Distributed Systems (5DV147)

Review and evaluation

Fall 2013

Learning outcomes

- Explain general properties, challenges, and characteristics of distributed systems
- Explain the notions of causality and time in light of the design and implementation of distributed systems
- Explain general distributed algorithms for synchronization and concurrency, coordination, transactions, and replication
- Compare replication schemes with respect to performance, availability, and consistency concerns
- Explain practical issues that need to be considered when designing, implementing, and debugging distributed systems

Skills and abilities

- Design, implement, and debug distributed systems
- Employ fundamental distributed algorithms to solve problems that arise when engineering distributed systems
- Explain the inner mechanisms of current production distributed systems

… in detail …

Time

- We don’t have universal time
- Logical clocks are based on events in processes and the inter-event relationships – Happened-before relation
  - Lamport's logical clocks are simple, but have problems with concurrent events
  - Vector clocks are more powerful, but also more costly

Global states

- Global state of a system
  - check that a property is true for the system as it executes
- Global state includes the state of processes and channels
- We learned how to captured consistent global states corresponding to consistent cuts

  ‘snapshot’ algorithm (Chandy & Lamport)

Coordination

- Failure detection, hard to get right in asynchronous systems
  - Unreliable failure detection, unsuspected vs. suspected
  - Reliable failure detection, unsuspected vs. failed
- Distributed mutual exclusion
  - Coordinate access to a shared resource (critical section)
  - Properties : *safety*, *liveness*, → *ordering* (ensures fairness)
  - central server, ring-based, Ricart and Agrawala, Maekawa's voting algorithm

Coordination and agreement

- Election algorithms
  - choose a process to play a particular role in the system
  - ring-based and bully algorithms
- Agreement
  - agree on a value after one or more of the processes has proposed one, even in the presence of faults
  - Consensus, Fisher impossibility result, Byzantine generals

Group communication

- Different types of groups: open, closed, overlapping, and non-overlapping
- Reliability in group communication: integrity, validity, and agreement
- Group membership management: changes, failure detection, notification of membership changes, group address expansion
- Message ordering
  - Ordering is expensive in terms of delivery latency and bandwidth consumption
  - FIFO: order messages from each sender
  - Causal: order messages across senders
  - Total: same message ordering on all recipients
  - Hybrid orderings

Replication

- Why replication is necessary?
  - Performance, availability, reliability, fault-tolerance
- General replication phases

  Request, coordination, execution, agreement, response
- Passive replication
  - A single primary replica manager and one or more backup replica managers
  - Implements linearizability
- Active replication
  - Independent replica managers executing all operations
  - Implements sequential consistency

Consistency

- Several data consistency models
  - strict, linearizability, sequential, causal
  - Differ in how restrictive they are, how complex their implementations are, ease of programming, performance
- Client-centric consistency models
  - Eventual consistency: Monotonic reads, Monotonic writes, Read your writes, Writes follow reads
- Consistency protocols
  - primary-based protocol (remote-write, local-write)
  - quorum-based protocols
- Cassandra tutorial

Transactions

- Series of requests to a server with two requirements:
  - Concurrency control: no interference by operations from other clients
  - Indivisibility: either all requests are completed successfully or have no effect at all (not durable)
  - in the presence of server crashes
- Concurrency control protocols – locks and optimistic concurrency control
  - Locks: (strict) two-phase locking, shared locks, nested transactions but give rise to deadlocks
  - Optimistic concurrency control: backward and forward validation
- Two-phase commit

Other topics

- Indirect communication
  - Publish/subscribe systems
- Highly available services
  - CAP Theorem
- Peer-to-peer
  - Routing overlays, DHTs, BitTorrent
- Introduction to security
- Introduction to system performance

So what will come in the exam?

$$\left.\begin{array}{l}\text{coordination and synchronization,}\\ \text{message orderings, group communication,}\\ \text{replication schemes, consistency, time,}\\ \text{transactions, global states, peer-to-peer}\end{array}\right\}\ 90\% - 95\%$$

Other topics (security, system performance) → 5% – 10%

Questions about persistent Gcom

Course evaluation

The End