A Reverse Aldous/Broder Algorithm

Yiping Hu* Russell Lyons† Pengfei Tang‡

July 23, 2019

Abstract

The Aldous/Broder algorithm provides a way of sampling a uniform spanning tree for finite connected graphs using simple random walk. Namely, start a simple random walk on a connected graph and stop at the cover time. The tree formed by all the first-entrance edges has the law of a uniform spanning tree. Here we show that the tree formed by all the last-exit edges also has the law of a uniform spanning tree. This answers a question of Tom Hayes and Cris Moore from 2010. The proof relies on a bijection that is related to the BEST theorem in graph theory.

1 Introduction

Let $G = (V, E)$ be a finite connected graph and let $\mathcal{T}$ be the set of spanning trees of G. Obviously $\mathcal{T}$ is a finite set. The ***uniform spanning tree*** is the uniform measure on $\mathcal{T}$, which is denoted by $\mathsf{UST}(G)$.

In the late 1980s, Aldous [1] and Broder [4] found an algorithm to generate a uniform spanning tree using simple random walk on G; both authors thank Persi Diaconis for discussions. The algorithm is now called the Aldous/Broder algorithm. It generates a sample of the uniform spanning tree as follows.

Start a simple random walk $(X_n)_{n\geq 0}$ on G and stop at the cover time, i.e., stop when it first visits all the vertices of G. Collect all the first-entrance edges, i.e., edges (X_n, X_{n+1}) for $n \geq 0$ such that $X_{n+1} \neq X_k$ for all $k \leq n$. These edges form a random spanning tree, T. Then this random tree T has the law $\mathsf{UST}(G)$. The simple random walk can start from any vertex of G, and so X_0 can have any initial distribution.

The proof of this result depends on ideas related to the Markov Chain Tree Theorem. Namely, let $(X_n)_{n\in\mathbb{Z}}$ be a biinfinite stationary random walk on G. Let the last exit from a vertex x before time 0 be $\lambda(x) := \max\{n < 0\,;\ X_n = x\}$. Then the random spanning tree $\{(x, X_{\lambda(x)+1})\,;\ x \in V \setminus \{X_0\}\}$ of last-exit edges has the uniform distribution. One then proves the validity of the Aldous/Broder algorithm by reversing time and using reversibility of simple random walk; see, e.g., [6, Section 4.4] for a proof. The algorithmic aspects were studied by [4], while theoretical implications were studied by [1] and, later, Pemantle [8].

In 2010 ("domino" forum email discussion, 9 Sep.; published in [7, p. 645]), Tom Hayes and Cris Moore asked whether the tree formed by all the last-exit edges starting at time 0 and ending at the cover time has the law of the uniform spanning tree. Here we give a positive answer to this question.

*Department of Mathematics, University of Washington. Partially supported by the McFarlan Fellowship from University of Washington. Email: huypken@uw.edu.

†Department of Mathematics, Indiana University. Partially supported by the National Science Foundation under grant DMS-1612363. Email: rdlyons@indiana.edu.

‡Department of Mathematics, Indiana University. Partially supported by the National Science Foundation under grant DMS-1612363. Email: tangp@iu.edu.

Theorem 1.1. *Let G be a finite, connected graph. Start a simple random walk on G and stop at the cover time. Let T be the random tree formed by all the last-exit edges. Then T has the law $\mathsf{UST}(G)$.*

Remark 1.2. *As we will show, the corresponding result holds for finite, connected networks and the associated network random walk.*

We call this algorithm the ***reverse Aldous/Broder*** algorithm.

Our proof shows a remarkable strengthening of this equality: Notice that the Aldous/Broder algorithm run up to time n gives a tree T_n on the vertices $\{X_k\,;\ k \le n\}$. The evolution of the rooted tree (T_n, X_n) to (T_{n+1}, X_{n+1}) is given by a Markov chain. For this chain, T_{n+1} contains T_n and may be equal to T_n. Furthermore, if τ denotes the cover time, then $T_n = T_\tau$ for all $n \ge \tau$. Similarly, the reverse Aldous/Broder algorithm run up to time n gives a tree $\overline{T}_n$ on the vertices $\{X_k\,;\ k \le n\}$. The evolution of $(\overline{T}_n, X_n)$ to $(\overline{T}_{n+1}, X_{n+1})$ is also given by a Markov chain. For this chain, $\overline{T}_{n+1}$ *need not* contain $\overline{T}_n$. In addition, $\overline{T}_n$ need not equal $\overline{T}_\tau$ for $n > \tau$. Our theorem above is that the distributions of T_τ and $\overline{T}_\tau$ are the same. Our strengthening will show that for every n, the distributions of T_n and $\overline{T}_n$ are the same, even though the two Markov chains have different transition probabilities. Moreover, for every n, the distributions of $T_{n\wedge\tau}$ and $\overline{T}_{n\wedge\tau}$ are the same. These are both proved as Corollary 2.2.

2 Proof of the main theorem

We allow weights $w\colon E \to (0,\infty)$ on the edges, so that (G, w) is a network, and the corresponding ***weighted uniform spanning tree*** measure puts mass proportional to $\prod_{e\in T} w(e)$ on a spanning tree T. We could also allow parallel edges and loops in G, but this would simply require more complicated notation, so we assume that G is a simple graph. Write $w(x)$ for the sum of $w(e)$ over all edges e incident to x.

Let x and y be two vertices of G. If there is an edge in G connecting x and y, then we write $x \sim y$. We call $\gamma = (v_0, \dots, v_n)$ a ***path*** (or a walk) on G from v_0 to v_n if $v_{i-1} \sim v_i$ for $i = 1, \dots, n$, and $|\gamma| := n$ is called the ***length*** of the path γ.

Let $\mathcal{P}_n^{x,y}$ denote the set of all paths in G from x to y with length n. Simple random walk is the Markov chain on V with transition probabilities $p(x,y) := \frac{1}{\deg(x)}\mathbf{1}_{\{x\sim y\}}$. The network random walk on (G, w) has instead the transition probabilities $p(x,y) := \frac{w(x,y)}{w(x)}\mathbf{1}_{\{x\sim y\}}$.

Let $\mathscr{T}$ be the set of all subtrees of G, including those that are not necessarily spanning.

For a path $\gamma = (v_0, \dots, v_n) \in \mathcal{P}_n^{x,y}$, we write $V(\gamma) = \{v_0, \dots, v_n\}$ for the set of vertices of γ. For each $u \in V(\gamma)\backslash\{x\}$, there is a smallest index $i \ge 1$ such that $v_i = u$; we call the edge (v_{i-1}, v_i) the ***first-entrance edge*** to u.

Define the ***first-entrance operator*** $F\colon \mathcal{P}_n^{x,y} \to \mathscr{T}$ by setting $F(\gamma)$ to be the tree formed by all the first-entrance edges to vertices in $V(\gamma)\backslash\{x\}$.

Similarly, for each $u \in V(\gamma)\backslash\{y\}$, there is a largest index $i \le n-1$ such that $v_i = u$, and we call the edge $e = (v_i, v_{i+1})$ the ***last-exit edge*** of u. Define the ***last-exit operator*** $L\colon \mathcal{P}_n^{x,y} \to \mathscr{T}$ by setting $L(\gamma)$ to be the tree formed by all the last-exit edges of vertices in $V(\gamma)\backslash\{y\}$.

For a finite path $\gamma = (v_0, \dots, v_n)$ in G with length n started at v_0, we write

$$p(\gamma) := \mathbb{P}_{v_0}\big[(X_0, \dots, X_n) = \gamma\big] = \frac{\prod_{k=0}^{n-1} w(v_k, v_{k+1})}{\prod_{k=0}^{n-1} w(v_k)}, \tag{2.1}$$

where $\mathbb{P}_{v_0}$ denotes the law of the network random walk on (G, w) started from v_0. It is known that the Aldous/Broder algorithm holds in this more general context: the network random walk

generates the weighted uniform spanning tree by collecting the first-entrance edges up to the cover time.

The main ingredient for proving our theorem is the following lemma.

Lemma 2.1. *For any $x, y \in V(G)$ and $n \geq 0$, there is a bijection $\Phi\colon \mathcal{P}_n^{x,y} \to \mathcal{P}_n^{x,y}$ such that $F = L \circ \Phi$ and*

$$\forall \gamma \in \mathcal{P}_n^{x,y} \qquad p(\gamma) = p\big(\Phi(\gamma)\big). \tag{2.2}$$

Moreover, Φ preserves the number of times each vertex is visited as well as the number of times that each (unoriented) edge is crossed.

It follows that Φ is also a measure-preserving bijection on each subset of $\mathcal{P}_n^{x,y}$ specified by how many times each edge is crossed or which vertices are visited. For example, such a subset is the set of paths such that every vertex is visited and y is visited only once.

Before proving Lemma 2.1, we show how it gives our claims.

Consider the network random walk $(X_n)_{n=0}^{\infty}$ on G with arbitrary initial distribution and define τ to be the cover time of the graph G. Write $U \overset{\mathscr{D}}{=} V$ when U and V have the same distribution.

Corollary 2.2. *We have for all $n \in \mathbb{N}$,*

(i) $F\big((X_0, \dots, X_n)\big) \overset{\mathscr{D}}{=} L\big((X_0, \dots, X_n)\big)$ *and*

(ii) $F\big((X_0, \dots, X_{\tau\wedge n})\big) \overset{\mathscr{D}}{=} L\big((X_0, \dots, X_{\tau\wedge n})\big)$.

Proof. This is immediate from Lemma 2.1 and the remarks following it. □

Proof of Theorem 1.1. Letting $n \to \infty$ in Corollary 2.2(ii) and noting that $\mathbb{P}[\tau \geq n] \to 0$ as $n \to \infty$, one obtains that

$$F\big((X_0, \dots, X_\tau)\big) \overset{\mathscr{D}}{=} L\big((X_0, \dots, X_\tau)\big).$$

By the Aldous/Broder algorithm, $F\big((X_0, \dots, X_\tau)\big)$ has the law of $\mathsf{UST}(G)$, and thus so does $L\big((X_0, \dots, X_\tau)\big)$. □

2.1 Augmented operators $\hat{L}$ and $\hat{F}$ and the reversing operator R

Note that if $x = y$, then Lemma 2.1 is trivial: take Φ to reverse the path. Thus, we assume from now on that $x \neq y$.

We encode every walk by a colored multi-digraph. We say a multi-digraph marked with a ***start*** vertex x and an ***end*** vertex y is ***balanced*** if the indegree of each vertex $u \notin \{x, y\}$ equals the outdegree of u, the outdegree of x is larger than its indegree by one, and the indegree of y is larger than its outdegree by one. The balance property is necessary for the existence of an Eulerian path in a multi-digraph. The coloring is either of the following two types.

By an ***exit coloring*** of a multi-digraph, we mean an assignment to each vertex v of a linear ordering on the set $\mathsf{Out}(v)$ of its outgoing edges. We call an edge ***lighter*** or ***darker*** if it is smaller or larger in the ordering. The maximal (darkest) edge in $\mathsf{Out}(v)$ is regarded as ***black***, except for $v = y$. For the ***entrance coloring***, the ordering is on the edges $\mathsf{In}(v)$ incoming to v instead, and there is no black edge leading into x.

Let $\mathcal{L}_n^{x,y}$ (resp., $\mathcal{F}_n^{x,y}$) denote the set of all colored multi-digraphs marked with the start x and end y satisfying the following five properties:

(i) the vertex set of the multi-graph contains the start x and end y and is a subset of the vertex set of the original graph G;

(ii) there are n directed edges in total;

(iii) the multi-digraph is balanced;

(iv) the coloring is an exit coloring (resp., an entrance coloring);

(v) all black edges form a directed spanning tree of the multi-digraph with all edges going from leaves to the root y (resp., from the root x to leaves).

We define the augmented last-exit operator $\hat{L}\colon \mathcal{P}_n^{x,y} \to \mathcal{L}_n^{x,y}$, which, given a walk, returns a colored multi-digraph in the most obvious way: The vertex set consists of all the visited vertices, and we draw one directed edge for each step of the walk. The multi-digraph is clearly balanced. The exit coloring is naturally given by the order in which these outgoing edges are traversed. Also recall that there is no black edge leading out of y. Note that the tree formed by all black edges, ignoring their orientation, is exactly the output of the last-exit operator L defined above, so $\hat{L}$ can be viewed as an augmentation of L. All conditions can be checked easily, so the map is well defined.

Similarly, we can define the augmented first-entrance operator $\hat{F}\colon \mathcal{P}_n^{x,y} \to \mathcal{F}_n^{x,y}$. The only difference is that we consider the entrance coloring on incoming edges instead of the exit coloring on outgoing edges, and we use the *reverse* of the order in which incoming edges are traversed, so that black edges, in particular, are first-entrance edges.

Clearly $\hat{L}$ and $\hat{F}$ are both injections.

Lemma 2.3. *The map $\hat{L}$ is a bijection.*

Proof. We describe the inverse operator for $\hat{L}$. Given a colored multi-digraph in $\mathcal{L}_n^{x,y}$, we can associate it with a walk by starting from x and traversing the multi-digraph according to the following instruction:

always exit from the lightest unused edge until you run out of edges to use.

Notice that by the balance property, the algorithm always terminates at y. It remains to show that the walk is an Eulerian path (i.e., uses all n edges), so that the operator is well defined. Given this, since the above rule respects the exit coloring, we may deduce that the traversal algorithm gives the inverse $\hat{L}^{-1}$.

If the walk is not Eulerian, i.e., if some edge has not been used, then some outgoing edge from some vertex u has not been used. According to our instructions, it follows that the black edge leaving u has not been used. Let that black edge be (u, v). By the balance condition, some outgoing edge from v has not been used, whence the black edge leaving v has not been used (unless $v = y$). By repeating this argument and using condition (v), we arrive at the conclusion that some edge leaving y has not been used, whence the walk cannot have terminated, a contradiction. □

This bijection $\hat{L}$ was also used in the proof of the so-called BEST theorem; see the proof of Theorem 5b in [2].

One can prove similarly that $\hat{F}$ is a bijection, but we will not need to do so.

Define a reversing operator $R\colon \mathcal{F}_n^{x,y} \to \mathcal{L}_n^{x,y}$ that maintains the vertex set but reverses all edges, except that the directions of the black edges on the unique path from x to y in the spanning tree remain unchanged. The coloring is also maintained: for $H \in \mathcal{F}_n^{x,y}$ and a vertex u, the set $\mathsf{In}(u)$ in H is mapped to $\mathsf{Out}(u)$ in $R(H)$, and so this set of edges can maintain its linear ordering, except that if $u \notin \{x, y\}$ is on the path from x to y, then one black edge is replaced by another, whereas x gains an outgoing black edge in $R(H)$ compared to the incoming edges to x in H, and y loses an outgoing black edge in $R(H)$ compared to the incoming edges to y in H. It is not hard to see that the operator R does have codomain $\mathcal{L}_n^{x,y}$ and is bijective.

2.2 Proof of Lemma 2.1

Proof of Lemma 2.1. We may now define the main construction, illustrated in Figure 1:

$$\Phi := \hat{L}^{-1} \circ R \circ \hat{F}.$$

Since R maintains the coloring, the black edges that formed the first-entrance spanning tree in $\mathcal{F}_n^{x,y}$ now form the last-exit version in $\mathcal{L}_n^{x,y}$ after reversal, that is, $F = L \circ \Phi$. Because $\hat{F}$, R, and $\hat{L}^{-1}$ are injections, so is Φ, which forces Φ to be a bijection. The map Φ preserves the random walk measure because all edges and vertices are visited same number of times: see (2.1). □

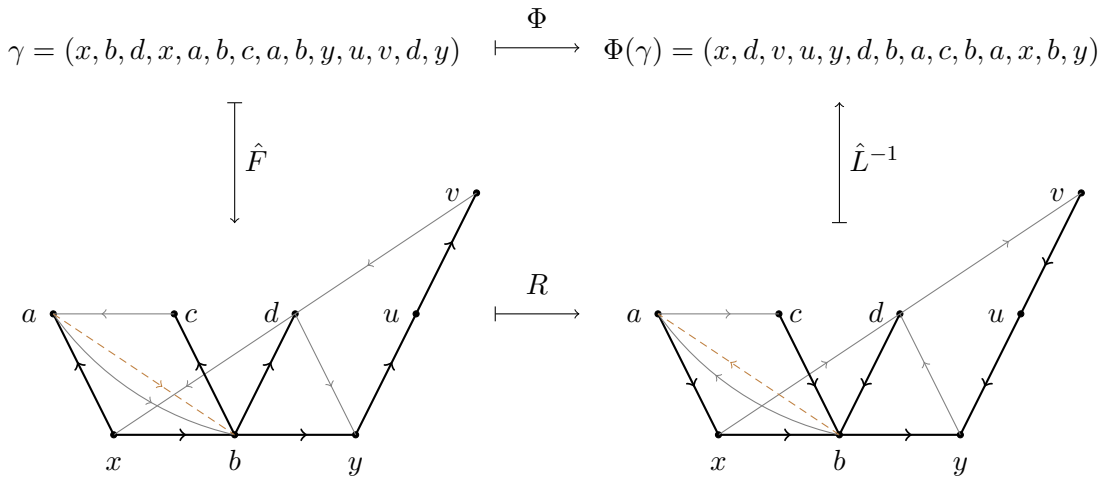

Figure 1: An example of the bijection R, where we use the order that brown, dashed edges are lighter than gray edges and gray edges are lighter than black edges. In addition, black edges are drawn thick; these form the spanning tree. Note that R reverses the orientation of every edge except for the black edges that form a path from x to y.

3 Concluding remarks

Suppose G is a locally finite, infinite connected graph. An ***exhaustion*** of G is a sequence of finite connected subgraphs $G_n = (V_n, E_n)$ of G such that $G_n \subset G_{n+1}$ and $G = \bigcup G_n$. Suppose that V_n induces G_n, i.e., G_n is the maximal subgraph of G with vertex set V_n. Let G_n^* be the graph formed from G by contracting all vertices outside V_n to a new vertex ∂_n. Let T_n be a sample of $\mathsf{UST}(G_n^*)$. Then the ***wired uniform spanning forest*** is the weak limit of T_n. If we orient T_n toward ∂_n and then take the weak limit, we get the ***oriented wired uniform spanning forest*** of G. For details, see [3] or [6, Chapter 10].

Wilson's algorithm [9] is another efficient way of sampling uniform spanning tree for finite connected graph using loop erasure of simple random walks. It can be applied to recurrent networks directly. For transient networks, Wilson's algorithm can also be applied with a simple modification. This is called Wilson's method rooted at infinity; see [3] for details.

The Aldous/Broder algorithm can also be used to sample the wired uniform spanning forest on recurrent networks directly. However, the extension of the Aldous/Broder algorithm for sampling

the wired uniform spanning forest on transient networks was found much later by Hutchcroft [5] using the interlacement process.

Here we simply state a similar interlacement using last-exit edges. It is not directly related to our reverse Aldous/Broder algorithm. Instead, it relates to the process created from the stationary random walk on the nonpositive integers that we recalled in our introduction. The interested reader can refer to [5] for details on the interlacement process and the interlacement Aldous/Broder algorithm.

Theorem 3.1. *Let G be a transient, connected, locally finite network, let $\mathscr{I}$ be the interlacement process on G, and let $t \in \mathbb{R}$. For each vertex v of G, let $\lambda_t(v)$ be the largest time before t such that there exists a trajectory $\big(W_{\lambda_t(v)}, \lambda_t(v)\big) \in \mathscr{I}$ passing through v, and let $e_t(v)$ be the oriented edge of G that is traversed by the trajectory $W_{\lambda_t(v)}$ as it leaves v for the last time before t. Then $\{e_t(v)\,;\ v \in V\}$ has the law of the oriented wired uniform spanning forest of G.*

The proof of Theorem 3.1 is simply an analogue to that of [5, Theorem 1.1]. This version can be used in place of that used by Hutchcroft and is perhaps more natural.

References

[1] D. Aldous. (1990). The random walk construction of uniform spanning trees and uniform labelled trees. *SIAM J. Discrete Math.* **3** no. 4, 450–465. MR1069105

[2] T. van Aardenne-Ehrenfest and N.G. de Bruijn. (1951). Circuits and trees in oriented linear graphs. *Simon Stevin* **28** 203–217. MR0047311

[3] I. Benjamini, R. Lyons, Y. Peres, and O. Schramm. (2001). Uniform spanning forests. *Ann. Probab.* **29** 1–65. MR1825141 (2003a:60015)

[4] A. Broder. (1989). Generating random spanning trees. In *30th Annual Symposium on Foundations of Computer Science (Research Triangle Park, North Carolina)*, pages 442–447. IEEE, New York.

[5] T. Hutchcroft. (2018). Interlacements and the wired uniform spanning forest. *Ann. Probab.* **46** 1170–1200. MR3773383

[6] R. Lyons and Y. Peres. (2016). *Probability on trees and networks*, volume 42 of *Cambridge Series in Statistical and Probabilistic Mathematics*. Cambridge University Press, New York. MR3616205

[7] C. Moore and S. Mertens. (2011). *The nature of computation*. Oxford University Press, Oxford. MR2849868

[8] R. Pemantle. (1991). Choosing a spanning tree for the integer lattice uniformly. *Ann. Probab.*, **19**(4), 1559–1574. MR92g:60014

[9] D. B. Wilson. (1996). Generating random spanning trees more quickly than the cover time. In *Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing (Philadelphia, PA, 1996)*, pages 296–303. ACM, New York. MR1427525