FPP Assignment 2 Writeup

Vishesh Rangwani

The following are the screenshots for running of ethe program on 1, 2, 3 and 4 threads:

```
(base) root@DESKTOP-68U5KCG:/home/FPP-Assignments/temp/A2 for sending/A2 for sending/Untitled# QUILL_WORKERS=1
/nqueens 12
OK
NQueens(12) Time = 1.605659sec
(base) root@DESKTOP-68U5KCG:/home/FPP-Assignments/temp/A2 for sending/A2 for sending/Untitled# QUILL_WORKERS=2
/nqueens 12
OK
NQueens(12) Time = 1.111587sec
```

```
# QUILL_WORKERS=3 ./nqueens 12
OK
NQueens(12) Time = 1.001204sec
```

```
# QUILL_WORKERS=4 ./nqueens 12
OK
NQueens(12) Time = 0.934369sec
```

$$Speedup = \frac{T_{sequential}}{T_{parallel}}$$

Consider the case for execution on 1 thread: This is essentially the sequential running of the NQueens program. So,

$$Speedup = \frac{1.605659}{1.605659} = 1$$

For 2 threads:

$$Speedup = \frac{1.605659}{1.111587} = 1.4444744316009452$$

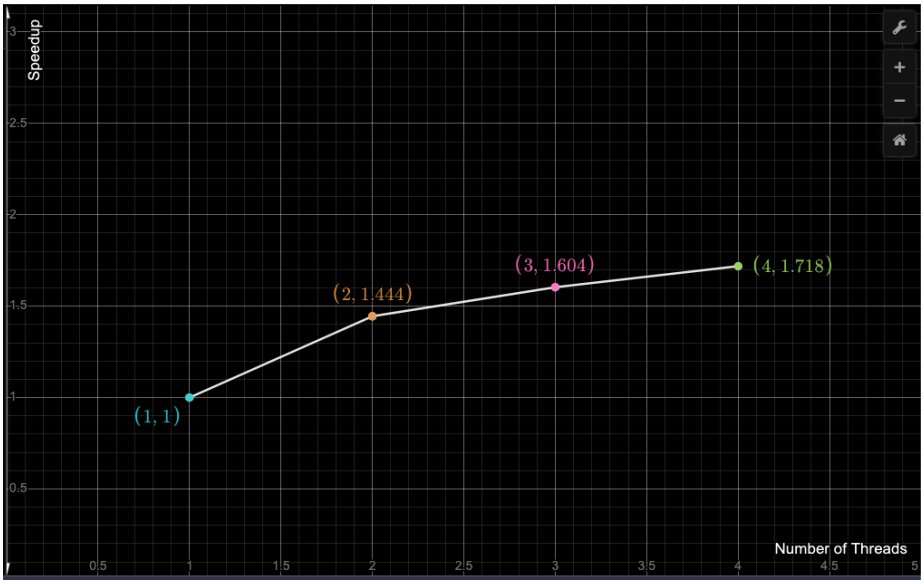
For 3 threads:

$$Speedup = \frac{1.605659}{1.00124} = 1.60372811135393$$

For 4 threads:

$$Speedup = \frac{1.605659}{0.934369} = 1.7184420716012625$$

Plotting the graph:



From the above graph, we can see that the speedup for the given problem lies in the sublinear speedup range.

The speedup does not grow linearly as the number of threads increase. It's slope begins to reduce to being less than 1. For each progressive thread, slope reduces even more.

The reason for the same is that the creation of threads also takes up a few thousand cycles of the CPU. Apart from that, we have used locks for the counter for the number of tasks, and for accessing the Deque for stealing and popping. These factors also stop the parallel program from achieving a linear speedup. The locks are used in the program to prevent data race conditions.