Multicore Lab One

1. **Fill out two tables**

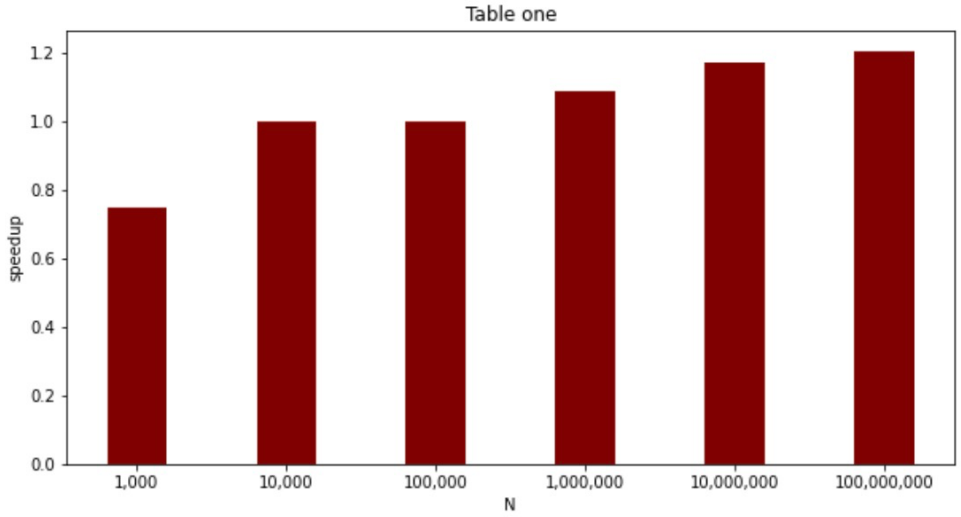
Table 1 contains the *real* part of the time command.

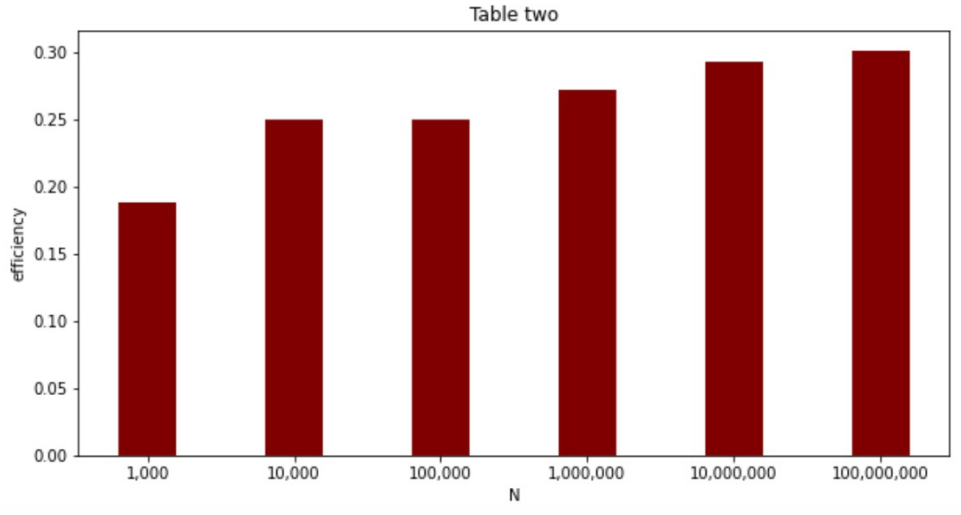
N	1,000	10,000	100,000	1,000,000	10,000,000	100,000,000
Seq	0.003	0.004	0.007	0.037	0.329	3.245
1-thread	0.003	0.004	0.008	0.038	0.338	3.344
4-thread	0.004	0.004	0.007	0.034	0.281	2.696

Table 2 contains the *cpu* + *sys* part of the time command.

N	1,000	10,000	100,000	1,000,000	10,000,000	100,000,000
Seq	0.003	0.003	0.006	0.036	0.328	3.244
1-thread	0.003	0.004	0.007	0.037	0.337	3.343
4-thread	0.004	0.005	0.015	0.103	0.961	9.491

2. **Draw two bar-graphs**





3. Answer the following questions:

(1) I have used crunchy 1 to do the experiment.

(2) The relationship between N and the speedup is that speedup becomes larger as N increases.

(3) Based on Amdahl's law Speedup = 1/[F+(1-F)/p] = 1/[(1-1/p)*F+1/p], when N increases, the proportion of parallelable tasks increases while the sequential tasks remain the same, so the fraction of the program that is sequential decreases. Therefore, speedup increases. And in terms of overhead reduction, when the problem size is small, the overhead can dominate the total execution time, reducing the potential speedup. However, as the problem size increases, the overhead becomes a smaller proportion of the total time spent on executing the parallelizable portion of the problem. This could lead to better speedup as the threads spend more time on actual computation rather than managing overhead.

(4) The *real* time was mostly smaller to *cpu* + *sys* time.

(5) Real time represents the actual elapsed time a program takes from start to finish, as if you were measuring time with a stopwatch. CPU time, on the other hand, measures the total time spent by the CPU executing a program, including user time (time spent executing user code) and system time (time spent performing kernel-level tasks related to the program). In a multithreaded (parallel) program, we can run multiple threads simultaneously on different cores of the CPU. Therefore, each core works independently to execute its program part. In this case, CPU time represents the sum of the time spent by all cores executing the program. Therefore, it is possible that the CPU time (*cpu*+*sys* time) is greater than the *real* time since multiple cores work at the same time.

(6) As the size of the problem increases, so does the efficiency.

(7) Since in graph 2, we are discussing the relationship between efficiency (efficiency= speedup / 4) and N (problem size), so the reason is same as question 3.