Thomas Mason

Multicore Processors - Lab 2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Table 1 (seconds)** | **10** | **100** | **1000** | **10000** |
| **1** | 0.131 | 0.13 | 0.554 | 48.633 |
| **2** | 0.124 | 0.131 | 0.564 | 48.763 |
| **10** | 0.147 | 0.173 | 0.581 | 49.393 |
| **20** | NA | 0.183 | 0.616 | 50.493 |
| **40** | NA | NA | 1.14 | 58.018 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Table 2** | **10** | **100** | **1000** | **10000** |
| **1** | 1 | 1 | 1 | 1 |
| **2** | 1.06 | 1.00 | 1.01 | 1.02 |
| **10** | 0.89 | 0.75 | 0.95 | 0.98 |
| **20** | NA | 0.71 | 0.90 | 0.96 |
| **40** | NA | NA | 0.49 | 0.84 |

1. During every trial of the 10, 20 and 40 core parallelization slowdown occurs (or speedup is a “break-even” time).
2. In my solution to this problem, I am using a Scatter() method which involves breaking the larger x array down into batches and sending each to its receiving process, and a termination condition that is set by using an Allreduce() method. These two communication calls between processes entail significant costs, as communication between processes is extremely expensive (much more so that between threads, for example). To put this more specifically, as the number of cores increases, the communication costs increase linearly while the computational work is divided by each additional parallel core, meaning that diminishing returns will quickly become apparent for relatively small problem sizes. If we had larger problem sizes we may have seen more significant speedup, as the slowdown at the 10,000 input level was substantially lower that at smaller levels. This is likely due to the overhead and communication costs of the large-process parallelization outstripping the gains, specifically with regards to the Scatter() and Broadcast() method, which entail communication between every process.

It is interesting to speculate that perhaps this algorithm is better suited for a sequential computation while another one would work better for a parallel computation.

1. The trials involving 2 parallel cores produce speedup, but only very small amounts.
2. For similar reasons stated in part (b), I am solving this method using a Scatter() and Broadcast() to send the required batches of data, which involves a fair amount of overhead in terms of communication time. However, in the case where there are only 2 processes, this is not as prohibitively expensive as it only has to be performed once (whereas for larger process jobs it will have to be done many times, contributing to significantly more overhead). This is due to the fact that communication between processes is extremely expensive.