Spanning Trees and Kirchoff's Matrix Theorem

MATH 3V03
Jeremy Lane

Abstract

The following is a note from MATH 3V03: Graph Theory at McMaster University which I taught in Fall, 2019. These notes are presented as-is and may contain errors. The textbook referred to in these notes is Pearls in Graph Theory by Ringel and Hartsfield.

1 Definition of spanning trees

Recall the following definitions from Pearls, Section 1.3.

Definition 1.1. A *spanning subgraph* of $G = (V, E)$ is a subgraph $H = (V', E')$ such that $V = V'$.

Definition 1.2. A *spanning tree* of a graph G is a spanning subgraph H that is a tree.

If G contains a spanning tree then it is connected. Conversely, we have Pearls, Theorem 1.3.6 (see the textbook for the proof),

Theorem 1.3. *If G is connected, then G contains a spanning tree.*

One problem of interest is counting the number of spanning trees in a given graph G. Denote the number of spanning trees in G by $s(G)$.

Example 1.4. $s(G) = 0$ if and only if G is disconnected.

Example 1.5 (Number of spanning trees in a cycle)**.** The number of spanning trees in C_n is

$$s(C_n) = n.$$

To see this, observe that deleting any edge from C_n yields a spanning subgraph that is isomorphic to P_{n-1}. In other words, it is a spanning tree. Since there are n possible ways to delete a single edge, there are n different spanning trees.

2 Finding spanning trees

This is discussed in Pearls, Section 7.1.

The proof of Theorem 1.3.6 in Pearls that every connected graph contains a spanning tree is also an algorithm for finding a spanning tree. It looks like this:

1. Start with $H = G$. If G is a tree, then it is a spanning tree so we are done.
2. Since it is not a tree, H contains at least one cycle. Find an edge e contained in a cycle in H and replace H with $H - e$.
3. If H is a tree then we are done. If it is not, repeat step 2.

Note that throughout this algorithm, H remains a connected spanning subgraph at every step.

In fact, this algorithm doesn't look so great because every time we do step 2 we have to look for a cycle in H. If H is small this is no big deal but if H is large it might take a while to find one. An alternate approach is the following:

1. Pick a starting edge e adjacent to vertices v_1 and v_2. Let T be the subgraph consisting of v_1, v_2 and e.
2. Check whether there exists a vertex v that is: not contained in T and adjacent to some vertex u in T.
3. If there is, then add v and the edge between v and u to the subgraph T and go back to step 2. Otherwise, T is a spanning tree, so we are done.

In this algorithm instead of looking for cycles we have to check for vertices adjacent to T. In this algorithm T is a tree at every step, but it isn't a spanning subgraph of G until the end of the algorithm.

3 Bridges and banks

Let $G = (V, E)$ be a finite simple graph. An edge $e \in E$ is a *bridge* if G is connected and $G - e$ is not connected. If e is a bridge then the connected components of $G - e$ are *banks*.

Lemma 3.1. *Let $G = (V, E)$ be a finite simple graph. If $e \in E$ is a bridge and H_1, H_2 are the banks, then $s(G) = s(H_1)s(H_2)$.*

Proof. Let T be a spanning tree of G. Since T is a connected spanning subgraph of G, it must contain the edge e. Otherwise it would be a connected subgraph of $G - e$, which would imply that it is contained in one of H_1 or H_2. This would contradict the assumption that T is a spanning subgraph. Observe that $T_1 = T \cap H_1$ is a spanning tree of H_1 and $T_2 = T \cap H_2$ is a spanning tree of H_2.

Conversely, every subgraph of the form $T = T_1 \cup e \cup T_2$ is a spanning tree of G. Thus the total number of spanning trees in G is equal to the number of spanning trees in H_1 times the number of spanning trees in H_2. □

4 The Deletion-Plus-Contraction Formula

The *contraction* of G by an edge $e \in E$ is the new graph obtained from G by first deleting e, then identifying the two vertices that were incident to e.

If G is simple, it is possible that G/e contains multiple edges. In what follows, we assume G is a finite simple graph and understand that we might be counting spanning trees in a multigraph when we write $s(G/e)$. Counting spanning trees in multigraphs works exactly as you would expect, so we won't say much about it here.

Theorem 4.1 (Deletion-Plus-Contraction Formula)**.** *Let e be an edge in a finite simple graph G. Then,*

$$s(G) = s(G - e) + s(G/e).$$

Proof. The set of all spanning trees of G can be partitioned into two disjoint subsets: spanning trees that contain e, and spanning trees that do not contain e. The number of spanning trees that do not contain e equals $s(G - e)$. The number of spanning trees that do contain e equals $s(G/e)$. □

One useful application of this theorem is the following: if $G - e$ is not connected, then $s(G - e) = 0$. Thus, when counting spanning trees, one can start by contracting all edges such that $G - e$ is disconnected. Sometimes when you contract an edge with this property, a new edge gains this property.

5 Cayley's Spanning Tree Formula

Theorem 5.1 (Cayley, 1889)**.** *The number of spanning trees in K_n is*

$$s(K_n) = n^{n-2}.$$

A slick proof of this theorem due to Prüfer is covered in Pearls, Section 5.2. One useful detail is that Cayley's theorem can be reformulated as counting the number of trees on n vertices labelled by $\{1, \ldots, n\}$:

Theorem 5.2. *The number of labelled trees of order n is n^{n-2}.*

There is also a nice formula for the number of spanning trees of a complete bipartite graph.

Theorem 5.3. *The number of spanning trees of $K_{n,m}$ is*

$$s(K_{n,m}) = m^{n-1}n^{m-1}.$$

Proofs of several cases of this theorem are given in Pearls, section 5.3. We won't cover that section of the book. One can prove this theorem in complete generality using the the Kirchoff Matrix Theorem in a manner similar to the proof of Cayley's theorem below.

6 The Kirchoff Matrix Theorem

Another way to prove Cayley's theorem is using the Kirchoff Matrix Theorem which we will now state.

Let $G = (V, E)$ be a finite simple graph. Fix a choice of enumeration of the vertices of G. Let A denote an adjacency matrix of G. Let D denote the diagonal matrix whose ith entry is $d_i = \deg(v_i)$.

Definition 6.1. The *Laplacian matrix* (or *Kirchoff matrix*) of G is $L = D - A$.

Observe that the column and row sums of L are all 0. It follows that the determinant of L is also 0.

Definition 6.2. The *Kirchoff minor*, M, is the $(n-1) \times (n-1)$ matrix obtained from $L = D - A$ by deleting the last row and column.

Theorem 6.3 (Kirchoff's Matrix Theorem)**.** *For any finite simple graph G with Kirchoff minor M,*

$$s(G) = \det(M).$$

In other words, we can compute the number of spanning trees in G simply by computing a determinant. This is sometimes quite useful as it allows us to prove theorems, as we will see in the next section. It also gives us a way to compute $s(G)$ from the adjacency matrix.

We won't concern ourselves with the proof of this theorem, which is a little complicated. The idea of the proof is to do induction on the number of vertices and edges, using the deletion-plus-contraction theorem as one of the main tools.

7 Proof of Cayley's Spanning Tree Formula using Kirchoff's Matrix Theorem

Fix arbitrary n. The Laplacian matrix of K_n is the $n \times n$ matrix

$$L = \begin{pmatrix} n-1 & -1 & -1 & \dots \\ -1 & n-1 & -1 & \dots \\ -1 & -1 & n-1 & \dots \\ \dots & \dots & \dots & \dots \end{pmatrix}$$

We want to compute the determinant of the $(n-1) \times (n-1)$ Kirchoff minor, obtained from L by deleting the last row and column. Call this matrix M.

If we add each of the rows $2, 3, \dots, n-1$ to the first row of M then the determinant is unchanged and the new matrix is

$$\begin{pmatrix} 1 & 1 & 1 & \dots \\ -1 & n-1 & -1 & \dots \\ -1 & -1 & n-1 & \dots \\ \dots & \dots & \dots & \dots \end{pmatrix}$$

Now add the first row to all the other rows, $2, 3, \dots, n-1$. This does not change the determinant either. The resulting matrix is

$$\begin{pmatrix} 1 & 1 & 1 & \dots \\ 0 & n & 0 & \dots \\ 0 & 0 & n & \dots \\ \dots & \dots & \dots & \dots \end{pmatrix}$$

This matrix is upper triangular so the determinant is the product of the diagonal entries, which is n^{n-2}. Thus

$$s(K_n) = \det(M) = n^{n-2}.$$

8 Minimum weight spanning trees

We cover Kruskal's algorithm for finding minimum weight spanning trees. See Pearls, Section 7.1.