CSE 551 Programming Assignment

April 12, 2021

Submission Instructions: Deadline is **11:59pm on 04/24**. Submit your file electronically, via *Canvas*. **We will be checking for plagiarism**.Late sub-missions will be penalized, therefore please ensure that you submit (*zip* file upload is completed) before the deadline. Additionally, you can download the submitted file to verify if the file was uploaded correctly.

The *zip* file should contain your *source code* along with a *report (PDF) containing the algorithm in pseudo-code format.* **Your source code can be any of the following: C, C++, Java, Python and/or Matlab.**. For evaluation, we will be testing with our own set of inputs.

PROBLEM: An $n \times n$ grid is an undirected graph consisting of n rows and n columns of vertices, as shown in the figure below. We denote the vertex in the i^{th} row and the j^{th} column by (i, j). All vertices in a grid have exactly four neighbors, except for the boundary vertices, which are the points (i, j) for which $i = 1, i = n, j = 1$, or $j = n$. Given $m \leq n^2$ starting points (x_1, y_1), (x_2, y_2), . . . (x_m, y_m) in the grid, the **Funny Problem** is to determine whether or not there are m *vertex-disjoint* paths from the starting points to any m different (distinct) points on the boundary. Two paths $P1$ and P_2 are said to be *vertex disjoint* if they don't have any common vertex present in both the paths. For example,the the Funny Problem has a solution in the grid shown in Figure (a) (the vertex disjoints paths are shown by grey lines), whereas the grid shown in Figure (b) does not.

This programming assignment will take as input: (i) an integer number n (for creating the $n \times n$ grid), and (ii) m grid points ($m \leq n^2$, specified by their row-column values $i \times j$). The output of the assignment will be (i) A *Yes/No* answer to whether a solution to the Funny Problem exists for the input problem instance, and (ii) If the answer to (i) is *Yes*, the vertex-disjoint paths that constitute the solution.

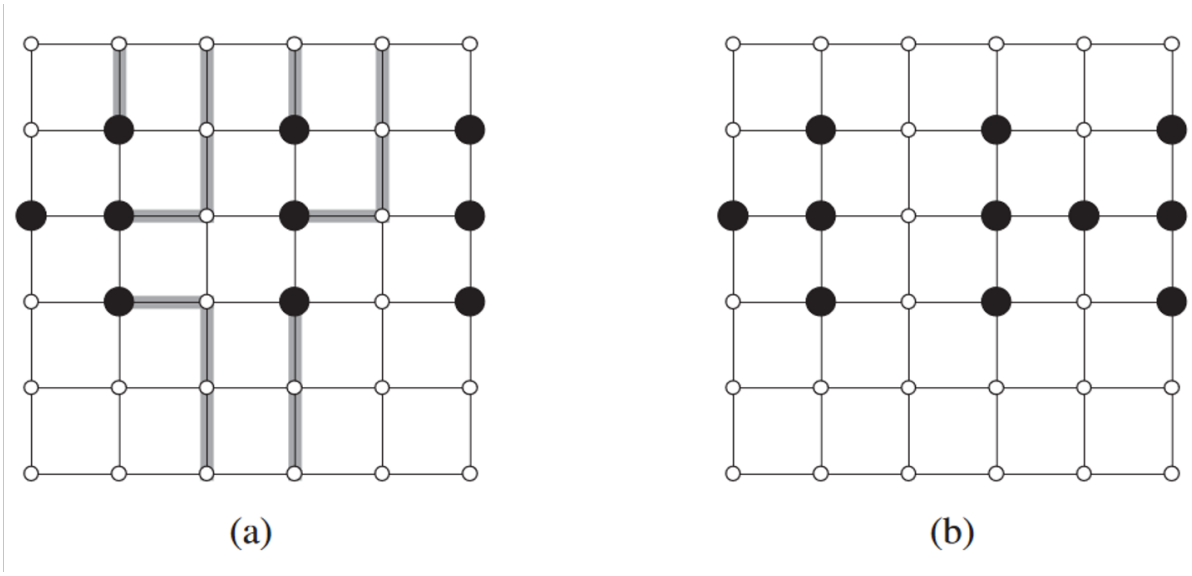

Figure 1: Grids for the escape problem. Starting points are black, and other grid vertices are white.