OS Assignment 3

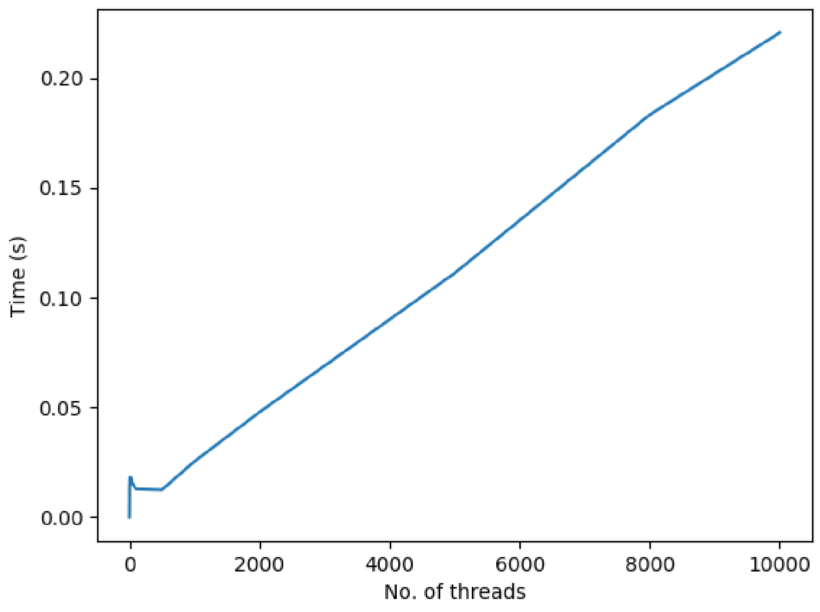
S. Vinu Sankar, 16110143

GIST link:
https://gist.github.com/vinusankarsiitgn/d062c9b0a890c2974b9a008fef690afc

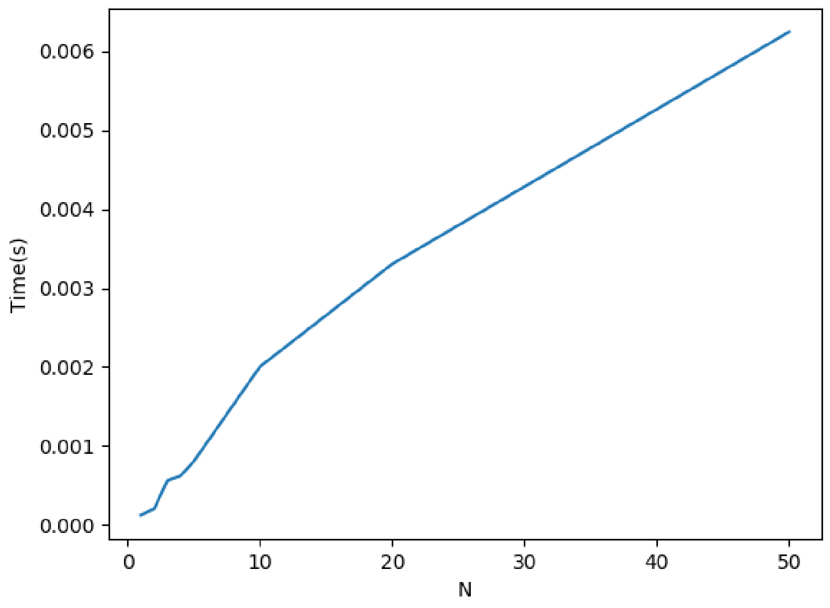
Q1.

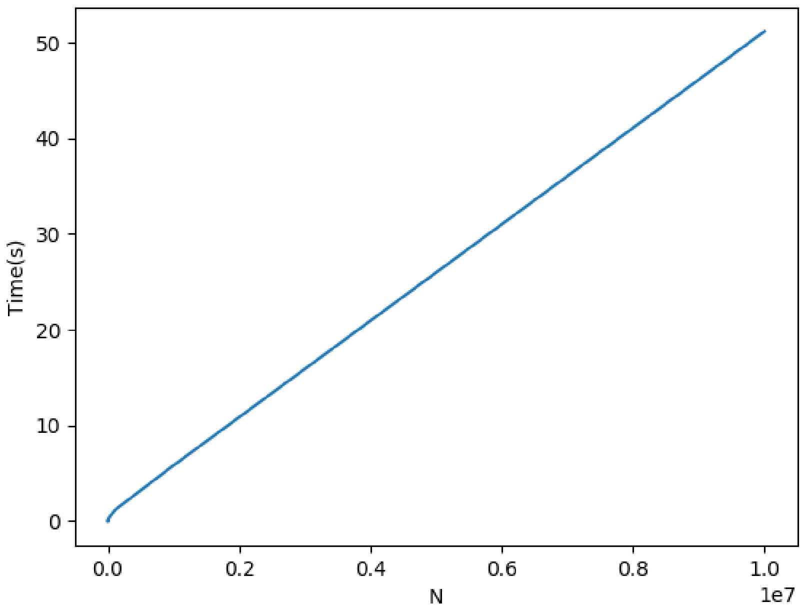
Q1_1.c

10,000 operations



Q1_2.c





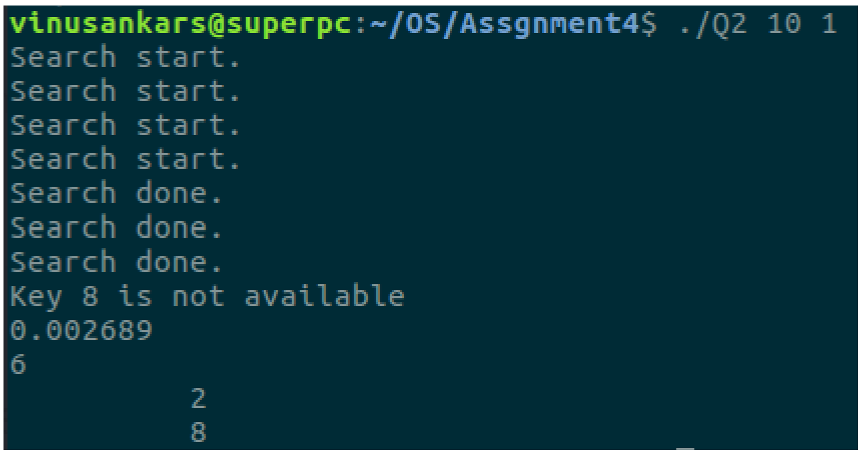
CPU configuration of my laptop is **Intel® Core™ i5-6200U CPU @ 2.30GHz × 4** .

The time taken for performing N operations is increasing linearly in general as the overhead increases due to time taken in destruction and creation of threads is a lot and adds up as number of threads increases. There is no direct relation of number of cores and threads in the plot. If the number of cores were more it would had taken lesser time, but still the time increases as number of threads increase, in a linear fashion.

Q2.

The insert and delete operations are writers, whereas the search operation is only reader as it does not edit the B-Tree. Search function happens more often than insert and delete, as they (insert and delete) use search function within them.

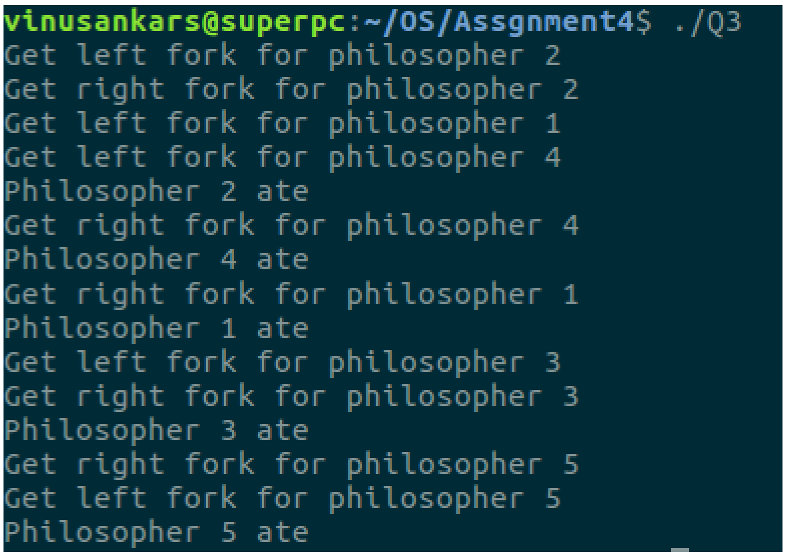
Hence we can provide a read-write lock which allows multiple reader threads. Basically, the pthread_rwlock lock will let multiple search threads to search simultaneously when no writing is happening. The ouput by the Q2.c shows that initiation and termination of various search (reader) threads can happen simulataneously.



```
vinusankars@superpc:~/OS/Assgnment4$ ./Q2 10 1
Search start.
Search start.
Search start.
Search start.
Search done.
Search done.
Search done.
Key 8 is not available
0.002689
6
          2
          8
```

Q3.

In secret GIST link given in header.



```
vinusankars@superpc:~/OS/Assgnment4$ ./Q3
Get left fork for philosopher 2
Get right fork for philosopher 2
Get left fork for philosopher 1
Get left fork for philosopher 4
Philosopher 2 ate
Get right fork for philosopher 4
Philosopher 4 ate
Get right fork for philosopher 1
Philosopher 1 ate
Get left fork for philosopher 3
Get right fork for philosopher 3
Philosopher 3 ate
Get right fork for philosopher 5
Get left fork for philosopher 5
Philosopher 5 ate
```

Q4.

In secret GIST link given in header.

The given solution suffers from starvation. The getfork() can be called to acquire a fork by only a philosopher at a time as it is within a lock m. Hence there will be waiting for philosophers while some other philosopher is getting fork as the lock is acquired. If philosopher 0 is eating and philosopher 1 wants to get a fork, it might have to wait for a while if it takes philosopher 0 more time to eat. This can potentially make philosopher 1 go to sleep without getting the fork, but acquiring the lock. So other philosophers can not also eat as the lock is with philosopher 1. This makes the text book solution more concurrent than the solution given. Hence starvation happens in solution provided in question. Both codes avoid dead locks.

References:

1. OS – 3 easy steps

2. https://github.com/AnthonyBobsin/2-3Binary-Tree/blob/master/btree.c