Calculate Lower Bound For All-Reduce Operation

Theory :

Latency: The lower bound on latency is derived by the simple observation that for all collective communications at least one node has data that must somehow arrive at all other nodes. At each step, we can at most double the number of nodes that get the data. So

$$LowerBound(Latency) = alpha * ceil(\log_2(p)) ;$$

Computation: Only the reduction communications require computation. The computation involved would require (p − 1)n operations if executed on a single node or time (p − 1)n*gamma. Distributing this computation perfectly among the nodes reduces the time to ((p − 1)/ p)n*gamma under ideal circumstances. Hence the lower bound.

$$LowerBound(Computation) = gamma * n * (p-1)/p$$

BandWidth: If the lower bound on computation is to be achieved, one can argue that ((p − 1)/ p)n items must leave each node, and ((p − 1)/ p)n items must be received by each node after the computation is completed for a total cost of at least Beta * 2((p − 1)/ p)n.

$$LowerBound(Bandwidth) = Beta * 2((p - 1)/ p)n$$

Therefore

LowerBound(AllReduce) :

$$alpha * ceil(\log_2(p)) + gamma * n * (p-1)/p + Beta * 2((p - 1)/ p)n$$

Here,

- -> p - number of process
- -> n - number of items

Calculation :

To calculate the lower bound for allreduce operation we have used the above logic, but instead using any theoretical values we have calculated the values experimentally as described below :

- To calculate the bandwidth part the easiest approach was to calculate the time taken to send 2*n*((p − 1) / p) items from one process to another.
- But to add the latency term, instead of doing it in the above manner, we divides the task into logP processes, i.e.

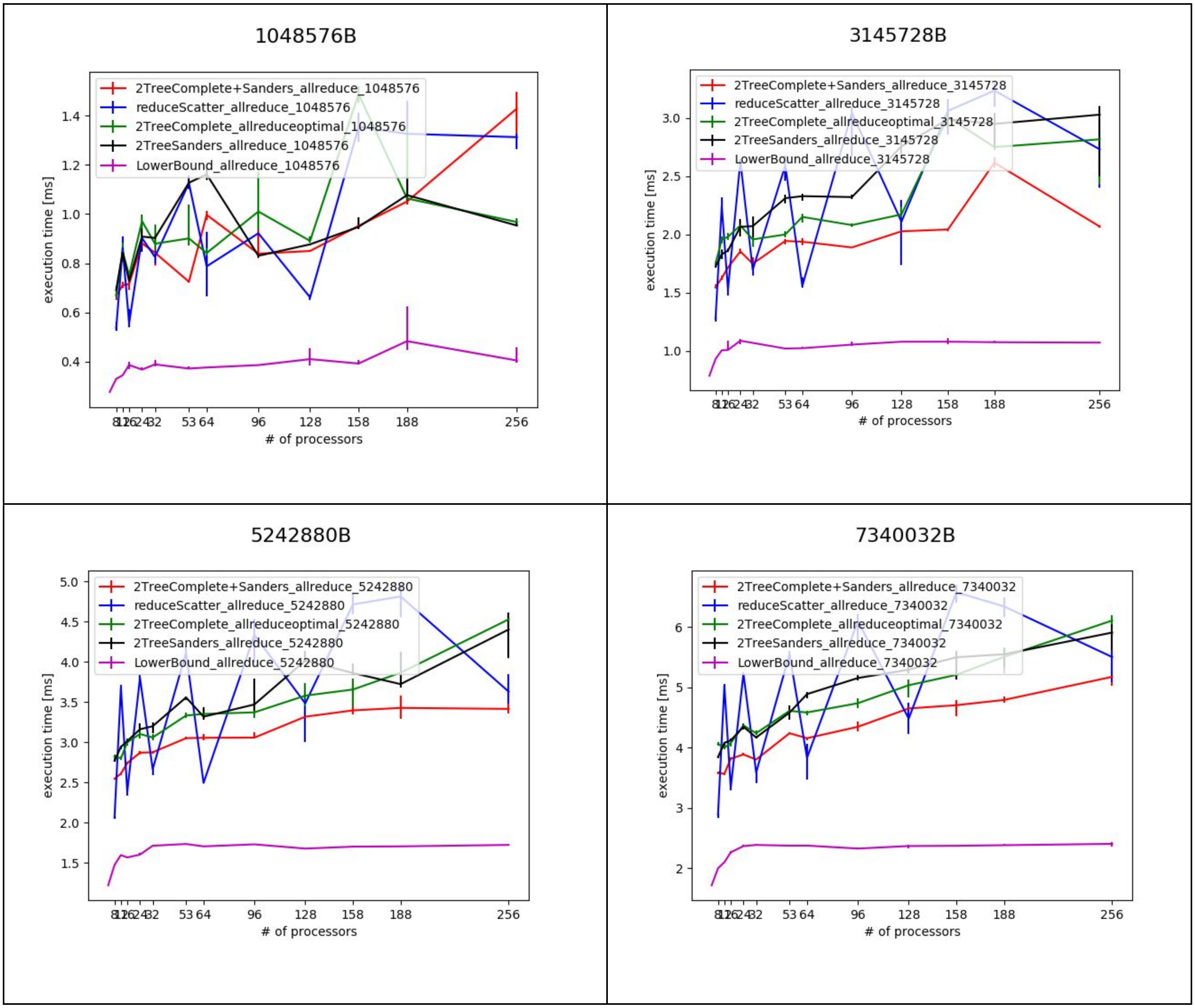
  For each p in 0..logP

  Each process p(except p=0) will receive (2*n*((p − 1)/ p) / logP) items from process p-1 and Each process p (except p=logP) will send (2*n*((p − 1)/ p) / logP) items to process p+1 after receiving from p-1.

  Hence, producing the latency + bandwidth lowerBound.
- Then the computation time for n * (p-1)/p items is added to add the computation lower bound (also calculated experimentally).

Results :

The below graphs shows the results for lowerbound along with the runtime results for different algorithms for different messages sizes.


1048576B
2TreeComplete+Sanders_allreduce_1048576
reduceScatter_allreduce_1048576
2TreeComplete_allreduceoptimal_1048576
2TreeSanders_allreduce_1048576
LowerBound_allreduce_1048576
execution time [ms]
# of processors
3145728B
2TreeComplete+Sanders_allreduce_3145728
reduceScatter_allreduce_3145728
2TreeComplete_allreduceoptimal_3145728
2TreeSanders_allreduce_3145728
LowerBound_allreduce_3145728
execution time [ms]
# of processors
5242880B
2TreeComplete+Sanders_allreduce_5242880
reduceScatter_allreduce_5242880
2TreeComplete_allreduceoptimal_5242880
2TreeSanders_allreduce_5242880
LowerBound_allreduce_5242880
execution time [ms]
# of processors
7340032B
2TreeComplete+Sanders_allreduce_7340032
reduceScatter_allreduce_7340032
2TreeComplete_allreduceoptimal_7340032
2TreeSanders_allreduce_7340032
LowerBound_allreduce_7340032
execution time [ms]
# of processors