

COP5615 – Project 4.2

Neel Rami, UFID: 7712-3151

Ma Haodi, UFID:7719-2198

Email: nrami@ufl.edu , ma.haodi@ufl.edu

Dec 13, 2018

1. Introduction:

The goal of this (and the previous) project is to implement Bitcoin and (part 2) to build a large simulator to determine its behavior.

In this part, we accomplish the following goals:

- 1) Finish the distributed protocol.
- 2) Implement a simulation with at least 100 participants in which coins get mined and transacted.
- 3) Implement a web interface using the Phoenix that allows access to the ongoing simulation using a web browser (use the matching JavaScript library that allows Phoenix messages to be received in the browser). For charting, we Charts.js library. As part of the simulation, we manage to capture various metrics and send them via Phoenix to the browser.

Assumption:

- The number of processes is 100 and is hard coded. But the project can still run for any number of processes.

Requirement to run the project:

- The machine that the code is tested on should install phoenix.
- Change the password from neel@123 to the password of the postgresql on which the code is tested.
- Change the password in the dev.exs file which is located in the config folder.

```
# Configure your database
config :example, Example.Repo,
  adapter: Ecto.Adapters.Postgres,
  username: "postgres",
  password: "neel@123",
  database: "example_dev",
  hostname: "localhost",
  pool_size: 10
```

- During the transaction, inputs should be provided as numbers and nothing else.
- In the transaction functionality, do not enter decimal values for amount and transaction fee and also do not enter random strings such as “asdf” etc in any of the four fields
- Sender and Receiver can take values between 1 to 100 and Transaction Fee can take 0,1,2,3,4.

2. Functionalities Implement:

- 1) **Basic functionalities:** 2 Functionalities are implemented on the webpage.
 - **Mining:** By clicking this button a mining process will be started and the result will be shown by a table as well as chart.
 - **Transaction:** By clicking this button, two miners will be randomly chosen and transact a random amount of coins between each other. The result will also be shown in a table as well as chart.

3. Implementation Details:

- 1) **Base58.ex:**
 - This file contains functions that encode given data to base58.
- 2) **Base58Check.ex:**
 - This file contains functions that encode given data to base58Check.
- 3) **Blocks.ex:**
 - This file writes the abstraction of a block. And contains various methods for the task of proof of work and block validation.
- 4) **Bitcoin.ex:**
 - Endpoint of the program.
- 5) **Chain.ex:**
 - This file contains various GenServer methods for the task.
- 6) **Mempool.ex:**
 - This file implement the functionality of mempool.
- 7) **MerkleTree.ex:**
 - This file contains methods to build a Merkle tree.
- 8) **MerkleTreeNode.ex:**
 - This file defines the abstraction of the node of Merkle tree.
- 9) **TransactionInput.ex:**
 - This file defines the abstraction of the transaction input.
- 10) **TransactionOutput.ex:**
 - This file defines the abstraction of the transaction output.
- 11) **Transactions.ex:**
 - This file defines the abstraction of the transaction and various methods to build a transaction.

12) UTXO.ex:

- This file defines the abstraction of UTXO and implement the functionality of calculating balance.

13) Wallets.ex:

- This file defines the abstraction of the wallet and contains methods to sign and verify transactions.

14) WalletFunctions.ex:

- This file contains utility functions for implementing the functionality of wallets

4. Instruction to run the project

To start your Phoenix server

1. Install dependencies with ``mix deps.get``
2. Create and migrate your database with ``mix ecto.create`` and change postgres details as stated above
3. Install Node.js dependencies with ``cd assets && npm install``
4. Start Phoenix endpoint with ``mix phx.server``
5. If hex is not available install using `"mix archive.install`
`https://github.com/phoenixframework/archives/raw/master/phx_new.ez"`
6. Now you can visit [``localhost:4000``](`http://localhost:4000`) from your browser.

5. Working of the project:

There are two buttons on the website page:

- **Mining:** When clicking on the mining button, a block will be mined and the information will be displayed in a mining table and chart below.
- **Transaction:** When clicking on the button, the user needs to provide four parameters: the process number of the transaction sender, process number of the transaction receiver, amount of this transaction and the transaction fee. Then, a mining process will be started and a new block will be mined and the information will also be displayed. Besides, the transaction information will also be displayed in a transaction chart below.

There will be 4 charts on the website page:

- **Chart 1:** Nonce for every block
- **Chart 2:** Number of blocks up to the present time

- **Chart 3:** Number of transactions mined up to the present time

6. Result

Bitcoin Mining Project - Google Chrome

localhost:4000

Block Table

Block Height	Block Hash	Nonce	Mined by
6	00004bf92acc4d92e801c2e2f1165e748fc81356c3827ba6584dec4b14923c8	16373	44
5	0000cda0ded5066bc4f0e35c9ac1a8544a6e33a838c9d6c44dcd344b32e8d3	44372	99
4	0000fa642dc4d684c40b1d0f91624f0c6220e2b9a382f6f268309b20703fd35	63991	25
3	00002f420b76398f5e10be854d0e513ff2adf4bcae0fe548c6b39bbe3a496dd	381	92
2	0000545c40ed3f1f52095c69a59920c97807e154c6e4f19bef0068869c3cc9ef	17708	42
1	00007e7e1fb323bb99cc2cffe41d92c7220680e9661187c12222c7eb64422c83	37157	81
0	00000000019d6689c085ae165831e934ff763ae46a2a6c172b3f1b60a8ce26f	208	1

Transaction Table

Transaction ID	Transaction Type	Sender	Receiver	Amount (in BTC)
----------------	------------------	--------	----------	-----------------

To start you....docx readme_part2.docx Show all

Bitcoin Mining Project - Google Chrome

localhost:4000

Transaction Table

Transaction ID	Transaction Type	Sender	Receiver	Amount (in BTC)
866cafb7114e3520bc133b3f34f91c17f0dbb5dd9f0c7f6adddb3c6b2db05f7f	coinbase	-	44	25
553062cd438ed639caaa1368450e67f9215dbee652b91a050f6b67a481d66831	normal	42	1	5
d61e280fdaf4ad521707fdde7cde71b4f009a85db75ca72218ace581dd0f516	coinbase	-	99	25
1117c4a482ff5c51b9371f61b86f0f1711f361d48f19791d310b28c6efcf69	normal	25	50	10
d4ddcc8dd38329e772ff1293da1bda0a6367178efb1324c1a37a348abc67f99b	coinbase	-	25	25
b2064742097205af7d0f9064e1b28c81a31b5622bf60599cef59d86e2c33fef	coinbase	-	92	25
28e71f2bc05c4ea56d40195172c8fbc2f8604050156f216fac6fa899276c1a8	coinbase	-	42	25
a168c2ee4a0d14888716082cbfd6f304bcb50f7e75e1cfb255c51d62c7708a6f	coinbase	-	81	25
dee756478fc15434063df7f011349797bdbc3923d0ffcddeb8543dc02bb891647	normal	1	3	12
4a5e1e4baab89f3a32518a88c31bc87f618f76673e2cc77ab2127b7afdeda33b	coinbase	-	1	25

Nonce Chart

To start you....docx readme_part2.docx Show all

