Announcements

- Still a few (<5) of proposals unapproved
  - Make sure you have the "approved" email.
- Milestone 1 is coming up
  - Will bring sign-up sheet Thursday

Outline for Today

- Distributed Synchronization
  - Logical Clocks
  - Example

```
```

```
```

Lamport’s Clock

```
```

Application of Lamport’s Clock

- Extends to total order of events
- Event with logical time $C_i$ occurred in process $P_m$:
  $(C_i, P_m) < (C_k, P_n)$
  iff $C_i < C_k$ || ($C_i == C_k$ && $m < n$)
  - Everybody can decide on order of events
  - Can be used to implemented totally-ordered multicast (e.g., for updating replicas)
Distributed Discrete Event Simulation

- Airtraffic Simulation
  - Processes = Airports
  - Messages = Planes
    - Send = take-off, receive = landing

- FRA Airport:
  - deboard flight
  - clean plane
  - refuel
  - plane departs

From SFO: [9, 6, 4]
From JFK: [5, 2, 9]
To SFO: [3, 10]
To JFK: [9]

DDES (cont’d)

- Naïve algorithm (pessimistic):

  ```
  while (!done) {
    wait until every input queue has at least one message
    remove message with lowest Lamport stamp
    process message to advance local simulation
  }
  ```

- Logical time of first message gives lower bound of future messages from that queue
- Null messages avoid deadlock (Chandy/Misra/Bryant)

Clock Consistency

- Lamport (“Scalar”) clock is
  - Consistent: a → b ⇒ C(a) < C(b)
- But not: C(a) < C(b) ⇒ a → b
  - (not strongly consistent)

- Example on next slide

Lack of Strong Consistency

Vector Clocks (1)

Vector Clocks (2)
Vector Clocks: Strong Consistency

- Definition:
  - \( V(a) < V(b) \):
    - \( V(a) \leq V(b) \) and there exists an \( i \) : \( V_i(a) < V_i(b) \)
  - \( V(a) \leq V(b) \): for all components \( i \) : \( V_i(a) \leq V_i(b) \)

- Strongly consistent:
  - \( a \rightarrow b \iff V(a) < V(b) \)

- Also:
  - \( a \parallel b \iff V(a) \parallel V(b) \)
  - \( \iff \neg (V(a) < V(b) \lor V(b) < V(a)) \)

VC: Proving Strong Consistency

1. \( a \rightarrow b \implies V(a) < V(b) \)
   - Follows from monotonicity

2. \( V(a) < V(b) \implies a \rightarrow b \)
   - Assume \( a \) occurs in \( p \); \( b \) occurs in \( q \); \( p \neq q \)
   - Construct chain:
     \[
     a \rightarrow s_1 \rightarrow r_1 \rightarrow s_2 \rightarrow \ldots \rightarrow r_{n-1} \rightarrow s_n \rightarrow r_n \rightarrow b
     \]
   - \( r_1 \) and up don’t happen on \( p \); \( s_n \) and down don’t happen on \( q \); let \( V(b) = [ v_1 \ v_2 \ v_3 \ldots \ v_{p=k} \ldots \ v_{n} ] \)
   - Induction:
     - \( r_n \) timestamp had \( v_{p=k} \)
     - \( s_n = s_i \) or \( s_n \) timestamp had \( v = k \); consider \( r_{n+1} \)

Efficiency Considerations

- Scalar clocks:
  - Cheapest, constant message overhead
- Vector clocks:
  - Need \( n \) components for strong consistency
  - Various optimizations: compression, etc.
- Matrix clocks

Summary

- Distributed Synchronization
  - Logical clocks + applications

- Thursday: Clusters
  - Inktomi
  - Google FS

Scratch