Introduction To Distributed Algorithms

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.

This book constitutes the refereed proceedings of the 5th International Conference on Parallel and Distributed Computing, Applications and Technologies; PDCAT 2004, held in Singapore in December 2004. The 173 papers presented were carefully reviewed and selected from 242 submissions. The papers focus on parallel and distributed computing from the perspectives of algorithms, networking and architecture, software systems and technologies, and applications. Besides classical topics from high performance computing, major recent developments are addressed, such as molecular computing, date mining, knowledge discovery, optical networks, secure computing and communications, wireless networks, mobile computing, component-based systems, Internet computing, and Web Technologies.

Provides an introduction to OCCAM and distributed algorithms and uses ELECTOR to simulate a class of distributed algorithms called election algorithms (three are examined). Includes a user's manual for ELECTOR for the benefit of others.

This concise text is designed to present the recent advances in parallel and distributed architectures and algorithms within an integrated framework. Beginning with an introduction to the basic concepts, the book goes on discussing the basic methods of parallelism exploitation in computation through vector processing, super scalar and VLIW processing, array processing, associative processing, systolic algorithms, and dataflow computation. After introducing interconnection networks, it discusses parallel algorithms for sorting, Fourier transform, matrix algebra, and graph theory. The second part focuses on basics and selected theoretical issues of distributed processing. Architectures and algorithms have been dealt in an integrated way throughout the book. The last chapter focuses on the different paradigms and issues of high performance computing making the reading more interesting. This book is meant for the senior level undergraduate and postgraduate students of computer science and engineering, and information technology. The book is also useful for the postgraduate students of computer science and computer application.

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".

An Introduction to Distributed Algorithms takes up some of the main concepts and algorithms, ranging from basic to advanced techniques and applications, that underlie the programming of distributed-memory systems such as computer networks, networks of workstations, and multiprocessors. Written from the broad perspective of distributed-memory systems in general it includes topics such as algorithms for maximum flow, program debugging, and simulation that do not appear in more orthodox texts on distributed algorithms. Moving from fundamentals to advances and applications, ten chapters—with exercises and bibliographic notes—cover a variety of topics. These include models of distributed computation, information propagation, leader election, distributed snapshots, network synchronization, self- stability, termination detection, deadlock detection, graph algorithms, mutual exclusion, program debugging, and simulation. All of the algorithms are presented in a clear, template- based format for the description of message-passing computations among the nodes of a connected graph. Such a generic setting allows the treatment of problems originating from many different application areas. The main ideas and algorithms are described in a way that balances intuition and formal rigor—most are preceded by a general intuitive discussion and followed by formal statements as to correctness complexity or other properties.

??????????????????????,?????????????????,???????????????

This book constitutes the refereed proceedings of the 5th International Symposium on Parallel and Distributed Processing and Applications, ISPA 2007, held in Niagara Falls, Canada, in August 2007. The 83 revised full papers presented together with three keynote are cover algorithms and applications, architectures and systems, datamining and databases, fault tolerance and security, middleware and cooperative computing, networks, as well as software and languages.

This volume presents the proceedings of the 2nd International Workshop on Distributed Algorithms, held July 8-10, 1987, in Amsterdam, The Netherlands. It contains 29 papers on new developments in the area of the design and analysis of distributed algorithms. The topics covered include, e.g. algorithms for distributed consensus and agreement in networks, connection management and topology update schemes, election and termination detection protocols, and other issues in distributed network control.

In modern computing a program is usually distributed among several processes. The fundamental challenge when developing reliable distributed programs is to support the cooperation of processes required to execute a common task, even when some of these processes fail. Guerraoui and Rodrigues present an introductory description of fundamental reliable distributed programming abstractions as well as algorithms to implement these abstractions. 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 specific class of abstractions, covering reliable delivery, shared memory, consensus and various forms of agreement. This textbook comes with a companion set of running examples implemented in Java. These can be used by students to get a better understanding of how reliable distributed programming abstractions can be implemented and used in practice. Combined, the chapters deliver a full course on reliable distributed programming. The book can also be used as a complete reference on the basic elements required to build reliable distributed applications.

Link reversal is a versatile algorithm design technique that has been used in numerous distributed algorithms for a variety of problems. The common thread in these algorithms is that the distributed system is viewed as a graph, with vertices representing the computing nodes and edges representing some other feature of the system (for instance, point-to-point communication channels or a conflict relationship). Each algorithm assigns a virtual direction to the edges of the graph, producing a directed version of the original graph. As the algorithm proceeds, the virtual directions of some of the links in the graph change in order to accomplish some algorithm-specific goal. The criterion for changing link directions is based on information that is local to a node (such as the node having no outgoing links) and thus this approach scales well, a feature that is desirable for distributed algorithms. This monograph presents, in a tutorial way, a representative sampling of the work on link-reversal-based distributed algorithms. The algorithms considered solve routing, leader election, mutual exclusion, distributed queueing, scheduling, and resource allocation. The algorithms can be roughly divided into two types, those that assume a more abstract graph model of the networks, and those that take into account more realistic details of the system. In particular, these more realistic details include the communication between nodes, which may be through asynchronous message passing, and possible changes in the graph, for instance, due to movement of the nodes. We have not attempted to provide a comprehensive survey of all the literature on these topics. Instead, we have focused in depth on a smaller number of fundamental papers, whose common thread is that link reversal provides a way for nodes in the system to observe their local neighborhoods, take only local actions, and yet cause global problems to be solved. We conjecture that future interesting uses of link reversal are yet to be discovered. Table of Contents: Introduction / Routing in a Graph: Correctness / Routing in a Graph: Complexity / Routing and Leader Election in a Distributed System / Mutual Exclusion in a Distributed System / Distributed Queueing / Scheduling in a Graph / Resource Allocation in a Distributed System / Conclusion

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.

This book constitutes the proceedings of the 9th International Workshop on Distributed Algorithms, WDAG '95, held in Le Mont-Saint-Michel, France in September 1995. Besides four invited contributions, 18 full revised research papers are presented, selected from a total of 48 submissions during a careful refereeing process. The papers document the progress achieved in the area since the predecessor workshop (LNCS 857); they are organized in sections on asynchronous systems, networks, shared memory, Byzantine failures, self-stabilization, and detection of properties.

This self-contained introduction to the distributed control of robotic networks offers a distinctive blend of computer science and control theory. The book presents a broad set of tools for understanding coordination algorithms, determining their correctness, and assessing their complexity; and it analyzes various cooperative strategies for tasks such as consensus, rendezvous, connectivity maintenance, deployment, and boundary estimation. The unifying theme is a formal model for robotic networks that explicitly incorporates their communication, sensing, control, and processing capabilities--a model that in turn leads to a common formal language to describe and analyze coordination algorithms. Written for first- and second-year graduate students in control and robotics, the book will also be useful to researchers in control theory, robotics, distributed algorithms, and automata theory. The book provides explanations of the basic concepts and main results, as well as numerous examples and exercises. Self-contained exposition of graph-theoretic concepts, distributed algorithms, and complexity measures for processor networks with fixed interconnection topology and for robotic networks with position-dependent interconnection topology Detailed treatment of averaging and consensus algorithms interpreted as linear iterations on synchronous networks Introduction of geometric notions such as partitions, proximity graphs, and multicenter functions Detailed treatment of motion coordination algorithms for deployment, rendezvous, connectivity maintenance, and boundary estimation

This volume contains the proceedings of the fifth International Workshop on Distributed Algorithms. The workshop was a forum for researchers in distributed algorithms, communication networks, and decentralized systems.

A lucid and up-to-date introduction to the fundamentals of distributed computing systems As distributed systems become increasingly available, the need for a fundamental discussion of the subject has grown. Designed for first-year graduate students and advanced undergraduates as well as practicing computer engineers seeking a solid grounding in the subject, this well-organized text covers the fundamental concepts in distributed computing systems such as time, state, simultaneity, order, knowledge, failure, and agreement in distributed systems. Departing from the focus on shared memory and synchronous systems commonly taken by other texts, this is the first useful reference based on an asynchronous model of distributed computing, the most widely used in academia and industry. The emphasis of the book is on developing general mechanisms that can be applied to a variety of problems. Its examples-clocks, locks, cameras, sensors, controllers, slicers, and synchronizers-have been carefully chosen so that they are fundamental and yet useful in practical contexts. The text's advantages include: Emphasizes general mechanisms that can be applied to a variety of problems Uses a simple induction-based technique to prove correctness of all algorithms Includes a variety of exercises at the end of each chapter Contains material that has been extensively class tested Gives instructor flexibility in choosing appropriate balance between practice and theory of distributed computing

Microsystem technology (MST) integrates very small (up to a few nanometers) mechanical, electronic, optical, and other components on a substrate to construct functional devices. These devices are used as intelligent sensors, actuators, and controllers for medical, automotive, household and many other purposes. This book is a basic introduction to MST for students,

engineers, and scientists. It is the first of its kind to cover MST in its entirety. It gives a comprehensive treatment of all important parts of MST such as microfabrication technologies, microactuators, microsensors, development and testing of microsystems, and information processing in microsystems. It surveys products built to date and experimental products and gives a comprehensive view of all developments leading to MST devices and robots.
?????????????????????????????????????,???
This book presents a comprehensive review of key distributed graph algorithms for computer network applications, with a particular emphasis on practical implementation. Topics and features: introduces a range of fundamental graph algorithms, covering spanning trees, graph traversal algorithms, routing algorithms, and self-stabilization; reviews graph-theoretical distributed approximation algorithms with applications in ad hoc wireless networks; describes in detail the implementation of each algorithm, with extensive use of supporting examples, and discusses their concrete network applications; examines key graph-theoretical algorithm concepts, such as dominating sets, and parameters for mobility and energy levels of nodes in wireless ad hoc networks, and provides a contemporary survey of each topic; presents a simple simulator, developed to run distributed algorithms; provides practical exercises at the end of each chapter.
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.
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.
This volume contains papers presented at the First International Workshop on Distributed Algorithms. The papers present solutions to a wide spectrum of problems (leader election, resource allocation, routing, etc.) and focus on a variety of issues that influence communications complexity.
This book constitutes the refereed proceedings of the 11th International Workshop on Distributed Algorithms, WDAG '97, held in Saarbrücken, Germany, in September 1997. The volume presents 20 revised full papers selected from 59 submissions. Also included are three invited papers by leading researchers. The papers address a variety of current issues in the area of distributed algorithms and, more generally, distributed systems such as various particular algorithms, randomized computing, routing, networking, load balancing, scheduling, message-passing, shared-memory systems, communication, graph algorithms, etc.
This book aims at being a comprehensive and pedagogical introduction to the concept of self-stabilization, introduced by Edsger Wybe Dijkstra in 1973. Self-stabilization characterizes the ability of a distributed algorithm to converge within finite time to a configuration from which its behavior is correct (i.e., satisfies a given specification), regardless the arbitrary initial configuration of the system. This arbitrary initial configuration may be the result of the occurrence of a finite number of transient faults. Hence, self-stabilization is actually considered as a versatile non-masking fault tolerance approach, since it recovers from the effect of any finite number of such faults in a unified manner. Another major interest of such an automatic recovery method comes from the difficulty of resetting malfunctioning devices in a large-scale (and so, geographically spread) distributed system (the Internet, Pair-to-Pair networks, and Delay Tolerant Networks are examples of such distributed systems). Furthermore, self-stabilization is usually recognized as a lightweight property to achieve fault tolerance as compared to other classical fault tolerance approaches. Indeed, the overhead, both in terms of time and space, of state-of-the-art self-stabilizing algorithms is commonly small. This makes self-stabilization very attractive for distributed systems equipped of processes with low computational and memory capabilities, such as wireless sensor networks. After more than 40 years of existence, self-stabilization is now sufficiently established as an important field of research in theoretical distributed computing to justify its teaching in advanced research-oriented graduate courses. This book is an initiation course, which consists of the formal definition of self-stabilization and its related concepts, followed by a deep review and study of classical (simple) algorithms, commonly used proof schemes and design patterns, as well as premium results issued from the self-stabilizing community. As often happens in the self-stabilizing area, in this book we focus on the proof of correctness and the analytical complexity of the studied distributed self-stabilizing algorithms. Finally, we underline that most of the algorithms studied in this book are actually dedicated to the high-level atomic-state model, which is the most commonly used computational model in the self-stabilizing area. However, in the last chapter, we present general techniques to achieve self-stabilization in the low-level message passing model, as well as example algorithms.
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.
AN ELABORATE YET BEGINNER-FRIENDLY GUIDE TO DISTRIBUTED ALGORITHMS Distributed Algorithms, a non-trivial and highly evolving field of active research, is often presented in most publications using a heavy accompaniment of mathematical techniques and notations. Aimed squarely at beginners as well as experienced practitioners, this book attempts to demystify and explicate the subject of distributed algorithms using a highly expansive and verbose style of treatment. Covering scores of landmark algorithms in the field of distributed computing, the approach is to present and analyse each topic using a minimum of mathematical exposition, reverting instead to a fluid style of description in plain English. A mathematical presentation is avoided altogether whenever such a move does not reduce the quality of the analysis at hand. Elsewhere, the effort always is to talk and guide the reader through the relevant math without resorting to a series of equations. To backup such a style of treatment, each topic is accompanied by a multitude of examples, flowcharts, and diagrams. The book is divided into three parts; the first part deals with fundamentals, the second and largest of the three is all about algorithms specific to message passing networks, while the last one focuses on shared memory algorithms. The beginning of the book dedicates a few chapters to the basics - including a quick orientation on the underlying platform, i.e. distributed systems, their characteristics, advantages, challenges, and so on. Some of the earlier chapters also address basic algorithms and techniques relevant to distributed computing environments before moving on to progressively complex algorithms and results - en route to the later chapters in the second part which deal with widely used 'industrial-strength' protocols such as Paxos and Raft. The third part of the book does assume a basic orientation towards computer programming, and presents numerous shared memory algorithms where each one is accompanied by a detailed description, analysis, pseudo code, and in some cases, code (C or C++). Whenever actual code is used, the syntax is kept as basic as possible - incorporating only elementary features of the language - so that newbie programmers can follow the presentation smoothly. Lastly, the target audience of the book is wide enough to cover beginners such as students or graduates joining the industry, experienced professionals wishing to migrate from monolithic frameworks to distributed ones, as well as readers with years of experience on the subject of distributed computing. The style of presentation is selected with the first two classes of readers in mind: those who wish to quickly ramp up on the subject of distributed algorithms for professional reasons or personal ones. While staying true to the stated aim, the book does not shy away from dealing with complex topics. A concise list of content information follows: Introduction to distributed systems Properties of distributed data stores and Brewer's theorem Building blocks: unicast, broadcast, algorithms in cubes Leader election algorithms: for ring/generic networks Consensus algorithms: synchronous/asynchronous variants for message passing and shared memory systems Distributed commits, Paxos, Raft Graph algorithms Routing algorithms Time and order Mutual exclusion: for message passing networks Debug algorithms: snapshot, deadlock/termination detection Shared memory: practical problems, mutual exclusion, consensus, resource allocation About the author Fourré Sigs is an industry veteran with over 25 years of experience in systems programming, networking, and highly scalable and secure distributed service architectures.
??????:???
Proceedings of the 4th of a series of workshops on distributed algorithms. The workshop was a forum for researchers and others to discuss recent results and trends in the design and analysis of distributed algorithms for communication networks and decentralized systems.
The use of distributed algorithms offers the prospect of great advances in computing speed. This book provides a clear, practical, and up-to-date guide to distributed algorithms and protocols in the area of control. Much of the material has been heretofore unavailable in English. Each chapter considers a specific aspect of control, with an analysis of the problem, a description of the algorithm for solving it, and proofs of correctness. Chapters can be studied independently to find solutions to particular problems.
Distributed algorithms have been the subject of intense development over the last twenty years. The second edition of this successful textbook provides an up-to-date introduction both to the topic, and to the theory behind the algorithms. The clear presentation makes the book suitable for advanced undergraduate or graduate courses, whilst the coverage is sufficiently deep to make it useful for practising engineers and researchers. The author concentrates on algorithms for the point-to-point message passing model, and includes algorithms for the implementation of computer communication networks. Other key areas discussed are algorithms for the control of distributed applications (wave, broadcast, election, termination detection, randomized algorithms for anonymous networks, snapshots, deadlock detection, synchronous systems), and fault-tolerance achievable by distributed algorithms. The two new chapters on sense of direction and failure detectors are state-of-the-art and will provide an entry to research in these still-developing topics.
????????????,???????????????,??????????????,????????????????????
This book includes the papers presented at the Third International Workshop on Distributed Algorithms organized at La Colle-sur-Loup, near Nice, France, September 26-28, 1989 which followed the first two successful international workshops in Ottawa (1985) and Amsterdam (1987). This workshop provided a forum for researchers and others interested in distributed algorithms on communication networks, graphs, and decentralized systems. The aim was to present recent research results, explore directions for future research, and identify common fundamental techniques that serve as building blocks in many distributed algorithms. Papers describe original results in all areas of distributed algorithms and their applications, including: distributed combinatorial algorithms, distributed graph algorithms, distributed algorithms for control and communication, distributed database techniques, distributed algorithms for decentralized systems, fail-safe and fault-tolerant distributed algorithms, distributed optimization algorithms, routing algorithms, design of network protocols, algorithms for transaction management, composition of distributed algorithms, and analysis of

distributed algorithms.
This book provides a comprehensive overview of both the hardware and software issues involved in designing state-of-the-art distributed and parallel computing systems. Essential for both students and practitioners, this book explores distributed computing from the bottom-up approach, starting with computing organization, communications and networks, and then discussing operating systems, client/server architectures, distributed databases and other applications. The book also includes coverage of parallel language design, including Occam and Linda. Each chapter ends with questions, and the book contains an extensive glossary and list of reference sources.
?????:????
Copyright: 2a985abc8c3c673d85e85d5e38475c76