security. These issues are discussed in Chaps. 6 and 7. In Chap. 6, we focus on security, privacy, and anonymity issues. We begin by discussing techniques designed to secure data as well as the overall P2P environment from different types of attacks. Then we present methods that prevent users from taking advantage of the system by freeloading off the resources contributed by a few. Finally, we look at techniques that are designed to support anonymity and privacy, to protect both the users that disseminate the data, as well as nodes that store the data. Techniques that authenticate third-party data publication are also examined in this chapter.
- Chapter 7 focuses on accountability, trust, and reputation. Here, we look at techniques that automate the collection and processing of information from previous queries to help users assess whether they can trust a server with a new query.
- In Chap. 8, we look at programming tools that are suitable for P2P environments. After having presented in the previous chapters the theoretical aspects of P2P systems, in this chapter we will identify tools to develop P2P systems, ranging from low level network programming tools, like sockets, to specific programming languages designed to be used for P2P applications.
- Chapter 9 describes some representative P2P systems and applications that have been deployed. We look at how different application environments and requirements drive the design and architecture of the systems. We discuss popular techniques employed in each type of applications. In particular, we present systems that support file sharing, data backup, structured data management, and data caching. Additionally, we also introduce mobile systems employing P2P technologies.

Finally, in Chap. 10, we make a conclusion of the book and suggest promising research topics that deserve further attention. Additionally, we also discuss a potential use of P2P in industry by analyzing a case of supply chain management system.

Acknowledgements

This book cannot be finished without the help of several people. The authors of this book would like to thank Dr. Kian-Lee Tan, Dr. Panos Kalnis, and Dr. Bei Yu at National University of Singapore, Dr. Aoying Zhou at East China Normal University, and Dr. Linhao Xu at the IBM China Research Laboratory for their contributions in the initial phase of the book.

Contents

1	Introduction	1
1.1	Peer-to-Peer Computing	1
1.2	Potential, Benefits, and Applications	3
1.3	Challenges and Design Issues	7
1.4	P2P vs. Grid Computing	8
1.5	Summary	10
2	Architecture of Peer-to-Peer Systems	11
2.1	A Taxonomy	12
2.1.1	Centralized P2P Systems	13
2.1.2	Decentralized P2P Systems	13
2.1.3	Hybrid P2P Systems	15
2.2	Centralized P2P Systems	15
2.2.1	Napster: Sharing of Digital Content	17
2.2.2	About SETI@home	18
2.3	Fully Decentralized P2P Systems	20
2.3.1	Properties	21
2.3.2	Gnutella: The First “Pure” P2P System	22
2.3.3	PAST: A Structured P2P File Sharing System	24
2.3.4	Canon: Turning Flat DHT into Hierarchical DHT	26
2.3.5	Skip Graph: A Probabilistic-Based Structured Overlay	28
2.4	Hybrid P2P Systems	31
2.4.1	BestPeer: A Self-Configurable P2P System	32
2.5	Summary	36
3	Routing in Peer-to-Peer Networks	39
3.1	Evaluation Metrics	40
3.2	Routing in Unstructured P2P Networks	40
3.2.1	Basic Routing Strategies	41
3.2.2	Heuristic-Based Routing Strategies	43
3.3	Routing in Structured P2P Networks	50
3.3.1	Chord	52

3.3.2	CAN	56
3.3.3	PRR Trees, Pastry and Tapestry	58
3.3.4	Viceroy	63
3.3.5	Crescendo	64
3.3.6	Skip Graph	65
3.3.7	SkipNet	67
3.3.8	P-Grid	67
3.3.9	P-Tree	69
3.3.10	BATON	71
3.4	Routing in Hybrid P2P Networks	73
3.4.1	Hybrid Routing	73
3.5	Summary	78
4	Data-Centric Applications	81
4.1	Multi-Dimensional Data Sharing	82
4.1.1	VBI-Tree	84
4.1.2	Mercury	86
4.1.3	SSP	88
4.2	High-Dimensional Indexing	90
4.2.1	CISS	91
4.2.2	ZNet	92
4.2.3	M-Chord	94
4.2.4	SIMPEER	96
4.2.5	LSH Forest	97
4.3	Textual Information Retrieval	98
4.3.1	Basic Techniques	100
4.3.2	PlanetP	104
4.3.3	Summary Index	106
4.3.4	pSearch	107
4.3.5	PRISM	109
4.4	Structured Data Management	111
4.4.1	Query Processing in Heterogeneous Data Sources	112
4.4.2	Piazza	118
4.4.3	Hyperion	121
4.4.4	PeerDB	123
4.5	Summary	125
5	Load Balancing and Replication	127
5.1	Load Balancing	128
5.1.1	When Load Balancing is Triggered	128
5.1.2	How Load Balancing is Performed	131
5.2	Load Balancing in Concrete Systems	132
5.2.1	Basic Load Balancing Schemes with Virtual Nodes	132
5.2.2	Y_0 Protocol	134
5.2.3	The S&M Protocol	135
5.2.4	A Combination of Both Local and Random Probes	137

5.2.5	Mercury	137
5.2.6	Online Balancing of Range-partitioned Data	139
5.3	Replication	140
5.3.1	Replica Granularity	140
5.3.2	Replica Quantity	142
5.3.3	Replica Distribution	143
5.3.4	Replica Consistency	144
5.3.5	Replica Replacement	144
5.4	Replication in Concrete Systems	145
5.4.1	Replication in Read-only Unstructured P2P Systems	145
5.4.2	Replication in Read-only Structured P2P Systems	145
5.4.3	Beehive	147
5.4.4	Symmetric Replication for Structured Peer-to-Peer Systems	149
5.4.5	CUP: Controlled Update Propagation in Peer-to-Peer Networks	150
5.4.6	Dynamic Replica Placement for Scalable Content Delivery	151
5.4.7	Updates in Highly Unreliable, Replicated P2P Systems	152
5.4.8	Proactive Replication	154
5.5	Summary	155
6	Security in Peer-to-Peer Networks	157
6.1	Routing Attacks	157
6.1.1	Incorrect Lookup Routing	158
6.1.2	Incorrect Routing Updates	158
6.1.3	Incorrect Routing Network Partition	158
6.1.4	Secure Routing Scheme	159
6.2	Storage and Retrieval Attacks	160
6.3	Denial-of-Service Attacks	162
6.3.1	Managing Attacks	163
6.3.2	Detecting and Recovering from Attacks	164
6.3.3	Other Attacks	166
6.4	Data Integrity and Verification	166
6.4.1	Verifying Queries in Relational Databases	167
6.4.2	Self-verifying Data with Erasure Code	170
6.5	Verifying Integrity of Computation	171
6.6	Free Riding and Fairness	172
6.6.1	Quota-Based System	173
6.6.2	Trading-Based Schemes	174
6.6.3	Distributed Auditing	175
6.6.4	Incentive-Based Schemes	176
6.6.5	Adaptive Topologies	178
6.7	Privacy and Anonymity	179
6.8	PKI-Based Security	181
6.9	Summary	182

7	Trust and Reputation	183
7.1	Concepts	184
7.1.1	Trust Definitions	184
7.1.2	Trust Types	185
7.1.3	Trust Values	186
7.1.4	Trust Properties	187
7.2	Trust Models	188
7.2.1	Trust Model Based on Credentials	189
7.2.2	Trust Model Based on Reputation	189
7.3	Trust Systems Based on Credentials	190
7.3.1	PolicyMaker	190
7.3.2	Trust-X	192
7.4	Trust Systems Based on Individual Reputation	194
7.4.1	P2PRep	194
7.4.2	XRep	196
7.4.3	Cooperative Peer Groups in NICE	197
7.4.4	PeerTrust	198
7.5	Trust Systems Based on Both Individual Reputation and Social Relationship	200
7.5.1	Regret	200
7.5.2	NodeRanking	202
7.6	Trust Management	203
7.6.1	XenoTrust	205
7.6.2	EigenRep	207
7.6.3	Trust Management with P-Grid	210
7.7	Summary	212
8	P2P Programming Tools	215
8.1	Low Level P2P Programming	215
8.1.1	Sockets	216
8.1.2	Remote Procedure Call	216
8.1.3	Web Services	217
8.2	High Level P2P Programming	217
8.2.1	JXTA	218
8.2.2	BOINC	221
8.2.3	P2	222
8.2.4	Mace	223
8.2.5	OverlayWeaver	223
8.2.6	Microsoft's Peer-to-Peer Framework	224
8.3	Deployment and Testing Environments	225
8.3.1	PlanetLab	225
8.3.2	Emulab	226
8.3.3	Amazon.com	227
8.4	Summary	227

9	Systems and Applications	229
9.1	Classic File Sharing Systems	229
9.1.1	Napster	229
9.1.2	Gnutella	233
9.1.3	Freenet	237
9.2	Peer-to-Peer Backup	243
9.2.1	pStore	244
9.2.2	A Cooperative Internet Backup Scheme	247
9.2.3	Pastiche	248
9.2.4	Samsara—Fairness for Pastiche	250
9.2.5	Other Systems	251
9.2.6	Analysis of Existing Systems	252
9.3	Data Management	254
9.3.1	Architectures for P2P Data Management Systems	254
9.3.2	XML Content Routing Network	258
9.3.3	Continuous Query Processing	260
9.4	Peer-to-Peer-Based Web Caching	262
9.4.1	Background of Web Caching	262
9.4.2	Squirrel	263
9.4.3	BuddyWeb: A P2P-based Collaborative Web Caching System	266
9.5	Communication and Collaboration	270
9.5.1	Instant Messaging	270
9.5.2	Jabber	270
9.5.3	Skype	271
9.5.4	Distributed Collaboration	272
9.6	Mobile Applications	273
9.6.1	Communication Applications	273
9.6.2	File Sharing Applications	274
9.7	Summary	276
10	Conclusions	279
10.1	Summary	279
10.1.1	Architecture	279
10.1.2	Routing and Resource Discovery	280
10.1.3	Data-Centric Applications	281
10.1.4	Load Balancing and Replication	282
10.1.5	Programming Models	282
10.1.6	Security Problems	283
10.1.7	Trust Management	284
10.2	Potential Research Directions	285
10.2.1	Sharing Structured Databases	285
10.2.2	Security	286
10.2.3	Data Stream Processing	287
10.2.4	Testbed and Benchmarks	287

10.3 Applications in Industry 288

10.3.1 Supply Chain Management Case Study 289

References 293

Index 309