CS 342

FALL 2019

PROJECT #2

PROJECT REPORT

BERK YILDIZ

21502040

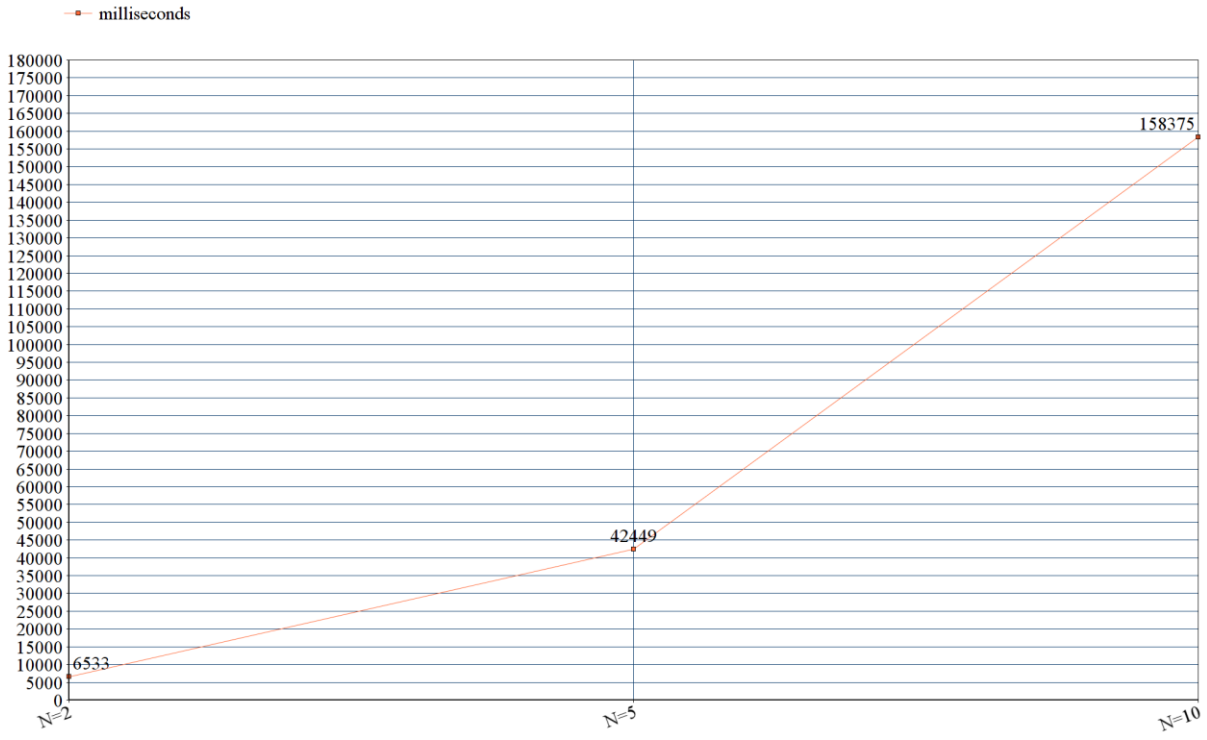
I did my experiments with a computer which has processor intel core i5 vPro. The computer is seven years old and it is a slow machine when compared to today's computers. I used two different file sizes which have 1000 integers and 5000 integers respectively.

Thread Sync Results

Findings for filesize with 1000 integers:

	k = 100	**k = 500**	**k = 900**
N = 2	6533ms	7584ms	9463ms
N = 5	42449ms	38853ms	42298ms
N = 10	158375ms	153911ms	157098ms

Table 1: Findings for filesize with 1000 integers (Thread – Sync)

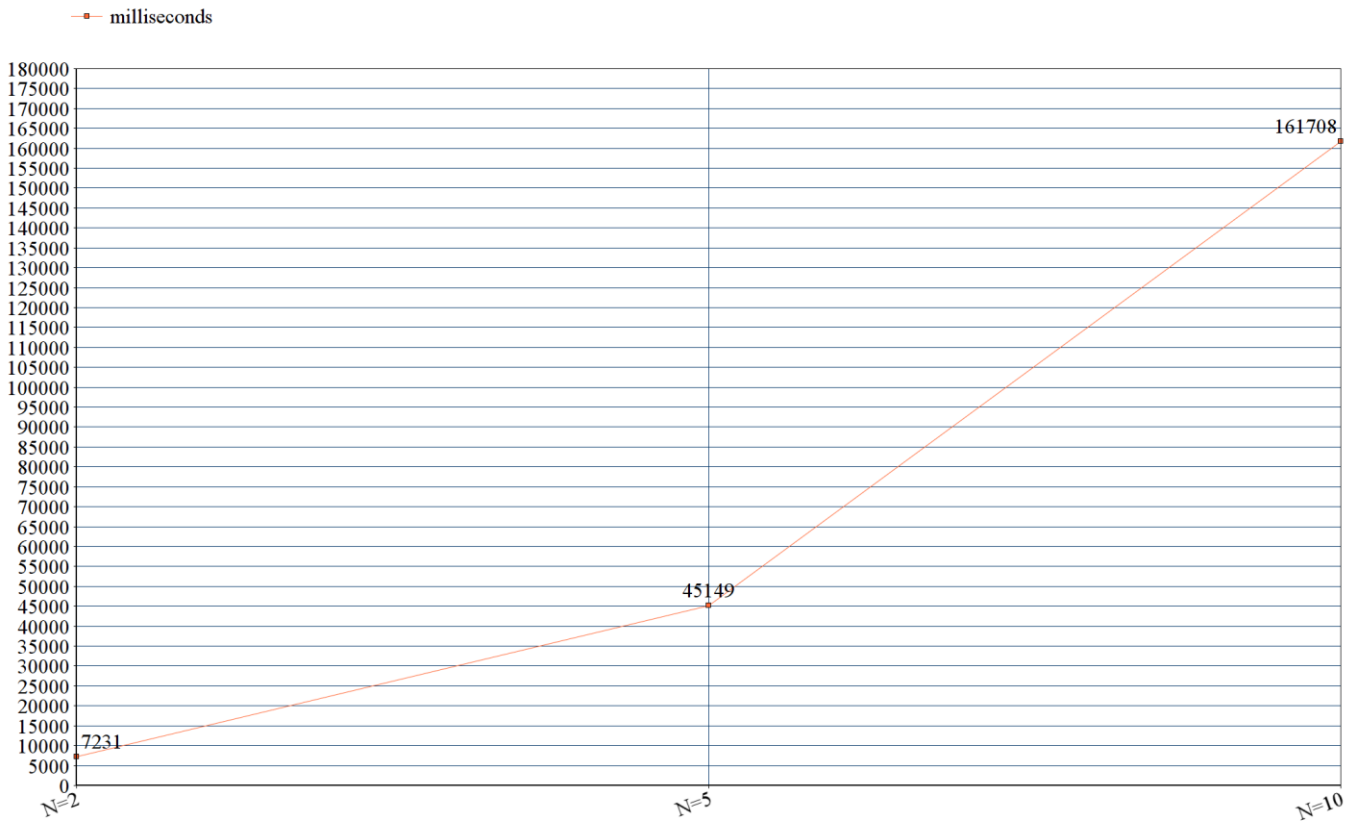


Graph1: Findings for filesize=1000 integers, k=100

Findings for filesize with 5000 integers:

	k = 100	k = 1000	k = 4000
N = 2	7231ms	8089ms	9211ms
N = 5	45149ms	46178ms	48239ms
N = 10	161708ms	158567ms	165231ms

Table 2: Findings for filesize with 5000 integers (Thread – Sync)



Graph2: Findings for filesize=5000 integers, k=100

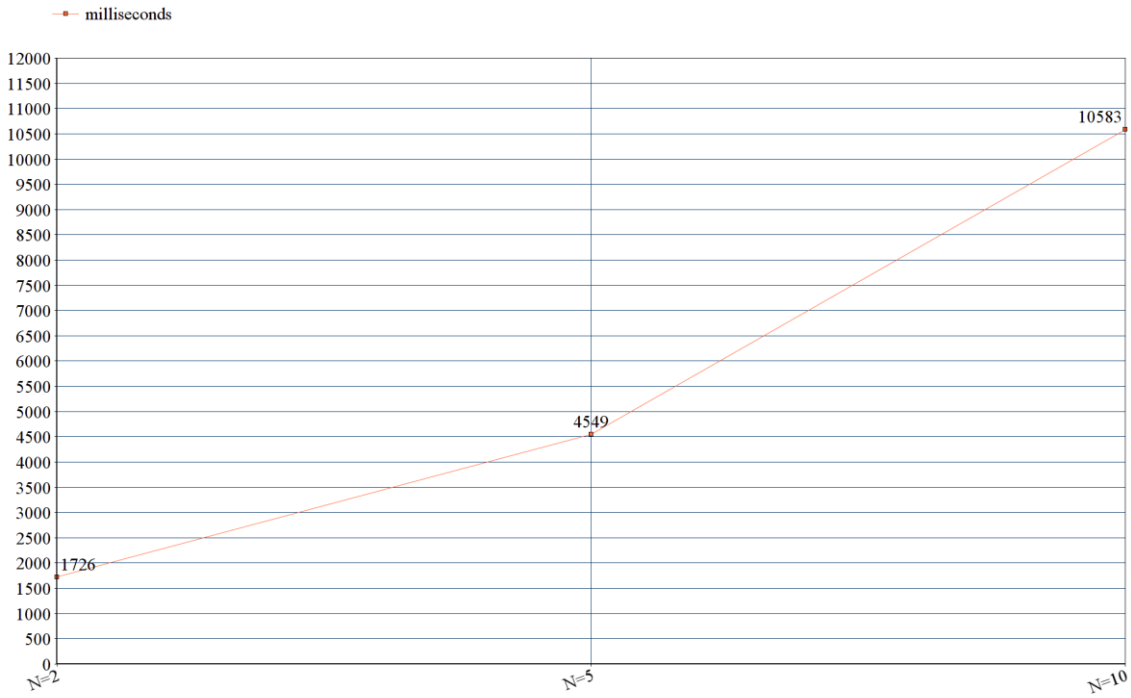
***My program is giving error without synchorinzation, so I could not handle experiment for unsynchronized threads**

Process Sync/No Sync Results

Findings for filesize with 1000 integers:

	k = 100	k = 500	k = 900
	Sync / No Sync	**Sync / No Sync**	**Sync / No Sync**
N = 2	1726ms/2264ms	3081ms/3243ms	4465ms/5097ms
N = 5	4549ms/4394ms	10810ms/10985ms	13761ms/14934ms
N = 10	10583ms/13701ms	18709ms/26952ms	20765ms/31165ms

Table 1: Findings for filesize with 1000 integers (Process)

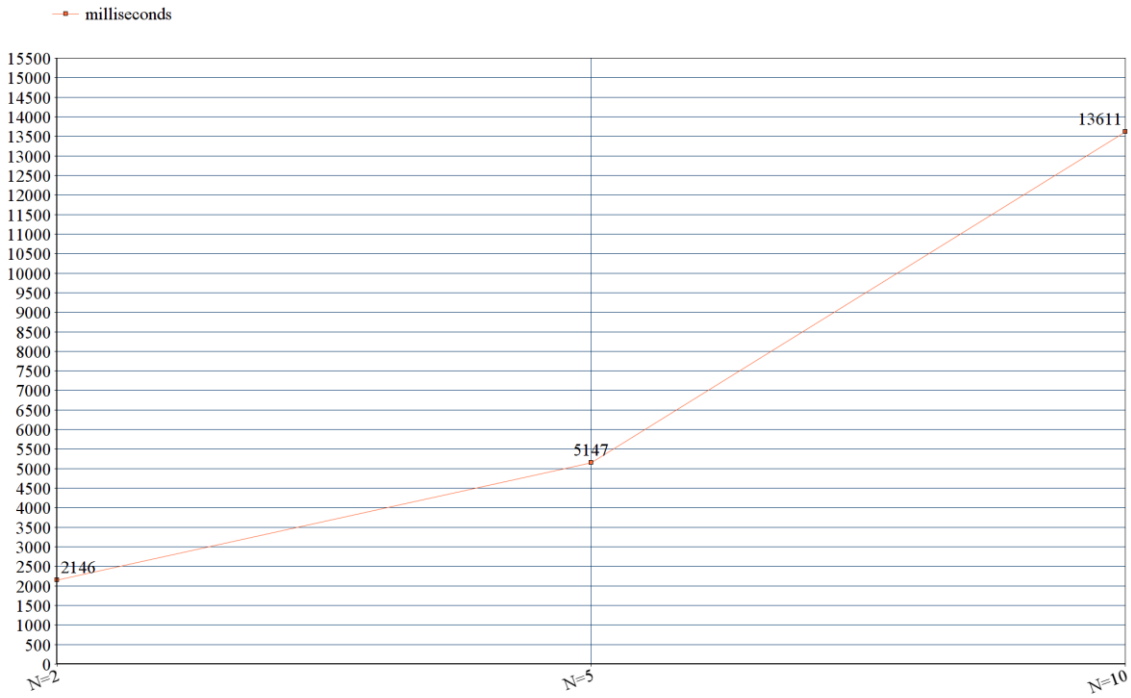


Graph3: Findings for filesize=1000 integers, k=100

Findings for filesize with 5000 integers:

	k = 100	k = 500	k = 900
N = 2	2146ms/2446ms	3512ms/3893ms	4945ms/5511ms
N = 5	5147ms/5312ms	12965ms/13676ms	16227ms/17549ms
N = 10	13611ms/14901ms	20112ms/24456ms	23615ms/31912ms

Table 2: Findings for filesize with 1000 integers



Graph4: Findings for filesize=5000 integers, k=100

As the result of the experiment the dominant linearity in the graphs are visible. As the number of input files increases with constant k value, execution time increases directly proportional. The reason behind linearity is already CPU executes one file at a time and there is not any module which will reduce the execution time of CPU in the program and in my computer. So as the number of data increases, time for execution increases too. Also there is not an visible affect of filesize on execution time. Also synchorinaziton of processes

do not reduce execution time significantly. Another point is that, implementation of array is less costly than a binary search tree. There is significant difference in terms of execution time between threads and processes where threads are implemented by binary search tree and processes implemented by an array.