Bilkent University

Department of Computer Engineering

CS 342 - Operating Systems
Spring 2024
Project 2 Report

Görkem Kadir Solun - 22003214 - Sec 1

Cahit Ediz Civan - 22003206 - Sec 1

Efe Kaan Fidancı - 22102589 - Sec 1

Gökalp Gökdoğan - 22102936 - Sec 2

Design

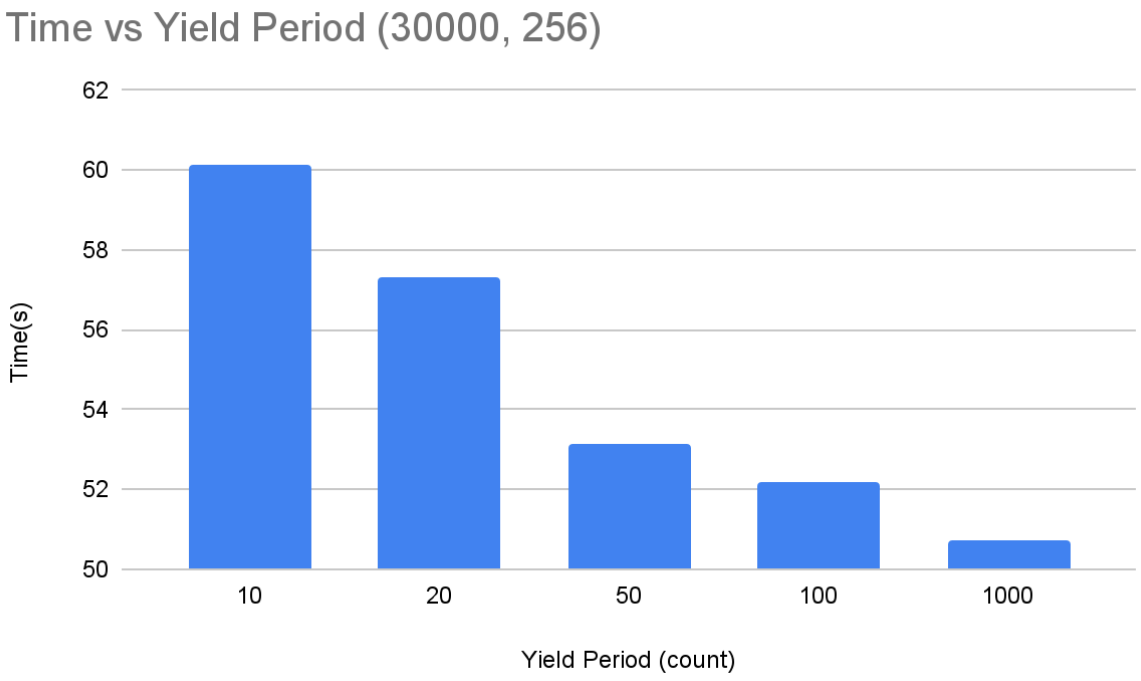
To evaluate the capabilities of our thread library, we created a basic counting application. This app counts from 1 to a specified number. Additionally, we can include instructions like yielding to a random or chosen thread at the Xth count (not excluding terminated threads, meaning it may not yield to a terminated thread), exiting the program, or canceling a random thread (again, it may choose previously terminated threads). This counting algorithm was developed to thoroughly test our library's functionalities. Finally, we use join to ensure that all counting threads are completed. We have conducted different tests to calculate context switch overhead occurred while yielding and how the thread count affects the runtime.

Experiments

To calculate context switch overhead, we made tests with different yield periods and threads. These tests are both conducted with FCFS algorithm.

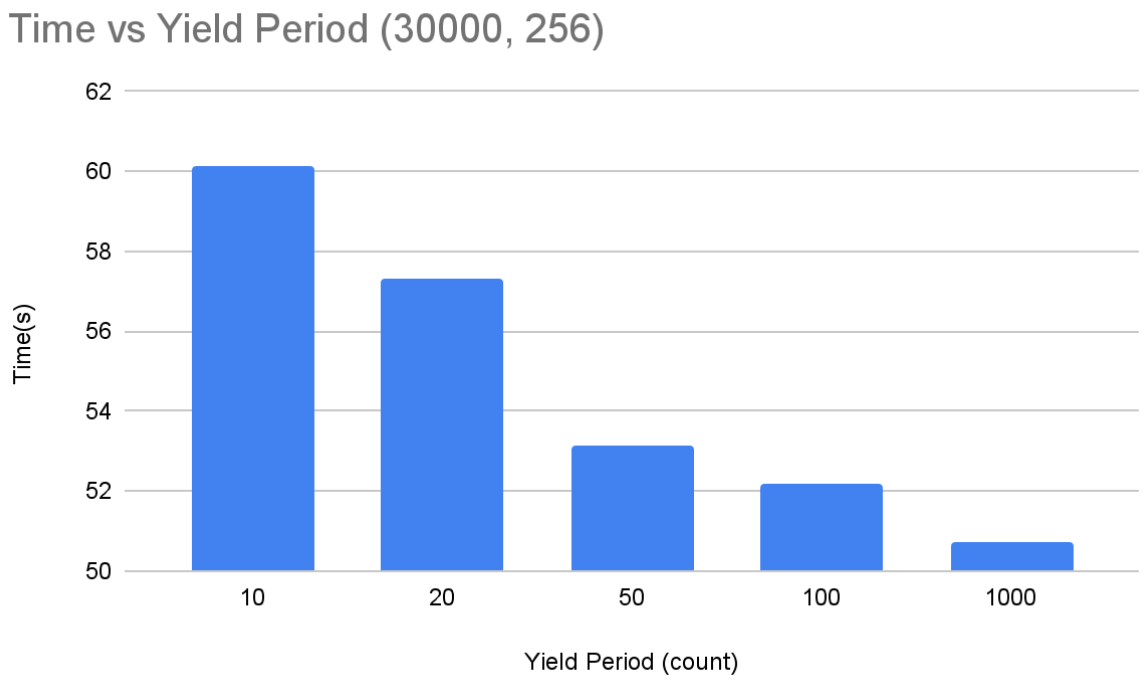
Counting to 10000 with 256 Threads

Yield Period (count)	Time (s)
10	17.96432
20	16.65467
50	16.17986
100	15.77556
1000	14.53452



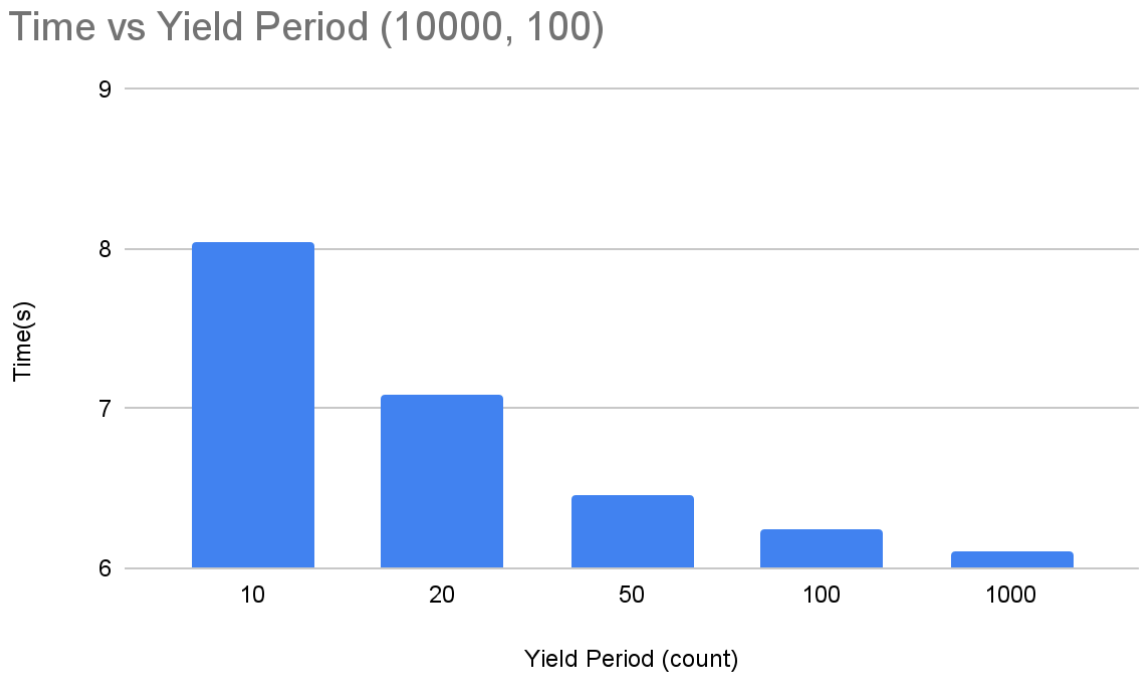
Counting to 30000 with 256 Threads

Yield Period (count)	Time (s)
10	60.13452
20	57.33445
50	53.14356
100	52.20983
1000	50.74567



Counting to 10000 with 100 Threads

Yield Period (count)	Time (s)
10	8.04567
20	7.08452
50	6.4627
100	6.23987
1000	6.10345



In these experiments, we observed a variation in overhead time across different yield periods. When counting up to 30000, we encountered a significant difference of up to 10 seconds. This overhead difference was reduced to 4 seconds when counting up to 10000. We also determined the variation in time in relation to thread count. As expected, reducing the thread count led to a corresponding decrease in execution time. We have also tried to make experiments with random scheduling algorithm but there were no difference against FCFS algorithm for this projectsIn addition, we attempted to conduct experiments using a random scheduling algorithm. However, in the context of this project, we observed no significant differences in performance compared to the FCFS algorithm.

Conclusions

Our experiments primarily focused on calculating the context switch overhead in threads and the thread count. Context switch overhead represents the time taken by a CPU to switch between executing different threads. This overhead can be significant, particularly in systems with numerous running threads. During our experiments, we observed an increase in time when we increased the thread count (note that we executed the same program in each thread) and when we increased the yield period. The results align with our expectations, demonstrating the effectiveness of our library.