Distributed Systems

Rik Sarkar
James Cheney

Course review

March 24, 2014

Introduction

- What is a distributed system?
  - Independent computers
  - Coordination achieved only through message passing
- Relationship to networks (substrate)
  - Networks: how to connect computers
  - Distributed systems: how to use that capability to do other things

- Examples:
  - Web browsing
  - Multiplayer Games
  - Stock markets
  - Hadoop and Big data processing
  - Networks
  - Mobile and sensor systems
  - Ubiquitous computing
  - Autonomous vehicles

Fundamental issue in a distributed system

- Information/knowledge is different for different nodes

- Practical challenges
  - Communication, number of nodes, time mismatch, failure, mobility, transparency, security…

Communication

- Types of network
- Communication by packets: More data means more packets
- Types of communication and their properties
  - wired/Ethernet point-to-point
  - wifi (broadcast)
- Routing tables
- Network as a graph

Communication costs

- Number of messages/packets
- Efficiency of a distributed computation is measured in communication cost
- Examples: Addition of numbers
  - Star network, chain network
- Asymptotic complexity of communication
- Big, O, Ω, θ

Basic algorithms

- Network as graph
  - Radius, diameter, spanning tree
- Size of node Ids in a network of n nodes
- Global message broadcast in a network
  - Flooding
  - Complexity of flooding O(|E|)

Trees and BFS trees

- BFS trees
- Construction of BFS trees
- Using trees for broadcast
- Use in routing and shortest paths

- Aggregation with & without trees
  - Convergecast

Other topics

- Directed graphs
- Bit complexity of communication
  - Counting number of bits instead of messages
- Bellman-Ford algorithm for shortest paths
  - (**will not be in exam**)

Systems and models

- How to think about
  - Hardware
  - Energy
  - Communication
  - Architecture: How software components are related
  - Failures
  - Computation
  - Time and synchronization
  - Security
  - Mobility

Models

- Modeling is necessary to think about complex systems
- There is no one "best" way of modeling
- Depends on system, problem, type of solution etc

- Overlay networks (virtual connection patterns on top of real networks)

- Synchronous and asynchronous communication

Clocks and Synchronization

- Impossibility of perfect synchronization
- Practical techniques for attaining approximate synchronization
  - Cristian's algorithm
  - Berkeley algorithm
  - NTP
- For Berkeley and NTP, covered high level idea only, **details not on exam**

Logical clocks

- Idea of "logical" time in a distributed system
- Happens-before relation & event diagrams
  - "Concurrent" events: neither happened before other
- Lamport clocks (map event order to single number)
- Vector clocks (map event order to vectors)
- True history vs. runs and linearizations
- How to recover ordering of events from vector clock values

Global state

- Infeasible to stop whole system to examine its state
- Cuts and consistency: possible "global" states of a process
- Chandy-Lamport algorithm for getting snapshot of global state
  - Application to *stable* properties: those that stay true once they become true
  - Reachability: computed state is indistinguishable from some actual state, up to reordering concurrent events

Distributed debugging

- Marzullo-Neiger algorithm
  - used to detect non-stable properties
- Compute lattice of possible global states from vector clock timestamps
- Breadth-first search of the lattice to determine
  - whether some state satisfies P (*possibly* P)
  - or all paths eventually go through a state satisfying P (*definitely* P)

Coordination and agreement

- In a distributed system, need ways for multiple processes to coordinate or agree on values
- Mutual exclusion:
  - Central server (token) algorithm
  - Ring algorithm
  - Ricart-Agrawala (timestamp-based) algorithm
  - Maekawa voting algorithm
- Key properties:
  - safety: only one process in CS
  - liveness: no deadlock due to the algorithm
  - fairness: if one request happens before another, then first is granted before second
  - Performance: number fo messages to acquire, release, etc.

Failure and leader election

- Models/meaning of failure
- Detection of crash failure
  - Synchronous and asynchronous
- Reliable and unreliable
  - Timeouts, probabilistic reasoning

Leader election

- Why we need leaders
- Examples of computations that need leaders

- Leader election
  - Aggregation tree
  - Ring based (Chang and roberts) messages going in in 1 direction
  - Ring based (Hirschberg Sinclair) messages going in both directions (search k neighborhoods)
  - Bully algorithm

Multicast

- Multicast and groups
  - Multicast is broadcast to a group (insed of all nodes)
  - In LANs implemented as broadcast + selection
  - More complex in internetworks
- IP multicast
- Reliable multicast as a service (say in a networked OS)
- Reliable multicast implemented using basic multicast
- Ordering of messages in delivery
  - FIFO, causal, total

Consensus

- Processes have to agree on something
- Basic consensus
- Byzantine agreement problem
- Why it is difficult
- Impossible in 3 node system with 1 traitor
- Impossible in N node system with N/3 byzantine failures
- Impossible in asynchronous systems

Termination detection

- How to detect a computation has ended
  - Initiator starts with a weight 1.0.
  - Every message carries some weight – added to weight of receiver
  - A process that has finished, sends its current weight to initiator/coordinator

Peer to peer

- Advantages/disadvantages of client-server wrt P2P
  - Scalability, fault tolerance, cost, participation
- Issues:
  - Bootstrapping, finding content, quality of service/data, (lack of) control

Peer to peer

- Examples
  - Early Internet/ARPAnet
  - SETI@Home
  - Napster
  - BitTorrent
  - Gnutella
  - Skype
- When to use and not to use p2p systems

Peer to peer: theory

- Hash tables
- Distributed hash tables
- Example system: chord
- Efficient search in chord
- Magnet links (**will not be in exam**)
- Grid based DHT and double rulings (**will not be in exam**)

Distributed operating systems

- OS as a resource manager
- Acts as arbitrator for demands of different resources
- Threads
- Networked OS vs Distributed OS
- Advantages and disadvantages of distributed OS

Virtualization

- Uses
- Why and when virtualization is useful in large distributed systems
- Analogy between virtualization and universal turing machines (**will not be in exam**)
- Current trends in OS are all related to distributed computing
    - Virtualization, mobiles OS, embedded/sensor OS etc..

Security in distributed systems

- Relationship of security to cryptography
  - not the same thing
- No "security thorough obscurity"
- Security goals:
  - **secrecy**: attacher cannot learn important (secret) data
  - **integrity**: attacker cannot change important (public) data
  - **availability**: normal users of system cannot be denied service by attacker
- Types of attacks arising from open communications, open interfaces, lack of built-in trust/authenticity

Cryptography and Security protocols

- Symmetric / shared secret key cryptography
  - Communication: same key to encode and decode
- Asymmetric / Public key cryptography
  - Communication: use public key to encode - can only be decoded with private key
  - Signing: use private key to encode - anyone with public key can decode & verify
- Common protocols:
  - shared secret communication
  - authentication (Needham-Schroeder)
  - bootstrapping public -> shared secret communication
  - digital signatures
  - public key certification (e.g. e-commerce)

Security case studies

- Needham-Schroeder and Kerberos
  - need for nonces, weaknesses of early versions
- TLS (used in HTTPS)
- IEEE 802.11 / WiFi / WEP
  - weaknesses in early versions, key lengths too short
- Bitcoin
- Should be able to explain what these do and how they work at high-level; **exam will not cover technical details.**

Examinable material

- Main definitions and properties of distributed systems
- Algorithms covered in lectures (with some exceptions)
  - should know properties/complexity/be able to explain examples
  - should be able to adapt algorithms to solve related problems
- Systems / applications covered in lectures
  - should know properties / be able to explain behavior
  - exam does not rely on knowledge of technical material not covered in lectures
- Theory questions from assignment are similar to exam questions
  - though assignment only covered material from early in course

Suggested readings

- The exam covers material from the lectures
- Readings from Coulouris et al. may be helpful to supplement review of the lecture slides:
  - Time & global state (ch. 14 5th ed)
  - Coordination & agreement (ch. 15 5th ed)
  - Security (ch. 11 5th ed)

Previous years' exams online

- http://www.ed.ac.uk/schools-departments/information-services/library-museum-gallery/exam-papers
- look for "Informatics" and "Distributed Systems"
- recent years exams are similar in structure
  - exact format and material covered may differ though