Ezgi Çakır – 21402334
Metehan Kaya – 21401258

REPORT

In this report, results of 4 algorithms will be shown. For fixed values on N (number of disks) random number of requests was given to our program as input. Values of N are 10, 100, 1000, 10000. We generated the inputs in such a way that there is a strong positive relationship between N (the number of cylinders) and M (the number of requests).

N	10	100	1000	1000
Execution time (s)	0.003	0.004	0.065	3.733

Table 1: Total time needed to execute the C program

Algorithms	Head Movement	Avg waiting time	Std. deviation of waiting time
FCFS	48	19.5	11.276820
SSTF	18	5.9	4.976612
LOOK	20	5.3	4.738729
CLOOK	22	5.5	4.169999

Table 2: Time results when N=10

Algorithms	Head Movement	Avg waiting time	Std. deviation of waiting time
FCFS	4914	2349	1438.757635
SSTF	200	98.368056	43.396166
LOOK	200	98.368056	43.396166
CLOOK	291	175.194444	59.131745

Table 3: Time results when N=100

Algorithms	Head Movement	Avg waiting time	Std. deviation of waiting time
FCFS	513283	255559.863309	147281.077153
SSTF	2225	669.999346	486.665307
LOOK	1999	685.575540	482.594550
CLOOK	2943	1020.597122	708.839275

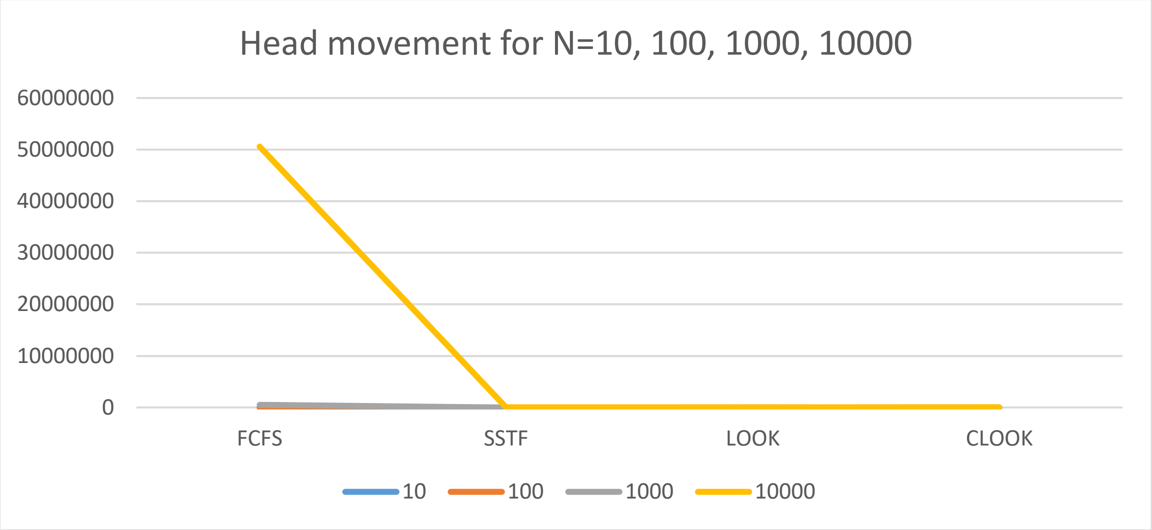
Table 4: Time results when N=1000

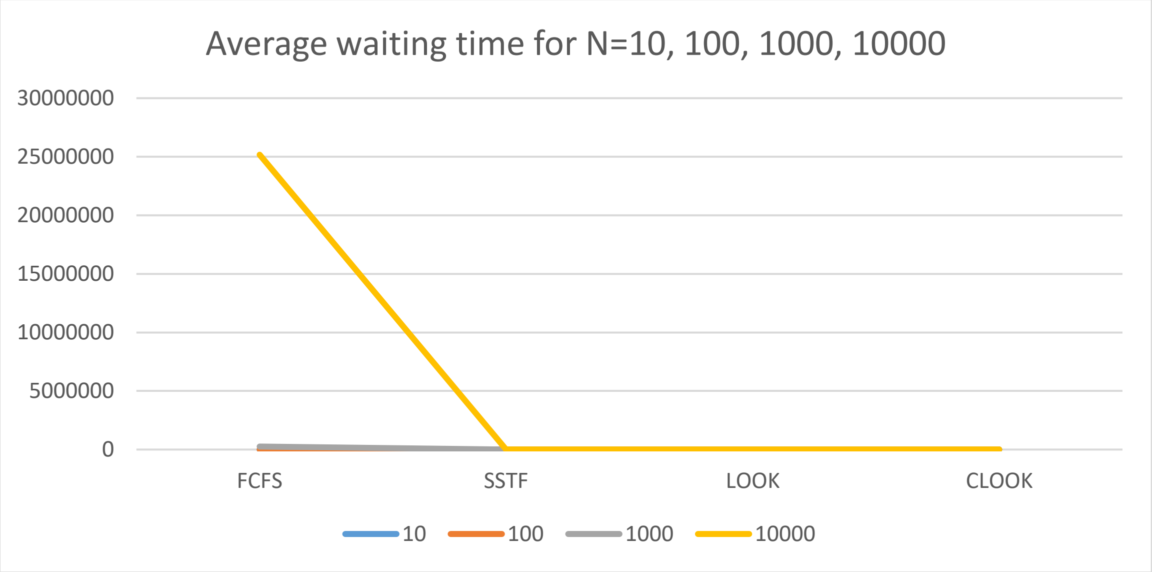
Algorithms	Head Movement	Avg waiting time	Std. deviation of waiting time
FCFS	50581409	25174719.300881	14569250.249337
SSTF	20003	7818.254821	5092.589451
LOOK	19999	7815.415269	5091.467340
CLOOK	29849	11538.651534	7188.788487

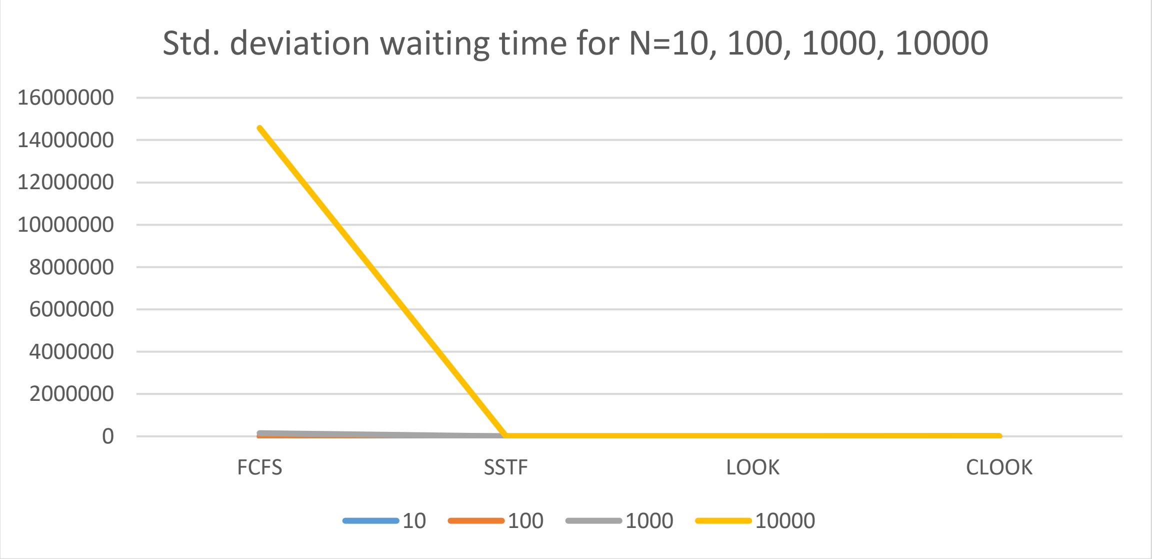
Table 5: Time results when N=10000

Ezgi Çakır – 21402334
Metehan Kaya – 21401258

Plots that are drawn respect to results which are seen above.







Ezgi Çakır – 21402334
Metehan Kaya – 21401258

Discussion

As we can realize from Table 1, total time to execute the whole code increases exponentially when value of M, the number of requests, increases exponentially. The reason is that except FCFS, each algorithm runs in O(M^2) time, where FCFS runs in O(M). Also, execution did not last long when M=10000, because of low constant of the time complexity.

Note (not important): For remaining 3 algorithms, it is possible to find waiting times of the requests in a better time complexity. Assume that we are implementing the code in C++. We can use C++'s *set* which is a *STL structure* and a binary balanced tree and has functions called *lower_bound* and *upper_bound* that runs in O(logM) time and these can be used to find closest cylinders to the current head.

As we can realize from Table 2, Table 3, Table 4, Table 5, and/or by three plots, FCFS algorithm has the worst case in terms of head movement and waiting time of requests. On the other hand, other three algorithms which are SSTF, LOOK, and CLOOK have not dramatic changes in terms of head movement and waiting time of requests.

Note: If we look SSTF and LOOK rows of the Table 4, we can observe that there is no strong positive relation between head movement and average waiting time of the requests.

When we give random inputs, since FCFS makes the head move forward and backward by not considering distance to go, it can be observed that in the most cases FCFS is the worst algorithm. However, in some rare cases, FCFS may be a right choice. Suppose that the number of cylinders is a high number and there is relationship between the requests where the requests with low cylinder number comes earlier. In that case, since waiting time of requests do not change for different algorithms, it can be beneficial to use FCFS in terms of execution time of algorithm.

Among the other three algorithms, CLOOK is observed as the worst algorithm. In the most cases, its head movement and average waiting time for requests more than the others' which are SSTF an LOOK. The reason is that CLOOK goes all way from the right-most cylinder to the left-most cylinder. Therefore, not considering cylinders between the left-most and the right-most cylinders may lead to inefficiency in terms of time. However, in some specific cases, it can be sensible to use CLOOK algorithm.

Note: Before the experiment, we thought that LOOK is much better than SSTF since SSTF is a too simple greedy algorithm. However, we faced with an unexpected outcome where the data for these algorithms was very close to each other.