CS636: Analysis of Concurrent Programs **Jan-April 2018**

Report: Assignment 1

Submitted By: Siddharth Agrawal (150716)

1.1 Problem 1

1.1.1 Compilation

For the sequential program:
`$ g++ seq.cpp -o seq -lpthread`
For the parallel program:
`$ g++ par.cpp -o par -lpthread`

1.1.2 Execution

For the sequential program:
`$ ./seq <No. of elements> <array 1 path> <array 2 path>`
For the parallel program:
`$ ./par <No. of worker threads> <No. of elements> <array 1 path> <array 2 path>`

1.1.3 Report

The graph below shows how the programs scales with the number of threads.
(Note: For the sequential program, result shown with a horizontal line)

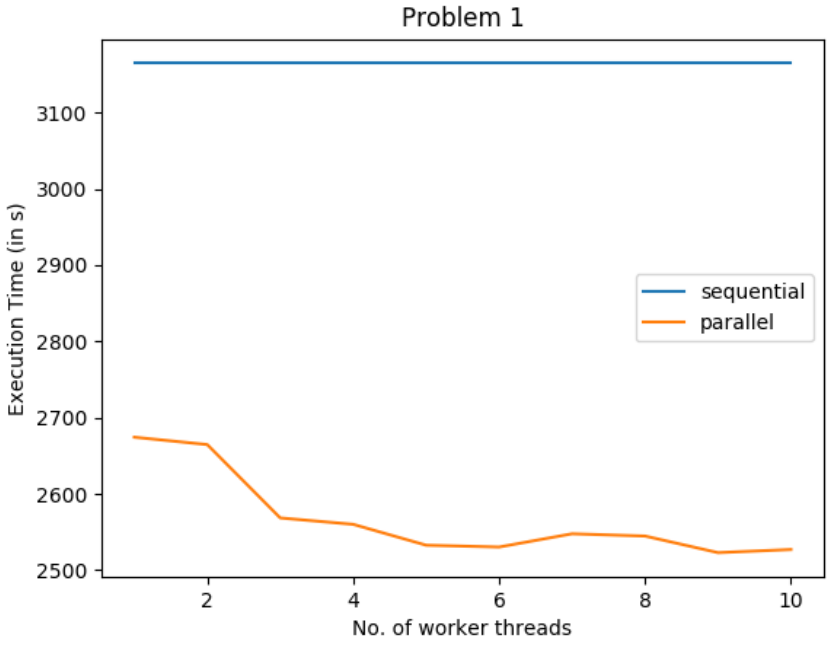

Figure 1.1: The number of elements in the two arrays used: 1,000,000

1.2 Problem 2

1.2.1 Compilation

For the sequential program:
`$ g++ seq.cpp -o seq -lpthread`
For the Hillis & Steele program:
`$ g++ hns.cpp -o hns -lpthread`
For the Blelloch program:
`$ g++ ble.cpp -o ble -lpthread`

1.2.2 Execution

For the sequential program:
`$ ./seq <No. of elements> <array path>`
For the Hillis & Steele program:
`$ ./hns <No. of worker threads> <No. of elements> <array path>`
For the Blelloch program:
`$ ./ble <No. of worker threads> <No. of elements> <array path>`

1.2.3 Report

The graph below shows how the programs scales with the number of threads.
(Note: For the sequential program, result shown with a horizontal line)

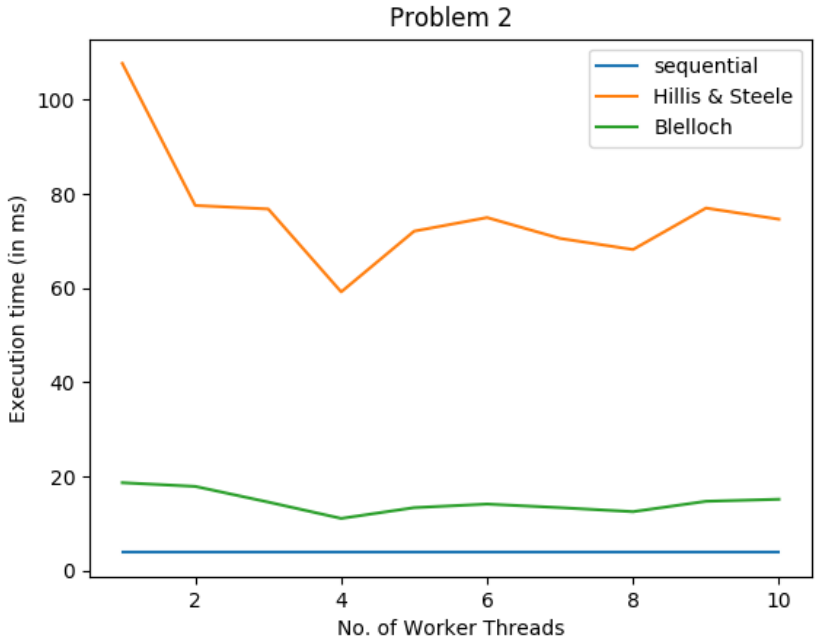

Figure 1.2: The number of elements in the two arrays used: 1,048,576

1.3 Problem 3

1.3.1 Compilation

```
$ g++ spawn.cpp -o spawn -lpthread
```

1.3.2 Execution

```
$ ./seq <Limit of shared counter>
```