CS 765 - Introduction to Blockchains, Cryptocurrencies and Smart Contracts

Assignment - 1 Report

CHIRANMOY BHATTACHARYA
22M0744

HEMANTH NARADASU
22M0777

SAI KUMAR ATLURI
22M0745

1) What are the theoretical reasons for choosing the interarrival time between transactions from exponential distribution?

Δ is the small time Interval and probability of mining of a block in time Δ is βΔ where β depends on the hashing power, Let $I(I = n\Delta)$ be the time interval. Probability to mine a block is time interval I can be drawn from an exponential distribution with mean $\frac{1}{\beta}$ show in below equations

$P[I = n\Delta] = (1 - \beta\Delta)^{n-1}\beta\Delta$ ← Geometric Distribution

$P[I \succ n\Delta] = (1 - \beta\Delta)^{n}$

Let $n\Delta = x$

$P[I \succ x] = (1 - \frac{\beta\Delta}{n})^{n}$ when $n \to \infty$ Geometric distribution behaves like a exponential distribution

$P[I \succ x] = e^{-\beta x}$

2) Why is the mean of dij inversely related to cij? Give justification for this choice.

cij is the link speed of the node, when the link speed is high, time required to buffer the packet in the queue is reduced and packets spend less time in the queue thus the average queuing delay(dij) is less.

3) Reason for choosing a particular mean T_k to get the Random Block Generation time from exponential distribution?

If the T_k is small then the large no of blocks will be generated in the and there is a high probability of forks happening and no of transactions per block will also decrease,and if the T_k very large forks rarely happens and number of blocks generated will be less, then the transaction will be stacked in pending transaction pool. So T_k has a significant effect on forks and the number of the

transactions in the block so $\mathbf{T_k}$ should be balanced in the context of all these effects.

Interarrival time(***I)*** = 100s in our since with less than that value we found forks in the block chain tree so to minimize the forks we have chosen *I* to greater than 100s.

PROGRAM OUTLINE

1) **Class - Node** :

   This class contains the miner information and the data stored in this mentioned below.

   a) Miner ID
   b) Network Graph
   c) BlockChain(List of Block pointers)
   d) Fast Cpu(Boolean value that stores weather miner has fast cpu or slow cpu)
   e) Fast Link(Boolean value that stores weather the miner is connected to fast link or slow link)
   f) List of Block Arrival Times
   g) Balance of the Miner

2) **Class - Block** :

   Block class holds the following data

   a) Miner ID (ID of the miner who mined the block)
   b) Block_id (Unique block Id)
   c) prevBlockId (It contains the pointer to Previous block it is pointing to)
   d) List of Transactions (List of the transactions the block holds)
   e) chianLen (height of the block in the tree)

3) **Class - Transaction** :
    a) txnID (Unique transaction id)
    b) From_id (Unique node id of the payer)
    c) To_id (Unique node id of the payee)
    d) Amount (value to be paid)

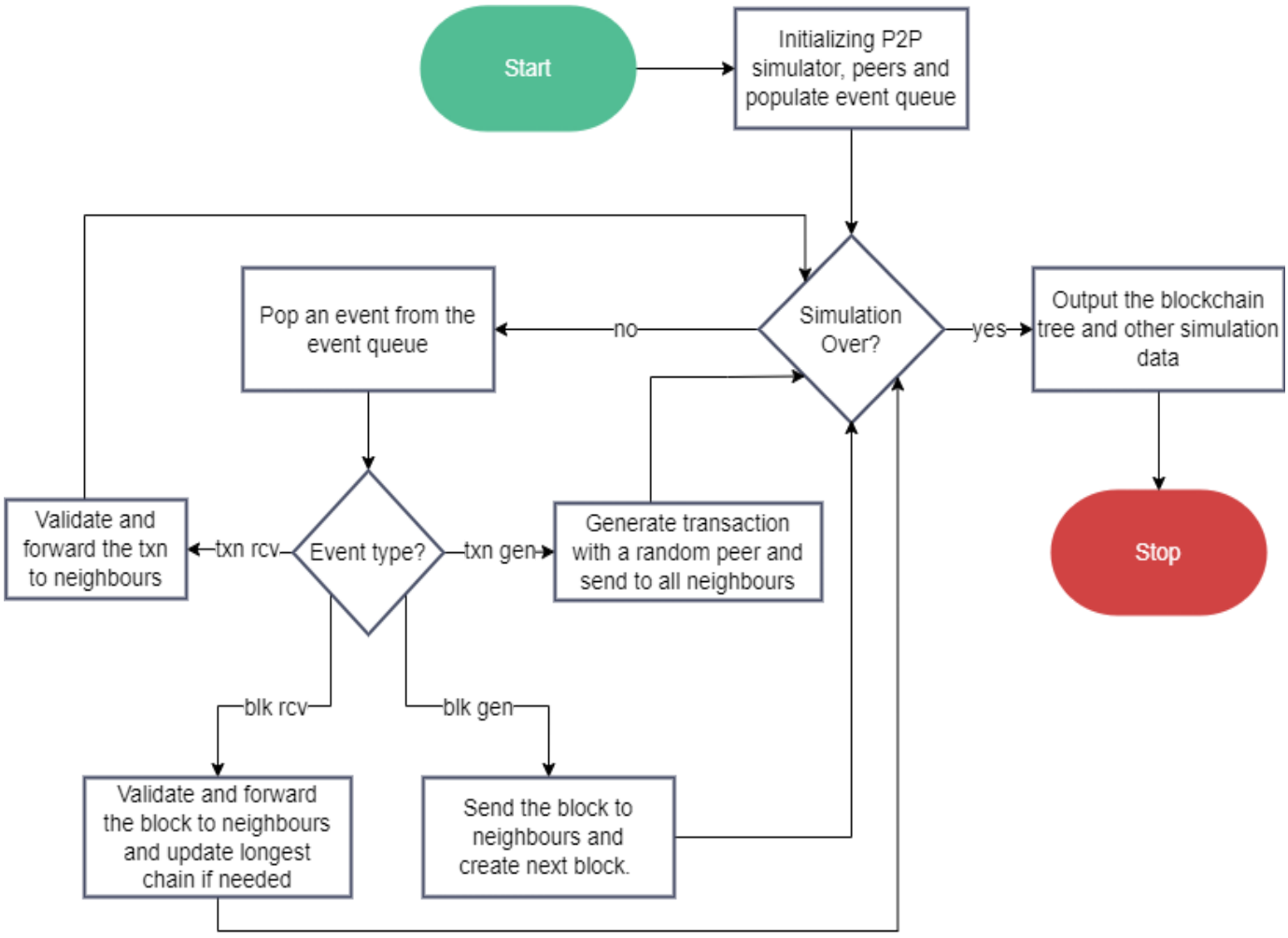
4) **Class - BlockChain :**

This class is part the Node class
    a) minerID (ID of miner block chain belongs to)
    b) lastBlockPointer (Pointer to the last block of the longest chain)
    c) Blocks (List of all blocks in the block chain)
    d) pendingTxns (Transaction pool which has all the pending transactions)
    e) parentLessBlock (Blocks whose parents are not yet received)

5) **Class - Event :**

This class stores all the values needed to process a particular event when popped from the event priority queue.
    a) Type (Type of the event)
    b) TimeStamp (stored in seconds, which is used invoke the event at that specific time)
    c) Node pointer (pointer to the node who invoked the event)

FLOW CHART


Start
Initializing P2P simulator, peers and populate event queue
Simulation Over?
no
Pop an event from the event queue
yes
Output the blockchain tree and other simulation data
Stop
Event type?
txn rcv
Validate and forward the txn to neighbours
txn gen
Generate transaction with a random peer and send to all neighbours
blk rcv
blk gen
Validate and forward the block to neighbours and update longest chain if needed
Send the block to neighbours and create next block.

Observations for different parameters

Following parameters are used in simulation

- Number of nodes (N)
- Percentage of nodes with fast links (Z_0)
- Percentage of node with fast cpus (Z_1)
- Mean for Exponential Distribution to get Interarrival Time between Transaction (T_{tx})
- Interarrival time between blocks (I)

Observations

1) N=10 , Z_0= 20 , Z_1 = 50 , T_{tx} = 60s , I = 120s

Node	CPU Type	Link Type	Total Blocks	Longest Chain Length	Contribution to Longest Chain
0	Fast	Slow	301	301	71 (23.58 %)
1	Slow	Slow	301	301	2 (0.66 %)
2	Slow	Fast	301	301	4 (1.33 %)
3	Fast	Slow	301	301	103 (34.22 %)
4	Slow	Fast	301	301	4 (1.33 %)
5	Slow	Slow	301	301	1 (0.33 %)
6	Slow	Slow	301	301	0 (0 %)
7	Slow	Slow	301	301	3 (1 %)
8	Slow	Fast	301	301	2 (0.66%)
9	Fast	Slow	301	301	110 (36.54 %)

No Forks were created above the parameters since the Interarrival time between blocks is more. The above table clearly shows that fast miners have

contributed more blocks to the final chain and slow miners only contributed less number blocks since fast miners are ten times faster than the slow miners this is very much expected behavior.

2) N=10 , Z_0= 50 , Z_1 = 50 , T_{tx} = 1000s , I = 1s

Node	CPU Type	Link Type	Total Blocks	Longest Chain Length	Contribution to Longest Chain
0	Slow	Fast	12986	12912	42 (0.32 %)
1	Fast	Slow	12986	12912	772 (5.94 %)
2	Slow	Slow	12986	12912	63 (0.49 %)
3	Slow	Fast	12986	12912	57 (0.17 %)
4	Fast	Fast	12986	12912	188 (1.45 %)
5	Fast	Slow	12986	12912	1025 (7.89 %)
6	Fast	Fast	12986	12912	933 (7.18)
7	Fast	Slow	12986	12912	5701 (44.15 %)
8	Fast	Fast	12986	12912	4070 (31.34 %)
9	Slow	Slow	12986	5959	60 (0.46 %)

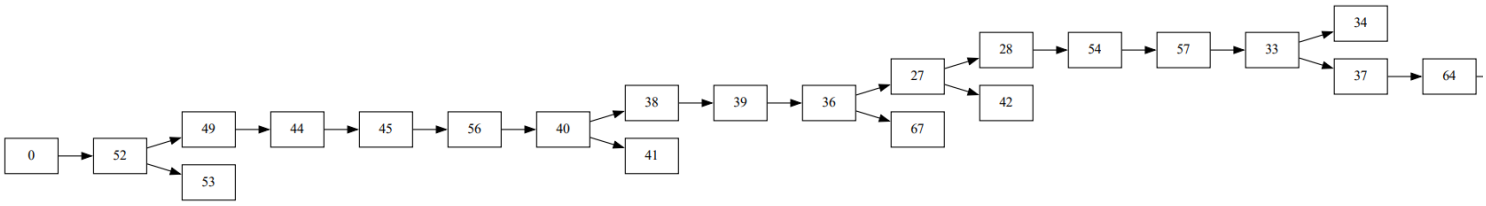
In this observation Inter arrival time between the block generation(*I)* is chosen to be less , to observe the number of forks happening to the block chain below table shows number of forks their lengths and no of time each length of that fork is happening in the chain we can see in the below table that there are more forks of length 1 and also as expected fast miners mine higher share of the blocks present in the blockchain.

Fork Length	Count
3	1

2	3
1	65

Blockchain tree with low block interarrival time:

Number of nodes = 10
Fast CPU node percent = 25%
Fast Link node percent = 40%
Block interarrival time = 5 seconds
Mean time to generate transaction = 60 seconds each



Conclusions

- If the inter arrival time is less then the number of forks in the blockchain are increasing.
- Miners with high CPU power (high hashing power) are significantly mining a very high percentage of blocks in the longest chain.