Parallel Linear Searching in C using Open-MP

Members:k190304,k190177

Umer Zaidi, Abdur Rehman

PDC PROJECT

REPORT

Abstract

Our Project is research into whether Parallel Speeds differ when linearly searching for different Data Types. Hence we will compare 3 different data types:

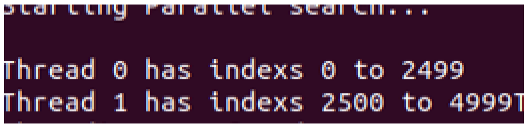
- String(char array)
- Integer
- Double

Hypothesis: Speeds will be faster for Integer as it is the smallest data type, while String will take most time as it is the largest data type with Double giving a average between the two.

-System Diagram

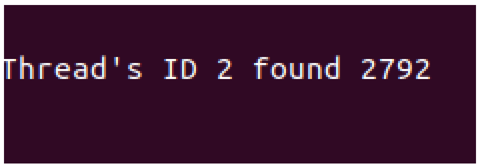
-Source Data Description
-Parallel Region Pseudocode

-Snapshot of Input Data (on each Thread)



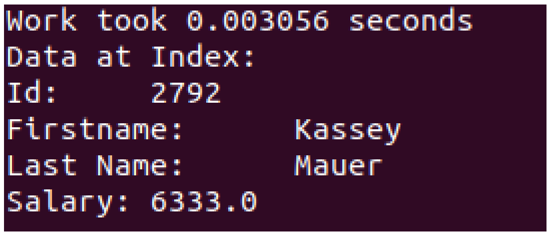
```
Starting Parallel search...

Thread 0 has indexs 0 to 2499
Thread 1 has indexs 2500 to 4999T
```

-Snapshot of Output Data (on each Thread)



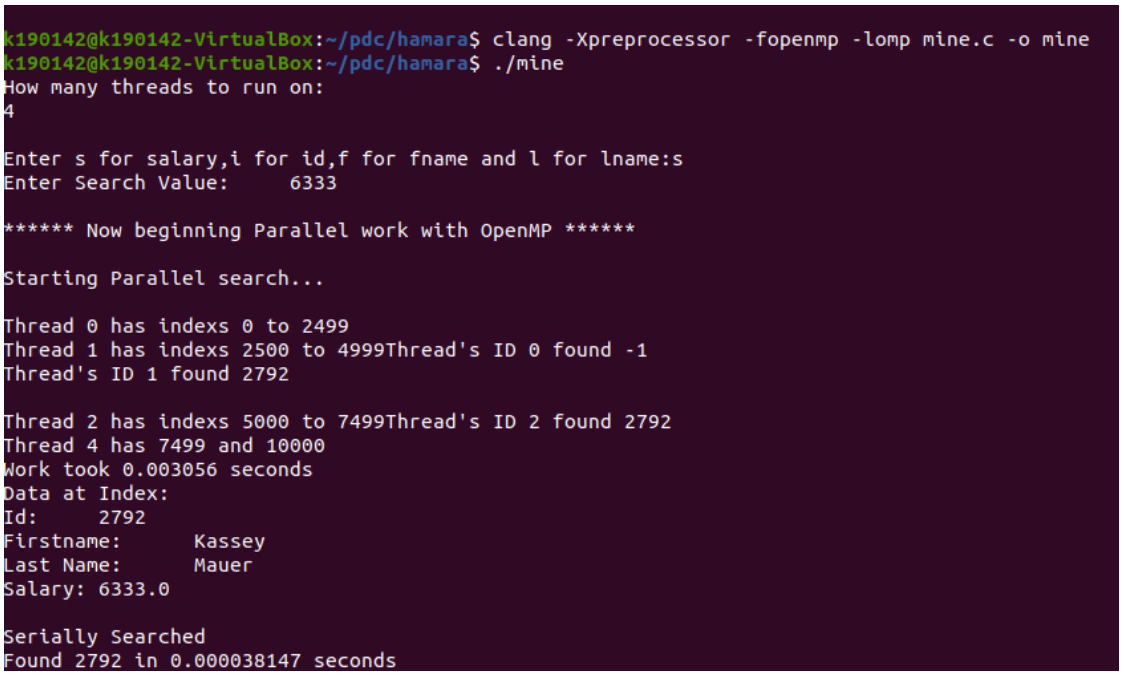
```
Thread's ID 2 found 2792
```

-Snapshots of Output Data (Final)



```
Work took 0.003056 seconds
Data at Index:
Id:     2792
Firstname:      Kassey
Last Name:      Mauer
Salary: 6333.0
```

Full Runtime Execution-



```
k190142@k190142-VirtualBox:~/pdc/hamara$ clang -Xpreprocessor -fopenmp -lomp mine.c -o mine
k190142@k190142-VirtualBox:~/pdc/hamara$ ./mine
How many threads to run on:
4

Enter s for salary,i for id,f for fname and l for lname:s
Enter Search Value:     6333

****** Now beginning Parallel work with OpenMP ******

Starting Parallel search...

Thread 0 has indexs 0 to 2499
Thread 1 has indexs 2500 to 4999Thread's ID 0 found -1
Thread's ID 1 found 2792

Thread 2 has indexs 5000 to 7499Thread's ID 2 found 2792
Thread 4 has 7499 and 10000
Work took 0.003056 seconds
Data at Index:
Id:     2792
Firstname:      Kassey
Last Name:      Mauer
Salary: 6333.0

Serially Searched
Found 2792 in 0.000038147 seconds
```

-Comparison Graphs (if Any)

Speeds of Searching

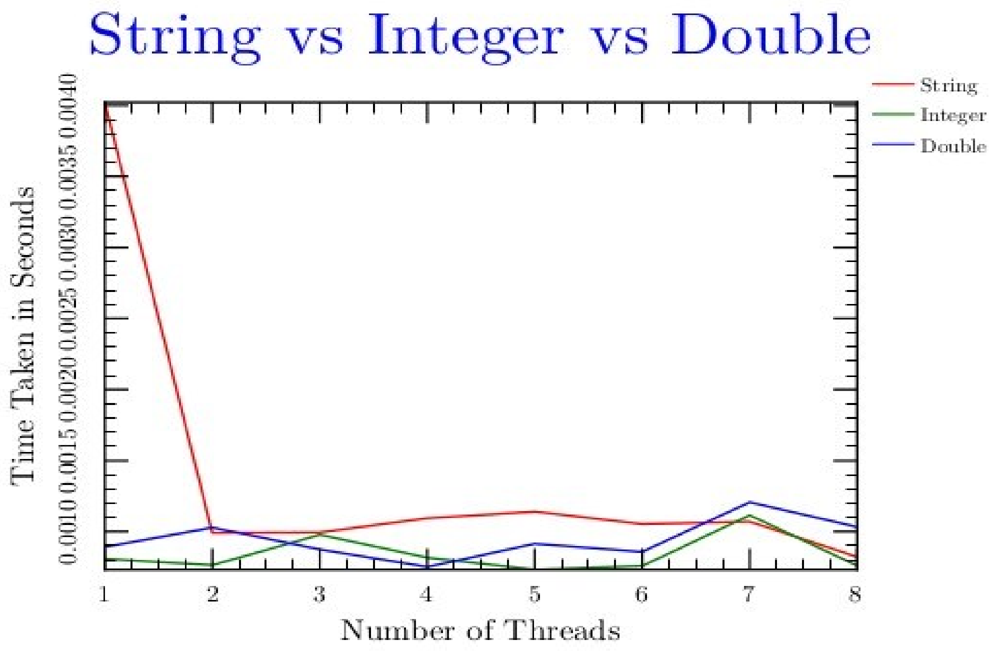


Figure 1. Speeds of searching a random data record by its respective attribute. String contains Searching speeds for First Name as searching for both First Name and Last Name would be bringing the same value within the range.

Conclusion: Speeds for Integer is indeed lowest but when number of threads is increasing their speeds all decrease exponentially with integer following double as a replica but faster than it with sting varying from faster than double to slowest. One crucial idea to note, when finding a parallel approach to a problem, is that the serial operation needs to be costly enough to computer resources to parallelize. Often times, complex calculations such as matrix multiplication or dot product are used to demonstrate parallel effectivity, for a specific reason. That is, the operation needs to be costly enough to outweigh the performance repercussions of a parallel solution. Specific performance repercussions such as thread spawning, excessive function calls, voltage and thermal limitations across several threads, and cache hierarchy issues can all play a part in slowing down performance. In the case of Linear Search, the operation simply is not costly enough to make useful out of a parallel approach. OpenMP is not a solution to all problems. .