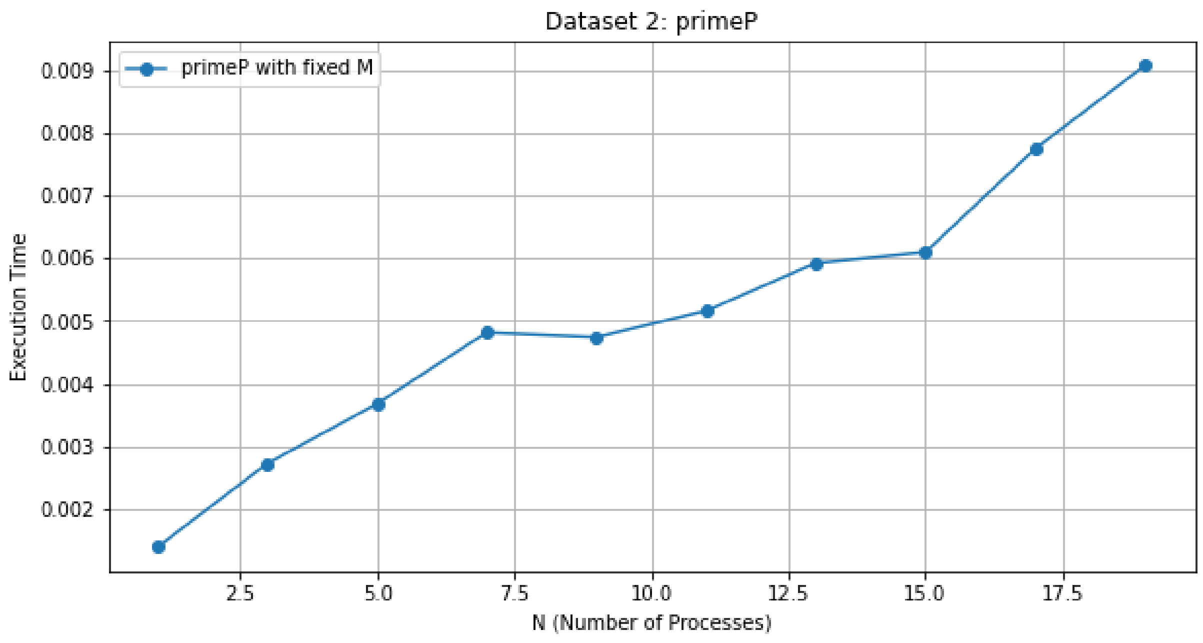
## Mert Gençtürk 22003506

## Experiment Results:

For each program and its experiments, an input file that contains 100 integers from 1 to 1000 has been used and 37 of these integers were prime. Execution times given below are in nanoseconds. For primeP program, two different experiments had been conducted:

### a) primeP – Number of Processes

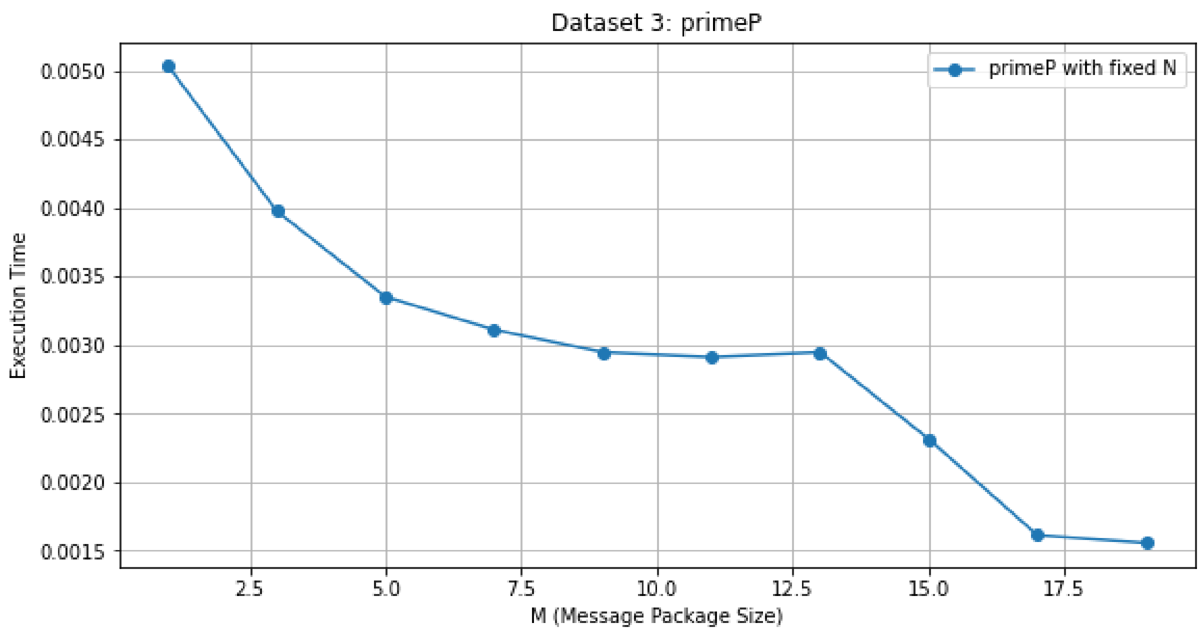
In the first experiment, number of M hold fixed to default value 3. Number of processes tested in different odd values in range 1 to 19.



Results have showed that execution time of the program had increases along with number of processes. The one of the reasons for that is the time lost in context switch overhead. Since the virtual machine these experiments had been done is allocated with only 2 cores, increasing number of processes couldn't be allocated to CPU. Therefore, number of context switches had risen up, resulting with more overhead and increased execution time. If the multiprocessor capabilities were better, more parallelism can be achieved with increasing number of processes. However, I think that since system capacities are not enough for efficient parallelism, more processes only resulted with overhead, without adding and executional benefit.

### b) primeP – Size of Message Packages

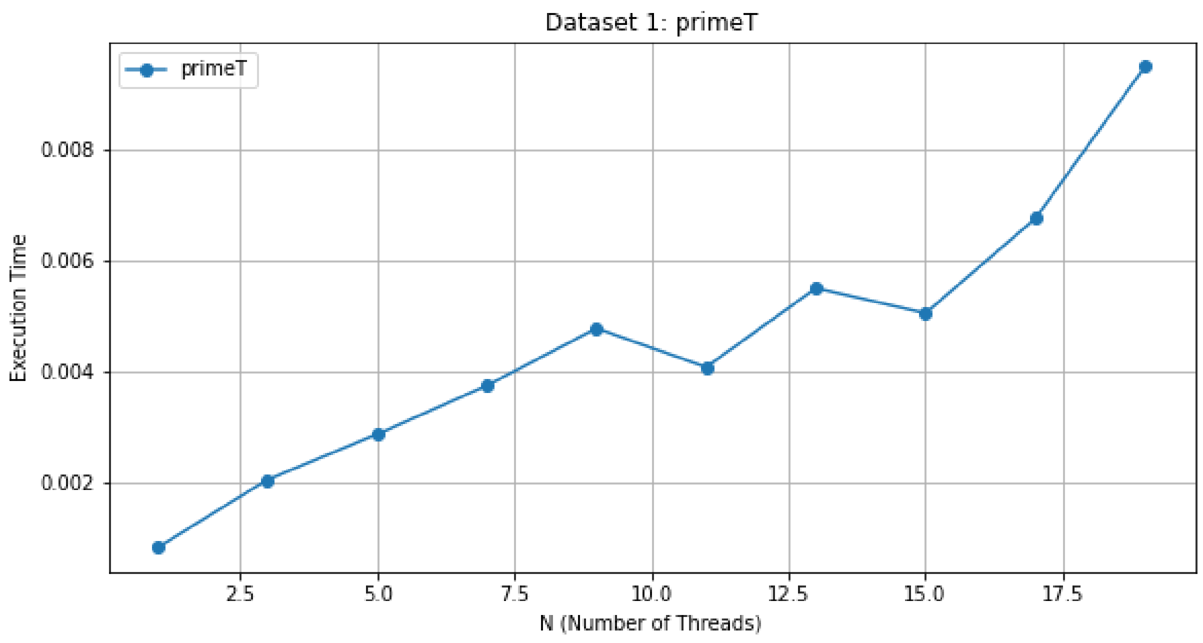
In the second experiment, number of N hold fixed to default value 5. Size of message packages tested in different odd values in range 1 to 19.



Since size of message packages directly effects the number of messages sent per execution, increase in size directly effects execution time too. While size of package is increased, number of messages is going to decrease, resulting with less overhead and less time spent in message sending processes.

## c) primeT – Number of Threads

In the third experiment, number of threads tested in different odd values in range 1 to 19.



The results indicate that as the number of threads increases, so does the execution time of the program. One factor contributing to this is the time lost in handling context switches. Additionally, another reason is the limited allocation of resources. Since the virtual machine used in these experiments only has 2 cores, increasing the number of threads cannot be effectively allocated to the CPU, resulting in resource contention. Consequently, the number of context switches rises, leading to greater overhead and longer execution times. If the system supported multi-threading more efficiently and had better resource allocation, increasing the number of threads could enhance parallelism. However, it's essential to consider that system capabilities may not be sufficient for effective parallelism, and adding more threads could result in increased overhead without a substantial improvement in execution.