# COP5615- Distributed Operating System Principles

Project 1

Developed by –
Shaik, Fayaz - UFID: 6326-9935
Mullapudi, Aseesh - UFID: 9175-1971

## 1) Aim –

Input: The input provided (as command line to your program, e.g. my app) will be two numbers: N and k. The overall goal of your program is to find all k consecutive numbers starting at 1 and up to N, such that the sum of squares is itself a perfect square (square of an integer).

Output: Print, on independent lines, the first number in the sequence for each solution.

## 2) Methodology –

i) Using Akka Actor Model in FSharp, we solved the problem by divided the bigger problem into smaller sub-problems and assigning these sub-problems to Worker Actors to concurrently solve the bigger problem.

ii) There is a Boss Actor, which takes the input to solve a bigger problem, divides the problem into sub-problems and then assigns it to the workers and asks them to solve the tasks, once the worker is finished executing every task, it reports back to the Boss saying that its work is done for a specific task.

## 3) How to Run –

Once Dotnet, Akka.Net and Akka.Fsharp are all installed in the Project environment, extract the zip file and run the program in the following manner:

Command: dotnet fsi “—langversion:preview” proj1.fsx <N> <k>

Where N, k are the command line arguments.

4) <u>Project Questions Answered</u>:

i) *<u>Size of the work unit</u>*:
The size of the work that is given to each worker actor, is determined based on the number of cores present on the machine and the input parameter N given to the program.

Lets suppose we have 8 cores and the input N, that means we would consider 8 Actors and each actor would be given a work load of (N / 8) if N is a multiple of 8, if we also have any reminder (i.e.; N % 8), then the first N%8 actors will take an extra work unit.

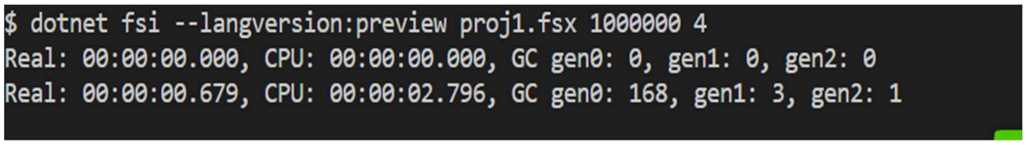
This is confirmed by the parallelism that we observed where the CPU to Run time ratio is almost 4 on Quad core machine, which means we are achieving maximum ratio when we consider the number of cores as the number of actors.

ii) <u>The result of running 1000000 4:</u>
There is no perfect square for any sequence of sum of 4 consecutive squares in the range of 1 to 1 Million. Hence, there are no results for this testcase.

iii) <u>Timings for running 1000000 4</u>:

CPU = 00:00:02.796 secs
Real = 00:00:00.679 secs

The ratio is: 4.11 (which is greater than 1, so parallelism is achieved)



```
$ dotnet fsi --langversion:preview proj1.fsx 1000000 4
Real: 00:00:00.000, CPU: 00:00:00.000, GC gen0: 0, gen1: 0, gen2: 0
Real: 00:00:00.679, CPU: 00:00:02.796, GC gen0: 168, gen1: 3, gen2: 1
```

iv) <u>The largest problem</u> –

The largest problem that we managed to solve is for –
N = 1000000000 (1 Billion)
K = 24

Results –
1
9
20
25
44
76
121
197
304
353
540
856
1301
2053
3112
3597
5448
8576
12981
20425
30908

```
35709
54032
84996
128601
202289
306060
353585
534964
841476
1273121
2002557
3029784
3500233
5295700
8329856
12602701
518925672
19823373
29991872
34648837
296889028
52422128
816241996
196231265
82457176
342988229
124753981
```

Timings –
CPU = 5820.109 secs
Run Time = 1420.924 secs
Ratio = 4.09 (which is greater than 1, so parallelism is achieved)