CSC 6220: Parallel Computing I: Programming Project Report Fall 2022

Student name: Riddhesh S. Markandeya (hm9650)

Execution time comparison

	Input Array size		
Sorting scheme	2^{16}	2^{20}	2^{24}
serial_quicksort	0.066041	1.315625	24.316088
original_odd_even	0.024181	0.299852	5.24099
modified_odd_even	0.024868	0.309786	5.349406

Table 1: Execution time(in sec) for different sorting schemes and different array size. Execution times are average of 3 runs.

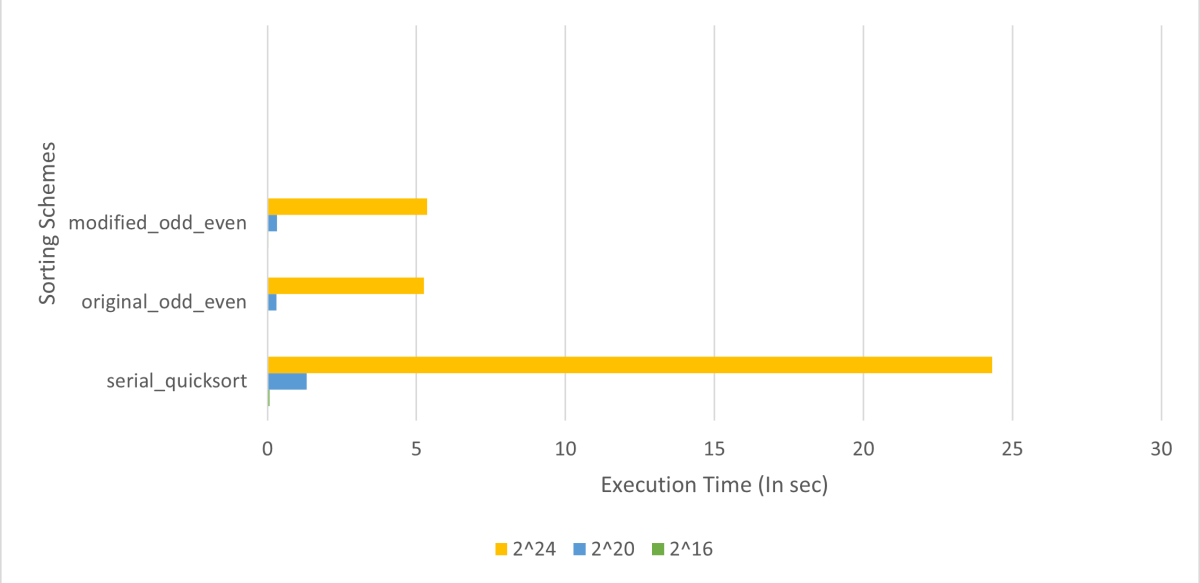

Figure 1: Plot of execution time for input array sizes and sorting schemes

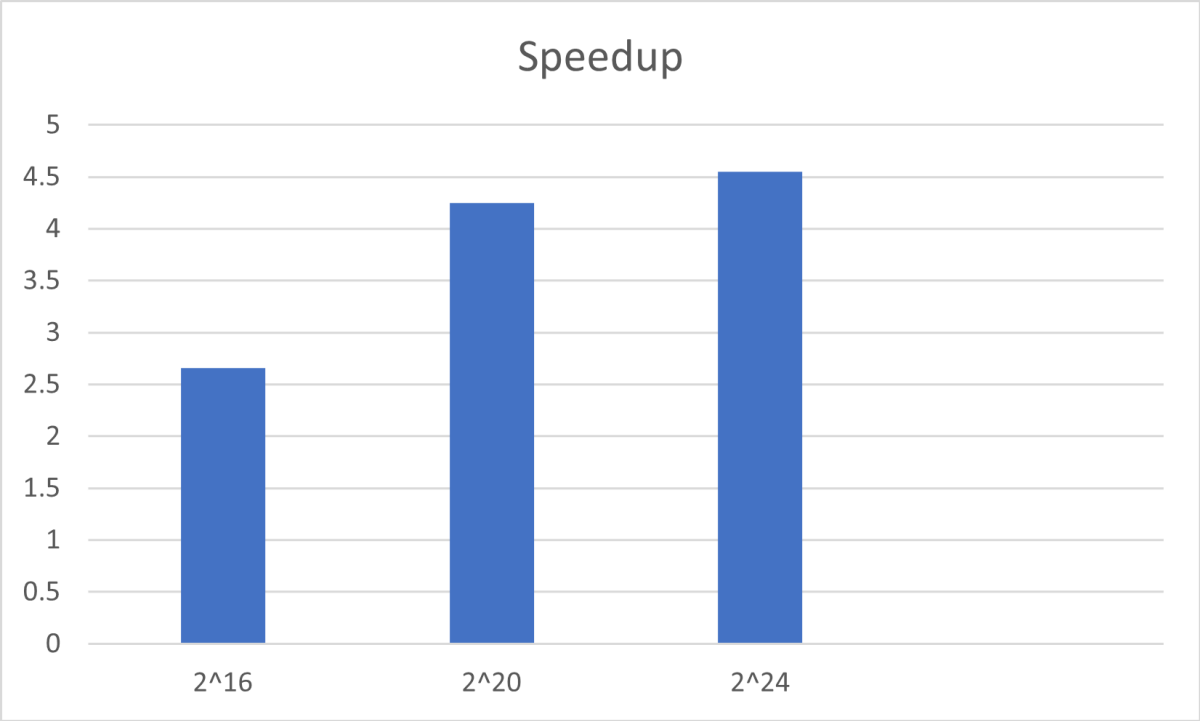

Figure 2: Plot of speedup of modified odd even sort with respect to serial quicksort.

The modified odd even sort has added step that after every iteration, it checks if the array was not modified(already fully sorted) and stops the loop to avoid extra unnecessary steps. In MPI implementation after each iteration, each individual process checks if no modification was done to local array. After that each process needs to get local array status of all other processes, so we use MPI_Allgather directive to accomplish that. This is a blocking operation and all process will need to synchronize here. Then all processes checks if all other one's arrays are unchanged and if that's true then all of them will break main loop.

From the Table 1 we see that the modified implementation is not producing better execution time. This is mainly due to MPI_Allgather adding another synchronization step and as we are generating random array input, chance of breaking the loop early is very less. So we still end up executing same amount of iterations with the added overhead.

Figure 2 calculates the speedup for modified odd even sort with respect to serial quicksort. We can see that for 2^{24} size we get speedup of around 4.55. Also from trend seems to be that for larger sizes speedup is not increasing much. The MPI implementation splits the main array in to n/p size and then sorts it with quicksort parallely in around $O((n/p)log(n/p))$ time, then there are $p-1$ iterations of odd even sort and each step performs $O(n/p)$ operations. So execution time should be of $O((n/p)log(n/p) + (p-1)(n/p))$ excluding communication overhead.