Leslie Xie, Bojun Lin

Introduction:

This paper was written to explore the difference between the map and pmap functions and how concurrency speeds up execution times. We attempt this by carrying out an experiment, utilizing the same data on both functions and measuring the time taken for execution.

Implementation:

Pmap uses two threads and calls a helper method, limited_map, which takes two additional parameters that mark the start and end of each thread's boundaries. This cuts the data processed by each thread in half compared to the single-threaded map function. These threads are joined using the standard library, but are fully cleaned up in the Rower's virtual join function.

Description of the analysis performed:

Data generation was handled via python script.

Input data:

The data in datafile.txt is formatted as such.

Each row is formatted as:

int int

There are **6250000** rows in datafile.txt. This amount, with these data types comprising one row, resulted in a datafile.txt size >= 100mb, as requested per specs. This would help further set apart the execution difference between map and pmap. We did not have any need to test the capabilities of handling each primitive type, we only wanted to test computational speed. Using ints, we would have been able to do a simple Rower case (addition) and a lengthier Rower case (trigonometric functions).

The actual data operated on would have been identical to datafile.txt, but would not have been read into driver code.

The tests would have been performed on a machine with the following configuration:

Hardware

CPU - Intel(R) Core(TM) i7-8750H @ 2.20GHz - 6 cores, 12 processors
OS - Windows 10 Personal Edition
Laptop - Fully charged, on “best performance” mode

Additionally, a test would have been performed on the **Khoury Linux server**.

Hardware

Intel(R) Xeon(R) Gold 5118 CPU @ 2.30GHz
48 processors

Important:

The experiment was not carried out due to lengthy delays in generating the instance of the datafile to be operated on.

Comparison of the experimental results:

N/A

Threats to validity:

Different hardware would drastically change the execution times of the pmap/map functions. However, unless there was a large discrepancy or limited amount of threads available in the testing machine’s processors, then the difference between the pmap/map execution times would still be upheld.

Additionally, different software (operating systems) would also change the execution times. Similarly though, it would likely not change the difference between the two tested functions, instead speeding up both or slowing both down.

Conclusions:

While untested, based on the existing code for pmap as compared to map, pmap is identical to map except for the parallel threads that run concurrently. While there is only two threads utilized by pmap, in a large enough scale benchmark, the difference in execution speed shows.

In the case that the execution times did not clearly show a difference in speed would have resulted in additional threads being allocated to split the datafile between.