FACULTY OF ENGINEERING

A Simple Blockchain

Distributed Systems

Presented by
1- Rita Samir [25]
2- Alaa Shehab [1]
3- Marina Zakaria[47]
4- Bassant Ahmed [18]
5- Fatma Mohamed[43]

5/19/2020

This is our report about our work on the problem"Simple Blockchain". It explains the problem definition and what is the goal to achieve with our solution, Algorithms we used to get our solution, How we implemented those algorithms in java programming language, Our result and the conclusion that is a summary and comment on our work.

Table of Contents

1 Problem Definition

- The problem is to implement a simple blockchain based on a peer to peer network, It is required to implement it by two variants :
    - POW
        - The network has two types of nodes :
            - Miners : Can receive transactions and mining that transactions to make a new block and can broadcast that block to every node in the network.
            - Clients : Can read the transactions file, receive blocks and check its verification.
    - BFT
        - The network has two types of nodes :
            - Primary : Can read the transactions file, receive transactions and blocks and mining that transactions to make a new block and can broadcast that block to every node in the network.
            - Nodes : Can receive blocks and check its verification.
- Transactions and Blocks have a specific format
- Security verification by keys is required.

2 Algorithms

- **POW**
  - The block has a nonce in its header whose initial value = 0.
  - We have a hardness number → number of zeros in the beginning of the hash.
  - We calculate the hash of the block and then solve the block and that process is :
    - Increment the nonce
    - calculate the hash again

    → Until The miner gets the required hardness.

- **BFT**
  - The primary node reads the transactions file and generates blocks from it.
  - The primary node broadcasts that block to all other Nodes in the network.
  - Every other node receives that block and sends that block again to all other nodes except the node that the block came from.
  - Repeat the last step for all nodes until all nodes in the network will have n (the number of nodes in the network-1) blocks
  - Every node will take a majority voting over n blocks to add that block into its chain or not

- **Peer to peer network**

  Each peerNode has a hostname, portNumber and ID.

  Each peerNode acts like a client(sends messages to all others peerNodes in the network) and a server(receives messages from other peerNodes).

  Thread: ClientHandler

  thread acts like a server in peerNode waits for others peerNodes for their messages

  if the message is a block

  add it to the blockList

  else if the message is a transaction

  add it to the txList

  Method : startConnection(String ip, int port)

  start connection with other machine to start sending messages to them

  Method : sendMessage(String msg)

  send message msg over the socket that has been created with start connection method

method : readTransaction(String filename)

reads the database file line by line and converts each line to transaction string to send it to all peerNodes in the network using sendMessage method.

- **Transaction validation**

  Method : validTransaction

  validSum <- sum(Input) >= sum(output)
  validInput <- check map of outputs to ensure no double spending
  validSignature <- verify signature in input with payer pk
  validPK <- verify pk of payer is that in the referenced previous tx

  IF validSum && validInput && validSignature && validPK **THEN**
  return valid transaction

- **Blockchain**

  Method : addBlock

  IF block.hash == newBlock.previousHash **THEN**
  add newBlock to block's children
  return true;

  FOR (Blockchain child : chain)
  child.addBlock(newBlock)

  Method : getChainHead (int treeDepth)

  IF chain.isEmpty **THEN**
  return this;

  For (Blockchain child : chain)
  depth <- child.getDepth
  IF depth == treeDepth **THEN**
  current <- child.getChainHead()
  head <- leastTimeStamp(head, current)

4 Implementation

- Environment
    - Java Programming
    - Windows/Linux Platform
- Machines Specifications
    - Architecture: x86_64
    - System type: 64-bit operating system
    - Core(s) i5
    - CPU family: Intel
    - Core Generation: 8th Gen
    - CPU GHz: 1.6
- Network Type
    - Wireless LAN
- Data sets
    - transaction dataset provided on piazza
- Runs
    - Number: 12 run performed
    - Min clients: 1
    - Max clients: 3
    - Miner : 4
    - Transactions : 100,000

5 Results

- Using POW we can obtain those results
    - Block size, 200 TXs

Hardness	3	5
Average message complexity per block	1502	3981
Number of stale blocks	4	4
Average time to mine a block.	42 ms	2259 ms

- Block size, 400TXs

Hardness	3	5
Average message complexity per block	2200	2674
Number of stale blocks	3	2
Average time to mine a block.	65 ms	3346 ms

- Block size, 800TXs.

Hardness	3	5
Average message complexity per block	8930	9969
Number of stale blocks	3	3
Average time to mine a block.	72 ms	3528 ms

Note :

- With the last run (Hardness = 10), it took more than 4 hours and to mine only 1 (nonce value exceeded integer) so we couldn’t record the results for this run

-BFT test :

Number of nodes	4	5	6
Message complexity per block	501047	1753480	9877503
Average time to agree on a block	45 ms	93 ms	127 ms

6 Conclusion

We try to find a solution for the problem “Simple Blockchain” with peer to peer network implementation with two variants of the implementation POW, BFT.