Homework 2

CS 5114 (Fall 2018)

Assigned on Monday, September 10, 2018.
Submit a PDF file containing your solutions on Canvas by the beginning of class on September 19, 2018.

Instructions:

- The Graduate Honor Code applies this to homework. In particular, you are not allowed to consult any sources other than your textbook, the slides on the course web page, your own class notes, and the instructor. Do not use a search engine.
- Do not forget to typeset your solutions. *Every mathematical expression must be typeset as a mathematical expression, e.g., the square of n must appear as n^2 and not as "n^2".* You can use the LaTeX version of the homework problems to start entering your solutions.
- Describe your algorithms as clearly as possible. The style used in the book is fine, as long as your description is not ambiguous. Explain your algorithm in words. A step-wise description is fine. *However, if you submit detailed code or pseudo-code without an explanation, we will not grade your solutions.*
- Do not make any assumptions not stated in the problem. If you do make any assumptions, state them clearly, and explain why the assumption does not decrease the generality of your solution.
- Do not describe your algorithms only for a specific example you may have worked out.
- You must also provide a clear proof that your solution is correct (or a counter-example, where applicable). Type out all the statements you need to complete your proof. *You must convince us that you can write out the complete proof. You will lose points if you work out some details of the proof in your head but do not type them out in your solution.*
- Describe an analysis of your algorithm and state and prove the running time. You will only get partial credit if your analysis is not tight, i.e., if the bound you prove for your algorithm is not the best upper bound possible.
- In general for a graph problem, you may assume that the graph is stored in an adjacency list and that the input size is $m + n$, where n is the number of nodes and m is the number of edges in the graph. Therefore, a linear time graph algorithm will run in $O(m + n)$ time.

Problem 1 (25 points) This problem is in three parts.

1. (10 points) If G is an undirected graph and v is a node in G, let us use $G - \{v\}$ to denote the graph formed by deleting v and all the incident edges on v from G. In general, even if G is connected, it is possible that $G - \{v\}$ may contain two or more connected components, e.g., if G is a tree and v is a node with degree greater than one. However, this fragmentation cannot happen for every node v. Prove this fact, i.e., prove that in every connected, undirected graph G, there is at least one node v such that the graph $G - \{v\}$ is connected. *Hint:* In the special case that G is a tree, any leaf can serve as v. How will you find a "leaf" in a general undirected graph?
2. (10 points) Now prove that if G is an undirected, connected graph on n nodes that G must contain at least $n - 1$ edges. *Hint:* Use the statement in part (a) in combination with mathematical induction. Your proof must clearly state what you are inducting over and prove the base case, state the inductive hypothesis, and then prove the inductive step.
3. (5 points) Finally, prove that if G is an undirected, connected graph on n nodes that contains exactly $n - 1$ edges, then G does not contain a cycle. *Hint:* The statement you proved in part (b) may be helpful.

Problem 2 (10 points) For each node u in an undirected graph $G = (V, E)$, let $q(u)$ be the sum of the squares of the degrees of u's neighbors, i.e.,

$$q(u) = \sum_{(u,v)\in E} (d(v))^2,$$

where $d(v)$ denotes the degree of node v. In this definition of $q(u)$, think of u as being fixed. Then the sum runs over all nodes v such that (u, v) is an edge in the graph. In other words, the sum runs over all neighbours of u. Describe an algorithm that computes the $q(u)$ value for every node u in G in linear time in the size of the graph. To be clear, the algorithm should take linear time over all the nodes in G and not linear time for each node in G.

Problem 3 (15 points) Given an undirected graph G and an edge e in this graph, develop a linear time algorithm to determine if there is a cycle in G that contains e.

Problem 4 (20 points) Solve exercise 7 in Chapter 3 (page 108-109) of your textbook. I provide the essential part of the problem here.

Claim: Let G be an undirected graph on n nodes, where n is an even number. If every node of G has degree at least $n/2$, then G is connected.

Decide whether you think the claim is true or false, and give a proof of either the claim or its negation.

Problem 5 (30 points) Solve exercise 9 in Chapter 3 (page 110) of your textbook. Suppose that an n-node undirected graph G contains two nodes s and t such that the distance between s and t is strictly greater than $n/2$. Show that G must contain some node v, not equal to s or t, such that deleting v from the graph destroys all s-t paths. (In other words, the graph obtained from G by deleting v contains no path from s to t.) Give an algorithm with running time $O(m + n)$ to find such a node v. *Hint:* At least one of the layers in the BFS tree rooted at s has a special property.