1 Uniform Spanning Trees

A graph typically has an enormous number of spanning trees. Becuase of this, it is not obvious how to choose one uniformly at random in a reasonable amount of time. We are going to present an algorithm that works quickly by exploiting some hidden independence in Markov chains. This algorithm is of enormous theoretical importance

Consider a finite markov

Definition 1.1. Every connected graph has a **Spanning Tree**, that is, a subgraph that is a tree and that includes every vertex.

Definition 1.2. Let $p(\cdot,\cdot)$ be the transition probability function of a finite-state irreducible Markov chain. The directed graph associated to this chain has for vertices the states and for edges all $\langle x, y\rangle$ for which $p(x, y) > 0$. Edges e are oriented from tail e^- to head e^+. We call a connected subgraph a **spanning tree** if it includes every vertex, there is no cycle, and there is one vertex, the **root**, such that every vertex other than the root is the tail of exactly one edge in the tree.

Definition 1.3. The **weight** of a spanning tree T (on a finite irreducible MC) is

$$\Psi(T) := \prod_{e\in T} p(e)$$

In the case of a reversible Markov chain, that is $\pi_i p_{ij} = \pi_j p_{ji} = c(e)$, with transition probabilities given by $p(e) := c(e)/\pi\left(e^-\right)$ and therefore

$$\sum_{x=e^-} p(e) = 1 = \sum_{x=e^-} c(e)/\pi(x) = 1/\pi\left(x\right)\sum_{x=e^-} c(e) \Rightarrow \pi\left(x\right) = \sum_{x=e^-} c(e)$$

and so the weight of the spanning tree is

$$\Psi(T) = \prod_{e\in T} p(e) = \prod_{e\in T} c(e)/\prod_{x\neq \text{root}} \pi(x)$$

The root is fixed, so we pick a tree with probability proportional to $\Psi(T)/\pi(\text{root}(T))$ which is proportional to

$$\Xi(T) := \prod_{e\in T} c(e)$$

Remark 1.1. If $c = 1$ for all edges, all spanning trees are equally likely. If all the weights $c(e)$ are positive integers, then we can replace each edge e by $c(e)$ parallel copies of e and interpret the uniform spanning tree measure in the resulting multigraph as the probability measure above with the probability of T proportional to $Xi(T)$. If all weights are divided by the same constant, then the probability measure does not change, so the case of rational weights can still be thought of as corresponding to a uniform spanning tree. Since the case of general weights is a limit of rational weights, we use the term **weighted uniform spanning tree** for such a probability measure.

Remark 1.2. Now suppose we have a method of choosing a rooted spanning tree at random proportional to the weights $\Psi(\cdot)$ for a reversible Markov chain. Consider any vertex u on a weighted undirected graph. If we choose a random spanning tree rooted at u proportionally to the weights $\Psi(\cdot)$ and forget about the orientation of its edges and also about the root, then we obtain an unrooted spanning tree of the undirected graph, chosen proportionally to the weights $\Xi(\cdot)$. In particular if the conductances are all equal, which corresponds to the Markov chain being simple random walk, then we get a uniformly chosen spanning tree.

2 Wilson's Method

Definition 2.1. If $\mathcal{P}$ is a finite path $\langle x_0, x_1, \cdots x_l \rangle$ in a directed or undirected graph G, we define the **loop erasure**[1] of $\mathcal{P}$, denoted $LE(\mathcal{P}) = \langle u_0, u_1, \cdots, u_m \rangle$, by erasing cycles in $\mathcal{P}$ in the order they appear.

Remark 2.1. In the case of a multigraph, one cannot notate a path merely by the vertices it visits. However, the notion of loop erasure should still be clear.

Definition 2.2. Wilson's Method. To generate a random spanning tree with a given root r with probability proportional to the weights $\Psi(\cdot)$

Theorem 2.1. Given any finite-state irreducible Markov chain and any state r, Wilson's method yields a random spanning tree rooted at r with distribution proportional to $\Psi(\cdot)$. Therefore, for any finite connected undirected graph, Wilson's method yields a random spanning tree that, when the orientation and root are forgotten, has distribution proportional to $\Xi(\cdot)$

Before proving this theorem, we introduce a couple of helpful images and notions.

For each state x, with the exception of an arbitrarily chosen root r, think of $\langle S_i^x; i \geq 1 \rangle$ as a stack of realizations of random variables, whose distributions are given by the MC, lying under the state x with s_1^x being on top.

Definition 2.3. The **visible graph** is the directed graph whose arrows are given by the topmost elements of each stack. We can modify the visible graph by **popping** elements from stacks.

We now have a different way of thinking about Wilson's method, which can be re-worded as follows: If the visible graph conains no (directed) cycles, then it is a spanning tree rooted at r. Otherwise, we **pop** a cycle, meaning that we pop all the top items of the stacks under the vertices of a cycle.

Example 2.1. Consider these three graphs.

[1] This should probably be called *cycle erasure*, but alas, the nomenclature is already established.

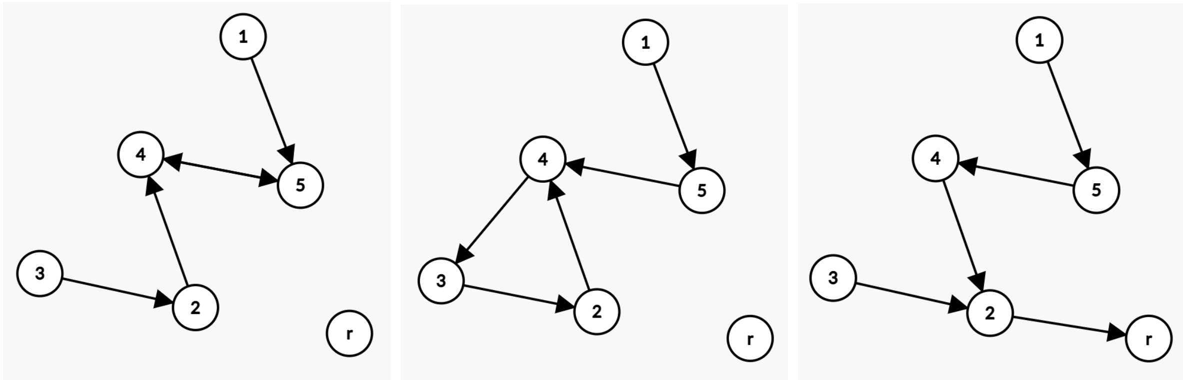

Figure 1: Three possible visible graphs that might occur consecutively upon cycle poppings.

Definition 2.4. To prove the theorem, we will keep track of the locations in the stacks of the edges that are popped, which we will call colors. That is, an edge (x, S_i^x) has **color** i. A **colored cycle** is simply a cycle whose edges are colored as such (the colors of the edges in a cycle do not have to be the same as each other).

Remark 2.2. While a cycle of vertices might be popped many times, a colored cycle can be popped at most once.

Lemma 2.1. Given any stacks under the states, the order in which cycles are popped is irrelevant in the sense that every order pops an infinite number of cycles (w/ probability 0) or every order pops the same (finite set of) coloured cycles, thus leaving the same coloured spanning tree on top in the latter case.

Remark 2.3. Draw some cycles and try popping them. You might find it intuitive that when a cycle is popped, it cannot grow by fusing into a larger cycle—because that would require an extra edge, it can only (1) stay the same size (though possibly in a different arangement), (2) split into smaller cycles, or (3) some subset may join the growing spanning tree connected to the root.

Exercise 1. Consider the following stacks. Draw the visible graph and pop cycles until you hit the spanning tree. When you draw graphs, label the edges according to their color.

1	2	3	4	5	r
2	1	4	5	3	
4	2	2	2	1	
r	3	1	1	r	
2	1	2	5	1	
4	r	r	1	3	

Table 1: The MC has six states, one called r, which is the root. The first five elemtns of each stack are listed under the corresponding states.

Proof of Lemma 2.1. We will show that if C is any colored cycle that can be popped, i.e. there is some sequence $C_1, C_2, \cdots, C_n = C$ that may be popped in that order, but some colored cycle $C' \neq C_1$ happens to be the first colored cycle popped, then (1) $C = C'$, or else (2) C can still be popped fromt he stacks after C' is popped. Once we show this, we are done, since if there are an infinite number of colored cycles that can be popped, the popping can never stop; whereas in the alternative case, every colored cycle that can be popped will be popped.

If $C' \cap C_k = \emptyset \forall k$, then we can still pop everything no problem. However, if the intersection is non-empty for at least one cycle, and let k be the index of the first cycle such that $C' \cap C_k \neq \emptyset$; then let x be a vertex in the intersection $C' \cap C_k$. Since all the edges in C' have color 1, the edge coming out of x has color 1. Since $x \notin C_1 \cup \cdots \cup C_{k-1}$, then the edge coming out of C_k must also have color 1. We keep applying the same argument to the successive vertices of C' and C_k, and conclude that $C' = C_k$. Therefore, we can still pop C_n like so $C' = C_k, C_1, C_2, \cdots, C_{k-1}, C_{k+1}, \cdots, C_n$. □

Exercise 2. To see that Wilson's method is one way of popping stacks, think of the random walks and loops as creating stacks. Sketch some graphs and simulate running Wilson's algorithm on them, pushing to onto queues that we then use those in the visible graph view of things. If we think of these queues as predetermined stacks, Wilson's algorithm is equivalent to popping cycles on those stacks in a particular order. Lemma 2.1 tells us that it doesn't matter how we pop those cycles.

Proof of Theorem 2.1. Wilson's method certainly stops with probability one at a spanning tree. Using stacks to run this process, we see that Wilson's method pops all the cycles lying over a spanning tree.

To show that the distribution is the desired one, think of a given set of stacks as defining a finite set O of coloured cycles lying over a a noncolored spanning tree T. We don't need to keep track of the colors in the spanning tree, since they are easily recovered from the colors in the cycles over it.

Let X be the set of all pairs (O, T) that can arise from stacks corresponding to our given MC chain. If $(O, T) \in X$ for any other spanning tree T': indeed anything at all can be in the stacks under any finite set O of colored cycles. That is, $X = X_1 \times X_2$, where X_1 is a certain collection of sets of colored cycles and X_2 is the set of all noncolored spanning trees. (TBC on Wednesday) □