

Homework 2 – Due Wednesday, September 12, 2007

Please refer to the general information handout for the full homework policy and options.

Reminders

- Your solutions are due before the lecture. Late homework will not be accepted.
- Collaboration is permitted, but you must write the solutions *by yourself without assistance*, and be ready to explain them orally to a member of the course staff if asked. You must also identify your collaborators. Getting solutions from outside sources such as the Web or students not enrolled in the class is strictly forbidden.
- To facilitate grading, please write down your solution to each problem on a separate sheet of paper. Make sure to include all identifying information and your collaborators on each sheet. Your solutions to different problems will be graded separately, possibly by different people, and returned to you independently of each other.
- *For all problems where you are asked to design an algorithm, do not forget to prove correctness and analyze your algorithms time and space complexity.*

Reading Review chapters 2.1-2.4, read chapter 3 in Kleinberg Tardos.

Exercises These should not be handed in, but the material they cover may appear on exams:

1. **(Graph representations)** Questions in this problem refer to the adjacency list and adjacency matrix representations defined on pages 87–89 of KT.

You are given a directed graph $G = (V, E)$ with $|V| = n$ vertices and $|E| = m$ edges. Let G^R be the graph obtained by reversing the directions of all the edges in G . For each of the following, give an efficient algorithm and analyze its running time.

If G is given in the adjacency list representation,

- (a) compute the out-degree of each vertex;
- (b) compute the in-degree of each vertex;
- (c) compute the adjacency list representation of G^R .

If G is given in the adjacency matrix representation,

- (d) compute the adjacency matrix of G^R .

2. Give two algorithms to detect whether a given undirected graph has a cycle. If the graph contains a cycle, your algorithms should output one (not all of them, just one). Base the first algorithm on BFS and the second, on DFS. The running time of your algorithms should be the same as the running times of BFS and DFS. Explain why your algorithms are correct and run in the required time.

Problems to be handed in

1. (**Holiday songs**) Chapter 2, problem 7.
2. (**Basic proof techniques**)
 - (a) (**Induction**) Chapter 3, problem 5.
 - (b) (**Contradiction**) Chapter 3, problem 7.
3. (**Number of shortest paths**) Chapter 3, problems 10. *Hint:* use BFS.
4. (**Singly connected**) A directed graph $G = (V, E)$ is **singly connected** if for all vertices u, v in V there is at most one simple path from u to v . (Recall that a path is simple if all vertices on the path are distinct.) Give an efficient algorithm to determine whether or not a directed graph is singly connected. *Hint:* Run DFS from each vertex.
- 5*. (**Classifying butterflies**) Chapter 3, problem 4.