

CME 305: Discrete Mathematics and Algorithms

Instructor: Professor Amin Saberi (saberi@stanford.edu)

January 13, 2009

Lecture 2: Trees and Cayley's Theorem

A **tree** is a connected graph with no cycles. A **forest** is a graph where each connected component is a tree. A node in a forest with degree 1 is called a **leaf**.

Claim 1 (*simple properties of trees*)

- Every tree has at least two leaves (vertices of degree 1), assuming $n \geq 2$.
- The number of edges in a tree of size n is $n - 1$

Proof: The first property can be seen by starting a path at an arbitrary node, walking to any neighbor that hasn't been visited before. After at most n steps, we must reach a vertex v where there are no untraversed neighbors. Therefore, v must have only one neighbor and is a leaf, otherwise there is a cycle and the graph is not a tree. We can then traverse a path starting at v , and the vertex where the path stops must be a second leaf.

We can show the second property by induction. The base case $n = 1$ is trivial. Any tree of size $n \geq 2$ has at least one leaf. Removing a leaf results in a tree with one less node and one less edge. Therefore, a tree with n vertices has one more edge than a tree with $n - 1$ vertices. Applying this to the base case proves the result. ■

Graph and Tree Combinatorics

How many graphs are there with n vertices? Since there are $\binom{n}{2}$ distinct edges, there are $2^{\binom{n}{2}}$ graphs.

We call a graph **labeled** if the vertex ordering is important. How many labeled trees are there for a given number of vertices n ?

Theorem 1 *Cayley's Theorem (19th century)*

The number of labelled trees with n vertices is n^{n-2} .

The idea of our proof will be to use a bijective mapping between labeled trees and something easier to count. To prove Cayley's theorem, we will use *Prüfer codes*.

To build the Prüfer code of a tree $T(V, E)$, we use the following iterative procedure.

At step $t = 1, 2, \dots, n - 1$

- Let vertex i be the leaf with smallest label in T at step t , and vertex j its parent.
- Remove i from T .
- Let $A'_t = j$, and let $B_t = i$.
- Repeat until there are no edges.

We call the two $n - 1$ length strings A' and B *extended Prüfer code* of T . Note that this is just a permuted edge list of T .

By construction, the last entry of A' must be n , since n will never be the leaf with smallest label and there are always at least two leaves (claim 1). The string A obtained by erasing the last entry of A' and is called the Prüfer code of T .

Lemma 1 *We can construct B and A' using only A .*

Proof:

Given A , we construct A' by appending n to A . Let $A' = a_1a_2\dots a_{n-2}n$, we construct $B = b_1b_2\dots b_{n-1}$ as follows:

$\forall i, b_i$ is the smallest number that hasn't appeared in $\{b_1, \dots, b_{i-1}\} \cup \{a_i, \dots, a_{n-2}\}$. This can be seen from the way in which Prüfer codes are constructed by 'pruning' leaf from the tree. At step $t = i$, we have already removed nodes $\{b_1, \dots, b_{i-1}\}$ and it cannot be a parent in future steps (so it's not in $\{a_i, \dots, a_{n-2}\}$). All remaining nodes correspond to precisely the set of leaves, so b_i must be the smallest of these as stated. ■

Proposition 1 *The following hold for Prüfer codes:*

- *The number of Prüfer codes is n^{n-2} .*
- *$\forall v \in V$, the degree of v is the number of times it appears in the code plus one.*
- *Each Prüfer code yields exactly one tree.*

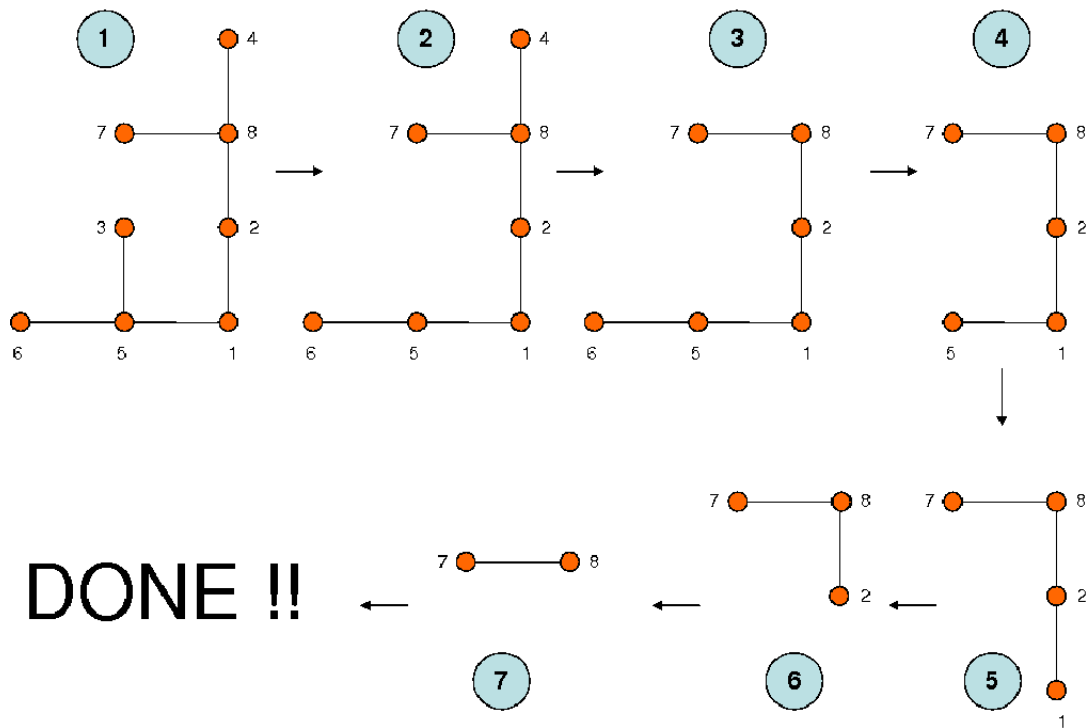
Proof:

The first statement is obvious from the Prüfer code representation: there are n choices at each of the $n - 2$ digits of the code. To see the second statement, simply note that the number of times it appears in the code is the number of "child" vertices it has, then add one for being a leaf at some point to account for the degree.

To prove the third statement, note that each tree produces a unique Prüfer code by the deterministic nature of their construction. Also, as shown in Lemma 1, we can uniquely construct an edge list from each Prüfer code. All that remains to be shown is that the edge list recovered must be a tree. This is easy to see, as at each stage in the reconstruction of B from Lemma 1, starting from the right (label n) we are adding nodes of degree one. Since we start with a tree, and adding a node of degree one to a tree produces another tree, the final graph must be a tree.

■

How to Create a Prufer Code of a given Tree



Evolution of the Prufer code

- In step 1, we remove node 3 and update $B = 3$ and $A = 5$
- In step 2, we remove node 4 and update $B = 34$ and $A = 58$
- In step 3, we remove node 3 and update $B = 346$ and $A = 585$
- In step 4, we remove node 4 and update $B = 3465$ and $A = 58512$
- In step 5, we remove node 3 and update $B = 34651$ and $A = 585127$
- In step 6, we remove node 4 and update $B = 346512$ and $A = 5851278$
- In step 7, we remove node 3 and update $B = 3465127$ and $A = 5851278$