Maze Solver

Rahul Ram & Surya Narayan

Link to Project Repo

April 14, 2024

Overview

Introduction

Problem Description

Given two points on a maze (start, end) we find the shortest path between them.

Applications of Maze Solver :

- Robotics
- Path-Planning

Maze Generation

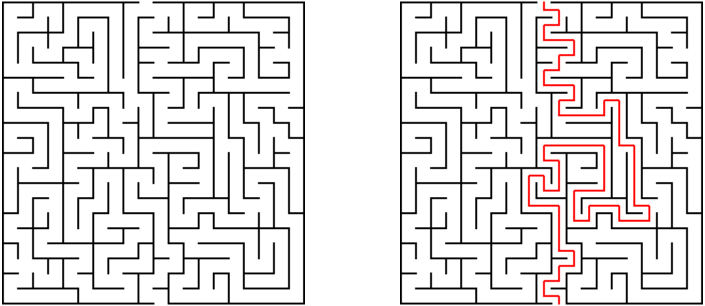


Figure: Generated Maze (on left), Solved Maze (on right)

Approach

1. Divide the graphs into subgraphs and stitch their solutions together.
2. Parallelize the search algorithm - BFS.
3. Apply mincut and perform search on two different graphs in parallel.

Our Implementation

We opted to implement the **second approach** - parallelizing the search algorithm. The limitations of the other approaches are discussed in the coming slides.

Parallelizing Maze Solving

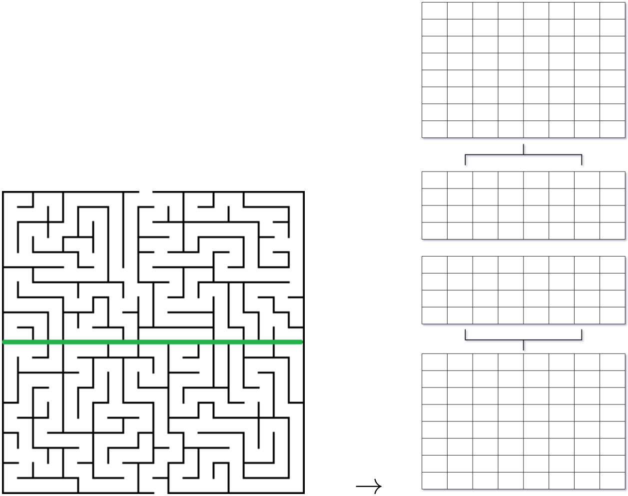
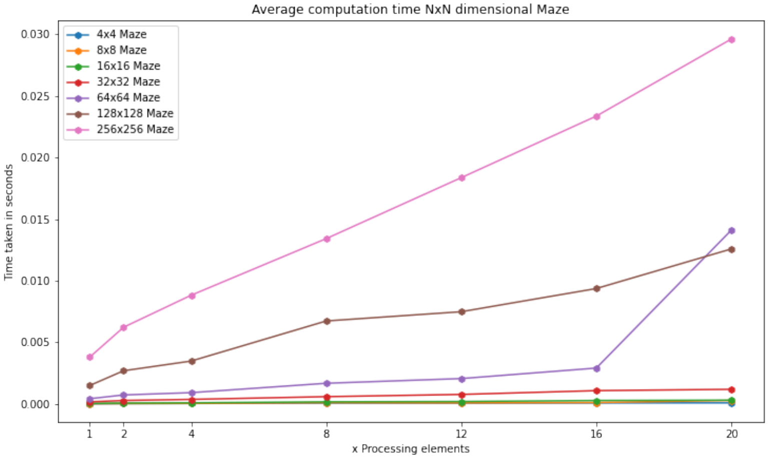


Figure: Generated Maze (on left), Solved Maze (on right)

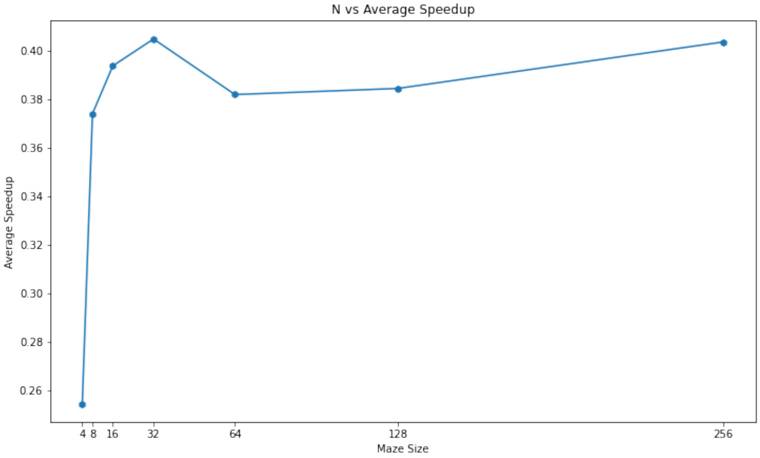
Drawbacks

- **Approach one and three**
  - Synchronizing solutions and re-running searches on them proved more time-consuming than in the 2nd approach.
- **Approach two**
  - In the Maze Graph, each traversal layer typically contains only 2 to 10 nodes.
  - Furthermore, the necessity for synchronization to update the queue introduced additional overhead.

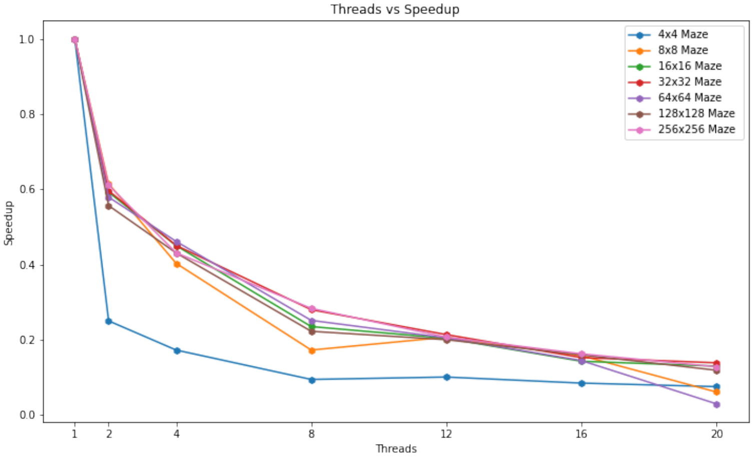
Performance Metrics



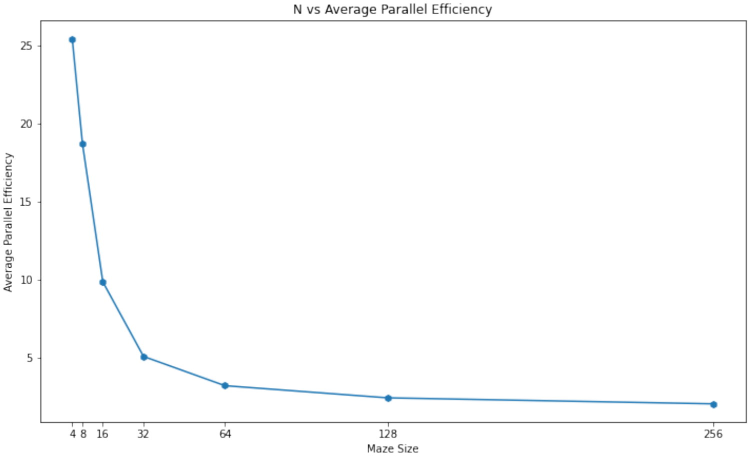
Performance Metrics



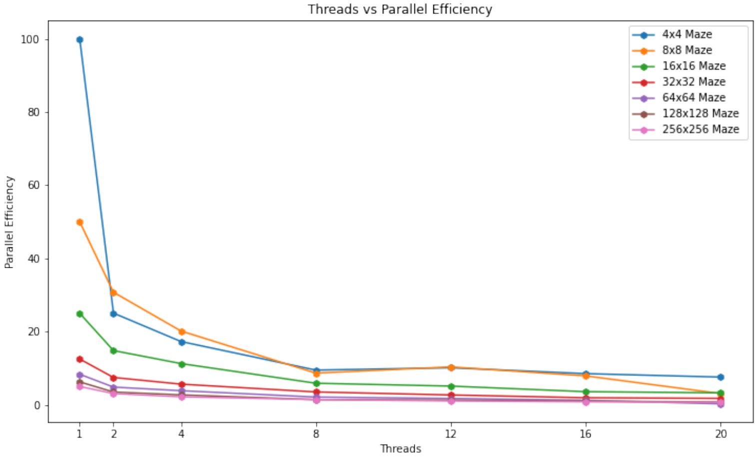
Performance Metrics



Performance Metrics



Performance Metrics



References

Samya Ahsan (2023)
Parallel Functional Programming Report: MazeSolver

A. Botea et al
Near Optimal Hierarchical Path-Finding
Journal of Game Development Vol. 1, pp.7-28, 2004