ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

School of Computer and Communication Sciences

Midterm **Graph Theory Applications**
Date: April 11, 2013 Spring 2013

The midterm exam is *closed* book, but you can use one A4 piece of paper where you can note anything you want. The exam lasts from 4:15pm till 6pm. DON'T PANIC. It is not necessarily expected that you solve all problems. Solve those problems first that you find easiest and then gradually move to the harder ones. Be concise! The solution for each of the four problems can be written down in a matter of some lines (not pages). We will subtract points for any material in your answer is not directly related to the solution.

Problem 1 (20pts)**.** Let G be a k-regular bi-partite graph on the vertex set $V = X \cup Y$, $|X| = |Y|$, and edge set E. Let A be the adjacency matrix corresponding to G. Show that A has an eigenvalue of k as well as an eigenvalue of $-k$. What are the two eigenvectors which correspond to these two eigenvalues?

Solution: Note that A has block-structure, i.e., it consists of 2×2 blocks each of size $|X| \times |X|$. The two blocks along the diagonal are zero and the other two are symmetric.

Let v be the all-one vector. Then $Av = kv$. This means, the eigenvector corresponding to the eigenvalue k is the all-one vector.

Next, consider the vector u, where the first $|X|$ components are $+1$ and the second $|X|$ components are -1. Again, an explicit check shows that $Au = (-k)u$.

Problem 2 (20pts)**.** Given a simple connected graph $G = (V, E)$ with edge costs w_e for each $e \in E$ (assume all edge costs are distinct), prove the following two fundamental properties that are used in all Minimum Spanning Tree (MST) algorithms.

a) (Cut Property) For any proper subset $S \subset V$ of nodes in G, let $e = (u, v)$ be the edge with minimum weight such that $u \in S$ and $v \in V \setminus S$. Show that every MST must contain e.

b) (Cycle Property) Let C be a cycle in G. Let $e = (u, v)$ be the edge with maximum weight in C. Show that e is not in any MST of G.

Solution:
Cut Property: To see the first claim, let $S \subset V$ be subset and let e be the edge with the given property. Assume that e is not already contained in the MST. We will show that this leads to a contradiction.

Add this edge to the spanning tree, hence creating a unique cycle. We can now drop from this cycle exactly the second edge which connects S to the outside, creating again a spanning tree. If e has, as given by assumption, strictly smaller weight, then the newly created spanning tree has strictly smaller weight, leading to the promised contradiction.

Cycle Property: Again we proceed by contradiction. Assume a MST does contain this edge. Then remove this edge from the spanning tree, so that we now get two components, let the vertices of these two components be called S and U and note that $S \cup U = V$. Note that the edge e has one end in S and the other end in U. Let C be the cycle containing edge e. We claim that then C must contain at least one more edge, call it e' which has one end in S and one end in U. By definition the edge e' has smaller weight. Hence by adding it to the the two components we get a new spanning tree, but this tree has strictly smaller weight.

Problem 3 (20pts)**.** Let G be a bipartite graph, with bipartition (X, Y) with no isolated vertices. Suppose that for every edge (x, y) with one end $x \in X$ and another end $y \in Y$, we have $\deg(x) \geq \deg(y)$. Prove that G has a matching that covers X.

Solution: Let us verify Hall's condition. Let $S \subset X$ and let $N(S)$ be the corresponding set of neighbors of S. Let E be the set of edges emanating from S. Order the nodes in S by increasing degree and let the degrees be $d_1 \leq d_2 \leq d_3 \leq \cdots \leq d_{|S|}$.

Start with the empty set and add the elements of S according to this ordering one by one. Let N_i, $1 \leq i \leq |S|$, denote the set of neighbors of the subset after i elements have been added. Order the neighbors of these subsets in the order they are added to and let D_j, $1 \leq j \leq |N_i|$.

We claim that at "time" i, $1 \leq i \leq |S|$, the following holds:

(i) $|N_i| \geq i$,

(ii) All D_j are upper bounded by d_i and $D_j \leq d_j$, $1 \leq j \leq i$.

Both conditions are trivially verified after the first step. After each additional step they follow by induction. (i) follows from (ii) since if the degrees of the neighbors are no larger than the degrees of the X subset, it means that we need to have at least as many neighbors as there are elements in the X subset in order to place all the edges. In a similar manner (ii) follows from (i): New neighbors which are added at a point fulfill this condition by assumption. On the other hand, neighbors which were already added fulfill the constraint since by definition the d_i sequence is increasing and no D_j degree at any point is larger than the largest d_i degree by construction.

Problem 4 (20pts)**.** [Who is afraid of directed graphs?] A *catenym* is a pair of words separated by a period such that the last letter of the first word is the same as the first letter of the second. For example, the following are catenyms: dog.gopher, gopher.rat, rat.tiger, etc. A *compound catenym* is a sequence of three or more words separated by periods such that each adjacent pair of words forms a catenym. For example, dog.gopher.rat.tiger is a compound catenym. Now, you are given a dictionary of words in English as input. Can you give an efficient procedure to determine if there is a compound catenym that uses each word in the dictionary exactly once?

Solution: Let G be the graph with $V = \{a, b, c, ..., z\}$ and edge set corresponding to the words in the dictonary. The graph is directed and eg. the word "tiger" corresponds to an edge which starts at the vertex "t" and ends at the vertex "r". The question of finding a single compound catenym then is equivalent of finding an Eulerian tour.

We have seen in class a simple condition for undirected graphs. In particular, for a graph to have an Eulerian cycle, all nodes had to have even degree. For the graph to have an Eulerian tour, all except exactly two, had to have even degree, and the two nodes of odd degree were the start and ending of the tour, respectively.

For directed graphs we quickly mentioned the equivalent procedure. Now we have *indegree* and *outdegree* of a node. The equivalent conditions are then the following. If we want an Eulerian cycle then for each node the indegree has to be equal to the outdegree. If we only want an Eulerian tour then the same condition has to hold except for exactly two nodes. For one of these two nodes the indegree has to be one larger and for the other one the indegree has to be one smaller than the outdegree.

Explicitly this means the following. For every letter of the alphabet except exactly two the number of words start and the number of words ending with this letter must be equal. For one letter there must be exactly one more words starting with this letter than ending and for the second special letter there must be exactly one more word ending with this letter than starting with this letter.