DISTRIBUTED SYSTEMS

Lecture 2

Warm Up
- DISTRIBUTED PROTOCOLS RUN ON MULTIPLE PROCESSES
- PROCESSES ARE CONNECTED BY AN ASYNCHRONOUS NETWORK [What?]
* Assume network is fair
* Everyone is eventually served

Today
* Tools to reason about distributed protocols
  + The need for ordering actions
  + The challenge with ordering actions
Theory? + Traces + Safety And Liveness

Practice? + Clocks

Swap Two Integers $x$ And $y$

1. $x \leftarrow x + y$
2. $y \leftarrow x - y$
3. $x \leftarrow x - y$

When Correct?

1 2 3

Ordering in Distributed Protocols

Things Processes Can Do

- Computation (Run A Normal Algorithm)
- Send (p, m)

Things That Can Happen

- Failures
- Spawns
- Send $e_1$, $m$
  \[ \text{SEND MESSAGE } m \text{ TO } p \]
  \[ \text{RECEIVE } m \text{ FROM } p \]

**DISTRIBUTED PROTOCOL**

**PROJECTION**

**Trace Schedule**

**DID** $e_5$ **occur before** $e_7$?

**Things we can be sure about**:
- $e_4$ before $e_5$; $e_7$ before $e_8$
- $e_1$ before $e_8$?
- $e_4$ before $e_7$?
DISTRIBUTED PROTOCOLS LIMIT THE SET OF ADMISSIBLE TRACES

ASSUME NO MESSAGE LOSS

Process 0 (P0)
Init:
SEND(P1, PING)

ON_RECV (P1, PONG):
SEND(P1, PING)

Process 1 (P1)
ON_RECV (P0, PING)
SEND(P0, PONG)

(P0, PING); (P1, PONG); (P1, PONG)
Correctness on traces

- From above: Distributed protocols limit set of admissible traces

To what?

"Correct traces"

Safety

Property P never happens

PO

Init:

Send(P1, Epong, 0^2)

OnRecv(P1, Epong, N^2)

Send(P1, Epong, N+1^2)

Safety: An s.to PO sends Epong, 2N+1^2
LIVENESS

• Property $P$ will eventually hold

Process $P_0$ will send $\langle \text{ping, 6} \rangle$

$\varepsilon_{P_0, \text{send}(P_1, \text{ping, 03})}$

$\varepsilon_{P_1, \text{send}(P_0, \text{pong, 15})}$

$\varepsilon_{P_0, \text{send}(P_1, \text{ping, 23})}$

$\varepsilon_{P_1, \text{send}(P_0, \text{pong, 33})}$

$\varepsilon_{P_0, \text{send}(P_1, \text{ping, 43})}$

$\times \text{ Fail } P$

Trace Prefix → (look above)
Uniform Liveness
Given a prefix P, \( \exists \) some trace T such that for \( P + T \) property holds

Absolute Liveness
Given a prefix P, and any trace T
\( P + T \) is live

Clocks

\[ p_0 \quad \cdots \quad p_m \]
\[ p_0 \quad p_1 \quad p_2 \]
\[ p_0 . log \]
\[ p_1 . log \]

\( p_0 . log + p_1 . log \xrightarrow{??} \text{Trace?} \)

Thoughts on how?
Sort by time?
Traces, Causality, and Partial Order

Causality

And Partial Order

Causally Linked

Topological Sort (S \rightarrow R)
CAUSALITY

PROCESS LOGS

LAMPORT CLOCK

ON SEND \( a \)

\[ t = t + 1 \]

ON RECEIVE \( n \)

\[ t = \max(t, t_n) + 1 \]

Graph showing the Lamport clock with processes and events.
VECTOR CLOCKS

$P_0$
$P_1$
$P_2$
$P_3$