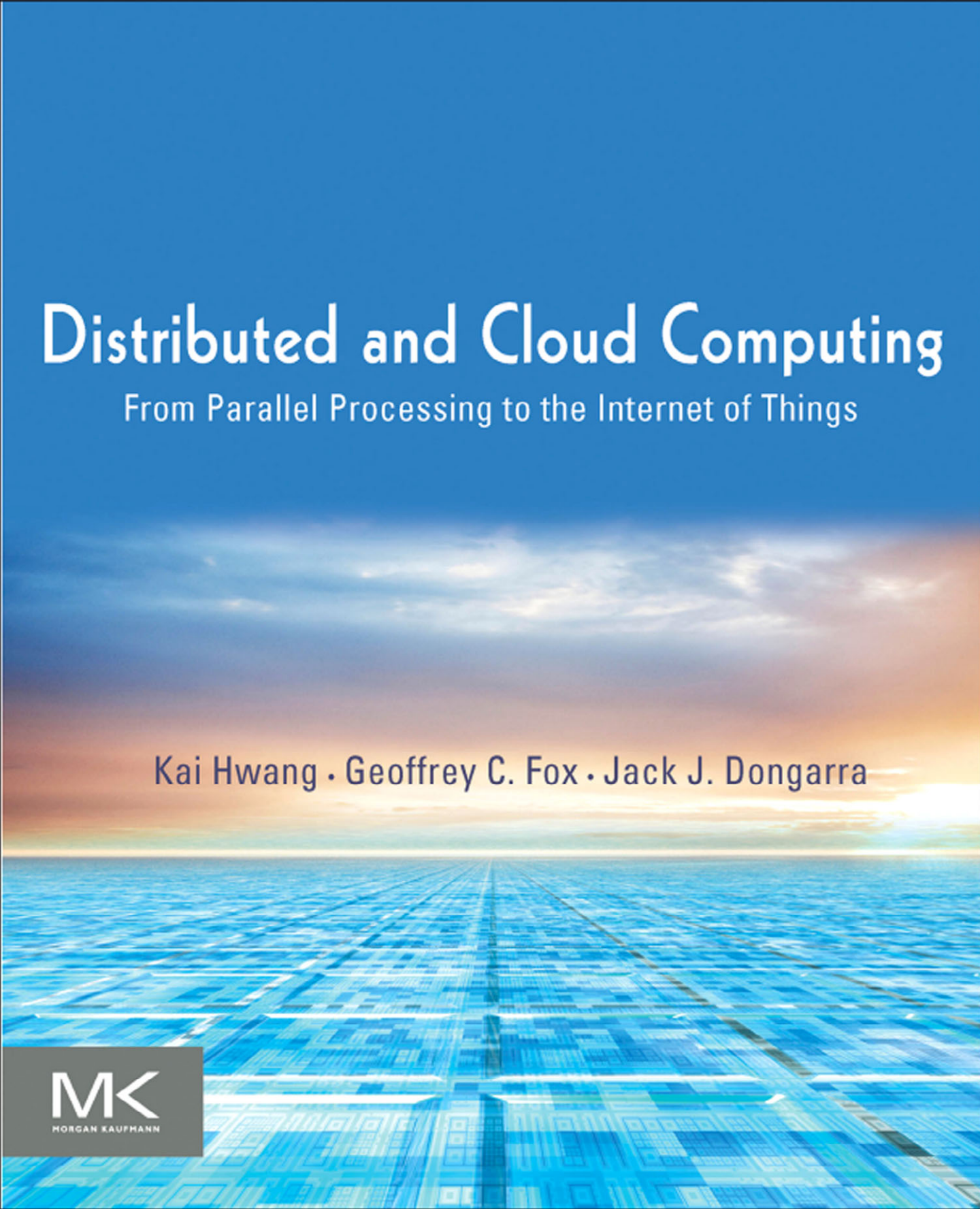
Distributed and Cloud Computing

From Parallel Processing to the Internet of Things

Kai Hwang · Geoffrey C. Fox · Jack J. Dongarra



Distributed and Cloud Computing

This page intentionally left blank

Distributed and Cloud Computing

From Parallel Processing to the Internet of Things

Kai Hwang

Geoffrey C. Fox

Jack J. Dongarra



AMSTERDAM • BOSTON • HEIDELBERG • LONDON
NEW YORK • OXFORD • PARIS • SAN DIEGO
SAN FRANCISCO • SINGAPORE • SYDNEY • TOKYO
Morgan Kaufmann is an imprint of Elsevier



Acquiring Editor: Todd Green
Development Editor: Robyn Day
Project Manager: Danielle S. Miller
Designer: Eric DeCicco

Morgan Kaufmann is an imprint of Elsevier
225 Wyman Street, Waltham, MA 02451, USA

© 2012 Elsevier, Inc. All rights reserved.

AMD, the AMD logo, AMD Opteron, and combinations thereof are trademarks of, and used with permission from, Advanced Micro Devices, Inc.

No part of this publication may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording, or any information storage and retrieval system, without permission in writing from the publisher. Details on how to seek permission, further information about the Publisher's permissions policies and our arrangements with organizations such as the Copyright Clearance Center and the Copyright Licensing Agency, can be found at our website: *www.elsevier.com/permissions*.

This book and the individual contributions contained in it are protected under copyright by the Publisher (other than as may be noted herein).

Notices
Knowledge and best practice in this field are constantly changing. As new research and experience broaden our understanding, changes in research methods or professional practices, may become necessary. Practitioners and researchers must always rely on their own experience and knowledge in evaluating and using any information or methods described herein. In using such information or methods they should be mindful of their own safety and the safety of others, including parties for whom they have a professional responsibility.

To the fullest extent of the law, neither the Publisher nor the authors, contributors, or editors, assume any liability for any injury and/or damage to persons or property as a matter of products liability, negligence or otherwise, or from any use or operation of any methods, products, instructions, or ideas contained in the material herein.

Library of Congress Cataloging-in-Publication Data
Application submitted

British Library Cataloguing-in-Publication Data
A catalogue record for this book is available from the British Library.

ISBN: 978-0-12-385880-1

For information on all MK publications visit our website at *www.mkp.com*

Typeset by: diacriTech, Chennai, India

Printed in the United States of America
13 14 15 10 9 8 7 6 5 4 3 2

Working together to grow libraries in developing countries

www.elsevier.com | www.bookaid.org | www.sabre.org

ELSEVIER BOOK AID International Sabre Foundation

This book is dedicated to our wives: Jennifer, Judy, and Sue; and to our children: Tony, Andrew, Katherine, Annie, Alexis, James, Heather, Pamela, Oliver, Nick, Ben, and Katie.

KH, GF, and JD

This page intentionally left blank

Contents

Preface xv
About the Authors xix
Foreword xxi

PART 1 SYSTEMS MODELING, CLUSTERING, AND VIRTUALIZATION 1

CHAPTER 1 Distributed System Models and Enabling Technologies 3
Summary 4
1.1 Scalable Computing over the Internet 4
1.1.1 The Age of Internet Computing 4
1.1.2 Scalable Computing Trends and New Paradigms 8
1.1.3 The Internet of Things and Cyber-Physical Systems 11
1.2 Technologies for Network-Based Systems 13
1.2.1 Multicore CPUs and Multithreading Technologies 14
1.2.2 GPU Computing to Exascale and Beyond 17
1.2.3 Memory, Storage, and Wide-Area Networking 20
1.2.4 Virtual Machines and Virtualization Middleware 22
1.2.5 Data Center Virtualization for Cloud Computing 25
1.3 System Models for Distributed and Cloud Computing 27
1.3.1 Clusters of Cooperative Computers 28
1.3.2 Grid Computing Infrastructures 29
1.3.3 Peer-to-Peer Network Families 32
1.3.4 Cloud Computing over the Internet 34
1.4 Software Environments for Distributed Systems and Clouds 36
1.4.1 Service-Oriented Architecture (SOA) 37
1.4.2 Trends toward Distributed Operating Systems 40
1.4.3 Parallel and Distributed Programming Models 42
1.5 Performance, Security, and Energy Efficiency 44
1.5.1 Performance Metrics and Scalability Analysis 45
1.5.2 Fault Tolerance and System Availability 48
1.5.3 Network Threats and Data Integrity 49
1.5.4 Energy Efficiency in Distributed Computing 51
1.6 Bibliographic Notes and Homework Problems 55
Acknowledgments 56
References 56
Homework Problems 58

CHAPTER 2 Computer Clusters for Scalable Parallel Computing 65
Summary 66
2.1 Clustering for Massive Parallelism 66
2.1.1 Cluster Development Trends 66
2.1.2 Design Objectives of Computer Clusters 68
2.1.3 Fundamental Cluster Design Issues 69
2.1.4 Analysis of the Top 500 Supercomputers 71
2.2 Computer Clusters and MPP Architectures 75
2.2.1 Cluster Organization and Resource Sharing 76
2.2.2 Node Architectures and MPP Packaging 77
2.2.3 Cluster System Interconnects 80
2.2.4 Hardware, Software, and Middleware Support 83
2.2.5 GPU Clusters for Massive Parallelism 83
2.3 Design Principles of Computer Clusters 87
2.3.1 Single-System Image Features 87
2.3.2 High Availability through Redundancy 95
2.3.3 Fault-Tolerant Cluster Configurations 99
2.3.4 Checkpointing and Recovery Techniques 101
2.4 Cluster Job and Resource Management 104
2.4.1 Cluster Job Scheduling Methods 104
2.4.2 Cluster Job Management Systems 107
2.4.3 Load Sharing Facility (LSF) for Cluster Computing 109
2.4.4 MOSIX: An OS for Linux Clusters and Clouds 110
2.5 Case Studies of Top Supercomputer Systems 112
2.5.1 Tianhe-1A: The World Fastest Supercomputer in 2010 112
2.5.2 Cray XT5 Jaguar: The Top Supercomputer in 2009 116
2.5.3 IBM Roadrunner: The Top Supercomputer in 2008 119
2.6 Bibliographic Notes and Homework Problems 120
Acknowledgments 121
References 121
Homework Problems 122

CHAPTER 3 Virtual Machines and Virtualization of Clusters and Data Centers 129
Summary 130
3.1 Implementation Levels of Virtualization 130
3.1.1 Levels of Virtualization Implementation 130
3.1.2 VMM Design Requirements and Providers 133
3.1.3 Virtualization Support at the OS Level 135
3.1.4 Middleware Support for Virtualization 138
3.2 Virtualization Structures/Tools and Mechanisms 140
3.2.1 Hypervisor and Xen Architecture 140
3.2.2 Binary Translation with Full Virtualization 141
3.2.3 Para-Virtualization with Compiler Support 143

3.3 Virtualization of CPU, Memory, and I/O Devices . . . 145
3.3.1 Hardware Support for Virtualization . . . 145
3.3.2 CPU Virtualization . . . 147
3.3.3 Memory Virtualization . . . 148
3.3.4 I/O Virtualization . . . 150
3.3.5 Virtualization in Multi-Core Processors . . . 153
3.4 Virtual Clusters and Resource Management . . . 155
3.4.1 Physical versus Virtual Clusters . . . 156
3.4.2 Live VM Migration Steps and Performance Effects . . . 159
3.4.3 Migration of Memory, Files, and Network Resources . . . 162
3.4.4 Dynamic Deployment of Virtual Clusters . . . 165
3.5 Virtualization for Data-Center Automation . . . 169
3.5.1 Server Consolidation in Data Centers . . . 169
3.5.2 Virtual Storage Management . . . 171
3.5.3 Cloud OS for Virtualized Data Centers . . . 172
3.5.4 Trust Management in Virtualized Data Centers . . . 176
3.6 Bibliographic Notes and Homework Problems . . . 179
Acknowledgments . . . 179
References . . . 180
Homework Problems . . . 183

PART 2 COMPUTING CLOUDS, SERVICE-ORIENTED ARCHITECTURE, AND PROGRAMMING 189

CHAPTER 4 Cloud Platform Architecture over Virtualized Data Centers . . . 191
Summary . . . 192
4.1 Cloud Computing and Service Models . . . 192
4.1.1 Public, Private, and Hybrid Clouds . . . 192
4.1.2 Cloud Ecosystem and Enabling Technologies . . . 196
4.1.3 Infrastructure-as-a-Service (IaaS) . . . 200
4.1.4 Platform-as-a-Service (PaaS) and Software-as-a-Service (SaaS) . . . 203
4.2 Data-Center Design and Interconnection Networks . . . 206
4.2.1 Warehouse-Scale Data-Center Design . . . 206
4.2.2 Data-Center Interconnection Networks . . . 208
4.2.3 Modular Data Center in Shipping Containers . . . 211
4.2.4 Interconnection of Modular Data Centers . . . 212
4.2.5 Data-Center Management Issues . . . 213
4.3 Architectural Design of Compute and Storage Clouds . . . 215
4.3.1 A Generic Cloud Architecture Design . . . 215
4.3.2 Layered Cloud Architectural Development . . . 218
4.3.3 Virtualization Support and Disaster Recovery . . . 221
4.3.4 Architectural Design Challenges . . . 225

4.4 Public Cloud Platforms: GAE, AWS, and Azure . . . 227
4.4.1 Public Clouds and Service Offerings . . . 227
4.4.2 Google App Engine (GAE) . . . 229
4.4.3 Amazon Web Services (AWS) . . . 231
4.4.4 Microsoft Windows Azure . . . 233
4.5 Inter-cloud Resource Management . . . 234
4.5.1 Extended Cloud Computing Services . . . 235
4.5.2 Resource Provisioning and Platform Deployment . . . 237
4.5.3 Virtual Machine Creation and Management . . . 243
4.5.4 Global Exchange of Cloud Resources . . . 246
4.6 Cloud Security and Trust Management . . . 249
4.6.1 Cloud Security Defense Strategies . . . 249
4.6.2 Distributed Intrusion/Anomaly Detection . . . 253
4.6.3 Data and Software Protection Techniques . . . 255
4.6.4 Reputation-Guided Protection of Data Centers . . . 257
4.7 Bibliographic Notes and Homework Problems . . . 261
Acknowledgements . . . 261
References . . . 261
Homework Problems . . . 265

CHAPTER 5 Service-Oriented Architectures for Distributed Computing . . . 271
Summary . . . 272
5.1 Services and Service-Oriented Architecture . . . 272
5.1.1 REST and Systems of Systems . . . 273
5.1.2 Services and Web Services . . . 277
5.1.3 Enterprise Multitier Architecture . . . 282
5.1.4 Grid Services and OGSA . . . 283
5.1.5 Other Service-Oriented Architectures and Systems . . . 287
5.2 Message-Oriented Middleware . . . 289
5.2.1 Enterprise Bus . . . 289
5.2.2 Publish-Subscribe Model and Notification . . . 291
5.2.3 Queuing and Messaging Systems . . . 291
5.2.4 Cloud or Grid Middleware Applications . . . 291
5.3 Portals and Science Gateways . . . 294
5.3.1 Science Gateway Exemplars . . . 295
5.3.2 HUBzero Platform for Scientific Collaboration . . . 297
5.3.3 Open Gateway Computing Environments (OGCE) . . . 301
5.4 Discovery, Registries, Metadata, and Databases . . . 304
5.4.1 UDDI and Service Registries . . . 304
5.4.2 Databases and Publish-Subscribe . . . 307
5.4.3 Metadata Catalogs . . . 308
5.4.4 Semantic Web and Grid . . . 309
5.4.5 Job Execution Environments and Monitoring . . . 312

5.5 Workflow in Service-Oriented Architectures . . . 314
5.5.1 Basic Workflow Concepts . . . 315
5.5.2 Workflow Standards . . . 316
5.5.3 Workflow Architecture and Specification . . . 317
5.5.4 Workflow Execution Engine . . . 319
5.5.5 Scripting Workflow System Swift . . . 321
5.6 Bibliographic Notes and Homework Problems . . . 322
Acknowledgements . . . 324
References . . . 324
Homework Problems . . . 331

CHAPTER 6 Cloud Programming and Software Environments . . . 335
Summary . . . 336
6.1 Features of Cloud and Grid Platforms . . . 336
6.1.1 Cloud Capabilities and Platform Features . . . 336
6.1.2 Traditional Features Common to Grids and Clouds . . . 336
6.1.3 Data Features and Databases . . . 340
6.1.4 Programming and Runtime Support . . . 341
6.2 Parallel and Distributed Programming Paradigms . . . 343
6.2.1 Parallel Computing and Programming Paradigms . . . 344
6.2.2 MapReduce, Twister, and Iterative MapReduce . . . 345
6.2.3 Hadoop Library from Apache . . . 355
6.2.4 Dryad and DryadLINQ from Microsoft . . . 359
6.2.5 Sawzall and Pig Latin High-Level Languages . . . 365
6.2.6 Mapping Applications to Parallel and Distributed Systems . . . 368
6.3 Programming Support of Google App Engine . . . 370
6.3.1 Programming the Google App Engine . . . 370
6.3.2 Google File System (GFS) . . . 373
6.3.3 BigTable, Google's NOSQL System . . . 376
6.3.4 Chubby, Google's Distributed Lock Service . . . 379
6.4 Programming on Amazon AWS and Microsoft Azure . . . 379
6.4.1 Programming on Amazon EC2 . . . 380
6.4.2 Amazon Simple Storage Service (S3) . . . 382
6.4.3 Amazon Elastic Block Store (EBS) and SimpleDB . . . 383
6.4.4 Microsoft Azure Programming Support . . . 384
6.5 Emerging Cloud Software Environments . . . 387
6.5.1 Open Source Eucalyptus and Nimbus . . . 387
6.5.2 OpenNebula, Sector/Sphere, and OpenStack . . . 389
6.5.3 Manjrasoft Aneka Cloud and Appliances . . . 393
6.6 Bibliographic Notes and Homework Problems . . . 399
Acknowledgement . . . 399
References . . . 399
Homework Problems . . . 405

PART 3 GRIDS, P2P, AND THE FUTURE INTERNET 413

CHAPTER 7 Grid Computing Systems and Resource Management 415

Summary 416

7.1 Grid Architecture and Service Modeling 416

7.1.1 Grid History and Service Families 416

7.1.2 CPU Scavenging and Virtual Supercomputers 419

7.1.3 Open Grid Services Architecture (OGSA) 422

7.1.4 Data-Intensive Grid Service Models 425

7.2 Grid Projects and Grid Systems Built 427

7.2.1 National Grids and International Projects 428

7.2.2 NSF TeraGrid in the United States 430

7.2.3 DataGrid in the European Union 431

7.2.4 The ChinaGrid Design Experiences 434

7.3 Grid Resource Management and Brokering 435

7.3.1 Resource Management and Job Scheduling 435

7.3.2 Grid Resource Monitoring with CGSP 437

7.3.3 Service Accounting and Economy Model 439

7.3.4 Resource Brokering with Gridbus 440

7.4 Software and Middleware for Grid Computing 443

7.4.1 Open Source Grid Middleware Packages 444

7.4.2 The Globus Toolkit Architecture (GT4) 446

7.4.3 Containers and Resources/Data Management 450

7.4.4 The ChinaGrid Support Platform (CGSP) 452

7.5 Grid Application Trends and Security Measures 455

7.5.1 Grid Applications and Technology Fusion 456

7.5.2 Grid Workload and Performance Prediction 457

7.5.3 Trust Models for Grid Security Enforcement 461

7.5.4 Authentication and Authorization Methods 464

7.5.5 Grid Security Infrastructure (GSI) 466

7.6 Bibliographic Notes and Homework Problems 470

Acknowledgments 471

References 471

Homework Problems 473

CHAPTER 8 Peer-to-Peer Computing and Overlay Networks 479

Summary 480

8.1 Peer-to-Peer Computing Systems 480

8.1.1 Basic Concepts of P2P Computing Systems 480

8.1.2 Fundamental Challenges in P2P Computing 486

8.1.3 Taxonomy of P2P Network Systems 490

8.2 P2P Overlay Networks and Properties 492

8.2.1 Unstructured P2P Overlay Networks 492

8.2.2 Distributed Hash Tables (DHTs) . . . 496
8.2.3 Structured P2P Overlay Networks . . . 498
8.2.4 Hierarchically Structured Overlay Networks . . . 501
8.3 Routing, Proximity, and Fault Tolerance . . . 505
8.3.1 Routing in P2P Overlay Networks . . . 505
8.3.2 Network Proximity in P2P Overlays . . . 507
8.3.3 Fault Tolerance and Failure Recovery . . . 509
8.3.4 Churn Resilience against Failures . . . 512
8.4 Trust, Reputation, and Security Management . . . 514
8.4.1 Peer Trust and Reputation Systems . . . 514
8.4.2 Trust Overlay and DHT Implementation . . . 517
8.4.3 PowerTrust: A Scalable Reputation System . . . 520
8.4.4 Securing Overlays to Prevent DDoS Attacks . . . 522
8.5 P2P File Sharing and Copyright Protection . . . 523
8.5.1 Fast Search, Replica, and Consistency . . . 524
8.5.2 P2P Content Delivery Networks . . . 529
8.5.3 Copyright Protection Issues and Solutions . . . 533
8.5.4 Collusive Piracy Prevention in P2P Networks . . . 535
8.6 Bibliographic Notes and Homework Problems . . . 538
Acknowledgements . . . 538
References . . . 539
Homework Problems . . . 541

CHAPTER 9 Ubiquitous Clouds and the Internet of Things . . . 545
Summary . . . 546
9.1 Cloud Trends in Supporting Ubiquitous Computing . . . 546
9.1.1 Use of Clouds for HPC/HTC and Ubiquitous Computing . . . 546
9.1.2 Large-Scale Private Clouds at NASA and CERN . . . 552
9.1.3 Cloud Mashups for Agility and Scalability . . . 555
9.1.4 Cloudlets for Mobile Cloud Computing . . . 558
9.2 Performance of Distributed Systems and the Cloud . . . 561
9.2.1 Review of Science and Research Clouds . . . 562
9.2.2 Data-Intensive Scalable Computing (DISC) . . . 566
9.2.3 Performance Metrics for HPC/HTC Systems . . . 568
9.2.4 Quality of Service in Cloud Computing . . . 572
9.2.5 Benchmarking MPI, Azure, EC2, MapReduce, and Hadoop . . . 574
9.3 Enabling Technologies for the Internet of Things . . . 576
9.3.1 The Internet of Things for Ubiquitous Computing . . . 576
9.3.2 Radio-Frequency Identification (RFID) . . . 580
9.3.3 Sensor Networks and ZigBee Technology . . . 582
9.3.4 Global Positioning System (GPS) . . . 587
9.4 Innovative Applications of the Internet of Things . . . 590
9.4.1 Applications of the Internet of Things . . . 591

9.4.2 Retailing and Supply-Chain Management 591
9.4.3 Smart Power Grid and Smart Buildings 594
9.4.4 Cyber-Physical System (CPS) 595
9.5 Online Social and Professional Networking 597
9.5.1 Online Social Networking Characteristics 597
9.5.2 Graph-Theoretic Analysis of Social Networks 600
9.5.3 Communities and Applications of Social Networks 603
9.5.4 Facebook: The World's Largest Social Network 608
9.5.5 Twitter for Microblogging, News, and Alert Services 611
9.6 Bibliographic Notes and Homework Problems 614
Acknowledgements 614
References 614
Homework Problems 618

Index 623

Preface

ABOUT THE BOOK

Over the past three decades, parallel processing and distributed computing have emerged as a well-developed field in computer science and information technology. Many universities and colleges are now offering standard courses in this field. However, the instructors and students are still in search of a comprehensive textbook that integrates computing theories and information technologies with the design, programming, and application of distributed systems. This book is designed to meet these demands. The book can be also used as a major reference for professionals working in this field.

The book addresses the latest advances in hardware and software, system architecture, new programming paradigms, and ecosystems emphasizing both speed performance and energy efficiency. These latest developments explain how to create high-performance clusters, scalable networks, automated data centers, and high-throughput cloud/grid systems. We also cover programming, and the use of distributed or cloud systems in innovative Internet applications. The book aims to transform traditional multiprocessors and multi-computer clusters into web-scale grids, clouds, and P2P networks for ubiquitous use in the future Internet, including large-scale social networks and the Internet of things that are emerging rapidly in recent years.

A GLANCE AT THE CONTENTS

We have included many milestone developments in a single volume. We present the contributions not only from our own research groups but also from leading research peers in the U.S., China, and Australia. Collectively, this group of authors and contributors summarize the progress that has been made in recent years, ranging from parallel processing to distributed computing and the future Internet.

Starting with an overview of modern distributed models, the text exposes the design principles, system architectures and innovative applications of parallel, distributed, and cloud computing systems. This book attempts to integrate parallel processing technologies with the network-based distributed systems. The book emphasizes scalable physical systems and virtualized data centers and cloud systems for research, e-commerce, social networking, supercomputing, and more applications, using concrete examples from open-source and commercial vendors.

The nine chapters are divided into three Parts: Part 1 covers system models and enabling technologies, including clustering and virtualization. Part 2 presents data center design, cloud computing platforms, service-oriented architectures, and distributed programming paradigms and software support. In Part 3, we study computational/data grids, peer-to-peer networks, ubiquitous clouds, the Internet of Things, and social networks.

Cloud computing material is addressed in six chapters (1, 3, 4, 5, 6, 9). Cloud systems presented include the public clouds: Google AppEngine, Amazon Web Service, Facebook, SalesForce.com, and many others. These cloud systems play an increasingly important role in upgrading the web services and Internet applications. Computer architects, software engineers, and system designers may want to explore the cloud technology to build the future computers and Internet-based systems.

KEY FEATURES

- Coverage of modern distributed computing technology including computer clusters, virtualization, service-oriented architecture, massively parallel processors, peer-to-peer systems, cloud computing, social networks, and the Internet of Things.
- Major emphases of the book lie in exploiting the ubiquity, agility, efficiency, scalability, availability, and programmability of parallel, distributed, and cloud computing systems.
- Latest developments in Hardware, Networks, and System Architecture:
  - Multi-core CPUs and Many-Core GPUs (Intel, Nvidia, AMD)
  - Virtual Machines and Virtual Clusters (CoD, Violin, Amazon VPC)
  - Top-500 Architectures (Tianbhe-1A, Jaguar, Roadrunner, etc.)
  - Google AppEngine, Amazon AWS, Microsoft Azure, IBM BlueCloud
  - TeraGrid, DataGrid, ChinaGrid, BOINIC, Grid5000 and FutureGrid
  - Chord, Napster, BiTorrent, KaZaA, PPlive, JXTA, and .NET
  - RFID, Sensor Networks, GPS, CPS, and the Internet of Things
  - Facebook, Force.Com, Twitter, SGI Cylone, Nebula, and GoGrid
- Recent advances in paradigms, programming, software and ecosystems:
  - MapReduce, Dryad, Hadoop, MPI, Twister, BigTable, DISC, etc
  - Cloud Service and Trust Models (SaaS, IaaS, PaaS, and PowerTrust)
  - Programming Languages and Protocols (Python, SOAP, UDDI, Pig Latin)
  - Virtualization Software (XEN, KVM, VMWare ESX, etc.)
  - Cloud OS and Meshups (Eucalyptus, Nimbus, OpenNebula, vShere/4, etc.)
  - Service-Oriented Architecture (REST, WS, Web 2.0, OGSA, etc.)
  - Distributed Operating Systems (DCE, Amoeba, and MOSIX)
  - Middleware and Software Libraries (LSF, Globus, Hadoop, Aneka)
- Over 100 examples are illustrated with 300 figures, designed to meet the need of students taking a distributed system course. Each chapter includes exercises and further reading.
- Included are case studies from the leading distributed computing vendors: Amazon, Google, Microsoft, IBM, HP, Sun, Silicon Graphics, Rackspace, SalesForce.com, netSuite, Enomaly, and many more.

READERSHIP AND SUGGESTIONS TO INSTRUCTORS/STUDENTS

The readership includes students taking a distributed systems or distributed computing class. Professional system designers and engineers may find this book useful as a reference to the latest distributed system technologies including clusters, grids, clouds, and the Internet of Things. The book gives a balanced coverage of all of these topics, looking into the future of Internet and IT evolutions.

The nine chapters are logically sequenced for use in an one-semester (45-hour lectures) course for seniors and graduate-level students. For use in a tri-semester system, Chapters 1, 2, 3, 4, 6, and 9 are suitable for a 10-week course (30-hour lectures). In addition to solving homework problems, the students are advised to conduct some parallel and distributed programming experiments on available clusters, grids, P2P, and cloud platforms. Sample projects and a solutions manual will be made available to proven instructors from Morgan Kaufmann, Publishers.

INVITED CONTRIBUTIONS

The book was jointly planned, authored, edited, and proofread by all three lead authors in four years (2007–2011). Over this period, we have invited and received partial contributions and technical assistance from the following scientists, researchers, instructors, and doctorial students from 10 top Universities in the U.S., China, and Australia.

Listed below are the invited contributors to this book. The authorship, contributed sections, and editorship of individual chapters are explicitly identified at the end of each chapter, separately. We want to thank their dedicated work and valuable contributions throughout the courses of repeated writing and revision process. The comments by the anonymous reviewers are also useful to improve the final contents.

Albert Zomaya, Nikzad Babaii Rizvandi, Young-Choon Lee, Ali Boloori, Reza Moraveji, Javid Taheri, and **Chen Wang**, Sydney University, Australia

Rajkumar Buyya, University of Melbourne, Australia

Yongwei Wu, Weimin Zheng, and **Kang Chen**, Tsinghua University, China

Zhenyu Li, Ninghui Sun, Zhiwei Xu, and **Gaogang Xie**, Chinese Academy of Sciences

Zhibin Yu, Xiaofei Liao and **Hai Jin**, Huazhong University of Science and Technology

Judy Qiu, Shrideep Pallickara, Marlon Pierce, Suresh Marru, Gregor Laszewski, Javier Diaz, Archit Kulshrestha, and **Andrew J. Younge**, Indiana University

Michael McLennan, George Adams III, and **Gerhard Klimeck**, Purdue University

Zhongyuan Qin, Kaikun Dong, Vikram Dixit, Xiaosong Lou, Sameer Kulkarni, Ken Wu, Zhou Zhao, and **Lizhong Chen**, University of Southern California

Renato Figueiredo, University of Florida

Michael Wilde, University of Chicago.

PERMISSIONS AND ACKNOWLEDGEMENTS

The permission to use copyrighted illustrations is openly acknowledged in the respective figure captions. In particular, we would like to thank Bill Dally, John Hopecroft, Mendel Roseblum, Dennis Gannon, Jon Kleinberg, Rajkumar Buyya, Albert Zomaya, Randel Bryant, Kang Chen, and Judy Qiu for their generosity in allowing us to use their presentation slides, original figures, and illustrated examples in this book.

We want to thank Ian Foster who wrote the visionary Foreword to introduce this book to our readers. The sponsorship by Todd Green and the editorial work of Robyn Day from Morgan Kaufmann Publishers and the production work led by Dennis Troutman of diacriTech are greatly appreciated. Without the collective effort of all of the above individuals, this book might be still in preparation. We hope that our readers will enjoy reading this timely-produced book and give us feedback for amending omissions and future improvements.

Kai Hwang, Geoffrey C. Fox, and Jack J. Dongarra

This page intentionally left blank

About the Authors

Kai Hwang is a Professor of Electrical Engineering and Computer Science, University of Southern California. Presently, he also serves as an Intellectual Venture-endowed Visiting Chair Professor at Tsinghua University in Beijing. He earned his Ph.D. in EECS from University of California at Berkeley in 1972. He has taught at Purdue University for 12 years prior to joining USC in 1985. Hwang has served as the founding Editor-in-Chief of the *Journal of Parallel and Distributed Computing* for 26 years. He is a world-renowned scholar and educator in computer science and engineering. He has published 8 books and 220 original papers in computer architecture, digital arithmetic, parallel processing, distributed systems, Internet security, and cloud computing.

Four of his published books: *Computer Arithmetic* (Wiley, 1978), *Computer Architecture and Parallel Processing* (McGraw-Hill, 1983), *Advanced Computer Architecture* (McGraw-Hill, 1993), and *Scalable Parallel Computing* (McGraw-Hill, 1998) have been translated into Spanish, Japanese, Chinese, and Korean from the English editions. By 2011, his published papers and books were cited over 9,000 times. Dr. Hwang has delivered three dozen keynote addresses in major IEEE/ACM conferences. In 1986, the IEEE Computer Society elevated him to an IEEE Fellow. He received the 2004 Outstanding Achievement Award from China Computer Federation. In May 2011, he received the IPDPS Founders' Award for his pioneering contributions in the field of parallel processing. For details, visit the web page: http://GridSec.usc.edu/hwang.html.

Geoffrey C. Fox is a Distinguished Professor of Informatics, Computing and Physics and Associate Dean of Graduate Studies and Research in the School of Informatics and Computing, Indiana University. He has taught and led many research groups at Caltech and Syracuse University, previously. He received his Ph.D. from Cambridge University, U.K. Fox is well known for his comprehensive work and extensive publications in parallel architecture, distributed programming, grid computing, web services, and Internet applications. His book on *Grid Computing* (coauthored with F. Berman and Tony Hey) is widely used by the research community. He has produced over 60 Ph.D. students in physics, computer science, and engineering over the years. Contact him via: gcf@indiana.edu.

Jack J. Dongarra is a University Distinguished Professor of Electrical Engineering and Computer Science, University of Tennessee, a Distinguished Research Staff, Oak Ridge National Laboratory, and a Turning Fellow at the University of Manchester. An ACM/IEEE/SIAM/AAAS Fellow, Dongarra pioneered the areas of supercomputer benchmarks, numerical analysis, PDE solvers, and high-performance computing and published extensively in these areas. He leads the Linpack benchmark evaluation of the Top-500 fastest computers over the years. Based on his high contributions in the supercomputing and high-performance areas, he was elected as a Member of the National Academy of Engineering in the U.S. Contact him at dongarra@eecs.utk.edu.

This page intentionally left blank

Foreword

Richard Feynman, in his wonderful autobiography *Surely You're Joking, Mr. Feynman*, recounts how at Los Alamos in 1944 he was responsible for supervising the human computers who performed the long and tedious calculations required by the Manhattan Project. Using the mechanical calculators that were then the state of the art, the best human computer could achieve only one addition or multiplication every few seconds. Feynman and his team thus developed methods for decomposing problems into smaller tasks that could be performed simultaneously by different people (they passed cards with intermediate results between people operating adders, multipliers, collators, and sorters); for running multiple computations at once in the same computing complex (they used different color cards); for prioritizing a more important computation (they eliminated cards of other colors); and for detecting and recovering efficiently from errors (relevant cards, and their descendants, were removed, and computations restarted).

Seventy years later, computer architects face similar challenges—and have adopted similar solutions. Individual computing devices are far faster, but physical constraints still limit their speed. Thus, today's computing landscape is characterized by pervasive parallelism. Individual processors incorporate pipelining, parallel instructions, speculative execution, and multithreading. Essentially every computer system, from the humblest desktop to the most powerful supercomputer, incorporates multiple processors. Designers of future exascale supercomputers, to be capable of 10^{18} operations per second, tell us that these computers will need to support 10^{7} concurrent operations.

Parallelism is fundamentally about communication and coordination, and those two activities have also been transformed over the past seventy years by dramatic technological change. Light is no faster, at 8 inches or 20 centimeters per nanosecond in fiber, than in Feynman's time; one can never expect to send a message in less than 50 milliseconds from Los Angeles to Auckland. But the rate at which data can be transmitted has changed dramatically, from a few characters per second in 1910 (early telegraphs) to thousands of characters per second in 1960 (ARPANET) to more than 10 billion characters per second over optical fibers in 2010.

Quasi-ubiquitous high-speed communications not only allows call centers to be relocated to India, it also allows computation to be moved to centralized facilities that achieve massive economies of scale, and permits enormous quantities of data to be collected and organized to support decision making by people worldwide. Thus, government agencies, research laboratories, and companies who need to simulate complex phenomena create and operate enormous supercomputers with hundreds of thousands of processors. Similarly, companies such as Google, Facebook, and Microsoft who need to process large quantities of data operate numerous massive "cloud" data centers that may each occupy tens of thousands of square feet and contain tens or hundreds of thousands of computers. Like Feynman's Los Alamos team, these computing complexes provide computing as a service for many people, and must juggle many computations performed for different purposes.

Massive parallelism, ultra-fast communication, and massive centralization are all fundamental to human decision making today. The computations that are used to forecast tomorrow's weather, index the web, recommend movies, suggest social connections, predict the future state of the stock market, or provide any one of a multitude of other desirable information products are typically distributed over thousands of processors and depend on data collected from sometimes millions of

sources worldwide. Indeed, little of the modern world could function as it does without parallel and distributed computing.

In this pervasively parallel and distributed world, an understanding of distributed computing is surely an essential part of any undergraduate education in computer science. (Indeed, I would argue, an understanding of these topics should be an essential part of *any* undergraduate education. But I leave that argument for another time.) The most complex computer systems today are no longer individual microprocessors, but entire data centers. The most complex computer programs written today are those that manage or run on data-center-scale systems. A student who graduates with a degree in computer science and does not understand how these systems and programs are constructed is profoundly unprepared to engage productively in the modern workforce.

Hwang, Fox, and Dongarra's text is thus especially timely. In its three sections, it covers progressively the hardware and software architectures that underpin modern massively parallel computer systems; the concepts and technologies that enable cloud and distributed computing; and advanced topics in distributed computing, including grid, peer-to-peer, and the Internet of Things. In each area, the text takes a systems approach, describing not only concepts but also representative technologies and realistic large-scale distributed computing deployments. Computing is as much an engineering discipline as a science, and these descriptions of real systems will both prepare students to use those systems and help them understand how other architects have maneuvered the constraints associated with large-scale distributed system design.

The text also addresses some of the more challenging issues facing computer science researchers today. To name just two, computers have emerged as a major consumer of electricity, accounting for several percent of all electricity used in the US. (In Japan, it is ironic that following the 2011 tsunami, the large supercomputers that may help prepare for future natural disasters must often be turned off to conserve power.) And, the fact that 98% of the roughly 10 billion processors sold each year are for embedded devices, and that these embedded devices are increasingly communication-enabled, introduces the opportunity and challenge of an "Internet of Things" that will be vastly larger, more complex, and more capable than today's Internet of People.

I hope that the appearance of this book will stimulate more teaching of distributed computing in universities and colleges—and not just as an optional topic, as is too often the case, but as a core element of the undergraduate curriculum. I hope also that others outside universities will take this opportunity to learn about distributed computing, and more broadly about what computing looks like on the cutting edge: sometimes messy; often complex; but above all tremendously exciting.

Ian Foster
Jackson Hole, Wyoming
August, 2011

PART

1 Systems Modeling, Clustering, and Virtualization

The first three chapters cover systems models and review two enabling technologies. We model distributed systems and cloud platforms in Chapter 1. The clustering technology is presented in Chapter 2, and virtualization technology in Chapter 3. These two technologies enable distributed and cloud computing. The system models include computer clusters, computing grid, P2P networks, and cloud computing platform. System clustering is supported by hardware, software, and middleware advances. Virtualization creates virtual machines, virtualized clusters, automation of datacenters, and building of elastic cloud platforms.

CHAPTER 1: DISTRIBUTED SYSTEM MODELS AND ENABLING TECHNOLOGIES

This introductory chapter assesses the evolutional changes in computing and IT trends in the past 30 years. We study both high-performance computing (HPC) for scientific computing and high-throughput computing (HTC) systems for business computing. We examine clusters/MPP, grids, P2P networks, and Internet clouds. These systems are distinguished by their platform architectures, OS platforms, processing algorithms, communication protocols, security demands, and service models applied. The study emphasizes the scalability, performance, availability, security, energy-efficiency, workload outsourcing, data center protection, and so on.

This chapter is authored by Kai Hwang with partial contributions by Geoffrey Fox (Section 1.4.1) and Albert Zomaya (Section 1.5.4), and assisted by Nikzad Babaii Rizvandi, Young-Choon Lee, Xiaosong Lou, and Lizhong Chen. The final manuscript was edited by Jack Dongarra.

CHAPTER 2: COMPUTER CLUSTERS FOR SCALABLE PARALLEL COMPUTING

Clustering enables the construction of scalable parallel and distributed systems for both HPC and HTC applications. Today's cluster nodes are built with either physical servers or virtual machines. In this chapter, we study clustered computing systems and massively parallel processors. We focus on design principles and assess the hardware, software, and middleware support needed. We study scalability, availability, programmability, single system image, job management, and fault tolerance. We study the clustering MPP architectures in three top supercomputers reported in recent years, namely China's Tiahe-1A, the Cray XT5 Jaguar, and IBM's RoadRunner.

This chapter is coauthored by Kai Hwang and Jack Dongarra with partial contributions by Rajkumar Buyya and Ninghui Sun. Special technical assistances are from Zhiwei Xu, Zhou Zhao, Xiaosong Lou, and Lizhong Chen.

CHAPTER 3: VIRTUAL MACHINES AND VIRTUALIZATION OF CLUSTERS AND DATA CENTERS

Virtualization technology was primarily designed for the sharing of expensive hardware resources by multiplexing virtual machines (VM) on the same set of hardware hosts. The surge of interest in installing virtual machines has widened the scope of system applications and upgraded computer performance and efficiency in recent years. We study VMs, live migration of VMs, virtual cluster construction, resource provisioning, virtual configuration adaptation, and the design of virtualized data centers for cloud computing applications. We emphasize the roles of using virtual clusters and virtualized resource management for building dynamic grids and cloud platforms.

This chapter is coauthored by Zhibin Yu and Kai Hwang with technical assistance from Hai Jin, Xiaofei Liao, Chongyuan Qin, Lizhong Chen, and Zhou Zhao.

CHAPTER

Distributed System Models and Enabling Technologies

1

CHAPTER OUTLINE

Summary **4**
1.1 Scalable Computing over the Internet **4**
1.1.1 The Age of Internet Computing 4
1.1.2 Scalable Computing Trends and New Paradigms 8
1.1.3 The Internet of Things and Cyber-Physical Systems 11
1.2 Technologies for Network-based Systems **13**
1.2.1 Multicore CPUs and Multithreading Technologies 14
1.2.2 GPU Computing to Exascale and Beyond 17
1.2.3 Memory, Storage, and Wide-Area Networking 20
1.2.4 Virtual Machines and Virtualization Middleware 22
1.2.5 Data Center Virtualization for Cloud Computing 25
1.3 System Models for Distributed and Cloud Computing **27**
1.3.1 Clusters of Cooperative Computers 28
1.3.2 Grid Computing Infrastructures 29
1.3.3 Peer-to-Peer Network Families 32
1.3.4 Cloud Computing over the Internet 34
1.4 Software Environments for Distributed Systems and Clouds **36**
1.4.1 Service-Oriented Architecture (SOA) 37
1.4.2 Trends toward Distributed Operating Systems 40
1.4.3 Parallel and Distributed Programming Models 42
1.5 Performance, Security, and Energy Efficiency **44**
1.5.1 Performance Metrics and Scalability Analysis 45
1.5.2 Fault Tolerance and System Availability 48
1.5.3 Network Threats and Data Integrity 49
1.5.4 Energy Efficiency in Distributed Computing 51
1.6 Bibliographic Notes and Homework Problems **55**
Acknowledgments **56**
References **56**
Homework Problems **58**

© 2012 Elsevier, Inc. All rights reserved.

SUMMARY

This chapter presents the evolutionary changes that have occurred in parallel, distributed, and cloud computing over the past 30 years, driven by applications with variable workloads and large data sets. We study both high-performance and high-throughput computing systems in parallel computers appearing as computer clusters, service-oriented architecture, computational grids, peer-to-peer networks, Internet clouds, and the Internet of Things. These systems are distinguished by their hardware architectures, OS platforms, processing algorithms, communication protocols, and service models applied. We also introduce essential issues on the scalability, performance, availability, security, and energy efficiency in distributed systems.

1.1 SCALABLE COMPUTING OVER THE INTERNET

Over the past 60 years, computing technology has undergone a series of platform and environment changes. In this section, we assess evolutionary changes in machine architecture, operating system platform, network connectivity, and application workload. Instead of using a centralized computer to solve computational problems, a parallel and distributed computing system uses multiple computers to solve large-scale problems over the Internet. Thus, distributed computing becomes data-intensive and network-centric. This section identifies the applications of modern computer systems that practice parallel and distributed computing. These large-scale Internet applications have significantly enhanced the quality of life and information services in society today.

1.1.1 The Age of Internet Computing

Billions of people use the Internet every day. As a result, supercomputer sites and large data centers must provide high-performance computing services to huge numbers of Internet users concurrently. Because of this high demand, the Linpack Benchmark for *high-performance computing (HPC)* applications is no longer optimal for measuring system performance. The emergence of computing clouds instead demands *high-throughput computing (HTC)* systems built with parallel and distributed computing technologies [5,6,19,25]. We have to upgrade data centers using fast servers, storage systems, and high-bandwidth networks. The purpose is to advance network-based computing and web services with the emerging new technologies.

1.1.1.1 The Platform Evolution

Computer technology has gone through five generations of development, with each generation lasting from 10 to 20 years. Successive generations are overlapped in about 10 years. For instance, from 1950 to 1970, a handful of mainframes, including the IBM 360 and CDC 6400, were built to satisfy the demands of large businesses and government organizations. From 1960 to 1980, lower-cost minicomputers such as the DEC PDP 11 and VAX Series became popular among small businesses and on college campuses.

From 1970 to 1990, we saw widespread use of personal computers built with VLSI microprocessors. From 1980 to 2000, massive numbers of portable computers and pervasive devices appeared in both wired and wireless applications. Since 1990, the use of both HPC and HTC systems hidden in

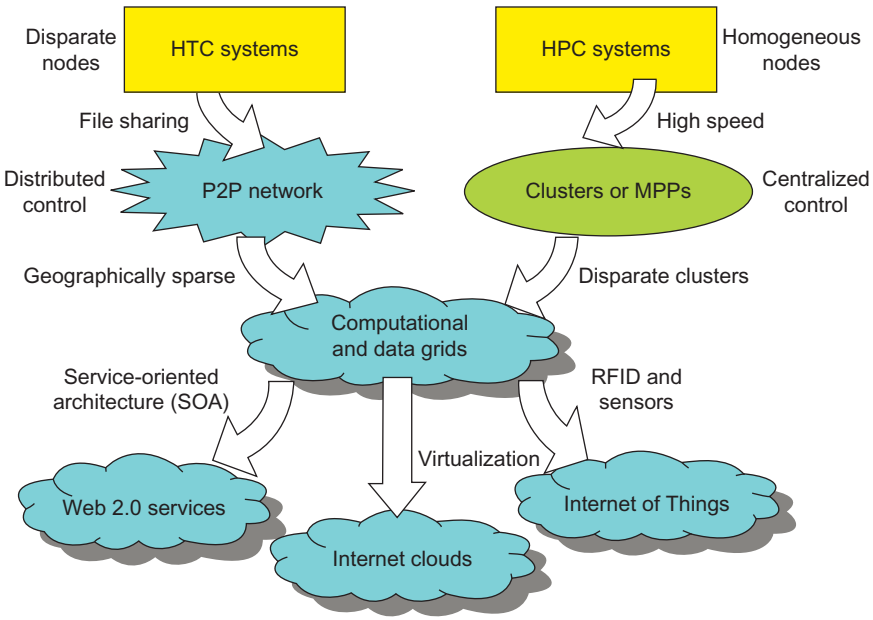


FIGURE 1.1

Evolutionary trend toward parallel, distributed, and cloud computing with clusters, MPPs, P2P networks, grids, clouds, web services, and the Internet of Things.

clusters, grids, or Internet clouds has proliferated. These systems are employed by both consumers and high-end web-scale computing and information services.

The general computing trend is to leverage shared web resources and massive amounts of data over the Internet. Figure 1.1 illustrates the evolution of HPC and HTC systems. On the HPC side, supercomputers (*massively parallel processors* or *MPPs*) are gradually replaced by clusters of cooperative computers out of a desire to share computing resources. The cluster is often a collection of homogeneous compute nodes that are physically connected in close range to one another. We will discuss clusters, MPPs, and grid systems in more detail in Chapters 2 and 7.

On the HTC side, *peer-to-peer (P2P)* networks are formed for distributed file sharing and content delivery applications. A P2P system is built over many client machines (a concept we will discuss further in Chapter 5). Peer machines are globally distributed in nature. P2P, cloud computing, and web service platforms are more focused on HTC applications than on HPC applications. Clustering and P2P technologies lead to the development of computational grids or data grids.

1.1.1.2 High-Performance Computing

For many years, HPC systems emphasize the raw speed performance. The speed of HPC systems has increased from Gflops in the early 1990s to now Pflops in 2010. This improvement was driven mainly by the demands from scientific, engineering, and manufacturing communities. For example,

the Top 500 most powerful computer systems in the world are measured by floating-point speed in Linpack benchmark results. However, the number of supercomputer users is limited to less than 10% of all computer users. Today, the majority of computer users are using desktop computers or large servers when they conduct Internet searches and market-driven computing tasks.

1.1.1.3 High-Throughput Computing

The development of market-oriented high-end computing systems is undergoing a strategic change from an HPC paradigm to an HTC paradigm. This HTC paradigm pays more attention to high-flux computing. The main application for high-flux computing is in Internet searches and web services by millions or more users simultaneously. The performance goal thus shifts to measure *high throughput* or the number of tasks completed per unit of time. HTC technology needs to not only improve in terms of batch processing speed, but also address the acute problems of cost, energy savings, security, and reliability at many data and enterprise computing centers. This book will address both HPC and HTC systems to meet the demands of all computer users.

1.1.1.4 Three New Computing Paradigms

As Figure 1.1 illustrates, with the introduction of SOA, Web 2.0 services become available. Advances in virtualization make it possible to see the growth of Internet clouds as a new computing paradigm. The maturity of *radio-frequency identification (RFID)*, *Global Positioning System (GPS)*, and sensor technologies has triggered the development of the *Internet of Things (IoT)*. These new paradigms are only briefly introduced here. We will study the details of SOA in Chapter 5; virtualization in Chapter 3; cloud computing in Chapters 4, 6, and 9; and the IoT along with cyber-physical systems (CPS) in Chapter 9.

When the Internet was introduced in 1969, Leonard Klienrock of UCLA declared: "As of now, computer networks are still in their infancy, but as they grow up and become sophisticated, we will probably see the spread of computer utilities, which like present electric and telephone utilities, will service individual homes and offices across the country." Many people have redefined the term "computer" since that time. In 1984, John Gage of Sun Microsystems created the slogan, "The network is the computer." In 2008, David Patterson of UC Berkeley said, "The data center is the computer. There are dramatic differences between developing software for millions to use as a service versus distributing software to run on their PCs." Recently, Rajkumar Buyya of Melbourne University simply said: "The cloud is the computer."

This book covers clusters, MPPs, P2P networks, grids, clouds, web services, social networks, and the IoT. In fact, the differences among clusters, grids, P2P systems, and clouds may blur in the future. Some people view clouds as grids or clusters with modest changes through virtualization. Others feel the changes could be major, since clouds are anticipated to process huge data sets generated by the traditional Internet, social networks, and the future IoT. In subsequent chapters, the distinctions and dependencies among all distributed and cloud systems models will become clearer and more transparent.

1.1.1.5 Computing Paradigm Distinctions

The high-technology community has argued for many years about the precise definitions of centralized computing, parallel computing, distributed computing, and cloud computing. In general, *distributed computing* is the opposite of *centralized computing*. The field of *parallel computing*

overlaps with distributed computing to a great extent, and *cloud computing* overlaps with distributed, centralized, and parallel computing. The following list defines these terms more clearly; their architectural and operational differences are discussed further in subsequent chapters.

- **Centralized computing** This is a computing paradigm by which all computer resources are centralized in one physical system. All resources (processors, memory, and storage) are fully shared and tightly coupled within one integrated OS. Many data centers and supercomputers are *centralized systems*, but they are used in parallel, distributed, and cloud computing applications [18,26].
- **Parallel computing** In parallel computing, all processors are either tightly coupled with centralized shared memory or loosely coupled with distributed memory. Some authors refer to this discipline as *parallel processing* [15,27]. Interprocessor communication is accomplished through shared memory or via message passing. A computer system capable of parallel computing is commonly known as a *parallel computer* [28]. Programs running in a parallel computer are called *parallel programs*. The process of writing *parallel programs* is often referred to as *parallel programming* [32].
- **Distributed computing** This is a field of computer science/engineering that studies distributed systems. A *distributed system* [8,13,37,46] consists of multiple autonomous computers, each having its own private memory, communicating through a computer network. Information exchange in a distributed system is accomplished through *message passing*. A computer program that runs in a distributed system is known as a *distributed program*. The process of writing distributed programs is referred to as *distributed programming*.
- **Cloud computing** An *Internet cloud* of resources can be either a centralized or a distributed computing system. The cloud applies parallel or distributed computing, or both. Clouds can be built with physical or virtualized resources over large data centers that are centralized or distributed. Some authors consider cloud computing to be a form of *utility computing* or *service computing* [11,19].

As an alternative to the preceding terms, some in the high-tech community prefer the term *concurrent computing* or *concurrent programming*. These terms typically refer to the union of parallel computing and distributing computing, although biased practitioners may interpret them differently. *Ubiquitous computing* refers to computing with pervasive devices at any place and time using wired or wireless communication. The *Internet of Things* (IoT) is a networked connection of everyday objects including computers, sensors, humans, etc. The IoT is supported by Internet clouds to achieve ubiquitous computing with any object at any place and time. Finally, the term *Internet computing* is even broader and covers all computing paradigms over the Internet. This book covers all the aforementioned computing paradigms, placing more emphasis on distributed and cloud computing and their working systems, including the clusters, grids, P2P, and cloud systems.

1.1.1.6 Distributed System Families

Since the mid-1990s, technologies for building P2P networks and *networks of clusters* have been consolidated into many national projects designed to establish wide area computing infrastructures, known as *computational grids* or *data grids*. Recently, we have witnessed a surge in interest in exploring Internet cloud resources for data-intensive applications. Internet clouds are the result of moving desktop computing to service-oriented computing using server clusters and huge databases