Solution to Homework 1

Meshach Aruoriwo Ogbor,
Md Rasel

October 18, 2023

1 Exercise 1.1 - Trivial Parallelization

```c
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char* argv[])
{
    int i,a=0,b=1000000;
    long tot=0;

    if(argc == 3)
    {
        a = atoi(argv[1]);
        b = atoi(argv[2]);
    }
    if ( !(b>=a) )
        exit(1);

    for (i=a; i<b; i++)
    {
        tot += i*i;
    }
    printf("%li\n",tot);
}
```

2 Submit Script

```
#!/bin/bash
```

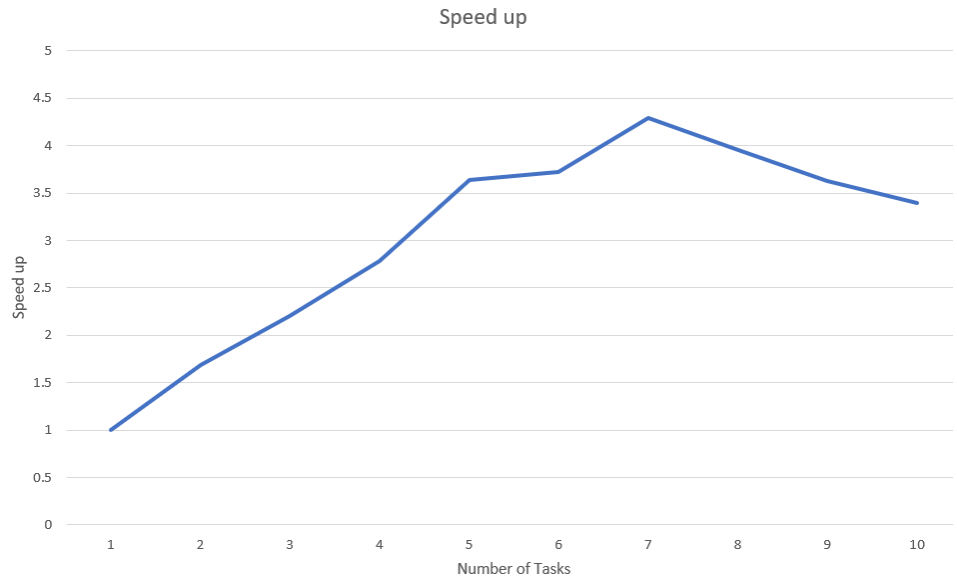

Figure 1: Spped up vs Number of Tasks

```
2
3 echo "100 millions in one go"
4 date +%T:%N
5 ./trivial 0 100000000
6 date +%T:%N
7
8 echo "50 millions and then remaining 50 millions"
9 date +%T:%N
10 ./trivial 0 50000001 &
11 ./trivial 50000001 100000000 &
12 wait  # don't measure the time before all background
     tasks finished
13 date +%T:%N
```

3 Results

Theoretically, if we split the tasks in 2 parts, we should get the increase in the speed up i.e., double. What we can see from our graph, we get an increasing trend for our speed up when we divided our task in more than one part. Though, when we split it 8/9/10 parts, our graph shows a decrease in speed up.

4 Exercise 1.2 - MPI program

Operating System: Linux
Number of Cores: 32
Processor Type: Intel(R) Xeon(R) Silver 4110 CPU @2.10GHz
Cache Levels and Sizes: 32, 11264KB
Memory (RAM) Size: 93Gi

```c
#include <stdio.h>
#include <mpi.h>

int main(int argc, char* argv[])
{
    MPI_Init(&argc, &argv);

    while(1) {
        // an infinite loop
    }

    MPI_Finalize();
    return 0;
}
```

5 Submit Script

```
#!/bin/bash
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=4
#SBATCH --partition=compute2011
#SBATCH --exclusive


module load mpi/openmpi/4.1.0
mpirun ./ex1_2
```

6 Results

We tried to find out on which node it ran on cluster but unfortunately, could not gather that information.