

Distributed Algorithms An Intuitive Approach

A friendly introduction to the most useful algorithms written in simple, intuitive English. The revised and updated second edition of *Essential Algorithms*, offers an accessible introduction to computer algorithms. The book contains a description of important classical algorithms and explains when each is appropriate. The author shows how to analyze algorithms in order to understand their behavior and teaches techniques that can be used to create new algorithms to meet future needs. The text includes useful algorithms such as: methods for manipulating common data structures, advanced data structures, network algorithms, and numerical algorithms. It also offers a variety of general problem-solving techniques. In addition to describing algorithms and approaches, the author offers details on how to analyze the performance of algorithms. The book is filled with exercises that can be used to explore ways to modify the algorithms in order to apply them to new situations. This updated edition of *Essential Algorithms*: Contains explanations of algorithms in simple terms, rather than complicated math. Steps through powerful algorithms that can be used to solve difficult programming problems. Helps prepare for programming job interviews that typically include algorithmic questions. Offers methods that can be applied to any programming language. Includes exercises and solutions useful to both professionals and students. Provides code examples updated and written in Python and C#. *Essential Algorithms* has been updated and revised and offers professionals and students a hands-on guide to analyzing algorithms as well as the techniques and applications. The book also includes a collection of questions that may appear in a job interview. The book's website will include reference implementations in Python and C# (which can be easily applied to Java and C++).

Distributed computing is at the heart of many applications. It arises as soon as one has to solve a problem in terms of entities -- such as processes, peers, processors, nodes, or agents -- that individually have only a partial knowledge of the many input parameters associated with the problem. In particular each entity cooperating towards the common goal cannot have an instantaneous knowledge of the current state of the other entities. Whereas parallel computing is mainly concerned with 'efficiency', and real-time computing is mainly concerned with 'on-time computing', distributed computing is mainly concerned with 'mastering uncertainty' created by issues such as the multiplicity of control flows, asynchronous communication, unstable behaviors, mobility, and dynamicity. While some distributed algorithms consist of a few lines only, their behavior can be difficult to understand and their properties hard to state and prove. The aim of this book is to present in a comprehensive way the basic notions, concepts, and algorithms of distributed computing when the distributed entities cooperate by sending and receiving messages on top of an asynchronous network. The book is composed of seventeen chapters structured into six parts: distributed graph algorithms, in particular what makes them different from sequential or parallel algorithms; logical time and global states, the core of the book; mutual exclusion and resource allocation; high-level communication abstractions; distributed detection of properties; and distributed shared memory. The author establishes clear objectives per chapter and the content is supported throughout with illustrative examples, summaries, exercises, and annotated bibliographies. This book constitutes an introduction to distributed computing and is suitable for advanced undergraduate students or graduate students in computer science and computer engineering, graduate students in mathematics interested in distributed computing, and practitioners and engineers involved in the design and implementation of distributed applications. The reader should have a basic knowledge of algorithms and operating systems.

Designing distributed computing systems is a complex process requiring a solid understanding of the design problems and the theoretical and practical aspects of their solutions. This comprehensive textbook covers the fundamental principles and models underlying the theory, algorithms and systems aspects of distributed computing. Broad and detailed coverage of the theory is balanced with practical systems-related issues such as mutual exclusion, deadlock detection, authentication, and failure recovery. Algorithms are carefully selected, lucidly presented, and described without complex proofs. Simple explanations and illustrations are used to elucidate the algorithms. Important emerging topics such as peer-to-peer networks and network security are also considered. With vital algorithms, numerous illustrations, examples and homework problems, this textbook is suitable for advanced undergraduate and graduate students of electrical and computer engineering and computer science. Practitioners in data networking and sensor networks will also find this a valuable resource. Additional resources are available online at www.cambridge.org/9780521876346.

Distributed Algorithms An Intuitive Approach MIT Press

Introduces machine learning and its algorithmic paradigms, explaining the principles behind automated learning approaches and the considerations underlying their usage.

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.

A comprehensive and rigorous introduction for graduate students and researchers, with applications in sequential decision-making problems.

EI-Rewini and Lewis were among the first researchers to recognize the problem of resource allocation (scheduling) inherent in parallel and distributed programs. Here they offer a clear explanation of the problems, methods to solve the problems under a variety of conditions, and an evaluation of the "goodness" of the solutions.

Over the past several years, cooperative control and optimization have increasingly played a larger and more important role in many aspects

of military sciences, biology, communications, robotics, and decision making. At the same time, cooperative systems are notoriously difficult to model, analyze, and solve — while intuitively understood, they are not axiomatically defined in any commonly accepted manner. The works in this volume provide outstanding insights into this very complex area of research. They are the result of invited papers and selected presentations at the Fourth Annual Conference on Cooperative Control and Optimization held in Destin, Florida, November 2003. This book has been selected for coverage in: • Index to Scientific & Technical Proceedings® (ISTP® / ISI Proceedings) • Index to Scientific & Technical Proceedings (ISTP CDROM version / ISI Proceedings) • CC Proceedings — Engineering & Physical Sciences Contents: Mesh Stability in Formation of Distributed Systems (C Ashokkumar et al.) On the Performance of Heuristics for Broadcast Scheduling (C Commander et al.) Coupled Detection Rates: An Introduction (D Jeffcoat) Decentralized Receding Horizon Control for Multiple UAVs (Y Kuwata & J How) Multitarget Sensor Management of Dispersed Mobile Sensors (R Mahler) K-Means Clustering Using Entropy Minimization (A Okafor & P Pardalos) Possibility Reasoning and the Cooperative Prisoner's Dilemma (H Pfister & J Walls) Coordinating Very Large Groups of Wide Area Search Munitions (P Scerri et al.) A Vehicle Following Methodology for UAV Formations (S Spry et al.) Decentralized Optimization via Nash Bargaining (S Waslander et al.) and other papers Readership: Graduate students and researchers in optimization and control, computer science and engineering. Keywords: Cooperative Systems, Cooperative Control; Optimization; Cooperative Networks Key Features: 25 chapters of creative approaches to modeling, analysis, and synthesis of cooperative systems Research results from top researchers in the field of cooperative systems Exciting insights to cooperative systems which have increasingly played a larger and more important role in many aspects of military sciences, biology, communications, robotics, and decision making

An introduction to algorithms for readers with no background in advanced mathematics or computer science, emphasizing examples and real-world problems. Algorithms are what we do in order not to have to do something. Algorithms consist of instructions to carry out tasks—usually dull, repetitive ones. Starting from simple building blocks, computer algorithms enable machines to recognize and produce speech, translate texts, categorize and summarize documents, describe images, and predict the weather. A task that would take hours can be completed in virtually no time by using a few lines of code in a modern scripting program. This book offers an introduction to algorithms through the real-world problems they solve. The algorithms are presented in pseudocode and can readily be implemented in a computer language. The book presents algorithms simply and accessibly, without overwhelming readers or insulting their intelligence. Readers should be comfortable with mathematical fundamentals and have a basic understanding of how computers work; all other necessary concepts are explained in the text. After presenting background in pseudocode conventions, basic terminology, and data structures, chapters cover compression, cryptography, graphs, searching and sorting, hashing, classification, strings, and chance. Each chapter describes real problems and then presents algorithms to solve them. Examples illustrate the wide range of applications, including shortest paths as a solution to paragraph line breaks, strongest paths in elections systems, hashes for song recognition, voting power Monte Carlo methods, and entropy for machine learning. Real-World Algorithms can be used by students in disciplines from economics to applied sciences. Computer science majors can read it before using a more technical text.

Planning algorithms are impacting technical disciplines and industries around the world, including robotics, computer-aided design, manufacturing, computer graphics, aerospace applications, drug design, and protein folding. This coherent and comprehensive book unifies material from several sources, including robotics, control theory, artificial intelligence, and algorithms. The treatment is centered on robot motion planning, but integrates material on planning in discrete spaces. A major part of the book is devoted to planning under uncertainty, including decision theory, Markov decision processes, and information spaces, which are the 'configuration spaces' of all sensor-based planning problems. The last part of the book delves into planning under differential constraints that arise when automating the motions of virtually any mechanical system. This text and reference is intended for students, engineers, and researchers in robotics, artificial intelligence, and control theory as well as computer graphics, algorithms, and computational biology.

Distributed Systems: An Algorithmic Approach, Second Edition provides a balanced and straightforward treatment of the underlying theory and practical applications of distributed computing. As in the previous version, the language is kept as unobscured as possible—clarity is given priority over mathematical formalism. This easily digestible text: Features significant updates that mirror the phenomenal growth of distributed systems Explores new topics related to peer-to-peer and social networks Includes fresh exercises, examples, and case studies Supplying a solid understanding of the key principles of distributed computing and their relationship to real-world applications, Distributed Systems: An Algorithmic Approach, Second Edition makes both an ideal textbook and a handy professional reference.

This text is based on a simple and fully reactive computational model that allows for intuitive comprehension and logical designs. The principles and techniques presented can be applied to any distributed computing environment (e.g., distributed systems, communication networks, data networks, grid networks, internet, etc.). The text provides a wealth of unique material for learning how to design algorithms and protocols perform tasks efficiently in a distributed computing environment.

About the book: The Internet is a distributed system, but so are wireless communication, cloud or parallel computing, multi-core systems, mobile networks. Also an ant colony, a brain, or even the human society can be modeled as distributed systems. In this book we will be highlighting common themes and techniques. In particular, we study some of the fundamental issues underlying the design of distributed systems, for example, communication, coordination, fault-tolerance, locality, parallelism, symmetry breaking, synchronization, and uncertainty. About the author: Roger Wattenhofer is a professor at ETH Zurich. Before joining ETH Zurich, he was at Brown University and Microsoft Research. His research interests include fault-tolerant distributed systems, efficient network algorithms, and cryptocurrencies such as Bitcoin. He has published more than 300 scientific articles. In 2017, he published the book Blockchain Science.

In modern computing a program is usually distributed among several processes. The fundamental challenge when developing reliable and secure distributed programs is to support the cooperation of processes required to execute a common task, even when some of these processes fail. Failures may range from crashes to adversarial attacks by malicious processes. Cachin, Guerraoui, and Rodrigues present an introductory description of fundamental distributed programming abstractions together with algorithms to implement them in distributed systems, where processes are subject to crashes and malicious attacks. The authors follow an incremental approach by first introducing basic abstractions in simple distributed environments, before moving to more sophisticated abstractions and more challenging environments. Each core chapter is devoted to one topic, covering reliable broadcast, shared memory, consensus, and extensions of consensus. For every topic, many exercises and their solutions enhance the understanding This book represents the second edition of "Introduction to Reliable Distributed Programming". Its scope has been extended to include security against malicious actions by non-cooperating processes. This important domain has become widely known under the name "Byzantine fault-tolerance".

Surveys the theory and history of the alternating direction method of multipliers, and discusses its applications to a wide variety of statistical and machine learning problems of recent interest, including the lasso, sparse logistic regression, basis pursuit, covariance selection, support vector machines, and many others.

Distributed Computing Through Combinatorial Topology describes techniques for analyzing distributed algorithms based on award winning combinatorial topology research. The authors present a solid theoretical foundation relevant to many real systems reliant on parallelism with unpredictable delays, such as multicore microprocessors, wireless networks, distributed systems, and Internet protocols. Today, a new student or researcher must assemble a collection of scattered conference publications, which are typically terse and commonly use different notations and terminologies. This book provides a self-contained explanation of the mathematics

to readers with computer science backgrounds, as well as explaining computer science concepts to readers with backgrounds in applied mathematics. The first section presents mathematical notions and models, including message passing and shared-memory systems, failures, and timing models. The next section presents core concepts in two chapters each: first, proving a simple result that lends itself to examples and pictures that will build up readers' intuition; then generalizing the concept to prove a more sophisticated result. The overall result weaves together and develops the basic concepts of the field, presenting them in a gradual and intuitively appealing way. The book's final section discusses advanced topics typically found in a graduate-level course for those who wish to explore further. Named a 2013 Notable Computer Book for Computing Methodologies by Computing Reviews Gathers knowledge otherwise spread across research and conference papers using consistent notations and a standard approach to facilitate understanding Presents unique insights applicable to multiple computing fields, including multicore microprocessors, wireless networks, distributed systems, and Internet protocols Synthesizes and distills material into a simple, unified presentation with examples, illustrations, and exercises

A comprehensive guide to distributed algorithms that emphasizes examples and exercises rather than mathematical argumentation. This book offers students and researchers a guide to distributed algorithms that emphasizes examples and exercises rather than the intricacies of mathematical models. It avoids mathematical argumentation, often a stumbling block for students, teaching algorithmic thought rather than proofs and logic. This approach allows the student to learn a large number of algorithms within a relatively short span of time. Algorithms are explained through brief, informal descriptions, illuminating examples, and practical exercises. The examples and exercises allow readers to understand algorithms intuitively and from different perspectives. Proof sketches, arguing the correctness of an algorithm or explaining the idea behind fundamental results, are also included. An appendix offers pseudocode descriptions of many algorithms. Distributed algorithms are performed by a collection of computers that send messages to each other or by multiple software threads that use the same shared memory. The algorithms presented in the book are for the most part "classics," selected because they shed light on the algorithmic design of distributed systems or on key issues in distributed computing and concurrent programming. Distributed Algorithms can be used in courses for upper-level undergraduates or graduate students in computer science, or as a reference for researchers in the field.

Table of contents

This textbook guides students through algebraic specification and verification of distributed systems, and some of the most prominent formal verification techniques. The author employs μ CRL as the vehicle, a language developed to combine process algebra and abstract data types. The book evolved from introductory courses on protocol verification taught to undergraduate and graduate students of computer science, and the text is supported throughout with examples and exercises. Full solutions are provided in an appendix, while exercise sheets, lab exercises, example specifications and lecturer slides are available on the author's website.

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.

The new edition of a guide to distributed algorithms that emphasizes examples and exercises rather than the intricacies of mathematical models. This book offers students and researchers a guide to distributed algorithms that emphasizes examples and exercises rather than the intricacies of mathematical models. It avoids mathematical argumentation, often a stumbling block for students, teaching algorithmic thought rather than proofs and logic. This approach allows the student to learn a large number of algorithms within a relatively short span of time. Algorithms are explained through brief, informal descriptions, illuminating examples, and practical exercises. The examples and exercises allow readers to understand algorithms intuitively and from different perspectives. Proof sketches, arguing the correctness of an algorithm or explaining the idea behind fundamental results, are also included. The algorithms presented in the book are for the most part "classics," selected because they shed light on the algorithmic design of distributed systems or on key issues in distributed computing and concurrent programming. This second edition has been substantially revised. A new chapter on distributed transaction offers up-to-date treatment of database transactions and the important evolving area of transactional memory. A new chapter on security discusses two exciting new topics: blockchains and quantum cryptography. Sections have been added that cover such subjects as rollback recovery, fault-tolerant termination detection, and consensus for shared memory. An appendix offers pseudocode descriptions of many algorithms. Solutions and slides are available for instructors. Distributed Algorithms can be used in courses for upper-level undergraduates or graduate students in computer science, or as a reference for researchers in the field.

"This textbook is designed to accompany a one- or two-semester course for advanced undergraduates or beginning graduate students in computer science and applied mathematics. - It gives an excellent introduction to the probabilistic techniques and paradigms used in the development of probabilistic algorithms and analyses. - It assumes only an elementary background in discrete mathematics and gives a rigorous yet accessible treatment of the material, with numerous examples and applications."--Jacket.

Why a book about logs? That's easy: the humble log is an abstraction that lies at the heart of many systems, from NoSQL databases to cryptocurrencies. Even though most engineers don't think much about them, this short book shows you why logs are worthy of your attention. Based on his popular blog posts, LinkedIn principal engineer Jay Kreps shows you how logs work in distributed systems, and then delivers practical applications of these concepts in a variety of common uses—data integration, enterprise architecture, real-time stream processing, data system design, and abstract computing models. Go ahead and take the plunge with logs; you're going love them. Learn how logs are used for programmatic access in databases and distributed systems Discover solutions to the huge data integration problem when more data of more varieties meet more systems Understand why logs are at the heart of real-time stream processing Learn the role of a log in the internals of online data systems Explore how Jay Kreps applies these ideas to his own work on data infrastructure systems at LinkedIn

This second edition of Distributed Systems, Principles & Paradigms, covers the principles, advanced concepts, and technologies of

distributed systems in detail, including: communication, replication, fault tolerance, and security. Intended for use in a senior/graduate level distributed systems course or by professionals, this text systematically shows how distributed systems are designed and implemented in real systems.

Numerical Algorithms: Methods for Computer Vision, Machine Learning, and Graphics presents a new approach to numerical analysis for modern computer scientists. Using examples from a broad base of computational tasks, including data processing, computational photography, and animation, the textbook introduces numerical modeling and algorithmic design

Principles of Concurrent and Distributed Programming provides an introduction to concurrent programming focusing on general principles and not on specific systems. Software today is inherently concurrent or distributed – from event-based GUI designs to operating and real-time systems to Internet applications. The new edition of this classic introduction to concurrency has been completely revised in view of the growing importance of concurrency constructs embedded in programming languages and of formal methods such as model checking that are widely used in industry.

The objective of our monograph is to cover the developments on the theoretical foundations of distributed symmetry breaking in the message-passing model. We hope that our monograph will stimulate further progress in this exciting area.

This book presents the most important fault-tolerant distributed programming abstractions and their associated distributed algorithms, in particular in terms of reliable communication and agreement, which lie at the heart of nearly all distributed applications. These programming abstractions, distributed objects or services, allow software designers and programmers to cope with asynchrony and the most important types of failures such as process crashes, message losses, and malicious behaviors of computing entities, widely known under the term "Byzantine fault-tolerance". The author introduces these notions in an incremental manner, starting from a clear specification, followed by algorithms which are first described intuitively and then proved correct. The book also presents impossibility results in classic distributed computing models, along with strategies, mainly failure detectors and randomization, that allow us to enrich these models. In this sense, the book constitutes an introduction to the science of distributed computing, with applications in all domains of distributed systems, such as cloud computing and blockchains. Each chapter comes with exercises and bibliographic notes to help the reader approach, understand, and master the fascinating field of fault-tolerant distributed computing.

The new edition of this bestselling title on Distributed Systems has been thoroughly revised throughout to reflect the state of the art in this rapidly developing field. It emphasizes the principles used in the design and construction of distributed computer systems based on networks of workstations and server computers.

Introduction : distributed systems - The model - Communication protocols - Routing algorithms - Deadlock-free packet switching - Wave and traversal algorithms - Election algorithms - Termination detection - Anonymous networks - Snapshots - Sense of direction and orientation - Synchrony in networks - Fault tolerance in distributed systems - Fault tolerance in asynchronous systems - Fault tolerance in synchronous systems - Failure detection - Stabilization.

The fundamental mathematical tools needed to understand machine learning include linear algebra, analytic geometry, matrix decompositions, vector calculus, optimization, probability and statistics. These topics are traditionally taught in disparate courses, making it hard for data science or computer science students, or professionals, to efficiently learn the mathematics. This self-contained textbook bridges the gap between mathematical and machine learning texts, introducing the mathematical concepts with a minimum of prerequisites. It uses these concepts to derive four central machine learning methods: linear regression, principal component analysis, Gaussian mixture models and support vector machines. For students and others with a mathematical background, these derivations provide a starting point to machine learning texts. For those learning the mathematics for the first time, the methods help build intuition and practical experience with applying mathematical concepts. Every chapter includes worked examples and exercises to test understanding. Programming tutorials are offered on the book's web site.

In Distributed Algorithms, Nancy Lynch provides a blueprint for designing, implementing, and analyzing distributed algorithms. She directs her book at a wide audience, including students, programmers, system designers, and researchers. Distributed Algorithms contains the most significant algorithms and impossibility results in the area, all in a simple automata-theoretic setting. The algorithms are proved correct, and their complexity is analyzed according to precisely defined complexity measures. The problems covered include resource allocation, communication, consensus among distributed processes, data consistency, deadlock detection, leader election, global snapshots, and many others. The material is organized according to the system model—first by the timing model and then by the interprocess communication mechanism. The material on system models is isolated in separate chapters for easy reference. The presentation is completely rigorous, yet is intuitive enough for immediate comprehension. This book familiarizes readers with important problems, algorithms, and impossibility results in the area: readers can then recognize the problems when they arise in practice, apply the algorithms to solve them, and use the impossibility results to determine whether problems are unsolvable. The book also provides readers with the basic mathematical tools for designing new algorithms and proving new impossibility results. In addition, it teaches readers how to reason carefully about distributed algorithms—to model them formally, devise precise specifications for their required behavior, prove their correctness, and evaluate their performance with realistic measures.

Distributed Computing is rapidly becoming the principal computing paradigm in diverse areas of computing, communication, and control. Processor clusters, local and wide area networks, and the information highway evolved a new kind of problems which can be solved with distributed algorithms. In this textbook a variety of distributed algorithms are presented independently of particular programming languages or hardware, using the graphically suggestive technique of Petri nets which is both easy to comprehend intuitively and formally rigorous. By means of temporal logic the author provides surprisingly simple yet powerful correctness proofs for the algorithms. The scope of the book ranges from distributed control and synchronization of two sites up to algorithms on any kind of networks. Numerous examples show that description and analysis of distributed algorithms in this framework are intuitive and technically transparent.

A friendly and accessible introduction to the most useful algorithms Computer algorithms are the basic recipes for programming. Professional programmers need to know how to use algorithms to solve difficult programming problems. Written in simple, intuitive English, this book describes how and when to use the most practical classic algorithms, and even how to create new algorithms to meet future needs. The book also includes a collection of questions that can help readers prepare for a programming job interview. Reveals methods for manipulating common data structures such as arrays, linked lists, trees, and networks Addresses advanced data structures such as heaps, 2-3 trees, B-trees Addresses general problem-solving techniques such as branch and bound, divide and conquer, recursion, backtracking, heuristics, and more Reviews sorting and searching, network algorithms, and numerical algorithms Includes general problem-solving techniques such as brute force and exhaustive search, divide and conquer, backtracking, recursion, branch and bound, and more In addition, Essential Algorithms features a companion website that includes full instructor materials to support training or higher ed adoptions.

