ICS 171 Spring 1999
Project: Subway Path Planning

Rina Dechter

April 27, 1999

The Path Planning Problem

Consider the following underground subway system:

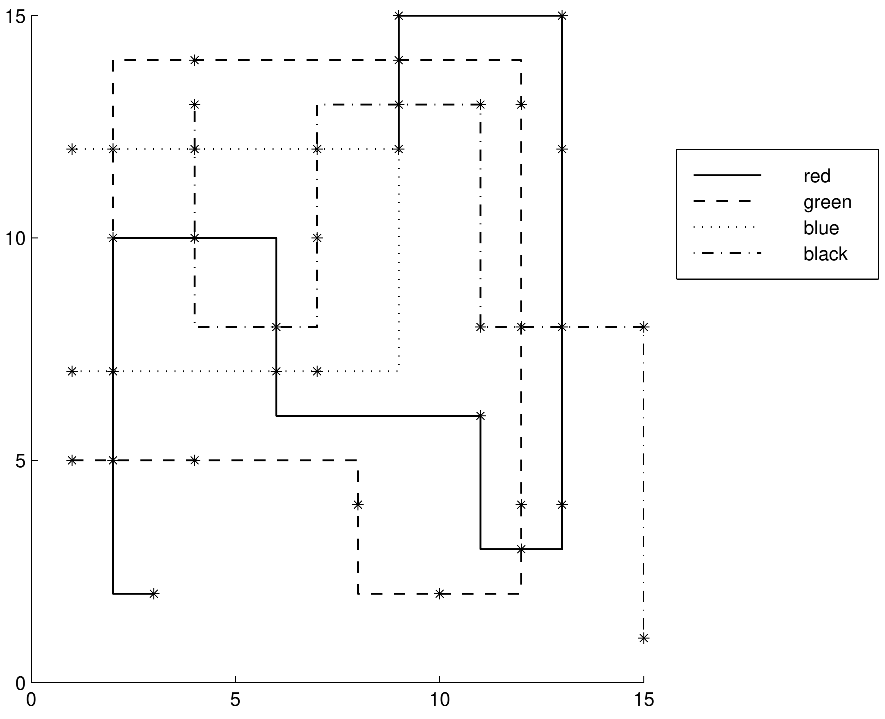

Figure 1: The Subway System

- There are four subway lines on horizontal and vertical line system.
- The lines are given by coordinates.

- The size of system is 15x15.

There are four subway lines:

1. Red line: (3,2) (2,2) (2,10) (6,10) (6,6) (11,6) (11,3) (13,3) (13,15) (9,15) (9,12)
2. Green line: (1,5) (8,5) (8,2) (12,2) (12,14) (2,14) (2,10)
3. Blue line: (1,7) (9,7) (9,12) (1,12)
4. Black line: (15,1) (15,8) (11,8) (11,13) (7,13) (7,8) (4,8) (4,13)

There are stations at the following points: (1,5) (1,7) (1,12) (2,5) (2,7) (2,10) (2,12) (3,2) (4,5) (4,10) (4,12) (4,13) (4,14) (6,7) (6,8) (7,7) (7,10) (7,12) (8,4) (9,12) (9,13) (9,14) (9,15) (10,2) (11,6) (11,8) (11,13) (12,3) (12,4) (12,8) (12,13) (13,4) (13,8) (13,12) (13,15) (15,1) (15,8). We will refer to these stations with numbers from 1 to 37 (i.e. (1,5) is station 1, (15,8) is station 37). Passengers may switch train lines at a station where two lines intersect. For example, at (2,5) passengers may get off the red line and then switch to a train on the green line.

There is a time involved in switching train lines which must be taken into account. The average time is equivalent to 3 units. One time unit is equivalent to the time a train takes to cover one (1) coordinate interval. For example, seven (7) time units is eqivalent to the time it takes to go from coordinate (15,1) to coordinate (15,8).

Write a program that instructs passengers in selecting a route.

Each passenger query is in the form *go source-station destination-station.* The answer will be the best route between the two. The output will be a path in the form: *(station, line taken, station...).* For example, (6 red 4 blue 2) means that from station 6 we will take the red line to station 4, we then switch to the blue line and go to station 2.

Part 0: (Due Tuesday May 6):
Devise a cost function that takes into account both the distance to be traveled and the transfer time when switching lines. Assume that all trains have the same speed.

Part 1 (Due Tuesday May 20):
Find an optimal route that minimize the cost function you developed with the following search algorithms:

1. Depth First search (with a depth limit that will guarantee finding a solution).
2. Iterative Deepening search.
3. Uniform Cost search.

Part 2 (Due Tuesday June 1st):

1. A^* algorithm.
2. modified A^* algorithm, $f(n) = (1 - w)g(n) + wh(n)$. Try $w = \{0, 3/4, 1\}$.
3. (extra-credit) Branch & Bound as discussed in lectures.

What to hand in

You will do this project in two parts. For each part you must implement the search algorithms listed above. You will also hand in a report containing the following:

- Source code of the program with clear comments.
- Examples demonstrating that your program works correctly.
- A comparison and analysis of the algorithms.
- Documentation describing where your program is located and how to run it.

For both parts 1 & 2 you must compare and contrast the search algorithms that you implemented. It is not enough to do a theorectical comparison. We want to see empirical results which compare algorithm properties such as the running time (or number of nodes expanded) and memory usage. Limit your discussion to 2 pages.

We require that you test your program on the following test cases and make up two test cases of your own:

1. go 6 21, that is from station (3,2) to station (11,6)
2. go 2 22, that is from station (1,12) to station (11,8)
3. go 31 10, that is from station (15,1) to station (4,13)

General Instructions

- You have one week to complete the preliminary part of the project (Part 0), two weeks to complete the first part and about one and a half week for the second part.
- You may discuss the problem with other groups but you must develop your program independently. Copying code from other teams or past projects in previous ICS 171 courses is prohibited.

- We require that you make your code available on your ICS UNIX account (so we can run the program). If you do not have an ICS account email `antenor@ics.uci.edu` immediately. Place your code in the directory `ics171` in your home directory. In this directory you must have your source code, executable, and program traces. Do not forget to set the security permissions so we can read and execute your program (double check this with your group partner – if we cannot run the program you will lose points).

- Your program should take as inputs the source station, destination station, and an option for which search algorithm to use. We suggest the following format: route_finder source_station destination_station search_algorithm. We want to be able to run your program without changes to your source code, so any input format within these requirements is fine.

- You may program in any language that is available on the ICS UNIX accounts including: C, C++, JAVA, LISP.

- You are allowed to do this project in groups of two students. Each group will submit one report. You may also work alone.

- Project descriptions will be discussed in sections; Feel free to contact your TA if you have any uncertainties.