We will prove Cayley's Formula by the Bijection Principle, finding a bijection between the set of trees with vertex set \( \{1, 2, \ldots, n\} \) and a set of sequences which has size \( n^{n-2} \). The sequence we attach to a tree is called its Prüfer sequence, which is defined by the following recursive algorithm.

**PRÜFER SEQUENCE.** The input is a tree \( T \) with \( n \geq 2 \) vertices, where the vertex set is a subset of the positive integers. The output is a sequence \( (p_1, p_2, \ldots, p_{n-2}) \) with \( n - 2 \) terms, all in the vertex set. The steps are:

1. If \( n = 2 \), return the empty sequence () and stop.
2. (To arrive here, we must have \( n \geq 3 \).) Let \( \ell \) be the smallest (in numerical order) of all the leaves of \( T \). Define the first term \( p_1 \) to be the unique vertex of \( T \) which is adjacent to \( \ell \).
3. Recursively call PRÜFER SEQUENCE to find the Prüfer sequence of the smaller tree \( T - \ell \), and let \( (p_2, \ldots, p_{n-2}) \) be this sequence.

In words, the algorithm progressively deletes the smallest leaf of whatever is remaining, and records in the sequence not the leaf itself but the vertex which was adjacent to it.

**Example 3.10.** Let \( T \) be the tree in Example 3.2. The smallest leaf is 1 which is adjacent to 5. So the first term of the Prüfer sequence is 5, and the rest is obtained by considering the tree \( T - 1 \):

![](image)

The smallest leaf in \( T - 1 \) is 2 which is adjacent to 4, so the next term of the Prüfer sequence is 4, and the rest is obtained by considering \( (T - 1) - 2 \). Continuing in this way, you can find the full Prüfer sequence of \( T \) (which should have \( 6 - 2 = 4 \) terms):

\[(5, 4, 3, 4)\]
Example 3.18. If you carry out a depth-first search on the graph $G$ in Example 3.14 with $v_1 = a$, and whenever you have a choice of vertices choose the one which is latest in the alphabet, then the order of the vertices is

$$a, c, z, y, v, e, w, x, d, b$$

and the spanning tree $T$ is

One useful feature of a DFS spanning tree is that it contains some information about all the edges of the graph.

Theorem 3.19. Let $T$ be a spanning tree of the graph $G$ obtained by applying the DFS algorithm, and let $v_1, v_2, \ldots, v_n$ be the resulting ordering of the vertices. If $v_i$ is adjacent to $v_j$ in $G$, with $i < j$, then $v_i$ lies on the path between $v_1$ and $v_j$ in $T$.

Proof*. Let $v_{i_1}, v_{i_2}, \ldots, v_{i_s}$ be the successive vertices of the path from $v_1$ to $v_i$, with $i_1 = 1$ and $i_s = i$; from the construction it is clear that $i_1 < i_2 < \cdots < i_s$. Similarly, let $v_{j_1}, v_{j_2}, \ldots, v_{j_t}$ be the successive vertices of the path from $v_1$ to $v_j$, with $j_1 = 1$, $j_t = j$, and $j_1 < j_2 < \cdots < j_t$. Let $k$ be maximal so that $i_k = j_k$; the vertex $v_{i_k}$ is the ‘latest common ancestor’ of $v_i$ and $v_j$, if you think of $T$ as a family tree of descendants of $v_1$. We want to prove that, given the adjacency of $v_i$ and $v_j$ in $G$, this latest common ancestor is $v_i$ itself, i.e. $k = s$. Suppose for a contradiction that $k < s$. It is impossible for $k$ to be $t$, because that would imply that $j < i$, so the two paths diverge and continue through the different vertices $v_{i_{k+1}}$ and $v_{j_{k+1}}$. From the fact that the descendant $v_i$ of $v_{i_{k+1}}$ was added to the tree before the descendant $v_j$ of $v_{j_{k+1}}$, we deduce that $i_{k+1} \leq i < j_{k+1}$. But then consider the pass through Step (3) of the algorithm which gave $v_{j_{k+1}}$ its name. At this point of the algorithm, the value of $m$ must have been $j_{k+1} - 1$, so $v_i$ was already in $T$;
Theorem 3.33 (Kirchhoff's Matrix–Tree Theorem). Let $M$ be the Laplacian matrix of a connected graph $G$ with vertex set $\{1, 2, \ldots, n\}$, where $n \geq 2$. Then for all $k, \ell \in \{1, 2, \ldots, n\}$, the $(k, \ell)$-cofactor $(-1)^{k+\ell} \det(M^{k,\ell})$ equals the number of spanning trees of $G$.

Before we give the proof, here are some examples of how to use the Matrix–Tree Theorem to count spanning trees.

Example 3.34. Let us apply the Matrix–Tree Theorem to find the number of spanning trees in the graph $G$ of Example 3.24, whose Laplacian matrix $M$ was found in Example 3.30. Since all the cofactors of $M$ are guaranteed to be the same, we may as well choose to delete the row and column with the most nonzero entries; the $(3,3)$-cofactor is

$$(-1)^{3+3} \det(M^{3,3}) = \det \begin{pmatrix} 2 & -1 & 0 & 0 \\ -1 & 2 & 0 & 0 \\ 0 & 0 & 2 & -1 \\ 0 & 0 & -1 & 2 \end{pmatrix}.$$ 

When a matrix has block-diagonal form like this, its determinant is the product of the determinants of the blocks. Since

$$\det \begin{pmatrix} 2 & -1 \\ -1 & 2 \end{pmatrix} = 2 \times 2 - (-1) \times (-1) = 3,$$

the $(3,3)$-cofactor of $M$ is $3 \times 3 = 9$. This of course agrees with the number of spanning trees found in Example 3.24.

Example 3.35. If $G$ is as in Example 3.25, the Laplacian matrix $M$ is

$$\begin{pmatrix} 3 & -1 & -1 & -1 \\ -1 & 2 & 0 & -1 \\ -1 & 0 & 2 & -1 \\ -1 & -1 & -1 & 3 \end{pmatrix}.$$

Its $(1,1)$-cofactor is

$$\det \begin{pmatrix} 2 & 0 & -1 \\ 0 & 2 & -1 \\ -1 & -1 & 3 \end{pmatrix} = 8.$$