CSE 486/586 Distributed Systems
Logical Time

Steve Ko
Computer Sciences and Engineering
University at Buffalo

Last Time
• Clock skews do happen
• External and internal synchronization
  – Cristian’s algorithm: external synchronization
  – Berkeley algorithm: internal synchronization
  – Both designed for LAN
• NTP (Network Time Protocol)
  – Hierarchy of time servers
  – Estimates the actual offset between two clocks
  – Designed for the Internet
• Logical time
  – For ordering events, relative time should suffice.
  – Will continue today

Basics: State Machine
• State: a collection of values of variables
• Event: an occurrence of an action that changes the state, (i.e., instruction, send, and receive)
• As a program,
  – We can think of all possible execution paths.
• At runtime,
  – There’s only one path that the program takes.
• Equally applicable to
  – A single process
  – A distributed set of processes

Abstract View
• Above is what we will deal with most of the time.
• Ordering question: what do we ultimately want?
  – Taking two events and determine which one happened before the other one.

Ordering Basics
• Why did we want to synchronize physical clocks?
• What we need: Ordering of events.
• Arises in many different contexts…

What Ordering?
• Ideal?
  – Perfect physical clock synchronization
• Reliably?
  – Events in the same process
  – Send/receive events
Logical Clocks

- Lamport algorithm assigns logical timestamps:
  - All processes use a counter (clock) with initial value of zero
  - A process increments its counter when a send or an instruction happens at it. The counter is assigned to the event as its timestamp.
  - A send (message) event carries its timestamp
  - For a receive (message) event the counter is updated by max(local clock, message timestamp) + 1
- Define a logical relation happened-before (→) among events:
  - On the same process: a → b, if time(a) < time(b)
  - If p1 sends m to p2: send(m) → receive(m)
  - (Transitivity) If a → b and b → c then a → c
  - Shows causality of events

CSE 486/586 Administrivia

- PA2 is out.
  - Due on 3/1
  - Start with the content provider.
- Please understand the flow of PA1.
- Please be careful about your coding style.
- Lecture slides
  - I will try posting them a day before.
  - I will also post a PDF version.
- There is a course website.
  - Schedule, syllabus, readings, etc.

Corrected Example: Lamport Logical Time

- 3 and 7 are logically concurrent events

Vector Timestamps

- With Lamport clock
  - a "happened-before" f ⇒ timestamp(a) < timestamp (f), but
  - timestamp(a) < timestamp (f) ⇒ a "happened-before" f
- Idea?

Find the Mistake: Lamport Logical Time

- Clock Value
  - Message

Physical Time

p 1
p 2
p 3
p 4

p 1
p 2
p 3
p 4
Vector Logical Clocks

• Vector Logical time addresses the issue:
  • All processes use a vector of counters (logical clocks), \( P \) element is the clock value for process \( i \), initially all zero.
  • Each process \( i \) increments the \( i \)th element of its vector upon an instruction or send event. Vector value is timestamp of the event.
  • A send(message) event carries its vector timestamp (counter vector).
  • For a receive(message) event, \( V_{\text{receiver}}[j] = \max(V_{\text{receiver}}[j], V_{\text{message}}[j]) \) if \( j \) is not self, \( V_{\text{receiver}}[j] + 1 \), otherwise.

Comparing Vector Timestamps

• \( VT_1 = VT_2 \) if \( VT_1[i] = VT_2[i] \), for all \( i = 1, \ldots, n \)
• \( VT_1 \leq VT_2 \) if \( VT_1[i] \leq VT_2[i] \), for all \( i = 1, \ldots, n \)
• \( VT_1 < VT_2 \) if \( VT_1[i] < VT_2[i] \) for some \( 1 \leq i \leq n \)
• \( VT_1 \) is concurrent with \( VT_2 \) if \( \neg (VT_1 \leq VT_2 \land VT_2 \leq VT_1) \)

Summary

• Relative order of events enough for practical purposes
  – Lamport’s logical clocks
  – Vector clocks
• Next: How to take a global snapshot

Find a Mistake: Vector Logical Time

The Use of Logical Clocks

• Is a design decision
  • NTP error bound
    – Local: a few ms
    – Wide-area: 10’s of ms
  • If your system doesn’t care about this inaccuracy, then NTP should be fine.
  • Logical clocks impose an arbitrary order over concurrent events anyway
    – Breaking ties: process IDs, etc.

Acknowledgements

• These slides contain material developed and copyrighted by Indranil Gupta at UIUC.