**Finding Perfect Squares using Elixir**

COP5615 - Distributed Operating Systems Principles Project 1

The goal of this project is to use Elixir and the actor model to build a good solution to find sequence of squares of integers that leads to a perfect square and runs well on multi-core machines.

**Authors**

* **Akash R Vasishta** - *UF ID: 5395 5080*
* **Aditya Bhanje** - *UF ID: 9697 1842*

**Algorithm**

In this program, after taking input from the user in the form of (N,k), where k is the number of integers in the sequence and N is the final number for the start of the sequence, we designate each actor process a range of sequences(work unit) for which it has to find whether the sum of the squares of those integers is a square of an integer or not. If the result comes out to be true, it then prints the start number of that sequence. Once all the sequences given to that actor are computed, it reports the process which spawned it about it’s exit.

**Size of the work unit and Running Time**

For the problem of **N=1000000 and k=24**, the following table shows the relation

between Work Unit,CPU TIME(SYS TIME + USER TIME), REAL TIME and CPU UTILISATION

|  |  |  |  |
| --- | --- | --- | --- |
| Work Unit | CPU TIME | REAL TIME | CPU UTILISATION |
| 10 | 31.7 | 29.52 | 1.07 |
| 100 | 1.4 | 0.63 | 2.22 |
| 1000 | 1.115 | 0.492 | 2.26 |
| 10000 | 1.124 | 0.503 | 2.23 |
| 50000 | 1.088 | 0.481 | 2.26 |
| 100000 | 1.097 | 0.506 | 2.16 |

For the problem of **N=10000000 and k=24**, the following table shows the relation

between WORKUNIT, CPU TIME(SYS TIME + USER TIME), REAL TIME and CPU UTILISATION

|  |  |  |  |
| --- | --- | --- | --- |
| Work Unit | CPU TIME | REAL TIME | CPU UTILISATION |
| 100 | 44.16 | 34.91 | 1.26 |
| 1000 | 10.73 | 2.96 | 3.625 |
| 10000 | 10.6 | 2.92 | 3.63 |
| 20000 | 10.61 | 2.93 | 3.62 |
| 100000 | 10.59 | 2.92 | 3.62 |
|  |  |  |  |

After observing these results, we came to the conclusion of choosing the size of work unit to be 20000, as for larger values of N it gave us a good CPU Utilisation value.

For work unit as **20000** and **k=4**, the following was observed:

|  |  |  |  |
| --- | --- | --- | --- |
| N | CPU TIME | REAL TIME | CPU UTILISATION |
| 10 | 0.444 | 0.324 | 1.37 |
| 100 | 0.451 | 0.329 | 1.37 |
| 1000 | 0.434 | 0.327 | 1.32 |
| 10000 | 0.451 | 0.328 | 1.37 |
| 100000 | 0.502 | 0.348 | 1.44 |
| 1000000 | 1.104 | 0.492 | 2.24 |
| 10000000 | 10.606 | 2.936 | 3.612 |
| 100000000 | 108.784 | 28.23 | 3.853 |
| 1000000000 | 1145.937 | 296.038 | 3.87 |

**Largest number of working machines on which the code was run**

The code was simultaneously run on 2 machines with the actors equally divided among the 2 machines.

**Output of N = 1000000, k = 4**

**rvakash@rvakash-HP-Pavilion-15-Notebook-PC:~/Documents/masters/Masters/UF/DOS/projects/FindPerfectSquares/proj1$ time mix run -e Proj1.scheduler 1000000 4 0**

**real 0m0.623s**

**user 0m1.190s**

**sys 0m0.059s**

**rvakash@rvakash-HP-Pavilion-15-Notebook-PC:~/Documents/masters/Masters/UF/DOS/projects/FindPerfectSquares/proj1$**

**Largest Problem successfully solved**

|  |  |
| --- | --- |
| The above code successfully runs for the values of **N=1000000000, k=24** and CPU Utilisation was found out to be **3.87** for the same. | |
|  |  |
|  |  |

**Running the code**

After unzipping the provided folder, open the command prompt in that folder and type the following commands:

$ mix run lib/proj1.ex <arg1> <arg2> <arg3>

where arg1 is the value of N and arg2 is the value of k.

arg3- 0 for local, 1 for multiple machines.

Example - time mix run lib/proj1.ex 1000000 24 0

**Running on Multiple machines(demo will be given during TA Hours)**

1. Machine – 1 and Machine -2 should be on the same network
2. Machine – 1
   1. $iex –name [node\_one@192.168.1.2](mailto:node_one@192.168.1.2) –cookie something
3. Machine – 2
   1. $ iex –name [node\_two@192.168.1.](mailto:node_two@192.168.1.2)47 –cookie something
4. Machine – 1
   1. $Node.connect :[node\_two@192.168.1.47](mailto:node_two@192.168.1.47)
5. Compile the code on both the machines –
   1. $c(“perfectSquares.exs”)
6. Run on Machine – 1
   1. $PerfectSquares.scheduler(1000000, 24, 1)

Where [node\_one@192.168.1.2](mailto:node_one@192.168.1.2) and [node\_two@192.168.1.47](mailto:node_two@192.168.1.47) are the IP Addresses of the two machines respectively.