Operating Systems-2

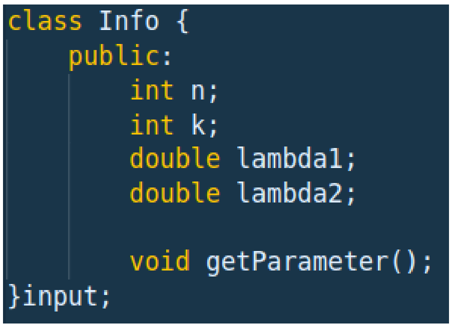
Report-Assignment3

Implementing TAS, CAS and Bounded Waiting CAS

Mutual Exclusion Algorithms

We have to implement TAS, CAS and Bounded Waiting with CAS
mutual exclusion (ME) algorithms in C++.

1. First of all I have defined a class namely *Info* which have the parameters *n, k, lambd1, lambda2.* It stores the data from file *inp-params.txt* into object *input*.

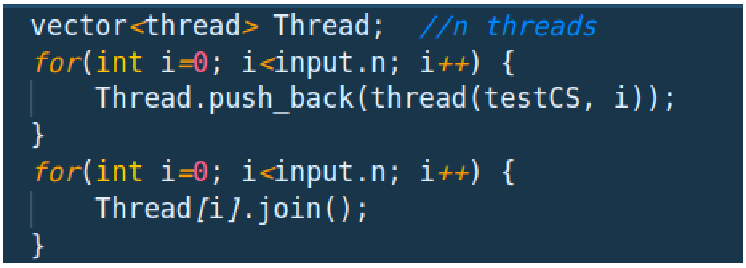


```
class Info {
    public:
        int n;
        int k;
        double lambda1;
        double lambda2;

        void getParameter();
}input;
```

About function ***getParameter*** : Assign all parameters from the input file to class members.

2. *Thread* is a vector of threads. Here I have passed *testCS* function to each thread along with the thread id and pushing it to Thread vector.



```
 vector<thread> Thread;   //n threads
 for(int i=0; i<input.n; i++) {
     Thread.push_back(thread(testCS, i));
 }
 for(int i=0; i<input.n; i++) {
     Thread[i].join();
 }
```

About function **testCS** : It takes thread id as parameter. And we will pass this function along with the thread id to each thread and calculate the average waiting time and max waiting time for all the three mutual exclusion algorithms.

3. *exponential_distribution* is used to generate random number which is basically a floating point value and it described by probability function:

$$\mathbf{P(x|k) = ke^{-kx}}$$

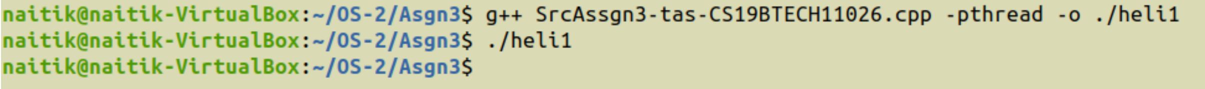
distribution1 and distribution2 are 2 exponential distributions defined inside main.

4. Few inbuilt functions and libraries were used to calculate average waiting time(*avgWaitingTime*) and maximum waiting time(*maxWaitingTime*).
Example: a. std::chrono::time_point
b.std::chrono::system_clock, std::chrono::system_clock::now()
c. struct tm - Time structure

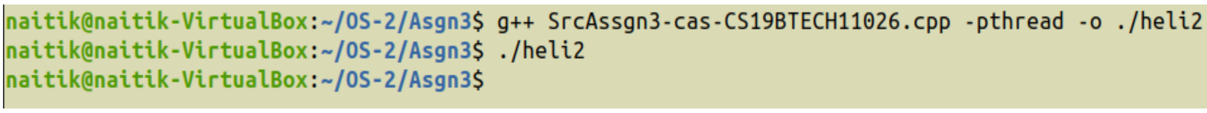
5. *usleep* is used to suspend the execution of a thread by few microseconds.

6. Few functions of atomic library is used like:
a. *atomic_flag_test_and_set* in TAS implementation.
b. *atomic_compare_exchange_strong* in CAS and CAS-Bounded implementation.

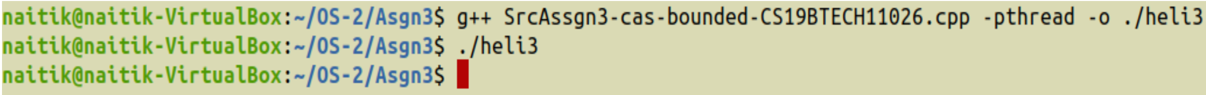
7. Compilation Screenshot



```
naitik@naitik-VirtualBox:~/OS-2/Asgn3$ g++ SrcAssgn3-tas-CS19BTECH11026.cpp -pthread -o ./heli1
naitik@naitik-VirtualBox:~/OS-2/Asgn3$ ./heli1
naitik@naitik-VirtualBox:~/OS-2/Asgn3$
```

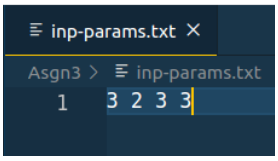


```
naitik@naitik-VirtualBox:~/OS-2/Asgn3$ g++ SrcAssgn3-cas-CS19BTECH11026.cpp -pthread -o ./heli2
naitik@naitik-VirtualBox:~/OS-2/Asgn3$ ./heli2
naitik@naitik-VirtualBox:~/OS-2/Asgn3$
```



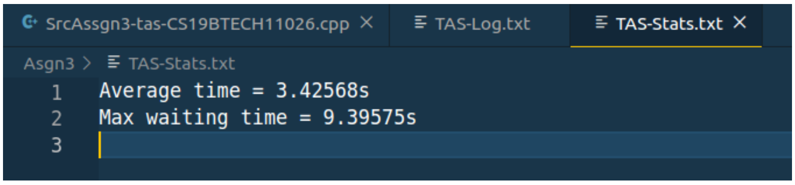
```
naitik@naitik-VirtualBox:~/OS-2/Asgn3$ g++ SrcAssgn3-cas-bounded-CS19BTECH11026.cpp -pthread -o ./heli3
naitik@naitik-VirtualBox:~/OS-2/Asgn3$ ./heli3
naitik@naitik-VirtualBox:~/OS-2/Asgn3$
```

8. For input parameter :

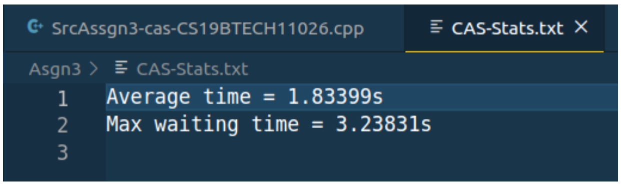


```
3 2 3 3
```

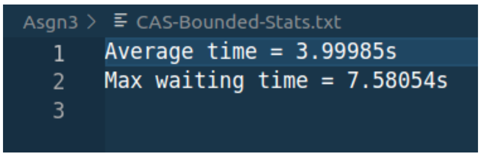
Stats outputs are as follows:



```
Average time = 3.42568s
Max waiting time = 9.39575s
```

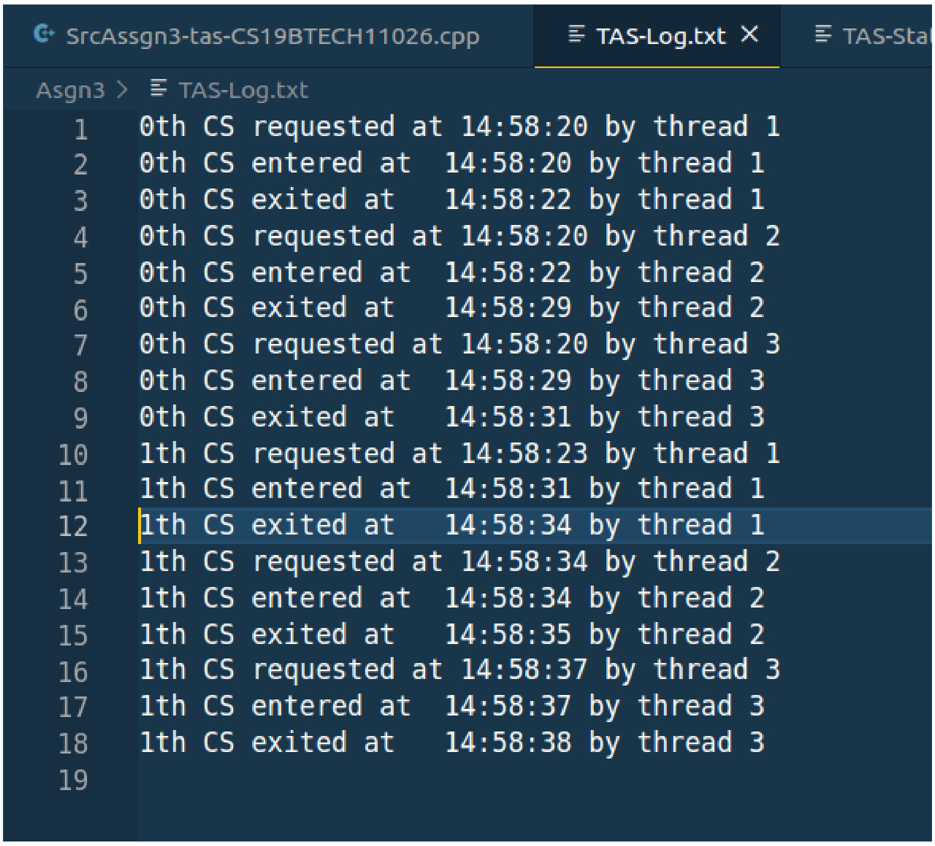


```
Average time = 1.83399s
Max waiting time = 3.23831s
```

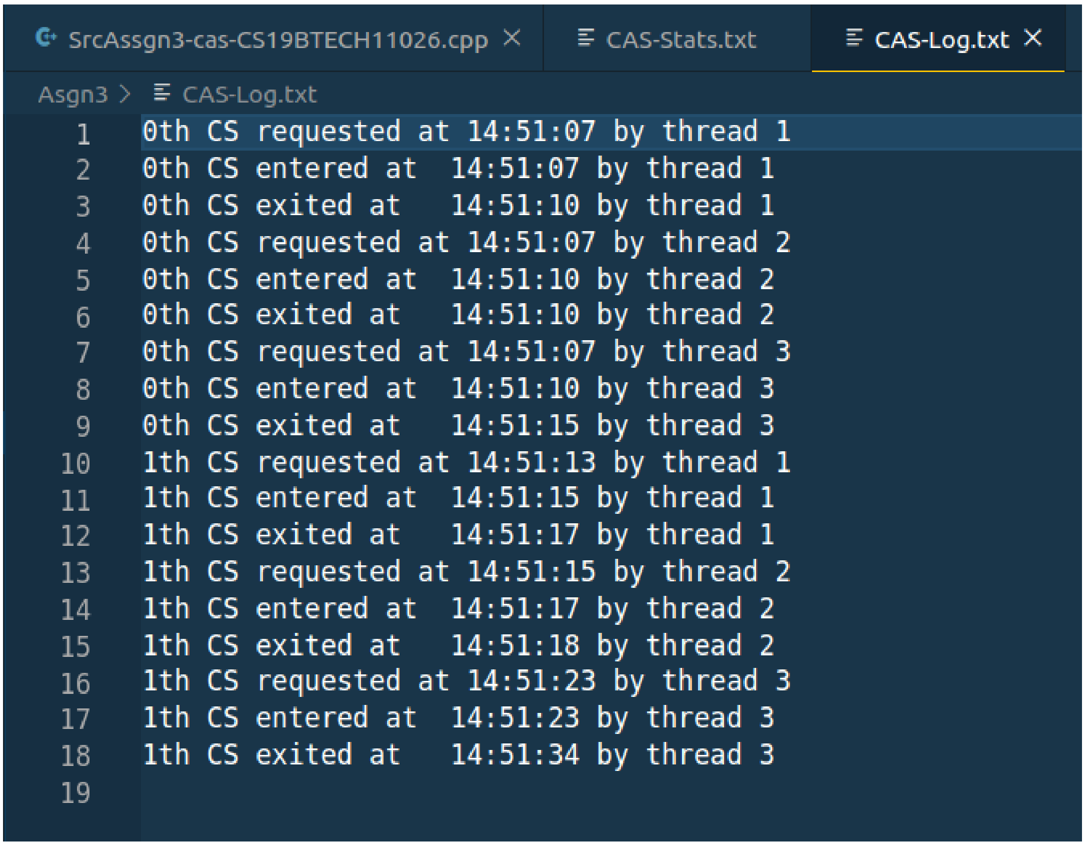


```
Average time = 3.99985s
Max waiting time = 7.58054s
```

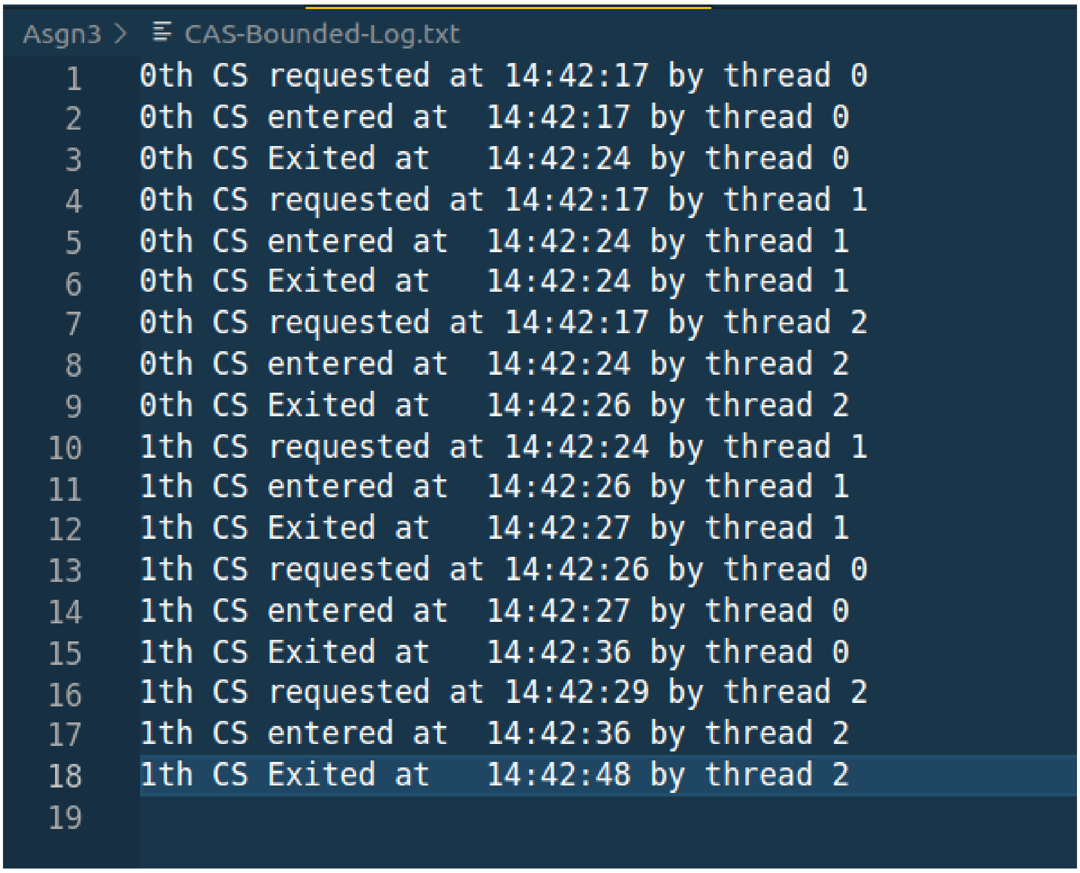
And Log outputs are as follows:



```
0th CS requested at 14:58:20 by thread 1
0th CS entered at  14:58:20 by thread 1
0th CS exited at   14:58:22 by thread 1
0th CS requested at 14:58:20 by thread 2
0th CS entered at  14:58:22 by thread 2
0th CS exited at   14:58:29 by thread 2
0th CS requested at 14:58:20 by thread 3
0th CS entered at  14:58:29 by thread 3
0th CS exited at   14:58:31 by thread 3
1th CS requested at 14:58:23 by thread 1
1th CS entered at  14:58:31 by thread 1
1th CS exited at   14:58:34 by thread 1
1th CS requested at 14:58:34 by thread 2
1th CS entered at  14:58:34 by thread 2
1th CS exited at   14:58:35 by thread 2
1th CS requested at 14:58:37 by thread 3
1th CS entered at  14:58:37 by thread 3
1th CS exited at   14:58:38 by thread 3
```



```
0th CS requested at 14:51:07 by thread 1
0th CS entered at  14:51:07 by thread 1
0th CS exited at   14:51:10 by thread 1
0th CS requested at 14:51:07 by thread 2
0th CS entered at  14:51:10 by thread 2
0th CS exited at   14:51:10 by thread 2
0th CS requested at 14:51:07 by thread 3
0th CS entered at  14:51:10 by thread 3
0th CS exited at   14:51:15 by thread 3
1th CS requested at 14:51:13 by thread 1
1th CS entered at  14:51:15 by thread 1
1th CS exited at   14:51:17 by thread 1
1th CS requested at 14:51:15 by thread 2
1th CS entered at  14:51:17 by thread 2
1th CS exited at   14:51:18 by thread 2
1th CS requested at 14:51:23 by thread 3
1th CS entered at  14:51:23 by thread 3
1th CS exited at   14:51:34 by thread 3
```



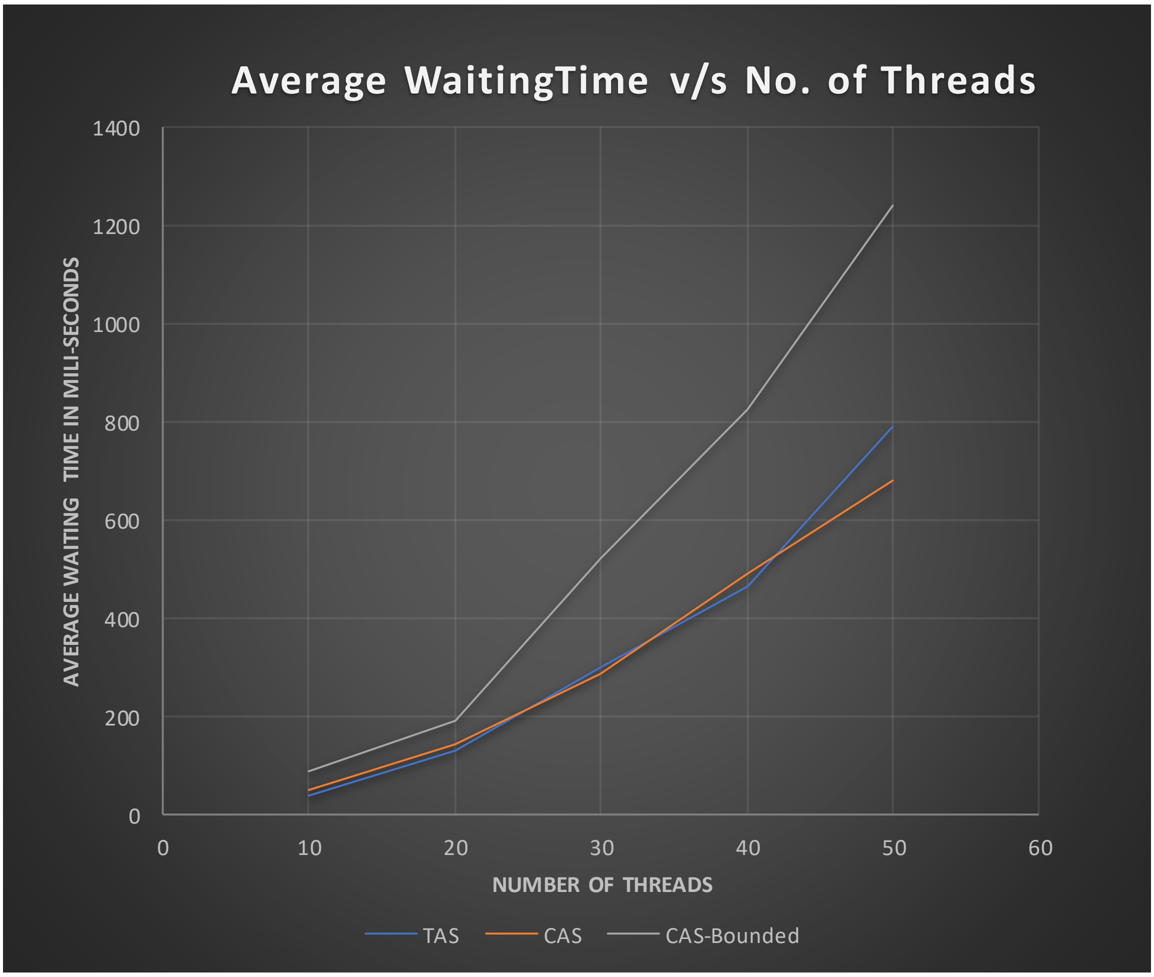
```
0th CS requested at 14:42:17 by thread 0
0th CS entered at  14:42:17 by thread 0
0th CS Exited at   14:42:24 by thread 0
0th CS requested at 14:42:17 by thread 1
0th CS entered at  14:42:24 by thread 1
0th CS Exited at   14:42:24 by thread 1
0th CS requested at 14:42:17 by thread 2
0th CS entered at  14:42:24 by thread 2
0th CS Exited at   14:42:26 by thread 2
1th CS requested at 14:42:24 by thread 1
1th CS entered at  14:42:26 by thread 1
1th CS Exited at   14:42:27 by thread 1
1th CS requested at 14:42:26 by thread 0
1th CS entered at  14:42:27 by thread 0
1th CS Exited at   14:42:36 by thread 0
1th CS requested at 14:42:29 by thread 2
1th CS entered at  14:42:36 by thread 2
1th CS Exited at   14:42:48 by thread 2
```

Graph1:

n	TAS	CAS	CAS-Bounded
10	38.4986	49.9623	88.1069
20	129.963	142.6972	192.1808
30	298.8874	286.4648	520.9872
40	464.6982	488.7217	826.7776
50	790.1129	680.1632	1240.1819

All the values are computed after running each program 5 times....
i.e. avgWaitingTime is itself of 5 avgWaitingTimes.

Conclusion: 1. Average waiting time taken by Bounded-CAS is greater than both TAS and CAS.

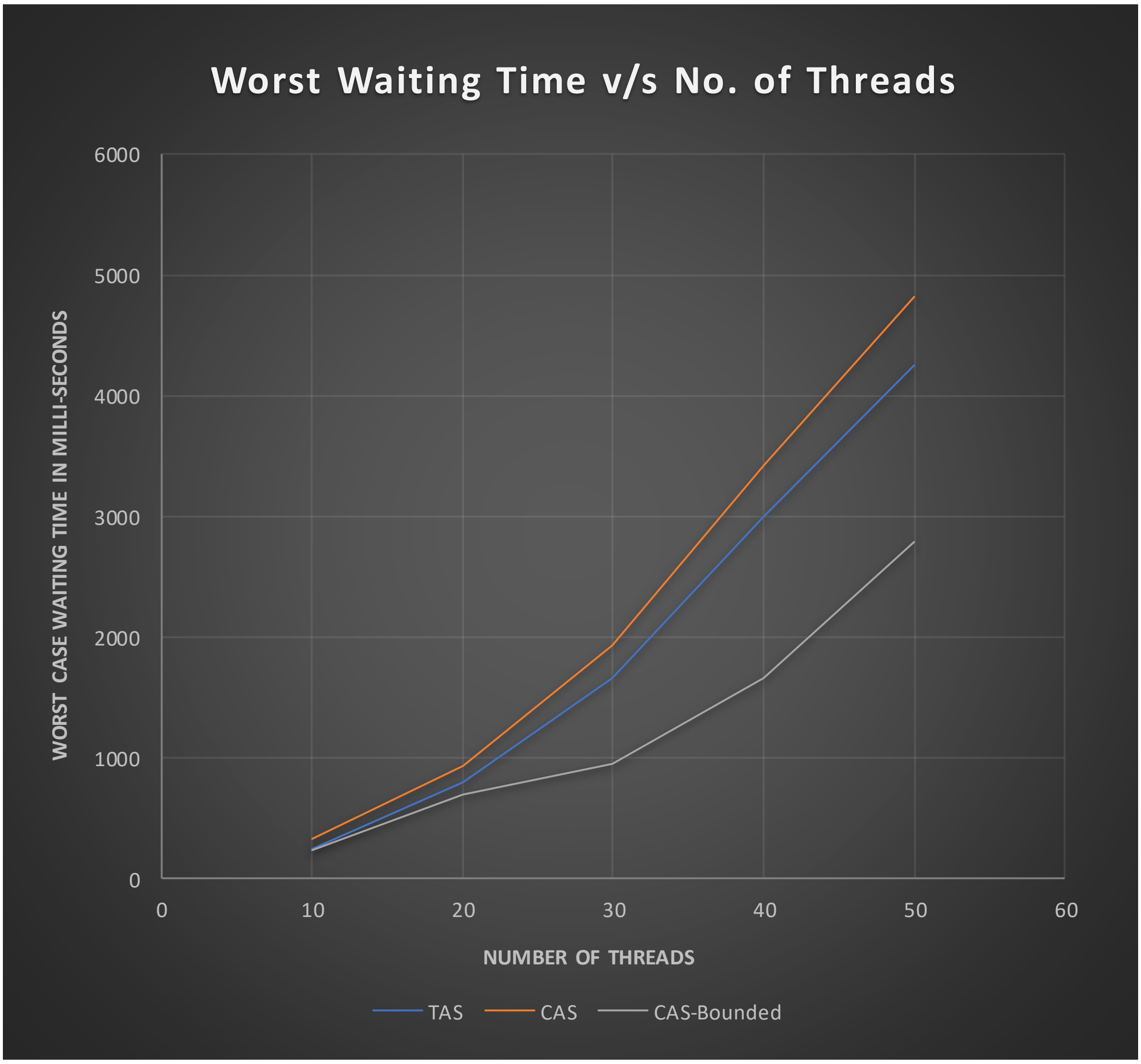


Graph2:

N	TAS	CAS	CAS-Bounded
10	243.1448	326.7564	232.734
20	798.9943	937.6274	685.4098
30	1655.981	1927.651	954.5693
40	2995.985	3420.879	1666.889
50	4248.856	4815.914	2785.982

Conclusion: 1. Worst waiting time taken by Bounded-CAS is much less than both TAS and CAS.

2. The difference between the worst waiting times of Bounded-CAS and CAS, TAS increases as the no. of threads increases.


Worst Waiting Time v/s No. of Threads
WORST CASE WAITING TIME IN MILLI-SECONDS
NUMBER OF THREADS
TAS
CAS
CAS-Bounded