Blockchain Simulator

A Project for Course “Introduction to Blockchain”

Naveen Tiragati

Masters Student, Computer Science

University Of Missouri –Kansas City

Outline:

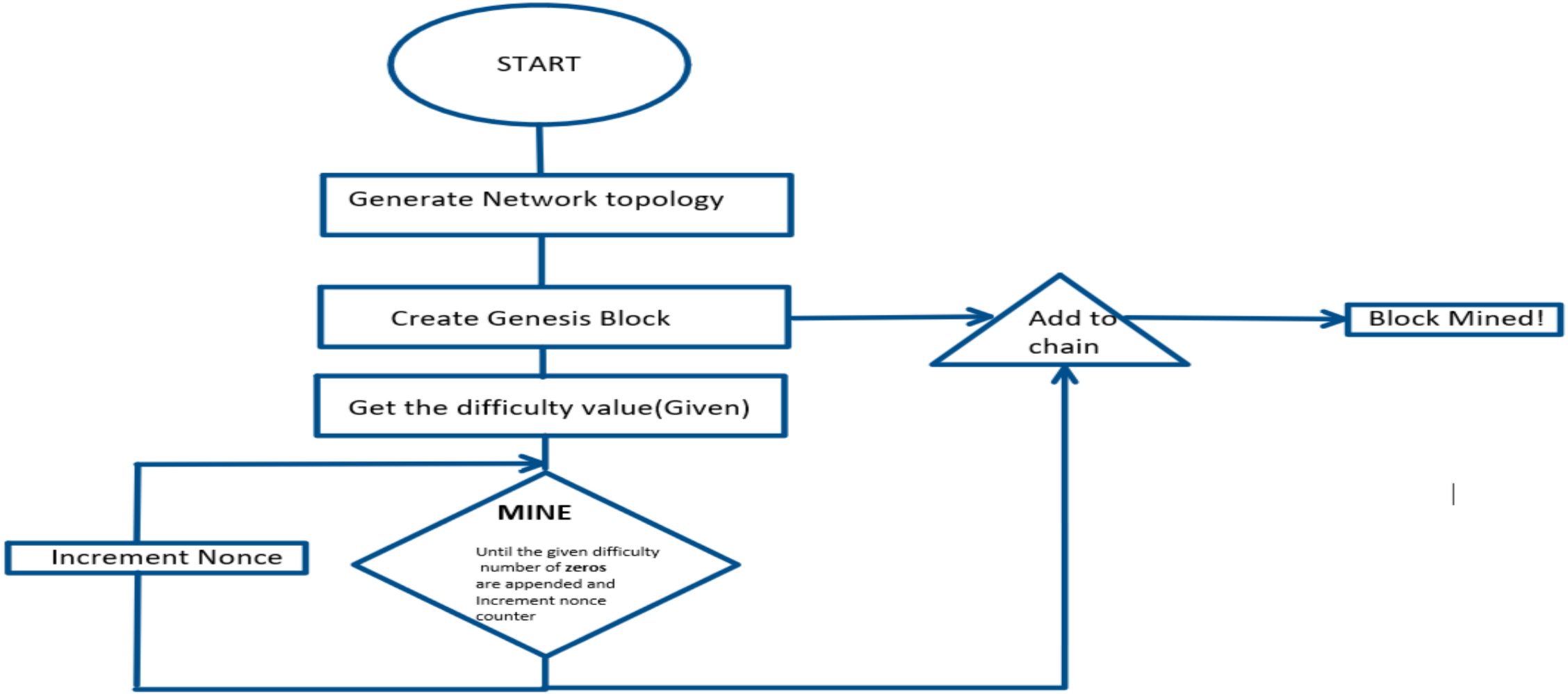
- GIT Repository
- My Project Flow Chart
- Introduction
- Requirements
- Network Topology (Linear Graph)
- Block Data Structure.
- Block Validation
- Difficulty (Mining)
- My Project Flow Chart(Again)
- Results
- Future Work

GIT Repository

Entire Code along with this PPT is secured in the following Git Repository:

https://github.com/naveentiragati/bcs

My Project Flow Chart



Introduction

Developed a blockchain simulator which can simulate network generation, block generation and validation. This simulator can be used for better understanding of the concepts that we learnt throughout the course.

For a given input of number of nodes our simulator generates simple P2P Network, generates genesis block and starts mining and validates the entire chain for false blocks and tell if the chain is valid or not.

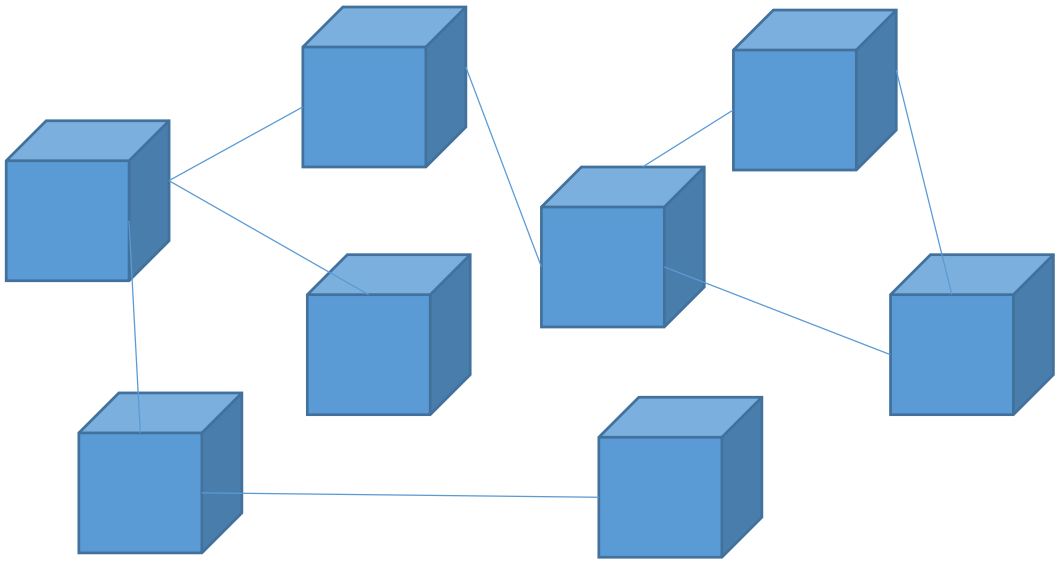
Requirements

Coding Language : JAVA 1.8

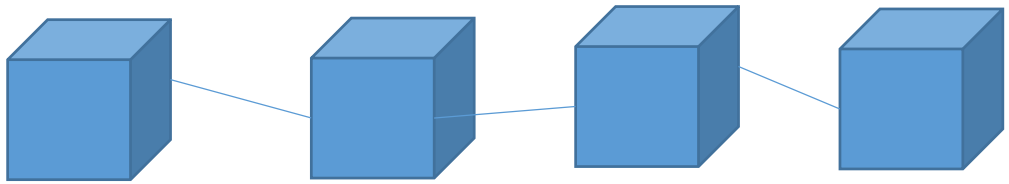
IDE used : Intellij

Network Topology (Linear Graph)

We know that the Blockchain runs on a peer to peer network. To simulate the peer to peer network we used a linear graph topology.

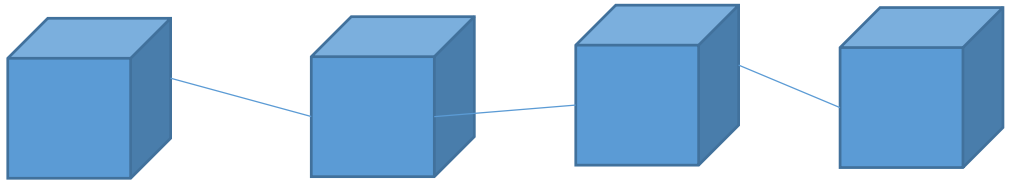


Actual P2P Network



Simulated Simple P2P Liner Graph topology

Network Topology (Linear Graph)



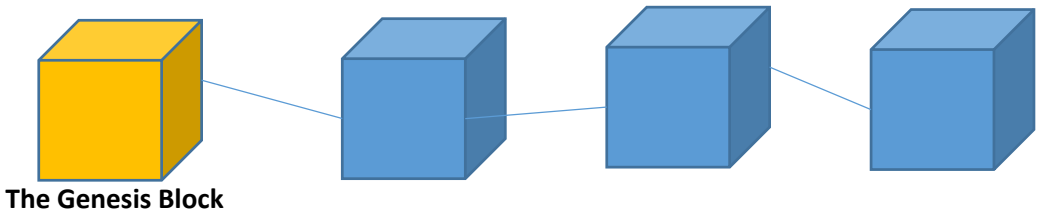
Simulated Simple P2P Liner Graph topology

Output of the linear graph generated:

```
"C:\Program Files\Java\jdk1.8.0_301\bin\java.exe" ...
Network Topology being generated!!..
The linear Graph Topology(Network topology):

[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15]
```

Network Topology (Linear Graph)



Simulated Simple P2P Liner Graph topology

The Genesis Block


```
{
  "EncryptedHash": "000034e5eda322676aacd6eb1c1fdd172b7efe5005ce044ba739897cd59a499a",
  "previousBlockHash": "0",
  "data": "Contains Data of all Transactions",
  "timeStamp": 1638993609225,
  "nonce": 42661
},
```


The Previous Hash value for the Genesis Block is 0

Block Data Structure

Now the each mined block should be stored in some form of data structure in order to access the transaction history, data, time stamp etc. and also we need to verify of the new block mined is valid is or not.

We used **ArrayList<>** in JAVA to implement this.

It is the simplest form of data structure which is backed by the basic arrays. An ArrayList, or dynamically resizing array, allows you to have the benefits of an array while offering flexibility in size. You won't run out of space in the ArrayList since it's capacity will grow as you insert elements.

More clear explanation at : https://medium.com/@Emmanuel.A/data-structures-arraylist-239a5b39cd7e

Block Data Structure

Output of Blockchain Structure:

```
The block chain Data Structure:
[com.bcsimulator.Block@3764951d, com.bcsimulator.Block@4b1210ee, com.bcsimulator.Block@4d7e1886, com.bcsimulator.Block@3cd1a2f1,

Blockchain is Valid: true
```

The **Arraylist** give us a serialized java objects which are not human readable

In order to **pretty print** those we use Gson library which converts serialized java objects to **Json and pretty print** them.

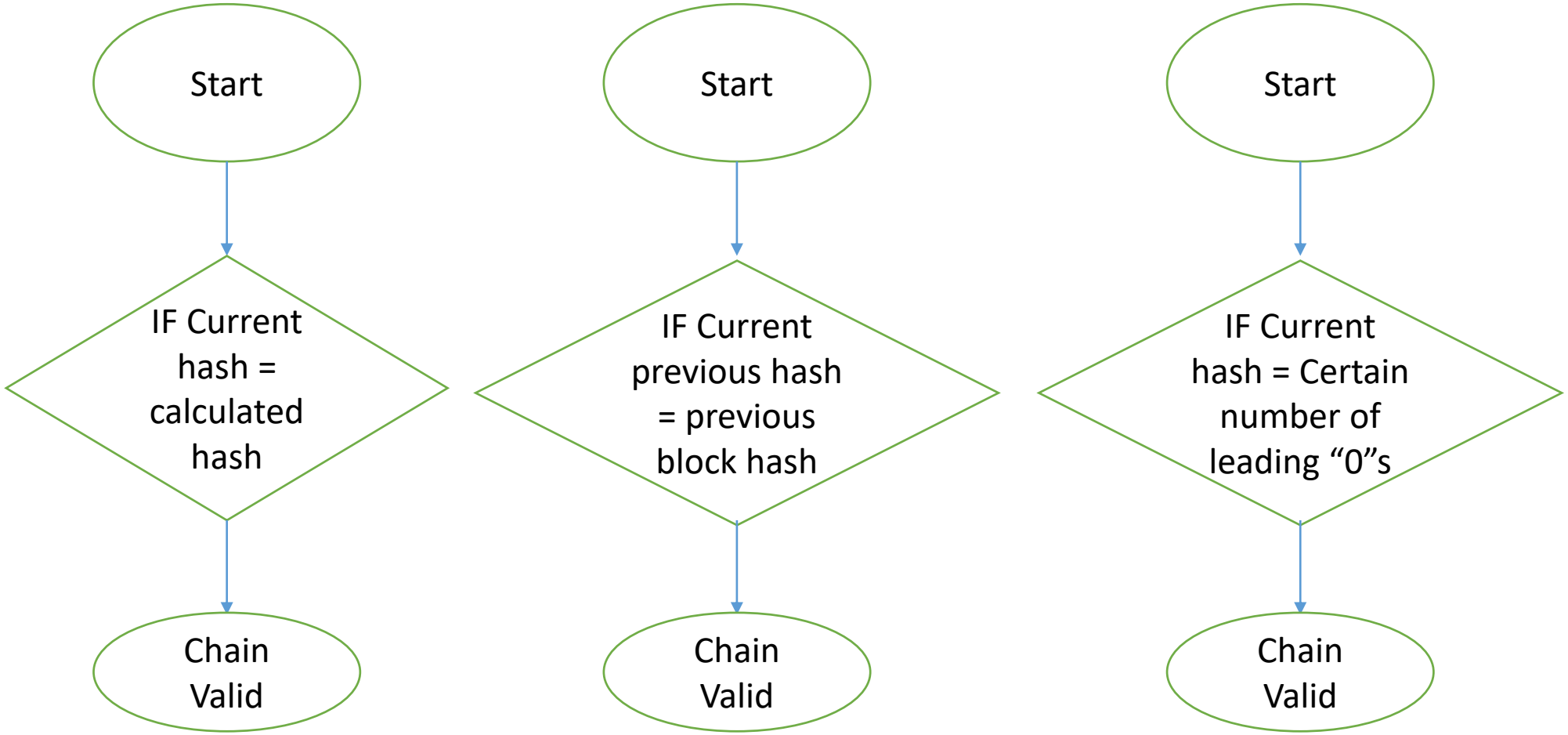
Block Validation

The Block Validation is implemented using multiples ways in order to make sure the entire chain of transactions are valid.

1) If the new block generated is having certain leading number of zeros as defined.
2) By checking if the hash of the previous block and the value of previous hash in the current block for entire chain of transactions if correct or not.
3) By Recalculating the hash values by looping through entire history and comparing with the existing hash values.

Block Validation

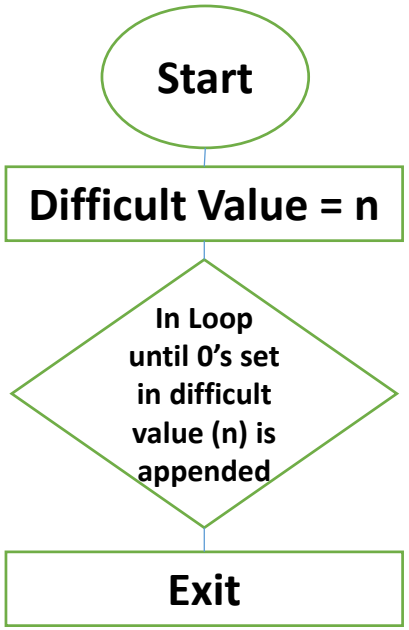
Pseudo Code for validation



Difficulty (Mining)

In POW the block is generated when we find the correct nonce value such that it is less than the target!

Also the target is flexible to change(In our code it is adjusted using difficulty value)

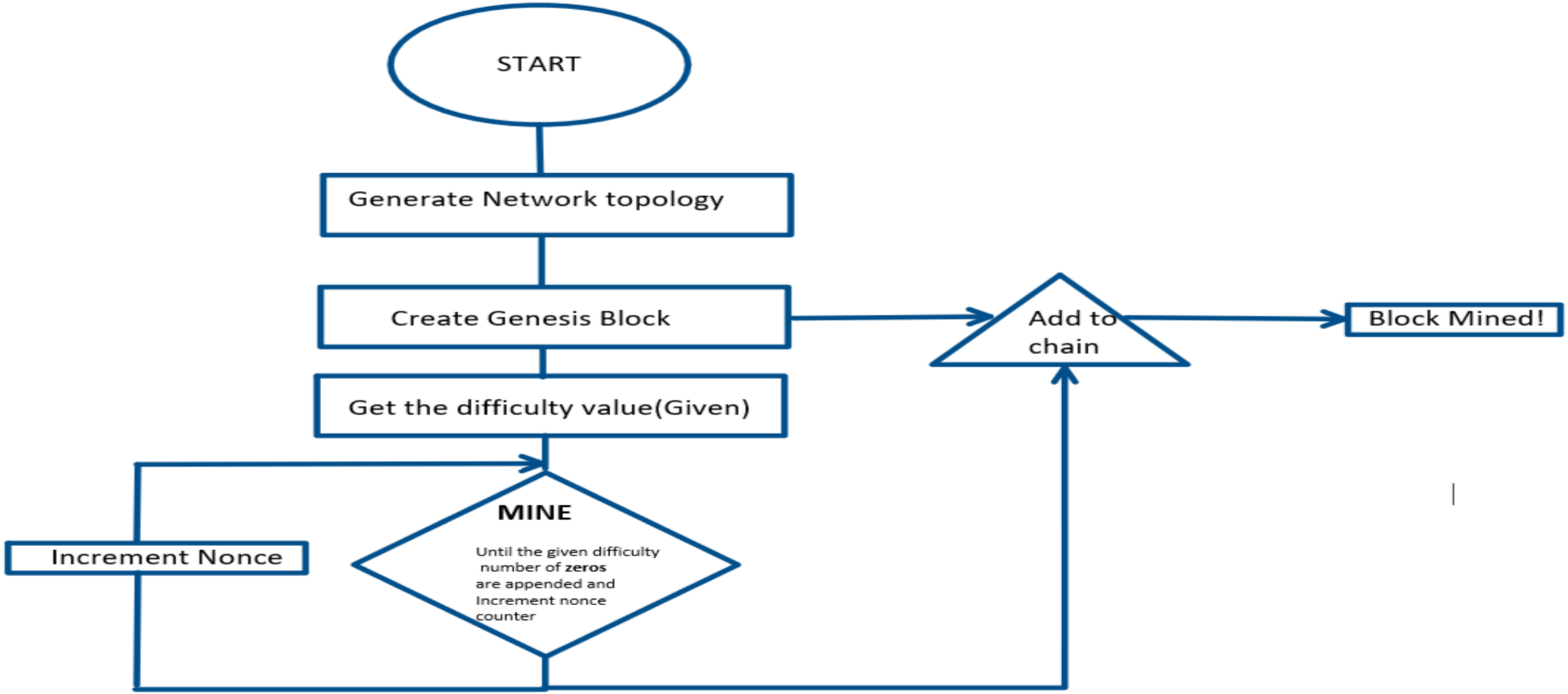


Difficulty (Mining)

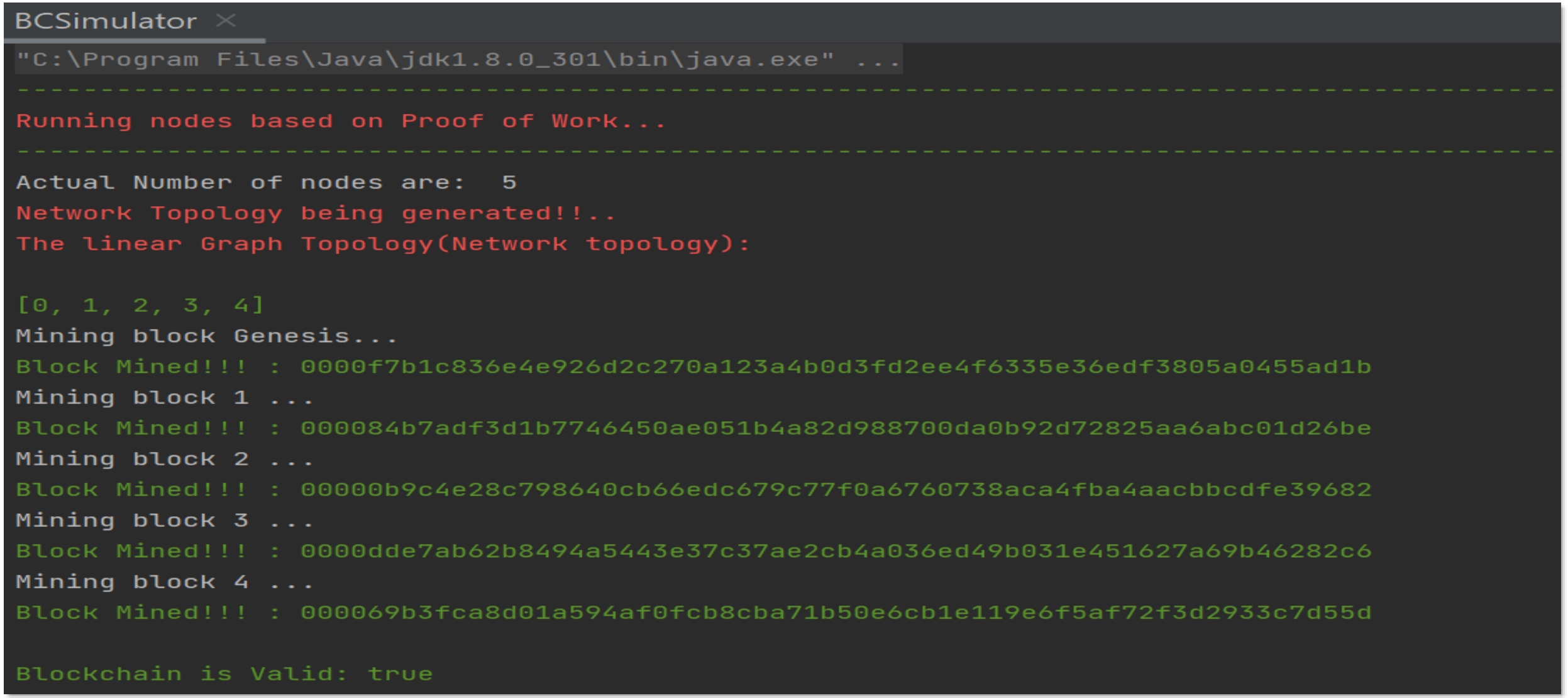
calculatedEncryptedhash = SHA 256(previousBlockHash + data + nonce)

Nonce is the value obtained from mining. Like explained in previous slide the nonce value is incremented until we get the difficulty number of zero’s

My Project Flow Chart



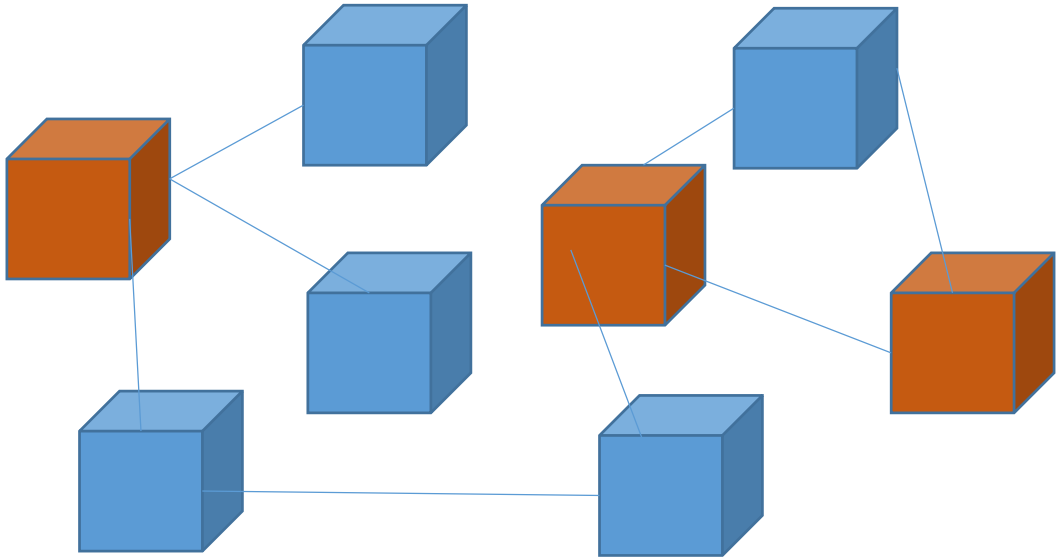
Results:



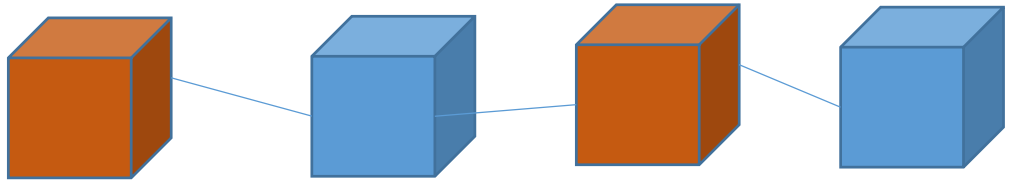
Using POW

Implementing DPOS

Delegated Proof of Stake(DPOS) is evolution of Proof of stake. Here we have delegates instead of miners. The way the delegates selected varies from one proof of stake mechanism to other. Once the delegates are determined they will validated the block and it to the chain of blocks.



DPoS depiction of Actual P2P Network



DPoS depiction of Simulated Simple P2P Liner Graph topology

Results:

```
BCSimulator
----------------------------------------------------------------------------------------------------
 Running nodes based on Deligated Proof of stake...
----------------------------------------------------------------------------------------------------
Actual Number of nodes are:  3
Deligate Number of nodes are:  2
Network Topology being generated!!..
The linear Graph:

[0, 1, 2]
The linear SubGraph of elected deligates:

[0, 2]
Trying to Mine block Genesis...
Block Mined!!! : 0021aa12b50b80b9b44ce88eae508c429797bff74358144bc8a7328b0ecb1f57
Elected Deligated  0 Trying to Mine a Block:
Block Mined!!! : 005c2289a14a74b43ed137cd223285df11ac9298a414d11fd519e37c6cccfe56
Elected Deligated  2 Trying to Mine a Block:
Block Mined!!! : 00c01d09fc020dbfc8232a01f67802ed0ecae99f5d8c10c9c61972291ab692ed
Block mining done for given nodes
Verifying if the chain is valid...

Blockchain is Valid: true
```

Using DPOS

UMKC

Future Implementations:

- Complicated network topology.
- Average extrapolated aggregated propagation time of different consensus and compare them.
- Implementing coin age based selection of delegates instead of random selection for DPoS.

Thanks to Professor and Teaching Assistant

Baek-Young Choi, Ph.D., Professor
Department of Computer Science and Electrical Engineering

Manasa Leela Gummadavelly, Ph.D., Student, Teaching Assistant
Department of Computer Science and Electrical Engineering

Your Knowledge, skills and insights have helped me learn new technology quickly. I Will be grateful for considering an extra credit Project for this course. ☺

End of Presentation ☺