

Distributed Systems

Bitcoin Mining

Project Execution Instructions:

For Server:

mix escript.build

./project1 k

- Where k represents the number of 0's in the bitcoin

For Worker/Client

./project1 server_ip_address

- Where server_ip_address represents the IP address of server

Implementation Details:

1. **Work Unit:** The nonce generation process for our Bitcoin Mining project should be such that
 - It should not be completely dependent on any randomness function
 - The work must be divided equally among all the miner workers
 - It should not have an upper bound and scalable to any number of workers, which can utilize it simultaneously without exhausting the range
 - It should assign unique work to all the workers

Keeping above points in mind we followed an incremental approach for nonce generation.

The strings are generated by encoding (with base 64) a string integer series starting at "1", which gets incremented by 1 in every iteration.

Number	Encoding	Nonce Generated
"1"	Base.encode64("1")	MQ==
"2"	Base.encode64("2")	Mg==
"3"	Base.encode64("3")	Mw==
.	.	.
.	.	.
.	.	.
"1000000"	Base.encode64("1000000")	MTAwMDAwMA

We have created an assign_work module which assigns work to server as well as client workers as a range of work unit. Each work unit range consists of **1000000** string integers. Workers receive a range of integers and encode each number in the range to base 64 to generate string. Once a worker completes a unit of work, it asks for more work from the server and our module assigns the next available range to the worker.

The worker takes around 20 – 30 seconds to use the range (work unit) of 1000000 (million) numbers. If we reduce this range, then the workers flood the message queue of server too frequently and till the time server does not respond back to worker, the worker remain in idle state. With a range of million numbers, we got the average performance of 700% CPU utilization for the 8-core machine. We could easily keep track of the work being executed by different workers and the workers does not flood message queue too frequently of server.

Workers	Range
Server	1-1000000
Worker1	1000001-2000000
Worker2	2000001-3000000
.	.
.	.
Worker1 asks for rework	5000000 - 6000000
Worker2 asks for rework	6000000 - 7000000
.	.

The benefits of the above approach are:

- Since multiple strings can never generate same encoded string, our nonce will remain **unique for every worker** and hence there will never be any duplicate work.
- We are dividing the range equally, hence the task is being **equally divided among workers**
- It makes our model **scalable**, as there is no upper bound on range of numbers and any number of workers can connect the server and ask for work.

2. The **result** of running the program for 4 0's bitcoin:
Please find below the screenshot:

```

Terminal
mmewara@lin114-04:~/Downloads/bitcoin_mining$ time ./bitcoin_mining 4
Connected to Node: : "mohit-server@128.227.248.170"
Bitcoin mining started for Node : "mohit-server@128.227.248.170"
mohitmewara;ODAwMDAyOTEwNw== 0000DE0168ACF494DDB2286888582A408270AF46996F59F3D1F0EBD4EA22FAD2
mohitmewara;ODAwMzAyNDA1NQ== 0000F74F6420F21CC089E1AB2F4E7236C99D36CCEE76F8295EFADE0BE69697FB
mohitmewara;ODAwMzAyMzA1MQ== 0000185494D27C297CA0E765FE3C9E67E3D3A862814C1C58C86E9A9563876FE4
mohitmewara;ODAwNzAxNjE2OQ== 0000E488FA9F34388FB8EF086E4E4AC536D584A832D6FE0EAE5798A5BA1C4BF5
mohitmewara;ODAwNTAxMzc2Nw== 00009A33CD4BA528E71614BCEA90C566D09160176E98FF5833FCB862B3882ED2
mohitmewara;ODAwMDA0ODA1NA== 00003D8C5C21F3F963047E48A931C2B192DAC7624A89D30BF3FFBD54A6F456BC
mohitmewara;ODAwMzA0NzIwNA== 0000E2672ADD15FC4E1F489D7EAA6F98C4F8E6EBAC9E3D6AE8E43AF71A1355EE
mohitmewara;ODAwMDAxMzYyMQ== 000006C3AE6A5DC9C51BB0FF684AEA7C0C644356490D36ABA921F51E103EFD8
mohitmewara;ODAwMzA1NTg2Mg== 0000E1D4C89D5F7459BB87D7A87C2F52BE87D246E6D7C4D85BB7DAA4A09E962
mohitmewara;ODAwMDAyMzI3Mw== 0000D4424C132095779D5BD885A4FCA123DFAAE72D0C898687AC0634C8E7DF33
mohitmewara;ODAwMDAyNjA1OQ== 0000AB7BC669C5602C86DB0E6A6CF62DFA544A6EC760191A90C1997FC04A862C
mohitmewara;ODAwMTAwNDMwOA== 0000912C38D8A0929C4943FBC3AE9D34B6A0E051B97C04C62C1B2D68485F4863
mohitmewara;ODAwMzAzODk0NQ== 00005EE4BB4D8A38C9EF78A047D76DA79AFCF950E3DEAAB2981E0F9043DB4BFC
mohitmewara;ODAwNjA0MjIzMw== 00003165FC130150DB763BFCAB45F3434359FD9321C6276DE6A7FC41E88292
mohitmewara;ODAwNTAyNjgxNA== 00006E664C9B8FC02C5C8ABE0DDDA715142EC5A9092472D4930B21730778B0AC
mohitmewara;ODAwNDA0OTcxOQ== 000081A172E6736843F8ADD1877E0DD8654067E9C91C503CFE543165B310C530
mohitmewara;ODAwNDA1NTk4Mg== 00005F2D1934E721ACB812CD8415900EDCF183F4DDE19B23C837163965A1AF2C
mohitmewara;ODAwMzA3NzcyNA== 000039E9ECC56345A11FEA5DBBD824550986E104C2CDB6C9D271F8E57551C96C
mohitmewara;ODAwMDA0OTkyNA== 00008C3F91954A1699DA80FDB9B2E5DA9F07F825E6AA274ECCE9A843446BF76
mohitmewara;ODAwNTA0NDM3Mg== 000093FDB0BC234F591CAF23F8A5E31F7C5821AC1168CE96A8AB6A9DA2E5A62B
^C

```

- We ran our program 'bitcoin_mining' on an **Intel i7 machine with 8 cores** in UF Computer Science Lab (Dungeon Lab) and recorded the below timings:

```

Terminal
mmewara@lin114-04:~/Downloads/bitcoin_mining$ mix escript.build
mmewara@lin114-04:~/Downloads/bitcoin_mining$ time ./bitcoin_mining 4
Connected to Node: : "mohit-server@128.227.248.170"
Bitcoin mining started for Node : "mohit-server@128.227.248.170"
mohitmewara;NjgyOA== 000010544A73DD6E2FCD05A4D92507AD61CB0D832C075B5B3B7AE457829FAA89
mohitmewara;MTQwMTM= 00009938F94C7D5D0C473D022D0968A443908EBA3D177C2425A22063CABC7A1D
mohitmewara;MjQ4MDE= 0000DEACD1EF7929290B5CA1DFD5DDB613615A88B85FDF2D9E7FA4414341DF29
mohitmewara;MjY4NTQ= 0000369C2F2A655C1A464FBA68039097C4081E066520BF9897519A42C1CAD0F
mohitmewara;NjAyMTc= 000081A2F7B08C7F1BB88C2F0B58E68B7216DBCCA0044A785834100CB5ED5BC7
mohitmewara;NDA0NDQyMw== 00006017461C10B482212C580BEC818567A1DA14BE9DF3048F6E3DB89C29EEBE2
mohitmewara;MjAwMDg4Mg== 000062C98DA47F437FB73BF6E8567EAFBACB1087231A3BD6867D9309484B3684
mohitmewara;MjAwODQ4Mg== 000056827715000446D959886429190B69774DC19A5692BAE5087A96F7406138
mohitmewara;NjAyOTc3OA== 000084BC8E9C0956E2B6A0E51B64146C967361674DB20A118C8AB7C3F96647D6
mohitmewara;MjAyMDg0NQ== 0000EEF2AB1A41BD8B673BF3C2A347563C0ECD927BB60A960678D4EE9E069A16
mohitmewara;OTAwNzU5Nw== 0000F8FFC6AF794831B28C5AE02BFDFAFE24B0F67B6418946EE01DC2EF2F8B391
mohitmewara;NjA1NzE1Nw== 0000C9EB9365AC8C4C87BD3283071741C9D1104089CF673EAE77051CF8571DCF
mohitmewara;MTQwMDE2MDA= 0000986E759B5CD69F6CC8A627656DF3CAFA2985DB1AAA4B3F82E51E1D37456B
mohitmewara;MTQwMDk0Tk= 00004CF8659E4F121051202FD002350DC1B47475092AEA6848340087AC75D8
mohitmewara;ODAwMTIwNw== 00000CE374CA9AC79386591D63DBB16B48A8C1427BF483B0D928BF0C4350C269C
mohitmewara;MTMwMTAwMDA= 00009A6049BFF37446BB5B56D7E1F7AE4EE8380907D398F88975B837BC3A92F1
mohitmewara;MTI4OTIy 00002A074565D9762E775A27B5B125DB14CECEAEAFAC6C94251D0C8761B5E5576
mohitmewara;MjAyNTE5Mw== 0000CAB978B3EB206F837DF74994F54C9E0ECD1CCE9BF73F61CDEED6E6E5DEB
mohitmewara;NTAzMTUyOQ== 0000592A0F25365147FC6ADCC0AA398721B1ACA8EBE918F99AC005B075224229
mohitmewara;NTA1NDUzOQ== 00009A9E75F704F89D7E31544B6595653262843EC58BF3701D2E69D10C001032
mohitmewara;MTIwNTczOTk= 0000334C70DE0449874941381DE7920D6D90682F00CC7EAF9FAFAA3CAE269E6
mohitmewara;NDA4MDUyOQ== 00002CFBEE82458566989233A0612B40C3C613A509DAD30B1E0AE1C8BD120E4
mohitmewara;NjA4OTQzOA== 00007AC012D969DDE1FEB7387802A8ABF2EE22824FC14663EB7003DCD297589
mohitmewara;NjExMTM0OA== 0000155FABC831058BAF1E53B1510554C734D81594D433CD51321CACF0C32944
^C
real    1m56.977s
user    13m30.256s
sys     1m22.160s
mmewara@lin114-04:~/Downloads/bitcoin_mining$

```

Real : 1m56.977sec = 60+56 → 116 seconds

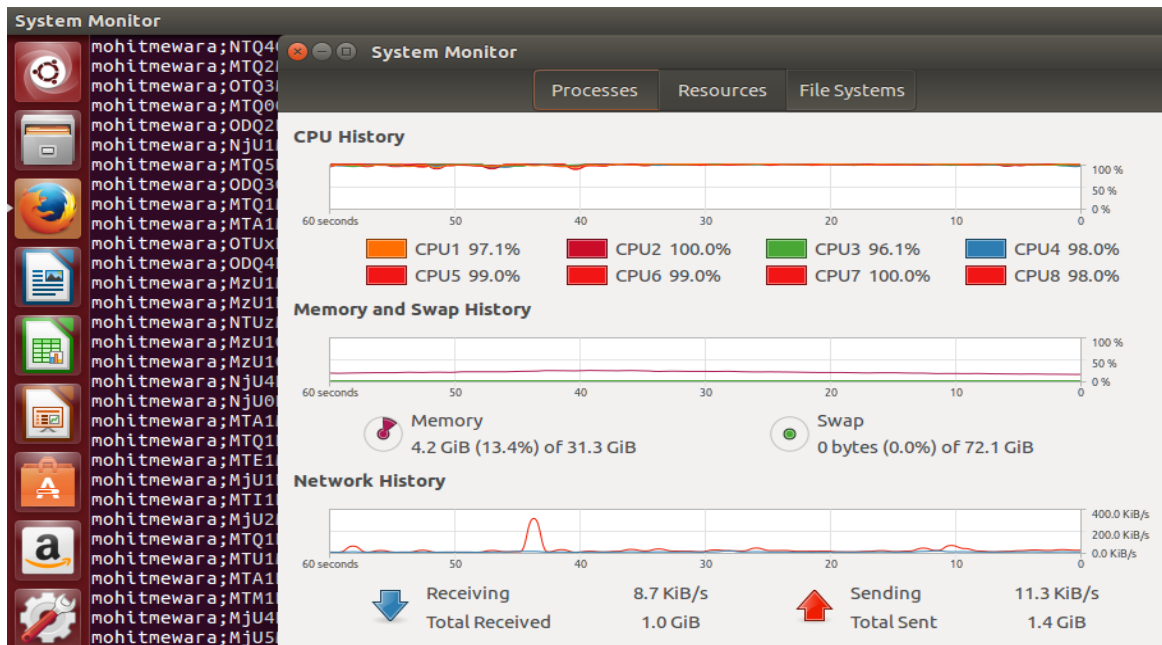
User : 13m30.256sec = 60+56 → 810 seconds

Sys : 1m22.160sec = 60+56 → 82 seconds

Ratio of CPU time to Real time = User/Real = 810/116 = **6.98 ~ 7**

Hence our project is giving a performance of **700%**.

The below is the screenshot of CPU utilization of all the cores. Almost all the cores are working on 100% CPU utilization.



- We found the below coin with most number of zeroes:

storm.cise.ufl.edu - PuTTY

```
storm:10% mix escript.build
warning: mtime (modified time) for "lib/mining/listener.ex" was set to the future, resetting to now
Compiling 1 file (.ex)
Generated escript bitcoin_mining with MIX_ENV=dev
storm:11% ./bitcoin_mining 7
Connected to Node: "mohit-server@128.227.205.236"
Bitcoin mining started for Node: "mohit-server@128.227.205.236"
mohitmewara;MTMzMDk4MTUyOA== 000000052A1EA17E076695CBA51763D85205DA99816B32EC98500861E079A0BC
^C
storm:12%
```

mohitmewara;MTMzMDk4MTUyOA==
000000052A1EA17E076695CBA51763D85205DA99816B32EC98500861E079A0BC
Number of zeroes in our Bitcoin: **7**

- The largest number of working machines we tested our code, was with 4 machines where 1 machine acted as a server and other three workers acted as clients.