

CSPP 55001 Algorithms — Autumn 2009

Homework 6 (assigned November 4, due November 11)

Reading: CLRS chapter 22.

Written assignment: Solve the following "Do" exercises and assigned problems. **Only solutions to the assigned problems should be turned in.**

Note: You are responsible for the material covered in **both** "Do" exercises and assigned problems.

Note: If you work with others, indicate their names at the top of your homework paper. Everyone must submit their own independently written solutions.

"Do" Exercises (*not to be handed in*):

- In an undirected graph, the *degree* $d(u)$ of a vertex u is the number of edges incident on u . In a directed graph, we distinguish between the *in-degree* $d_{\text{in}}(u)$, which is the number of edges into u , and the *out-degree* $d_{\text{out}}(u)$, the number of edges leaving u .
 - Prove that in an undirected graph the sum of the degrees of the vertices is equal to twice the number of edges.
 - Use part (1) to show that in an undirected graph, there must be an even number of vertices with odd degree.
 - Does a similar statement hold for the number of vertices with odd in-degree in a directed graph? Justify your answer.
- Exercise 22.1-6 on page 593.
2nd Edition, Exercise 22.1-6, page 530.
- Exercise 22.1-8 on page 593.
2nd Edition, Exercise 22.1-8, page 531.
- An undirected graph $G = (V, E)$ is called **2-colorable** if all vertices can be assigned color RED or BLUE and then all edges have one RED endpoint and one BLUE endpoint. Such graphs are also called **bipartite**, since the vertices can be partitioned into two disjoint sets $V = V_1 \cup V_2$ such that all edges go between the two sets V_1 and V_2 . Given an undirected graph $G = (V, E)$ in adjacency list representation, give an $O(V + E)$ -time algorithm which determines if G is 2-colorable. If G is 2-colorable, then your algorithm should construct a valid 2-coloring of G within $O(V + E)$ time. Describe your algorithm in pseudocode. Analyze the running time and argue that your algorithm is correct. Your algorithm should work if G is not connected.
- Exercise 22.2-8 on page 602.
2nd Edition, Exercise 22.2-7, page 539.

Problems (to be handed in):

- In each of the following problems, the input is a directed graph $G = (V, E)$ in adjacency list representation (singly-linked lists). The set of vertices is $V = \{1, \dots, n\}$. Give algorithms that solve each of the following problems in $O(V + E)$ time. Describe your algorithms in pseudocode. Explain in English the meaning of the variables you introduce. Comment the lines of your pseudocode, i.e., say briefly in English what is being done.
 - The **reverse** of a directed graph $G = (V, E)$ is a directed graph $G^R = (V, E^R)$ on the same vertex

set, but with edges reversed, i.e., $E^R = \{(i, j) : (j, i) \text{ in } E\}$.

Construct an adjacency list representation of the reverse of graph G . (10 points)

2. Construct a *monotone* adjacency list representation for G . A *monotone* list means that the neighbors of each vertex should be listed in increasing order. (10 points)
 3. Determine whether or not G is *undirected*. A directed graph is *undirected* if whenever an edge (i, j) belongs to E , its reverse (j, i) also belongs to E . (10 points)
2. Let $G = (V, E)$ be an undirected connected graph. Give an $O(V + E)$ -time algorithm to compute a path in G that traverses each edge of G exactly once in each direction. Argue that your algorithm is correct and analyze its running time. Describe your solution in simple pseudocode. (15 points)
 3. Often there are multiple shortest paths between two vertices in a graph. Give an $O(V + E)$ -time algorithm for the following task.

Input: Undirected graph $G = (V, E)$ in adjacency list representation; vertices u and v in V .

Output: The number of distinct shortest paths from u to v .

Describe your algorithm in pseudocode. Explain in English the meaning of the variables introduced in your algorithm. Argue that your algorithm is correct and analyze its running time. (15 points)

Note: Your algorithm should not list all the paths; just compute their number.
 4. An undirected graph $G = (V, E)$ is **connected** if for every pair of vertices u and v in V there is a path from u to v .

If an undirected graph is not connected, it will consist of several connected pieces that are called connected components of the graph. Formally, a **connected component** is a maximal (not expandable by inclusion of an extra vertex) connected subgraph of a given undirected graph.

Given an undirected graph $G = (V, E)$ in adjacency list representation, with $V = \{1, \dots, n\}$, give an $O(V + E)$ -time algorithm to determine the connected components of G . Describe your algorithm in pseudocode. Explain in English the meaning of the variables you introduce. Comment the lines of your pseudocode. Analyze your algorithm's running time. (15 points)
 5. Given an undirected graph $G = (V, E)$ in adjacency list representation, give an algorithm which determines in $O(V)$ time if G contains a cycle. Describe your algorithm in pseudocode. Explain in English the meaning of the variables you introduce and comment the lines of your pseudocode. Analyze the running time of your algorithm. Your algorithm should work if the input graph G is not connected. (20 points)

Gerry Brady

Thursday November 5 13:21:53 CDT 2009