Abstract

Spanning trees have been studied extensively and have many practical applications. From astronomers using spanning trees to find quasar superstructures, to biomedical engineers detecting actin fibers in cell images, tree enumeration remains an interesting problem. In this paper, we examine a simple, practical, algebraic approach for the enumeration of spanning trees in complete and non-complete graphs based on Cayley’s formula and Kirchoff’s Theorem.

1 Introduction

What does calculating the electrical coverage of an eastern European city, astronomy, and biomedical engineering have in common? Spanning trees. It was Otakar Boruvka, a Czech mathematician in 1926 who developed a method based on minimum spanning trees to construct an efficient electricity network for his native region of Moravia. Today, astronomers still use spanning trees to find quasar superstructures [7] and biomedical engineers to detect actin fibers in cell images [9].

While finding the minimum spanning tree has many practical applications, finding all spanning trees in a graph is equally interesting. For example, ensuring the existence of multiple alternative connection arrangements of all nodes (the spanning trees) in a telecommunication network is an indication of the network’s resilience and survivability against security or infrastructure compromises.

Spanning tree enumeration can be approached both algorithmically and mathematically. In computer science, the algorithmic enumeration of span-
ning trees has been studied for nearly 30 years [12, 5, 6, 10, 14, 11]. In mathematics, various methods have been studied [13] and here we examine an approach based on simple algebraic calculations.

2 Cayley’s Formula: Spanning Trees in Complete Graphs

A graph is *connected* if it has no parts that are isolated from each other, therefore a path exists between every pair of distinct vertices. A *complete* graph is a graph in which an edge joins every pair of distinct vertices [1]. A complete graph $K_n$ has $n$ vertices and $n(n − 1)/2$ edges.

Every connected graph has a spanning tree. A *spanning tree* $T(G_n)$ is a subgraph of graph $G$ with a subset of $n − 1$ edges that form an acyclic graph [15]. While any two spanning trees comprised of different sets of edges are considered to be different, any two spanning trees for a graph have the same number of edges. We use $\tau(G)$ to express the number of spanning trees of a graph $G$.

Computing the number of spanning trees may be obvious for small graphs but harder for those with a higher number of vertices.

Arthur Cayley (1821-1895), a British mathematician and lawyer who abandoned a lucrative legal career to hold a professorship in pure mathematics at the University of Cambridge [3], developed a formula that provides the number of spanning trees in a complete graph.

*Cayley’s formula* states that the number of spanning trees $\tau(K_n)$ of a complete graph $K_n$ is equal to $n^{n-2}$ when $n > 1$.

![Diagram](image)

Figure 1: (a) The complete graph $K_3$ (b) The spanning trees of $K_3$

By applying the formula, we can see for a graph $K_3$ (Figure 1a), there are $\tau(K_3) = 3^{3-2} = 3$ spanning trees (Figure 1b), and for graphs $K_4$ and
$K_5$ (Figure 2a) there are $\tau(K_4) = 16$ (Figure 2b) and $\tau(K_5) = 125$ spanning trees respectively.

![Figure 2: (a) The complete graph $K_4$ (b) The non-complete graph $G_5$](image)

The first explicit combinatorial proof of Cayley's formula is due to the German mathematician Heinz Prüfer (1896-1934) in 1918 using a Prüfer sequence or code. The code of a labeled tree is a unique sequence associated with the tree. The sequence for a tree on $n$ vertices has length $n-2$, and can be generated by an iterative algorithm. The idea behind Prüfer sequences is to provide a bijection between the set of labeled trees on $n$ vertices and the set of sequences of length $n-2$ on the labels 1 to $n$. The latter set has size $n^{n-2}$, so the existence of this bijection proves Cayley’s formula of $n^{n-2}$ labeled trees on $n$ vertices [8].

However, Cayley’s formula only works for complete graphs where the number of edges is always known and fixed. For a $K_3$ complete graph, the number of edges is always three, for $K_4$ the number of edges is always six and for $K_{100}$ the number of edges is always 4,950. In contrast, the number of edges in non-complete, connected graphs is not always known or fixed for a given set of vertices. For example, a non-complete graph with four vertices may have anywhere from three to five edges. Similarly, a non-complete graph with four vertices and five edges has eight spanning trees but a complete graph with also four vertices has sixteen spanning trees.

Therefore, to calculate the number of spanning trees in non-complete graphs, a method other than Cayley’s formula is required, one that also considers the graph’s number of edges in addition to the set of vertices.
3 Kirchhoff’s Theorem: Spanning Trees in Non-complete Graphs

The German physicist Gustav Kirchhoff (1824-1887) developed a theorem about counting the number of spanning trees in a graph, known as the matrix tree theorem. Kirchhoff is well known for his contributions to electrical engineering and physics. In electrical engineering, he worked on the conservation of charge and energy in electrical circuits and developed what is known today as Kirchhoff’s Circuit Laws [2, 16]. In physics, he worked on spectrum analysis and formulated Kirchhoff’s three laws of spectroscopy that describe the spectral composition of light emitted by incandescent objects [4].

The matrix tree theorem states that the number of unique spanning trees of a graph $G$ is equal to any cofactor of the degree matrix of $G$ minus the adjacency matrix of $G$ [15]. For an $n \times n$ matrix, the cofactor of $A_{ij}$ is defined as $A_{ij} = (-1)^{i+j} \text{det}(M_{ij})$, where $M_{ij}$ is the $(n-1) \times (n-1)$ sub-matrix of $A$ by deleting the $i$-th row and the $j$-th column of $A$. The degree matrix is a diagonal matrix which contains information about how many edges (the degree) are connected to each vertex. The adjacency matrix on the other hand is the graph representation as an $n \times n$ matrix $A=(a_{i,j})$. In matrix $A$, the entry $a_{i,j}$ is equal to "1" when it represents an edge connecting vertices $i$ and $j$ and equal to "0” if there is no edge connecting the two vertices.

Figure 3: A non-complete graph $G$ with four vertices.

**Example 1.** Graph $G$ (Figure 3) is sufficiently small to make it obvious even by visual observation that it contains three spanning trees. The following steps use $G$ as to exercise the matrix tree theorem.

**Step 1.** Generate the adjacency ($A_G$) and degree ($D_G$) matrices for $G$:

$$A_G = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \quad \text{and} \quad D_G = \begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$
Step 2. Take the difference between the degree matrix $D_G$ and the adjacency matrix $A_G$ of $G$:

\[
D_G - A_G = \begin{bmatrix}
2 & 0 & 0 & 0 \\
0 & 2 & 0 & 0 \\
0 & 0 & 3 & 0 \\
0 & 0 & 0 & 1 \\
\end{bmatrix} - \begin{bmatrix}
0 & 1 & 1 & 0 \\
1 & 0 & 1 & 0 \\
1 & 1 & 0 & 1 \\
0 & 0 & 1 & 0 \\
\end{bmatrix} = \begin{bmatrix}
2 & -1 & -1 & 0 \\
-1 & 2 & -1 & 0 \\
-1 & -1 & 3 & -1 \\
0 & 0 & -1 & 1 \\
\end{bmatrix} = C.
\]

Step 3. Following the earlier definition of the matrix tree theorem, computing any cofactor from the difference matrix $C$ will provide the number of spanning trees. Practically, this means we need to eliminate any $i$-th row and $j$-th column from the matrix to generate a minor matrix $(M_{ij})$ and then compute the determinant of that matrix. Taking the cofactor $C_{33}$:

\[
C_{33} = \begin{bmatrix}
2 & -1 & -1 & 0 \\
-1 & 2 & -1 & 0 \\
-1 & -1 & 3 & -1 \\
0 & 0 & -1 & 1 \\
\end{bmatrix} = \begin{vmatrix} 2 & -1 & 0 \\ -1 & 2 & 0 \\ 0 & 0 & 1 \end{vmatrix} = M_{33}.
\]

The general form of the determinant of a 3x3 matrix $B$ is:

\[
|B| = \begin{vmatrix} a_1 & b_1 & c_1 \\ a_2 & b_2 & c_2 \\ a_3 & b_3 & c_3 \end{vmatrix} = a_1b_2c_3 + a_2b_3c_1 + a_3b_1c_2 - (a_3b_2c_1 + a_1b_3c_2 + a_2b_1c_3).
\]

By applying this general form to compute the determinant of $M_{33}$ we obtain:

\[
|M_{33}| = \begin{vmatrix} 2 & -1 & 0 \\ -1 & 2 & 0 \\ 0 & 0 & 1 \end{vmatrix} = 4 + 0 + 0 - (0 + 0 + 1) = 4 - 1 = 3.
\]

Therefore, graph $G$ from Figure 2 has three spanning trees. Notice we could have taken any cofactor of $C$ and the answer would have been the same. For example, for $M_{22}$ and $M_{44}$:

\[
|M_{22}| = \begin{vmatrix} 2 & -1 & 0 \\ -1 & 3 & -1 \\ 0 & -1 & 1 \end{vmatrix} = 3 and |M_{44}| = \begin{vmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ 0 & 0 & -1 \end{vmatrix} = 3.
\]

Performing the computations described in these two steps are not complex but as the number of vertices increase, the required calculations of the higher order determinants becomes progressively more laborious and error-prone for manual calculations.
4 Tools

One way to avoid the laborious manual calculations is to use tools to perform the determinant calculations. Using a free open-source numerical computation software package like Octave (www.octave.org), the calculations become trivial. To find the relevant code and instructions for Octave for the presented example and two other examples, please visit: http://cs.hood.edu/~dimitoglou/SpanningTrees/

Acknowledgment I am thankful to Kimber Tysdal for her inquiry on the extent of the practical applications of spanning tree enumeration in computer science. My response was perhaps not very helpful to her but the question triggered my interest to study the problem from a mathematics perspective.

References


