HYPER QUICK SORT

Vinay Vardhaman

CSE 702: Programming Massively Parallel Systems

Instructor: Prof. Dr. Russ Miller

University at Buffalo The State University of New York

Agenda

- Overview of Parallel Algorithm
- Modified Hyper Quick Sort Algorithm
- Working Example
- Results on Small Data
- Results on Big Data
- Speedups for different data
- Learnings
- References

Algorithm

- We randomly choose a pivot from one of the processers and broadcast it to every processor.
- Each processor divide its unsorted list into two lists: those smaller than (or equal) the pivot, those greater than the pivot.
- Each processor in the upper half of the processor list sends its “low list” to a partner processor in the lower half of the processor list and receives a “high list” in return.
- Now, the upper-half processors have only values greater than The pivot, and the lower-half processors have only values smaller than the pivot.
- Thereafter, the processors divide themselves into two groups and the algorithm continues recursively.
- After log(P) recursions, every processor has an unsorted list of values completely disjoint from the values held by the other processers.
- The largest value on processor i will be smaller than the smallest value held by processor i + 1
- Each processor can sort its list using sequential quicksort.

Modified Algorithm

- Each processor starts with a sequential quicksort on its local list
- Now we have a better chance to choose a pivot that is close to the true median.
  - The processor that is responsible for choosing the pivot can pick the median of its local list.
- The three next steps of hyper quick sort are the same as in parallel algorithm 1.
  - Broadcast
  - Division of “low list” and high list”
  - Swap between partner processors
- The next step is different in hyper quick sort.
  - On each processor, the remaining half of local list and the received half-list are merged into a sorted local list.
- Recursion within upper-half processors and lower-half processors.

Example

1,5,10,12,17 || 2,6,9,14,19 || 3,8,13,15,20 || 4,7,11,16,18

Example

1,5,10,12,17 || 2,6,9,14,19 || 3,8,13,15,20 || 4,7,11,16,18

1,5,10,12,17 || 2,6,9,14,19 || 3,8,13,15,20 || 4,7,11,16,18

Example

1,5,10,12,17 || 2,6,9,14,19 || 3,8,13,15,20 || 4,7,11,16,18

1,5,10,12,17 || 2,6,9,14,19 || 3,8,13,15,20 || 4,7,11,16,18

1,3,5,8,10 || 2,4,6,7,9 || 12,13,15,17,20 || 11,14,16,18,19

Example

1,5,10,12,17 || 2,6,9,14,19 || 3,8,13,15,20 || 4,7,11,16,18

1,5,10,12,17 || 2,6,9,14,19 || 3,8,13,15,20 || 4,7,11,16,18

1,3,5,8,10 || 2,4,6,7,9 || 12,13,15,17,20 || 11,14,16,18,19

1,3,5,8,10 || 2,4,6,7,9 ||| 12,13,15,17,20 || 11,14,16,18,19

Example

1,5,10,12,17 || 2,6,9,14,19 || 3,8,13,15,20 || 4,7,11,16,18

1,5,10,12,17 || 2,6,9,14,19 || 3,8,13,15,20 || 4,7,11,16,18

1,3,5,8,10 || 2,4,6,7,9 || 12,13,15,17,20 || 11,14,16,18,19

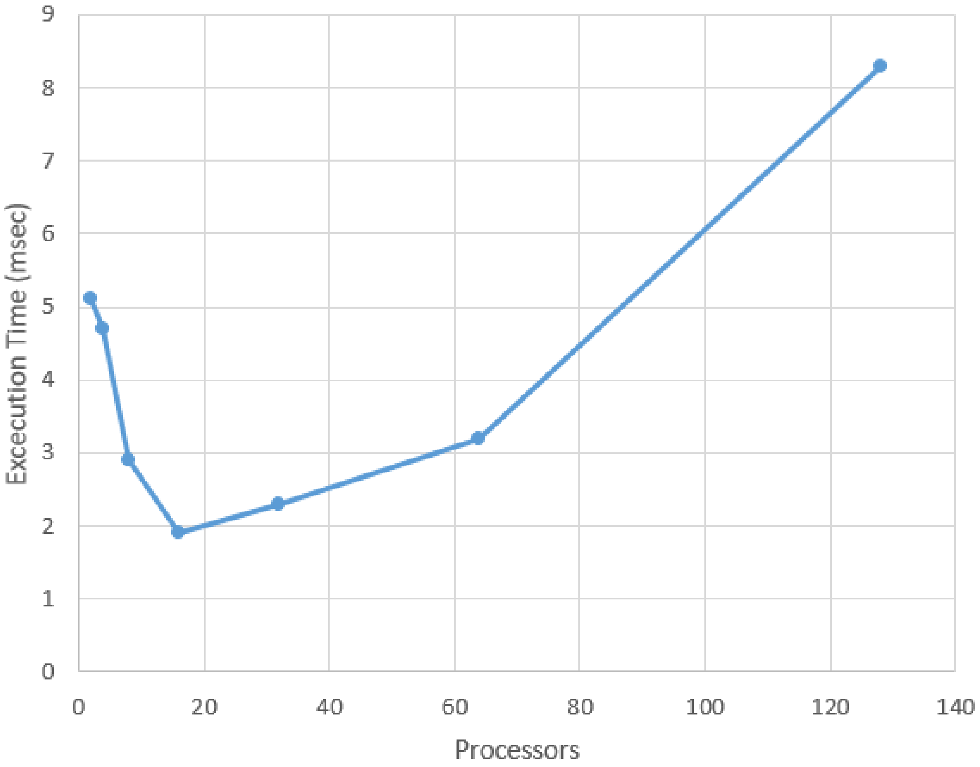
1,3,5,8,10 || 2,4,6,7,9 ||| 12,13,15,17,20 || 11,14,16,18,19

1,2,3,4,5 || 6,7,8,9,10 ||| 11,12,13,14,15 || 16,17,18,19,20

OBSERVATIONS

Small Data (100k)

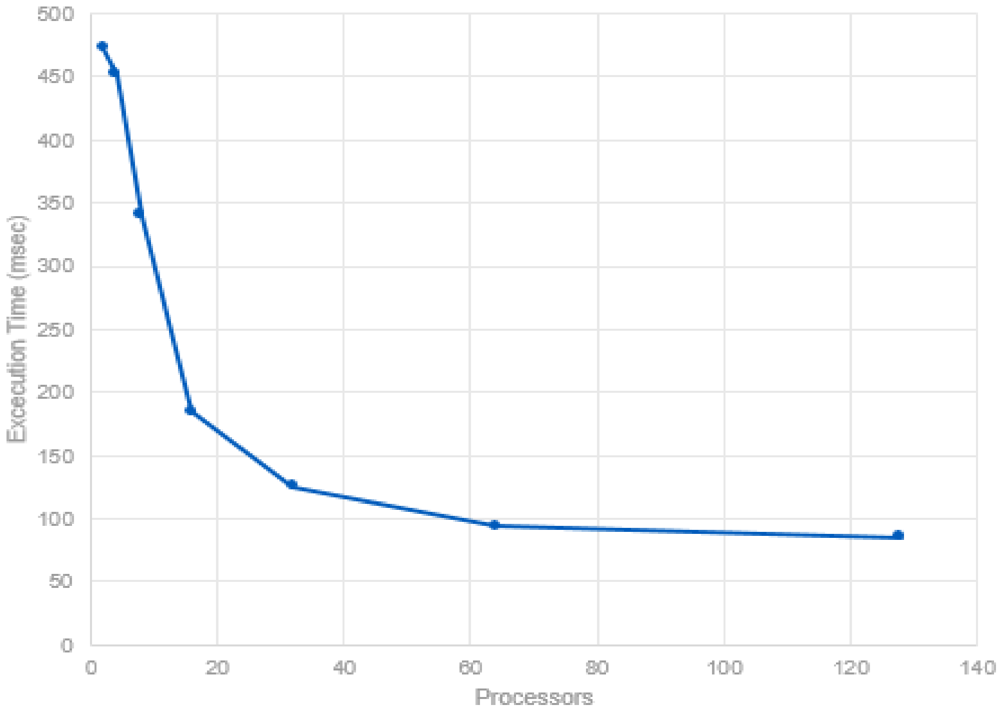
Number of Processors	Execution Time (msec)
2	5.1
4	4.7
8	2.9
16	1.9
32	2.3
64	3.2
128	8.3



OBSERVATIONS

Large Data (10 Million)

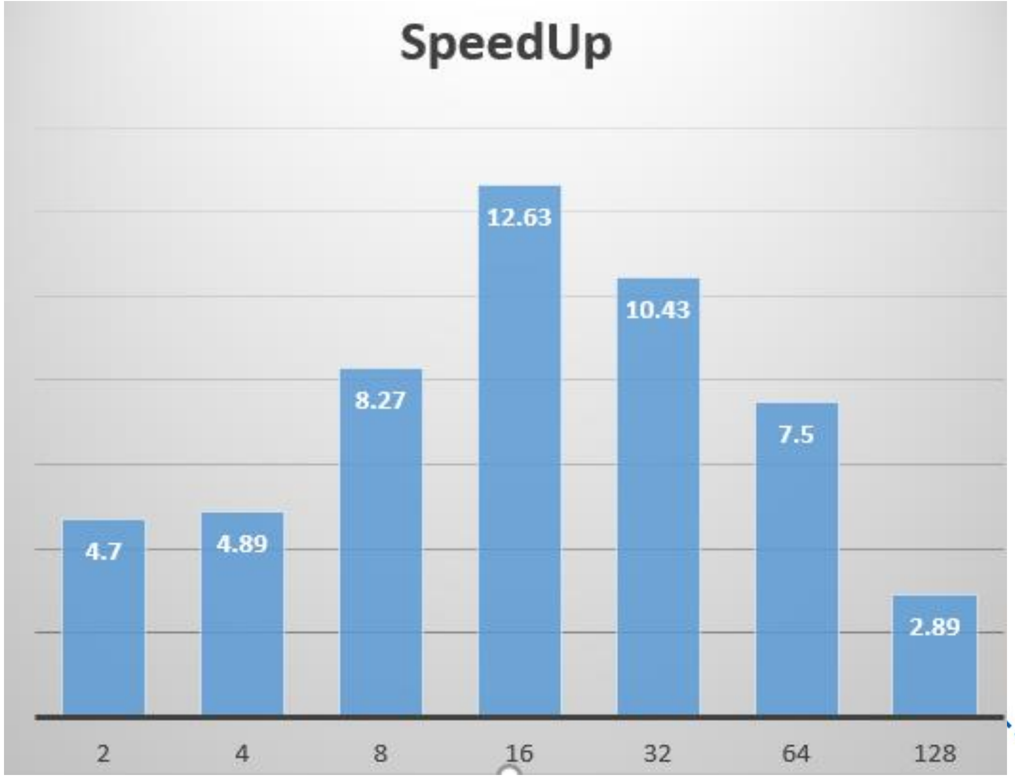
Number of Processors	Execution Time (msec)
2	473
4	452
8	340
16	185
32	125
64	94
128	85



OBSERVATIONS SPEED UP

Small Data (100k)

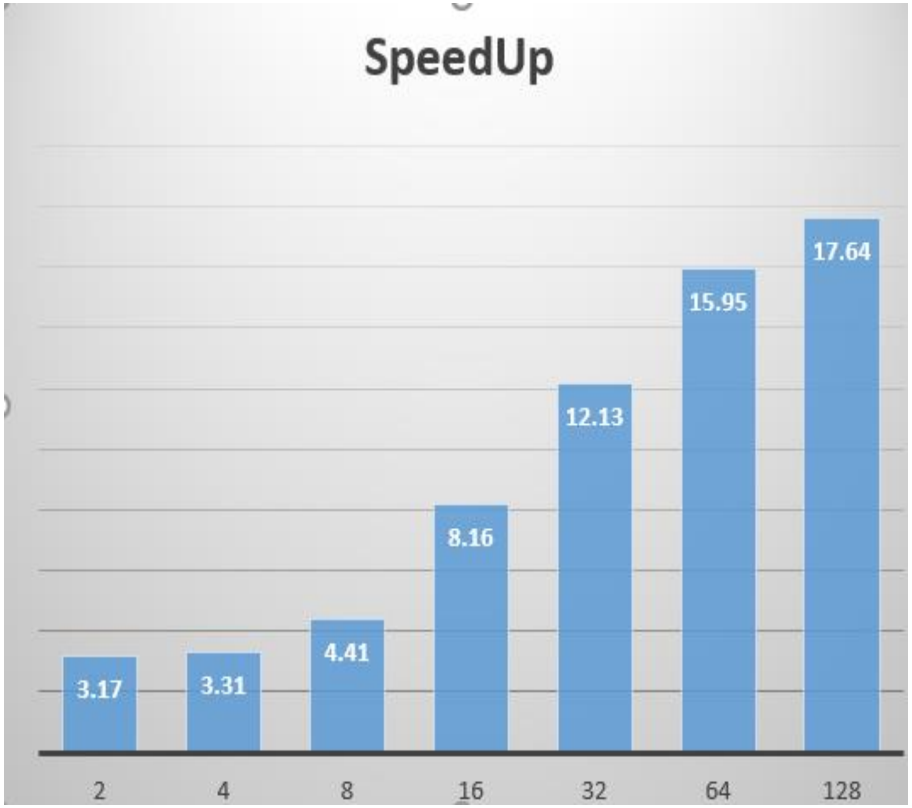
Number of Processors	Speedup
2	4.70
4	4.89
8	8.27
16	12.63
32	10.43
64	7.5
128	2.89



OBSERVATIONS

Large Data (10 Million)

Number of Processors	SpeedUp
2	3.17
4	3.31
8	4.41
16	8.16
32	12.13
64	15.95
128	17.64



Learnings

- Implementation of parallel algorithm using Message Passing Interface
- Observed the difference in runtimes for different number of processors. As the no of processors increase runtime decrease up to certain level and then decrease.
- Its always better to limit the number of processors to get maximum speedup

References

http://www.cas.mcmaster.ca/~nedialk/COURSES/4f03/Lectures/quicksort.pdf

http://parallelcomp.uw.hu/ch09lev1sec4.html

https://www.uio.no/studier/emner/matnat/ifi/INF3380/v10/undervisningsmateriale/inf3380-week12.pdf

THANK YOU