Reliable Distributed System Approaches

Manuel Graber
Seminar of Distributed Computing
WS 03/04

The Papers

- The Process Group Approach to Reliable Distributed Computing
  K. Birman; Communications of the ACM, 1993
- Spinglass: Secure and Scalable Communication Tools for Mission-Critical Computing
  K. Birman, R. van Renesse, W. Vogels; DARPA DISCEX-2001

Goal of this talk

- Be aware of the problems in distributed systems
- Overview two proposed solutions
- Pros and cons of these solutions
- Comparison of the two papers

Not Goal of this Talk

- Transfer of detail knowledge
- Proofs
- Implementations

Contents of this Talk

1. Part: Process Group Approach and Virtual Synchrony
2. Part: Downside of Virtual Synchrony
3. Part: Gossip Algorithms

Paper #1

The Process Group Approach to Reliable Distributed Computing
K. Birman; Communications of the ACM, 1993

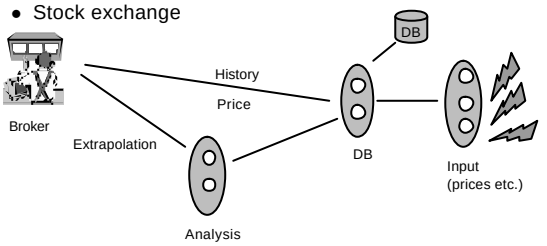
- Developer of ISIS, a Group Communication “Middleware”.
  Paper reviews 10 years of research.
  Commercialized

Why Process Groups?

- Every job in a Distributed System (DS) is assigned to several processes or nodes.
- Improving performance
- Improving reliability

Example Application

- Stock exchange



Classification of Groups: Anonymous Groups

- Publish/Subscribe paradigm
- Properties needed
  - Membership
    join/leave, group address, state transfer
  - Multicast
    exactly once semantics, message delivery in some sensible order

Classification of Groups: Explicit Groups

- Several nodes cooperate to solve a task
  Examples:
  - parallel database search
  - backup processes
- Additional property:
  Membership list must be consistent at all nodes.

How to implement?

What do we get from conventional systems:

- unreliable datagrams (example UDP)
  loss, duplicates, out-of-order
- remote procedure call
  relatively reliable, but when failure unable to distinguish where
- reliable data streams (example TCP)
  better than unreliable, but also inconsistencies possible

ISIS LAN-Model

- message loss in transit
- out-of-order arrival
- duplicates
- discard messages due to buffer space
- partitions are rare

ISIS Failure Model

- fail-stop
  + simple
  + easy to deal with
  - realistic?
  - accuracy?
  - transient problems?
  - performance?

The Group Addressing Problem

- Send messages to “all” members of a group

BUT: What means all, when members can leave or join?

Simple Solution:
Think like Database guys.
Send message: acquire “read” lock first
Change group membership: acquire “write” lock first

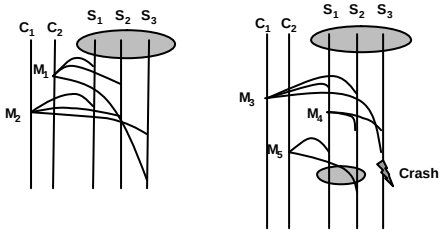
Using a lock-style mechanism

+ well researched
+ well accepted
- performance
- reliability (central database)

Message delivery ordering problems

- Real time not possible due to unpredictable delays
- Ordering of concurrent / sequentially related messages
- Causal dependency:
  P receives a message m_1 and then P sends a message m_2 **because** it earlier received m_1.

Message ordering examples



Fault tolerance problems

- Protocols to solve can be quite complex
- There is an easy solution called Three-round reliable Multicast…
- **… but**:
  synchronous, performance is achieved though by asynchrony.

Summary of Problems

- weak support for reliable communication
- group address expansion
- delivery ordering concurrent messages
- delivery ordering sequentially related messages
- state transfers
- failure atomicity

Close Synchrony: Definition

- events are in the same order for any two processes
- multicasts delivered to all members send/receive at the same moment

All problems of above solved, but…

Close Synchrony: Drawback

CS cannot be applied in a practical setting

- impossible in the presence of failures
- very expensive

This leads towards Virtual Synchrony…

Virtual Synchrony: Definition

Asynchronous execution as long as its indistinguishable from the synchronous one.

Or:
Events need to be synchronized only to the degree the application is sensitive to event ordering.

Virtual Synchrony: Atomic delivery ordering

- atomic delivery ordering (ABCAST)
  - like in close synchrony
  - Useful to keep replicated data consistent.
  - expensive

Virtual Synchrony: Causal Delivery Ordering

- causal delivery ordering (CBCAST)
  Only messages that are causally dependent are delivered in the same order.
- Often causal ordering is strong enough
- less expensive than ABCAST

Virtual Synchrony: Summary

- code can be developed assuming close synchrony
- asynchronous, pipelined
- a single event oriented execution model
- failure handling through a consistent membership list

Virtual Synchrony in ISIS: Limitations

- ISIS is built using the virtual synchrony model
- Reduced availability during partitions
  ⇨ allows progress in a single partition
- Risks incorrectly classifying an operational node as faulty!

ISIS Toolkit

- ISIS offers tools for programming DS
  - NEWS
  - NMGR
  - DECEIT
- Commercially used for several applications example Swiss Stock Exchange

Virtual Synchrony?

- Virtual Synchrony is an easy programming model
- ISIS is commercialized and works properly
- Are there any negative points?

Paper #2

- Spinglass: Secure and Scalable Communication Tools for Mission-Critical Computing
  K. Birman, R. van Renesse, W. Vogels; DARPA DISCEX-2001

Why a new Technology when we have ISIS?

- most existing systems do not scale
- small networks ⇔ large networks
- DS grow larger
- New: ad-hoc networking, wireless networking
  ⇨ dynamic systems

⇨ need for a new style of guarantees: scalability, performance and throughput even under a high rate of packet loss

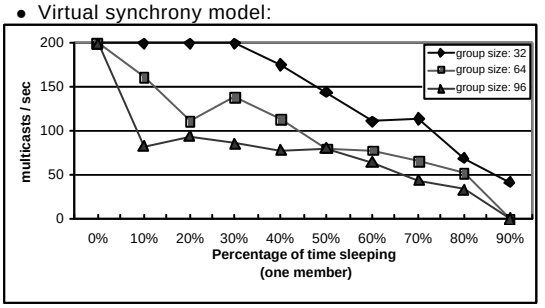
Analysis of conventional Systems: Scalability and Reliability

Many flavors of reliable MC:

- virtual synchrony model
  ⇨ example ISIS
- models with weaker reliability goals
  ⇨ example SRM (scalable reliable multicast)

Analysis of ISIS: Throughput instability

- Virtual synchrony model:



Analysis of ISIS: Micropartitions

- **Failure detectors are problematic**
  time vs. accuracy
- detector too aggressive ⇨ pay leave/rejoin
  Otherwise ⇨ pay for slow nodes
- It's a tradeoff.
  example Swiss Stock Exchange:
  FD very aggressive ⇨ less nodes per hub

Limits to Scale for traditional Models

What's the problem of all the traditional models?

- they depend on assumptions that are very rarely violated ⇨ as system grows probability grows
- they have a recovery mechanism with potentially global cost ⇨ as system scales up…

Why does the Internet work?

- Why does the internet and all the services over the internet work at all?

  We tolerate disruptions.
  As soon as we try to overcome disruptions the result is a bad scalability.

  But there is a way out…

Spinglass Approach: Epidemic-style or Gossip Algorithms

- Sites periodically compare their states
- reconcile inconsistencies with other members of group
- choose randomized when and with whom

  Similar to NNTP (network-news transport protocol, USENET)

Epidemic Protocols: Bimodal Multicast

1. - unreliable Multicast
   - messages buffered on arrival
   - delivery in FIFO order
   - empty buffer after some time

2. - partial list of group members at every node
   - send list of messages to randomly picked node
   - push/pull for exchanging missing messages

Bimodal Multicast: Optimizations

- gossip nearby nodes
- gossip also for group membership
- use a “local” multicast for push/pull
- don’t buffer every message at every node

Bimodal Multicast: Advantages

- What is now better with Bimodal Multicast?
  - constant load on participants
  - constant load on communication links
  - tunable reliability
  - very steady data delivery rates with low variability in throughput

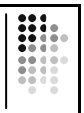
  All these characteristics are preserved as the size of the system increases.

Virtual Synchrony with Bimodal Multicast

- The reliability guarantees of Bimodal MC are different to these of Virtual Synchrony.
- It’s possible to implement Virtual Synchrony over Bimodal MC.

  For small groups slower than Virtual Synchrony but scales far better for large groups.

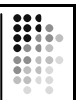
Spinglass: Probabilistic Tools



Operate directly with Bimodal MC

- Astrolabe
- Gravitational Gossip
- Anonymous Gossip

Spinglass: Applications



- Joint Battlespace Infosphere
- Galaxy
- Electric Power Grid

ISIS

- Group Management Service
  ⇨ easy to use model, but expensive
- Multicast Service
- Virtual Synchrony model
- does not scale

Spinglass

- Main goal: Scalability
- Different reliability guarantees
  (user defined)
- Uses gossiping (epidemic protocols)
- good scalability

Take Home Messages

- Process Groups are a widely used model for DS. Important applications are built on this model.
- Building a group communication application is very difficult without some helping middleware
- The traditional Virtual Synchrony solution does not scale
- Gossip is a solution which scales

Questions?

- Thanks for your attention!