COL730 Parallel Programming

Assignment-3

Divyans Yadav 2020EE30593

October 29, 2023

Algorithm ParallelSort using MPI & OMP

1 Methodology

- Since, This required creating new Data type to be transferred between process. So, instead of creating new data type, I used `MPI::BYTE` to transfer bytes of C++ STL `pair<int,int>`.
- Then I linked the MPI program from Assignment 2 to OMP `ParallerSort`. Thus, creating parallelism at both node and core level.
- For writing output data in sequential manner from parallel program. I used `MPI::Send` and `MPI::Recv` to pass the messages.

2 Remarks

- Combining both MPI and OpenMP achieved lot more speedup compared to stand lone MPI and OpenMP sorting. It opens lots of way doing sorting in much fast manner of much large data.
- I couldn't able to test with more node and per node core due to budget restriction.
- If there comes a issue with respect to endianess, we can omit `__builting_bswap32` at Line 170 and 172 in `main.cpp`.

3 Results

3.1 Time vs CPU Count

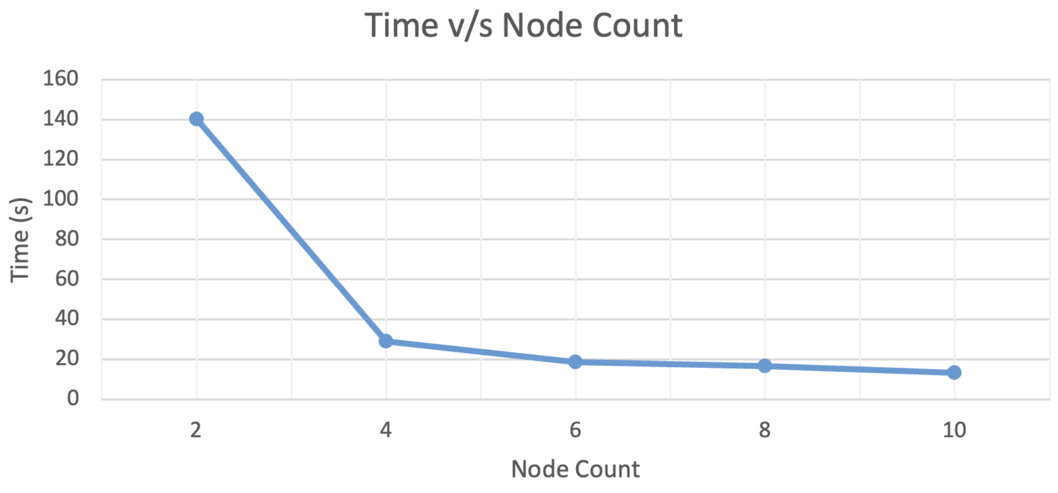

Figure 1: Array Size fixed at 2^{27} and cores per node at 2

3.2 Time vs Array Size

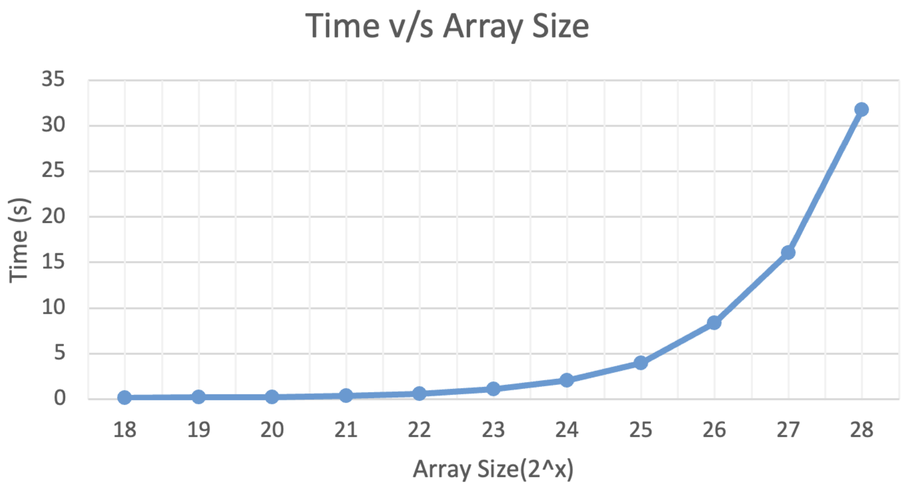

Figure 2: Node Count fixed at 32

The timing for array size below 2^{18} have less time taken by using conventional sequential sorting rather than our parallel. So, I did not included their data in the graph.