

PROF. R. DE PRISCO


BISS 2009
Bertinoro International Spring School

Distributed Algorithms

Roberto De Prisco
University of Salerno, Italy

1


PROF. R. DE PRISCO

Distributed Algorithms  Bertinoro, March 2009
Intro SLIDE # 2

PROF. R. DE PRISCO


Distributed systems

- From “Distributed algorithms”
 - N. A. Lynch
 - Morgan Kaufman, ISBN 1-55860-348-4
- “*Distributed algorithms* are algorithms designed to run on hardware consisting of many interconnected processors. Pieces of a distributed algorithm run concurrently and independently, each with only a limited amount of information. The algorithms are supposed to work correctly, even if the individual processors and communication channels operate at different speeds and some components fail.”

Distributed Algorithms  Bertinoro, March 2009
Intro SLIDE # 3
PROF. R. DE PRISCO


Distributed systems

- From “Distributed computing”
 - H. Attiya, J. Welch
 - Wiley, ISBN 978-0-471-45324-6
- “A *distributed system* is a collection of individual computing devices that can communicate with each other. This very general definition encompasses a wide range of modern-day computer systems, ranging from a VLSI chip, to a tightly-coupled shared memory multiprocessor, to a local area cluster of workstations, to the Internet.”

Distributed Algorithms  Bertinoro, March 2009
Intro SLIDE # 4
PROF. R. DE PRISCO


Distributed systems

- From “Reliable distributed programming”
 - R. Guerraoui, L. Rodrigues
 - Springer, ISBN 3-540-28845-7
- “*Distributed computing* has to do with devising algorithms for a set of processes that seek to achieve some form of cooperation. Besides executing concurrently, some of the processes of a distributed system might stop operating, for instance, by crashing or being disconnected, while others might stay alive and keep operating.”

Distributed Algorithms  Bertinoro, March 2009
Intro SLIDE # 5
PROF. R. DE PRISCO


Distributed systems

- From “Distributed algorithms”
 - G. Tel
 - Cambridge U. Press, ISBN 0-521-47069-2
- “By *distributed system* we mean all computer applications where several computers or processors cooperate in some way. This definition includes wide-area computer communication networks, but also local-area networks, multiprocessor computers in which each processor has its own control unit, and systems of cooperating processes.”

Distributed Algorithms  Bertinoro, March 2009
Intro SLIDE # 6
PROF. R. DE PRISCO

Distributed systems

- From “Reliable Distributed Systems”
 - K.P. Birman
 - Springer, ISBN 978-0-387-21509-9
- “A *distributed computing system* is a set of computer programs, executing on one or more computers, and coordinating actions by exchanging *messages*. A *computer network* is a collection of computers interconnected by hardware that directly supports message passing.”




Distributed Algorithms
PROF. R. DE PRISCO

Distributed systems

Bertinoro, March 2009
Intro SLIDE # 7

- From “Distributed systems”
 - A. S. Tanenbaum, M. Van Steen
 - Pearson, ISBN 0-13-239227-5

- “A *distributed system* is a collection of independent computers that appears to its users as a single system.”



Distributed Algorithms
PROF. R. DE PRISCO

Distributed systems

Bertinoro, March 2009
Intro SLIDE # 8

Received: by jumbo.dec.com (5.54.3/4.7.34)
 id AA09105; Thu, 28 May 87 12:23:29 PDT
 Date: Thu, 28 May 87 12:23:29 PDT
 From: lamport (Leslie Lamport)
 Message-Id: 8705281923.AA09105@jumbo.dec.com
 To: src-t
 Subject: distribution


There has been considerable debate over the years about what constitutes a distributed system. It would appear that the following definition has been adopted at SRC:

A distributed system is one in which the failure of a computer you didn't even know existed can render your own computer unusable.

The current electrical problem in the machine room is not the culprit--it just highlights a situation that has been getting progressively worse. It seems that each new version of the nub makes my FF more dependent upon programs that run elsewhere. Having to wait a few seconds for a program to be swapped in is a lot less annoying than having to wait an hour or two for someone to reboot the servers. I therefore propose a development project to make our system more robust. I am not proposing any particular approach (enabling stand-alone operation is just one possibility).


I will begin the effort by volunteering to gather some data on the problem. If you know of any instance of user's FF becoming inoperative through no fault of its own, please send me a message indicating the user, the time, and the cause (if known).

Leslie

Distributed Algorithms  Bertinoro, March 2009
Intro SLIDE # 9
PROF. R. DE PRISCO


Distributed systems

- The term “distributed system”
 - Denotes the overall hardware and software infrastructure
- Today’s systems
 - Inherently distributed
- The term “distributed algorithms”
 - refer to the algorithms for a distributed system
 - fundamental to make distributed systems work

Distributed Algorithms  Bertinoro, March 2009
Intro SLIDE # 10
PROF. R. DE PRISCO


Computer networks

- Using a broad definition of “computer” and of “network” ...
- ... a distributed system is a computer network
- Abstraction: graph
 - Nodes are the computing devices
 - Edges are the channels

Distributed Algorithms  Bertinoro, March 2009
Intro SLIDE # 11
PROF. R. DE PRISCO


Distributed algorithms

- Distributed algorithms
 - Many interconnected processors
 - Concurrent, independent
 - Each with limited information
 - about the network
 - about the clocks
 - about the input
- Problems
 - Different speeds (processors and channels)
 - Failures

Distributed Algorithms  Bertinoro, March 2009
Intro SLIDE # 12
PROF. R. DE PRISCO


Distributed systems

- Good reasons for using them
 - Information exchange
 - Resource sharing
 - Increased reliability (replication)
 - Increased performance (parallelization)
- Wide range of applications
 - Telecommunications
 - Distributed information processing
 - Scientific computing
- Real life examples
 - Telephone systems
 - Airline reservation systems
 - Banking systems

Distributed Algorithms  Bertinoro, March 2009
PROF. R. DE PRISCO Intro SLIDE # 13


Classification

- Models
 - Interprocess communication
 - Timing assumption
 - Failures
- Uncertainty and independence
 - Unknown number of processors, topology
 - Independent inputs at different locations
 - Programs started at different times, different speeds
 - Uncertain message delivery time

Distributed Algorithms  Bertinoro, March 2009
PROF. R. DE PRISCO Intro SLIDE # 14


Interprocess communication

- How process communicate?
- Commonly
 - Sending point-to-point or broadcast messages
 - Remote procedure calls
 - Shared memory
- Computer networks
 - messages

Distributed Algorithms  Bertinoro, March 2009
Intro SLIDE # 15
PROF. R. DE PRISCO


Timing model

- Synchronous
 - Processors and communication work at perfect lock-step synchrony
- Asynchronous
 - Processors and communication take arbitrary time
- Partially synchronous
 - Wide range of assumptions about timing

Distributed Algorithms  Bertinoro, March 2009
Intro SLIDE # 16
PROF. R. DE PRISCO

Failures

- Distributed systems without failures
 - ... piece of cake
- Failures are always possible
 - Harder problem solutions, sometimes impossible
 - f -failure resilient algorithms
 - wait-free algorithms
- Common type of failures
 - Process stop or Byzantine failures
 - Duplication, loss or reordering of messages



Distributed Algorithms
PROF. R. DE PRISCO


Our focus

Bertinoro, March 2009
Intro SLIDE # 17

- Fundamental distributed problems
 - Abstractions
 - E.g. consensus

- Algorithms
 - ... or impossibility results

- Message passing: Network algorithms
 - Also shared memory



Distributed Algorithms
PROF. R. DE PRISCO


Course Plan

Bertinoro, March 2009
Intro SLIDE # 18

- PART I
 - Synchronous Network Algorithms
 - Some times easy but often a useful intermediate step towards more realistic models
 - Impossibility results carry over


- PART II
 - Asynchronous Network Algorithms
 - Harder, algorithms designed for this model work in practical settings
 - A few things about partially synchronous Network Algorithms
 - More realistic models

- PART III
 - Shared memory

Distributed Algorithms  Bertinoro, March 2009
Intro SLIDE # 19
PROF. R. DE PRISCO


Complexity measures

- Complexity measures
 - Time
 - Message complexity
- Time
 - synchronous system: rounds
 - asynchronous systems: need external observer
 - partially synchronous: there are time bounds
- Messages
 - number of messages
 - sometime also size (number of bits sent)

Distributed Algorithms  Bertinoro, March 2009
Intro SLIDE # 20
PROF. R. DE PRISCO

Reasoning about distributed systems

- Distributed algorithms can be deceptive
 - Look simple but can be complex
 - Unanticipated behavior
- Rigorous reasoning is necessary
 - e.g. Invariant assertions



PROF. R. DE PRISCO


Describing distributed systems

Bertinoro, March 2009
Intro SLIDE # 21

- A general model
 - A set of states (s)
 - A set of actions (α)
 - A set of executions

$$s_0 \xrightarrow{\alpha_0} s_1 \xrightarrow{\alpha_1} s_2 \xrightarrow{\alpha_2} s_3 \dots$$

- State
 - Complete instantaneous state of the system
- Action
 - A system operation




PROF. R. DE PRISCO

Safety and liveness

Bertinoro, March 2009
Intro SLIDE # 22

- Safety
 - Something bad never happens
 - Sequential computing: Correctness
 - the program never gives a wrong answer
 - Examples
 - Mutual exclusion: No two processes are in the critical section
 - Consensus: No two processes output a different decision
- Liveness
 - Something good eventually does happen
 - Sequential computing: Termination
 - the program eventually gives an answer
 - Examples
 - Mutual exclusion: If a process asks to enter the critical section it will eventually enters it
 - Consensus: Processes eventually output a decision

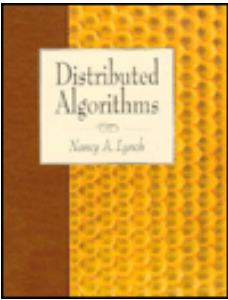



PROF. R. DE PRISCO

Textbook

Bertinoro, March 2009
Intro SLIDE # 23

- Distributed Algorithms
 - by Nancy Lynch
- Book parts
 - Part I
 - Synchronous systems
 - Part II
 - Asynchronous systems
 - Shared memory
 - Message passing
 - Part III
 - Partially synchronous






PROF. R. DE PRISCO


Conferences

Bertinoro, March 2009
Intro SLIDE # 24

- PODC
- DISC
- SPAA
- DSN
- CONCUR
- ICDCS
- IPDPS
- SIROCCO
- many more

Other conferences: STOC, FOCS, ICALP, SODA

 Distributed Algorithms PROF. R. DE FRISCO	<h2>PODC Dijkstra Awards</h2>	Bertinoro, March 2009 Intro SLIDE # 25
<ul style="list-style-type: none"> • 2000: Leslie Lamport "Time, Clocks, and the Ordering of Events in a Distributed System" <i>Communications of the ACM</i>, 21(7):558-565, July 1978. • 2001: M. J. Fischer, N. A. Lynch, and M. S. Paterson "Impossibility of Distributed Consensus with One Faulty Process," <i>Journal of the ACM</i>, 32(2):374-382, April 1985. • 2002: Edsger W. Dijkstra "Self-stabilizing systems in spite of distributed control" <i>Communications of the ACM</i>, 17(11):643-644, November 1974. • 2003: Maurice Herlihy "Wait-Free Synchronization", <i>ACM Trans. on Progr. Lang. and Systems</i>, 13(1):124-149, Jan 1991. • 2004: R. G. Gallager, P. A. Humblet, and P. M. Spira "A Distributed Algorithm for Minimum-Weight Spanning Trees", <i>ACM Trans. on Progr. Lang. and Systems</i>, 5(1):66-77, Jan 1983. 		

 Distributed Algorithms PROF. R. DE FRISCO	<h2>PODC Dijkstra Awards</h2>	Bertinoro, March 2009 Intro SLIDE # 26
<ul style="list-style-type: none"> • 2005: M. Pease, R. Shostak, and L. Lamport "Reaching agreement in the presence of faults," <i>Journal of the ACM</i>, 27(1):228-234, April 1980. • 2006: John M. Mellor-Crummey and M. L. Scott "Algorithms for scalable synchronization on shared-memory multiprocessors," <i>ACM Trans. on Computer Systems</i>, 9(1):21-65, Feb 1991. • 2007: C. Dwork, N. Lynch, and L. Stockmeyer "Consensus in the presence of partial synchrony," <i>Journal of the ACM</i>, 35(2):288-323, April 1988. • 2008: B. Awerbuch and D. Peleg "Sparse Partitions," <i>Proc. of the 31st Symp. on Foundations of Computer Science (FOCS)</i>, 503-513, October 1990. 		