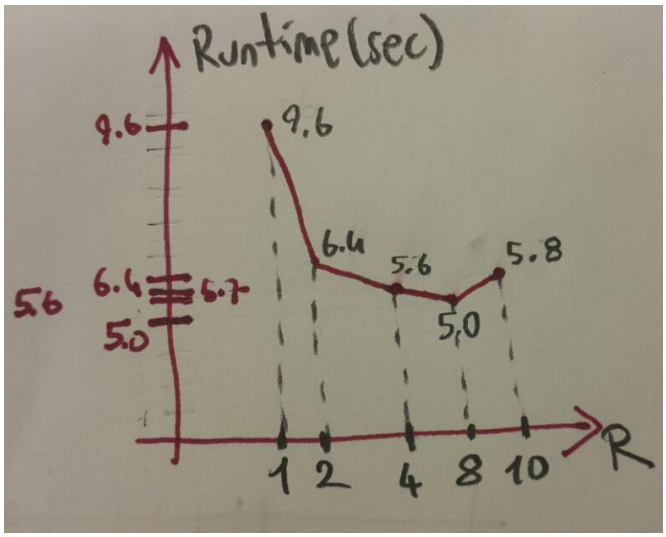
CS342 Project 2 - Report
Doğukan Yiğit Polat
21401797

Runtime graph for changing R (N = 10):



Measurement method:

I have run the time command in bash to do the measurements. I compared "real" part of the output.

Observations:

As we can see clearly, runtime decreases when we increase R until R = 8.

It increases with R after that point.

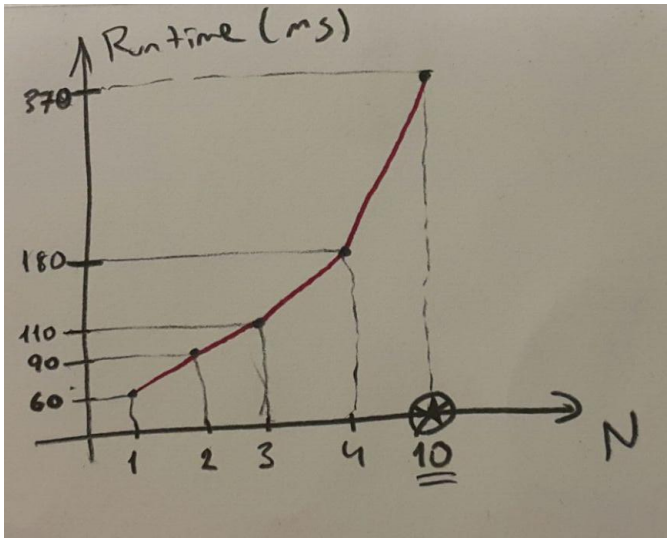
Another observation is that the runtime is not decreasing linearly with R.

Discussions:

Decrease in runtime can be explained with utilization of a multi-processor system. Non-linearity is a result of **Amdahl's Law**. According to Amdahl's law, runtime improvement is only dependent on the parallelizable part of the code. Since we have non-parallel parts in the code, run-time improvement is bounded by the serial part of the code.

Runtime increase after R=8 is a result of the system that I am running this experiment. It has 4 cores with 2x**Hyper-Threading**. So we can run only 8 threads concurrently at the same time with **no context switches** happening between them. If we increase R furthermore than 8, we will have **more I/O operations** since we create N times more files (hard-drive operations are costly) and we will have **context switches** between the threads. So these will cause the program to run slower.

Runtime graph for changing N (R = 2 and same input file):



Measurement method:

I have run the time command in bash to do the measurements. I compared “real” part of the output.

Observations:

Run-time increased -almost- linearly with N.

Discussions:

Keeping R value same among the experimental runs cause the run time to increase with respect to N. This is because X more files only add X times more linear-time I/O operations as workload.