Project 1 - Bitcoin Mining in Erlang

COP5615 - Distributed Operating System Principles

Goal:

Implement the actor model in Erlang functional language to solve the bitcoin mining problem and make the distribution of the task among the actors that run on the multi-core machines.

Project Members:

- Ratna Prabha Bhairagond (UFID – 8827 4983)
- Varad Rajeev Sanpurkar (UFID – 1782 9883)

Requirements:

- Latest version of Erlang
- Multicore machines (intel i5/ i7)

Steps for compiling and running the code:

1. **Establish the connection between the server and the client**
   - Both the server and client machine should be connected to the same network.
   - Any protection mechanisms like firewall should be disabled on both the machines.
   - Both the machines should have private network profile.
   - Use the following command to establish the cookie connection between the server and client machine. Use this command on both the terminals of the client and the server.

     erl -name *uniqueCharacter@IP_address* -setcookie *cookie_name*

2. **Server-side compilation:**
   - In the terminal, change the directory to the folder having erlang project source file.
   - Run the following command to compile and run the erlang code on server machine.

     c(server).

     server: start(N). where N is the total number of zeroes with which hash of the bitcoin should start.

3. **Client-side compilation**
   - In the terminal, change the directory to the folder having erlang project source file.
   - Run the following command to compile and run the erlang code on server machine:

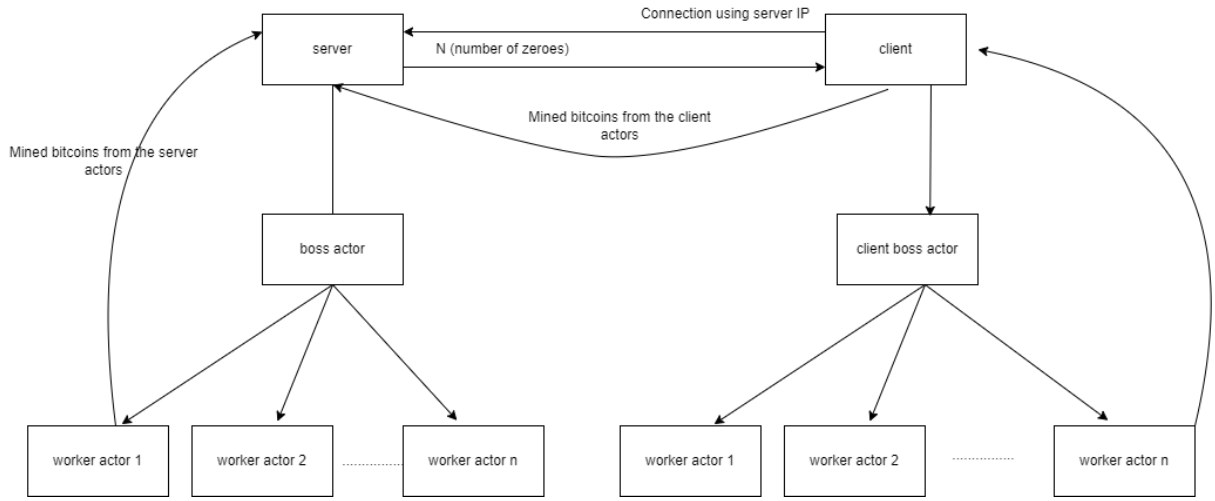
     c(client).

     client:send_request('parameter'). where parameter is the IP address of the client set while generating the cookie.

Size of the work unit:

We have first created a Boss Actor which will further create a total of NUM*250 worker actors where NUM is the total number of logical processors available on the individual machine. Each actor will mine a total N*1000 number of random strings where N is the number of leading zeroes for which bitcoin is to be mined.

Mathematical Model:



Program flow:

1. Start() function in the server program will take the input as a number of leading zeroes (N). This function will spawn the boss actor.

2. Boss actor will create worker actors for the server's machine based on a number of logical processors.

3. Each actor will perform the following steps:

- Generate a random string.
- Generate the hash of this string.
- Compare the number of leading zeroes in the hash with N and send the result to printer actor if it is matched.

These steps will be performed by the each actor parallelly N *1000 times on the server machine.

4. Client will request the work from the server using the server IP address.

5. Clienthandler actor in the server will provide the value of N to the client.

6. Client will accept this N and perform the following steps on the client machine separately.

7. Boss actor will create worker actors for the client's machine on the basis of number of logical processors.

8. Each actor will perform following steps:

- Generate a random string.
- Generate the hash of this string.
- Compare the number of leading zeroes in the hash with N and send the result to same printer actor in the server machine if it is matched.

Program Results/ Outputs:

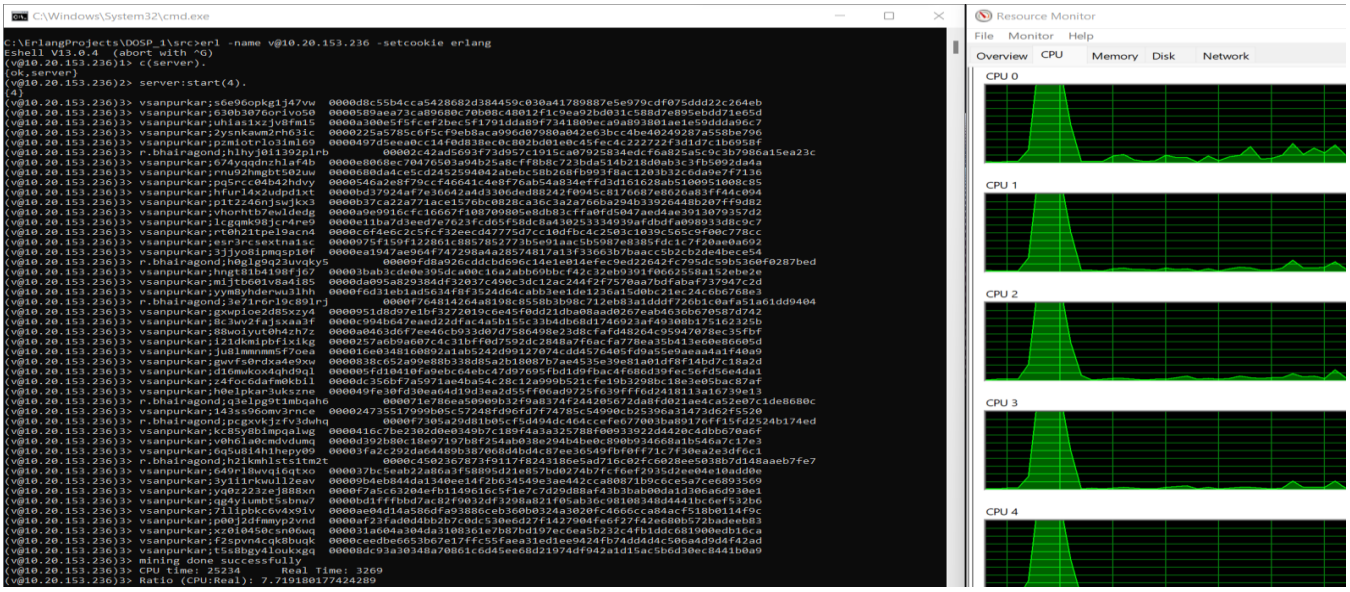
1. Server Machine configuration for checking 4 leading zeroes:

- Number of actors: 2000 actors
- Coined to be mined per actor: 4000
- Total work size: 8000000
- Number of logical processors: 8

2. Client Machine configuration for checking 4 leading zeroes:

- Number of actors: 1000 actors
- Coined to be mined per actor: 4000
- Total work size: 4000000
- Number of logical processors: 4

3. Output for 4 leading zeroes:

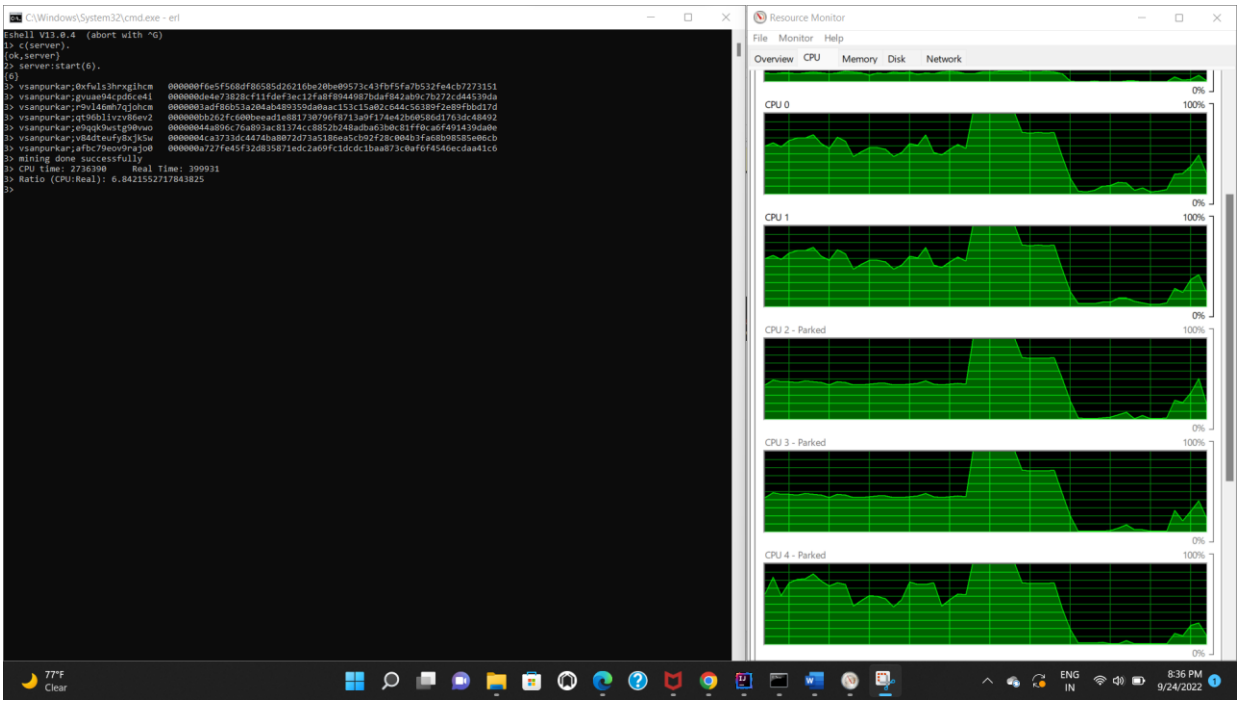


- Coins mined by the server will have the prefix '**vsanpurkar'** and coins mined by the client will have the prefix '**r.bhairagond'.**
- **CPU time –** 25234 ms
- **Real Time –** 3269 ms
- **CPU Time/Real Time –** 7.719
- **Largest number of working machines:** 2
- **There is a clear spike for all the cores during this bitcoin mining.**

4. Output for 5 leading zeroes:

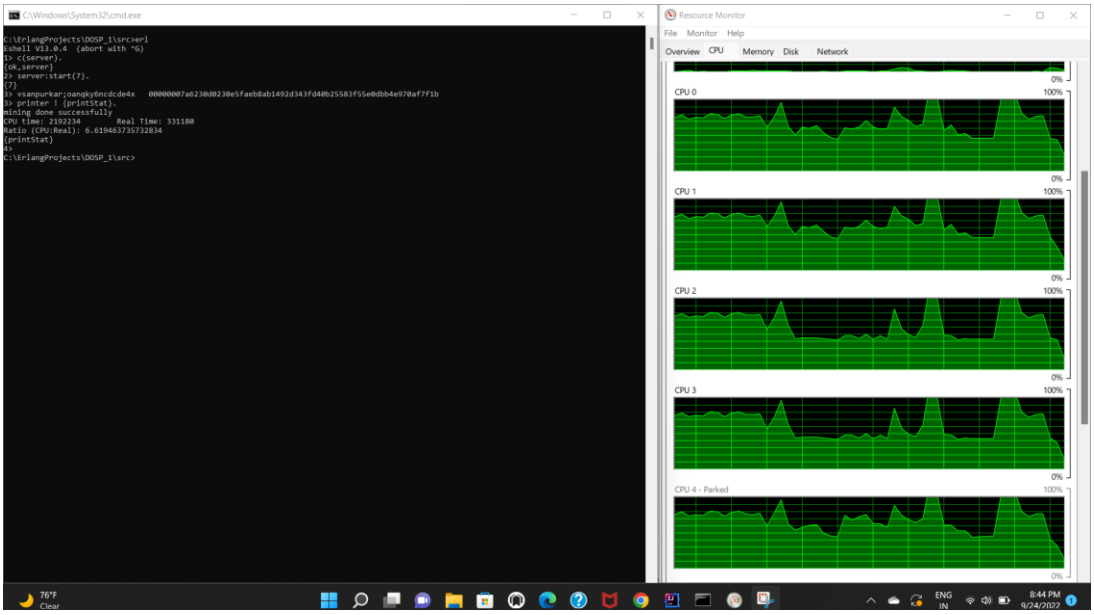
- Coins mined by the server will have the prefix '**vsanpurkar'** and coins mined by the client will have the prefix '**r.bhairagond'.**
- **CPU time –** 71375 ms
- **Real Time –** 9120 ms
- **CPU Time/Real Time –** 7.82

5. Coins with the most zeroes (6):



- **CPU time –** 71375 ms
- **Real Time –** 9120 ms
- **CPU Time/Real Time –** 7.82

6. Coins with the most zeroes (7):



- **CPU time –** 2192234 ms
- **Real Time –** 331180 ms
- **CPU Time/Real Time –** 6.61