CSE 633 Parallel Algorithms

Maze Generation and Solving Algorithm

By

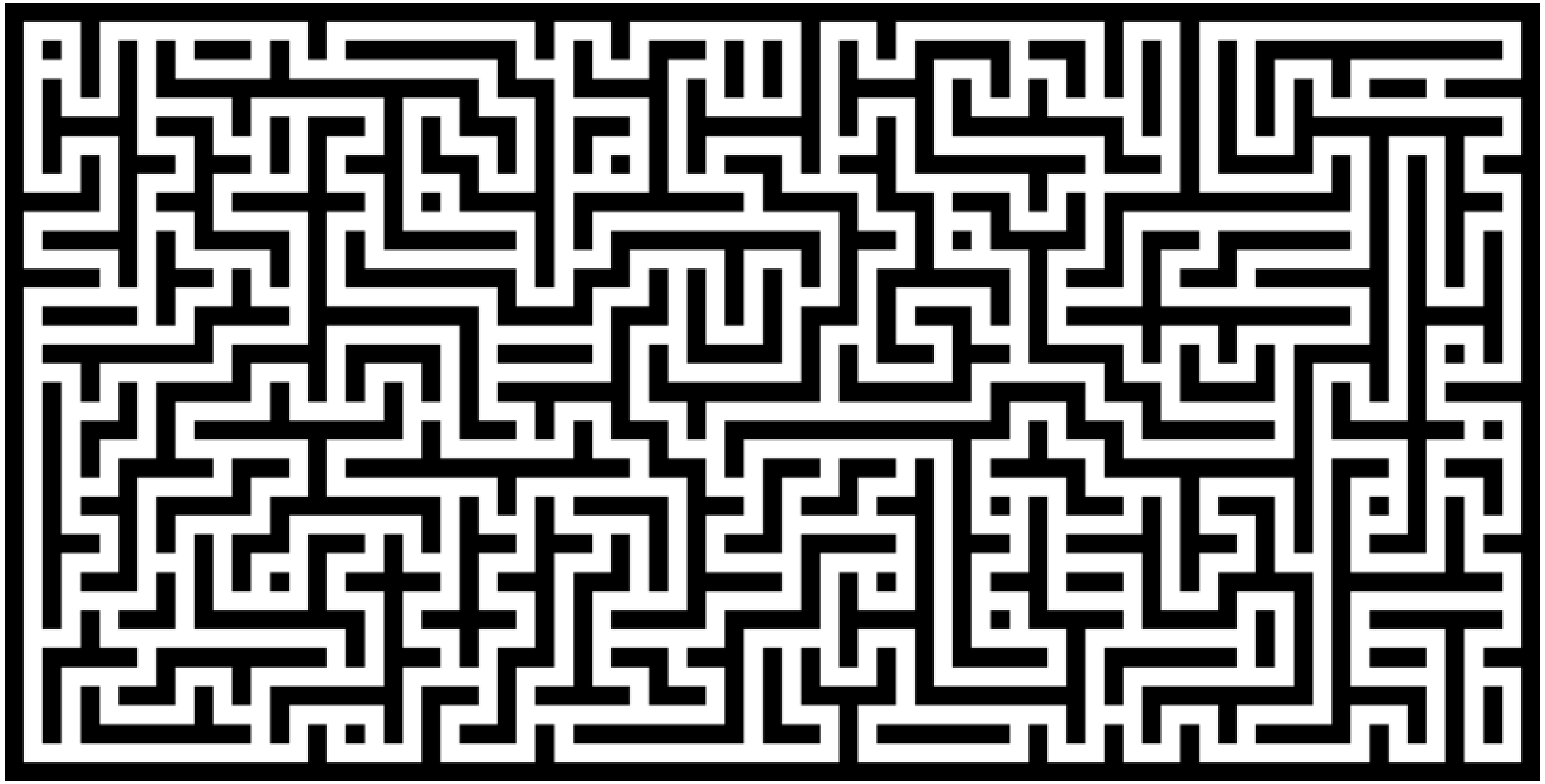
Divya Gorey

Sagar Vishwakarma

Saurabh Warvadekar

Shubhi Jain

Maze Generation and Solving



Algorithms Used

- Algorithms used:
  - Maze Generation
  - Maze Searching and Solving

Sequential Approach

Maze Generation Algorithm

- Sequential Approach

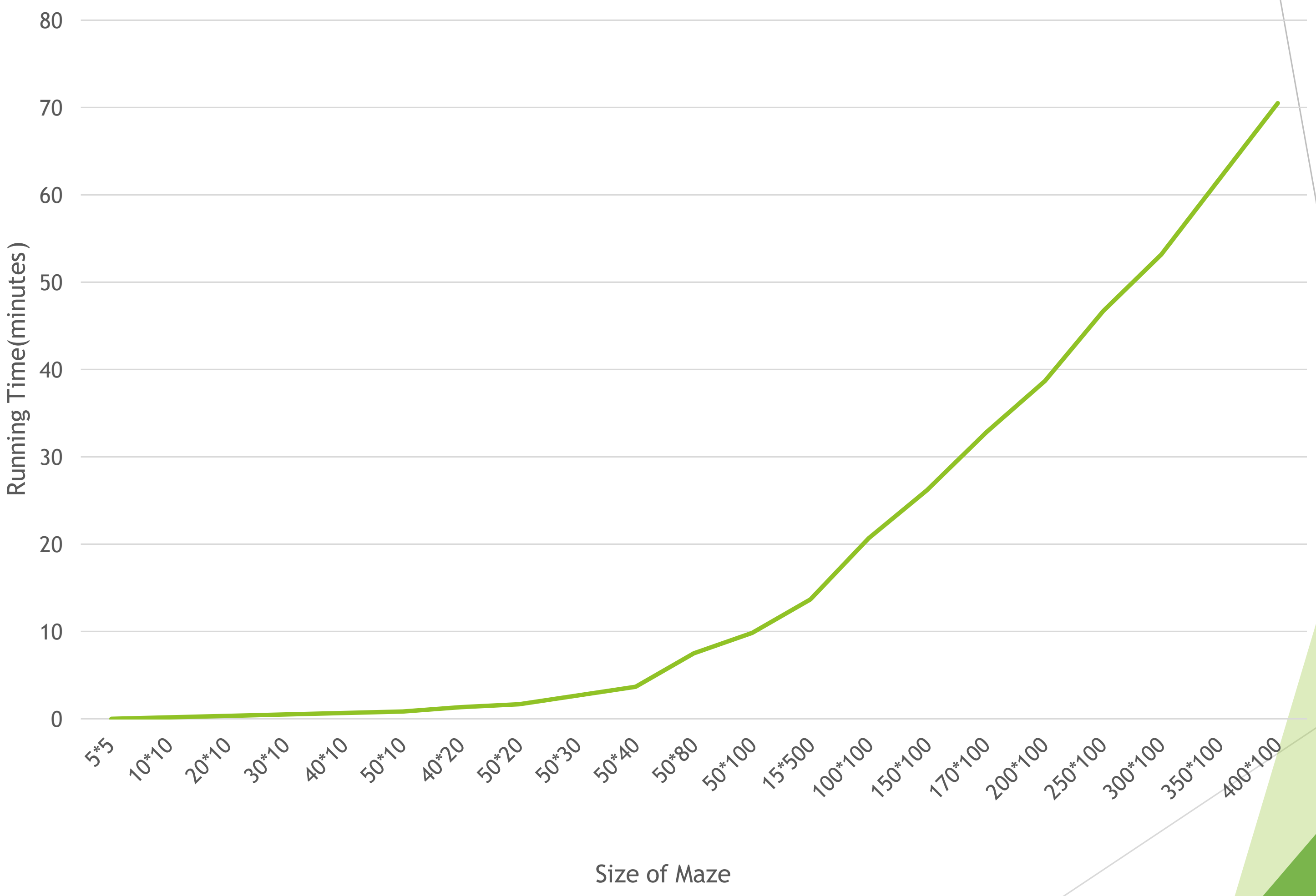
  - We are using Matrix to create a maze
  - Every 1 is a wall
  - Every 0 is a path

Pseudo Code Sequential Approach

- Initialize Matrix with 1
- Select a random x and y coordinate
- Check if x >0, y>0, x<maxx and y<maxy where maxx is the maximum value of x coordinate in the grid while maxy is the maximum value of y coordinate in the grid
- Check if any 2 neighbor is 0, don’t move ahead, otherwise initialize element as zero and call generate on all neighbors having value 1.
- Repeat Step 3

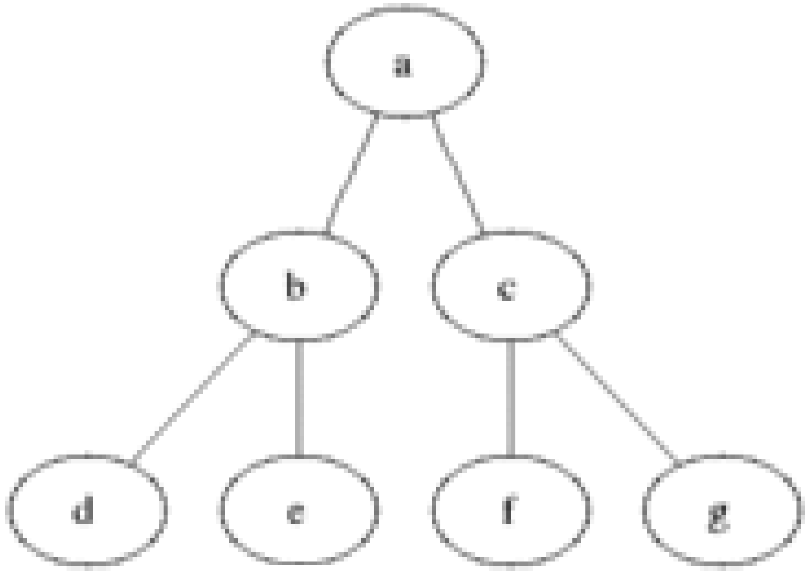
Sequential Maze Generation Graph

Size	Running Time(min)
10*10	0.136
50*10	0.805
50*100	9.76
15*500	13.66
100*100	20.61
150*100	26.21
300*100	53.13
350*100	61.84
400*100	70.54



Maze Solving Algorithm

- Sequential Approach

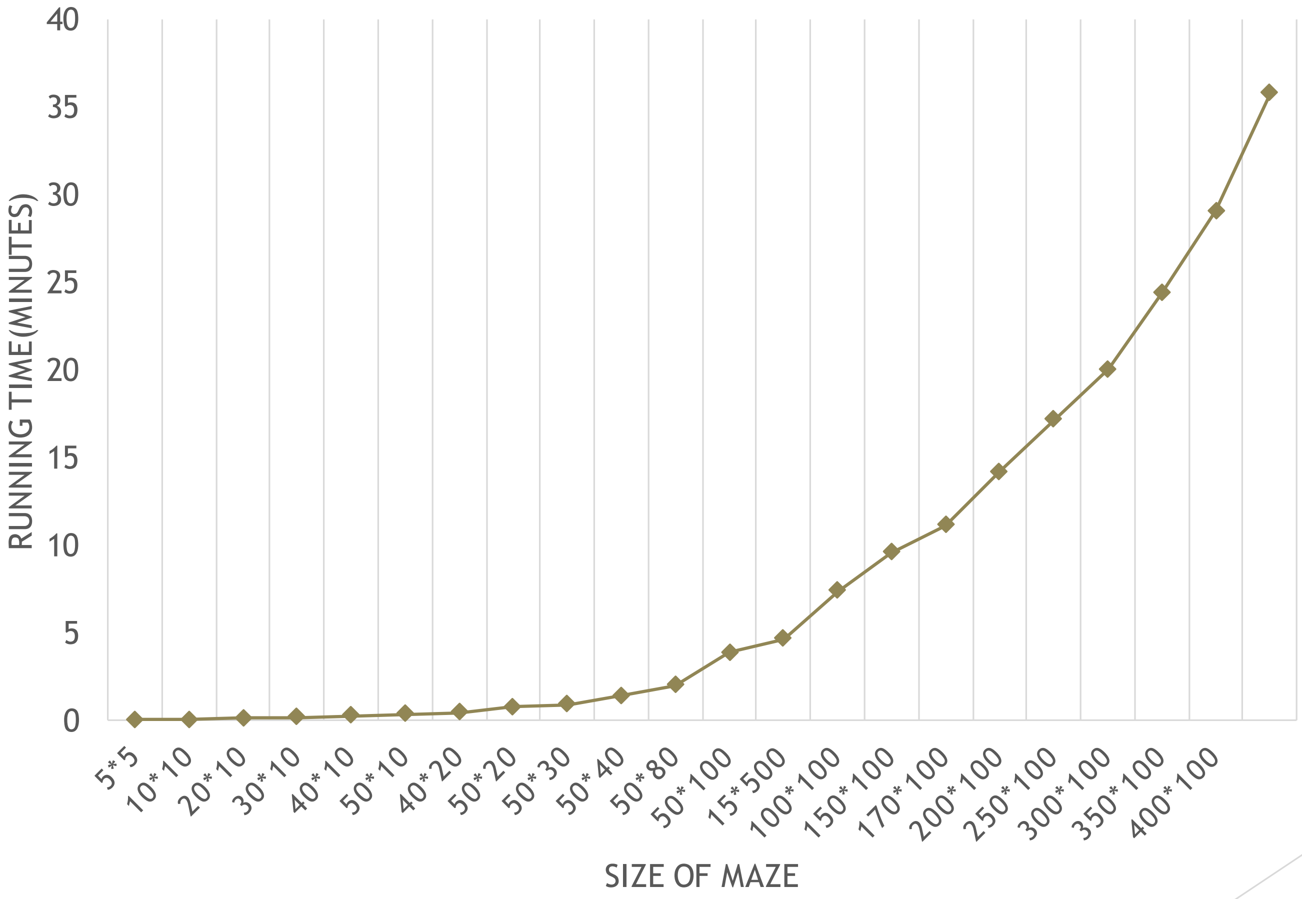


Implementing Searching Algorithm

1. Start from the starting coordinates given.
2. Check if x>0 , y>0 , x<maxx,y<maxy where maxx is the maximum value of x coordinate in the grid while maxy is the maximum value of y coordinate in the grid
3. If x, y = target coordinates return
4. If neighbor1 is 0 , add list to neighbor
5. If neighbor2 is 0, add list to neighbor and so on
6. Go back to step 2 with new coordinates of neighbors.

Maze Solving Sequential Approach Graph

Size	Running Time(min)
10*10	0.083
50*10	0.40
50*100	4.61
15*500	7.35
100*100	9.54
150*100	11.09
300*100	24.36
350*100	29.08
400*100	35.76



Parallel Approach

Assumptions

- Number of nodes can be taken as 2,4,8.. so on for log n approach and any number of nodes for master-slave
- Number of parts of Maze is a factor of size of maze
- All the vertices are joined vertically or horizontally(no diagonal component)
- Individual mazes are appended vertically downwards

Size of Maze for Parallel Operation

$$(N*X-(2N-2))*Y$$

- N=Number of Nodes
- X=Number of Rows in Single Maze
- Y=Number of Columns in Single Maze

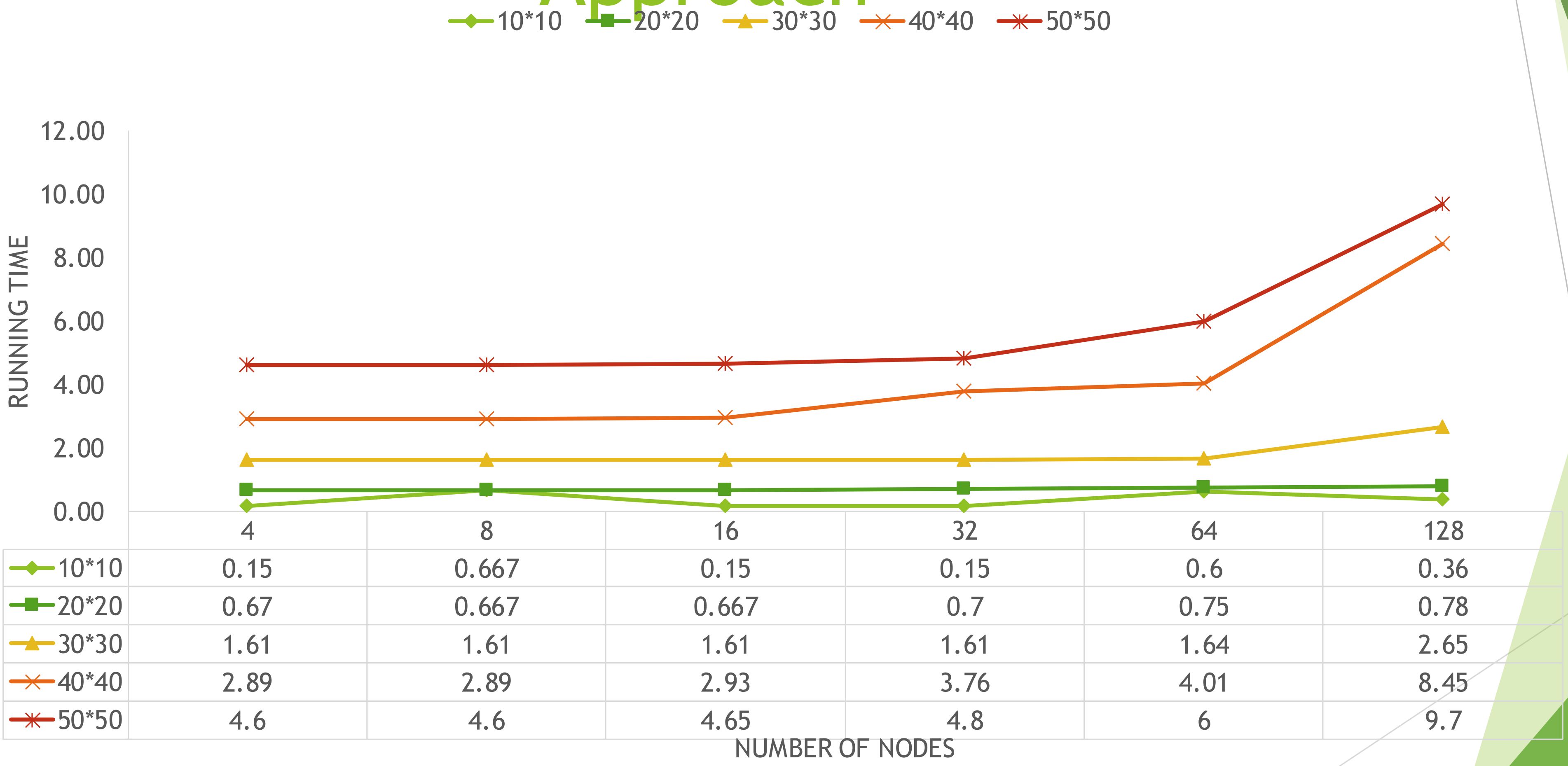
Master-Slave Approach

- Each node creates a maze of specified size
- All nodes generates maze in parallel
- 0th node is Master
- All other nodes sends its maze to master
- Master joins all the maze to a single maze

Pseudo Code Master-Slave Approach

```
If(myRank==0)     //Master Node
{
for(i=1 to n-1)
{
MPI_recv from each source
Append to previous maze
}}
else{                  //Slave Nodes
MPI_send to master
}
```

Maze Generation in Master-Slave Approach



	4	8	16	32	64	128
◆ 10*10	0.15	0.667	0.15	0.15	0.6	0.36
■ 20*20	0.67	0.667	0.667	0.7	0.75	0.78
▲ 30*30	1.61	1.61	1.61	1.61	1.64	2.65
× 40*40	2.89	2.89	2.93	3.76	4.01	8.45
✳ 50*50	4.6	4.6	4.65	4.8	6	9.7

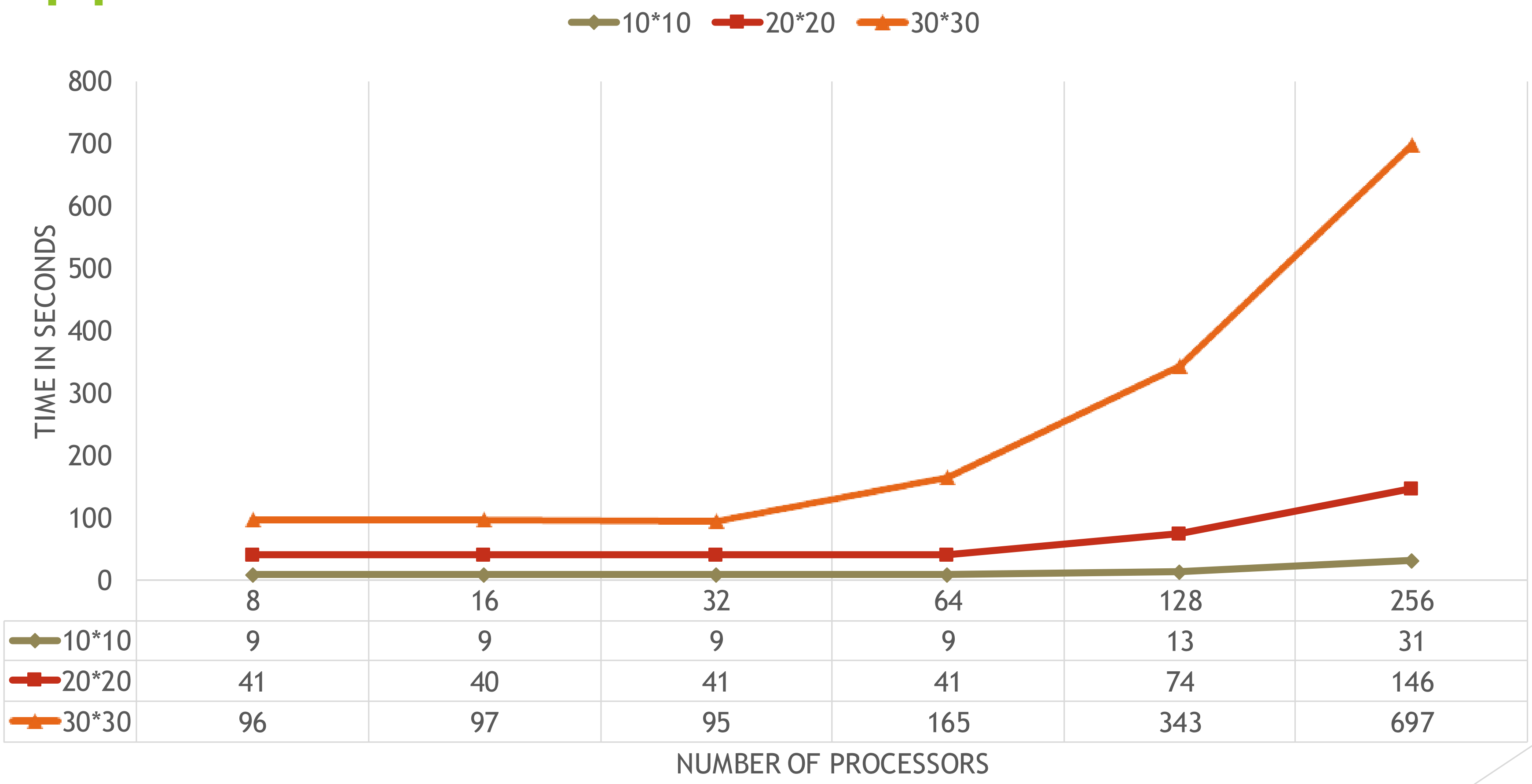
Log n level Approach

- Total log(no. of nodes) levels for sending and receiving messages
- All the odd number of nodes perform only send
- Some even nodes perform both single sending and multiple receiving
- The 0th node receives the final message
- Transmission time reduced by a factor of log n as compared to Master-Slave approach

Pseudo Code for Log n Level Approach

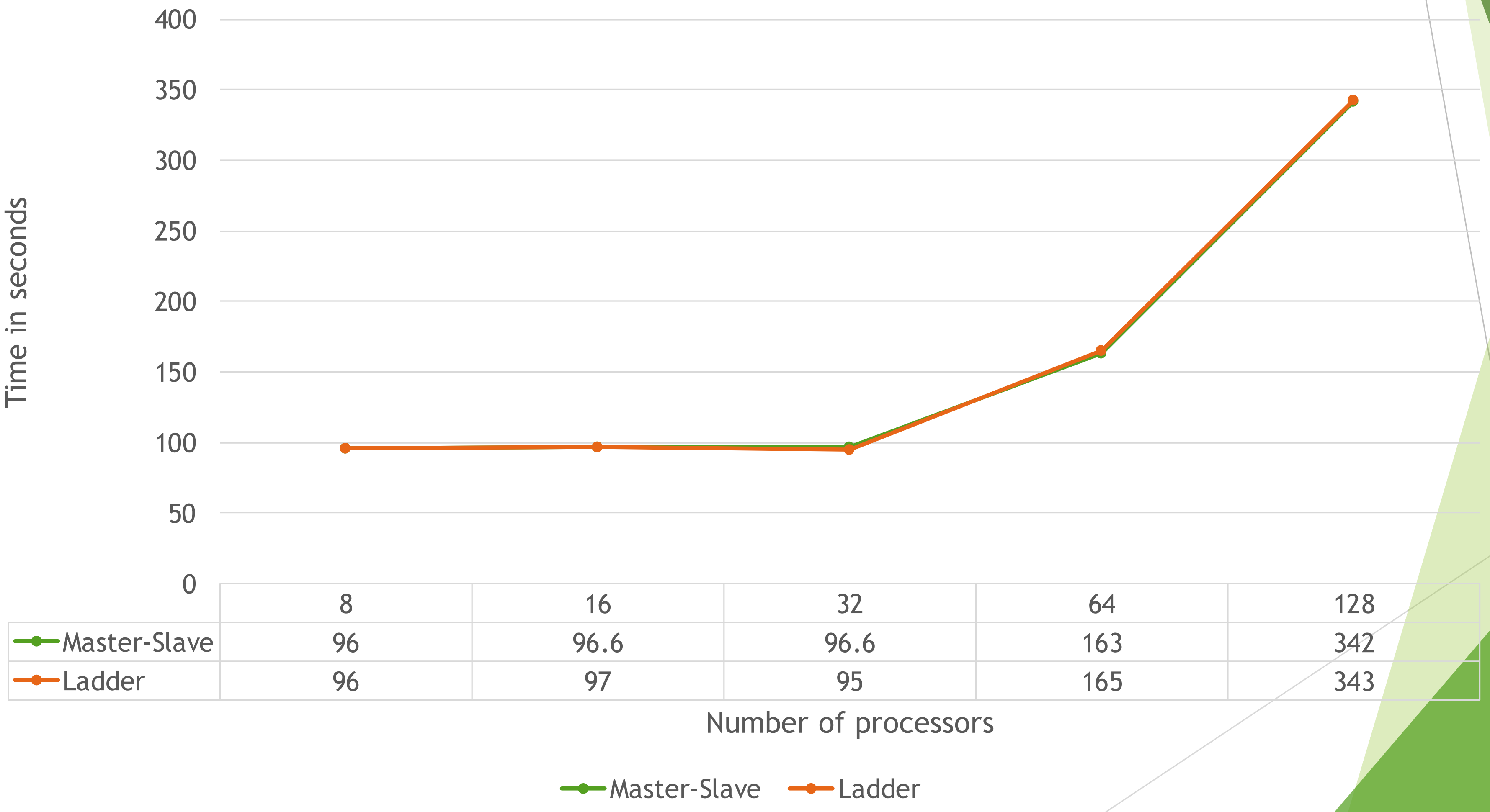
```
For (i= 1 to logn)
{
  MPI_Send ( // to the left processor);
  MPI_Recv( // from the left processor);
  // update the buffer
}
```

Maze Generation in Parallel with Ladder Approach



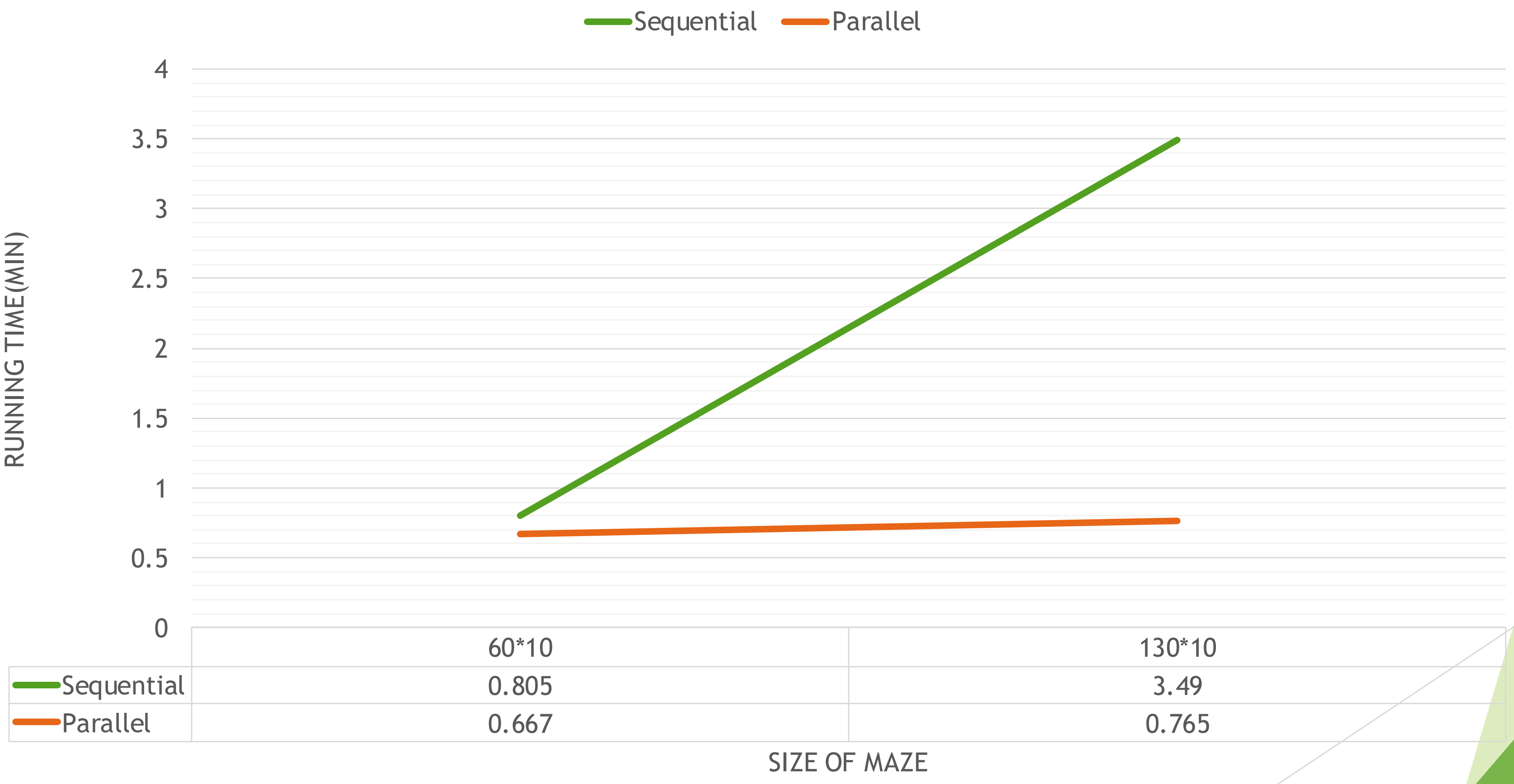
	8	16	32	64	128	256
10*10	9	9	9	9	13	31
20*20	41	40	41	41	74	146
30*30	96	97	95	165	343	697

Comparison of Maze Generation in two Approaches



	8	16	32	64	128
Master-Slave	96	96.6	96.6	163	342
Ladder	96	97	95	165	343

Maze Generation Sequential Vs Parallel

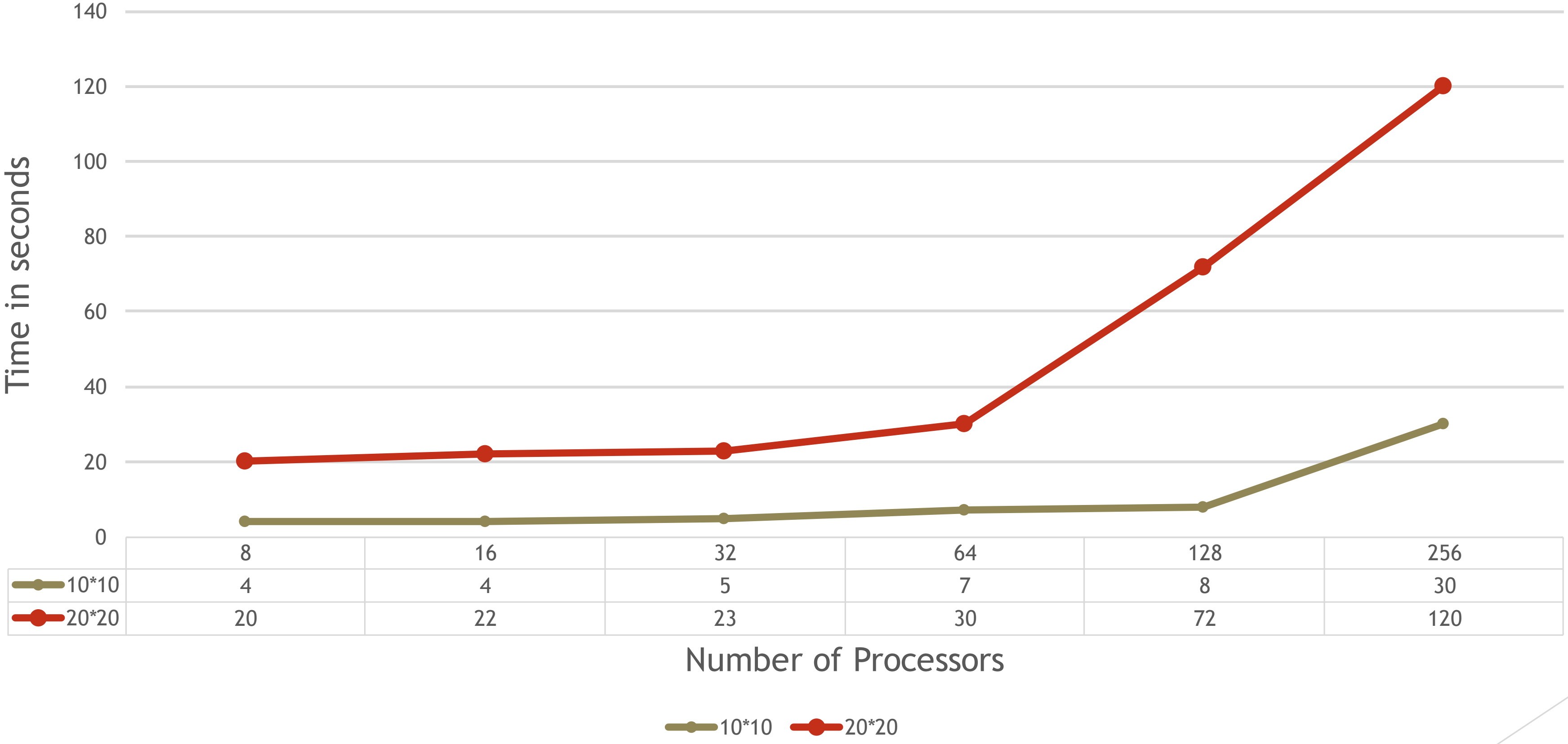


	60*10	130*10
🟩 Sequential	0.805	3.49
🟧 Parallel	0.667	0.765

Maze Solving in Parallel

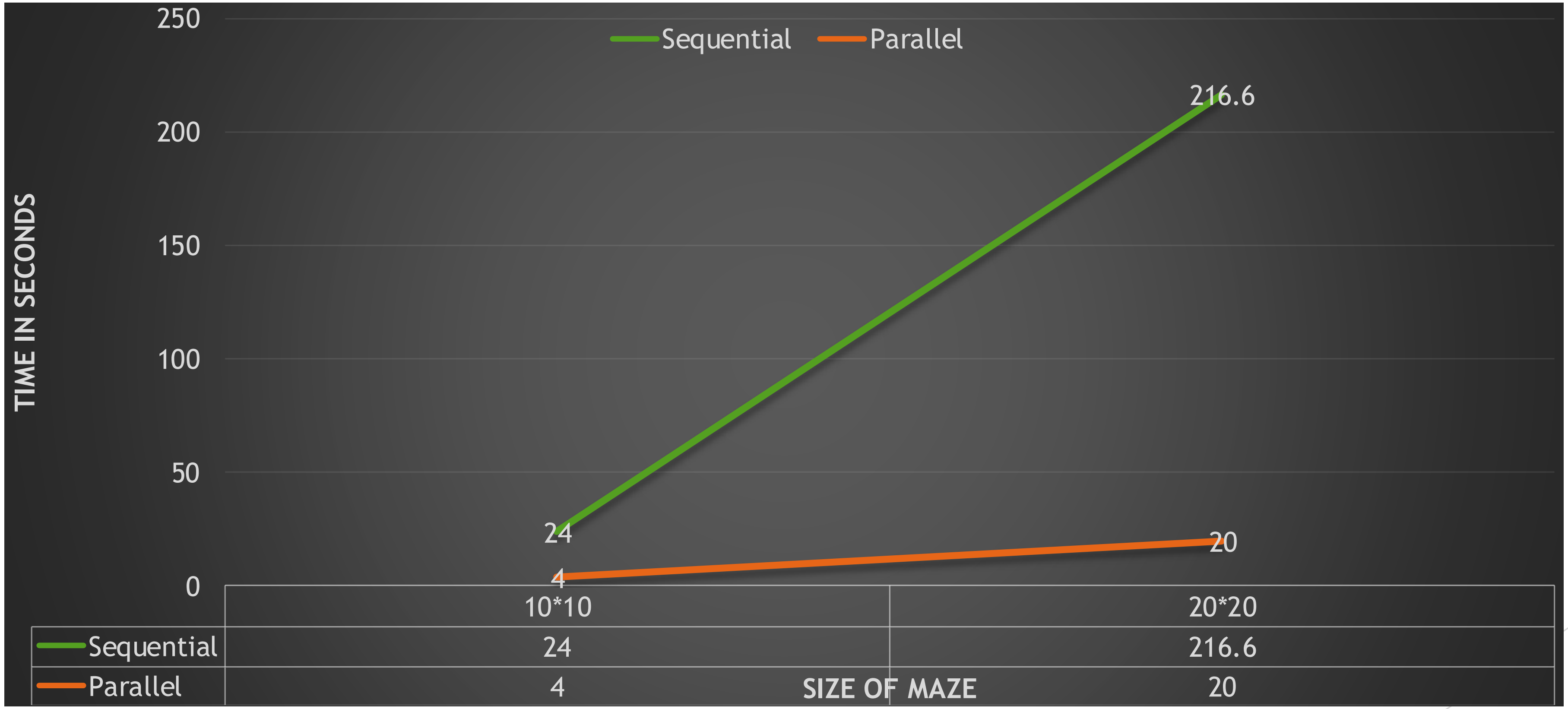
- Same approach as maze generation
- The whole maze is split into parts
- The source and destination are assumed to be in the 0th and Nth processor respectively
- The final point for a single part is the starting point for the consecutive part
- The path is sent as an array via the log levels approach

Maze Solving in Parallel



	8	16	32	64	128	256
10*10	4	4	5	7	8	30
20*20	20	22	23	30	72	120

Maze Solving Sequential v/s Parallel



	10*10	20*20
Sequential	24	216.6
Parallel	4	20

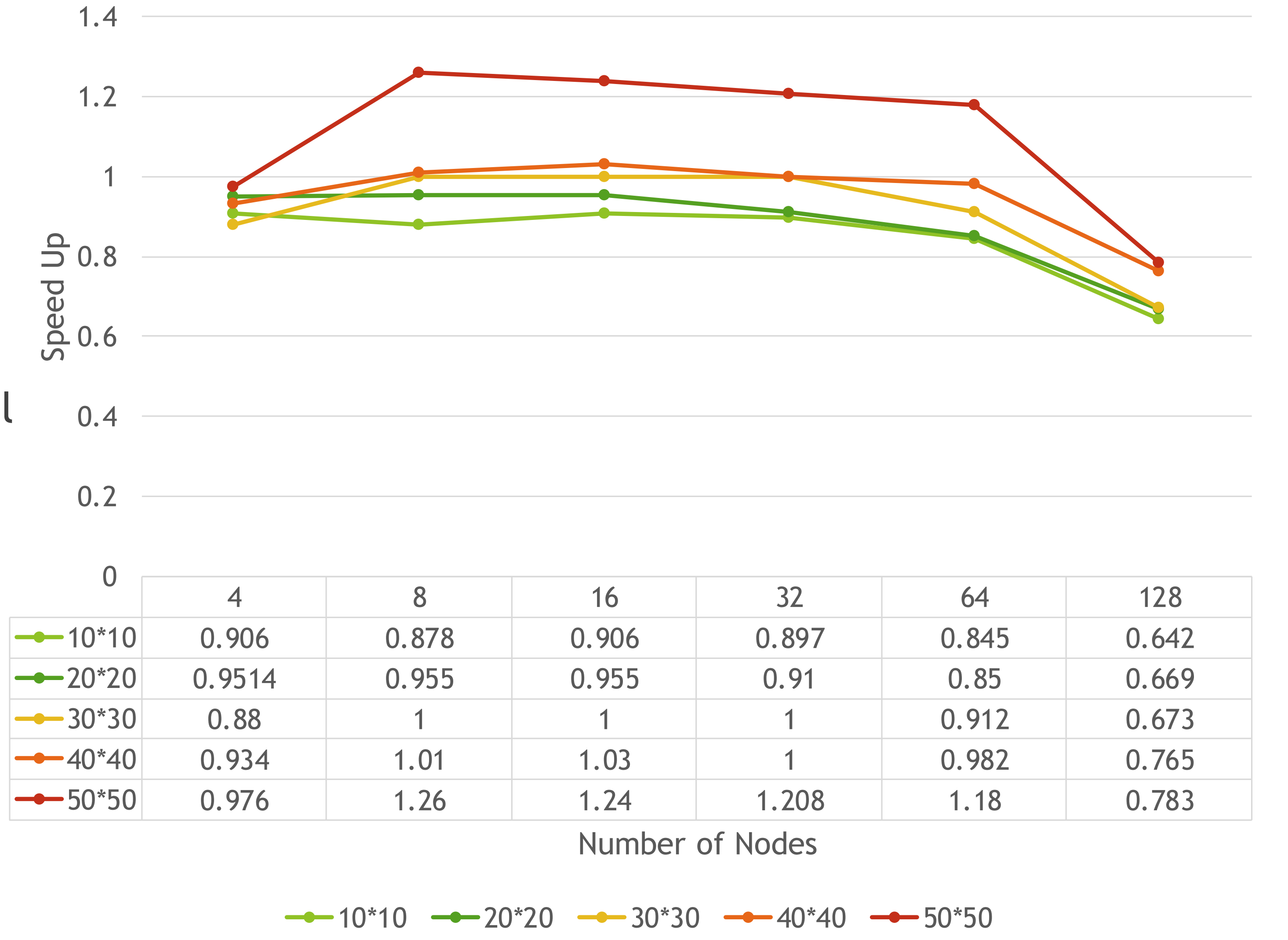
Speed-Up

Speed Up is defined as:

- S=Ts/Tp

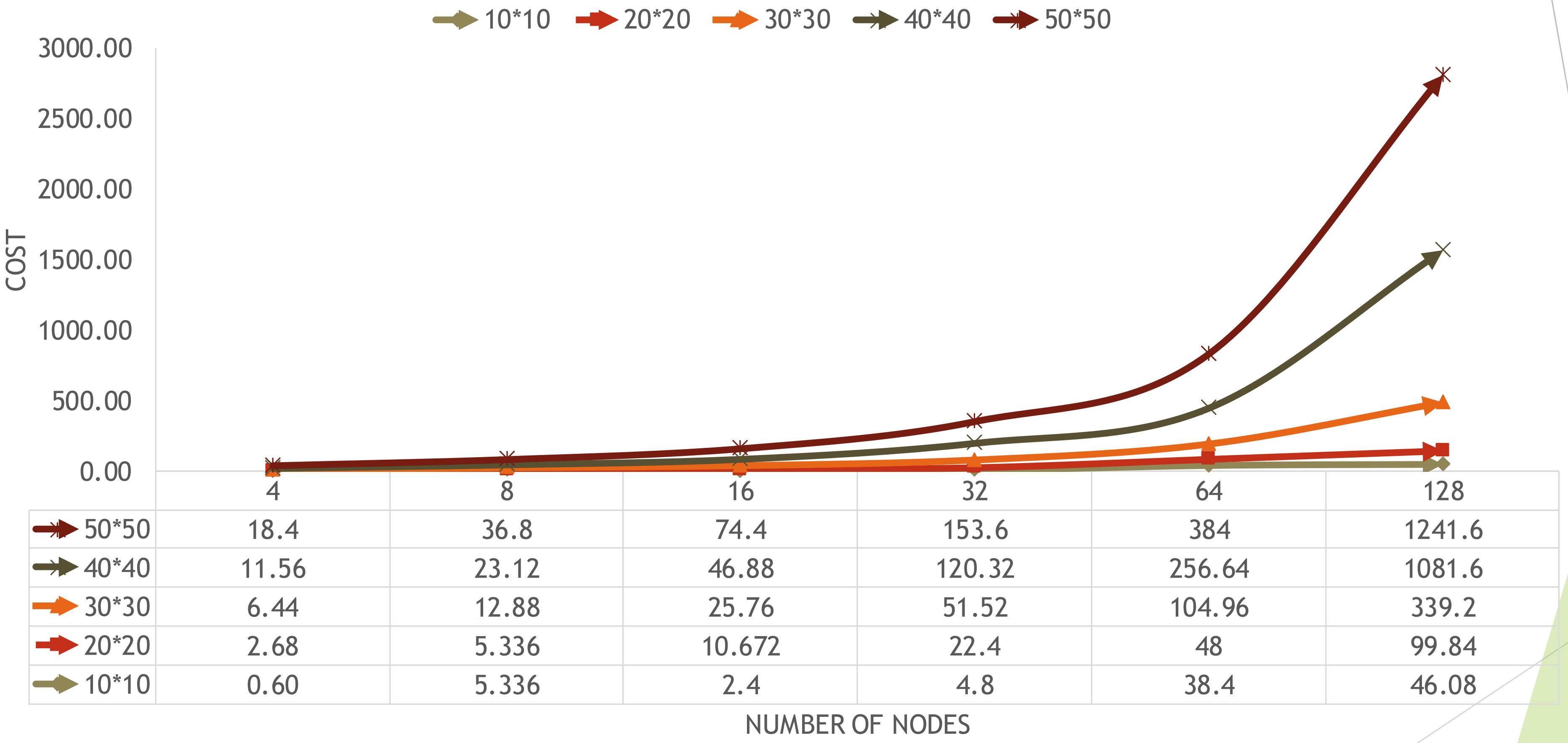
where:

- Ts is the time taken in sequential operation
- Tp is time taken in parallel operation
- P is the number of processors



	4	8	16	32	64	128
10*10	0.906	0.878	0.906	0.897	0.845	0.642
20*20	0.9514	0.955	0.955	0.91	0.85	0.669
30*30	0.88	1	1	1	0.912	0.673
40*40	0.934	1.01	1.03	1	0.982	0.765
50*50	0.976	1.26	1.24	1.208	1.18	0.783

Cost Analysis



	4	8	16	32	64	128
50*50	18.4	36.8	74.4	153.6	384	1241.6
40*40	11.56	23.12	46.88	120.32	256.64	1081.6
30*30	6.44	12.88	25.76	51.52	104.96	339.2
20*20	2.68	5.336	10.672	22.4	48	99.84
10*10	0.60	5.336	2.4	4.8	38.4	46.08

References:

- A New Parallel Algorithm for Minimum Spanning Tree Problem-Rohit Setia, Arun Nenunchezhian, Shankar Balachandran
- Introduction to Parallel Computing –Ananth Grama, George Karypis, Vipin Kumar, Anshul Gupta
- Algorithms Sequential and Parallel- Unified Approach –Russ Miller, Laurence Boxer
- http://profstewart.org/pm1/talks_09/MazeCreating.pdf

Thank You!!