18CSC204J Design and Analysis of Algorithms Lab Assignment (CLA-P4)

Maze game is a well-known problem, where we are given a grid of 0's and 1's, 0's correspond to a place that can be traversed, and 1 corresponds to a place that cannot be traversed (i.e. a wall); the problem is to find a path from bottom left corner of grid to top right corner; immediate right, immediate left, immediate up and immediate down only are possible (no diagonal moves). We consider a variant of the maze problem where a cost is attached to visiting each location in the maze, and the problem is to find a path of least cost through the maze.

Solution

Consider a maze given by a cost c_{ij} such that c_{ij} takes a finite value for traversable locations and ∞ for walls (non-traversable locations), with n rows and m columns.

We convert the maze to a weighted graph G(V, E) as follows: each location (i, j) in the maze corresponds to a node in the graph v_{ij}, and $V = \cup\, v_{ij}$ and $E = \{ (u, v) \mid u, v \in V \}$. Each node v_{ij} is connected to other nodes with weights in the following manner:

$d(v_{ij}, v_{i,j+1}) = c_{i,j+1}$
$d(v_{ij}, v_{i+1,j}) = c_{i+1,j}$
$d(v_{ij}, v_{i,j-1}) = c_{i,j-1}$
$d(v_{ij}, v_{i-1,j}) = c_{i-1,j}$
$d(v_{ij}, v_{rs}) = \infty$ otherwise

This also means that the weight for an adjacent location in the maze (e.g. $d(v_{ij}, v_{i,j+1})$) takes the value ∞ if the destination ($v_{i,j+1}$) represents a wall. Now, consider a path $t_0, t_1, t_2, \ldots t_k$ in the maze, where $t_i \in Z_+ \times Z_+$ is a 2D tuple indicating the location (row, col) in the maze); this has corresponding nodes in G(V, E) (say) $u_0, u_1, u_2, \ldots u_k$. Then, the cost of the path $t_0, t_1, t_2, \ldots t_k$ in the maze is
c0,0+i=0k-1d(ui, ui+1)
where $c_{0,0}$ remains the same for all paths and may be ignored. So, the optimal path minimizes i=0k-1d(ui, ui+1) where $u_0 = v_{n,1}$ and $u_k = v_{1,m}$. This can be accomplished by computing the optimal path from $v_{n,1}$ to $v_{1,m}$ using Djikstra's algorithm on the graph G(V, E).

Hints: The problem can have multiple solutions. Students can use any design technique such as greedy method, backtracking, dynamic programming. Students can choose their own conditions, positive or negative costs for the graph.