CS 6601 Assignment 2_ Search_Beta

CS 6601 Assignment 2: Search (Beta)

Due September 25th, 2016 by 11:59PM UTC-12 (Anywhere on Earth)

Abstract

You will implement several graph search algorithms with the goal of solving tridirectional search.

Motivation

Search is an integral part of AI. It helps in problem solving across a wide variety of domains where a solution isn't immediately clear.

Objectives

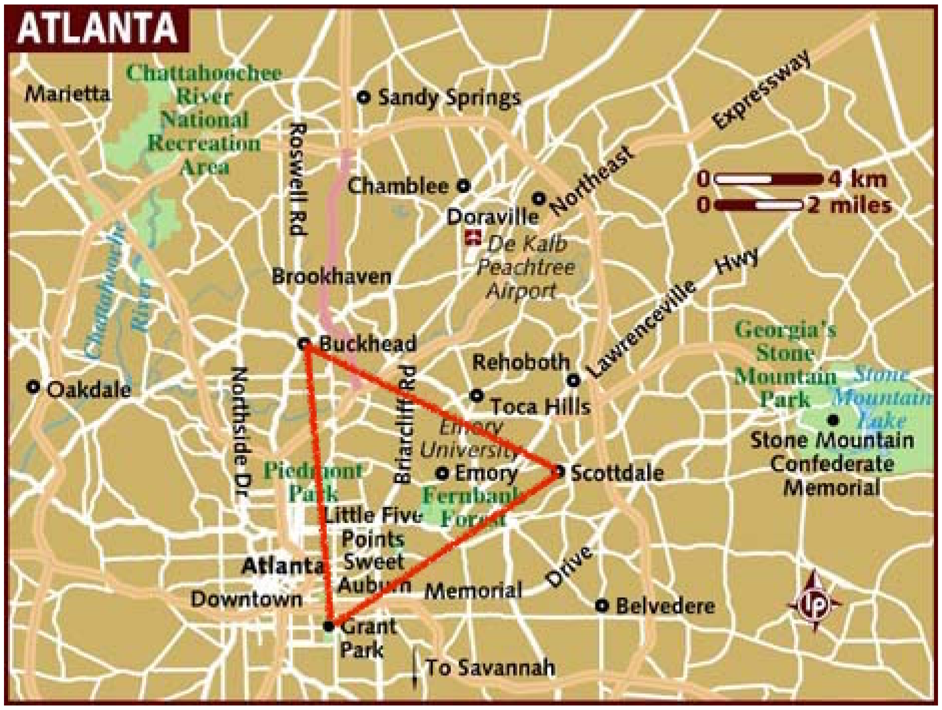
Students should be able to develop and intuition about the various tradeoffs involved in searching using different algorithms.
Students should also be able to pick an algorithm to use given a problem.

Evaluation

Evaluation is done using the student's last submission on Bonnie.

1. The Challenge

Graph search is an integral part of AI. You've seen how to find the shortest distance between two points; now, let's try three. With tridirectional search the goal is to find the shortest path between three points. The order of visiting doesn't matter. In this case, we'll be trying to connect three points in the city of Atlanta.



2. Your Assignment

Your task is to implement several informed search algorithms that will calculate a walking route between three points in Atlanta with a minimal time and space cost.

You will do this in search notebook.ipynb, and there are tests along the way. Your searches should be executed with minimal runtime and memory overhead.

We will provide the following classes and files:

File	Description
*.pickle	serialized graph files for Romania and ATL data
visualize_graph.py	a module to visualize search results in Gist
search_submission.py	Python file to submit to Bonnie

In the pickle files, we've provided you with the graph of the Romania map from Russell and Norvig (Chapter 3) and a graph of the OpenStreetMap of Atlanta for use in testing. You can also test against Bonnie. We'll be updating our test server during the Beta.

3. Grading (Tentative)

Each section of the assignment is associated with a number of points, as follows (out of 100 points total):

Warmup 1: Implement a priority queue to demonstrate its improvement in performance over an insert-and-sort queue. (5 points)
Warmup 2: Implement breadth-first search. (5 points)
Warmup 3: Implement uniform-cost search. (10 points)
Warmup 4: Implement A* search with a corresponding admissible heuristic.
(10 points)
Exercise 1: Implement bidirectional uniform cost search. (15 points)
Exercise 2: Implement bidirectional A* search. (20 points)
Exercise 3: Implement tridirectional uniform cost search. (20 points)
Exercise 4: Implement an improvement on tridirectional search. (15 points)

4. Due date

This assignment is due on T-Square September 25th by 11:59PM UTC-12 (Anywhere on Earth). The deliverables for the assignment are:

• A filled out version of the Python file provided. (search_submission.py)

5. Resources

If you want to know how the OpenStreetMap data works, check out their wiki.
The software requirements are listed in search_notebook.ipynb. You'll want to skim the networkx documentation before starting.

The GitHub repo also has some useful papers that might point you in the right direction.