Logistics

Last class!

PS5 clarification

No regular OH next week
  - Doug, Wednesday 3:30-4:30, CSE 314
  - And, by appointment
FULL FORMAL VERIFICATION

MODEL CHECKING

PROOFS ON PAPER

TESTING

THINKING REALLY HARD
Distributed systems correctness

A distributed system is a collection of behaviors

- Sequences of states
- Potentially infinite
- Non-deterministic
Distributed systems correctness
Distributed systems correctness
Safety and liveness

Safety
- The system never does bad stuff
- Property of any prefix of any sequence

Liveness
- The system sometimes does good stuff
- Property of any (infinite?) postfix of any sequence
Safety and liveness

Safety
- The system never does bad stuff
- Property of any prefix of any sequence

Liveness
- The system sometimes does good stuff
- Property of any (infinite?) postfix of any sequence
Proving safety properties

Safety properties are invariants: P is always true

- Mutual exclusion
- Only one Multi-Paxos leader at a time
- If any two-phase commit node has aborted, none have committed

These properties must be true of any reachable state

- There are probably infinite reachable states!
Proving safety properties
Proving stuff about unbounded objects

One weird trick for proving things about all objects, even if there are an infinite number of them

Hint: how to prove something about all natural #’s?
Induction

Proving $P$ for all natural #’s:

- Prove $P(0)$

- Prove that if $P(n)$, $P(n+1)$ for all $n$
Induction for safety properties

Proving P for all reachable states:

- Prove P(Init)

- Prove that if P(S) and S \rightarrow S', then P(S')
Proving mutual exclusion by induction

Does mutual exclusion hold in the start state?

How about the inductive step?
Proving mutual exclusion by induction

Often need to strengthen the induction hypothesis

Need I such that:

- \( I \rightarrow P \)
- \( I \) is actually inductive

Examples of strengthening:

- Constrain network contents
- Further constrain state (matching between nodes)
Proving safety properties

Diagram:
- S
- S1
- S2
- Init
- P
- I
Proving mutual exclusion by induction

Coming up with inductive invariants is hard!
  - Requires really understanding how a system works
  - Provably impossible to do automatically

Proving invariants inductive is doable
  - Requires only “local” reasoning—single-step
Machine-checked proofs

Easy to get proofs about software wrong

- Math proofs: often (not always!) short, elegant
- Software proofs almost always long and boring

Idea: have machines check proofs!

- Machines can also help write the proofs
- For (way) more on this, take CSE 505
Hints for system design

Written in 1983

- Before I was born
- Year Tom graduated from college

Lampson: now researcher at MSR, MIT

- Turing award in 1992
- Laser printers, two-phase commit, WYSIWYG, Ethernet, and the personal computer

This paper: some hints for building large systems
<table>
<thead>
<tr>
<th>Why?</th>
<th>Functionality</th>
<th>Speed</th>
<th>Fault-tolerance</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Does it work?</td>
<td>Is it fast enough?</td>
<td>Does it keep working?</td>
</tr>
<tr>
<td>Where?</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Completeness</td>
<td>Separate normal and worst case</td>
<td>Shed load</td>
<td>End-to-end</td>
</tr>
<tr>
<td></td>
<td></td>
<td>End-to-end</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Safety first</td>
<td></td>
</tr>
<tr>
<td>Interface</td>
<td>Do one thing well:</td>
<td>Make it fast</td>
<td>End-to-end</td>
</tr>
<tr>
<td></td>
<td>Don’t generalize</td>
<td>Split resources</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Get it right</td>
<td>Static analysis</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Don’t hide power</td>
<td>Dynamic translation</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Use procedure arguments</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Leave it to the client</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Keep basic interfaces stable</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Keep a place to stand</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Plan to throw one away</td>
<td>Cache answers</td>
<td>Make actions atomic</td>
</tr>
<tr>
<td></td>
<td>Keep secrets</td>
<td>Use hints</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Use a good idea again</td>
<td>Use brute force</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Divide and conquer</td>
<td>Compute in background</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Batch processing</td>
<td></td>
</tr>
</tbody>
</table>

Figure 1: Summary of the slogans
Wrapping up

Thanks!!!