Discussion: Minimizing Commit Latency of Transactions in Geo-Replicated Data Stores

Scribed by Jen-Yang Wen
Summary

Feature:
● Low latency serializable transactions on geo-replicated data stores

Technique:
● Timestamps by local loosely synchronized clocks
● Shared Log
● Optimizing average commit latency by linear programming
● Configurable f-resilient

Evaluation:
● On Amazon AWS with 5 geo-replicated data centers
Discussion

Pros:

● High performance
● Novel theory, proof, and protocol
● Flexibility
  ○ Separate serializability from liveness
  ○ Able to manual tuning parameters
● Evaluation on AWS geo-replicated systems
● Use shared log for higher stability
● Extensive analysis
● Well organized
Discussion

Pros:
- High performance
- Novel theory, proof, and protocol
- Flexibility
  - Separate serializability from liveness
  - Able to manual tuning parameters
- Evaluation on AWS geo-replicated systems
- Use shared log for higher stability
- Extensive analysis
- Well organized

Cons:
- Irrealistic assumptions
- Performance sensitive to clock sync.
- Not the best in all the three evaluation aspects
- Experiment only over 10 mins
- Proof not formal
- Liveness and commit latency tradeoff
- Tedious configuration process
- No test under failure
- Focus on average, not tail latency
- Storage overhead of the full copy shared logs
- Limited discussion on Grace Time/ f-value
Questions

- A quick poll: Does the “Proof of lower-bound“ seem formal to you?
- Different servers have different commit speed, a good idea?
- It would be interesting to see how multiple applications running on cloud platform and requiring different average commit latencies can be handled.
- Any additional questions or comments?