Weekplan: Introduction to Graphs

The 02105+02326 DTU Algorithms Team

Reading

Introduction to Algorithms, Cormen, Rivest, Leisersons and Stein (CLRS): Introduction to Part VI + Chapter 22.1-22.4 + Appendix B.4-B.5.

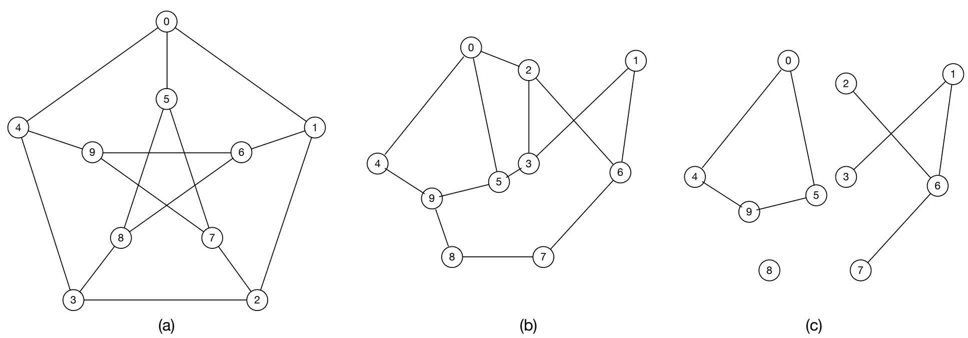


Figure 1: Graphs for the exercises. (a) is the *Petersen graph*.

Exercises

1 Representation, Properties and Algorithms Consider the graphs in Figure 1. Solve the following exercises.

1.1 [w] Show adjacency lists and adjacency matrices for (a) and (c).

1.2 [w] Simulate DFS on (a) starting in node 0. Assume the adjacency lists are sorted. Specify the DFS-tree, and discovery and finish times.

1.3 [w] Simulate BFS on (a) starting in node 0. Assume the adjacency lists are sorted. Specify the BFS-tree, and the distance for each node.

1.4 Specify the connected components of (a), (b), and (c).

1.5 Which of (a), (b), and (c) are bipartite?

2 Depth-First Search using a Stack Explain how to implement DFS without using recursion. *Hint:* use an (explicit) stack.

3 Find a Cycle Give an algorithm that determines if a graph is *cyclic*, ie. contains a cycle. How fast is your algorithm?

4 Number of Shortest Paths Give an algorithm that given two nodes s and t in G returns the *number* of shortest paths between s and t in G.

5 Mazes and Grid Graphs (exam 2010) A $k \times k$ *grid graph* is a graph where the vertices are arranged in k rows each containing k vertices. Only vertices that are adjacent in the horizontal or vertical direction may have an edge between them. See Figure 2(a). Solve the following exercises.

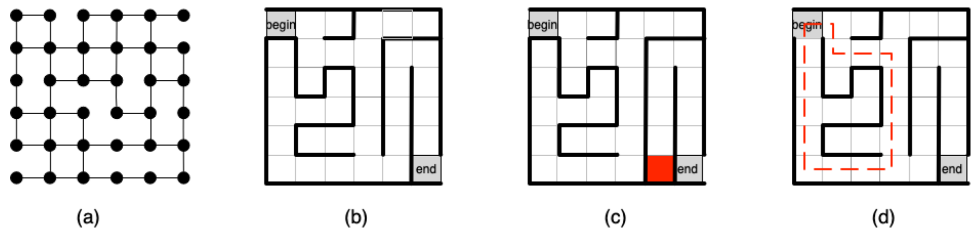


Figure 2: (a) a 6×6 grid graph. (b), (c), and (d) are 6×6 mazes. (b) is happy, (c) is unhappy since the end field cannot be reached, and (d) is unhappy since it contains a circular path.

5.1 Let the n and m denote the number of vertices and edges, respectively, in a $k \times k$ grid graph. Express n and m as a function of k in asymptotic notation.

A $k \times k$ *maze* is a square drawing consisting k^2 *fields* arranged in k rows each containing k fields. Each of the four sides of each field is either a *wall* or *empty*. A *walk* in a maze is a sequence of fields $f_1, \ldots, f_j$ such that any pair f, f' of consecutive fields in the sequence are adjacent in the horizontal or vertical direction and the shared side of f and f' is empty. A special field in the maze is designated as *begin* and another special field is designated as *end*. A maze is *happy* if the following conditions hold:

- There is exactly one unique walk in the maze from begin to end.
- There is a walk from start to any field in the maze.
- There are no circular walks, i.e., walks that start and end in the same field.

A maze that is not happy is *unhappy*. See Figure 2(b)-(d).

5.2 Explain how to model a $k \times k$ maze as a $k \times k$ grid graph.

5.3 Draw the maze in Figure2(b) as a grid graph.

5.4 Give an algorithm, that given a $k \times k$ maze modelled as a $k \times k$ grid graph, determines if the maze is happy. Argue the correctness of your algorithm and analyze it's running time as a function of k.

6 Implementation of Graphs We want to support the following operations on a dynamic graph G.

- ADDEDGE(u, v): add an edge between the nodes u and v.
- ADJACENT(u, v): return if u and v are adjacent in G.
- NEIGHBOURS(v): prints all neighbors of node v.

Solve the following exercises.

6.1 [†] Implement the operations on an adjacency matrix.

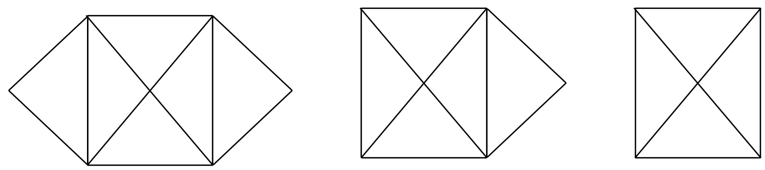
6.2 [†] Implement the operations on an adjacency list.

7 Euler Tours and Euler Paths Let G be a connected graph with n nodes and m edges. An *Euler tour* in G is a cycle that contains all edges in G exactly once. An *Euler path* in G is a path that contains all edges in G exactly once. Solve the following exercises.

7.1 [$*$] Show that G has an Euler tour if and only if all nodes have even degree.

7.2 [$*$] Show that G has an Euler path if and only if at most two nodes have have odd degree.

7.3 Which of the drawings below can you draw without lifting the pencil? Can you start and end at the same place?



7.4 Give an $O(n+m)$ time algorithm that determines if G has an Euler tour.

7.5 [∗] Give an $O(n+m)$ algorithm that finds an Euler tour in G if it exists.

8 Diameter of Trees Let T be a tree with n nodes. The *diameter* of T is the longest shortest path between a pair of nodes in T. Solve the following exercises.

8.1 Give algorithm to compute the diameter of T in $O(n^2)$ time.

8.2 [∗∗] Give algorithm to compute the diameter of T in $O(n)$ time.