COP5615 -DISTRIBUTED OPERATING SYSTEMS
PROJECT 1

SUBMITTED BY:
CAROL NAVYA PAGOLU: 7374-7683
MONICA BHARGAVI KODALI: 1678-3481

PROBLEM STATEMENT

Bitcoins are one of the most popular cryptocurrencies that are used commonly. The bitcoins utilize the hardness of cryptographic hashing to ensure only a limited supply of coins are available. The aim of this project is to mine coins by generating random input strings and hashing them using the SHA256 algorithm. These mined coins are checked to ensure they have the required number of leading zeros in their prefix specified by the user. For this purpose, we use F# and an actor model to build a good solution that runs efficiently on multi-core machines.

SOLUTION

AKKA.net and f# are being used to implement this project. The following models were built keeping the problem requirements in mind.

1. SINGLE MACHINE IMPLEMENTATION

- In this implementation, all the actors are spawned in the same machine. The program takes the number of leading zeros as input from the command line and spawns the SUPERVISOR actor.
- The SUPERVISOR initializes and supervises a pool of WORKER actors. The task of the WORKER actors is to generate random strings, hash them using SHA256, and verify if the hash has the specified leading zeros in the prefix.
- If the string hash contains the specified number of zeros, the WORKER sends the resulting hash and the coin to the SUPERVISOR who displays the result to the user.

TO RUN THIS MODEL:

```
cd single_machine/ dotnet fsi --langversion:preview AkkaSingleMachine.fsx <number of leading zeros>
```

2. DISTRIBUTED IMPLEMENTATION

- In this implementation, there exists a server and several clients.
- The server takes the number of leading zeros as input from the command line and spawns a SUPERVISOR actor.
- The clients take the IP Address of the host and initialize WORKER actors of their own who participate in the mining process.

- The SUPERVISOR actor initializes and manages its own pool of WORKER actors. In addition to this, it also handles WORKERS initiated by the Clients.
- The Server's SUPERVISOR can run independently if there are no Clients available

TO RUN THIS MODEL:

```
cd distributed_machine/ dotnet fsi --langversion:preview RemoteServer.fsx <number of leading zeros>
cd distributed_machine/ dotnet fsi --langversion:preview RemoteClient.fsx <server IP address>
Press ctrl+Z to stop the mining process
```

NOTE:

Please make sure you change the hostname to the IP address of your server in RemoteServer.fsx (line 36). Always run the server before the clients.

ANALYSIS

WORKER COUNT:

To decide the number of workers, the project was run with input = 4 for various actor counts as shown below.

ACTOR COUNT	*NUMBER OF COINS MINED*	*CPU TIME (ms)*	*REAL-TIME (ms)*	*CPU/REAL RATIO*
2	1	13714	8334	1.645
4	1	14953	7168	2.085
6	1	17306	6792	2.547
8	17	40577	12269	3.307
10	12	55439	16791	3.266
12	23	61331	21701	2.845
14	28	51406	18414	2.790
16	47	118660	54111	2.192
18	52	211939	107097	1.978
20	61	214171	109240	1.960
22	73	249333	146860	1.697
24	102	338752	214099	1.582

From the table above, we notice that the program works the best when the number of actors is initialized to the **number of processor cores** in your system which is **8** in this case. Too less or too many actors lead to the loss of parallelism since the CPU to Real-time ratio gets closer to 1.

WORKER SIZE:

From our experiments, we notice the program mines the optimal number of coins when the maximal number of strings that can be generated by all workers is initialized between 10^6 to 10^8. This number is distributed evenly among all the WORKER actors. That means that each WORKER can generate strings anywhere between $125*10^3$ to $125*10^5$ strings roughly at the most. So the work size is directly proportional to the number of Actors in the System. If there are more Actors, we can increase the value of the Maximal number of Strings that can be generated. If the work size is too small, no coins are mined. On the other hand, if the work size is too big, the program takes a long time to mine the coins.

OUTPUT FOR N=4:

```
cpagolu;qkcw9n355l1198yjwvckgy4l0sf35   0000697C1EA78A2786C2E681AE3414BE4FB675F4795BAB29FC0CC20205EFF294

cpagolu;0agahfm694y9ig9m7f7apyw0qw3uvr  0000924CE9A683D986B9A400D7DF81FEAA80FD2C0BA3DABA3C57FF9FC387E3D6

cpagolu;psrgwmg 000008F3EB8C150B7680B819A4F513907922F46174951C081A29F45AFE5846E2

cpagolu;lon6uixfgovahqef78      0000D43AFD552739CDFBB2608FB3BF75E2E7579288DC717A6976DB0281F17369

cpagolu;59j     0000ACAEF6E91AA389BA19E6A0B5663BE676284944470C1722B12411ECEF5B9C

cpagolu;nmp     000023B57B86C0808E519AD5D265B1CE8BE6A92A8ABD7859797360DE063881C7

cpagolu;b5u42tp8jgrfjzp2hqjibpw5rn9v    000065FD4AD6E95EDDED8D87DA65C19DEB74497EF64BDAA89A148AB7D7C1D710

cpagolu;u4hmr0igof3otd8o6uz1    0000796772C51FA16E149E1EBDAA9BABB7BB627F51293C10851BE0E8ED71B4C3

cpagolu;j50x9   00001A413A0B66FCF513133CB8A475C177EEC32A9681CC8B0355E75C75A096F6

cpagolu;rdb4lp1tckrp    00002E6DAF63EB7C991CADC5B85D30AE653A55A5B5B55391468DB5D665346482

cpagolu;z4t0    0000AA2CCD50FEDDFCAA1CB28A581C6B28FB4C16696141D933F971BB925D88EE

cpagolu;wpwy0thm8       00009ACCCFB2367BE01ADE5631B7A0A72553D788660B85B35678327540CAD581

cpagolu;boohi   0000B8095A08D6AE1D8FF0F5E0C562EFED0DDBFB78463B864B11ECDC0E89B6DC

cpagolu;gjpbfk4urblqktazmqogt53 00005C9D557ADB7068C6C2DC3348BFC47C71023E99FECFA883F1667F50443C18

cpagolu;vmrthdvkb9idvzn82nmxcyk 000088EC67E7CB82A8FF4F1A2B2884A2BE86D8FD7839447B80477D33BBFED5F4

cpagolu;l9lqr31j9       0000D520277CD4141B1F0180A33FBA4244609445AFE54901D89F8BF44A558028

cpagolu;ti0nxwny4k2k    0000658039B29AF6990B59C0D735CFF73479976D6D0FAA644E03816E40E1680E

CPU time = 40577 ms
Real time = 12269 ms
CPU/RealTime ratio 3.306761

Press Enter to exit

Real: 00:00:15.202, CPU: 00:00:50.714, GC gen0: 3001, gen1: 197, gen2: 1
```

TIME ANALYSIS:

CPU time = 40577 ms
Real-time = 12269 ms
Ratio = 3.307

LARGEST NUMBER OF 0s:

The largest number of 0s the program was able to find was **7**.

LARGEST NUMBER OF WORKING MACHINES:

We were successfully able to run the distributed implementation on **4 machines** with processors 2.3 GHz Quad-Core Intel Core i7, Apple M1,2.3 GHz Quad-Core Intel Core i7, and Apple M1.