COMP4801 Final Year Project

Project Plan

Building Fast Blockchain Applications on an In-datacenter Blockchain Platform

Focus: Network Ordering on Consensus Protocols

Supervisor: Dr. Cui Heming

Mentor: Qi ji

Member: Pang Chun Ho Nelson, BEng(CompSc), 3035568403

Wong Sai Kit, BEng(EngSc), 3035475307
Contents

1. Introduction  3
   1.1 Blockchain  3
   1.2 Application of Blockchain  4
   1.3 Consensus Protocol  4
   1.4 State Machine Replication (SMR) Protocol  5

2. Project Background  6
   2.1 Current Problem: Communication Overhead of Consensus Protocol  6
   2.2 Current Solution: BIDL - Blockchain-powered In-datacenter Ledger  7
   2.3 Further Implementation: HotStuff  8
   2.4 Objective  9
   2.5 Project Scope  9

3. Methodology  10
   3.1 Literature Review  10
   3.2 Modification of HotStuff  11
   3.3 Implementation of Kauri or Pompē  11
   3.4 Evaluation of Consensus Protocols  11

4. Schedule and Milestones  12

5. Reference List  13
1. Introduction

This is a research-based project with a focus on examining the effectiveness of network ordering on consensus protocols. The project is an extension of a blockchain system, BIDL, built by Dr. Cui’s research team. Dr. Cui’s PhD candidate, Ji Qi, is our mentor of the project and a contributor of BIDL. The goal of the system is to execute large amounts of transactions using the latest blockchain technology released in recent years. The building blocks of a blockchain system are introduced below.

1.1 Blockchain

Blockchain is a distributed database for transaction records. Machine participating in the blockchain network is called node. When transactions are sent to node, it will execute the transactions and batch them into a block. The block will be verified by the blockchain network and will be chained to the previous block, forming a blockchain.
1.2 Application of Blockchain

Given the security (transaction record can hardly be tampered with) and transparency (transaction is visible to the permitted member) of blockchain, it can be applied into the trade finance industry where enterprises can trust blockchain, instead of bank, to handle the trade document. In the context of stock exchange, the research on high-performance blockchain-based trading systems is ongoing.

1.3 Consensus Protocol

A blockchain network relies on a number of nodes to execute the transactions. The execution results need to be consistent across a majority of nodes in order to reach a consensus on the result. Therefore, a blockchain network requires a consensus protocol to ensure there is only one consensus reached per request. The consensus protocol is run by a separate group of nodes, other than the node for transaction execution.

Figure 2: Input and Output of Consensus Protocol
1.4 State Machine Replication (SMR) Protocol

SMR protocol is a kind of consensus protocol. Given the same initial state and input, replicas (state machines) should produce a replicated result so that the SMR protocol can collect the results from different nodes and ensure the majority of results are identical for reaching a consensus.
2. Project Background

2.1 Current Problem: Communication Overhead of Consensus Protocol

Figure 4. Ordinary and Network Ordering Workflow

In the ordinary workflow, consensus protocol is responsible for ordering the transactions and ensuring reception of transactions in all consensus nodes. This situation generates complex implementation and significant overhead in the consensus protocol.

To solve this problem, a concept of network ordering on consensus protocol is introduced. Figure 4 shows a new division of responsibilities in different components. Sequencer is a network-level device which can establish order on transactions by stamping a sequencer number on each transaction. Hence, consensus protocol only needs to ensure a reliable reception of ordered transactions in all consensus nodes. This new workflow fundamentally simplifies the implementation of consensus protocols and reduces the overhead (Li, J.L. et al., 2016).
BIDL is a system designed to execute large-scale trading requests. It incorporates performance boosting features proposed in recent years with an attempt to meet the demanding performance requirement of stock exchange systems.

Figure 5. HLF’s sequential workflow

Figure 6. BIDL’s parallel workflow

(1) **Parallel Workflow**

Sequential workflow brings low performance. A typical example of this type of workflow is HLF’s *execute* -> *consensus* -> *validate* workflow as shown in Figure 5 (Androulaki, E. et al., 2018), whereas the *consensus* and *execute* workflows (Phase 2 and 3 in Figure 6) run in parallel in BIDL (Ji Q. et al., 2021).

(2) **Network Ordering**

BIDL adopts a software-based sequencer as programmable switches are not yet available. The sequencer provides the important network ordering feature for reducing communication time in consensus protocol.

(3) **Modular Consensus Protocol**

Since no protocol fits in all scenarios, BIDL is capable of integrating multiple consensus protocols. Up to this date, several consensus protocols have been implemented into testing, namely HotStuff, SBFT, BFT-Smart, Zyzzyva.
2.3 Further Implementation : HotStuff

PBFT and HotStuff are both SMR protocols but with different communication networks. When a client sends a request to the consensus nodes, the request needs to be processed through multiple phases in order to reach a consensus.

In PBFT, whenever a node receives a request from a client, it needs to broadcast the request to peers and wait for their confirmation. Upon receiving confirmations from a majority of peers, the node can confirm that the request is valid. Since every node needs to communicate with each other per phase, as shown in Figure 7, the communication time complexity is $O(n^2)$ which is expensive.

In order to reduce the complexity of communication in PBFT, HotStuff adopts a new communication network which is of $O(n)$ complexity. In HotStuff, the client only sends messages to the leader node (illustrated as Node N1 in Figure 7). Leader node needs to broadcast the messages and then collect votes from peer nodes. Peer node (Node N2, N3, N4) only needs to send confirmation to the leader once per phase. Instead of messaging between every node, in HotStuff, peer nodes listen and talk to the leader node once per phase (Yin, M.F. et al, 2018).

The reduced frequency of communication favours the improvement of throughput. The new communication network of HotStuff is therefore valuable to be integrated in BIDL and evaluate the extra performance enhancement provided by network ordering.
2.4 Objective

First, since the existing HotStuff, one of the consensus protocols in BIDL, is not stable and does not fully comply with the implementation defined in the paper, we need to modify the current implementation of HotStuff running in the system and evaluate our modification.

Second, we need to investigate Kauri and Pompê, the latest consensus protocols published in SOSP 21 and OSDI 20, then choose one of them to implement and integrate it into BIDL. Then we evaluate it with and without network ordering, to investigate the improvement of throughput and latency given by network ordering.

Lastly, the consensus protocols currently have their own interface to receive transactions from the sequencer because the protocols are implemented in different programming languages. This situation generates redundant code across different protocol codebases. To facilitate a simpler protocol implementation, we need to research a high-performance cross-language communication mechanism and then develop a common interface which serves all protocols.

To conclude, the objectives are evaluating HotStuff and a new consensus protocol, and implementing a common packet receiver interface if possible, depending on the progress.

2.5 Project Scope

The project scope is limited to the implementation and evaluation of HotStuff and a new consensus protocol, i.e. Kauri or Pompê. The evaluation goal is to examine the extra enhancement introduced by the network ordering feature in BIDL on these two protocols. Hence, performance comparisons with other consensus protocols will not be conducted.

The overall performance comparison of BIDL with other blockchain systems, i.e. FastFabric, Hyperledger Fabric, and StreamChain, will not be examined because it is not the focus of this project.
3. Methodology

As this research-based project is currently at a nascent stage and involves a considerable amount of time on literature review, the methodology proposed in this project plan does not cover the full details of the implementation and is tentative in nature. Nonetheless, the flow of investigation and development is clear.

3.1 Literature Review

Literature listed below is the prerequisite knowledge for our implementation.

1. All about Eve: Execute-Verify Replication for Multi-Core Servers
2. Hyperledger Fabric: A Distributed Operating System for Permissioned Blockchains
3. ParBlockchain: Leveraging Transaction Parallelism in Permissioned Blockchain Systems
4. Just say NO to Paxos Overhead: Replacing Consensus with Network Ordering
5. HotStuff: BFT Consensus in the Lens of Blockchain
6. BIDL: A High-throughput, Low-latency Permissioned Blockchain Framework for Datacenter Networks
7. Kauri: The quest for scaling BFT Consensus through Tree-Based Vote Aggregation
8. Byzantine Ordered Consensus without Byzantine Oligarchy
3.2 Modification of HotStuff

The new implementation should comply with the algorithm defined in the paper, i.e. algorithm of Event-driven HotStuff as shown in Figure 8.

Algorithm 4 Event-driven HotStuff (for replica u).

```plaintext
1: procedure CREATELEAF(parent, cmd, qc, height)
2: b.parent ← parent, b.cmd ← cmd;
3: b.justify ← qc, b.height ← height, return b
4: procedure UPDATE(b')
5: b'' ← b'.justified.node; b' ← b''.justified.node
6: b ← b'.justified.node
7: // PRE-COMMIT phase on b''
8: UPDATEQC_HIGH(b'.justify)
9: if h'.height > hlock.height then
10:   block ← b' // COMMIT phase on b'
11: if (b'.parent = b') ∧ (b'.parent = b) then
12:   onCommit(b)
13: procedure onCommit(b)
14: if block.height < b.height then
15:   onCommit(b.parent); execute(b, cmd)
16: procedure onReceiveProposal(Msg, (generic, bnew, ...))
17: if bnew.height > height ∧ (bnew extends block ∧
18:   bnew.justified.node.height > block.height) then
19:   height ← bnew.height
20:   SEND(GETLEADER(), VOTEMSG, (GENERIC, bnew, ...))
21:   UPDATE(bnew)
22: procedure onReceiveVote(m = VOTEMSG, (GENERIC, b, ...))
23: if ∃(v, σ') ∈ V[b] then return // avoid duplicates
24: V[b] ← V[b]∪{(v, m.partialSig)} // collect votes
25: if |V[b]| ≥ n - f then
26:   qc ← QC(Σ(v, σ) v, σ ∈ V[b]))
27:   UPDATEQC_HIGH(qc)
28: function onPropose(binf, cmd, qc_high)
29: bnew ← CREATELEAF(binf, cmd, qc_high, binf.height + 1)
30:   // send to all replicas, including u itself
31:   return bnew
```

Figure 8. Algorithm for Event-driven HotStuff

Figure 8 describes the functions which are used by Leader Node and Normal Node to perform a consensus protocol. The current implementation of HotStuff does not fit in Figure 8. Modification details will be discussed in the Interim Report. Before the modification, intensive investigation on the existing HotStuff implementation is required.

3.3 Implementation of Kauri or Pompē

Implementation details will be discussed in the Interim Report, as the implementation of the new consensus protocol is the deliverable of Phase III.

3.4 Evaluation of Consensus Protocols

Our goal is evaluating the performance improvement provided by network ordering on HotStuff and Kauri / Pompē. Therefore, we need to evaluate the protocols in a controlled experiment, i.e. with and without the network ordering feature. Experiment setups will be introduced in the Interim Report. The evaluation of HotStuff is the deliverable of Phase II.
### 4. Schedule and Milestones

<table>
<thead>
<tr>
<th>Time</th>
<th>Work</th>
</tr>
</thead>
<tbody>
<tr>
<td>3 Oct, 2021</td>
<td>Deliverables of Phase I</td>
</tr>
<tr>
<td></td>
<td>- Project Plan</td>
</tr>
<tr>
<td></td>
<td>- Project Website</td>
</tr>
<tr>
<td>10 - 14 Jan, 2021</td>
<td>First Presentation</td>
</tr>
<tr>
<td>23 Jan, 2022</td>
<td>Deliverables of Phase II</td>
</tr>
<tr>
<td></td>
<td>- Modification of HotStuff</td>
</tr>
<tr>
<td></td>
<td>- Evaluation of HotStuff</td>
</tr>
<tr>
<td></td>
<td>- Interim Report</td>
</tr>
<tr>
<td>18 Apr, 2022</td>
<td>Deliverables of Phase III</td>
</tr>
<tr>
<td></td>
<td>- Implementation of New Protocol</td>
</tr>
<tr>
<td></td>
<td>- Evaluation of New Protocol</td>
</tr>
<tr>
<td></td>
<td>- Final Report</td>
</tr>
<tr>
<td>19 - 22 Apr, 2022</td>
<td>Final Presentation</td>
</tr>
<tr>
<td>4 May, 2022</td>
<td>Exhibition of Project</td>
</tr>
<tr>
<td>31 May, 2022</td>
<td>Completion of Project</td>
</tr>
</tbody>
</table>
5. Reference List


