README.md

Project 1

Group Members:

1. Adish Someshwar Rao
2. Sanjana Rao Guttalu Prasan

Outline:

We have implemented the Actor Model in Erlang to hash and mine bitcoins with the desired number of leading zeros. We have also implemented the Remote Configuration using client and server machines.

System 1: bitcoin_server.erl

The server is started prior to the client with the number of leading zeros (K) and the number of nodes/actors on each system (N_Nodes) as its arguments. The server spawns N_Nodes (actors) to mine coins and then waits to receive messages either from the spawned actors that a coin is found, which the server will print, or from a client that wishes to participate in the mining. Upon receiving a message from a client, the server further spawns multiple children (actors) to continuously mine for coins on a client. Any number of clients may connect. These actors on the client behave the same as actors on the server and will send a message to the server when a coin is found. Each child actor is designed to continue hashing/mining until the server finds a total of 100 coins.

System 2: bitcoin_client.erl

Client takes in the server node name as the argument and a connection is established with the server after which the server spawns actors on the client to mine for coins.

1. Work Unit:

Every child actor is tasked with generating a random string, computing the hash, and mining for bitcoins. Each actor runs until the minimum leading zero condition is satisfied and the generated hash is less than the target value. Upon finding 100 coins, all the actors are killed. We determined that each actor should take on the complete responsibility of generating the string, hashing it, and checking if its a valid coin as when the number of work units were broken down, the execution time (real time) increased due to the exccess communication latency. Many workers stood idle as the server program would generate a string for each actor (thousands of them) and then send the string to each actor for hashing. Like wise, once a hash was found and sent to the server, the check for valid coin was moved to the child actor as the server spent a large number of time verifying each hash generated by these large number of actors. Thus, the child actor now takes all the responsibility and the server only tells the child actor whether to continue mining or not.

2. Result of Program for Input 4:

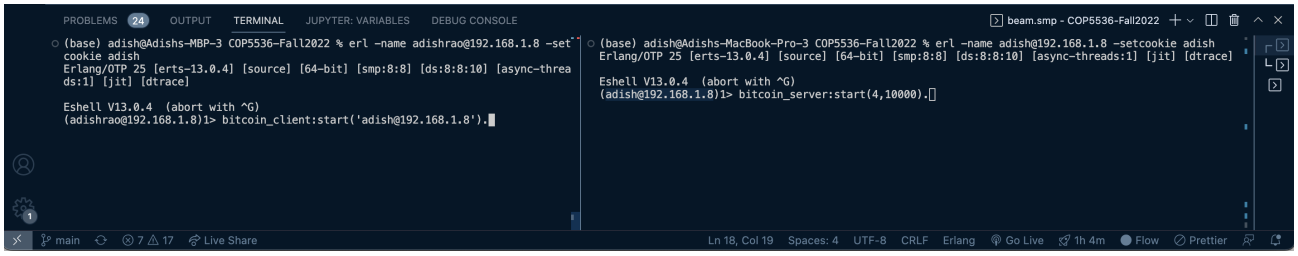

Fig 1: Running the program on 4 nodes.

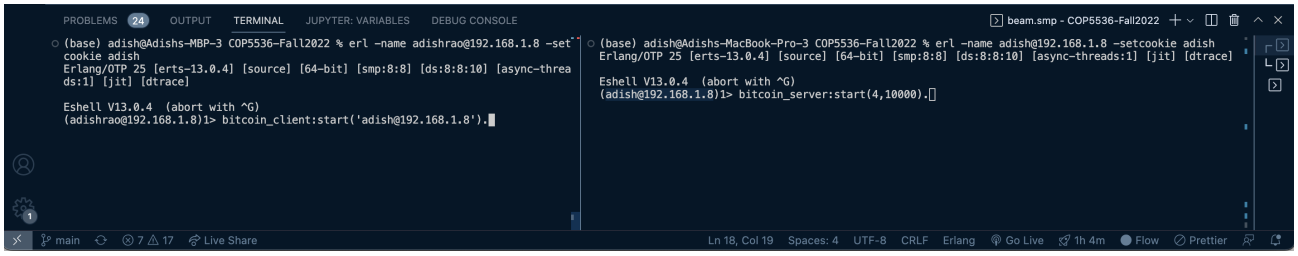

Fig 2: Running the program on 4 nodes (output).

Result for running the program on two complete different systems can be found in section 4 (finding the largest coin) where the local server system and a completely different client system each find the largest coin.

3. Running Time: The runtime was tested for finding only one coin, however, the same runtime would be obtained for any number of coins.

```
Eshell V12.2.1  (abort with ^G)
1> bitcoin_server:start(4,8).
String : 4 "0000"
Coin "adishsomeshwarao;aj04FdFT24jXrIUYPTTNqKuIF2lNo2VQ2Z0dzq9yqtVbmqxCZPrPU+4wKG2cQEdFoKnZGIPUragMkc7s/y3TkI/ddi52fRSe" and
Hash "0000abfc38e9c53918e9754dae207802d048d35d7e9705a885e10d3abc34a999" Found by <0.82.0>

Total clock time: 722.389
Toal CPU time 4897
 CPU time/ Run Time 6.7788961349079235
ok
2>
```

Fig 3: Runtimes when run on four nodes.

Total Clock time = 722.389

Total CPU time = 4897

CPU time/ Run time ratio = 6.778

4. Largest coin found (i.e., the coin with the most number of leading 0's)

We found the largest coin to have seven zeros. The program was run for 1 hour to find 8 zeros, but no coins were found and the program was stopped.

```
(base) adish@Adishs-MacBook-Pro-3 COP5536-Fall2022 % erl -name adish@192.168.1.8 -setcook
ie adish
Erlang/OTP 25 [erts-13.0.4] [source] [64-bit] [smp:8:8] [ds:8:8:10] [async-threads:1] [ji
t] [dtrace]

Eshell V13.0.4  (abort with ^G)
(adish@192.168.1.8)1> bitcoin_server:start(7,5000).
String : 7 "0000000"
Spawning on Node: 'shiv@iMac.local'
Coin "adishsomeshwarao;tD/xq9Hd8L8+zu6iWvvR5585JyqUnaAXTHZZllaB21Yb/gV7phW1r9qCUy/+iMvTDA
IijOfjFWzNuSJbCbIf1LIETuTyTDeJ" and Hash "00000004f42ba1500792ad53398e2f954a9ed826af2b34f
7f538c1b6477ac7b7" Found by <0.3340.0>

Coin "adishsomeshwarao;5A3SMHrN5mhKrcqHOGuIuBIFt/z49aMqj8M+o44ITvxg8siDm2Rrl1XE4AE+IO2JCb
R90mkoPW3YY4HCi/SJNUwDekfFPDmC" and Hash "0000000e7992b5590e6d458a73153ac938a16695f6140f1
39016d082073e6101" Found by <10085.571.0>
```

Fig 4: Coin starting with maximum number of zeros (7).

5. Largest number of working machines

We were able to run the script on four nodes (systems) but the same could be scaled to run on many more.