proof (Polya's thm d=3 is rec.) Let $u \sim \text{Uniform}(S^2)$ (law $P$) and let $L_u$ be the unique $0 \to \infty$ half-line.

Let $r^u = (r_1^u, r_2^u, \ldots)$ be the (a.s.) unique closest path to $L_u$ in $E$ directed from $0 \to \infty$.

$i(e) = 1(e \in r^u) - 1(-e \in r^u)$ is a unit flow $0 \to \infty$ (check). If we average this flow with $u$ we get another unit flow $0 \to \infty$.

$s(e) = \mathbb{E}(P(r^u = e) - P(r^u = -e))$

We now show that this flow has finite energy (geometric argument).

For any $e \in E$ with midpoint a distance $R$ from 0 we have $EA$ indep. of $R$ s.t. $P(\exists T^u \leq R^2)$ (just by construction), the best if $e \in r^u$ then $L_u$ must pass within a distance of at most 2 of $e$. Also $EA$ s.t. there are at most $Bn^2$ edges distance $n$ between $n$ and $n+1$ of the origin. So

$\mathbb{E}(s) \leq \sum_{n=1}^{\infty} A^2 n^{-4} Bn^2 < \infty.$

So distance $R \in [n, n+1)$ edges
USTs connect with RWs (irreducible Markov processes) on graphs and electrical networks. Surprisingly, broad theoretical importance. Also connections with our next topic: Percolation.

**Def (Spanning tree)** Let $G = (V,E)$ be a finite connected graph. A spanning tree of $G$ is a connected subgraph containing all vertices (spans) and no cycles (tree). Let $\mathcal{T}$ be the set of spanning trees.

$\mathcal{T}$ is a UST if it has distribution

$$P(T = \omega) = \frac{1}{|\mathcal{T}|},$$

We may also

**Def (Directed (rooted) spanning tree (arborescence))**

Let $G = (V,E)$ be a finite connected directed graph and fix $v \in V$ called the root. For $e = (v, w) \in E$ we call $e^{-}$ the tail, $e^{+} = w$ the head.

We call a subgraph a rooted spanning tree if it includes every vertex (state), there are no cycles, and every vertex except the root has a unique tail connected to it (directed toward root).

Clearly, if we ignore direction and root this is a spanning tree of $\mathcal{T}$.
There could be a huge number of spanning trees of any remotely big graph. We would like a way to construct them.

Wilson's algorithm is an efficient way and is linked to Markov pro chains on G and therefore the electrical net analogy.

In fact we can sample more general random spanning trees.

If \((X_n)_{n \geq 0}\) is a MCT with trans matrix \(P\) then Wilson's algorithm will so can sample trees proportional to the weights

\[
\Psi(T) = \prod_{e \in T} P(e^-) P(e^+)
\]

by explicitly using \((X_n)_{n \geq 0}\). If \(P\) is reversible w.r.t. \(\pi\) then

\[
\Psi(T) = \prod_{e \in T} \pi(e^-) \pi(e^+) \frac{\pi(e)}{\pi^T(e)}
\]

in particular if \((X_n)_{n \geq 0}\) is a SMT on \(G\)

\[
P(1,0) = \{ \deg(1) \}
\]

if \((1,0) \in E\), then

\[
\Psi(T) \propto \deg(1) \text{ and } \Psi(T) \text{ is indep of } T
\]

That is we get a UST.
To describe the algorithm we will need the concept of loop-erasure. For a path $P = (y_0, y_1, \ldots, y_n)$ on $G$, we construct $LE(P)$ by removing cycles in the order they appear.

Let $J = \min\{ j \geq 1 : y_j = y_i \text{ for some } i < j \}$ and $I$ is the unique index of the point it revisits.

Let $P(J) = (y_0, \ldots, y_J, y_{J+1}, \ldots, y_n)$ be the sub-path obtained by removing the first cycle. We iterate to get $EJ LE(P)$.

**Wilson's algorithm**

1. If $T_i$ spans $\mathcal{E}$ we are done or and the alg. stops.

2. Otherwise pick the smallest element (vertex) of $T_i$ and run $(X_n)_{n \geq 0}$ started here until it hits $T_i$. Call the random path it followed $T_{i+1}$.
Each stage of the alg. gives a sub-tree of $G$, directed towards $r$. The alg. stops when this tree spans $G$. We just need to check it has the correct distribution.

Thus 4.7, Wilson's alg. yields a random spanning tree rooted at $r$ with dist prop to $\Psi(\cdot)$.

In order to prove this we first need an important deterministic result about removing cycles.

Similar to the graphical construction of its tiny MCs on finite state space (e.g. S33a notes) we can construct the Markov process on a big prob. space where we think of all the possible transitions as being a priori assigned. For each state $i \in I$ we put a "stack" of commands (sequence of moves) with the correct distribution. Each time the process visits a state it follows the instruction at the top of the pile and throws it away.

For each $I \in \mathbb{P}(E \setminus I)$ (we only care about the process up to the first time it hits $r$)

Let $S$ be indep (in $i$ and $\eta$) with laws $\mathbb{P}(S_i = \eta) = \mathbb{P}(\eta_i = \eta)$ for $\forall \eta \in \Sigma$

Clearly this graphical construction can be used to construct $(X_n)_{n \geq 0}$ with the correct distr.
The stacks use an edge removal algorithm. This is called depth-first search of the remaining graph. To reach an edge, we can use a stack. Initially, the stack is empty. The top of the stack is defined as the visible graph. The visible graph is the graph with edge colours. We never remove an edge from the stack. For each edge, we add its colour to the visible graph. If the top of the stack has a new visible graph, we pop the edge from the stack. We then remove the top of the stack. The visible graph is the graph with edge colours. We never remove an edge from the stack. For each edge, we add its colour to the visible graph.

Example

<table>
<thead>
<tr>
<th>Colour</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
</tbody>
</table>

Initial

To reach an edge, we can use a stack. Initially, the stack is empty.

Initial
Example cont. pop the next cycle.

A spanning tree.

If the visible graph contains cycles we remove them (pop them) according to some order. Next we show the actual order doesn't matter. We could have chosen to pop $(2,1)$ or $(4,5,3)$ first in the example above.

Lemma 4.3 The order in which cycles are popped is irrelevant, either (1) every order pops an infinite number of cycles or (2) every order pops the same finite set of cycles so the final tree is the same.