CSE 21 Summer Academy 2016
Homework 5

Instructions

Homework should be done in groups of **one to three** people. You are free to change group members at any time throughout the quarter. Problems should be solved together, not divided up between partners. A **single representative** of your group should submit your work through Gradescope. Submissions must be received by 11:59pm on the due date, and there are no exceptions to this rule.

Homework solutions should be neatly written or typed and turned in through **Gradescope** by 11:59pm on the due date. No late homeworks will be accepted for any reason. You will be able to look at your scanned work before submitting it. Please ensure that your submission is legible (neatly written and not too faint) or your homework may not be graded.

Students should consult their textbook, class notes, lecture slides, instructors, TAs, and tutors when they need help with homework. Students should not look for answers to homework problems in other texts or sources, including the internet. Only post about graded homework questions on Piazza if you suspect a typo in the assignment, or if you don't understand what the question is asking you to do. Other questions are best addressed in office hours.

Your assignments in this class will be evaluated not only on the correctness of your answers, but on your ability to present your ideas clearly and logically. You should always explain how you arrived at your conclusions, using mathematically sound reasoning. Whether you use formal proof techniques or write a more informal argument for why something is true, your answers should always be well-supported. Your goal should be to convince the reader that your results and methods are sound.

For questions that require pseudocode, you can follow the same format as the textbook, or you can write pseudocode in your own style, as long as you specify what your notation means. For example, are you using "=" to mean assignment or to check equality? You are welcome to use any algorithm from class as a subroutine in your pseudocode. For example, if you want to sort list A using InsertionSort, you can call InsertionSort(A) instead of writing out the pseudocode for InsertionSort.

Required Reading Rosen 10.1, 10.2, 10.3, 10.4 through Theorem 1, 10.5 through Example 7.

Key Concepts Graphs (definitions, modeling problems using graphs), Hamiltonian tours, Eulerian tours, Fleury's algorithm, DAGs.

---

1. Let G be a DAG with vertices labeled $1, 2, \ldots, n$. Define the adjacency matrix and adjacency list representations of G as in Rosen, page 669.

   (a) (3 points) Let A be the adjacency matrix that represents G. Write pseudocode that takes A as an input and outputs the array `InDegree[]` for the graph G.

   Solution:
   procedure MatrixInDegreeArray(A, an Adjacency Matrix for G)

   ```
   1.  for $j := 1$ to $n$
   2.     Indegree[$j$]=0
   3.     for $i := 1$ to $n$
   4.         Indegree[$j$]=Indegree[$j$]+$A[i,j]$
   5.  return Indegree[]
   ```

   Here, $A[i, j]$ means the entry in row i and column j of the adjacency matrix A. This algorithm sums the columns of the adjacency matrix to produce the array `InDegree[]`.

   (b) (2 points) Let A be the adjacency matrix that represents G. Write a high-level English description of an algorithm that takes as inputs A and a source vertex v and outputs the adjacency matrix that represents $G - v$.

   Solution: Remove the column and row associated to the source vertex v and reduce the matrix in size, so now it has dimensions $n - 1$ by $n - 1$.

Say we have a graph where the vertices represent functions and there is an edge from f_1 to f_2 if and only if $f_1 \in O(f_2)$. When is this graph a DAG?

We claim the graph is a DAG if and only if there are no two distinct functions in the graph f_1, f_2 with $f_1 \in \Theta(f_2)$. In one direction, we show that if the graph is a DAG, there are no two such functions by proving the contrapositive: if there are two such functions, the graph is not a DAG. Assume there are two such functions $f_1 \in \Theta(f_2)$. Then since $f_1 \in O(f_2)$ and $f_2 \in O(f_1)$, there are edges from the vertex represnting f_1 to that representing f_2 and back, which is a cycle. Therefore G is not a DAG.

In the other direction, we show that if there are no such functions, then the graph is a DAG, again by proving the contra-positive: If the graph is not a DAG, then there are two functions that are Θ of each other. If the graph is not a DAG, then there is a cycle, $v_1 \leftarrow v_2 \leftarrow v_3 ... \leftarrow v_k \leftarrow v_1$, with associated functions $f_1 ... f_k$. We prove by induction on i that $f_1 \in O(f_i)$ for $i = 1..k$. Any function is in O of itself, so the base case is true. Say that $f_1 \in O(f_i)$. Since $f_i \in O(f_{i+1})$, and O is transitive, $f_1 \in O(f_i)$. So by induction the claim holds for each i. In particular, $f_1 \in O(f_k)$, and $f_k \in O(f_1)$ since there is an edge from v_k to v_1. Thus, $f_1 \in \Theta(f_k)$, so there is such a pair of functions.

2. You are planning on taking a road trip, and you have a list of cities that you might want to visit on this trip: city 1, city 2, ..., city n. You make an undirected graph where the vertices are cities on your list, and you connect two cities with an edge if you are willing to drive between them in one day. Let A be the adjacency matrix for this graph, and let a_{ij} represent the entry in row i and column j of A. Let A^k be the adjacency matrix A raised to the k^{th} power, and let $a_{ij}^{(k)}$ represent the entry in row i and column j of A^k.

   (a) (1 point) What does it mean in terms of your road trip if $a_{21} = 0$?

   (b) (2 points) What does it mean in terms of your road trip if $a_{23}a_{31} = 0$?

   (c) (2 points) What does it mean in terms of your road trip if $a_{21}^{(2)} = 0$?

   (d) (2 points) What does it mean in terms of your road trip if $a_{ij}^{(k)} = 0$?

(e) (2 points) For r a nonnegative integer, what does it mean in terms of your road trip if $a_{ij}^{(k)} = r$?

(f) (2 points) For r a nonnegative integer, what does it mean in terms of your road trip if $a_{ij} + a_{ij}^{(2)} + \cdots + a_{ij}^{(k-1)} + a_{ij}^{(k)} = r$?

3. A daily flight schedule is a list of all the flights taking place that day. In a daily flight schedule, each flight F_i has an origin airport OA_i, a destination airport DA_i, a departure time d_i and an arrival time $a_i > d_i$. This is an example of a daily flight schedule for February 10, 2016, listing flights as $F_i = (OA_i, DA_i, d_i, a_i)$:
(This isn't exactly the question asked this quarter, but is similar enough to give the idea.) $F_1 =$
(BWI, MIA, 6:00am, 8:00am)

$F_2 =$ (BWI, JFK, 8:00am, 9:00am)

$F_3 =$ (MIA, ATL, 8:30am, 10:00am)

$F_4 =$ (JFK, ATL, 10:00am, 12:30pm)

$F_5 =$ (ATL, BWI, 12:00pm, 2:00pm)

$F_6 =$ (MIA, JFK, 2:30pm, 5:00pm)

(a) (4 points) Describe how to construct a DAG so that paths in the DAG represent possible sequences of connecting flights a person could take. What are the vertices, and when are two vertices connected with an edge?

(b) (2 points) Why is your graph always a DAG?

(c) (2 points) Draw the DAG you described for the given example of February 10, 2016.

(d) (2 points) Use your DAG to help you determine the maximum number of connecting flights a person could take on February 10, 2016.

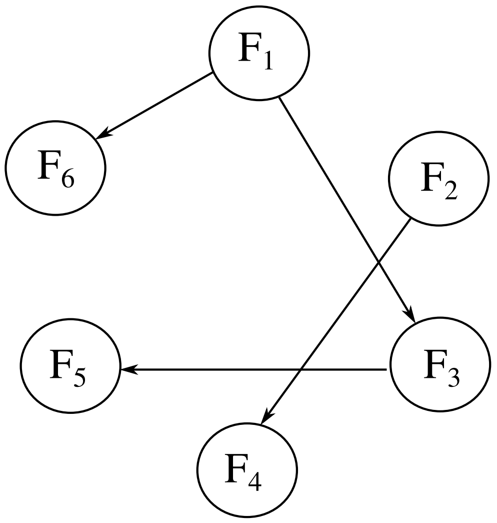
Solutions:

(a) The vertices are the flights in the daily flight schedule. Draw a directed edge from flight F_i to flight F_j if

- $DA_i = OA_j$ (flight F_j leaves from where flight F_i lands), and
- $a_i < d_j$ (flight F_j leaves after flight F_i lands).

(b) Our graph is always a DAG because it would be impossible to take the same flight more than once on February 10, 2016. This is clear because once you have taken a flight, some time has elapsed, and you cannot go back in time to take the same flight again.

(c)



(d) Since paths in the DAG represent possible sequences of connecting flights, the maximum number of connecting flights a person could take is determined by the longest path in the graph. Using the graph shown above, we see that the longest path includes three flights (F_1, F_3, F_5), so the maximum number of connecting flights a person could take on February 10, 2016 is three.

4. Prove that any tournament has a Hamiltonian path.

   We prove this by induction on n , the number of vertices in the tournament. The smallest tournament has $n = 2$, so to prove the base case, consider any tournament with two vertices, u and v. If there is an edge from u to v, (u, v) is a Hamiltonian path. Otherwise, by the definition of tournament, there is an edge from v to u, and v, u is such a path.

   Assume the claim is true whenever $n = k$, and consider a tournament on $k + 1$ vertices. Let v be a vertex, and consider the tournament on the remaining k vertices other than v. By the induction hypothesis, there is a Hamilitonian path in this sub-tournament, which passes through all the other vertices in some order, $v_1 \leftarrow v_2 \leftarrow ... \leftarrow v_k$.

   If there are no edges from v to any v_i, then there must be an edge from v_k to v. Then $v_1...v_k, v$ is a Hamiltonian path covering all $k + 1$ vertices.

   Otherwise, let i be the smallest number so that there is an edge from v to v_i. If $i = 1$, there is an edge from v to v_1, and $v, v_1...v_k$ is a Hamiltonian path. Otherwise, there is an edge from v to v_i, but no edge from v to v_{i-1}. By the definition of tournament, then, there is an edge from v_{i-1} to v_i. So then $v_1..v_{i-1}vv_i...v_k$ is a Hamiltonian path for the entire tournament.

   So in all cases, there is a Hamiltonian path for the tournament of size $k+1$. Thus, by induction, there is a Hamiltonian path for any tournament of any size n.