

Synchronization Algorithms And Concurrent Programming

Right here, we have countless books **Synchronization Algorithms And Concurrent Programming** and collections to check out. We additionally present variant types and after that type of the books to browse. The usual book, fiction, history, novel, scientific research, as without difficulty as various new sorts of books are readily friendly here.

As this Synchronization Algorithms And Concurrent Programming, it ends occurring being one of the favored books Synchronization Algorithms And Concurrent Programming collections that we have. This is why you remain in the best website to see the incredible book to have.

Euro-Par 2010 - Parallel Processing - Pasqua D'Ambra 2010-09-02

Annotation This book constitutes the refereed proceedings of the 16th International Euro-Par Conference held in Ischia, Italy, in August/September 2010. The 90 revised full papers presented were carefully reviewed and selected from 256 submissions. The papers are organized in topical sections on support tools and environments; performance prediction and evaluation; scheduling and load-balancing; high performance architectures and compilers; parallel and distributed data management; grid, cluster and cloud computing; peer to peer computing; distributed systems and algorithms; parallel and distributed programming; parallel numerical algorithms; multicore and manycore programming; theory and algorithms for parallel computation; high performance networks; and mobile and ubiquitous computing.

Euro-Par 2015: Parallel Processing - Jesper Larsson Träff 2015-07-24

This book constitutes the refereed proceedings of the 21st International Conference on Parallel and Distributed Computing, Euro-Par 2015, held in Vienna, Austria, in August 2015. The 51 revised full papers presented together with 2 invited papers were carefully reviewed and selected from 190 submissions. The papers are organized in the following topical sections: support tools and environments; performance modeling, prediction and evaluation; scheduling and load balancing; architecture and compilers; parallel and distributed data management; grid, cluster and cloud computing; distributed systems and algorithms; parallel and distributed programming, interfaces and languages; multi- and many-core programming; theory and algorithms for parallel computation; numerical methods and applications; and accelerator computing.

Principles of Distributed Systems - Michel Raynal 2009-12

OPODIS, the International Conference on Principles of Distributed Systems, is an annual forum for presentation of state-of-the-art knowledge on principles of distributed computing systems, including theory, design, analysis, implementation and application of distributed systems, among researchers from around the world. The 13th edition of OPODIS was held during December 15-18, in Nimes, France. There were 71 submissions, and this volume contains the 23 regular contributions and the 4 brief announcements selected by the Program Committee. All submitted papers were read and evaluated by three to five PC members assisted by external reviewers. The final decision regarding every paper was taken after long discussions through EasyChair. This year the Best Paper Award was shared by two papers: "On the Computational Power of Shared Objects" by Gadi Taubenfeld and "Transactional Scheduling for Read-Dominated Workloads" by Hagit Attiya and Alessia Milani. The Best Student Paper Award was given to the paper "Decentralized Polling with Respectable Participants" co-authored by Kevin Huguenin and Maxime Monod and their advisors. The conference also featured two very interesting invited talks by Anne-Marie Kermarrec and Maurice Herlihy. Anne-Marie's talk was on "Navigating Web 2.0 with Gossple" and Maurice's talk was on "Transactional Memory Today: A Status Report." OPODIS has now found its place among the international conferences related to principles of distributed computing and distributed systems. We hope that this 13th edition will contribute to the growth and the development of the conference and continue to increase its visibility. Finally we would like to thank Nicola Santoro, Conference General Chair, Hachem Fouchal, Steering Committee Chair, and Bernard Thibault for their constant help.

Algorithms and Parallel Computing - Fayez Gebali 2011-03-29

There is a software gap between the hardware potential and the performance that can be attained using

today's software parallel program development tools. The tools need manual intervention by the programmer to parallelize the code. Programming a parallel computer requires closely studying the target algorithm or application, more so than in the traditional sequential programming we have all learned. The programmer must be aware of the communication and data dependencies of the algorithm or application. This book provides the techniques to explore the possible ways to program a parallel computer for a given application.

Mastering Concurrency Programming with Java 9 - Javier Fernandez Gonzalez 2017-07-17

Master the principles to make applications robust, scalable and responsive About This Book Implement concurrent applications using the Java 9 Concurrency API and its new components Improve the performance of your applications and process more data at the same time, taking advantage of all of your resources Construct real-world examples related to machine learning, data mining, natural language processing, and more Who This Book Is For This book is for competent Java developers who have basic understanding of concurrency, but knowledge of effective implementation of concurrent programs or usage of streams for making processes more efficient is not required What You Will Learn Master the principles that every concurrent application must follow See how to parallelize a sequential algorithm to obtain better performance without data inconsistencies and deadlocks Get the most from the Java Concurrency API components Separate the thread management from the rest of the application with the Executor component Execute phased-based tasks in an efficient way with the Phaser components Solve problems using a parallelized version of the divide and conquer paradigm with the Fork / Join framework Find out how to use parallel Streams and Reactive Streams Implement the "map and reduce" and "map and collect" programming models Control the concurrent data structures and synchronization mechanisms provided by the Java Concurrency API Implement efficient solutions for some actual problems such as data mining, machine learning, and more In Detail Concurrency programming allows several large tasks to be divided into smaller sub-tasks, which are further processed as individual tasks that run in parallel. Java 9 includes a comprehensive API with lots of ready-to-use components for easily implementing powerful concurrency applications, but with high flexibility so you can adapt these components to your needs. The book starts with a full description of the design principles of concurrent applications and explains how to parallelize a sequential algorithm. You will then be introduced to Threads and Runnable, which are an integral part of Java 9's concurrency API. You will see how to use all the components of the Java concurrency API, from the basics to the most advanced techniques, and will implement them in powerful real-world concurrency applications. The book ends with a detailed description of the tools and techniques you can use to test a concurrent Java application, along with a brief insight into other concurrency mechanisms in JVM. Style and approach This is a complete guide that implements real-world examples of algorithms related to machine learning, data mining, and natural language processing in client/server environments. All the examples are explained using a step-by-step approach.

Scalable Parallel Programming Applied to H.264/AVC Decoding - Ben Juurlink 2012-06-01

Existing software applications should be redesigned if programmers want to benefit from the performance offered by multi- and many-core architectures. Performance scalability now depends on the possibility of finding and exploiting enough Thread-Level Parallelism (TLP) in applications for using the increasing numbers of cores on a chip. Video decoding is an example of an application domain with increasing

computational requirements every new generation. This is due, on the one hand, to the trend towards high quality video systems (high definition and frame rate, 3D displays, etc) that results in a continuous increase in the amount of data that has to be processed in real-time. On the other hand, there is the requirement to maintain high compression efficiency which is only possible with video codes like H.264/AVC that use advanced coding techniques. In this book, the parallelization of H.264/AVC decoding is presented as a case study of parallel programming. H.264/AVC decoding is an example of a complex application with many levels of dependencies, different kernels, and irregular data structures. The book presents a detailed methodology for parallelization of this type of applications. It begins with a description of the algorithm, an analysis of the data dependencies and an evaluation of the different parallelization strategies. Then the design and implementation of a novel parallelization approach is presented that is scalable to many core architectures. Experimental results on different parallel architectures are discussed in detail. Finally, an outlook is given on parallelization opportunities in the upcoming HEVC standard.

Pro TBB - Michael Voss 2019-07-09

This open access book is a modern guide for all C++ programmers to learn Threading Building Blocks (TBB). Written by TBB and parallel programming experts, this book reflects their collective decades of experience in developing and teaching parallel programming with TBB, offering their insights in an approachable manner. Throughout the book the authors present numerous examples and best practices to help you become an effective TBB programmer and leverage the power of parallel systems. Pro TBB starts with the basics, explaining parallel algorithms and C++'s built-in standard template library for parallelism. You'll learn the key concepts of managing memory, working with data structures and how to handle typical issues with synchronization. Later chapters apply these ideas to complex systems to explain performance tradeoffs, mapping common parallel patterns, controlling threads and overhead, and extending TBB to program heterogeneous systems or system-on-chips. What You'll Learn Use Threading Building Blocks to produce code that is portable, simple, scalable, and more understandable Review best practices for parallelizing computationally intensive tasks in your applications Integrate TBB with other threading packages Create scalable, high performance data-parallel programs Work with generic programming to write efficient algorithms Who This Book Is For C++ programmers learning to run applications on multicore systems, as well as C or C++ programmers without much experience with templates. No previous experience with parallel programming or multicore processors is required.

Concurrent Programming on Windows - Joe Duffy 2009

This practical book includes a tutorial of the entire set of Windows and .NET APIs required to write concurrent programs. Because so much of the threading and synchronization features of the platform are Windows-general, the author, Joe Duffy, focuses first on the general behavior and then on the API details of native and managed code. Interspersed among the tutorial are many difficult-to-discover, useful insights, and internal details about how things work.

Parallel and Distributed Computation: Numerical Methods - Dimitri Bertsekas 2015-03-01

This highly acclaimed work, first published by Prentice Hall in 1989, is a comprehensive and theoretically sound treatment of parallel and distributed numerical methods. It focuses on algorithms that are naturally suited for massive parallelization, and it explores the fundamental convergence, rate of convergence, communication, and synchronization issues associated with such algorithms. This is an extensive book, which aside from its focus on parallel and distributed algorithms, contains a wealth of material on a broad variety of computation and optimization topics. It is an excellent supplement to several of our other books, including *Convex Optimization Algorithms* (Athena Scientific, 2015), *Nonlinear Programming* (Athena Scientific, 1999), *Dynamic Programming and Optimal Control* (Athena Scientific, 2012), *Neuro-Dynamic Programming* (Athena Scientific, 1996), and *Network Optimization* (Athena Scientific, 1998). The on-line edition of the book contains a 95-page solutions manual.

Encyclopedia of Algorithms - Ming-Yang Kao 2008-08-06

One of Springer's renowned Major Reference Works, this awesome achievement provides a comprehensive set of solutions to important algorithmic problems for students and researchers interested in quickly locating useful information. This first edition of the reference focuses on high-impact solutions from the most recent decade, while later editions will widen the scope of the work. All entries have been written by

experts, while links to Internet sites that outline their research work are provided. The entries have all been peer-reviewed. This defining reference is published both in print and on line.

Concurrent Patterns and Best Practices - Atul S. Khot 2018-09-27

A definitive guide to mastering and implementing concurrency patterns in your applications Key Features Build scalable apps with patterns in multithreading, synchronization, and functional programming Explore the parallel programming and multithreading techniques to make the code run faster Efficiently use the techniques outlined to build reliable applications Book Description Selecting the correct concurrency architecture has a significant impact on the design and performance of your applications. This book explains how to leverage the different characteristics of parallel architecture to make your code faster and more efficient. To start with, you'll understand the basic concurrency concepts and explore patterns around explicit locking, lock free programming, futures & actors. Then, you'll get insights into different concurrency models and parallel algorithms and put them to practice in different scenarios to realize your application's true potential. We'll take you through multithreading design patterns, such as master, slave, leader, follower, map-reduce, and monitor, also helping you to learn hands-on coding using these patterns. Once you've grasped all of this, you'll move on to solving problems using synchronizer patterns. You'll discover the rationale for these patterns in distributed & parallel applications, followed by studying how future composition, immutability and the monadic flow help create more robust code. Toward the end of the book, you'll learn about the actor paradigm and actor patterns - the message passing concurrency paradigm. What you will learn Explore parallel architecture Get acquainted with concurrency models Internalize design themes by implementing multithreading patterns Get insights into concurrent design patterns Discover design principles behind many java threading abstractions Work with functional concurrency patterns Who this book is for This is a must-have guide for developers who want to learn patterns to build scalable and high-performing apps. It's assumed that you already have a decent level of programming knowledge.

Distributed Computing - Yehuda Afek 2013-10-04

This book constitutes the proceedings of the 27th International Symposium on Distributed Computing, DISC 2013, held in Jerusalem, Israel, in October 2013. The 27 full papers presented in this volume were carefully reviewed and selected from 142 submissions; 16 brief announcements are also included. The papers are organized in topical sections named: graph distributed algorithms; topology, leader election, and spanning trees; software transactional memory; shared memory executions; shared memory and storage; gossip and rumor; shared memory tasks and data structures; routing; radio networks and the SINR model; crypto, trust, and influence; and networking.

Encyclopedia of Parallel Computing - David Padua 2011-09-08

Containing over 300 entries in an A-Z format, the Encyclopedia of Parallel Computing provides easy, intuitive access to relevant information for professionals and researchers seeking access to any aspect within the broad field of parallel computing. Topics for this comprehensive reference were selected, written, and peer-reviewed by an international pool of distinguished researchers in the field. The Encyclopedia is broad in scope, covering machine organization, programming languages, algorithms, and applications. Within each area, concepts, designs, and specific implementations are presented. The highly-structured essays in this work comprise synonyms, a definition and discussion of the topic, bibliographies, and links to related literature. Extensive cross-references to other entries within the Encyclopedia support efficient, user-friendly searches for immediate access to useful information. Key concepts presented in the Encyclopedia of Parallel Computing include; laws and metrics; specific numerical and non-numerical algorithms; asynchronous algorithms; libraries of subroutines; benchmark suites; applications; sequential consistency and cache coherency; machine classes such as clusters, shared-memory multiprocessors, special-purpose machines and dataflow machines; specific machines such as Cray supercomputers, IBM's cell processor and Intel's multicore machines; race detection and auto parallelization; parallel programming languages, synchronization primitives, collective operations, message passing libraries, checkpointing, and operating systems. Topics covered: Speedup, Efficiency, Isoefficiency, Redundancy, Amdahl's law, Computer Architecture Concepts, Parallel Machine Designs, Benchmarks, Parallel Programming concepts & design, Algorithms, Parallel applications. This authoritative reference will be published in two

formats: print and online. The online edition features hyperlinks to cross-references and to additional significant research. Related Subjects: supercomputing, high-performance computing, distributed computing

Transactional Memory. Foundations, Algorithms, Tools, and Applications - Rachid Guerraoui 2014-12-29

The advent of multi-core architectures and cloud-computing has brought parallel programming into the mainstream of software development. Unfortunately, writing scalable parallel programs using traditional lock-based synchronization primitives is well known to be a hard, time consuming and error-prone task, mastered by only a minority of specialized programmers. Building on the familiar abstraction of atomic transactions, Transactional Memory (TM) promises to free programmers from the complexity of conventional synchronization schemes, simplifying the development and verification of concurrent programs, enhancing code reliability, and boosting productivity. Over the last decade TM has been subject to intense research on a broad range of aspects including hardware and operating systems support, language integration, as well as algorithms and theoretical foundations. On the industrial side, the major players of the software and hardware markets have been up-front in the research and development of prototypal products providing support for TM systems. This has recently led to the introduction of hardware TM implementations on mainstream commercial microprocessors and to the integration of TM support for the world's leading open source compiler. In such a vast inter-disciplinary domain, the Euro-TM COST Action (IC1001) has served as a catalyzer and a bridge for the various research communities looking at disparate, yet subtly interconnected, aspects of TM. This book emerged from the idea having Euro-TM experts compile recent results in the TM area in a single and consistent volume. Contributions have been carefully selected and revised to provide a broad coverage of several fundamental issues associated with the design and implementation of TM systems, including their theoretical underpinnings and algorithmic foundations, programming language integration and verification tools, hardware supports, distributed TM systems, self-tuning mechanisms, as well as lessons learnt from building complex TM-based applications.

Distributed Computing - Andrzej Pelc 2007-09-07

This book constitutes the refereed proceedings of the 21st International Symposium on Distributed Computing, DISC 2007, held in Lemesos, Cyprus, in September 2007. The 32 revised full papers, selected from 100 submissions, are presented together with abstracts of 3 invited papers and 9 brief announcements of ongoing works; all of them were carefully selected for inclusion in the book. The papers cover all current issues in distributed computing - theory, design, analysis, implementation, and application of distributed systems and networks - ranging from foundational and theoretical topics to algorithms and systems issues and to applications in various fields. This volume concludes with a section devoted to the 20th anniversary of the DISC conferences that took place during DISC 2006, held in Stockholm, Sweden, in September 2006

Distributed Computing and Networking - Davide Frey 2013-01-05

This book constitutes the refereed proceedings of the 14th International Conference on Distributed Computing and Networking, ICDCN 2013, held in Mumbai, India, during January 3-6, 2013. The 27 revised full papers, 5 short papers presented together with 7 poster papers were carefully reviewed and selected from 149 submissions. The papers cover topics such as distributed algorithms and concurrent data structures; integration of heterogeneous wireless and wired networks; distributed operating systems; internetworking protocols and internet applications; distributed database systems; mobile and pervasive computing, context-aware distributed systems; embedded distributed systems; next generation and converged network architectures; experiments and performance evaluation of distributed systems; overlay and peer-to-peer networks and services; fault-tolerance, reliability, and availability; home networking and services; multiprocessor and multi-core architectures and algorithms; resource management and quality of service; self-organization, self-stabilization, and autonomic computing; network security and privacy; high performance computing, grid computing, and cloud computing; energy-efficient networking and smart grids; security, cryptography, and game theory in distributed systems; sensor, PAN and ad-hoc networks; and traffic engineering, pricing, network management.

CONCUR 2012- Concurrency Theory - Maciej Koutny 2012-09-02

This book constitutes the thoroughly refereed proceedings of the 23rd International Conference on

Concurrency Theory, CONCUR 2012, held in Newcastle upon Tyne, UK, September 4-7, 2012. The 35 revised full papers presented together with 4 invited talks were carefully reviewed and selected from 97 submissions. The papers are organized in topics such as reachability analysis; qualitative and timed systems; behavioural equivalences; temporal logics; session types; abstraction; mobility and space in process algebras; stochastic systems; probabilistic systems; Petri nets and non-sequential semantics; verification; decidability.

Java Concurrency in Practice - Tim Peierls 2006-05-09

Threads are a fundamental part of the Java platform. As multicore processors become the norm, using concurrency effectively becomes essential for building high-performance applications. Java SE 5 and 6 are a huge step forward for the development of concurrent applications, with improvements to the Java Virtual Machine to support high-performance, highly scalable concurrent classes and a rich set of new concurrency building blocks. In *Java Concurrency in Practice*, the creators of these new facilities explain not only how they work and how to use them, but also the motivation and design patterns behind them. However, developing, testing, and debugging multithreaded programs can still be very difficult; it is all too easy to create concurrent programs that appear to work, but fail when it matters most: in production, under heavy load. *Java Concurrency in Practice* arms readers with both the theoretical underpinnings and concrete techniques for building reliable, scalable, maintainable concurrent applications. Rather than simply offering an inventory of concurrency APIs and mechanisms, it provides design rules, patterns, and mental models that make it easier to build concurrent programs that are both correct and performant. This book covers: Basic concepts of concurrency and thread safety Techniques for building and composing thread-safe classes Using the concurrency building blocks in `java.util.concurrent` Performance optimization dos and don'ts Testing concurrent programs Advanced topics such as atomic variables, nonblocking algorithms, and the Java Memory Model

Shared-Memory Synchronization - Michael L. Scott 2022-05-31

From driving, flying, and swimming, to digging for unknown objects in space exploration, autonomous robots take on varied shapes and sizes. In part, autonomous robots are designed to perform tasks that are too dirty, dull, or dangerous for humans. With nontrivial autonomy and volition, they may soon claim their own place in human society. These robots will be our allies as we strive for understanding our natural and man-made environments and build positive synergies around us. Although we may never perfect replication of biological capabilities in robots, we must harness the inevitable emergence of robots that synchronizes with our own capacities to live, learn, and grow. This book is a snapshot of motivations and methodologies for our collective attempts to transform our lives and enable us to cohabit with robots that work with and for us. It reviews and guides the reader to seminal and continual developments that are the foundations for successful paradigms. It attempts to demystify the abilities and limitations of robots. It is a progress report on the continuing work that will fuel future endeavors. Table of Contents: Part I: Preliminaries/Agency, Motion, and Anatomy/Behaviors / Architectures / Affect/Sensors / Manipulators/Part II: Mobility/Potential Fields/Roadmaps / Reactive Navigation / Multi-Robot Mapping: Brick and Mortar Strategy / Part III: State of the Art / Multi-Robotics Phenomena / Human-Robot Interaction / Fuzzy Control / Decision Theory and Game Theory / Part IV: On the Horizon / Applications: Macro and Micro Robots / References / Author Biography / Discussion

Nonsequential and Distributed Programming with Go - Christian Maurer 2021-01-19

Der Band bietet eine kompakte Einführung in die Nichtsequentielle Programmierung als gemeinsamen Kern von Vorlesungen über Betriebssysteme, Verteilte Systeme, Parallele Algorithmen, Echtzeitprogrammierung und Datenbanktransaktionen. Basiskonzepte zur Synchronisation und Kommunikation nebenläufiger Prozesse werden systematisch dargestellt: Schlösser, Semaphore, Monitore, lokaler und netzweiter Botschaftenaustausch. Die Algorithmen sind in der Programmiersprache Google Go formuliert, mit der viele Synchronisationskonzepte ausgedrückt werden können.

Concurrent and Distributed Computing in Java - Vijay K. Garg 2005-01-28

Concurrent and Distributed Computing in Java addresses fundamental concepts in concurrent computing with Java examples. The book consists of two parts. The first part deals with techniques for programming in shared-memory based systems. The book covers concepts in Java such as threads, synchronized methods,

waits, and notify to expose students to basic concepts for multi-threaded programming. It also includes algorithms for mutual exclusion, consensus, atomic objects, and wait-free data structures. The second part of the book deals with programming in a message-passing system. This part covers resource allocation problems, logical clocks, global property detection, leader election, message ordering, agreement algorithms, checkpointing, and message logging. Primarily a textbook for upper-level undergraduates and graduate students, this thorough treatment will also be of interest to professional programmers.

Shared-Memory Synchronization - Michael L. Scott 2013-06-01

From driving, flying, and swimming, to digging for unknown objects in space exploration, autonomous robots take on varied shapes and sizes. In part, autonomous robots are designed to perform tasks that are too dirty, dull, or dangerous for humans. With nontrivial autonomy and volition, they may soon claim their own place in human society. These robots will be our allies as we strive for understanding our natural and man-made environments and build positive synergies around us. Although we may never perfect replication of biological capabilities in robots, we must harness the inevitable emergence of robots that synchronizes with our own capacities to live, learn, and grow. This book is a snapshot of motivations and methodologies for our collective attempts to transform our lives and enable us to cohabit with robots that work with and for us. It reviews and guides the reader to seminal and continual developments that are the foundations for successful paradigms. It attempts to demystify the abilities and limitations of robots. It is a progress report on the continuing work that will fuel future endeavors. Table of Contents: Part I: Preliminaries/Agency, Motion, and Anatomy/Behaviors / Architectures / Affect/Sensors / Manipulators/Part II: Mobility/Potential Fields/Roadmaps / Reactive Navigation / Multi-Robot Mapping: Brick and Mortar Strategy / Part III: State of the Art / Multi-Robotics Phenomena / Human-Robot Interaction / Fuzzy Control / Decision Theory and Game Theory / Part IV: On the Horizon / Applications: Macro and Micro Robots / References / Author Biography / Discussion

Formal Methods for Industrial Critical Systems - Stefan Kowalewski 2010-09-09

This book constitutes the proceedings of the 15th International Workshop on Formal Methods for Industrial Critical Systems, FMICS 2010 held in Antwerp, Belgium, in September 2010 - co-located with ASE 2010, the 25th IEEE/ACM International Conference on Automated Software Engineering, The 14 papers presented were carefully reviewed and selected from 33 submissions. The aim of the FMICS workshop series is to provide a forum for researchers who are interested in the development and application of formal methods in industry. It also strives to promote research and development for the improvement of formal methods and tools for industrial applications.

Networked Systems - Parosh Aziz Abdulla 2016-09-14

This book constitutes the refereed post-proceedings of the 4th International Conference on Networked Systems, NETYS 2016, held in Marrakech, Morocco, in May 2016. The 22 full papers and 11 short papers presented together with 19 poster abstracts were carefully reviewed and selected from 121 submissions. They report on best practices and novel algorithms, results and techniques on networked systems and cover topics such as multi-core architectures, concurrent and distributed algorithms, parallel/concurrent/distributed programming, distributed databases, cloud systems, networks, security, and formal verification.

Principles of Distributed Systems - Antonio Fernández Anta 2011-12-09

This book constitutes the refereed proceedings of the 15th International Conference on Principles of Distributed Systems, OPODIS 2011, held in Toulouse, France, in December 2011. The 26 revised papers presented in this volume were carefully reviewed and selected from 96 submissions. They represent the current state of the art of the research in the field of the design, analysis and development of distributed and real-time systems.

Foundations of Multithreaded, Parallel, and Distributed Programming - Gregory R. Andrews 2000

Foundations of Multithreaded, Parallel, and Distributed Programming covers, and then applies, the core concepts and techniques needed for an introductory course in this subject. Its emphasis is on the practice and application of parallel systems, using real-world examples throughout. Greg Andrews teaches the fundamental concepts of multithreaded, parallel and distributed computing and relates them to the implementation and performance processes. He presents the appropriate breadth of topics and supports

these discussions with an emphasis on performance. Features Emphasizes how to solve problems, with correctness the primary concern and performance an important, but secondary, concern Includes a number of case studies which cover such topics as pthreads, MPI, and OpenMP libraries, as well as programming languages like Java, Ada, high performance Fortran, Linda, Occam, and SR Provides examples using Java syntax and discusses how Java deals with monitors, sockets, and remote method invocation Covers current programming techniques such as semaphores, locks, barriers, monitors, message passing, and remote invocation Concrete examples are executed with complete programs, both shared and distributed Sample applications include scientific computing and distributed systems 0201357526B04062001

Synchronization Algorithms and Concurrent Programming - Gadi Taubenfeld 2006

The first textbook that focuses purely on Synchronization - a fundamental challenge in Computer Science that is fast becoming a major performance and design issue for concurrent programming on modern architectures, and for the design of distributed systems.

The Art of Concurrency - Clay Breshears 2009-05-07

If you're looking to take full advantage of multi-core processors with concurrent programming, this practical book provides the knowledge and hands-on experience you need. The Art of Concurrency is one of the few resources to focus on implementing algorithms in the shared-memory model of multi-core processors, rather than just theoretical models or distributed-memory architectures. The book provides detailed explanations and usable samples to help you transform algorithms from serial to parallel code, along with advice and analysis for avoiding mistakes that programmers typically make when first attempting these computations. Written by an Intel engineer with over two decades of parallel and concurrent programming experience, this book will help you: Understand parallelism and concurrency Explore differences between programming for shared-memory and distributed-memory Learn guidelines for designing multithreaded applications, including testing and tuning Discover how to make best use of different threading libraries, including Windows threads, POSIX threads, OpenMP, and Intel Threading Building Blocks Explore how to implement concurrent algorithms that involve sorting, searching, graphs, and other practical computations The Art of Concurrency shows you how to keep algorithms scalable to take advantage of new processors with even more cores. For developing parallel code algorithms for concurrent programming, this book is a must.

Algorithms for Concurrent Systems - Rachid Guerraoui 2018

On Concurrent Programming - Fred B. Schneider 2012-12-06

Here, one of the leading figures in the field provides a comprehensive survey of the subject, beginning with propositional logic and concluding with concurrent programming. It is based on graduate courses taught at Cornell University and is designed for use as a graduate text. Professor Schneider emphasises the use of formal methods and assertional reasoning using notation and paradigms drawn from programming to drive the exposition, while exercises at the end of each chapter extend and illustrate the main themes covered. As a result, all those interested in studying concurrent computing will find this an invaluable approach to the subject.

Algorithms and Architectures for Parallel Processing - Jesus Carretero 2016-11-24

This book constitutes the refereed proceedings of the 16th International Conference on Algorithms and Architectures for Parallel Processing, ICA3PP 2016, held in Granada, Spain, in December 2016. The 30 full papers and 22 short papers presented were carefully reviewed and selected from 117 submissions. They cover many dimensions of parallel algorithms and architectures, encompassing fundamental theoretical approaches, practical experimental projects, and commercial components and systems trying to push beyond the limits of existing technologies, including experimental efforts, innovative systems, and investigations that identify weaknesses in existing parallel processing technology.

Programming Erlang - Joe Armstrong 2013-09-23

A multi-user game, web site, cloud application, or networked database can have thousands of users all interacting at the same time. You need a powerful, industrial-strength tool to handle the really hard problems inherent in parallel, concurrent environments. You need Erlang. In this second edition of the bestselling Programming Erlang, you'll learn how to write parallel programs that scale effortlessly on

multicore systems. Using Erlang, you'll be surprised at how easy it becomes to deal with parallel problems, and how much faster and more efficiently your programs run. That's because Erlang uses sets of parallel processes-not a single sequential process, as found in most programming languages. Joe Armstrong, creator of Erlang, introduces this powerful language in small steps, giving you a complete overview of Erlang and how to use it in common scenarios. You'll start with sequential programming, move to parallel programming and handling errors in parallel programs, and learn to work confidently with distributed programming and the standard Erlang/Open Telecom Platform (OTP) frameworks. You need no previous knowledge of functional or parallel programming. The chapters are packed with hands-on, real-world tutorial examples and insider tips and advice, and finish with exercises for both beginning and advanced users. The second edition has been extensively rewritten. New to this edition are seven chapters covering the latest Erlang features: maps, the type system and the Dialyzer, WebSockets, programming idioms, and a new stand-alone execution environment. You'll write programs that dynamically detect and correct errors, and that can be upgraded without stopping the system. There's also coverage of rebar (the de facto Erlang build system), and information on how to share and use Erlang projects on github, illustrated with examples from cowboy and bitcask. Erlang will change your view of the world, and of how you program. What You Need The Erlang/OTP system. Download it from erlang.org.

Concurrent Programming: Algorithms, Principles, and Foundations - Michel Raynal 2012-12-30

This book is devoted to the most difficult part of concurrent programming, namely synchronization concepts, techniques and principles when the cooperating entities are asynchronous, communicate through a shared memory, and may experience failures. Synchronization is no longer a set of tricks but, due to research results in recent decades, it relies today on sane scientific foundations as explained in this book. In this book the author explains synchronization and the implementation of concurrent objects, presenting in a uniform and comprehensive way the major theoretical and practical results of the past 30 years. Among the key features of the book are a new look at lock-based synchronization (mutual exclusion, semaphores, monitors, path expressions); an introduction to the atomicity consistency criterion and its properties and a specific chapter on transactional memory; an introduction to mutex-freedom and associated progress conditions such as obstruction-freedom and wait-freedom; a presentation of Lamport's hierarchy of safe, regular and atomic registers and associated wait-free constructions; a description of numerous wait-free constructions of concurrent objects (queues, stacks, weak counters, snapshot objects, renaming objects, etc.); a presentation of the computability power of concurrent objects including the notions of universal construction, consensus number and the associated Herlihy's hierarchy; and a survey of failure detector-based constructions of consensus objects. The book is suitable for advanced undergraduate students and graduate students in computer science or computer engineering, graduate students in mathematics interested in the foundations of process synchronization, and practitioners and engineers who need to produce correct concurrent software. The reader should have a basic knowledge of algorithms and operating systems.

Euro-Par 2015: Parallel Processing Workshops - Sascha Hunold 2015-12-17

This book constitutes the thoroughly refereed post-conference proceedings of 12 workshops held at the 21st International Conference on Parallel and Distributed Computing, Euro-Par 2015, in Vienna, Austria, in August 2015. The 67 revised full papers presented were carefully reviewed and selected from 121 submissions. The volume includes papers from the following workshops: BigDataCloud: 4th Workshop on Big Data Management in Clouds - Euro-EDUPAR: First European Workshop on Parallel and Distributed Computing Education for Undergraduate Students - Hetero Par: 13th International Workshop on Algorithms, Models and Tools for Parallel Computing on Heterogeneous Platforms - LSDVE: Third Workshop on Large Scale Distributed Virtual Environments - OMHI: 4th International Workshop on On-chip Memory Hierarchies and Interconnects - PADAPS: Third Workshop on Parallel and Distributed Agent-Based Simulations - PELGA: Workshop on Performance Engineering for Large-Scale Graph Analytics - REPPAR: Second International Workshop on Reproducibility in Parallel Computing - Resilience: 8th Workshop on Resiliency in High Performance Computing in Clusters, Clouds, and Grids - ROME: Third Workshop on Runtime and Operating Systems for the Many Core Era - UCHPC: 8th Workshop on UnConventional High Performance Computing - and VHPC: 10th Workshop on Virtualization in High-Performance Cloud

Computing.

Topics in Parallel and Distributed Computing - Sushil K Prasad 2015-09-16

Topics in Parallel and Distributed Computing provides resources and guidance for those learning PDC as well as those teaching students new to the discipline. The pervasiveness of computing devices containing multicore CPUs and GPUs, including home and office PCs, laptops, and mobile devices, is making even common users dependent on parallel processing. Certainly, it is no longer sufficient for even basic programmers to acquire only the traditional sequential programming skills. The preceding trends point to the need for imparting a broad-based skill set in PDC technology. However, the rapid changes in computing hardware platforms and devices, languages, supporting programming environments, and research advances, poses a challenge both for newcomers and seasoned computer scientists. This edited collection has been developed over the past several years in conjunction with the IEEE technical committee on parallel processing (TCPP), which held several workshops and discussions on learning parallel computing and integrating parallel concepts into courses throughout computer science curricula. Contributed and developed by the leading minds in parallel computing research and instruction Provides resources and guidance for those learning PDC as well as those teaching students new to the discipline Succinctly addresses a range of parallel and distributed computing topics Pedagogically designed to ensure understanding by experienced engineers and newcomers Developed over the past several years in conjunction with the IEEE technical committee on parallel processing (TCPP), which held several workshops and discussions on learning parallel computing and integrating parallel concepts *The Art of Multiprocessor Programming, Revised Reprint* - Maurice Herlihy 2012-06-25 Revised and updated with improvements conceived in parallel programming courses, *The Art of Multiprocessor Programming* is an authoritative guide to multicore programming. It introduces a higher level set of software development skills than that needed for efficient single-core programming. This book provides comprehensive coverage of the new principles, algorithms, and tools necessary for effective multiprocessor programming. Students and professionals alike will benefit from thorough coverage of key multiprocessor programming issues. This revised edition incorporates much-demanded updates throughout the book, based on feedback and corrections reported from classrooms since 2008 Learn the fundamentals of programming multiple threads accessing shared memory Explore mainstream concurrent data structures and the key elements of their design, as well as synchronization techniques from simple locks to transactional memory systems Visit the companion site and download source code, example Java programs, and materials to support and enhance the learning experience

The Origin of Concurrent Programming - Per Brinch Hansen 2013-06-29

An essential reader containing 19 important papers on the invention and early development of concurrent programming and its relevance to computer science and computer engineering. All of them are written by the pioneers in concurrent programming, including Brinch Hansen himself, and have introductions added that summarize the papers and put them in perspective. The editor provides an overview chapter and neatly places all developments in perspective with chapter introductions and expository apparatus. Essential resource for graduates, professionals, and researchers in CS with an interest in concurrent programming principles. A familiarity with operating system principles is assumed.

Distributed Computing Pearls - Gadi Taubenfeld 2022-05-31

Computers and computer networks are one of the most incredible inventions of the 20th century, having an ever-expanding role in our daily lives by enabling complex human activities in areas such as entertainment, education, and commerce. One of the most challenging problems in computer science for the 21st century is to improve the design of distributed systems where computing devices have to work together as a team to achieve common goals. In this book, I have tried to gently introduce the general reader to some of the most fundamental issues and classical results of computer science underlying the design of algorithms for distributed systems, so that the reader can get a feel of the nature of this exciting and fascinating field called distributed computing. The book will appeal to the educated layperson and requires no computer-related background. I strongly suspect that also most computer-knowledgeable readers will be able to learn something new.

Concurrency in Go - Katherine Cox-Buday 2017-07-19

Concurrency can be notoriously difficult to get right, but fortunately, the Go open source programming language makes working with concurrency tractable and even easy. If you're a developer familiar with Go, this practical book demonstrates best practices and patterns to help you incorporate concurrency into your systems. Author Katherine Cox-Buday takes you step-by-step through the process. You'll understand how Go chooses to model concurrency, what issues arise from this model, and how you can compose primitives within this model to solve problems. Learn the skills and tooling you need to confidently write and implement concurrent systems of any size. Understand how Go addresses fundamental problems that make concurrency difficult to do correctly. Learn the key differences between concurrency and parallelism. Dig into the syntax of Go's memory synchronization primitives. Form patterns with these primitives to write maintainable concurrent code. Compose patterns into a series of practices that enable you to write large, distributed systems that scale. Learn the sophistication behind goroutines and how Go's runtime stitches everything together.

Parameterized Verification of Synchronized Concurrent Programs - Zeinab Ganjei 2021-03-19

There is currently an increasing demand for concurrent programs. Checking the correctness of concurrent programs is a complex task due to the interleavings of processes. Sometimes, violation of the correctness properties in such systems causes human or resource losses; therefore, it is crucial to check the correctness of such systems. Two main approaches to software analysis are testing and formal verification. Testing can help discover many bugs at a low cost. However, it cannot prove the correctness of a program. Formal verification, on the other hand, is the approach for proving program correctness. Model checking is a formal verification technique that is suitable for concurrent programs. It aims to automatically establish the correctness (expressed in terms of temporal properties) of a program through an exhaustive search of the behavior of the system. Model checking was initially introduced for the purpose of verifying finite-state concurrent programs, and extending it to infinite-state systems is an active research area. In this thesis, we

focus on the formal verification of parameterized systems. That is, systems in which the number of executing processes is not bounded a priori. We provide fully-automatic and parameterized model checking techniques for establishing the correctness of safety properties for certain classes of concurrent programs. We provide an open-source prototype for every technique and present our experimental results on several benchmarks. First, we address the problem of automatically checking safety properties for bounded as well as parameterized phaser programs. Phaser programs are concurrent programs that make use of the complex synchronization construct of Habanero Java phasers. For the bounded case, we establish the decidability of checking the violation of program assertions and the undecidability of checking deadlock-freedom. For the parameterized case, we study different formulations of the verification problem and propose an exact procedure that is guaranteed to terminate for some reachability problems even in the presence of unbounded phases and arbitrarily many spawned processes. Second, we propose an approach for automatic verification of parameterized concurrent programs in which shared variables are manipulated by atomic transitions to count and synchronize the spawned processes. For this purpose, we introduce counting predicates that related counters that refer to the number of processes satisfying some given properties to the variables that are directly manipulated by the concurrent processes. We then combine existing works on the counter, predicate, and constrained monotonic abstraction and build a nested counterexample-based refinement scheme to establish correctness. Third, we introduce Lazy Constrained Monotonic Abstraction for more efficient exploration of well-structured abstractions of infinite-state non-monotonic systems. We propose several heuristics and assess the efficiency of the proposed technique by extensive experiments using our open-source prototype. Lastly, we propose a sound but (in general) incomplete procedure for automatic verification of safety properties for a class of fault-tolerant distributed protocols described in the Heard-Of (HO for short) model. The HO model is a popular model for describing distributed protocols. We propose a verification procedure that is guaranteed to terminate even for unbounded number of the processes that execute the distributed protocol.