CS 415 Algorithm Analysis Fall 2021

Project 3

This project will be done in teams of two. Due: Nov 20, 2021.

The goal of this project is to use BFS and its improvement called Best-First Search (also known as A* when an additional condition holds) to find the shortest path from a starting position to a goal position in an unweighted graph. Both BFS and Best-First Search find the shortest path in the unweighted case. However, the number of nodes visited by A* is usually smaller than in the case of BFS. The graph in this project is an image in which each pixel (that satisfies some constraint) is a vertex and two pixels are considered adjacent if they are adjacent (i.e., if their pixels coordinates are (a, b) and (c, d) then a = c and b = d±1, or a = c±1 and b = d.) One constraint to define a vertex is that its RGB values are above a certain threshold, say (200, 200, 200). For example, in a maze each pixel with RGB values below 200 could represent an obstacle. In this project, any pixel with R > 100 or G > 100 or B > 100 will be considered a vertex. The starting vertex and the goal vertex will be specified by their pixel coordinates.

You are to implement two versions of a graph search algorithm. The first one the Breadth-First Search that we discussed in class and is described below:

```
Breadth-First-Search (Image: I, Vertices: s, t)
   // s is the start vertex, t is the destination
   Initialize a queue Q;
   Q.insert(s);
   Set visited[v] = false all vertices;
   visited[s] = true;
   set d[s] = 0 and d[u] = MaxInt for all other u
   while (Q is not empty and (not visited[t])):
       u = Q.delete( )
       for each neighbor v of u:
           if (not visited[v])
               visited[v] = true
               set color of v to Green
               d[v] = d[u] + 1
               prev[v] = u
               Q.insert(v)
     // end-for
     //end-while
    v = t
    while (v != s):
       set color of v to Red
       v = prev[v]
    create an output image with new pixel values
    output d[t]
```

The second algorithm that will be implemented is known as *Best-First-Search* and it uses a priority Queue instead of the Queue. Recall that the d values for a pixel p in the queue represent the shortest distance from the source vertex s to p. The A* or best-first search algorithm uses the value d[u] + h[u] for each pixel u in the queue where h[u] is an estimate of the distance from u to the goal node. The algorithm chooses the pixel u from the queue such that d[u]+h[u] is as small as possible. When the goal vertex leaves the queue, the algorithm terminates. It keeps the prev pointers similar to BFS. In your implementation, the estimate h[u] will be the shortest distance from u to goal t if there are no obstacles. i.e., if the row, column values of u and t are (p1,p2) and (t1, t2) then h[u] = |p1 – t1| + |p2 – t2|.

```
Best-First-Search (Image: I, Vertices: s, t):
   // s is the start vertex, t is the destination
   // h is a function as described above
   Initialize a queue Q;
   Q.insert(s);
   Set visited[v] = false all vertices;
   visited[s] = true;
   set d[s] = h[s] and d[u] = MaxInt for all other u.
   while (Q is not empty and (not visited[t])):
       u = Q.deleteMin( )
       for each neighbor v of u:
           if (not visited[v])
               visited[v] = true
               set color of v to Green
               d[v] = d[u] + 1
               prev[v] = u
               Q.insert(v, d[v] + h[v])
        // end-for
    //end-while
    v = t
    while (v != s):
       set color of v to Red
       v = prev[v]
    create an output image with new pixel values
    output d[t]
```

You are to create two images one for each algorithm. The output image will show the following modifications to the input: (a) the shortest path from s to t (pixels on this will turn red) and (b) all pixels visited by the algorithm will be shown in green.

What should be submitted? Your main function shall ask the name of the input image (a BMP file), the row and column numbers of the starting pixel s and the goal pixel t. It displays the length of the shortest path from s to t (as an integer) and asks the user to enter the names of two output files. It creates the output images generated by the Breadth-First-Search and the Best-First-Search algorithms (as BMP files) and terminates.