BITONIC SORT USING MPI AND OPENMP

Presented for CSE702 Fall 2021

Instructor: Dr. Russ Miller

Presenter: Abheejeet Singh

University at Buffalo The State University of New York

Agenda

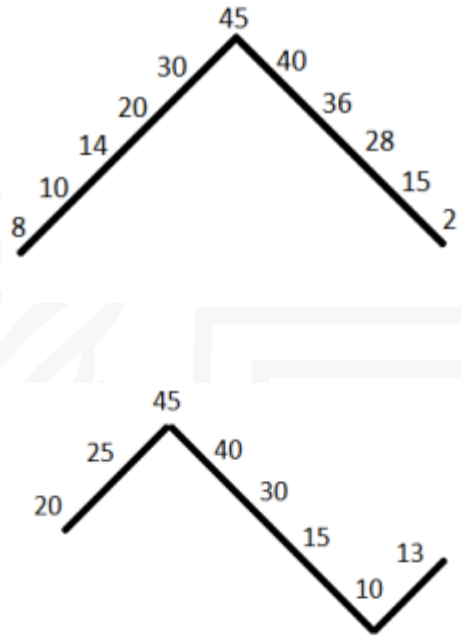
- Bitonic Sequence
- Bitonic Sort
- Bitonic Sort (Alternate)
- QuickSort Using OpenMP
- Results
- References

Bitonic Sequence

1. A sequence of numbers which monotonically increases , reaches a maximum and then monotonically decreases is called a Bitonic Sequence.
2. A sequence is also Bitonic if the above condition can be met by shifting the numbers cyclically(left or right).

Examples:

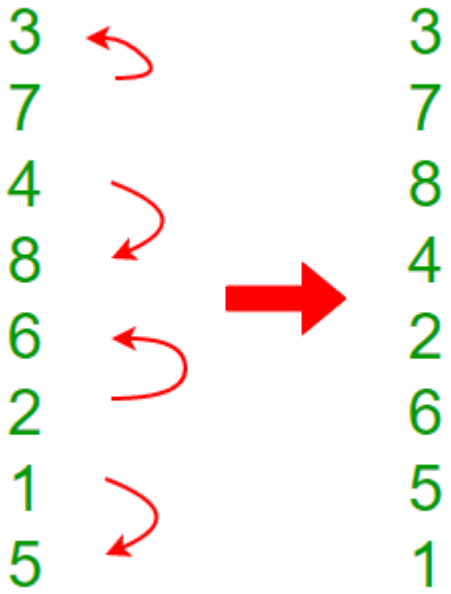
20 30 40 35 19 20 30 40 50 60 60 50 30 20 10



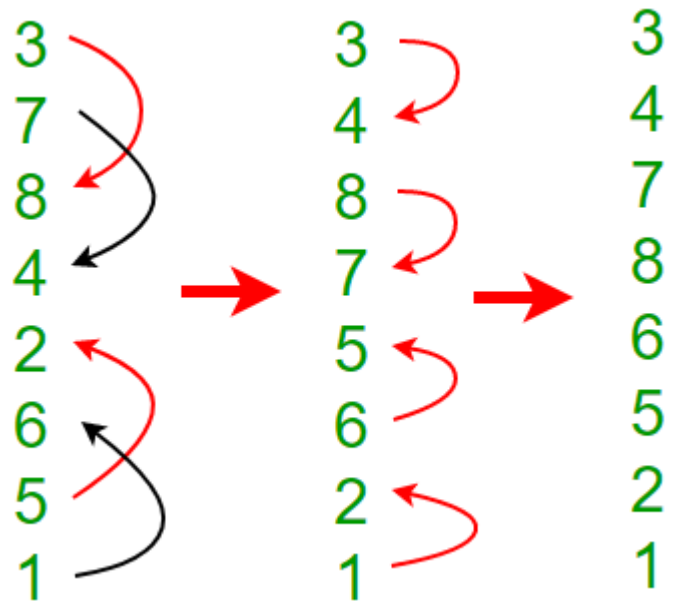
Bitonic Sort

1. To sort an unordered sequence, sequences are merged into larger bitonic sequences, starting with pair of adjacent numbers.
2. By compare and exchange operation, pairs of adjacent numbers are formed into increasing and decreasing sequences.
3. Pairs of which form larger bitonic sequence twice the size of the original sequence.
4. By repeating this process, bitonic sequences of larger and larger lengths are obtained.

Bitonic Sort

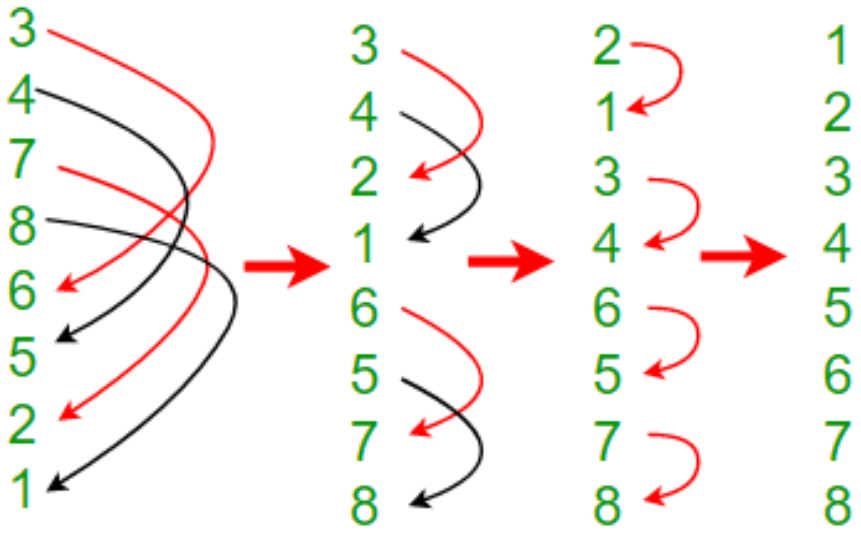


Phase 1



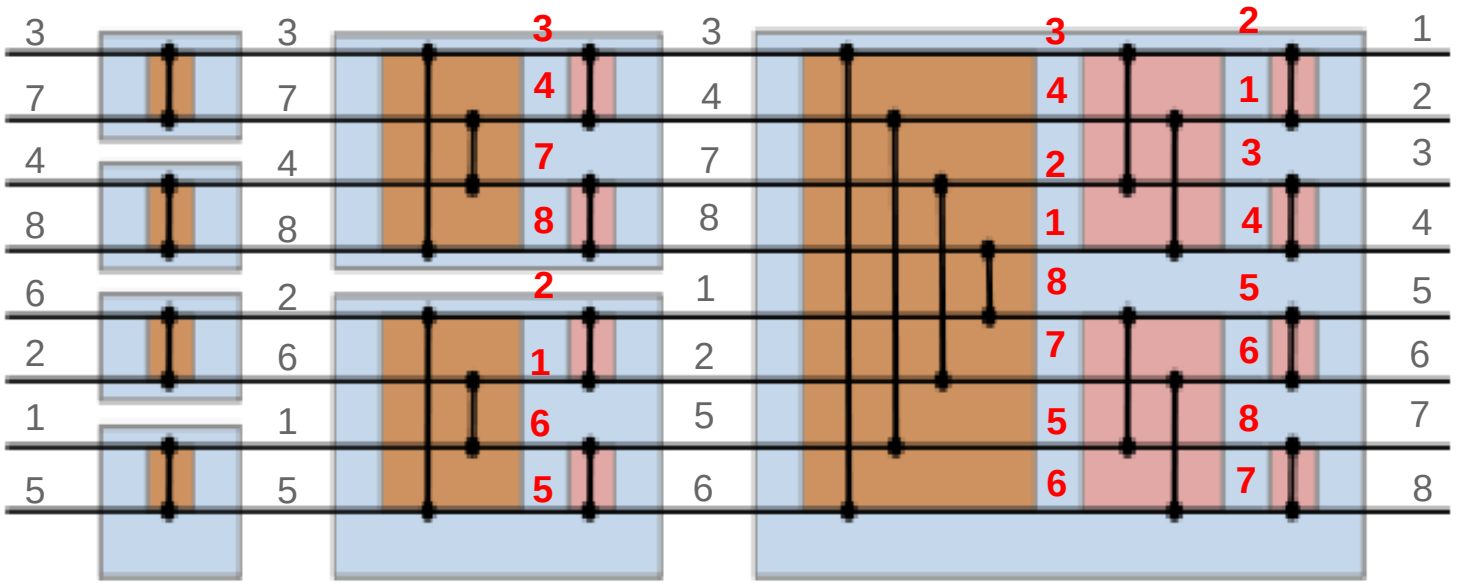
Phase 2

Bitonic Sort

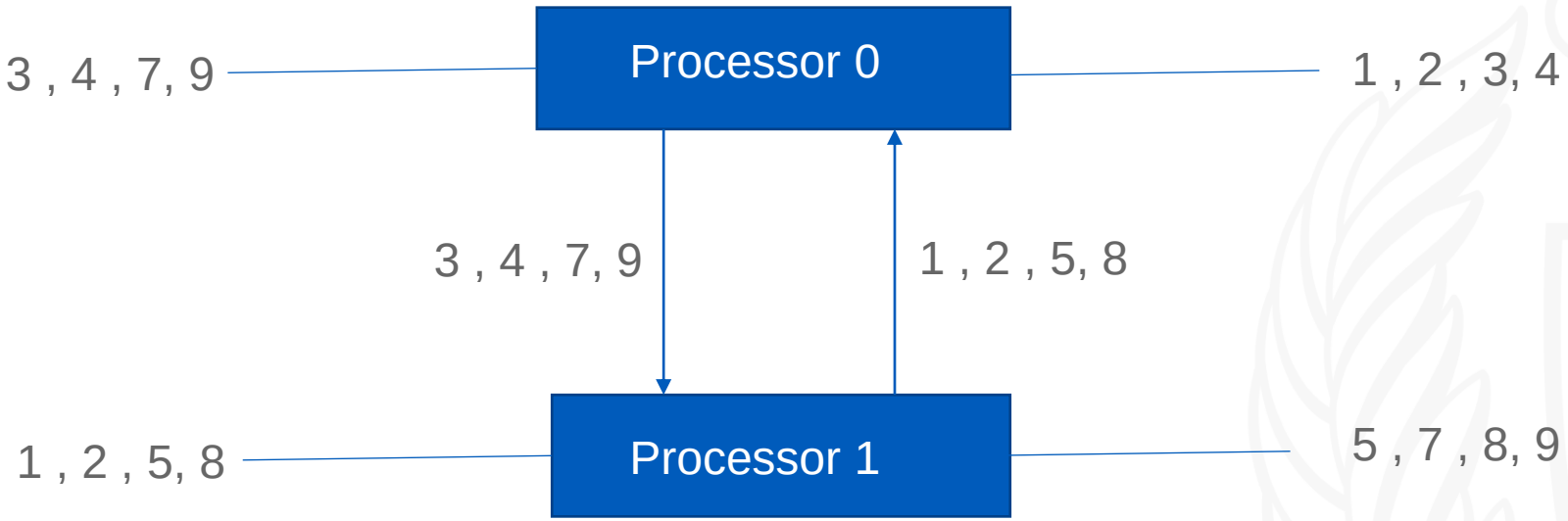


Phase 3

Bitonic Sort (Alternate)



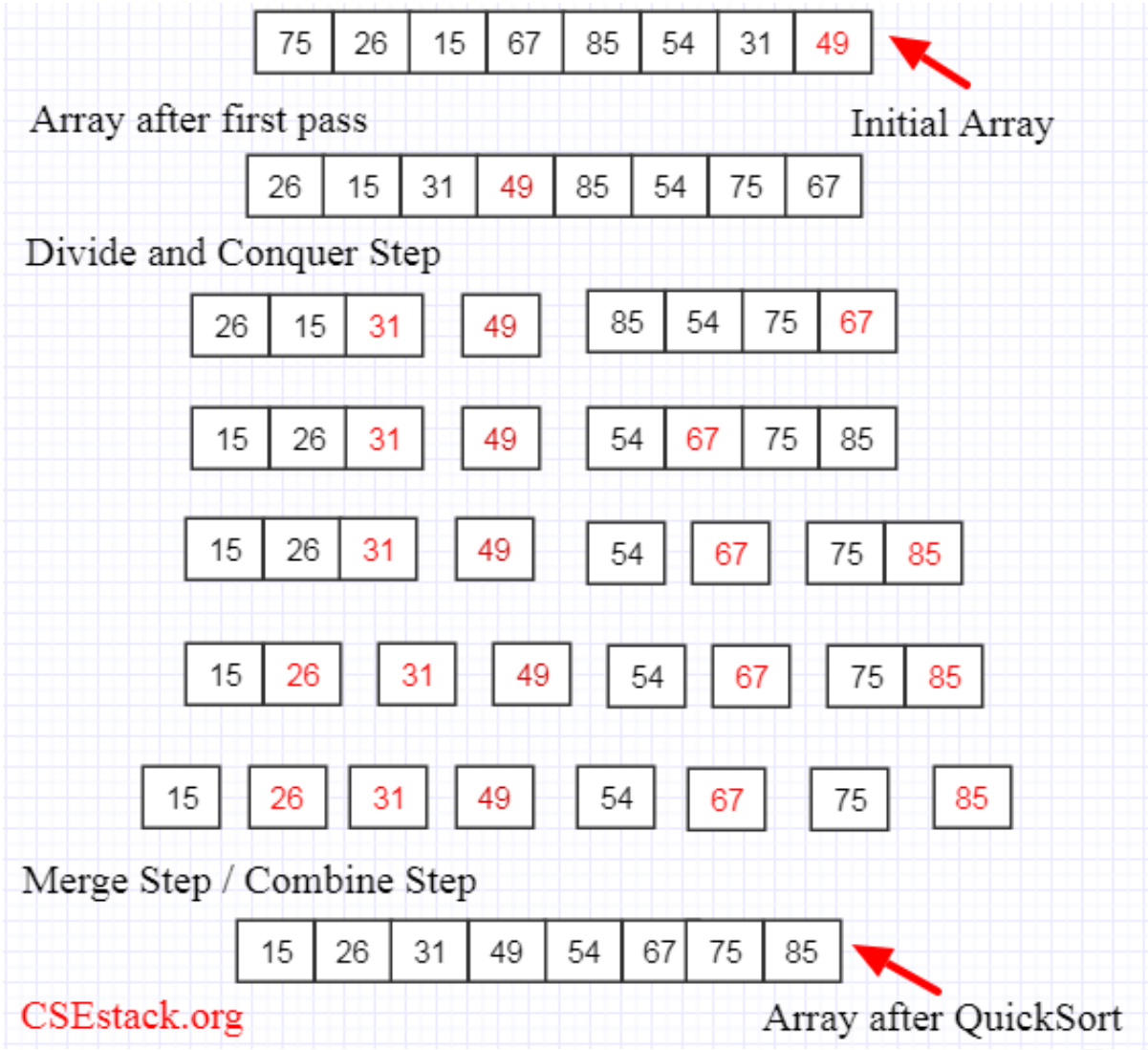
Bitonic Sort (Alternate)



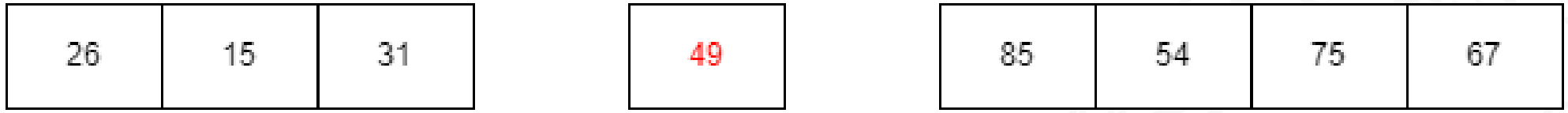
QuickSort Using OpenMP

QuickSort

1. Select Pivot element from the array. Then partition the array around this pivot element.
2. Multiple strategies to select the pivot element it can be the first element , the last element or some random element from the array. We can also take pivot as a median of subset of elements from the array.
3. Partition the element into two subarrays such that the subarray to the left of pivot element contains numbers smaller than the pivot and the subarray to the right of the pivot element contains numbers larger than the pivot.
4. Repeat this process for the two subarrays recursively.
5. The final result is the concatenation of the left subarray , the pivot element and the right subarray.


75 26 15 67 85 54 31 49
Initial Array
Array after first pass
26 15 31 49 85 54 75 67
Divide and Conquer Step
26 15 31 49 85 54 75 67
15 26 31 49 54 67 75 85
15 26 31 49 54 67 75 85
15 26 31 49 54 67 75 85
15 26 31 49 54 67 75 85
Merge Step / Combine Step
15 26 31 49 54 67 75 85
CSEstack.org
Array after QuickSort

OpenMP
Pragma Task



Repeat for Left
Subarray

Thread
T1

OpenMP
Pragma Task

Repeat for Right
Subarray

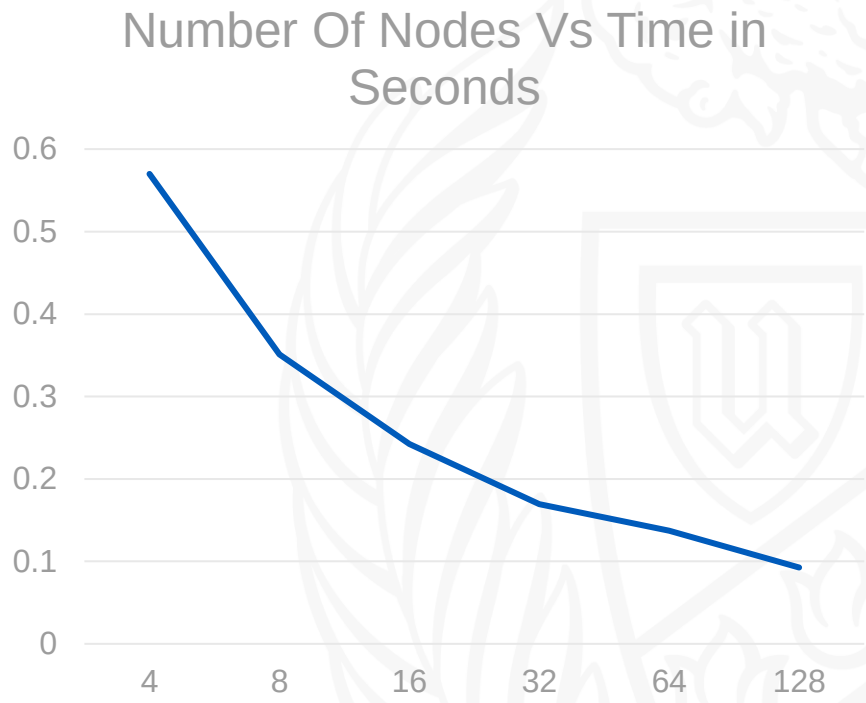
Thread
T2

Results

MPI

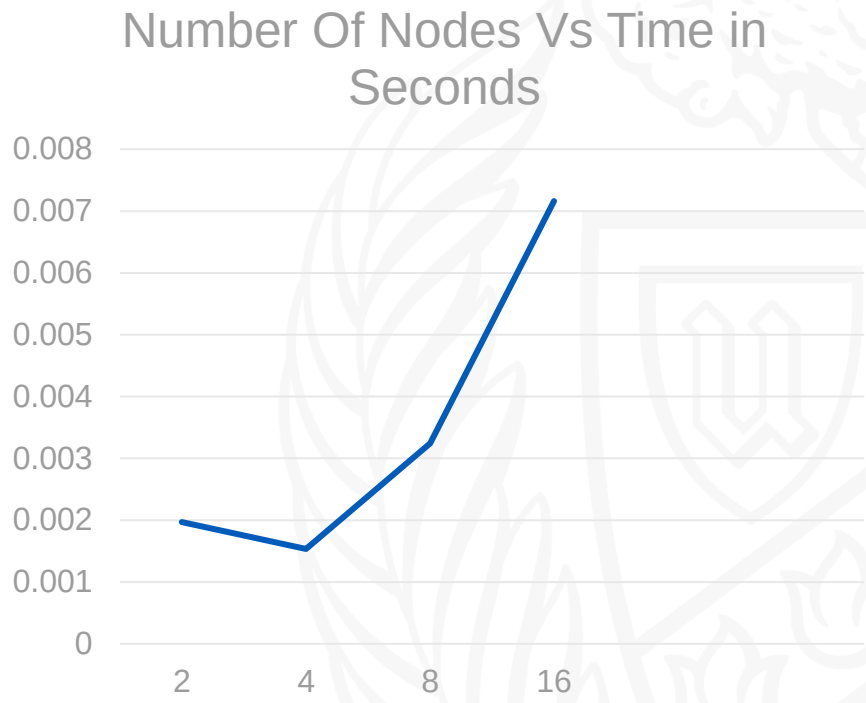
8 Million Numbers

No of Nodes	Time in seconds
4	0.569966
8	0.351311
16	0.242491
32	0.169394
64	0.137101
128	0.092568



10000 Numbers

No of Nodes	Time in seconds
2	0.001970
4	0.001534
8	0.003242
16	0.007160

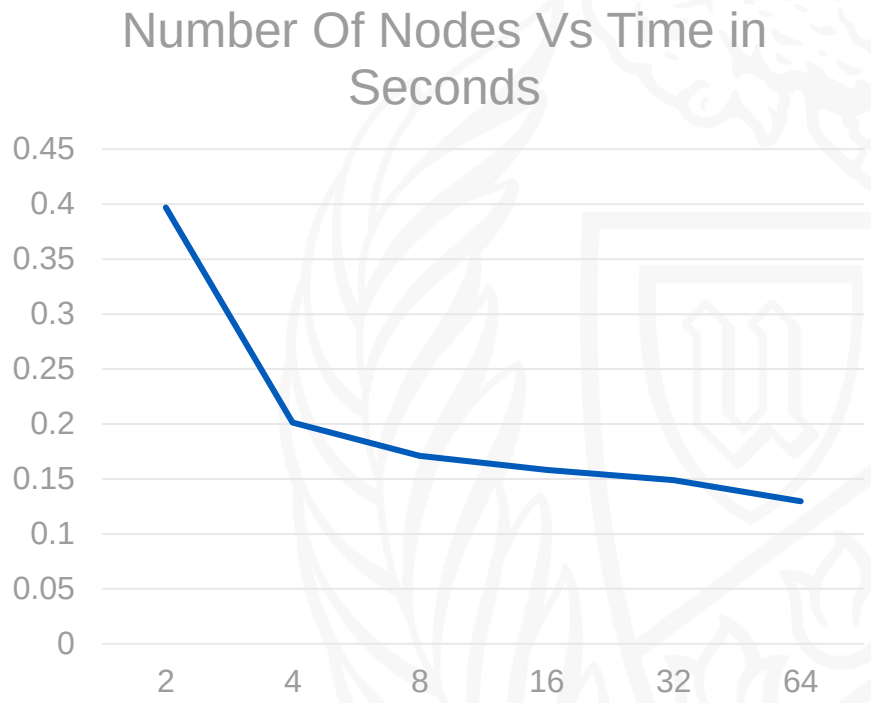


MPI + OpenMP

8 Million Numbers

No of Threads Per Node : 8

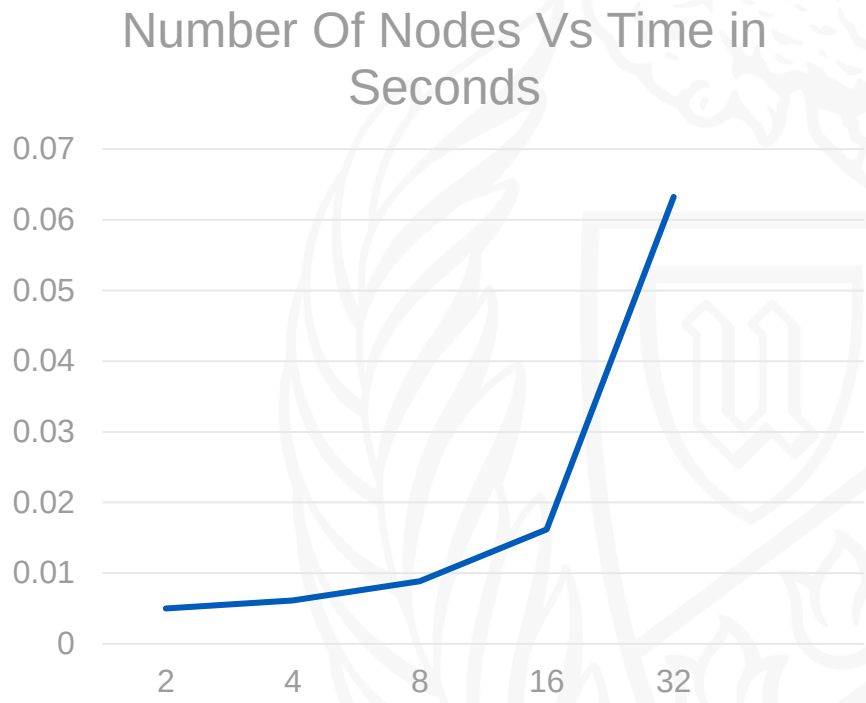
No of Nodes	Time in seconds
2	0.397091
4	0.201192
8	0.171047
16	0.158210
32	0.149009
64	0.129688



10000 Numbers

No of Threads Per Node : 8

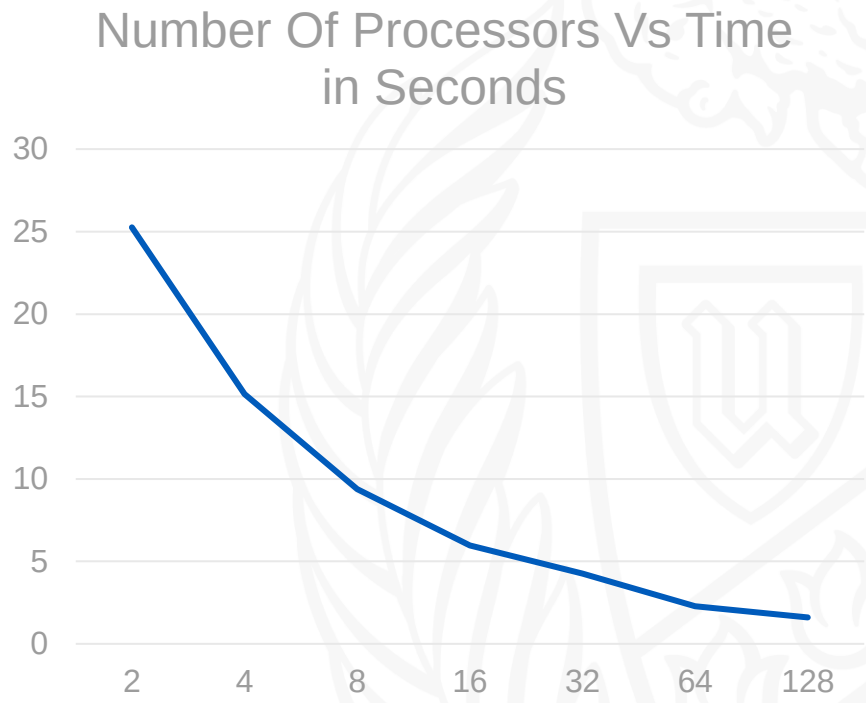
No of Nodes	Time in seconds
2	0.004991
4	0.006159
8	0.008857
16	0.016174
32	0.063268



500 Million Numbers

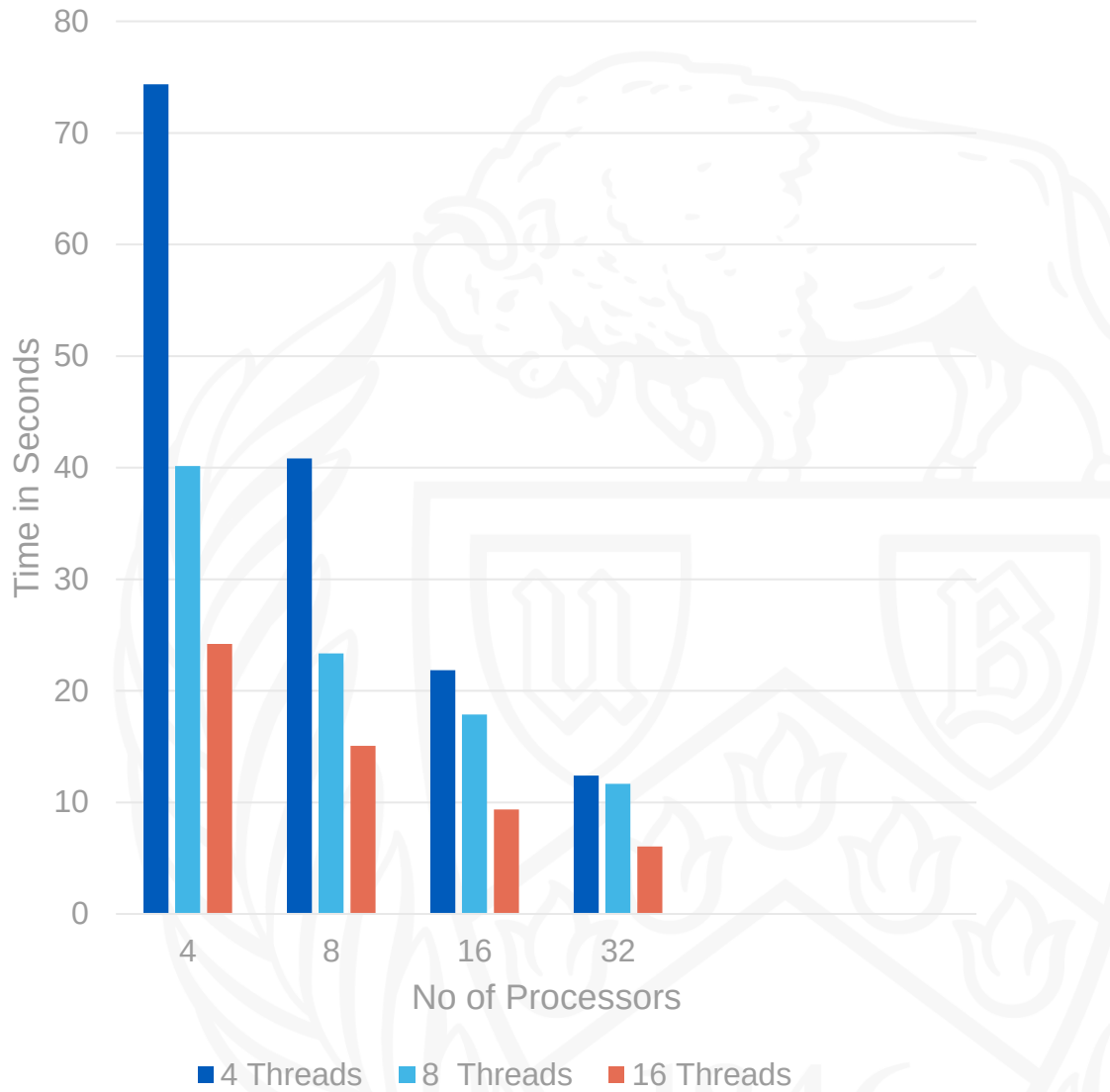
No of Threads Per Node : 16

No of Processors	Time in seconds
2	25.254875
4	15.149277
8	9.386049
16	5.969205
32	4.260776
64	2.285556
128	1.598354



800 Million Numbers

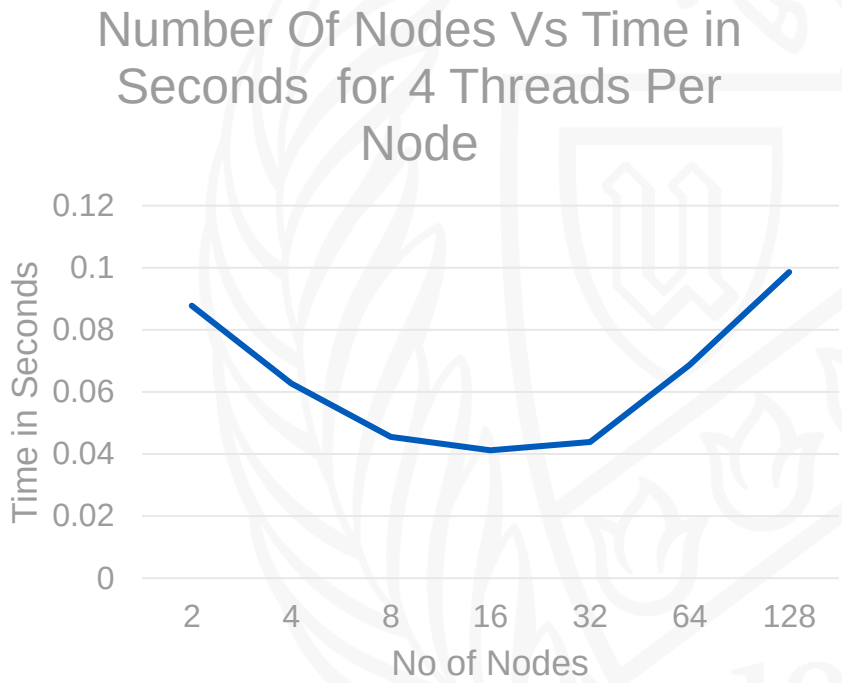
No of Processors	4 Threads	8 Threads	16 Threads
4	74.361584	40.142211	24.203003
8	40.830573	23.347264	15.055175
16	21.836126	17.869357	9.378940
32	12.402513	11.665426	6.040916



1 Million Numbers

No of Threads Per Node : 4

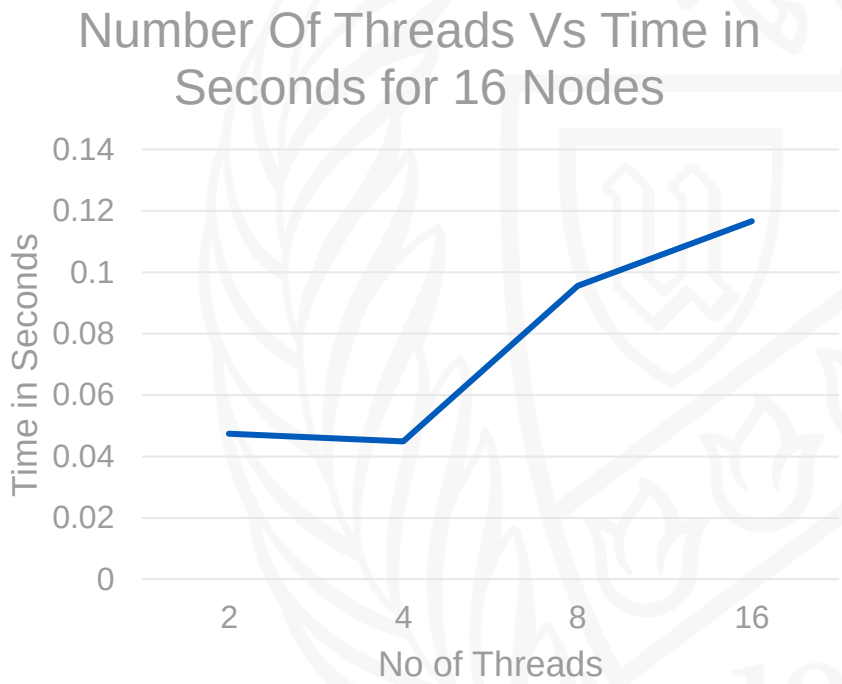
No of Nodes	Time in seconds
2	0.087750
4	0.062701
8	0.045576
16	0.041226
32	0.043855
64	0.068665
128	0.098548



1 Million Numbers

No of Nodes : 16

No of Threads	Time in seconds
2	0.047395
4	0.044915
8	0.095601
16	0.116581



References

- https://cse.buffalo.edu/faculty/miller/teaching.shtml
- Russ Miller, “Algorithms Sequential & Parallel: A Unified Approach”
- https://en.wikipedia.org/wiki/Bitonic_sorter
- https://www.geeksforgeeks.org/bitonic-sort/
- https://people.cs.rutgers.edu/~venugopa/parallel_summer2012/bitonic_overview.html
- Parallel Programming: Techniques and Applications Using Networked Workstations and Parallel Computers by Barry Wilkinson and Michael Allen
- https://pages.tacc.utexas.edu/~eijkhout/pcse/html/index.html

Thank You !