RELIABLE AND SCALABLE DISTRIBUTED SYSTEMS

SEBASTIAN AMENT
ONE INCREASINGLY IMPORTANT APPLICATION

HIGH-ASSURANCE CLOUD COMPUTING

*Examples from Ken Birman’s Presentation for CS6410 Fall 2015*
AGENDA

- Presentation and Discussion of

- Background
  - Eric Brewer’s CAP Conjecture / Theorem
  - Bimodal Multicast
PROCESS GROUP TAKEAWAYS

- Reliable components do not make a reliable system
- Many applications are naturally structured into groups
- In distributed computing, Performance ~ Asynchrony
- Virtual Synchrony
COMPARISON TO STATE MACHINES

- “The *state machine approach* is a general method for implementing a fault-tolerant service by *replicating servers* and coordinating client interactions with server replicas” - Fred B. Schneider - Implementing Fault-Tolerant Services Using the State Machine Approach: A Tutorial

- Both approaches deal with reliable interprocess communication

- Members of process groups can collaborate in a variety ways, not just replicate
SINFONIA TAKEAWAYS

- Paradigm for distributed infrastructure applications
- Interprocess communication via atomic minitransactions
- Allows for implementation of process group approach
- Support for caching at application level
- Clever increase of asynchrony / performance through a pre-computation step
COMPARISON BETWEEN PROCESS GROUP APPROACH AND SINFONIA

- Both are concerned with making distributed systems reliable and efficient
- Sinfonia possesses more generality
- Both identified the main performance factor of distributed systems to be the decoupling of processes
  - Process group: Virtual Synchrony
  - Sinfonia: Minitransactions, Pre-computation, Caching
ERIC BREWER’S CAP CONJECTURE

- **CAP** stands for*
  - **Consistency**: reads receive the most recent version of the data or an error
  - **Availability**: every request receives a response
  - **Partition tolerance**: the system is fault-tolerant with respect to network failures

- States that one cannot have all three simultaneously.

- Trade-offs are possible

A naive approach to reliable data streams

What happens if connection between C and S1 breaks?

Not partition-tolerant

Figure from Birman 1993
CHALLENGES OF RELIABLE DISTRIBUTED SYSTEMS

MESSAGE-ORDERING PROBLEMS

- Concurrent messages
- Dependencies
- Note: This figure does not seem to fit the text

Figure from Birman 1993
S3 is not consistent

Figure from Birman 1993
CHALLENGES OF RELIABLE DISTRIBUTED SYSTEMS

SUMMARY OF CHALLENGES

- Reliable communication, especially in the presence of network failures
- Consistent group membership information
- Sensible group message ordering
- Consistent state transfer
- Fault tolerance
ONE MIGHT EXPECT THE RELIABILITY OF A DISTRIBUTED SYSTEM TO CORRESPOND DIRECTLY TO THE RELIABILITY OF ITS CONSTITUENTS, BUT THIS IS NOT ALWAYS THE CASE.

Kenneth P. Birman – The Process Group Approach to Reliable Distributed Computing
THE PROCESS GROUP APPROACH

- Goals: scalability, reliability, and simplicity
- Structure processes in groups
- Focus on inter- and intra group communication
- Example: Isis
  - Group multicast
  - Virtual synchrony
  - Persistent storage
TWO BASIC TYPES OF GROUPS

▷ Anonymous groups
  ▷ Arise with producer/consumer applications
  ▷ Messages should be delivered once, in sensible order

▷ Explicit groups
  ▷ Arise when applications are cooperating
  ▷ Need consistent membership information to be effective
COMMUNICATION BETWEEN GROUP MEMBERS

- **Multicast**: message from one process to multiple destination processes
  - Ensure that a message is received once by every destination
  - Ensure that multicasts are received in the same order by each process
- Explicit groups can communicate in ways unique to the application
COMMUNICATION BETWEEN GROUP MEMBERS

THREE-ROUND MULTICAST

- No pipelining, or asynchronous communication

![Three-round multicast diagram](Figure from Birman 1993)
THE PROCESS GROUP APPROACH

CLOSERLY SYNCHRONOUS EXECUTION

- Execution of each process consists of events
- A global execution consists of a set of process executions
- Every global event is seen in the same order by each process in a group
- A multicast to a group is delivered to all processes of that group

Definition from Birman 1993
CLOSELY SYNCHRONOUS EXECUTION

- Synchronous execution grants reliable, predictable behavior.
- What about efficiency?

Figure from Birman 1993
ACHIEVING CLOSE SYNCHRONY IS IMPOSSIBLE IN THE PRESENCE OF FAILURES.

Kenneth P. Birman – The Process Group Approach to Reliable Distributed Computing
VIRTUAL SYNCHRONY

- Allows for those asynchronous executions, which are indistinguishable from synchronous ones
- Prospect of dramatic performance improvement
- Sensible event ordering is crucial
- In the absence of synchronized clocks, this cannot be a temporal ordering
- See Lamport’s “Time, Clocks, and the Ordering of Events in a Distributed System”
FIFO message delivery between a client-pair won’t work.

M1 has to be delivered to both S1 and S2 before M2.
THE PROCESS GROUP APPROACH

ASYNCHRONOUS PIPELINING

- Allows P1 to make progress while S1 has not issued a read
- This works in the presence of multiple S’s

Figure from Birman 1993
THE PROCESS GROUP APPROACH

BENEFITS OF VIRTUAL SYNCHRONY

- Efficient, asynchronous communication
- Allows developers to assume close synchrony
- The notion of group state is sensible
- Dynamically changing group membership can be handled easily
- Any disadvantages? Let’s look at the assumptions.
WE ALSO ASSUME THAT LAN COMMUNICATION PARTITIONS ARE RARE.
Processes and processors are assumed to fail by halting without initiating erroneous actions or sending incorrect messages.
Uses two stages:

1. Unreliable “best effort” delivery attempt
2. Conditionally executed protocol which corrects message losses

Stage 1 is efficiently implemented using a spanning-tree over the application nodes
Graph assumes unreliable stage 1 failed

Figure from Birman et al 1999
The efficacy of groups relies on membership information.
Similar system to Isis, but more recent

Targets data center infrastructure applications

Goals: reliability, consistency, scalability

Core idea: atomic minitransactions

Message-passing protocols are not exposed to developer
The core to achieving scalability is to decouple operations executed by different hosts as much as possible.

Aguilera et al – Sinfonia: A New Paradigm for Building Scalable Distributed Systems
Minitransactions

- Reduced complexity as compared to database transactions
- Consist of three steps:
  - Pre-computation
  - Validation step
  - Action step

*Slide from Ken Birman’s Presentation for CS6410 Fall 2015
MINITRANSACTIONS: PRE-COMPUTATION

- Gives Sinfonia remarkable performance and scalability

- Computations can be executed asynchronously on cached copies of the system state without updating disk image

- System states include uniquely identifiable version numbers

- If version numbers of state replicas match, an update to the core state can be made

- This can hide compute and communication costs

*From Ken Birman’s Presentation for CS6410 Fall 2015*
Figure 12: Performance of Sinfonia with 1 memory node.

Figure from Aguilera et al 2009
Figure 14: Sinfonia scalability.
Messages are stored in local circular queue

Also, a reference to them is put in a global list

Figure 10: Basic design of SinfoniaGCS. Gray messages were successfully broadcast: they are threaded into the global list. Cross-hatched messages are waiting to be threaded into the global list, but they are threaded locally.

Figure from Aguilera et al 2009
Figure 23: SinfoniaGCS base performance as we vary the number of writers. There are 16 readers, and 8 memory nodes or Spread daemons.
COMPARISON BETWEEN PROCESS GROUP APPROACH AND SINFONIA

- Both are concerned with making distributed systems reliable and efficient
- Sinfonia possesses more generality
- Both identified the main performance factor of distributed systems to be the decoupling of processes
  - Process group: Virtual Synchrony
  - Sinfonia: Minitransactions, Pre-computation, Caching
DISCUSSION

- What trade-offs are Sinfonia and ISIS making with respect to CAP? Are the trade-offs different?
- How does Sinfonia GCS’s multicast strategy compare to ISIS?
- What did you find interesting about the papers?
- Feel free to ask your own question!
THANK YOU FOR LISTENING


wikipedia.com: CAP theorem