Announcements

- Everybody please send your final copy to me electronically
- Midterm upcoming: Nov 2
  - Material from lectures (covered in Tanenbaum book)
  - Plus selected papers from student presentations
  - Will detail on next week’s handout
- Office hours
  - Use for small things
  - Otherwise, drop by or email for appointment
    • If you have an appointment and someone else is there, interrupt

Webmail

- Please enter your name in webmail
  - makes it so much easier to file and find your mail

Recap: Threads vs. Events

- Trade-offs in programming model, ease of implementation + use
  - New approaches: compiler support, resource-aware scheduling
- History:
  - Decade-long battle
  - First paper: Lauer/Needham, 1979: On the duality of operating system structures

Outline for Today

- Distributed Synchronization
  - Clock Synchronization
    • Physical clocks
    • Logical clocks
  - Election algorithms
  - Distributed mutual exclusion

Clock Synchronization

- Effects of clock skew
Clock Synchronization Goal

- UTC – Universal Coordinated Time
  - WWV broadcast
- Maximum drift rate $\rho$:
  - $\rho \leq |dC/dt - 1|
- Synchronize every $\delta/2\rho$ seconds to keep skew within $\delta$

Cristian’s Algorithm

- Synchronize clocks gradually by slowing down or speeding up clock rate

Berkeley Algorithm

Advanced Clock Synchronization

- May use multiple sources
- Must account for multiple paths
  - averaging
- NTP most widely used
  - accuracy $\approx 1$-50ms worldwide

Uses of Synchronized Clocks

- Generally: saves communication
- Typically for performance (not correctness)
- Leases
- At-most-once delivery of messages (Liskov’s SCMP)
  - Use timestamps to identify messages that may be duplicates
  - Server discards messages older than $G$
  - $G :\ CurrentTime - MaxLifetime - MaxClockSkew$

Logical Clocks

- Lamport’s Observation:
  - Processes don’t need to agree on what time it is, but rather on the order of (observable) events
- Events:
  - Intraprocess events
  - Sending & receiving of messages
Lamport’s “happens-before”

- “a happens before b”: \( a \rightarrow b \)
- Definition
  - If \( a \) occurs before \( b \) in the same process, then \( a \rightarrow b \)
  - If \( a \) is sending a message and \( b \) its receipt, then \( a \rightarrow b \)
- Happens-before is
  - transitive
  - partial: if neither \( x \rightarrow y \) nor \( y \rightarrow x \), then \( x \) and \( y \) are concurrent

Lamport’s Clocks

- Define \( C(a) \) such that:
  - \( a \rightarrow b \) implies \( C(a) < C(b) \)
- Computed as logical clocks \( C_i \):
  - Send own time as timestamp with each message \( C_i \)
  - Upon receipt of message with timestamp \( C_k > C_i \) forward local clock
    - \( C_i := C_i + 1 \) if \( C_k > C_i \)
- Idea: capture causality

Lamport Example

Application: Database Replication

Totally-ordered Multicast

- Application of Lamport’s Timestamp
  - Timestamp each message
  - Multicast acknowledgements
  - Deliver in Lamport order once all ack’s have been received
- Assuming reliable point-to-point in-order communication

Discrete Event Simulation

- Events
  - “packet received”, “car arrives”
- Activities
  - “process packet”, “wash car”
  - (Seen as two events)
- Schedule
  - Real-time does not count – ordering is important
Vector timestamps

- Question: Does \( C(a) < C(b) \) imply \( a \rightarrow b \)?
- Vector timestamps:
  - Each node keeps track of logical time of other nodes (as far as it’s seen messages from them) in \( V[i] \)
  - Send vector timestamp \( VT \) along with each message
  - Reconcile vectors timestamp with own vectors upon receipt using \( \text{MAX}(vt[k], V, [k]) \)
- Can implement “causal message delivery”
  - Newsgroups example: reply after message

End2End Issues & Ordered Message Delivery

- Where should ordered message delivery be implemented?
- Communication layer:
  - Disadvantage: introduces false (unnecessary) causality; may not capture all causality
- Application level:
  - Disadvantage: more complicated application

Summary

- Distributed Synchronization
  - Physical clock synchronization + applications
  - Logical clocks to maintain causality
- Thursday: File Systems
  - Soft Updates vs. Journaling
  - Coda FS

Scratch