Bitcoin Mining System

Team members

Surya Prasanna Kumar Yalla
UFID: 36436341

Divya Yadla
UFID: 94134866

Running Instructions:

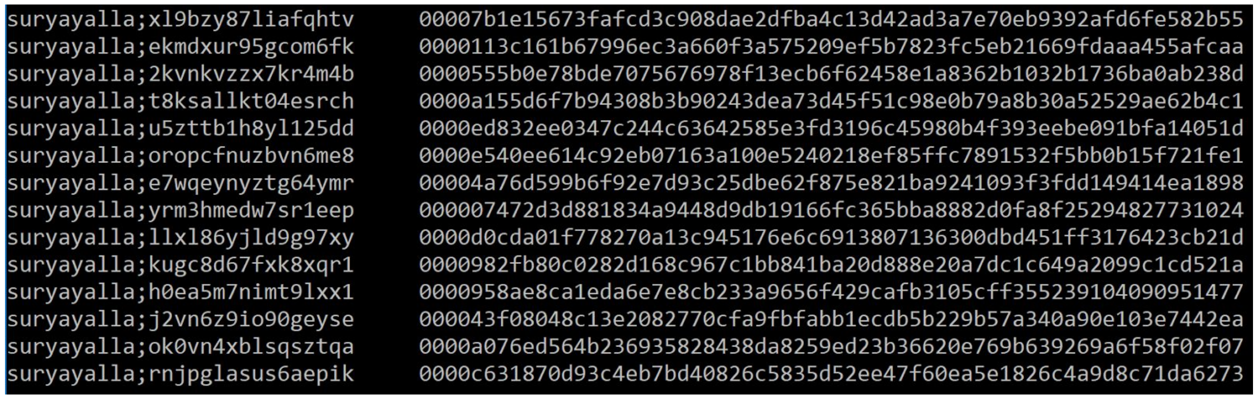
In server:

```
mix escript.build
./project1 4
```

In worker:

```
mix escript.build
./project1 “ip address of server”
```

Result:



```
suryayalla;xl9bzy87liafqhtv     00007b1e15673fafcd3c908dae2dfba4c13d42ad3a7e70eb9392afd6fe582b55
suryayalla;ekmdxur95gcom6fk     0000113c161b67996ec3a660f3a575209ef5b7823fc5eb21669fdaaa455afcaa
suryayalla;2kvnkvzzx7kr4m4b     0000555b0e78bde7075676978f13ecb6f62458e1a8362b1032b1736ba0ab238d
suryayalla;t8ksallkt04esrch     0000a155d6f7b94308b3b90243dea73d45f51c98e0b79a8b30a52529ae62b4c1
suryayalla;u5zttb1h8yl125dd     0000ed832ee0347c244c63642585e3fd3196c45980b4f393eebe091bfa14051d
suryayalla;oropcfnuzbvn6me8     0000e540ee614c92eb07163a100e5240218ef85ffc7891532f5bb0b15f721fe1
suryayalla;e7wqeynyztg64ymr     00004a76d599b6f92e7d93c25dbe62f875e821ba9241093f3fdd149414ea1898
suryayalla;yrm3hmedw7sr1eep     000007472d3d881834a9448d9db19166fc365bba8882d0fa8f25294827731024
suryayalla;llxl86yjld9g97xy     0000d0cda01f778270a13c945176e6c6913807136300dbd451ff3176423cb21d
suryayalla;kugc8d67fxk8xqr1     0000982fb80c0282d168c967c1bb841ba20d888e20a7dc1c649a2099c1cd521a
suryayalla;h0ea5m7nimt9lxx1     0000958ae8ca1eda6e7e8cb233a9656f429cafb3105cff3552391040909514 77
suryayalla;j2vn6z9io90geyse     000043f08048c13e2082770cfa9fbfabb1ecdb5b229b57a340a90e103e7442ea
suryayalla;ok0vn4xblsqsztqa     0000a076ed564b236935828438da8259ed23b36620e769b639269a6f58f02f07
suryayalla;rnjpglasus6aepik     0000c631870d93c4eb7bd40826c5835d52ee47f60ea5e1826c4a9d8c71da6273
```

The command used to find the time required to find the bitcoins can be found by using the command:

```
time project1 k
```

Here, k is the number of zeroes. We have used a value of k=4 and the time was fixed to 30s. The unit of time used in the calculation is seconds.

The results are tabulated as below:

Actors	Real time	User time	System time	Num of bitcoins	Bitcoins/real time	CPU time/real time

1	30.472	30.156	0.438	12	0.3938041481	1.004003676
2	30.451	59.875	0.656	21	0.6896325244	1.987816492
3	30.454	89.734	0.641	33	1.083601497	2.967590464
4	30.469	116.078	3.484	33	1.083068036	3.924053956
5	30.565	116.672	3.031	32	1.046949125	3.916342221
6	30.581	116.656	3.078	33	1.079101403	3.91530689
7	30.504	115.875	3.031	37	**1.212955678**	3.898046158
8	30.468	116.188	3.219	28	0.9189969804	3.919095444
9	30.461	116.438	3.109	31	1.017694757	3.924592101
10	30.519	116.625	3.156	25	0.8191618336	3.924800944
11	30.691	116.203	3.469	29	0.9449024144	3.899253853
12	30.505	116.188	3.328	27	0.8851008031	3.917915096
13	30.466	116.594	3.063	27	0.8862338344	**3.92755859**
14	30.586	116.531	3.031	25	0.817367423	3.909043353

The server and client are both 4 core 10 thread machines with a frequency of 2.60GHz.
Most number of bitcoins were obtained when 7 actors were used as can be seen from the table, and the maximum number of bitcoins found were 37.
Highest CPU utilization was when the number of actors was 13.

The bitcoin with the most number of zeroes found is:

suryayalla;pzu88gytj4eyon2e and it's hash is
000000007da1bd04ae6d2966819a6edde83e245a28f2090d03be52989096dbbc

The highest number of zeroes found are 8.

The highest number of machines this can be implemented with is 36 because the pool size is 36. We have only tried running it on 4.

We are generating random strings of 16 character length and appending them as suffixes to the gatorlink username.

The characters are randomly picked from the pool <0123456789abcdefghijklmnopqrstuvwxyz>.

Depending on the number of nodes and actors used the pool sizes of 1st and 2nd characters are determined.

For example:

Case 1: Only master node is present with 3 actors:

In this case, while generating random strings, the first character is taken from the pool <0..z>.

Node number	Actor number	1st character pool size	2nd character pool size	Next 14 chars pool size
Server	Actor 1	<0-z>	<0-b>	<0-z>
Server	Actor 2	<0-z>	<c-n>	<0-z>
Server	Actor 3	<0-z>	<o-z>	<0-z>

Case 2: Two nodes are present with 3 actors each:

Node number	Actor number	1st character pool size	2nd character pool size	Next 14 chars pool size
Server	Actor 1	<0-h>	<0-b>	<0-z>
Server	Actor 2	<0-h>	<c-n>	<0-z>
Server	Actor 3	<0-h>	<o-z>	<0-z>
Worker 1	Actor 1	<i-z>	<0-b>	<0-z>
Worker 1	Actor 2	<i-z>	<c-n>	<0-z>
Worker 1	Actor 3	<i-z>	<o-z>	<0-z>

As the work load is being distributed equally, there is no communication between the server and worker once the rules are specified.