Project Report – Bitcoin Miner

Ammar Amjad 5992-1730, Mohammad Anas 5981-5998

September 19, 2022

Task:
To make a Bitcoin miner able to mine 0's based on the input that user enters, Outputting the required stats and messages.

Components:
1- Server: That supervises all the worker nodes.
2- Local workers: Workers that fire up after the server is instantiated.
3- Remote workers: Added to server's nodes as they become available. Remote workers communicate their availability to the server.

Instructions:
1- Start server terminal with `erl -sname server`.
2- Complie `c(project).`
3- Run `project:server(n).` where n is the number of bitcoin zeros we want. Not only the server but the maximum number of workers the server can have will also be spawned to start mining on the server.
4- In another terminal, Start worker terminal with `erl -sname w1`.
5- Run `project:starthasher(server@IP).` where server is name of server, IP is the IP address of server.
6- Run multiple instances or same or different machines with above command at step 5.
7- Hash and the string used to find the hash, will be displayed by server when any of the workers/hashers finds the desired hash.
8- For distributed communication, we can run this on multiple machines on different PC's, the commands are on page 3. We will additionally need to setup the connection for remote workers and an example is shown on page 4.

Input:
n - desired number of zeros - input from terminal

Output:
1- String used to find hash
2- The hash

Read Me File Statistics:
The following are the contents of the required stats of Read Me file.

Worker unit size:
max subset a single worker was given by server - only **1 string per hasher/node** was passed that connected

to server.
The result of the string appended to the gator ID is again passed to the hashing function. This allows us to minimize needless communication between worker and server.

Result of running program for input 4:
We ran the code for input 4 and received the following output.

Original Msg: aamjad;adba78f1755c3c420771d408edfa4d49dc2db83a98557a040bbbd7963
Hash: 00003ca5af47bbf62485a1232d94fcbb620240393d65ba7590dba7507f5181b3

Cores:
The number of cores for finding 7 zeros was **15 on laptop locally** as shown in below figure.

```
(server@AMMAR)2>
C:\Users\Ammar\LearnErlang\src>erl -sname server
Eshell V13.0.4  (abort with ^G)
(server@AMMAR)1> project:server(7).
Hasher Started:
Hasher Started:
Hasher Started:
Hasher Started:
Hasher Started:
Hasher Started:
Hasher Started:
Hasher Started:
Hasher Started:
Hasher Started:
Hasher Started:
Hasher Started:
Hasher Started:
Hasher Started:
Hasher Started:

Original Msg: aamjad;39e2aa7544b14f8abc12da2f44428ac7028d4a5dd3c77c69c5aecd7ff
Hash: 0000000b6932e71ce65f2ca2f180c3f32e82076b56e01a1452a262e649ccc658
Total CPU Time: 2354703
Total Run Time: 159719
Total Cores Used: 14.742785767504179
true
(server@AMMAR)2>
```

Another example with remote + local cores running is presented at end of this file.

Coins with most zeros we managed to find:
We managed to find upto **7 zero** string, the output is shown below.

Original Msg: aamjad;a76f78af03eb05b819aa90f1de18cf9dcffb4d8cf06981262a008cd04
Hash: 00000006505632a55167113210e6968d6df5084459f1fcacd331a5237b00d3b2

Largest number of worker machines able to run code:
There is no limit to connections, but the most we connected were 8 workers from remote terminals on a different pc + 15 local workers on server pc = **23 total worker nodes** working for server.

Commands to connect from remote PC:
Run these commands based on either windows or linux machines, Terminal must be run under administrator modes on both machines:

```
On both machine start terminal in administrator mode.

For Linux:

Step 1:
Linux machine 1:
erl -name freebsd_node1@10.20.23.44 -setcookie 'mycookie'
Linux machine 2:
erl -name freebsd_node2@10.20.23.37 -setcookie 'mycookie'
Step 2:
Linux machine 1:
net_kernel:connect_node('freebsd_node2@10.20.23.37')
Linux machine 2:
net_kernel:connect_node('freebsd_node1@10.20.23.44')

------------------
For Windows:

Step 1:
Windows machine 1:
werl -name windows_node1@10.20.23.44 -setcookie 'mycookie'
Windows machine 2:
werl -name windows_node2@10.20.23.37 -setcookie 'mycookie'
Step 2:
Windows machine 1:
net_adm:ping('windows_node2@10.20.23.37')
Windows machine 2:
net_adm:ping('windows_node1@10.20.23.44')

nodes( ).
```

Executing commands above to connect to remote server, the below is a screenshot of a successful connection with a remote PC.

```
Eshell V13.0.4  (abort with ^G)
(windows_node1@10.20.23.44)1> net_adm:ping('windows_node2@10.20.23.37').
pong
(windows_node1@10.20.23.44)2> nodes().
['windows_node2@10.20.23.37']
(windows_node1@10.20.23.44)3> c(project).
```

Results for input 7 :

Here is an example where we connected remotely using the commands above and then started 15 nodes on server PC and then connecting 8 nodes from a remote PC. The output is shown below for finding 7 zero's.

```
{ok,project}
(windows_node1@10.20.23.44)4> project:server(7).
Hasher Started:
Hasher Started:
Hasher Started:
Hasher Started:
Hasher Started:
Hasher Started:
Hasher Started:
Hasher Started:
Hasher Started:
Hasher Started:
Hasher Started:
Hasher Started:
Hasher Started:
Hasher Started:
Hasher Started:
Hasher Started:
Hasher Started:
Hasher Started:
Hasher Started:
Hasher Started:
Hasher Started:
Hasher Started:
Hasher Started:

Original Msg: aamjad;e011125bcb4653c078b96c4d0dd018bd3dbde1bab30f0a9ff94fd299e
Hash: 000000059fce5bea171264dab4c700f265cb723a6b78305b8dff1a0410ccb6ff
Total CPU Time: 1634094
Total Run Time: 176239
Total Cores Used: 9.272033999285062
true
(windows_node1@10.20.23.44)5>
```

Conclusion:

We are able to mine 0's based on the input that user enters, Outputting the required stats and messages.

Reach out to us at for any question at:

ammar.amjad@ufl.edu
5992-1730
mo.anas@ufl.edu
5981-5998