COP5615: Fall 2015

PROJECT1

**Project Members**

Praneeth Singh Rajput UFID: 6159 1691 prajput@ufl.edu

Sirivella Ananda Kishore UFID: 9951 5080 asirivella@ufl.edu

**READ ME**

The project deals with mining bitcoins with required number of zeroes in sha-256 implementation of random strings with exclusive usage of actor facility in Scala. The zipped folder includes client-server system for the project implementation.

The directory hierarchy is as follows:

**Project1**

**|+ BitcoinClient**

**|+ src**

**|+ main**

**|+ scala**

**|+ client.scala**

**|+resources**

**|+ localapplication.conf**

**|+ build.sbt**

**|+ BitcoinServer**

**|+ src**

**|+ main**

**|+ scala**

**|+ com.dos.bitcoin**

**|+ Server.scala**

**|+resources**

**|+ Remoteapplication.conf**

**|+ build.sbt**

**How to execute:**

Sample commands to start the server and client with 4 leading zeroes

Server:

>cd BitcoinServer/

sbt “run 4”

Client:

>cd BitcoinClient/

sbt “run 127.0.0.1”

Observations as specified.

1. *Determining the optimal size of work unit.*

We have used 4 workers with each working on an input size of 1 – 10 million strings to generate SHA 256 hash. We have considered computation on the input size of string for each worker as a unit of work. Our best performance was realized with 4 workers and the work (25 units of 1 million strings of sub work) was divided equally amongst these workers by utilizing the Round Robin Router.

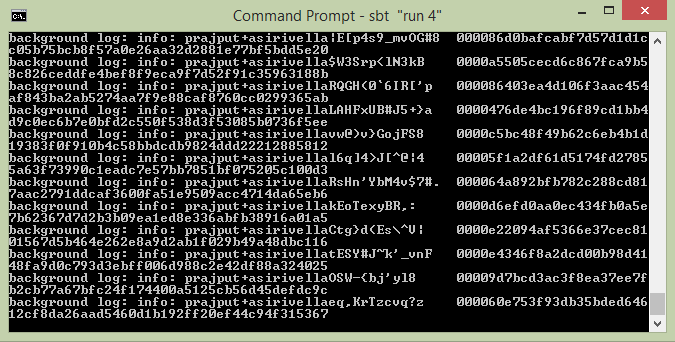
Sub problem for each worker: 1 million

Number of workers: 4

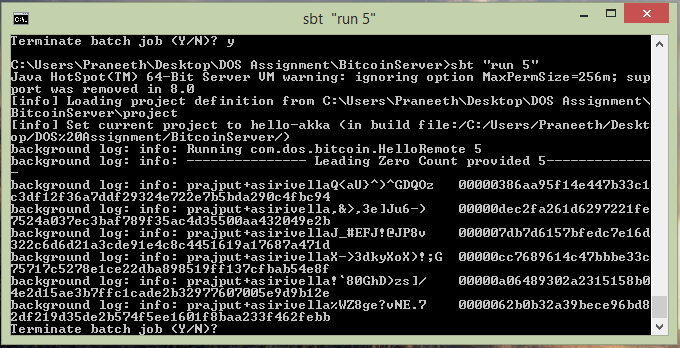
Number of units of sub problems divided using the router: 25

This observation was realized by running the same on our machines by changing the workers and units.

1. Output for 4 leading zeroes:



1. Output for 5 leading zeroes:



1. The coin with the most 0s you managed to find: **7**

Key: prajput+asirivellaKsY!8oJseBjJ%&`J>:ro)t\*kNHrl

SHA 256 hash: 0000000311d0c993ef230d83fb0007e325344396210dbe09f452c0935a6de90f

1. The largest number of working machines used for the project was **4**