CS361 Programming Project: Searching

Fall 2024

Due: Dec 8th (Sunday) End of Day

Project Description

Searching is at the core of a lot of algorithms. The goal of this project is to implement 5 search algorithms on a given problem and compare their performances.

1 Implementation

You are allowed to work in groups of up to **4** students.

1.1 Problem statement:

Your task is to design a logistics system for a warehouse that needs to manage the movement of items. The warehouse is represented as a 2D grid where each cell can either be empty, contain an obstacle, or hold an item that needs to be collected. The goal is to design a program that can find the shortest path from a starting point to collect all items using various graph search algorithms. You will implement and compare the performance of different algorithms.

1.2 Requirements

1.2.1 Graph Representation:

- Represent the warehouse grid as a graph where:
  - Each cell corresponds to a node.
  - Each possible move corresponds to edges.

1.3 Algorithms to implement:

- *Bread-First Search(BFS)*
- *Depth-First Search (DFS)*
- *Dijkstra's Algorithm*

- *A* Search Algorithm*
- *Another algorithm of your choosing*

1.3.1 Programming Instructions:

- You may choose to use C, C++, Java, or Python. However, all algorithms must be implemented using the same programming language.
- You may **NOT** use any data structures from existing libraries except:
  1. Basic data structures such as arrays.
  2. A random number generator.
  3. System functions, such as print to screen, write to file, etc.
- Create a function that executes all algorithms on the given grid and provides output.

2 Performance Benchmarking

1. For each of the algorithms, you will benchmark:
   (a) Shortest Path Length: The total distance taken to collect all items
   (b) Path Taken: A representation of the coordinates visited in the process.
   (c) Execution Time: The time taken by each algorithm to complete the task. You should exclude the I/O time when benchmarking. In other words, if your program is reading in the input from a file, your timer should start after all the data is loaded into the main memory.
   (d) Memory Usage: The number of nodes explored during the search.
2. Your performance metrics should be tested with the following scenarios:
   (a) Case with no obstacles and several items.
   (b) Case with multiple obstacles and fewer items.
   (c) Case with a complex grid layout requiring different paths to optimize.
   (d) Case with only one item in a corner of the grid.
3. Compare the performance metrics for all 5 algorithms for each test scenario and present the comparison in a tabular way in the report.
4. **Extra credit** What happens if a robot performs the task, and after collecting all the items it returns to the starting point?

3 Project Report

You will need to submit a final report of your findings. The report must be typewritten in a scientific paper format with a minimum 12pt font size. The report should include the following key sections:

1. **Title, authors, and institutions of the authors**.
2. **Introduction**.
3. **Preliminaries and Methods**: You will introduce the key ideas and performance characteristics of the implemented algorithms, and describe the algorithms exactly as implemented.
4. **Performance analysis and discussions**: Here are some of the things you should discuss in your report:
   (a) Are the asymptotic running times of these algorithms close to their actual performance? What is the constant factor you have found in your running time analysis?
   (b) Do all 5 programs have the same coding complexity? What about debugging effort?
   (c) Which algorithm is the fastest or slowest? Why?
   (d) For every algorithm, what is the highest grid size that the algorithms can handle without running into memory issues in your machine?
5. **Conclusion**.
6. **Bibliography** : Reference your work.
7. **Contributions of Each Team Member**: Describe in detail the contribution of each member of your team. For example, member A implemented the BFS, member B implemented DFS, member C implemented Dijkstra's algorithm or any other, and member D is the lead in performing benchmarks and compiling the final report. This is just a sample division of labor. You will decide in your group what will be the division of tasks among team members.
8. **Acknowledgement**.
9. If you are attempting the extra credit you have to discuss in detail the implementation changes in code and performance changes in your algorithms.

4 Submission

You will be uploading the following on Canvas by midnight on Dec 8th (Sunday):

- Your final report in a single PDF file. Only one report is needed for a team.
- A single zipped file containing your source code, 4 testing scenarios with the dataset (can be .txt file), and a .txt file with instructions on how to run the code.

Note: There will not be an extension for the final project. Each team has almost 8 weeks to work on the project and there is no excuse for not completing it on time. Start now!