PARALLEL A* ALGORITHM

CSE 708

Harsahavardan Ramadas

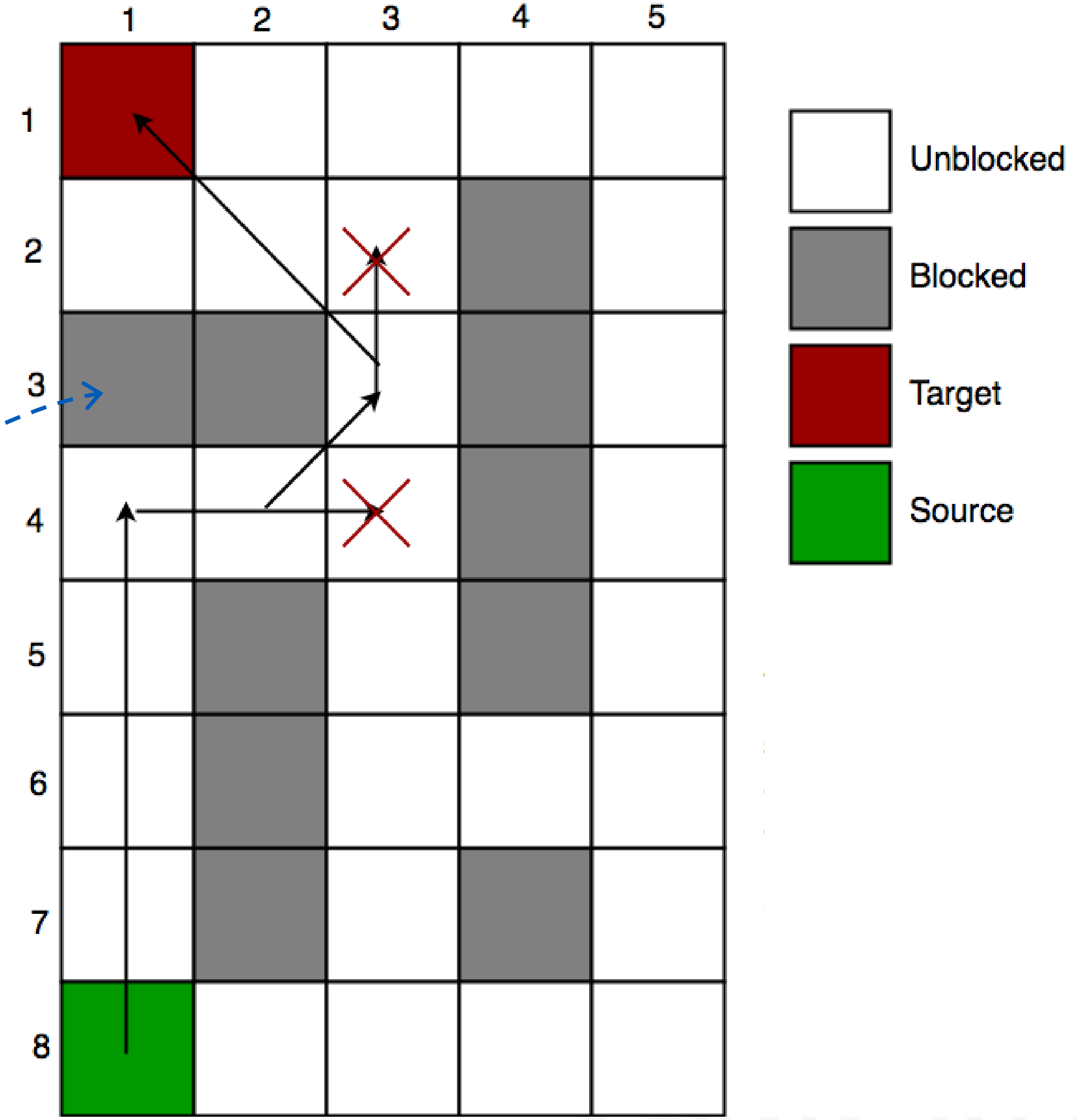
Instructor: Dr. Russ Miller

University at Buffalo The State University of New York

The Problem

Goal : To find the shortest(best) path between 2 nodes(or cells) in a connected graph(or grid)

Constraint : Cannot travel on blocked cells(wall) in the grid



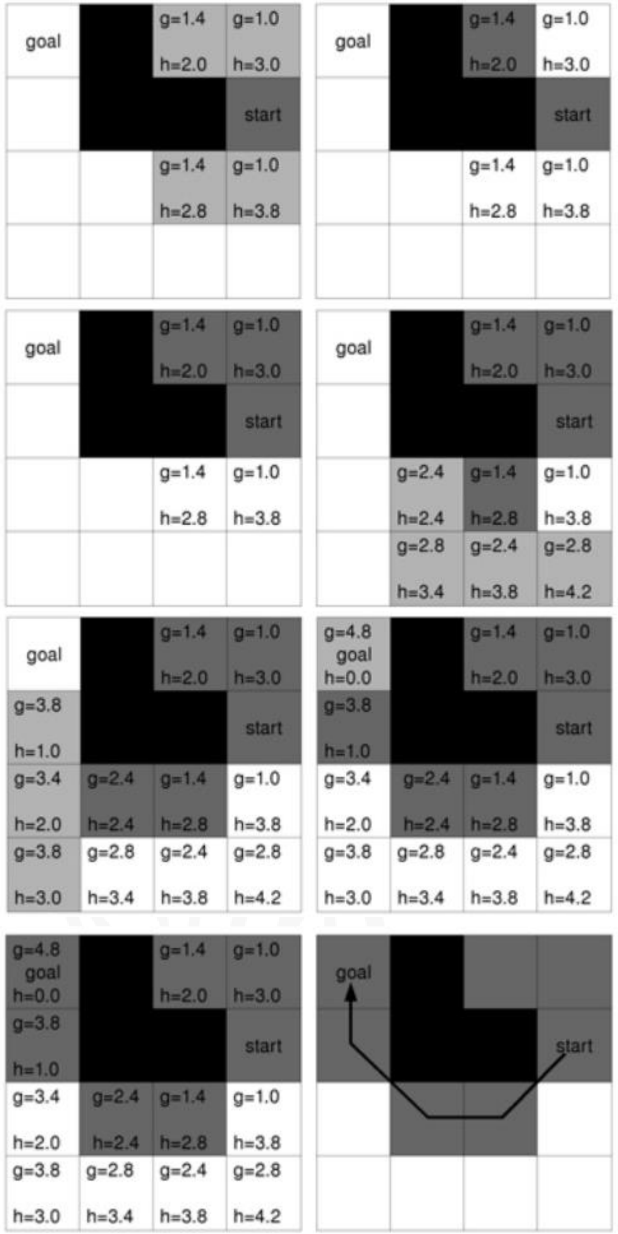
The Solution – A*

- Path finding algorithm (can be seen as extension of Dijkstra's algorithm)
- Cost estimated using the function:

  f(n) = g(n) + h(n)

  g(*n*) => cost so far to reach node *n*

  h(*n*) => estimated cost from *n* to goal.
- Heuristic function to estimate cost - Manhattan, Diagonal, Euclidean

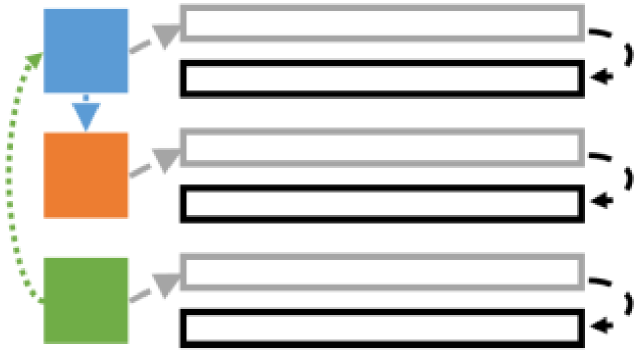


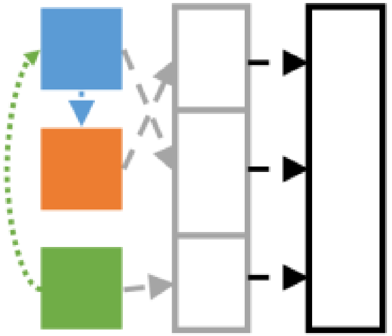
Sequential Approach - Pseudo code

```
A* (start, goal)
Closed set = the empty set
Open set = includes start node
G[start] = 0, H[start] = H_calc[start, goal]
F[start] = H[start]
While Open set ≠ Ø
    do CurNode ← EXTRACT-MIN- F(Open set)
    if ( CurNode  == goal ), then return BestPath
    For each Neighbor Node N of CurNode
        If ( N is in Closed set ), then Nothing
        else if ( N is in Open set ),
            calculate N's G, H, F
            If ( G[N on the Open set] > calculated G[N] )
                RELAX(N, Neighbor in Open set, w)
                N's parent=CurNode & add N to Open set
        else, then calculate N's G, H, F
                N's parent = CurNode & add N to Open
```

Parallel Approach

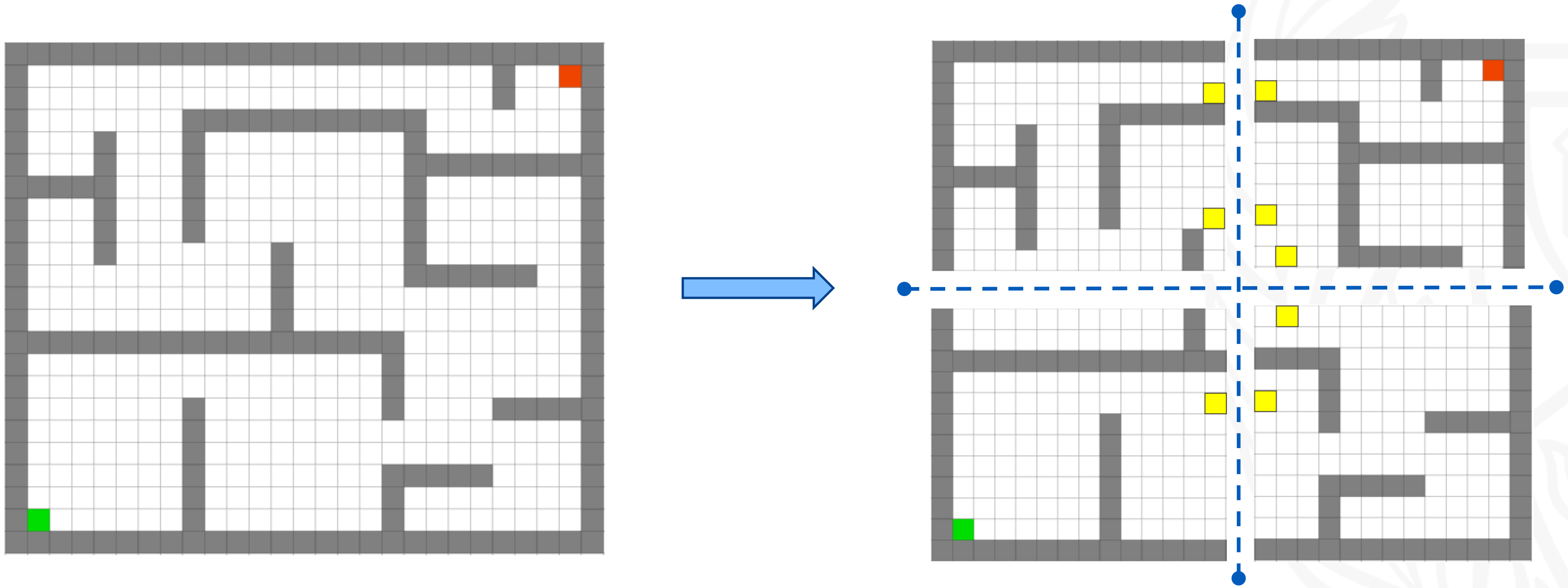
- Graph(size NxN) is divided into equal size sub-graphs and assigned to different processors
- For each sub-graph, there are a set of entry and exit points
- Every processor runs A* algorithm for the entry/exit points within each sub-graph based on global avg heuristic
- Processors communicate local paths(Queue) with each other
- When solution is found, broadcast and stop loop.





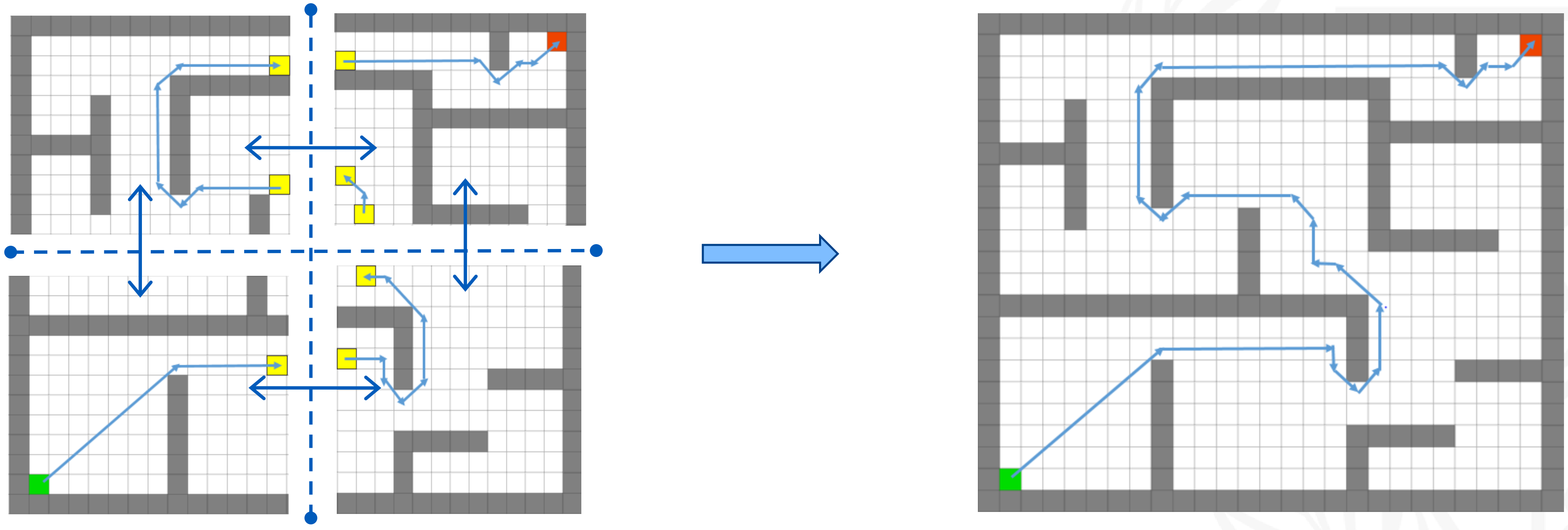
Parallel Approach

- Graph(size NxN) is divided into equal size sub-graphs and assigned to different processors
- For each sub-graph, there are a set of entry and exit points



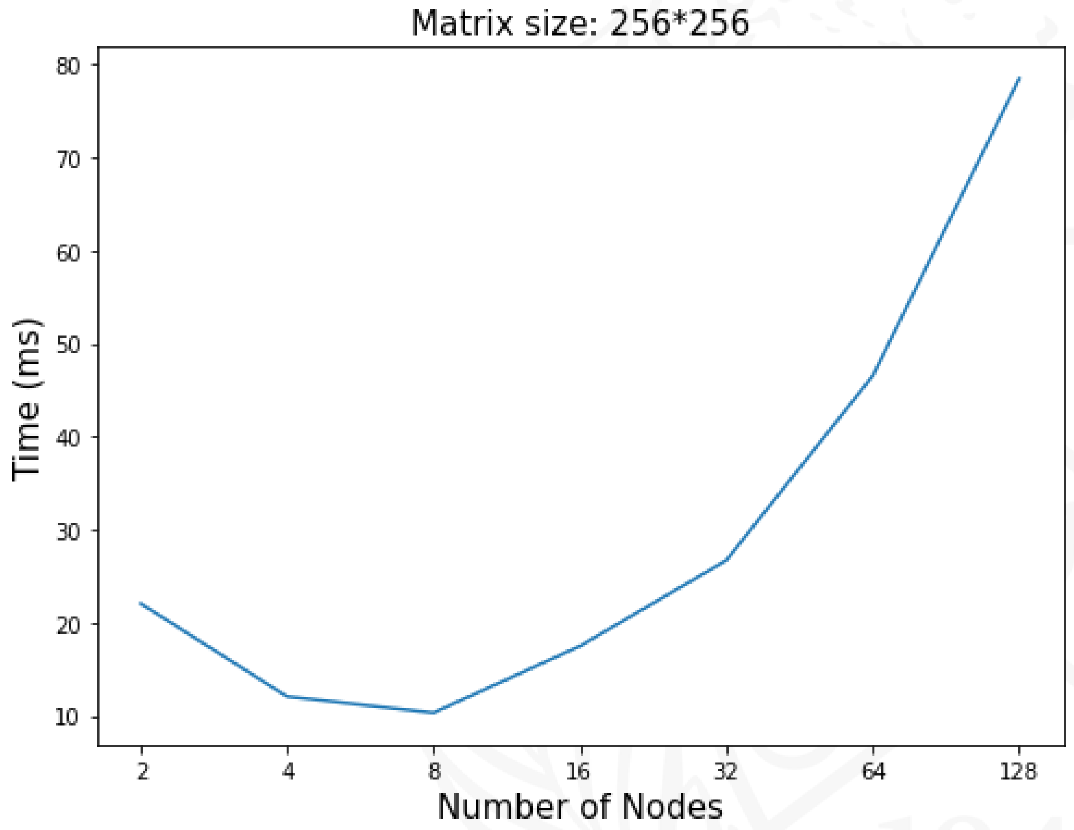
Parallel Approach

- Every processor runs A* algorithm for the entry/exit points and communicate local paths(Queue) with each other
- When solution is found, broadcast and merge paths.



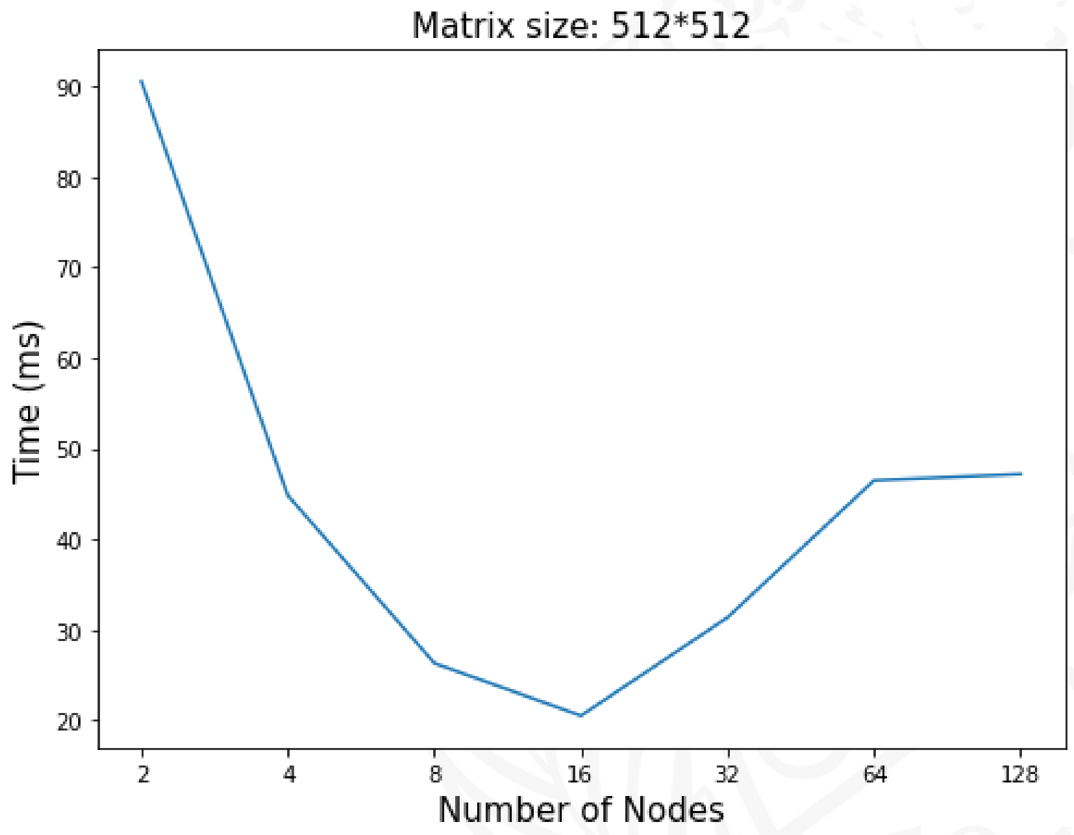
Results

Number of Nodes	Time (ms)
2	22.07
4	12.07
8	10.32
16	17.49
32	26.71
64	46.54
128	78.48



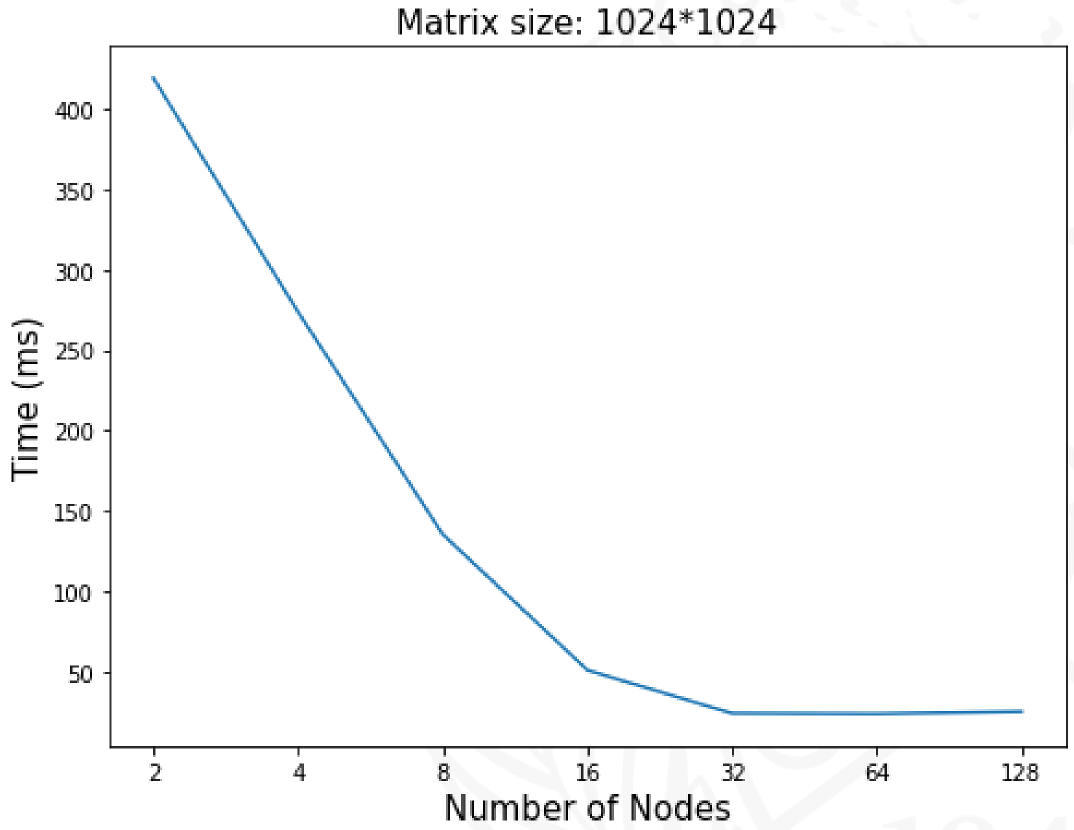
Results

Number of Nodes	Time (ms)
2	90.52
4	44.81
8	26.30
16	20.51
32	31.35
64	46.47
128	47.18



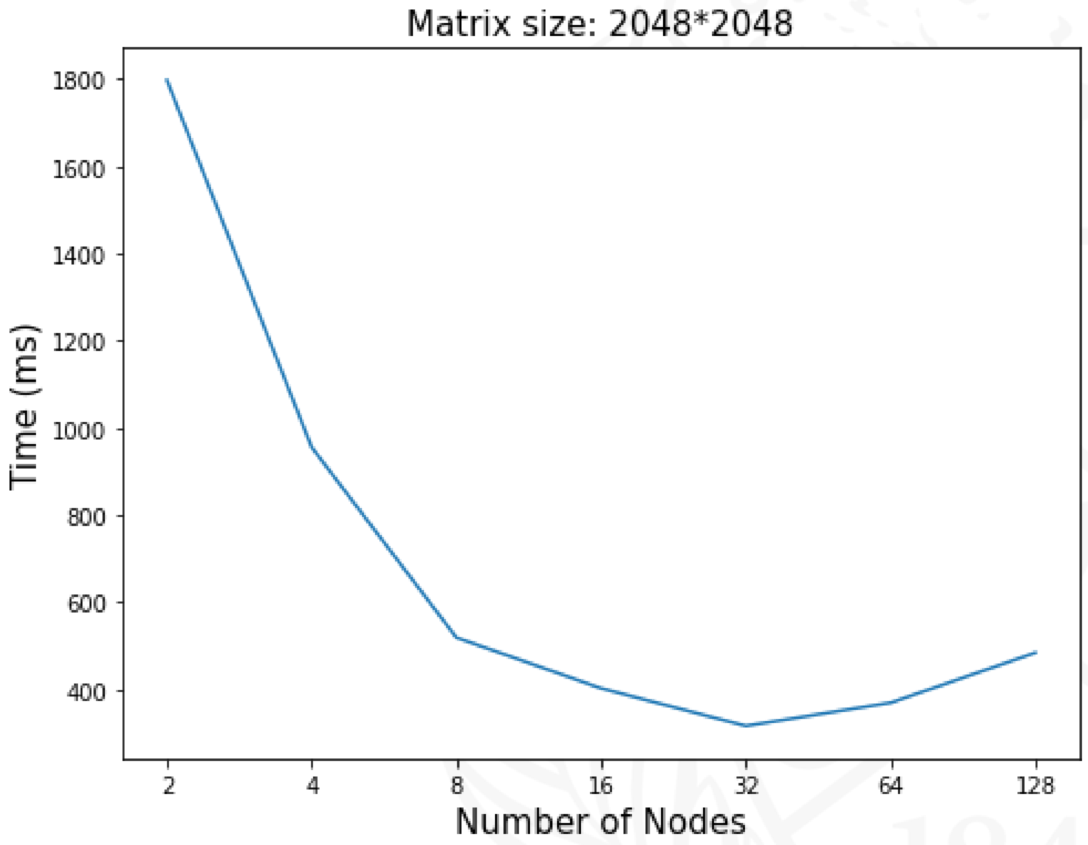
Results

Number of Nodes	Time (ms)
2	419.01
4	273.68
8	135.51
16	51.13
32	24.28
64	23.97
128	25.41



Results

Number of Nodes	Time (ms)
2	1797.48
4	955.69
8	518.88
16	402.51
32	316.08
64	369.35
128	484.11



Inference

- Better performance compared to sequential
- Results depend on the nature of the graph/Matrix
- Multiple methods for achieving parallelism

References

- Visuals, http://qiao.github.io/PathFinding.js/visual/
- Parallel A* Graph Search, Ariana Weinstock and Rachel Holladay , https://people.csail.mit.edu/rholladay/docs/parallel_search_report.pdf
- A* Algorithms, https://www.geeksforgeeks.org/a-search-algorithm/
- Parallel A* Search on Message Passing Architectures, https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=205103
- A* Path finding Project, https://arongranberg.com/astar/docs_beta/class_pathfinding_1_1_threading_1_1_parallel.html

Thank You!