COMPUTING OVERLAPPING LINE SEGMENTS

- A parallel approach by Vinoth Selvaraju

MOTIVATION

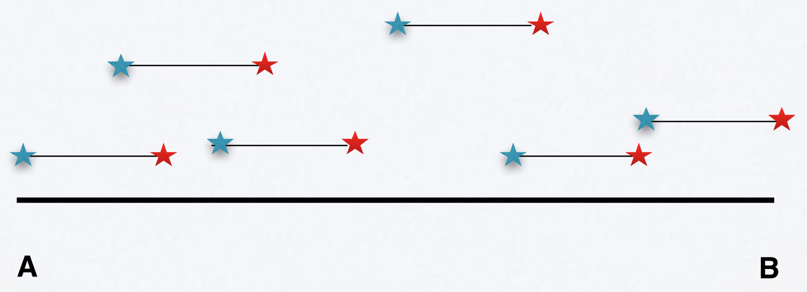
- Speed is fun, isn’t it?!
- There are 2 main reasons to parallelize the code
  - solve a bigger problem
  - reach solution faster

I tried to parallelize a simple problem in computation geometry to achieve a faster solution

PROBLEM STATEMENT

For a given set of "n" ordered line segments along the same line (x-axis),

Find whether or not the x-axis is completely covered by the set of n line segments between two given x-coordinates.



SEQUENTIAL SOLUTION

- Input consists of an array with 2n entries and end points of the X-axis coverage (A & B)
- Scan through the array, for each of the points between A & B, create an operand field that is set to 1 if that represents the left end point and is set to -1 if that represents the right end point
- Once again scan through the array, for all the 2n points, compute the parallel prefix sum
- Final scan to find the break point
- Running time - Theta(n)

PARALLEL APPROACH

On CCR Cluster

- More than 1 record is distributed across number of nodes taken into processing
- Broadcast the values A & B to first and last node respectively
- OR semigroup operation to compute the desired comparison for A and B and a broadcast of the halting condition
- Each node performs a local parallel prefix operation at the same time
- Globally, parallel prefix is computed over total number of nodes and final prefix values stored one per node
- Broadcast the results within each of the nodes, the final prefix value determined by the previous processor
- A semigroup operation can be performed to find the first break point i.e. value of 0 corresponds to break point
- Running time - Depends upon the number of data distributed per processor and total number of processor taken for computing. Running time will be dominated by the local parallel prefix compute or the global parallel prefix compute whichever is larger

SAMPLE SIZE

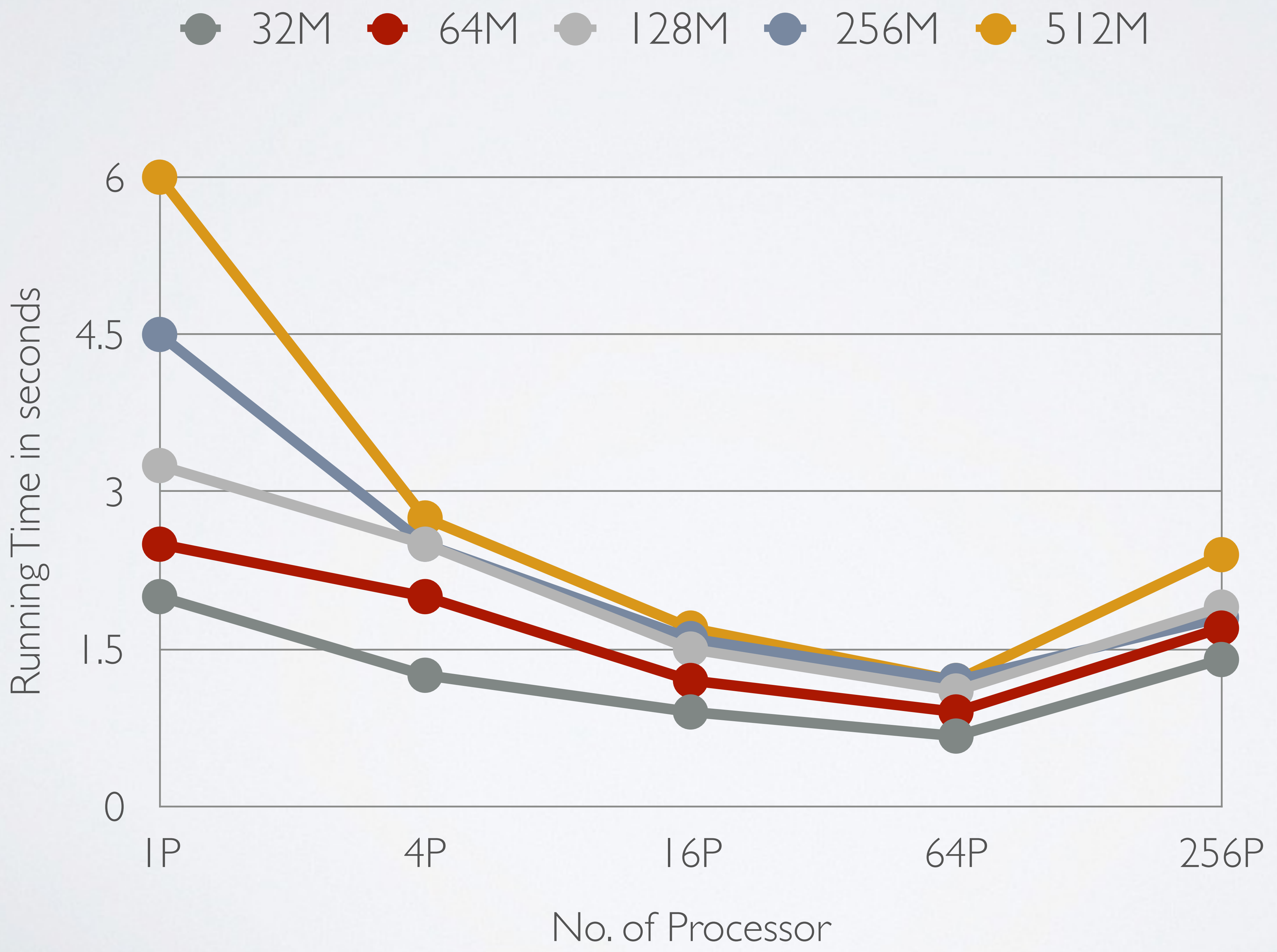
- Input size has been varied from 1000 to 512 million line segment points and nodes has been varied from 1 to 256

TEST RESULT ANALYSIS

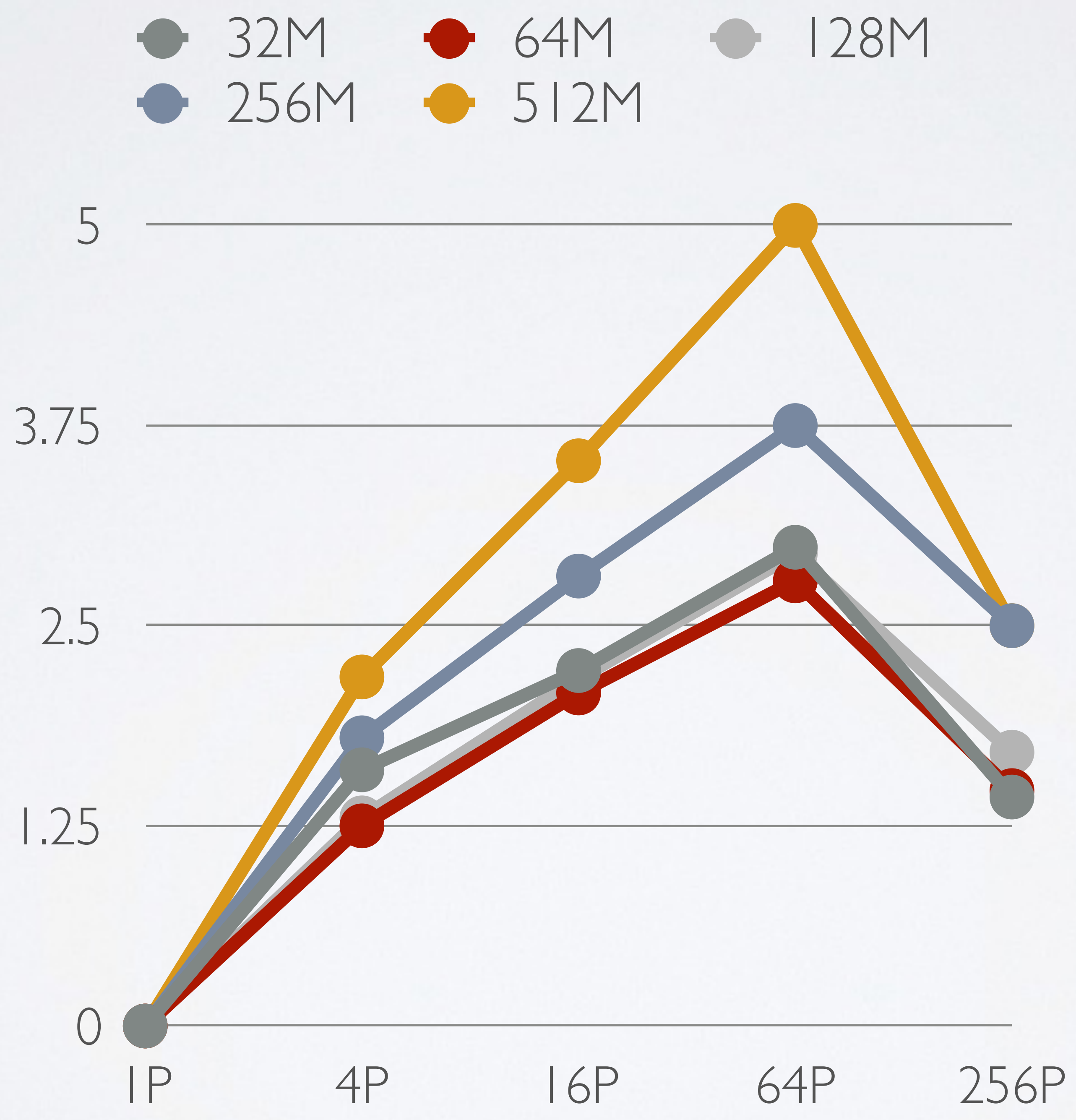
Approach:

- The whole idea of this project is to find the running time of the algorithm in Parallel architecture and compare against the Sequential version.
- Running time has been studied by varying no. of nodes, input size, number of data per nodes etc.
- Each test was carried out 10 times and Tmin, Tmax & Tavg has been computed. Tavg has been used in analysis of the running time.
- The number of processor varied from 1 to 256 and the number of data has been varied from 1000 to 512 Million

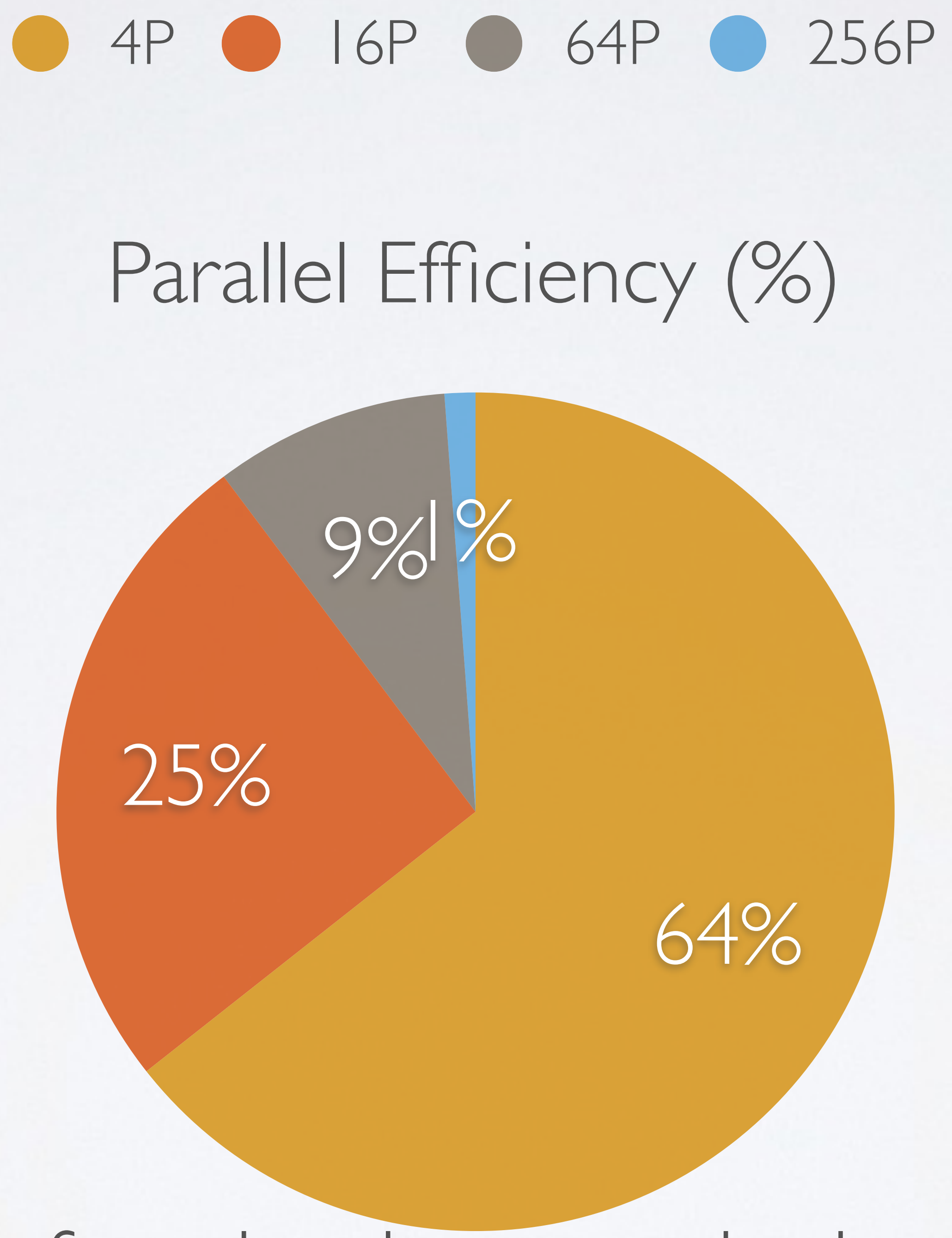
RUNNING TIME VS NO. OF PROCESSORS



CLUSTER - SPEED UP

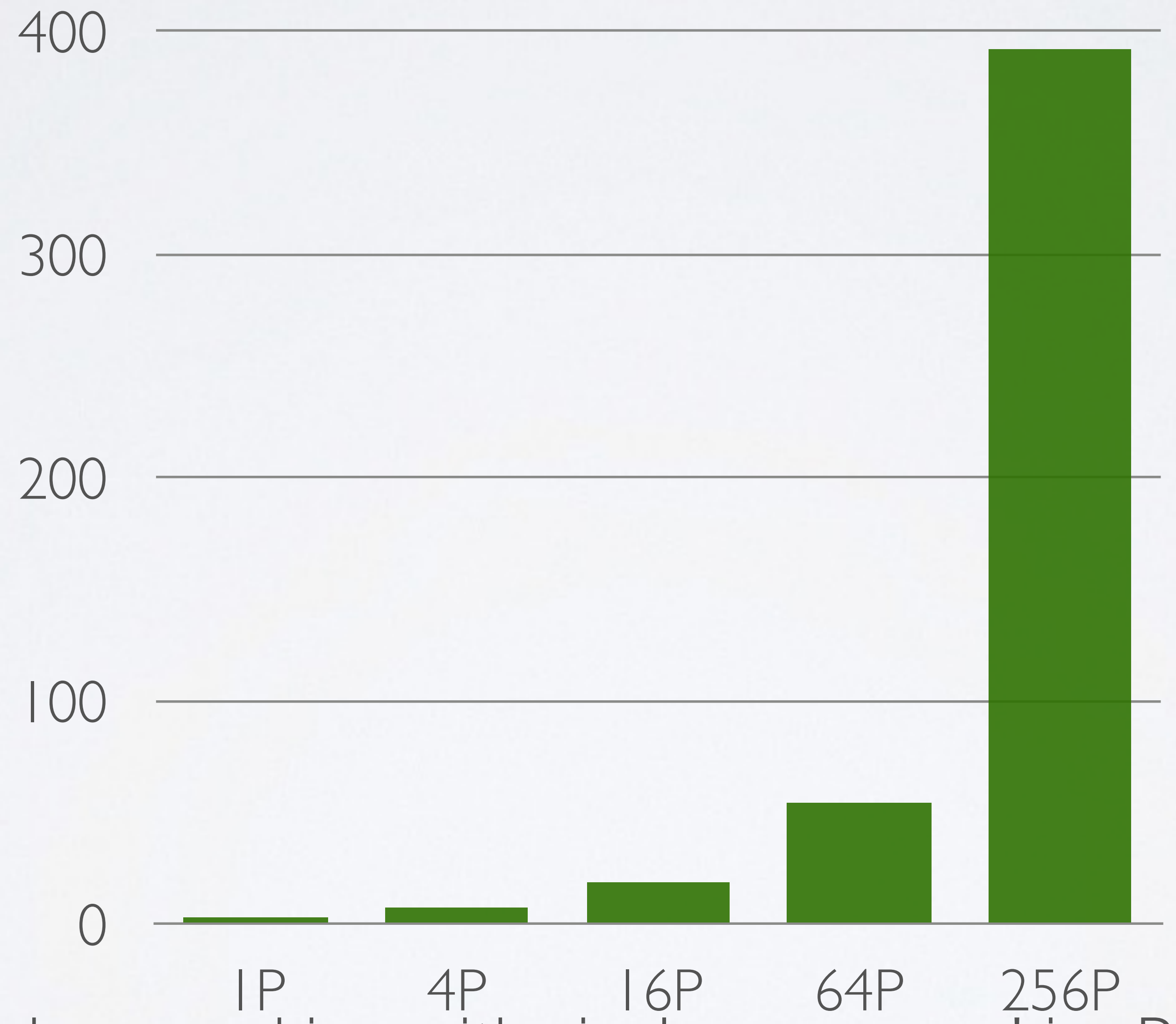


PARALLEL EFFICIENCY



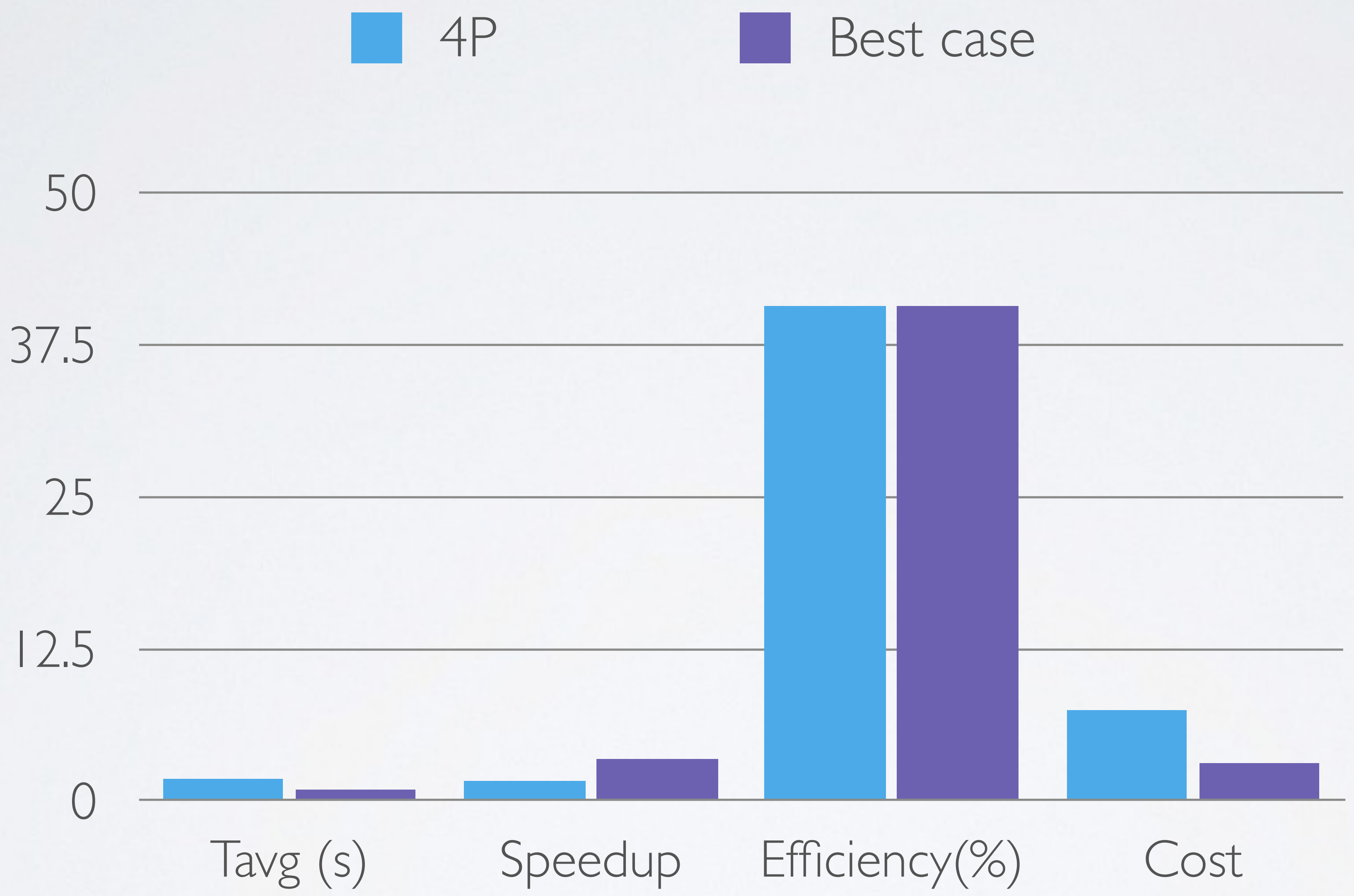
Parallel efficiency is found to be more in the case of 4 Processor!

COST ANALYSIS



Cost is very less when working with single processor! i.e. RAM implementation is efficient for this case!

CHOOSING THE BEST



Choosing 4 Processors seems to be a better solution for computing line segment problem considering a balance between running time, speedup, parallel efficiency & cost.

CONCLUSIONS

- Having more data per processor has it's own limitations. The more the data, communication time dominates the computation time as the processors communicate among each other during computation of parallel prefix operation globally. This case is very evident in case of 256 processors.

- Computing line segment problem is efficient in case of 4 Processors for a particular set of data considering running time, speedup, parallel efficiency & cost.

CHALLENGES FACED

- Parallel code is pretty tough to debug! I have been getting lot’s of segmentation faults during coding
- Preparing data for this project was a difficult one. I wrote a separate code to write data to a file on disk and used that file as the input to the main code

FUTURE WORK

- Try to use more varieties of test data and run rigorous number of tests (more than 50) to obtain a fair running time of the program
- Implementation of same algorithm using OpenMP

REFERENCE

- Miller, Russ and Laurence Boxer. Algorithms Sequential and Parallel: A Unified Approach, Third Edition
- Class slides by Dr. M. D. Jones, Ph.D.
- http://mpitutorial.com/beginner-mpi-tutorial/

THANK YOU