The Costs and Limits of Availability for Replicated Services

Authors: Yu and Vahdat
Presented by: Chris Moran
CAP Theorem

Consistency

Partition

Availability

CP Category
- BigTable
- HyperTable
- HBase
- MongoDB
- Redis
- MemCacheDB
- Scalaris etc

CA Category
- RDBMS
- GreenPlum etc

AP Category
- Dynamo
- CouchDB
- Cassandra
- SimpleDB
- Tokyo Cabinet
- Riak
- Voldemort etc
Significance

• First to quantify availability versus consistency tradeoff
  ○ Internet availability has only become more important since 2001 (EC2 outages anyone?)

• Upper bounds on availability based on network failures
TACT

- Tunable Availability and Consistency Tradeoffs
  - Series of works by Yu and Vahdat
  - System should be able to provide a spectrum of consistency and availability based on application's needs
Error Types

- Numerical Error
  - Maximum number of writes not seen by a replica
- Order Error
  - Maximum weight of writes that have not established commit order at the local replica
- Staleness
  - Maximum amount of time a replica is guaranteed to observe a write by a remote replica
Simple Error Example

(Assume Serialization Order = W1 W2 W3 W4)
Experimental Assumptions

● Model replica failures as singleton network partitions
  ○ Fault events either divide network into two components or merge two into one
  ○ Assume fault load observed by application is not affected by application's communication

● Single data consistency unit system
  ○ TACT is not providing different consistency levels for different data types
Availability

● Function of:
  ○ Consistency
  ○ Workload
  ○ Faultload

● Model using two sets of questions
  ○ $Q_{offline}$ - Set of questions whose answers can be pre-determined
  ○ $Q_{online}$ - Optimal answers depending on consistency, workload, and faultload
Problems

- Search space is exponential
  - Use pruning!
    - Borrow concept of Pareto Dominance from economics to compare and discard strategies
  - Maximize offline work
    - Ignore order error to simplify problem
Upper bound $F(NE, S)$

- **Aggressive Write Propagation**
  - Propagate everything as soon as possible
  - Best possible strategy for consistency, but not practical to implement

- Uses grouping to cut down search space
Upper bound $F(OE)$

- Number of possible serialization orders is factorial with number of writes
  - Again use idea of dominance to restrict search space
- Consider 3 orderings:
  - ALL - all possible orderings
  - CAUSAL - orders compatible with causal ordering
  - CLUSTER - orders where writes accepted by same partition during an interval cluster together
- Use CLUSTER ordering, since search size is smaller and it dominates other two
Sample Faultloads

<table>
<thead>
<tr>
<th>Faultload</th>
<th>Description</th>
<th>Avg. Fail. Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>SAMPLED1</td>
<td>First day of the RMI-ping trace</td>
<td>0.17%</td>
</tr>
<tr>
<td>SIM0_10</td>
<td>Simulated trace</td>
<td>0.11%</td>
</tr>
<tr>
<td>SIM1_00</td>
<td>Simulated trace</td>
<td>1.05%</td>
</tr>
<tr>
<td>SIM5_00</td>
<td>Simulated trace</td>
<td>4.12%</td>
</tr>
</tbody>
</table>

Table 1: This table summarizes the characteristics of faultloads.
Prototype algorithms

- **Primary Copy**
  - Write is committed once reaching primary replica

- **Golding's**
  - Writes are assigned logical timestamps that determine serialization order
  - Nodes keep version vector to figure out when to commit writes

- **Voting**
  - Nodes vote to determine serialization order, voting for first uncommitted write in log
Network Emulation

- **LAN**
  - Majority of eval, use WAN to validate results
  - Does not emulate variable latency, bandwidth, or drop rate

- **WAN**
  - Two 24 hour experiments using 6 nodes running a application
  - One experiment with NE set to 6
  - Second sets both NE and OE to 0, forcing system to use an atomic commit protocol
Availability vs NE

Figure 2: Availability as function of numerical error (SIM1_00).
Availability vs OE

Figure 3: Availability as function of order error (SIM1_00).
Algorithm effectiveness

- **Golding's**
  - Require communication with all nodes in many cases to advance write logs
  - Do not work well with network partition

- **Primary copy**
  - Advantages will increase as # of nodes increases
  - Smaller chance primary copy is unreachable

- **Voting**
  - Aggressive order bounding behaves similarly to aggressive write propagation
Theoretical bounds can be approached with simple protocols!
Availability vs Failure Rate

Figure 8: Availability with different average failure rate.
Availability vs Communication

- Waiting to push writes can reduce communication costs
- Having a large number of writes waiting can cause system to reject writes in event of a partition
- Should push writes when communication good to limit downtime b/c of partitions
Figure 10: Number of messages to maintain order error.
Client Availability

Figure 11: $Avail_{client}$ as function of replication scale with $Avail_{network} = 1 - 1\%/n$. 
Summary

This paper was all about trade-offs

Questions?