Complex Distributed Computing

Let’s first look at the Quicksort sequential runtimes that I calculated-

**For sorting a size 100 array:**



Sequential Takes 5ms

**For sorting a size 1000 array:**

****

Sequential Takes 62ms

**For sorting a size 10000 array:**

****

Sequential Takes 2020ms

Let’s look at the MPI Quicksort runtimes-

**For sorting a size 100 array:**

Text

Description automatically generated

MPl Takes 25ms

**For sorting a size 1000 array:**

Calendar

Description automatically generated

MPI takes 132ms

**For sorting a size 10000 array:**

Calendar

Description automatically generated with low confidence

MPI takes 4605ms

Let’s look at the MPI+OpenCL Quicksort runtimes-

**For sorting a size 100 array:**

A screenshot of a computer

Description automatically generated with medium confidence

OpenCL takes 430488ms

**For sorting a size 500 array:**

A picture containing calendar

Description automatically generated

OpenCL takes 433809ms

**For sorting a size 1000 array:**

Background pattern

Description automatically generated

OpenCL takes 492801

Conclusion

As we can see, from the above runtimes, sequential and MPI implementation have similar performances. However, with OPENCL+MPI implementation, the quicksort function is worst with smaller sized arrays. This is obvious because there is a lot of overhead when we are using OpenCL. The overhead in implementing the parallel sort overwhelms any performance gain that we could have obtained. With increasing array sizes, OpenCL+MPI will perform better, and operations will be much quicker.

The concept behind the decomposition of MPI version of the quicksort is similar to the sequential quicksort. The difference lies in parallelising the sorting function. After partition, higher end will be sorted in one node and lower partition will be sent to a different child node. Each then makes a recursive call and sorts itself. If the array size is too small to have distributed nodes, then it performs normal quicksort which saves a lot of computational power.

With addition of OpenCl, I was having difficulty with running the recursive quicksort function. Hence, I implemented an iterative quicksort using the concept of stacks. A similar concept here when compared to the MPI version. Instead, we are using different buffers to store each partition, reading from the buffer, sorting it and then returning it to an output array.