Christine Chung
47-835 Graph Theory
Homework 2 **Due:** Wednesday, Sept 13, 2006

1. **Problem 2.2.14** (page 93) Let $f(r,s)$ be the number of trees with vertex set $[n]$ that have partite sets of sizes r and s (with $r+s=n$). Prove that $f(r,s) = \binom{r+s}{s}s^{r-1}r^{s-1}$ if $r \neq s$. What is the formula when $r=s$? (Hint: First show that the Prüfer sequence for such a tree will have $r-1$ of its terms from the partite set s and $s-1$ of its terms from the partite set of size r.)

   Let a be the Prüfer sequence for any tree T with vertex set $[n]$ and partite sets R and S where $|R| = r$, $|S| = s$ and $r+s=n$. After following the procedure for constructing the Prüfer sequence, one edge $e = xy$ will remain. That means x must be in one set while y is in the other. Without loss of generality, assume $x \in R$ and $y \in S$. Since R started out with r vertices and in the end only vertex x remains, $r-1$ vertices were removed from R in the course of constructing the Prüfer sequence. Each time one of the $r-1$ vertices was removed from R, its neighbor, which must have been a vertex from S since T is bipartite, was added to the sequence a. That means $r-1$ entries in a refer to nodes in S. By analogous reasoning, $s-1$ entries in a refer to nodes in R. For each of the $r-1$ entries in a that refer to a vertex in S, there were s possible values to choose from (since the neighbor of a leaf in R could be any one of the s vertices in S). That gives us a total of s^{r-1} different possible settings for those $r-1$ entries. The same goes for the $s-1$ entries in a that each refer to a vertex in R to get r^{s-1} possible settings. So for a given R and S, there are $s^{r-1}r^{s-1}$ possible Prüfer sequences. There are also $\binom{r+s}{s}$ possible ways to choose a set of size s from the set $[n = r+s]$, leaving a set of size r behind, in effect partitioning $[n]$ into the two sets R and S. So to include all possible partitionings of $[n]$ into sets of size r and s, we have $\binom{r+s}{s}s^{r-1}r^{s-1}$ trees. ■

   When $r=s$, the formula is $f(r,s) = \binom{n}{n/2}(n/2)^{n/2-2}$.

2. Find a graceful labeling of C_7 and of C_8.

   For C_7, $m=7$. Label the 7 vertices (starting from any vertex then proceeding around the cycle) in the order $(0,7,1,6,2,4,3)$. This gives differences on the edges of 7, 6, 5, 4, 2, 1, and 3, which is precisely the set [7].

   For C_8, $m=8$. Label the 8 vertices in the order $(0,8,1,7,2,6,4,3)$. This gives edge differences 8, 7, 6, 5, 4, 2, 1, and 3, which is precisely the set [8].

3. **Problem 2.3.8** (page 104) Let G be a weighted connected graph. Prove that no matter how ties are broken in choosing the next edge for Kruskal's Algorithm, the list of weights of a minimum-weight spanning tree (in nondecreasing order) is unique.

   Suppose by way of contradiction that there are two different tie-breaking orders o and o' for which running Kruskal's on G produces two different list of weights, W and W' and thus two different trees, T and T'. Consider the first value at which the lists W and W' differ. Let the value at this point in W be called w and in W' be called w'. Without loss of generality, assume w is smaller than w'. That means Kruskal's algorithm using o added an edge of weight w to T without creating a cycle, which means it was not the heaviest edge in a cycle of G. But every edge that is not the heaviest edge in a cycle of G will be added to T' at some point (by definition of Kruskal's algorithm). If there are $k > 1$ edges in a cycle that are all equivalent to the heaviest edges in the cycle, all but one of them will be added, i.e. $k-1$ of them must be added in any execution of Kruskal's on G regardless of tie-breaking order. So there could be no such differing values w and w'.

4. **Problem 2.3.11** (page 104) For a spanning tree T in a weighted graph, let $m(T)$ denote the maximum among the weights of the edges in T. Let x denote the minimum of $m(T)$ over all spanning trees of a weighted graph G. Prove that if T is a spanning tree in G with minimum total weight, then $m(T) = x$. Construct an example to show that the converse is false.

   Assume by way of contradiction that $m(T) \neq x$. Let ST be the set of all spanning trees in G. Since $x \leq m(S)$ for all $S \in ST$, $m(T)$ must be greater than x. Let $e = uv$ be the max edge in T whose weight is greater than x. If $m(T) > x$, then there exists some T' such that $x = m(T') < m(T)$. That means $e \notin T'$. By proposition 2.1.6 in the text, there is an edge $e' \in T'$ that is not in T such that $T - e + e'$ is a spanning tree of G. We know this edge e' was cheaper than e since $m(T') < m(T) = w(e)$. Therefore, the weight of T has been reduced. Which is impossible if T was a minimum weight spanning tree.

5. **Problem 2.3.19** (page 105) Prove that the following algorithm correctly finds the diameter of a tree. First, run BFS from an arbitrary vertex w to find a vertex u at maximum distance from w. Next, run BFS from u to reach a vertex v at maximum distance from u. Report diam $T = d(u, v)$.

   We begin by noting that the diameter of a tree is simply the longest path between any two vertices in the tree (since there is a unique path between any two vertices in a tree).

   Running BFS from any vertex w of a tree T to find a vertex u at maximum distance from w implies that u is a leaf vertex. (For if u was not a leaf vertex, then it would have a child or parent node that would be further from w than u.)

   If w is already an endpoint of a longest path in T, then we are done because the path returned by BFS from w to u is a longest path in T, and the second run of BFS from u will return $v = w$ or some other vertex that is as far from u as w was. So by definition of diamter, $d(u, v)$ would be the diameter of T.

   The interesting case is when w is not already an endpoint of a longest path in T. We already established that u must be a leaf vertex, but in addition, u must also be one endpoint of a longest path in T. To see why, suppose for sake of contradiction that u was not an endpoint of a longest path in T. Then there must be a longest path with two other endpoints, call them s and t.

   Case 1: The path from w to u intersects the path from s to t. Call the vertex where they intersect x. Then either $d(x, s) > d(x, u)$ or $d(x, t) > d(x, u)$. (For if not, then $d(s, t)$ could not be the length of a longest path in T since $d(s, u)$ and $d(t, u)$ would both be greater.) Without loss of generality, assume it is $d(x, s)$ that is greater than $d(x, u)$. Then the furthest point from w would not be u, it would be s, which contradicts the definition of u that was given.

Case 2. The path from w to u does not intersect the path from s to t. (For a visual, imagine that the two paths are either contained in two different subtrees of T or they run parallel to each other through T.) Define $r(v_1, v_2)$ for $v_1, v_2 \in V(T)$ to be the vertex on the path from v_1 to v_2 that is closest to the root of T. Either $d(r(s,t), s) > d(r(w,u), u)$ or $d(r(s,t), t) > d(r(w,u), u)$. (For if not, then $d(s,t)$ wouldn't be the length of a longest path in T since the path from s (or t) $\rightsquigarrow r(s,t) \rightsquigarrow r(w,u) \rightsquigarrow u$ would be longer than the path from s to t.) Without loss of generality, assume that $d(r(s,t), s) > d(r(w,u), u)$. Then the path from $w \rightsquigarrow r(w,u) \rightsquigarrow r(s,t) \rightsquigarrow s$ would be longer than $w \rightsquigarrow u$. But this contradicts the fact that u is defined as one of the furthest nodes from w.

Now that we haven proven u must be an endpoint of a longest path in T, it immediately follows from the fact that BFS from u finds a vertex v at maximum distance from u that $u \rightsquigarrow v$ is a longest path in T and therefore $d(u,v)$ is the diameter of T.

6. True or False?

(a) Let T be a tree with a graceful labeling L. Let T' be a tree obtained from T by adding a new node v with an edge joining v to a node $u \in V(T)$. Define the labeling L' of $V(T')$ as $L'(w) = L(w)$ if $w \in V(T)$ and $L'(v) = L(u) + |E(T')|$. Then L' is a graceful labeling of T'.

False. $L(u)$ may be greater than 0, which would make $L'(v)$ greater than $m = |E(T')|$, which means it is not a graceful labeling since all labels have to be in the set $[m]$.

(b) Let G_1 and G_2 be two simple connected graphs with n nodes. If G_1 has more edges than G_2, then G_1 has more spanning trees than G_2.

False (assuming we are counting distinct spanning trees and not isomorphism classes of spanning trees). Let G_2 be the graph C_n for some very large n. Then G_2 has n spanning trees. Then construct G_1 as follows: build a tree on n vertices with two extra edges added that each form a 3-cycle somewhere in the tree (so G_1 is a very large tree-like graph with two 3-cycles in it somewhere). G_1 has $n+1$ edges but only nine spanning trees.

(c) Let T be a spanning tree of a simple graph G and let $e \in E(G) - E(T)$. Then there exists at least two distinct edges $e_1, e_2 \in E(T)$ such that $(T \cup e) - e_i$ is a spanning tree of G, for $i = 1, 2$.

True. When e is added to T it creates a cycle and all cycles in a simple graph have at least 3 edges. Therefore either of the other two edges in the cycle can be removed and what remains will still be connected and acyclic, i.e. a spanning tree.

(d) If e is an edge with minimum weight in a graph G, then e is in every minimum weight spanning tree of G.

False. If $G = C_n$ and all the edges have the same weight, then $G - e$ is a spanning tree of minimum weight of G that doesn't include e.

(e) If a simple graph has exactly two cycles, then it has at least four distinct spanning trees.

True (assuming we are counting distinct spanning trees, not distinct isomorphism classes of them). The smallest possible simple graph with two cycles (it looks like C_4 but with a diagonal drawn in) has 8 distinct spanning trees. Although it has only 3 isomorphism classes of spanning trees so if that is what you mean then the answer is False.