On the Liveness Properties of the Stellar Consensus Protocol

Giuliano Losa

UCLA
SCP should satisfy the safety and liveness properties of Consensus

Safety:

Validity: an *intertwined* node must not externalize an invalid value
Agreement: *intertwined* nodes must never externalize different values

Liveness:

If we wait long enough, all *intact* nodes should externalize a value
SCP is not live under eventual synchrony if some nodes are malicious

<table>
<thead>
<tr>
<th>Asynchronous</th>
<th>Eventually synchronous</th>
</tr>
</thead>
<tbody>
<tr>
<td>Crash-stop</td>
<td>Malicious</td>
</tr>
<tr>
<td>Malicious</td>
<td>Crash-stop</td>
</tr>
<tr>
<td>Intertwined are safe</td>
<td>Intertwined are safe</td>
</tr>
<tr>
<td></td>
<td>Intact are live</td>
</tr>
<tr>
<td></td>
<td>with probability 1</td>
</tr>
</tbody>
</table>
SCP is not live under eventual synchrony if some nodes are malicious

<table>
<thead>
<tr>
<th>Asynchronous</th>
<th>Eventually synchronous</th>
</tr>
</thead>
<tbody>
<tr>
<td>Crash-stop</td>
<td>Crash-stop</td>
</tr>
<tr>
<td>Malicious</td>
<td>Malicious</td>
</tr>
<tr>
<td>Eventually crash-stop</td>
<td>Intertwined are safe</td>
</tr>
<tr>
<td>Intertwined are safe</td>
<td>Intertwined are safe</td>
</tr>
<tr>
<td>Intact are live with probability 1</td>
<td>Eventually crash-stop</td>
</tr>
</tbody>
</table>

Impossible
SCP is not live under eventual synchrony if some nodes are malicious

<table>
<thead>
<tr>
<th>Asynchronous</th>
<th>Eventually synchronous</th>
</tr>
</thead>
<tbody>
<tr>
<td>Crash-stop</td>
<td>Malicious</td>
</tr>
<tr>
<td>Intertwined are safe</td>
<td>Intertwined are safe</td>
</tr>
</tbody>
</table>

**IMPOSSIBLE**
SCP is not live under eventual synchrony if some nodes are malicious

<table>
<thead>
<tr>
<th>Asynchronous</th>
<th>Eventually synchronous</th>
</tr>
</thead>
<tbody>
<tr>
<td>Crash-stop</td>
<td>Crash-stop</td>
</tr>
<tr>
<td>Malicious</td>
<td>Malicious</td>
</tr>
<tr>
<td>Intertwined are safe</td>
<td>Intertwined are safe Intact are live with probability 1</td>
</tr>
<tr>
<td></td>
<td>Eventually crash-stop</td>
</tr>
</tbody>
</table>

No liveness
SCP is not live under eventual synchrony if some nodes are malicious

<table>
<thead>
<tr>
<th></th>
<th>Asynchronous</th>
<th>Eventually synchronous</th>
</tr>
</thead>
<tbody>
<tr>
<td>Crash-stop</td>
<td>Malicious</td>
<td>Crash-stop</td>
</tr>
<tr>
<td>Intertwined are safe</td>
<td>Intertwined are safe</td>
<td>Intertwined are safe</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Intact are live with probability 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>No liveness</td>
</tr>
</tbody>
</table>

**Impossible**
The Intact Set is a set of nodes that can enjoy safety and liveness

**Whitepaper:** a set $I$ is intact when
- After deleting $V \setminus I$, $I$ is intertwined
- $I$ is a quorum

**New definition:** a set $I$ is intact when
- After deleting $B$, $I$ is intertwined
- $I$ is a quorum
The Intact Set is a set of nodes that can enjoy safety and liveness

Whitepaper: a set $I$ is intact when
• After deleting $V \setminus I$, $I$ is intertwined
• $I$ is a quorum

New definition: a set $I$ is intact when
• After deleting $B$, $I$ is intertwined
• $I$ is a quorum

Note that $B$ may be smaller than $V \setminus I$; in this case the new Intact Set is larger than the old one
The Intact Set is a set of nodes that can enjoy safety and liveness

**Whitepaper:** a set $I$ is intact when
- After deleting $V \setminus I$, $I$ is intertwined
- $I$ is a quorum

**New definition:** a set $I$ is intact when
- After deleting $B$, $I$ is intertwined
- $I$ is a quorum

Note that $B$ may be smaller than $V \setminus I$; in this case the new Intact Set is larger than the old one

Conjecture: no protocol can ensure safety and liveness for a larger set
Eventual synchrony allows implementing synchronous rounds

Synchronous rounds in a crash-stop system
Eventual synchrony allows implementing synchronous rounds

Decide max value seen

Synchronous rounds in a crash-stop system
Eventual synchrony allows implementing synchronous rounds

In SCP, rounds correspond to the ballot counters

Synchronous rounds in a crash-stop system
Malicious nodes may not follow the round structure

In SCP, n4 cannot cause disagreement but can delay a decision forever.
Malicious nodes may not follow the round structure

In SCP, n4 cannot cause disagreement but can delay a decision forever
Classic solution in a closed system: use a round-robin leader

- Statically map rounds to leaders round-robin (e.g. node number i is leader of every ith round out of N)
- Nodes do not accept values not signed by the leader
- Still need to be wary of a malicious leader: cross-check value to ensure safety
- There must come a round in which the leader is well-behaved, which ensures liveness
In SCP, the nomination protocol can achieve the effect of leaders

• Round-robin leader not possible without known, fixed configuration
• Nomination guarantees agreement on a value with non-zero probability; like having a well-behaved leader with non-zero proba.
• Idea: run nomination at the beginning of every round
New phase diagram

- **nominate**
  - valid candidate value $v$
  - set $b.v = v$

- **prepare**
  - ballot counter changes
  - accepted "commit(b)"

- **commit**
  - ballot counter changes
  - confirmed "commit(b)"

- **externalize**
SCP is live if malicious nodes can be identified and removed from slices

In practice, is it worth changing SCP?
Upcoming

• Streamlined theory of slice infrastructures (including new definition of intact)

• New, simpler consensus algorithm