6.856
Problem Set 13(Final)
Dec. 2nd, 2002

1. (a) The loops make it aperiodic, and so we merely need show irreducibility(strong connectedness) to get unique stationary distribution. We need to show that from a pair (T_1, r_1), we can get to another pair (T_2, r_2). Note first that for a given spanning tree T, we can always get from (T, r_1) to (T, r_2) simply by moving the root along the edges of the tree. So it remains to show that we can get from one spanning tree to another. Note that since all spanning trees have the same number of edges, the set of edges that belong to T_1 and not T_2 must be equal in size to the set that belongs to T_2 and not T_1, s.t. the union of these two sets U is even in size. Pick any edge e in $T_2 - T_1$. This creates a cycle in $T_1 \cup e$, hence there must be some e' on the cycle that is also in $T_1 - T_2$. We walk the root to an endpoint of e and move the root over e to add e, and remove the edge after e in the direction of root movement, as dictated by the rooted spanning tree transition rule. But now we can continue doing this around the cycle until we add the edge before e', and delete e'. Now, we have added e to and removed e' from T_1, and reduced the difference set U in size by 2. We continue until $|U| = 0$, at which point we have changed T_1 into T_2, and thus this MC is irreducible.

   (b) Other than the self-transition, there are exactly d rooted spanning trees that can transition to a given rooted spanning tree. To see this, consider the edge incident on r that must be dropped from the spanning tree when we transition to r as root. If it is an edge in the given spanning tree, then this corresponds to the rooted tree with the same spanning tree but rooted at a neighbor of r. If the edge is not in the given spanning tree, then it determines the rooted spanning tree that transitions to the given one, since it determines the neighbor v of r that was root, and designates that the edge (v,r) is not in this spanning tree(but will be in the given one). Thus, each such edge corresponds to exactly one distinct rooted spanning tree, and so there are exactly d+1 of them that can transition to a given one. Since each transition occurs with probability $\frac{1}{d+1}$, the columns of this transition matrix also sum to 1 and so we have a doubly stochastic matrix for which the uniform distribution is stationary. Since there are n possible roots per spanning tree, and the stationary distribution is uniform over rooted spanning trees, it is also uniform over spanning trees.

   (c) We will show that if r is the last vertex to replace v as root, then the edge (v,r) is in the final spanning tree, and hence such a sequence of identities suffices to determine the spanning tree of n-1 edges. There are only two ways in which (v,r) could possibly be removed: if the root transitions to v or if the root transitions to r. The former cannot happen since v is never root again. In the latter case, without transitioning to v, (v,r) will never be on the root to leaf path of r, and so it will always be some other edge with endpoint r that is deleted if we ever transition to r. Thus, (v,r) is never removed, and so such a sequence suffices to tell us the spanning tree at r_0.

   (d) Note that each r_i represents a uniform distribution on rooted spanning trees(since we started in the stationary, uniform distribution), and so r_i is a vertex drawn from the uniform distribution of vertices. The standard random walk on G is time reversible and so the uniform distribution is stationary for such a MC. But since we start at r_0 with a uniform distribution by picking a vertex at random, subsequent vertices r_{-k} will all be from the uniform distribution(by definition of stationary), and so we have the correct distribution on sequences of roots.

   (e) Running in reverse, and using the intended spanning tree generation algorithm, we see that we generate exactly the spanning tree at r_0! This is because if r_i is the last root to take over

from v_i, then in reverse, it is the first time that we see v_i, and so the edge (v_i, r_i) is put in the spanning tree(and stays). But, this is exactly the spanning tree at r_0, and since that one is a uniformly random spanning tree, so is the one we generate.

(f) We only run until we see all vertices, so the expected running time is the expected cover time, which is polynomial.

2. (a) Let p be the fraction of spanning trees containing e. One of p, 1-p is large enough to get us a FPRAS for p or 1-p, by the standard sample and check method, using the sampling method generalized from problem 1.

(b) If we are estimating p(as defined above), then it suffices to "protect" edge e by contracting it and estimating the number of spanning trees in the remaining graph G', since any spanning tree in G' corresponds to a spanning tree in G that includes e. Similarly, if we are estimating 1-p, then we delete edge e and estimate the number of spanning trees in G'. In the end, to estimate the number of spanning trees S from the estimate S' in the residual graph G', we merely calculate S=S'/p or S=S'/(1-p) depending on which of the fractions we chose to estimate.

(c) Sample until we get $\mu_{\frac{\epsilon}{m}\frac{\delta}{m}}$ hits of either spanning trees with e or spanning trees without e, for some e, where m is the number of edges. Since one of p, 1-p is big enough, this takes polynomial time in expectation. If we ended up estimating p, recurse on the residual graph with e contracted, otherwise recurse on the residual graph with e deleted. In the base case of a single spanning tree, we terminate and declare(correctly) that the count in the residual graph is 1. Now, by (b), we merely calculate $S = \frac{1}{\prod q_i}$, where q_i is the estimate of the i-th fraction(p or 1-p value), to get our estimate for the number of spanning trees. There are at most m iterations, and each iteration takes expected polynomial time, so the entire algorithm takes expected polynomial time. Each $1/q_i$ has error ϵ/m(actually it is the inverse of $1+\epsilon$, but this is off by a constant factor which we can tweak away), so that the product has error at most ϵ. By Union bounding the failure probabilities of each step, we get that we fail overall with probability at most δ. This gives us our FPRAS.

(d) We sample as follows. For some edge e, count the number T of spanning trees, and the number C of spanning trees with e. With probability $\frac{C}{T}$, include the edge e and recurse on the residual graph with e contracted, otherwise recurse on the residual graph with e deleted. When we arrive at the base case of a single spanning tree, return it as our sample. Note that this takes at most m iterations and each one requires 2 count invocations, for a total of at most 2m calls to the counting algorithm. We now show that this is uniform sampling. Let C_i be the sequence of number of spanning trees in each residual graph that contains a specific spanning tree, where $C_1 = T$, the total number of spanning trees. For our given spanning tree, the probability that we pick it is $\frac{C_2}{T} \cdot \frac{C_3}{C_2} \cdots \frac{1}{C_k} = \frac{1}{T}$ by telescoping, and so our sample is uniform.