CSE 633 - Parallel Algorithm

Parallel BFS 1D Partitioning

Under guidance of Dr. Russ Miller

Submitted by – Arshabh Semwal
UB Person Number - 50419031

Motivations

Graph processing operates on a large volume of highly connected data.

Real-world applications of graph processing includes:

- Social network
- Digital maps
- Webpage hyperlinks
- Very Large-Scale Integration (VLSI) layout of integrated circuit (IC) and more

Applications of BFS

Finding Shortest Path: In an unweighted graph, the shortest path is the path with least number of edges. With Breadth First, we always reach a vertex from given source using the minimum number of edges.

Finding Minimum Spanning Tree for unweighted graph In an unweighted graph, in case of unweighted graphs, any spanning tree is Minimum Spanning Tree, and we can use either Depth or Breadth first traversal for finding a spanning tree.

Applications of BFS

Peer to Peer Networks: In Peer-to-Peer Networks like BitTorrent, Breadth First Search is used to find all neighbor nodes.

Social Networking Websites: In social networks, we can find people within a given distance ‘k’ from a person using Breadth First Search till ‘k’ levels.

GPS Navigation systems: Breadth First Search is used to find all neighboring locations.

Broadcasting in Network: In networks, a broadcasted packet follows Breadth First Search to reach all nodes.

Sequential BFS

BredthFirstSerach(**G, A**): //G is graph and A is source node

1. Let **Q** be the queue
2. **Q**.enqueue(**A**)
3. Mark **A** node as visited.
4. While (**Q** is not empty)
5. **B** = **Q**.dequeue( )
6. Processing all the neighbors of **B**
7. For all neighbors **C** of **B**
8. If **C** is not visited, **Q**. enqueue(**C**)
9. Mark **C** a visited

Parallel BFS 1-D Partition

The main steps of BFS traversal in the following algorithm are:

- Construct the frontier with vertexes from local storage.
- Terminate the traversal if frontier from all processors are empty.
- Construct the next frontier based on the neighbor's vertex of its frontier, although some of their neighbors may be stored in other processors.
- Run an all-to-all communication to let each processor know, which local vertexes should be put into its local next-frontier.
- Receive messages from all other processors, update the distance value of their local vertexes in the current frontier, change its next-frontier to next-frontier.

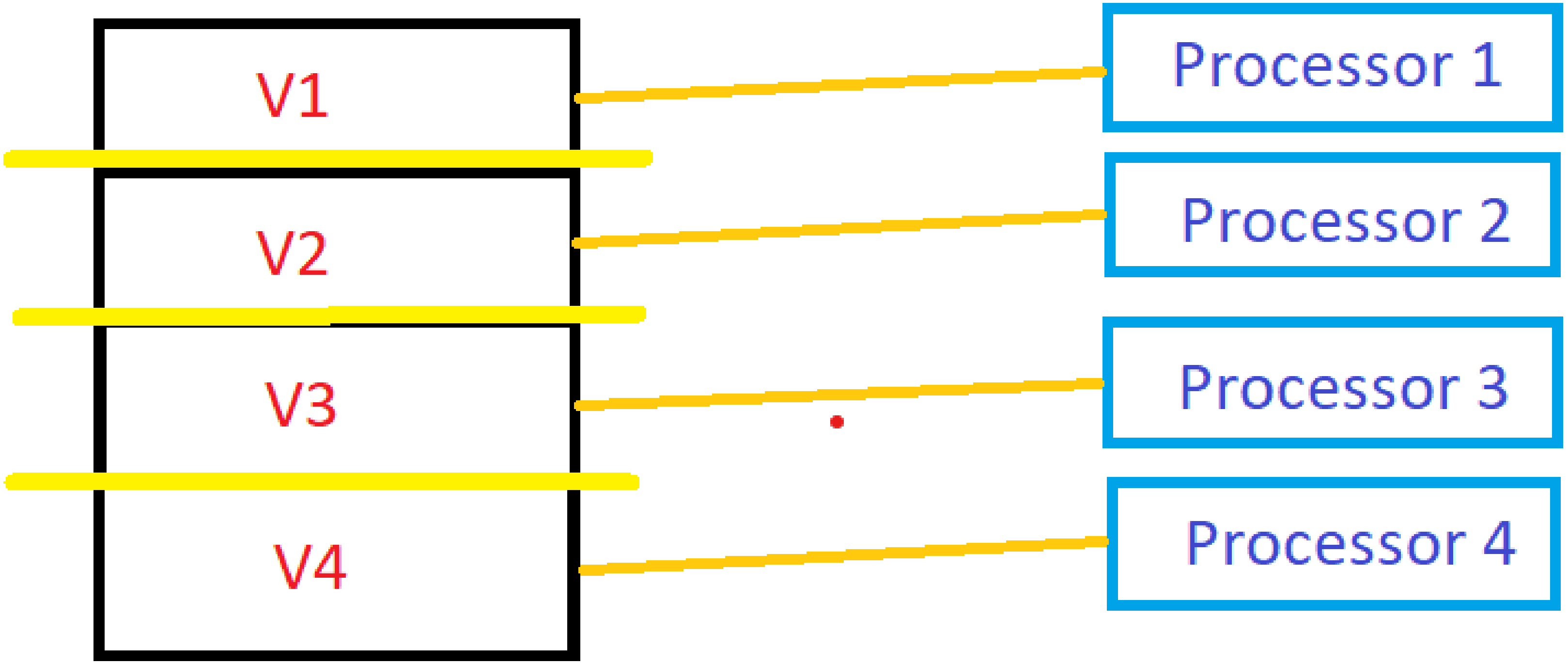
1-D Partition Parallel BFS Algorithm

```
define 1_D_distributed_memory_BFS( graph(V,E), source s):
    //normal initialization
    for all v in V do
        d[v] = -1;
    d[s] = 0; level = 0; FS = {}; NS = {};
    //begin BFS traversal
    while True do:
        FS = {the set of local vertexes with level}
        //all vertexes traversed
        if FS = {} for all processors then:
            terminate the while loop
        //construct the NS based on local vertexes in current frontier
        NS = {neighbors of vertexes in FS, both local and not local vertexes}
        //synchronization: all-to-all communication
        for 0 <= j < p do:
            N_j = {vertexes in NS owned by processor j}
            send N_j to processor j
            receive N_j_rcv from processor j
        //combine the received message to form local next vertex frontier then update the level for them
        NS_rcv = Union(N_j_rcv)
        for v in NS_rcv and d[v] == -1 do
            d[v] = level + 1
```

Image Source Wikipedia: https://en.wikipedia.org/wiki/Parallel_breadth-first_search

1-D Partition of Data

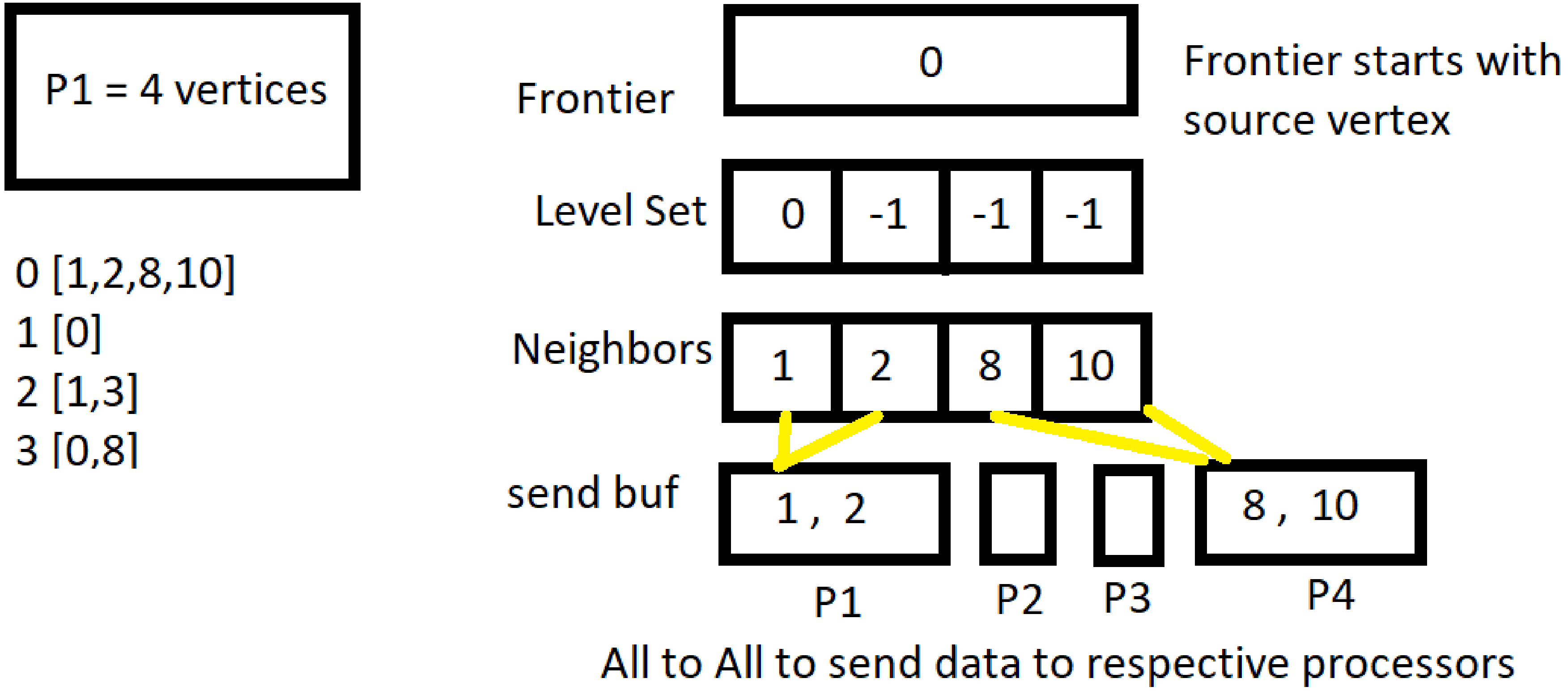
Graph G is split into 4 part and sent to each of the four processors



The split is vertex only

Working of Algorithm

The algorithm is like sequential BFS but is adapted to send and receive neighbors' data from other processors.



Data Used

Two data are used:

1. SNAP California road graph:

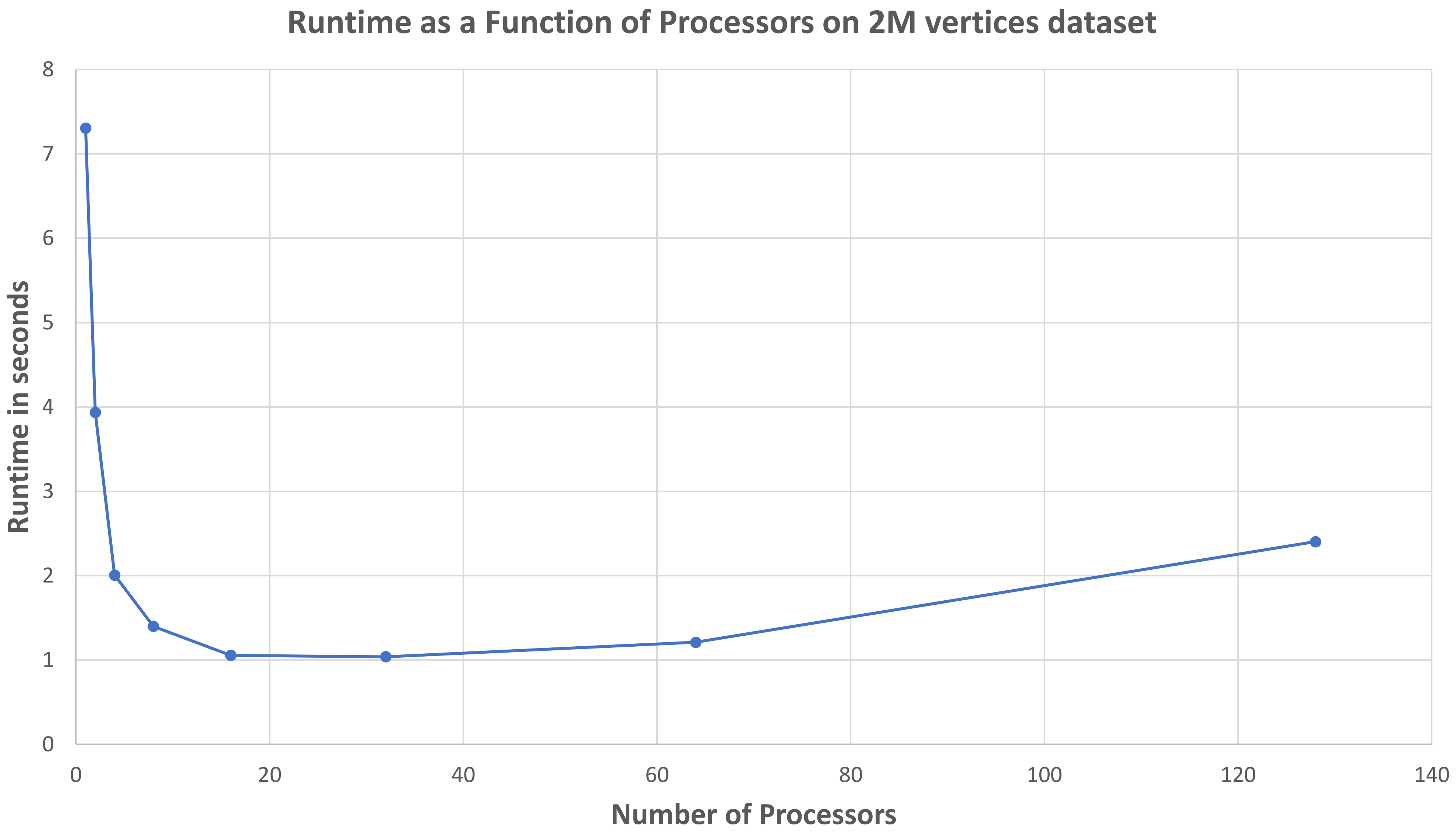
   Dataset statistics

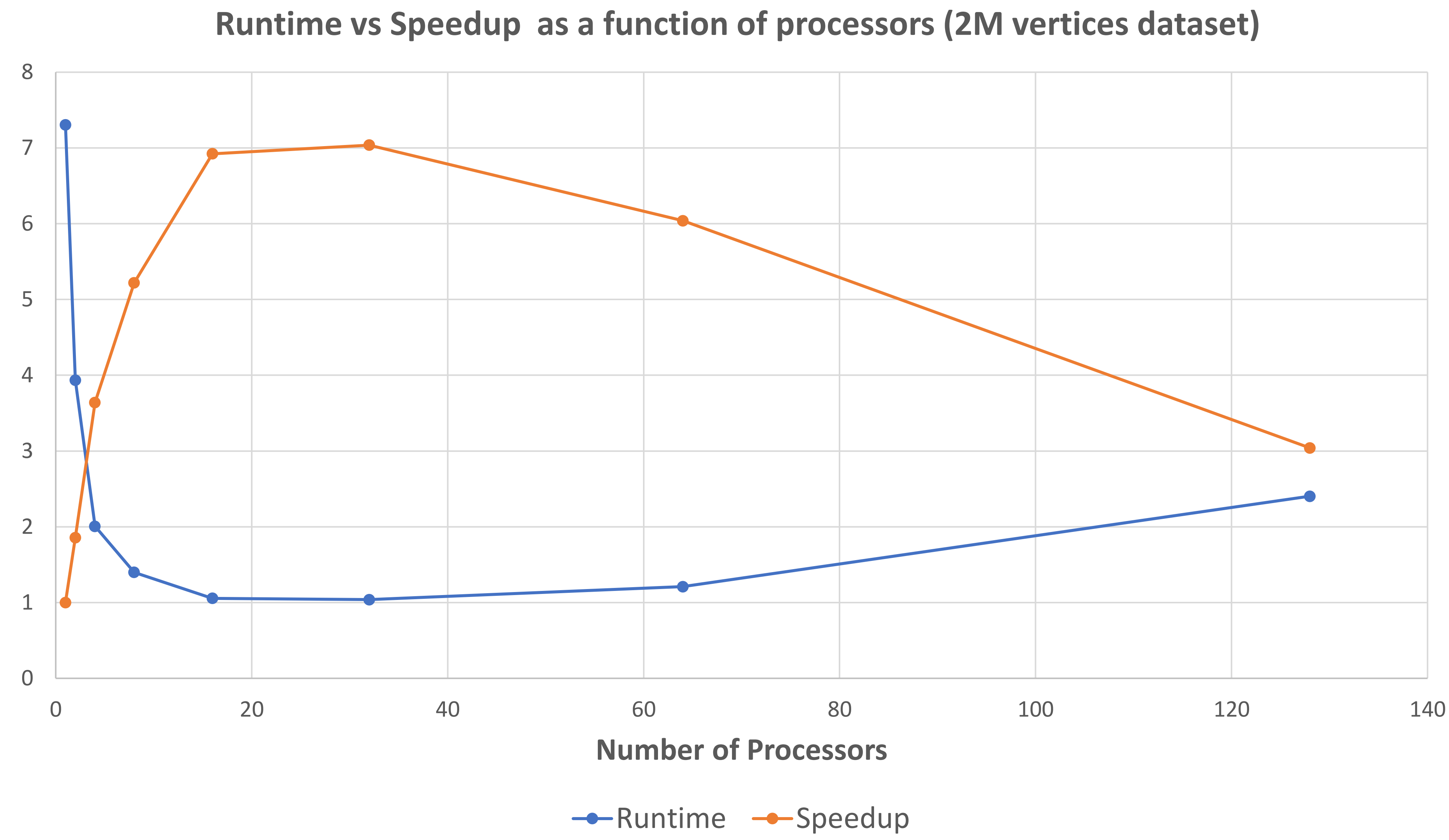
   - Vertices - 1971281
   - Edges - 2766607
   - Diameter (longest shortest path) – 849

2. The Network Data Repository with Interactive Graph Analytics and Visualization (Road Network):

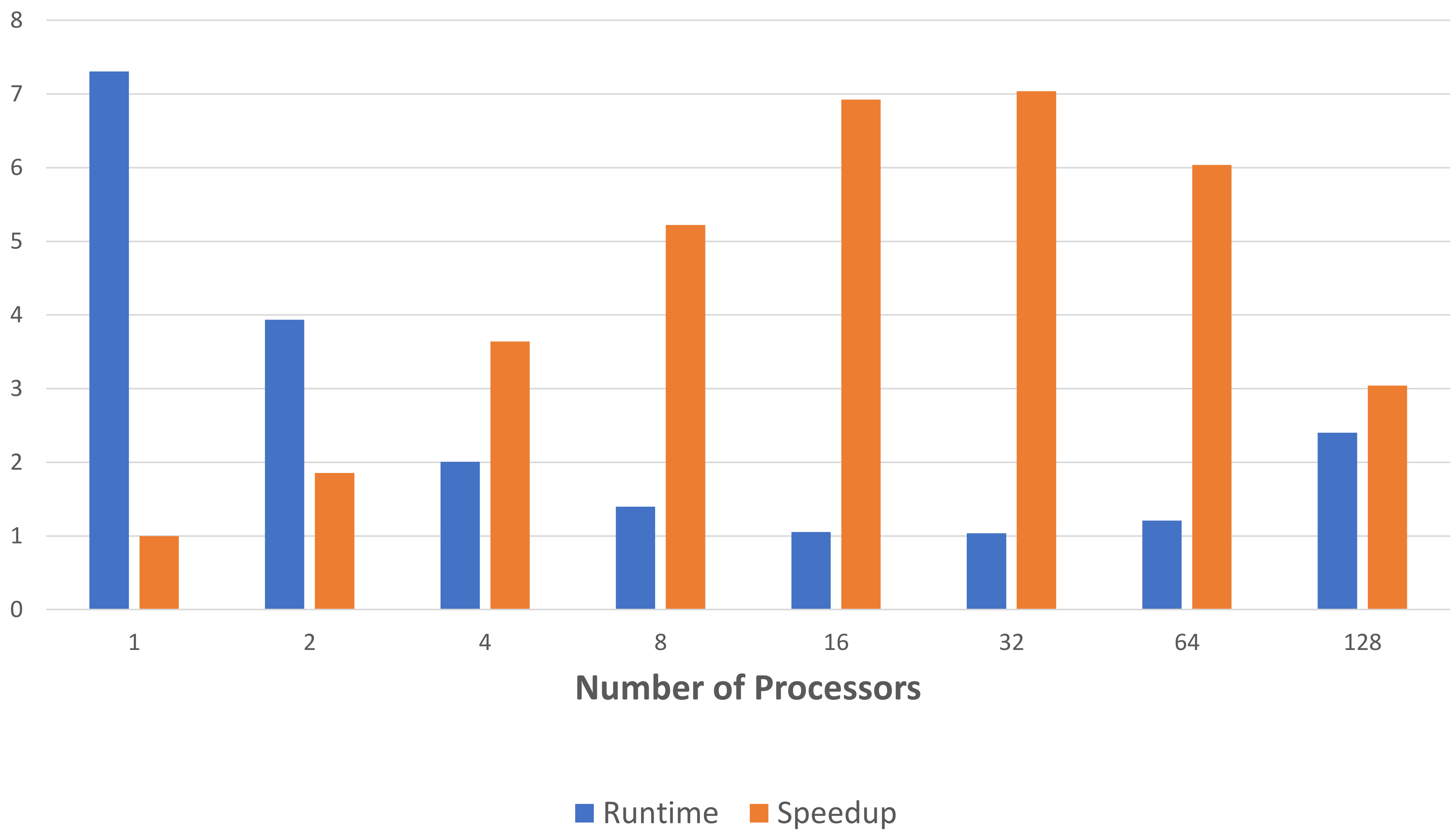
   Dataset statistics

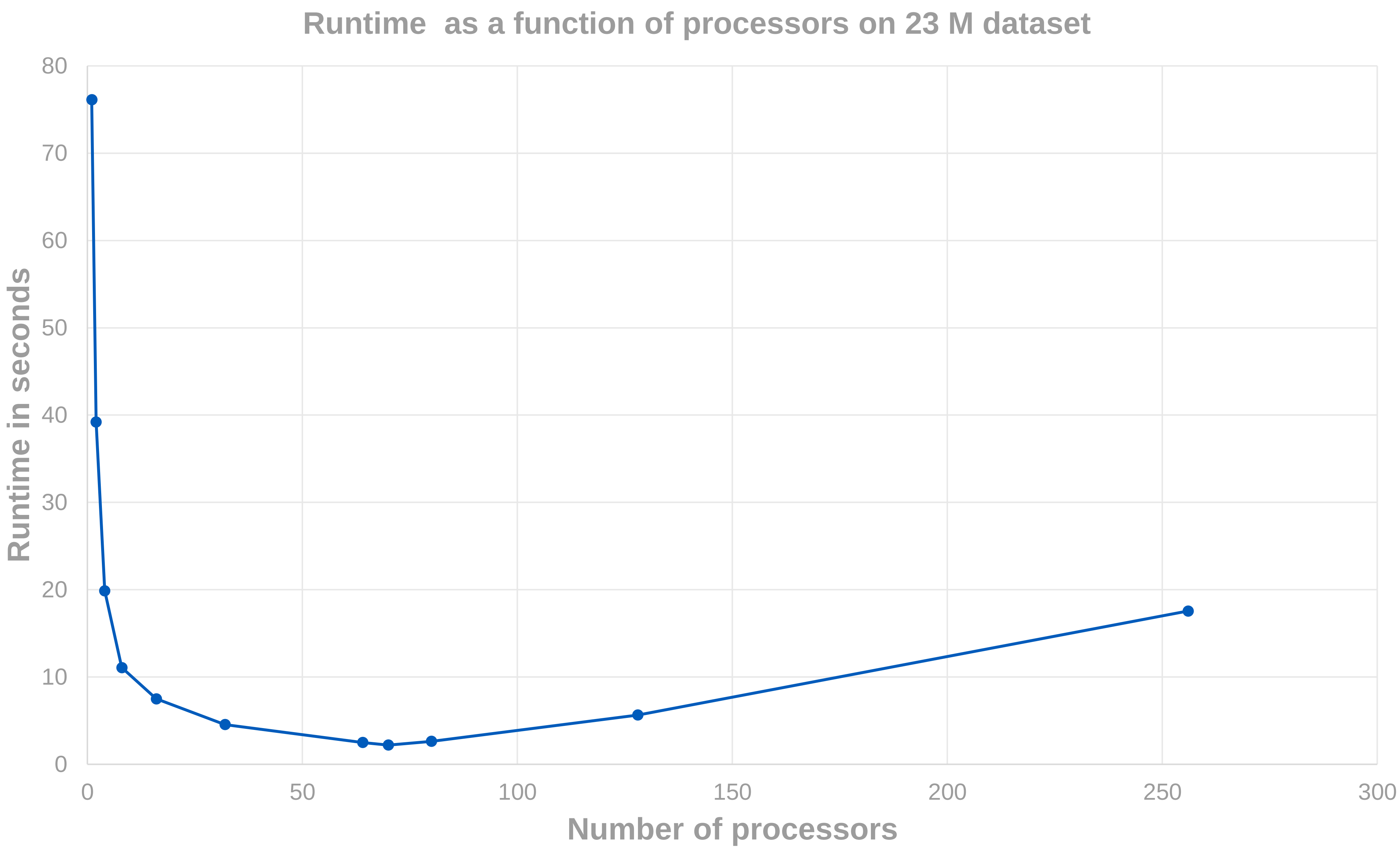
   - Vertices – 23.9 M
   - Edges – 28.9 M
   - Density - 1.0063e-07

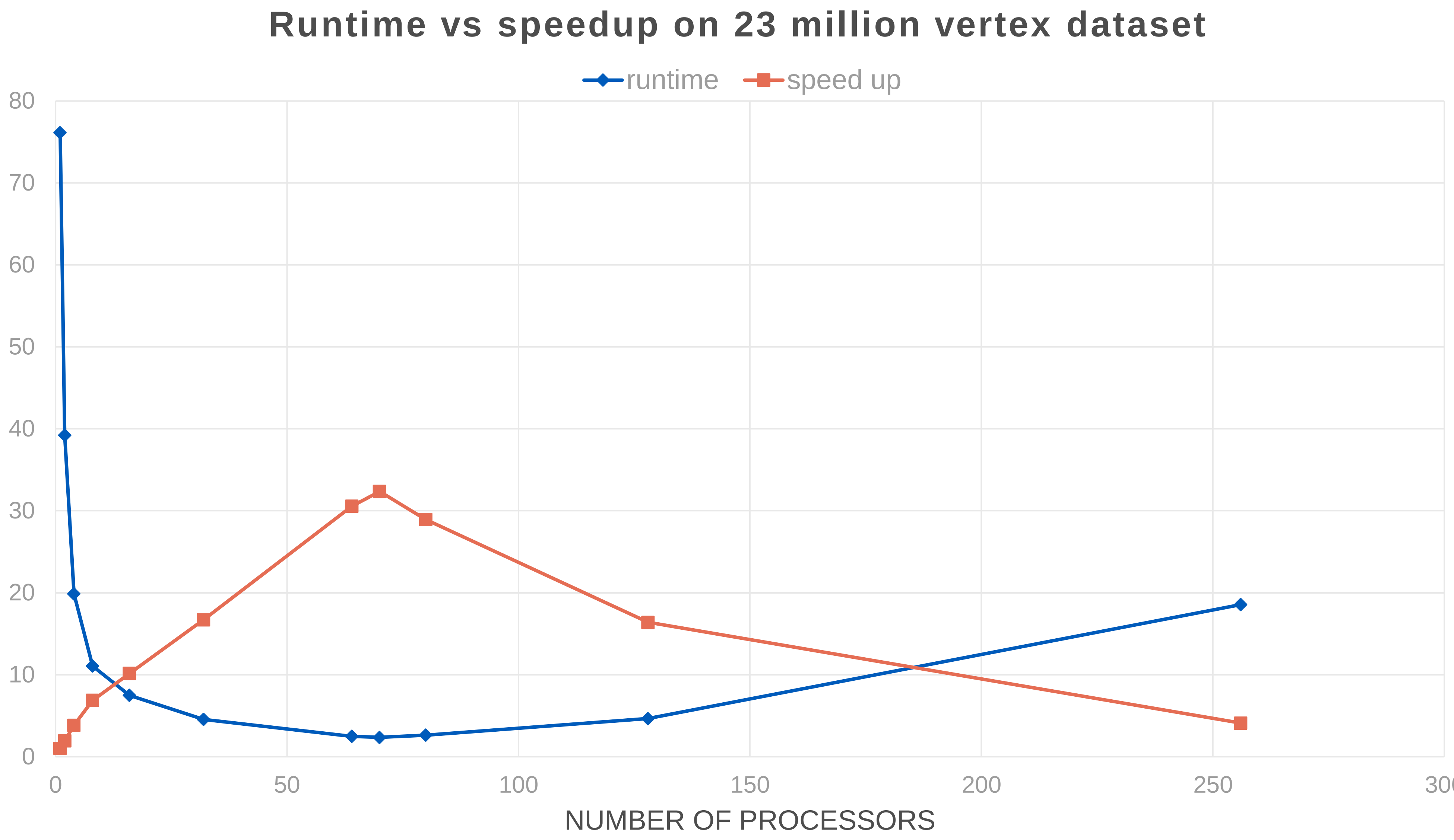

Runtime as a Function of Processors on 2M vertices dataset
Runtime in seconds
0
1
2
3
4
5
6
7
8
0
20
40
60
80
100
120
140
Number of Processors


Runtime vs Speedup as a function of processors (2M vertices dataset)
8
7
6
5
4
3
2
1
0
0
20
40
60
80
100
120
140
Number of Processors
Runtime
Speedup

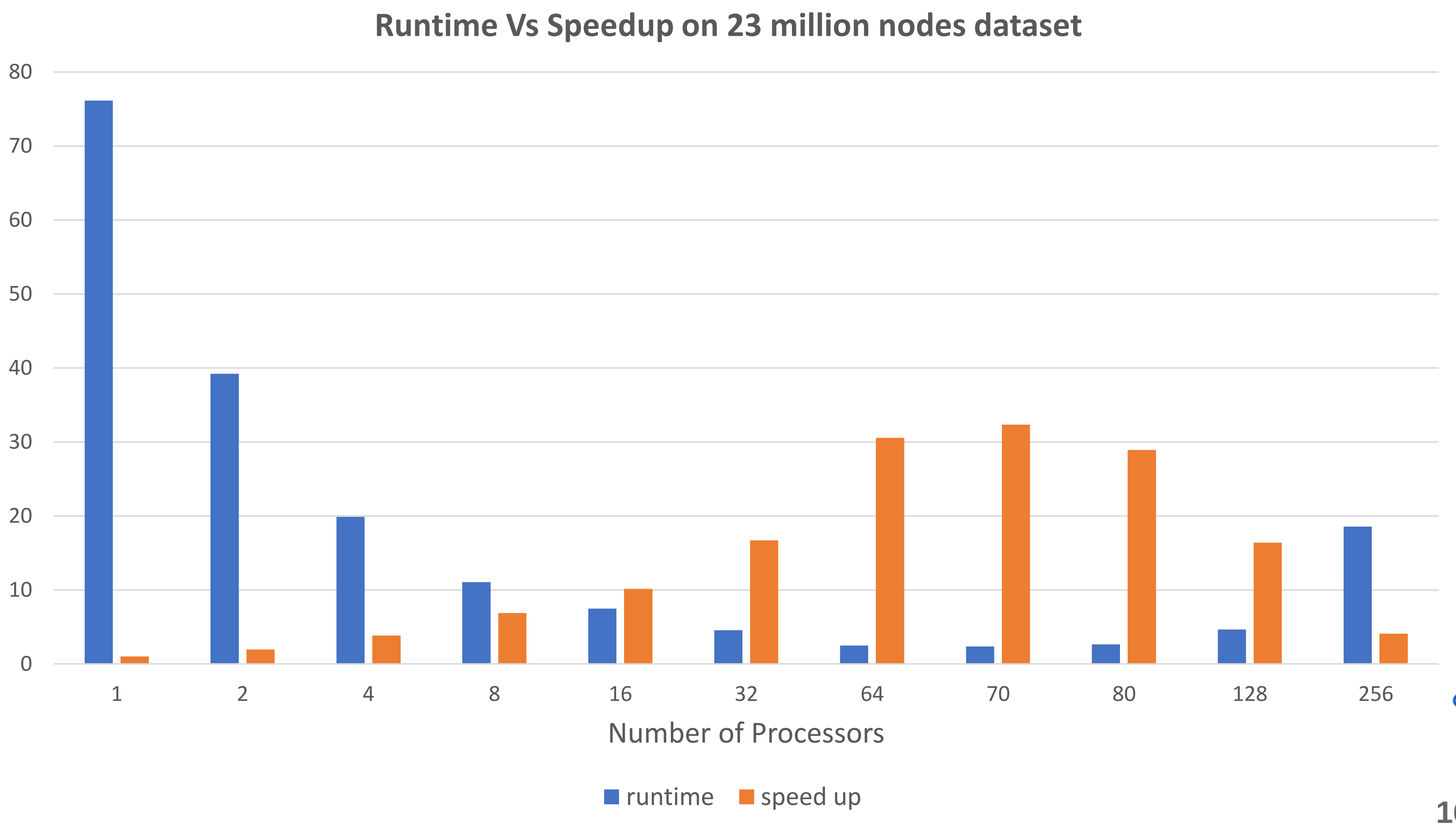
Runtime Vs Speedup on 23 million nodes dataset




Runtime as a function of processors on 23 M dataset
Runtime in seconds
Number of processors
0
10
20
30
40
50
60
70
80
0
50
100
150
200
250
300



Runtime vs speedup on 23 million vertex dataset
runtime
speed up
80
70
60
50
40
30
20
10
0
0
50
100
150
200
250
300
NUMBER OF PROCESSORS


Runtime Vs Speedup on 23 million nodes dataset
80
70
60
50
40
30
20
10
0
1
2
4
8
16
32
64
70
80
128
256
Number of Processors
runtime
speed up

Results:

1. For 2M vertex dataset the algorithm run optimally till 32 processors after which the time consumed during inter-processor communication exceeded the speedup gained by parallel computation.

2. For 23M vertex dataset the algorithm gained speedup till 70 processors after which the time consumed during inter-processor communication exceeded the speedup gained by parallel computation.

3. It is observed that the data used is very sparsely populated which is contributing negatively to the performance of algorithm. It is assumed that a densely populated graph will perform better and might gain speed up beyond the current processor threshold as a densely populated graph will require more computation than a sparsely populated graph.

References

1. Snap California Road Dataset. J. Leskovec, K. Lang, A. Dasgupta, M. Mahoney. Community Structure in Large Networks: Natural Cluster Sizes and the Absence of Large Well-Defined Clusters. Internet Mathematics 6(1) 29--123, 2009. https://snap.stanford.edu/data/roadNet-CA.html

2. Parallel BFS 1-D Partition - https://en.wikipedia.org/wiki/Parallel_breadth-first_search
3. The Network Data Repository with Interactive Graph Analytics and Visualization- road dataset by Ryan A. Rossi and Nesreen K. Ahmed. https://networkrepository.com/road-road-usa.php