|  |  |  |  |
| --- | --- | --- | --- |
| Solution type | Number of slaves | Execution time | Explain the result |
| Sequential solution | 1 | 8.781149 seconds | This is the slowest option, none of the work is done in parallel |
| Static task pool | 2 | 5.712547 seconds | This time two processes split the tasks between them, which gives a big improvement relative to running serially |
| Static task pool | 4 | 4.344777 seconds | Although we doubled the amount of slaves, performance didn’t rise accordingly. This is most likely due to the overhead of having more communication done between processes |
| Dynamic task pool | 2 | 4.395106 seconds | Dynamic approach with less processes gave a similar result to the static approach, this shows that our specific problem is well suited for the dynamic approach |
| Dynamic task pool | 4 | 2.316057 seconds | Doubling the amount of slaves almost halved our runtime, showing that our problem scales well for dynamic task pool |
| Dynamic task pool | 20 | 2.561379 seconds | We actually see the runtime getting worse when we maximize the amount of processes, this is again due to the overhead of the communication |