Data Structure and Programming, Spring 2022 programming assignment #2 **Monotonic Routing**

Chin-Chia, Yang
r10942093@ntu.edu.tw

RELEASE DATE: 04/07/2022
DUE DATE: 04/21/2022

Late policy: $Score = Score \times 0.7^{(h/24)}$, $h =$ late hours ≤ 48

1 Problem Description

Given an $m \times n$ grid graph, the cost of each graph edge (u, v) is $d(u, v)$, a source grid is S, and a target grid is T (see Figure 1(a)). Let $D(S, g)$ denote the minimum cost of a non-detour path from S to a grid g. Please write a program that finds a monotonic path from the source grid to the target grid, i.e., compute $D(S, T)$, and reports the total edge cost, the number of grids, and the grid coordinates of the path. Notice that,

1. Please use python3 to run your program.
2. Monotonic means that you can only go up or right, and you **can't** go back (left or below).

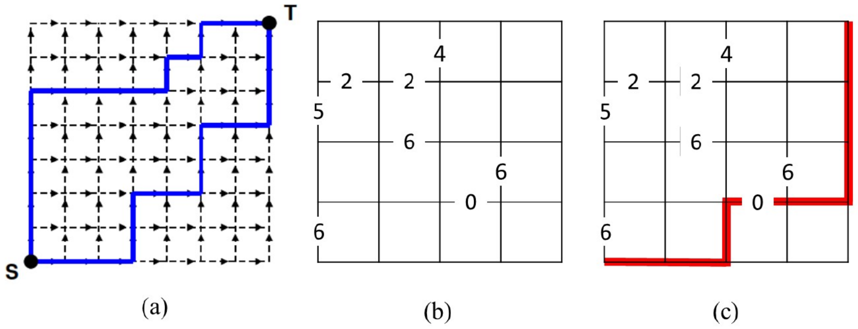

Figure 1: An example of monotonic routing.

2 Input/Output Specification

2.1 Input format

The file format for the monotonic routing is illustrated, with comments in italics (these will not be in actual input files). The 1st line gives the problem size in terms of the indices of the left-bottom and the right-top grids (the indices of the left-bottom grid will always be (0,0)). Each edge connecting two grids has either default cost or non-default cost. The default cost is given in the 2nd line. The 3rd line gives the number of edges that have non-default costs. Start from the 4th line, the cost difference compared to the default cost of each non-default edge is separately given. Finally, the coordinates of the source and the target grids are listed in the last two lines.
The input file format is as follows:

BoundaryIndex # # # # *//the indices of the left-bottom and the right-top grids*

DefaultCost # *//the default cost*

NumNonDefaultCost # *//the number of non-default edges*

x1 y1 x2 y2 # *// the cost difference of the non-default edge between (x1,y1) and (x2,y2)*

...

//repeat for the number of non-default edges

Source xS yS *//the coordinate of the source grid*

Target xT yT *//the coordinate of the target grid*

2.2 output format

The resulting monotonic path needs to be described in the output file. The 1st line gives the total edge cost in the resulting path, and the 2nd line gives the number of grids on the path. After that, the consecutive grids in the path from source to target have to be listed in order.
The output file format is as follows:

RoutingCost [total edge cost]

RoutingPath [# of grids, k]

[x_1] [y_1]

[x_2] [y_2]

...

[x_k] [y_k]

//repeat for the number of grids in the path

Notice: x1 and y1 must be the same as xS and yS, and xk and yk must be the same as xT and yT in the input file, respectively.

Here is an input/output example of Figures 1(b)/(c):

Sample Input	Sample Output
BoundaryIndex 0 0 4 4	RoutingCost 21
DefaultCost 3	RoutingPath 9
NumNonDefaultCost 8	0 0
3 1 3 2 3	1 0
2 3 2 4 1	2 0
2 1 3 1 -3	2 1
1 3 2 3 -1	3 1
1 2 2 2 3	4 1
0 3 1 3 -1	4 2
0 2 0 3 2	4 3
0 0 0 1 3	4 4
Source 0 0	
Target 4 4	

3 Command-line parameter

In order to test your program, please add the following command-line parameters to your program (e.g., python mono.py –input 5x5.in –output 5x5.out).

4 Submission

Please put mono.py and readme.txt into a directory named studentID and compress the directory into studentID.zip. Finally, upload studentID.zip to ceiba. The homework is due on 4/21, at 23:59.

5 Grading policy

This programming assignment will be graded based on (1) the correctness (2) solution quality, and (3) running time. Please check these items before your submission. We provide you three test data: 5x5.in, 50x50.in, 500x500.in, and we will score your program by two private data. If the output format, the number of grids on the path and the consecutive grids in the path from source to target are right, you will get 10% each private data. If the time of executing your algorithm is within the time limit, you will get 20% each private data. The remaining score depends on your solution quality. We will sort all the routing cost and give score from the least routing cost to the largest routing cost.

Notice: If the time of executing your algorithm exceeds the time limit, we'll stop the execution, and you will get 0% on this item. (Brute-Force method will definitely exceed the time limit.)