Name: Vishal Vijay Devadiga

Roll Number: CS21BTECH11061

Code Flow

- Opens input file in main and get value of n(number of threads), k(number of iterations of critical section), and average values of sleep function λ_1 and λ_2
- Create n threads that execute the TestCS function.
- Each thread waits till it enters the critical section, sleeps then exits the critical section, sleeps then repeats till it finishes all iterations of critical section
- The two algorithms, TAS(Test and Serve) and CAS(Compare and Swap) are implemented by the std library of C++. The mutual exclusion part of BCAS(Bounded Compare and Swap) is implemented by me.

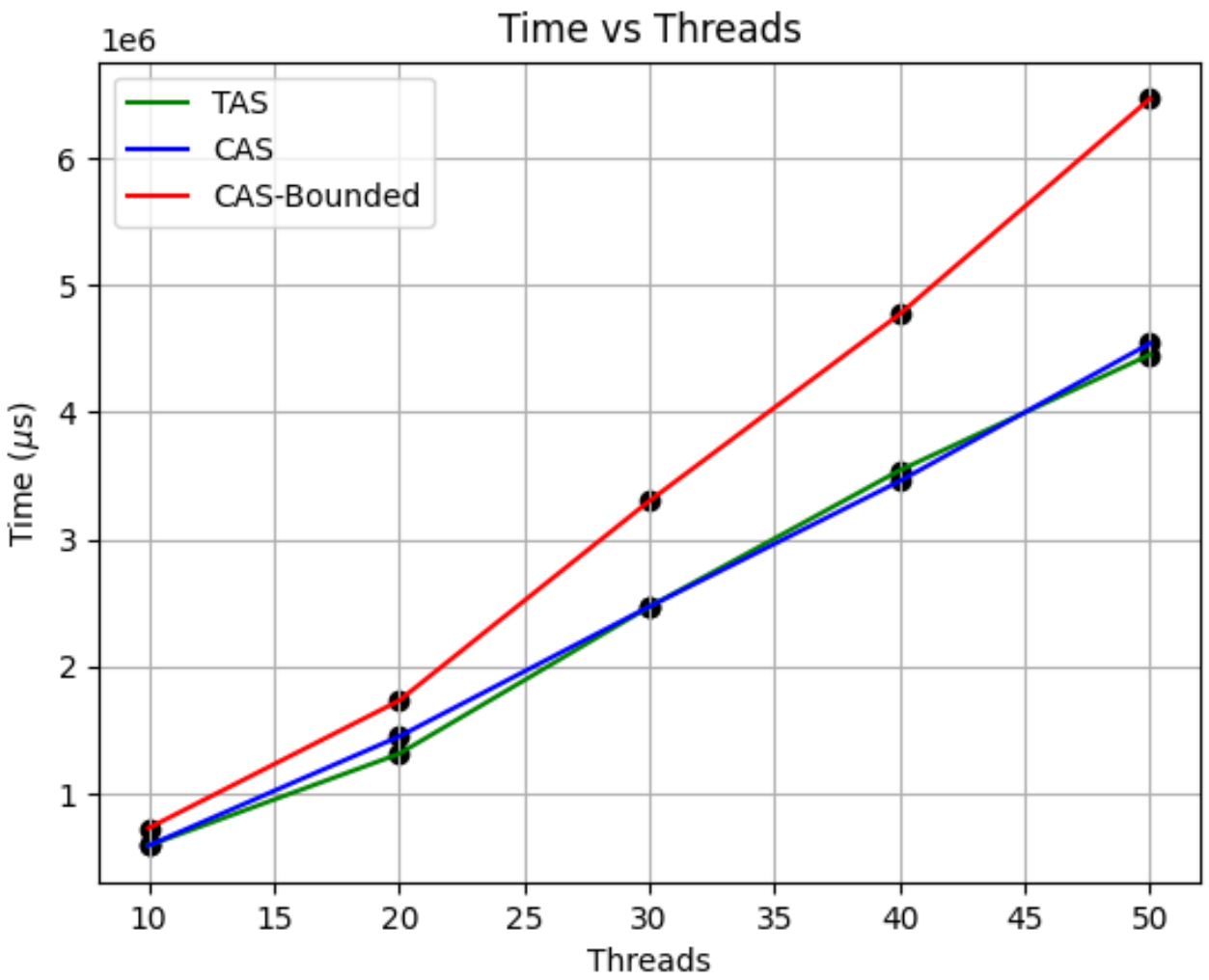
Plots

My system has the processor AMD Ryzen 7 5800u, which has 8 cores and 16 threads. Note that **all measurements of time are in microseconds(μs)** Values for $\lambda_1 = 20ms$ and $\lambda_2 = 40ms$

Some Points

`fprintf`, `ofstream` are thread-safe depending on the implementation(compiler). It is thread-safe on linux and windows. Thus, there was no need to use a mutex while writing to the output file. Mutexes on a calculative block of code significantly reduces the speedup(**Amadahl's law**).

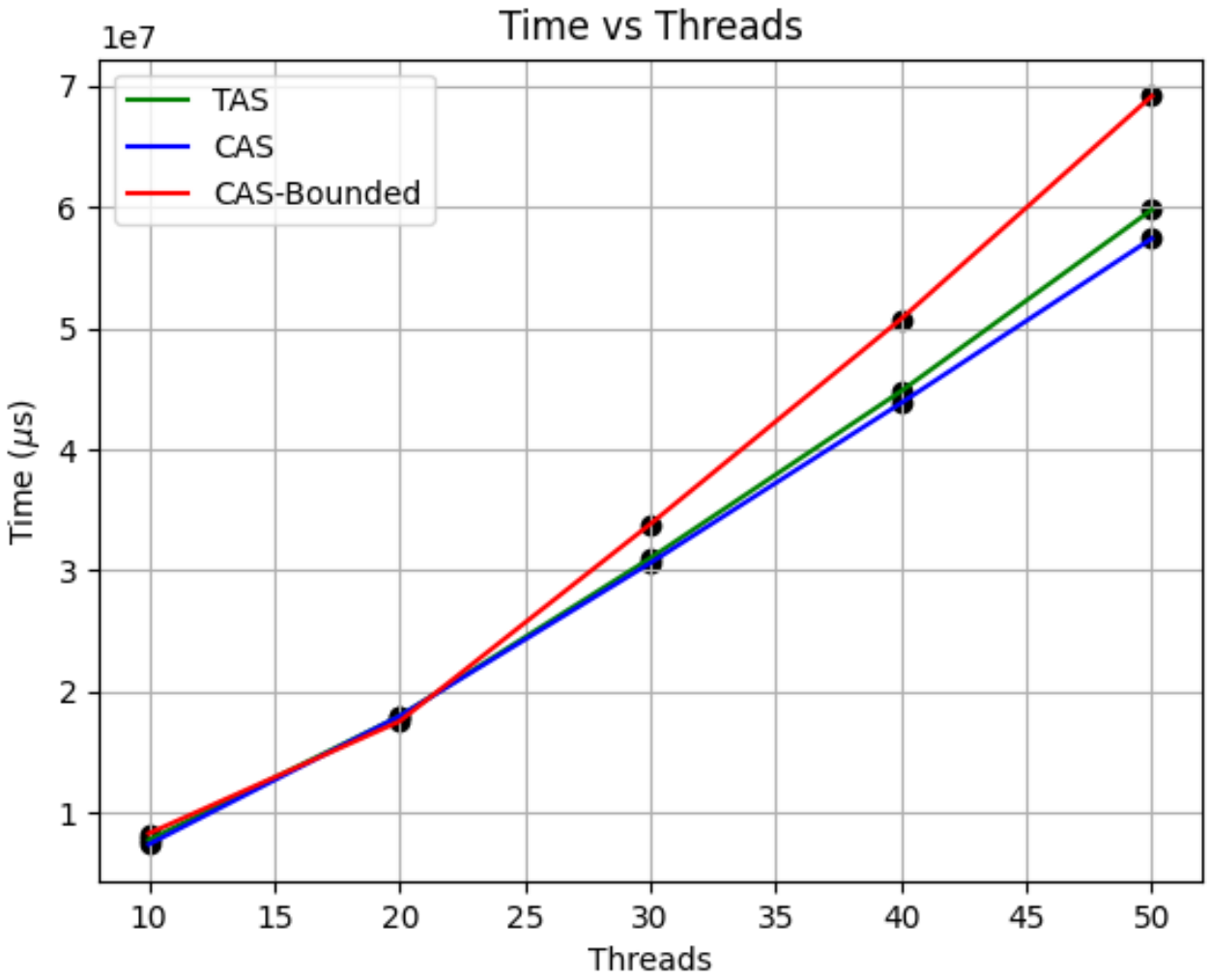
Threads vs Average Time Taken



Algorithm	$n = 10$	$n = 20$	$n = 30$	$n = 40$	$n = 50$
TAS	592360.25	1313320.5	2469710.75	3542727.0	4452966.0
CAS	592046.5	1449637.5	2471364.75	3458167.25	4541104.0
BCAS	729806.0	1731231.25	3302061.5	4771305.75	6465414.5

TAS and CAS are comparable in the average time taken. BCAS ensures that a thread does not starve, thus it takes a lot more time compared to the former two.

Threads vs Worst Time Taken



Algorithm	$n = 10$	$n = 20$	$n = 30$	$n = 40$	$n = 50$
TAS	7787857	17987242	31026030	44795671	59762862
CAS	7424892	17998843	30628876	43819711	57387426
BCAS	8319932	17567932	33838852	50749163	69134056

Theoretically, BCAS should have a lower worst case waiting time since it avoids starvation by checking by going in order of threadNo(if it is waiting for CS at that specific moment).

However, it really depends on the value of lambda given in this program. In this case, CAS and TAS have a lower worst case time than BCAS, due to extra calculations and checking done in BCAS to ensure mutual exclusion, when it was not really needed for these values. If $\lambda_2 <<< \lambda_1$, that is, if λ_2 was much lower than λ_1 then probably BCAS would have had a much lower worse case waiting time than CAS and TAS as the latter 2 would not have been able to ensure mutual exclusion, and would have a process starve.