ATT A03: Parallelization

2018-04-06

Colin J. Mills, Kyle a kreutzer

Contents

[Results: 2](#_Toc510788342)

[Conclusions 2](#_Toc510788343)

# Results

|  |  |  |  |
| --- | --- | --- | --- |
| **# of iterations** | **Sequential Time / ms** | **Parallel Time / ms** | **Fastest** |
| 10 | 0 | 4 | Sequential |
| 100 | 17 | 23 | Sequential |
| 1000 | 63 | 103 | Sequential |
| 10000 | 28 | 13 | Parallel |
| 50000 | 477 | 131 | Parallel |
| 100000 | 1735 | 457 | Parallel |

# Conclusions

Overall the results indicate that the trend of performance follows the number of iterations. In smaller ranges of iterations, sequential was always faster, whereas when iterations approached larger numbers parallel became faster. This is due to the initialization that parallelization takes, for instance if the test case was 10 iterations, the parallelization needs to find a core to be able to use.

Whereas the sequential test case uses the CPU which is immediate. The overhead for parallelization is attributed to the Task Scheduler partitioning the workload. Although it only has to partition if the workload is imbalanced the process of balancing adds a timing overhead.