The Groundskeeper's Algorithm Works

Will Dana

July 17, 2019

The point of this writeup is to distill the argument, presented in [2], that the "groundskeeper's algorithm" for generating uniformly random spanning trees works. (The beginning of [1] also provides a helpful account.) This algorithm is as follows:

- Start at some vertex of a graph G.
- Perform a simple random walk: choose an adjacent vertex of the graph uniformly at random, move to it, and then repeat this process.
- Whenever you visit a vertex for the first time, add the edge you just crossed to your spanning-tree-in-progress.
- Repeat until all vertices are visited (which will happen with probability 1).

Theorem 1. *The above algorithm produces a spanning tree of G uniformly at random.*

Proof. It is straightforward to check that this actually produces a spanning tree. We thus consider the probability distribution induced on the set of spanning trees by this process.

We will want to track the additional information of where the algorithm starts, so we define a **rooted spanning tree** to be a spanning tree with a distinguished vertex called the **root**. We say that the groundskeeper's algorithm produces a tree whose root is the starting vertex. The reason we keep track of the root is that we will want to deal with a situation in which it changes, as explained next.

Instead of just the portion of the random walk which generates our tree, consider a (doubly) infinite random walk indexed by the integers

$$\ldots, w_{-1}, w_0, w_1, w_2, \ldots$$

Then this walk induces a sequence of spanning trees

$$\ldots, T_{-1}, T_0, T_1, T_2, \ldots$$

where T_m is the rooted spanning tree obtained by applying the groundskeeper's algorithm with the portion of the infinite walk starting at w_m.

Now we consider the resulting sequence of trees traversed backwards

$$\ldots, T_2, T_1, T_0, T_{-1}, \ldots$$

as a Markov chain. We want to know how T_m can be obtained from T_{m+1}. This is simple to describe, and depends only on the tree T_{m+1} and the vertex w_m.

Starting our walk from w_m rather than w_{m+1} adds a new edge to the tree, from w_m to w_{m+1}. If there was already an edge there, the same tree is traced out, but with a different root. Otherwise, the edge in T_{m+1} created when the walk visited w_m for the first time is no longer present, but the rest of the tree is unchanged. The crucial observation here is that we can figure out which edge in T_{m+1} was created by visiting w_m for the first time just by looking at the tree, without considering the precise steps in the walk: since there is a unique path between any two vertices in the tree, we know this edge must be the last edge of the unique path in T_{m+1} from w_{m+1} to w_m. This is illustrated in Figure 1.

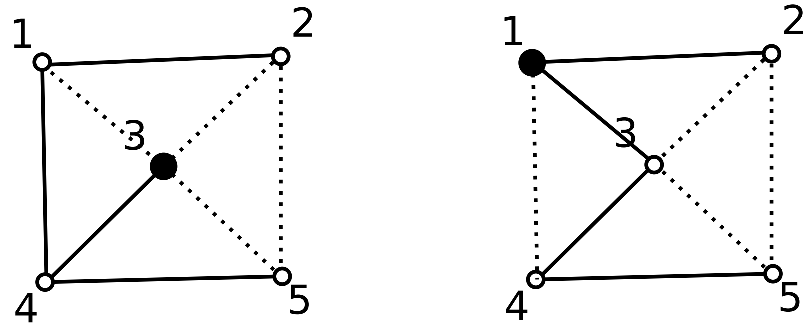

Figure 1: An example of the passage from T_{m+1} to T_m in the case that $w_{m+1} = 3$, $w_m = 1$. The edge omitted from the original tree, between 1 and 4, is the one on the path from 1 to 3.

We introduce a bit of notation to talk about this transformation. If t is a rooted spanning tree of G with root v and w is a neighbor of v in G, construct $F(t, w)$ as follows:

- Find the path from v to w in t, and delete the edge incident to w.
- Attach an edge from w to v, and place the root at w.

Then we've almost proven the following lemma:

Lemma 1. *For trees t and u,*

$$P(T_m = u \mid T_{m+1} = t) = \begin{cases} \frac{1}{r(t)} & \exists w \text{ such that } F(t,w) = u \\ 0 & \text{otherwise} \end{cases}$$

where $r(t)$ is the degree of t's root.

Proof. We've already seen that $T_m = F(T_{m+1}, w)$ for some neighbor w of the root of v, which explains the second line of the piecewise equation. There are $r(t)$ such neighbors, and they are all equally likely to be w_m, which explains the first line. □

Although it may seem straightforward, this lemma is at the heart of why the groundskeeper's algorithm is uniform: it shows that the rough behavior of the Markov chain we've constructed is independent of the particular structure of the trees involved.

For later use, we note that the transformation F defined above has an inverse F^{-1}. For a spanning tree u with root w and a neighbor x of w in G, defined $F^{-1}(u, x)$ as follows:

- Follow the path from x to w in u. Let v be the last vertex reached before w.
- Delete the edge between v and w.
- Add an edge between x and w. Make v (note, not x) the new root.

The reader should verify that if t is a tree with root v, w is a neighbor of v, and x is the first vertex (after w) encountered on the path from w to v in t, then

$$F^{-1}(F(t, w), x) = t.$$

For a particular example, note that reading Figure 1 right-to-left shows $F^{-1}(u, 4) \hookleftarrow u$.

Having this inverse gives us an important corollary. Let u be a spanning tree with root w, and $\mathcal{C}(u)$ be the set of all rooted spanning trees t such that $F(t, w) = u$.

Corollary 1.

$$|\mathcal{C}(u)| = r(u)$$

Proof. Suppose $t \in \mathcal{C}(u)$, so that $F(t, w) = u$. Then there exists a vertex x, a neighbor of w in the graph, such that $F^{-1}(F(t, w), x) = F^{-1}(u, x) = t$. This shows that all elements of $\mathcal{C}(u)$ are obtained from neighbors of w in this way, so $|\mathcal{C}(u)| \leq r(u)$. Additionally, for different neighbors x, x' the trees $F^{-1}(w, x), F^{-1}(w, x')$ are distinct, since a different edge is added for every choice of neighbor. Thus we have equality. □

Now that we have a handle on how moving around our starting vertex within a random walk changes the trees involved, we examine the probability of a particular rooted spanning tree appearing. By the nature of the process we've described, for a particular rooted spanning tree u, $P(T_m = u)$ is independent of m. Because of this, we say that the vector of probabilities $P(T_m = u)$ as u ranges over all rooted spanning trees is the **stationary distribution** of the Markov chain. The significance of being the stationary distribution is reflected in the following equation: we have

$$\begin{aligned} P(T_m = u) &= \sum_{\text{all trees } t} P(T_m = u \mid T_{m+1} = t) P(T_{m+1} = t) \\ &= \sum_{t \in \mathcal{C}(u)} \frac{1}{r(t)} P(T_{m+1} = t) \\ &= \sum_{t \in \mathcal{C}(u)} \frac{1}{r(t)} P(T_m = t) \end{aligned}$$

Thus our probabilities will solve this system of equations. Now we deploy the one black-boxed fact about Markov chains we need:

Fact. *A sufficiently nice Markov chain*[1] *has a unique stationary distribution.*

[1]Essentially, one in which it's possible to get from any state to any other state, which includes our chain of trees.

(If you'd like to learn more about this fact, the buzzwords to look up, besides "Markov chain", are "Perron-Frobenius theorem".)

So in order to describe the probability of a particular rooted tree appearing, we just need to give a probability distribution that satisfies the above equation and is thus the stationary distribution. Here it is!

Lemma 2. *Let K be the normalizing factor*[2]

$$K = \left(\sum_{v \text{ vertex}} \deg(v) \right) \tau(G)$$

where $\tau(G)$ is the total number of (unrooted) spanning trees of G. Then

$$P(T_m = u) = \frac{r(u)}{K}.$$

Proof. We just plug this formula into the equation defining the stationary distribution, and check that the equation is satisfied:

$$\begin{aligned} P(T_m = u) &= \sum_{\text{all trees } t} P(T_m = u \mid T_{m+1} = t) P(T_{m+1} = t) \\ &= \sum_{t \in \mathcal{C}(u)} \frac{1}{r(t)} P(T_{m+1} = t) \\ &= \sum_{t \in \mathcal{C}(u)} \frac{1}{r(t)} P(T_m = t) \\ &= \sum_{t \in \mathcal{C}(u)} \frac{1}{r(t)} \frac{r(t)}{K} \\ &= |\mathcal{C}(u)| \frac{1}{K} = \frac{r(u)}{K} \end{aligned}$$

where in the last line we use Corollary 1. □

The takeaway from this lemma is that the probability that a tree obtained through the groundskeeper's algorithm is a specific tree depends only on the degree of its root. In particular, if we restrict to the assumption that our tree has a specific root, then all trees are equally likely. □

References

[1] Aldous, David J. "The Random Walk Construction of Uniform Spanning Trees and Uniform Labelled Trees." SIAM J. Discrete Math v. 3 (1990), p. 450–465.

[2] Pemantle, Robin. "Uniform random spanning trees". `arXiv:math/0404099`.

[2] That is, we need this so that summing the probability over all trees will give 1.