Bitcoin Generator Project

Arijit Dutta (55889097)
Prasad Ramakant Hadkar (41075255)

24th September, 2021

Bitcoin Generator

The project is divided into Server and Client. The server program can run independently of the client and mine bitcoins with the required number of zeroes. When the client program is turned on, it informs the server that it's available and then the server gives the client the required number of zeroes. Once the client receives input from the server, it starts running and tries to find a valid bitcoin. Once it finds a valid bitcoin, it sends that information back to the client.

There are 3 types of actors in the server program; Controller, Worker, and Accumulator. Depending on the input received, the controller sends the number of zeroes as a parameter to the worker actor or sends the number of zeroes to the Controller actor of the client. So the controller does the job of assigning work to actors. Controller divides total work units into work batches of fixed sizes and then assigns different batches of strings to different Worker actors. Work is assigned to different Worker actors in a Round-Robin Pool fashion. Once the Worker actor receives the batch that needs to be processed, it calculates different hash values based on the SHA256 algorithm. Then it checks if the calculated hash value has the required number of zeroes in the prefix in which case it becomes a valid hash. If it finds a valid hash then it sends it to the Accumulator worker, which does the job of maintaining the count of valid bitcoins and printing them. Once a Worker actor finishes the assigned batch of strings, it requests a new batch of strings from the Controller actor. We tried testing the code with different batch sizes by checking CPU utilization for each batch size. We started increasing the batch size from 1 and we saw gradual improvement in CPU utilization. CPU utilization peaked at batch size of 100. **Thus, the size of the work unit for which our implementation worked the best is 100.**

The client program has only 2 kinds of workers: Controller and Worker. The client controller informs the server controller of its availability and receives the number of zeroes as input from the server Controller. The client Controller then sends the received input to its Worker actors. Once a Worker actor in the client program finds a valid bitcoin it directly sends the valid bitcoin to the Accumulator actor in the server.

We managed to mine multiple bitcoins with **7 leading zeros** and connected **5 machines** at the same time. We believe we could connect more machines if they were available.

The highest number of leading zero bitcoin that we could mine is 7.
Input String - a.duttaXOPhS4vaR0235552673
Bitcoin - 0000000945d198c957cd0e8b18ec12d6a48a535dc6f16a7334bf25b928560e7c

Input

BitGenServer

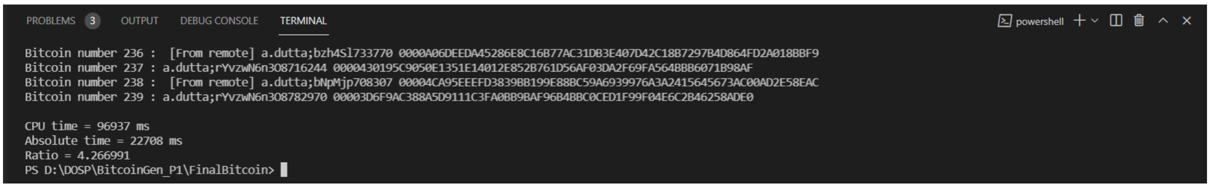
```
dotnet add package Akka.FSharp –version 1.4.25
dotnet add package Akka.Remote –version 1.4.25
dotnet add package Akka.Routing –version 1.4.25
dotnet run <numberOfLeadingZeroes>
```

BitGenClient

dotnet add package Akka.FSharp –version 1.4.25
dotnet add package Akka.Remote –version 1.4.25
dotnet add package Akka.Routing –version 1.4.25
dotnet run localhost (In case the server is running in a different machine, mention the ip of that machine. IP of the server machine should also be updated in the configuration string in the server code)

Sample Output (noOfLeadingZeroes = 4)

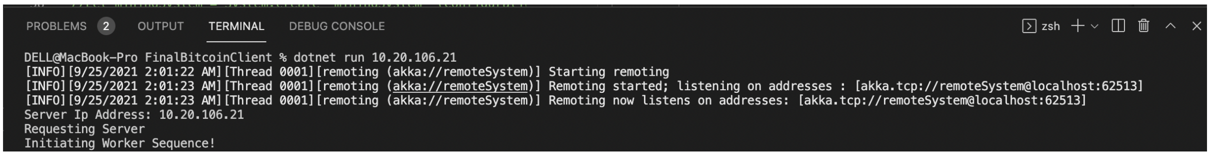
Server(Currently named as BitGenServer) -



```
Bitcoin number 236 :  [From remote] a.dutta;bzh4Sl733770 0000A06DEEDA45286E8C16B77AC31DB3E407D42C18B7297B4D864FD2A018BBF9
Bitcoin number 237 : a.dutta;rYvzwN6n3O8716244 0000430195C9050E1351E14012E852B761D56AF03DA2F69FA564BBB6071B98AF
Bitcoin number 238 :  [From remote] a.dutta;bNpMjp708307 00004CA95EEEFD3839BB199E88BC59A6939976A3A2415645673AC00AD2E58EAC
Bitcoin number 239 : a.dutta;rYvzwN6n3O8782970 00003D6F9AC388A5D9111C3FA0BB9BAF96B4BBC0CED1F99F04E6C2B46258ADE0

CPU time = 96937 ms
Absolute time = 22708 ms
Ratio = 4.266991
PS D:\DOSP\BitcoinGen_P1\FinalBitcoin> █
```

Client(Currently named as BitGenClient) -



```
DELL@MacBook-Pro FinalBitcoinClient % dotnet run 10.20.106.21
[INFO][9/25/2021 2:01:22 AM][Thread 0001][remoting (akka://remoteSystem)] Starting remoting
[INFO][9/25/2021 2:01:23 AM][Thread 0001][remoting (akka://remoteSystem)] Remoting started; listening on addresses : [akka.tcp://remoteSystem@localhost:62513]
[INFO][9/25/2021 2:01:23 AM][Thread 0001][remoting (akka://remoteSystem)] Remoting now listens on addresses: [akka.tcp://remoteSystem@localhost:62513]
Server Ip Address: 10.20.106.21
Requesting Server
Initiating Worker Sequence!
```