NOPaxos

Network Ordered Paxos

Dason Gillespie

Utah Tech Univeresity, Department of Computing

Introduction

- NOPaxos is a distributed consensus protocol that attempts to utilize the network layer to provide message ordering.
- Appends message numbers to packets using a programmable switch.
- All nodes in a replicated group must be decedents of a common switch.

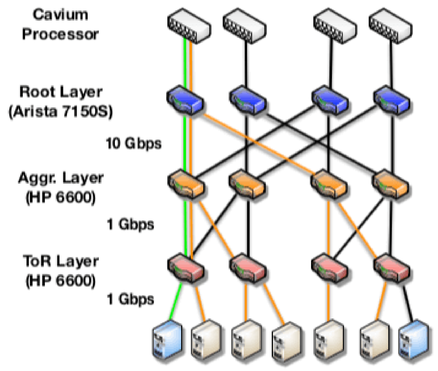

Figure 3: Testbed network topology. Green lines indicate the upward path from a client to the sequencer, and orange lines indicate the downward path from the sequencer to receivers.

Figure 1: Figure from the original paper

- Each node maintains a log of client operations.
- Nodes detect dropped messages from gaps in the appended message numbers.

Objectives

- Recreate the NOPaxos protocol using an end-host implementation that mimics the behavior of a programmable switch.
- Evaluate the performance of the protocol.
- Examine the efficacy of the protocol in the context of a larger system and the possible advantages/disadvantages it may provide.

Implementation

- Wrote an end-host switch using raw sockets and a custom serialized packet utilizing the python package 'pickle'
- Predefined network
- Protocol completed and working under normal operations. Further work needed for fault tolerance
- Client node sends request to switch, switch appends message number and forwards the message to all replicated nodes
- All nodes append client's request to log, additionally, leader applies the requested command.

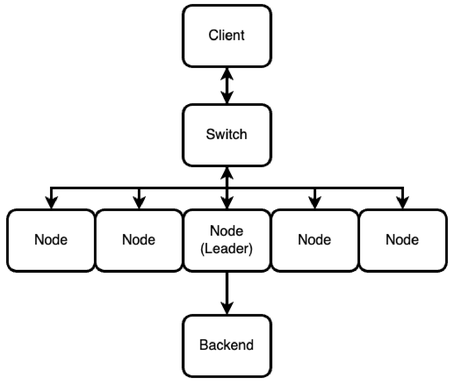

Figure 2: Network Topology

- Custom packet implementation includes various fields used by the switch, the client, and each of the nodes
- Notable fields: command - the command issued from the client that each node will append to their log, sequence - the message number that the switch inserts

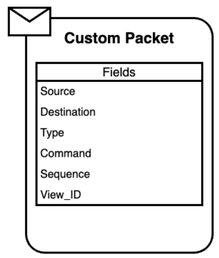

Figure 3: Packet fields

- Additional packet fields would be needed to complete the entirety of the NOPaxos protocol. Included fields are enough to support the protocol during normal operations.

Results/Discussion

- Authors of NOPaxos paper reported high throughput. Close to throughput achieved with unreplicated data. (see figure below)

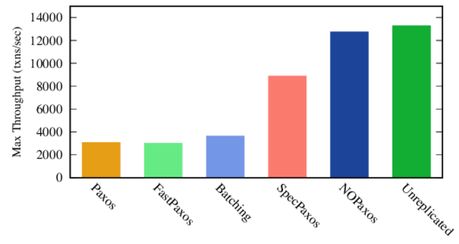

Figure 4: NOPaxos author's results

- Tests ran on current implementation measuring the total time taken for a single round. ie. the time taken from when the client issues the command until it has received a confirmation message back from each node. See figure below
- Important to note that all nodes were running locally on a single machine

Response Time (milliseconds)	
Average	1.054
Max	1.944
Min	0.590

Table 1: Times Taken

Conclusions

- Under normal operations, NOPaxos performs very well, being able to complete most commands in a single round and behaving similarly to the throughput of unreplicated data
- Although it would not match up to a real programmable switch in performance, the use of an endhost switch and a custom serialized packet implementation worked really well and was still able to perform quickly
- The implementation of NOPaxos places certain demands on the network that might not be the best fit for a particular system. ie the necessary use of programmable switches and the requirement that all nodes in a group must be descendants of a particular switch. Also, switch failover could cause delays while rerouting.

References

Just say NO to Paxos Overhead: Replacing Consensus with Network Ordering, Jialin Li, Ellis Michael, Naveen Kr. Sharma, Adriana Szekeres, and Dan R. K. Ports, University of Washington, https://www.usenix.org/conference/osdi16/technical-sessions/presentation/li