Joyal's Proof of Cayley's Formula

Gyu Eun Lee and Doron Zeilberger

July 16, 2012

1 Abstract.

We introduce labeled trees via examples and ask how many of them exist on n vertices. The result, called Cayley's Formula, has a beautifully simple form. We give a combinatorial proof of Cayley's Formula first provided by André Joyal in his seminal article, that laid the foundation to the theory of species, given in

A. Joyal, *Une théorie combinatoire des séries formelles*, Advances in Mathematics **42** (1981), 1-82.

2 Building roads between cities.

Suppose we are civil engineers and are given the task of connecting n cities via roads. For this particular scenario, the names of the cities matter. By this is meant that if we need to connect New York, Boston, and Philadelphia via roads, then one possible configuration is a road from New York to Boston and a road from Boston to Philadelphia. Another possible configuration is a road from New York to Philadelphia and a road from Philadelphia to Boston. These two configurations are considered distinct.

This seems perfectly obvious, but it is an important distinction because the problem translates quite naturally into a graph theoretic problem. If we consider the cities to be vertices and roads between them to be edges, then the distinction between labeled cities and unlabeled cities becomes quite clear. In graph theory, the locations of the vertices in the plane are irrelevant, as a graph is characterized completely by its vertices and edges. (We are referring to regular elementary graph theory; some aspects of graph theory deal with embeddings of graphs on nonplanar surfaces.) Hence in an unlabeled, undirected graph with 3 vertices, any configuration with 2 edges is equivalent to any other configuration with 2 edges. If the graph is labeled, say the vertices are called 1, 2, 3, then an edge between 1 and 2 is distinct from an edge between 2 and 3.

Returning to our problem of connecting cities via roads, we are asked to connect the cities so that there is a way to reach any city starting at any city. In graph theoretic terminology, we are asked to find a connected graph on n vertices. One way that comes to mind is to build a road between every pair of cities. To do this, we need to build $\binom{n}{2}$ roads, which grows $\mathcal{O}(n^2)$. This is a valid but very expensive solution: for 100 cities we are looking at building 4950 roads. (Also, at least two roads are guaranteed to intersect each other, since K_5 is nonplanar. This may prove a further hassle for highway engineers.)

Is there a simpler way? Of course! For $n = 4$ we can get away with 3 roads. Call the cities A, B, C, D. We can build a road from A to B, then from B to C, then from C to D. A resident of A who has relatives in D isn't going to be particularly happy with this configuration, but the fact remains that going from A to D is at least possible via our roads.

In fact for n cities, $n-1$ roads will suffice to connect all the cities. The procedure to do so is clear from the case of $n=4$, and we will provide a proof shortly. Of course, you may have noticed that not all configurations of $n-1$ roads will connect n cities: take $n=5$, and call the cities A, B, C, D, E. Build a road between A and B, B and C, C and A, and D and E. Then there is no way to get from any of A, B, C to any of D, E. But if we build our roads carefully, then 4 roads will certainly suffice: build the roads (excuse the shorthand) AB, BC, CD, and DE.

Proposition 1. *The smallest number of roads required to connect n cities is $n-1$.*

Proof. We proceed by induction on n. For $n=1$ we have 1 city, and we don't need to build any roads since we only have 1 city. Now suppose $n-1$ roads suffice for n cities. Build such a configuration for n cities, and add one more city A. We simply have to build a road between A and any one of the other cities, and it is connected to all of them since the other cities are all connected. Hence n roads suffice for $n+1$ cities, and the claim is proved. □

3 Labeled trees.

In graph theory, a connected undirected graph on n vertices with $n-1$ edges like in our example is called a labeled tree, also known as a minimally connected graph for clear reasons. A tree is simply a connected graph that contains no cycles: starting at a vertex and following edges, it is impossible to return to the original vertex without repeating an edge.

Since we are learning combinatorics, a good question to ask might be: how many distinct labeled trees can we build on n vertices? By distinct labeled trees we mean this: consider the labeled trees on the 3 vertices A, B, C. One labeled tree, given by its edges, is $\{AB, BC\}$. Another, considered distinct from the first, is $\{AC, BC\}$. If we were considering unlabeled trees, then these two trees would be considered equivalent. Unlabeled trees on n vertices are considered equivalent whenever they can be put into the same shape, sort of like the idea of isomorphic labeled graphs.

Let us denote by u_n denote the number of unlabeled trees on n vertices, and by a_n the number of labeled trees on n vertices,.

n	u_n	a_n
1	1	1
2	1	1
3	1	3
4	2	16
5	3	125

You may verify these values yourself. A pattern is beginning to emerge for a_n: note that each value of a_n factorizes into some power. Even more explicitly, $a_1 = 1 = 1^{-1}$, $a_2 = 1 = 2^0$, $a_3 = 3 = 3^1$, $a_4 = 16 = 4^2$, $a_5 = 125 = 5^3$. It's an unexpectedly simple pattern! (Those of you that verified the values of a_n by counting would probably agree.) We may even venture to guess a general formula for a_n:

$$a_n = n^{n-2}.$$

The formula happens to be true, and is named Cayley's Formula after Arthur Cayley, who wrote about it in a note in 1889. The original result is due to Carl Wilhelm Borchardt, who discovered it in 1860. It's a particularly beautiful formula because of its simplicity and unexpectedness: rarely in combinatorics do we get such a simple formula for a nontrivial problem, and if you verified the values of a_n by hand you would certainly agree that this equation is not obvious.[1] As it turns out,

[1] The same problem for undirected graphs is unresolved, even though u_n is certainly much smaller than a_n.

there is a very elegant combinatorial proof given by André Joyal in 1980 for this formula, and the rest of this transcription will be dedicated to explaining it. First, however, we need to make an apparent digression, and discuss the representation of permutations.

4 Representing permutations.

We are all familiar with the idea of permutations: given a set $\{1, 2, ..., n\}$, a permutation of the n elements of this set is some sequence of all n elements in some order. What we may be less familiar with is the variety of ways to represent a permutation.

4.1 One-line notation.

The most familiar representation of a permutation is probably one-line notation. To illustrate, take the set $\{1, 2, 3, 4\}$. The permutations of this set in one-line notation are $1234, 1243, 1324,$ $1342, \ldots, 4312, 4321$. The meaning of this notation is quite clear.

4.2 Two-line notation and decomposition into cyclic permutations.

In group theory, however, this notation is frequently abandoned in favor of two-line notation, because permutations in group theory are studied as one-to-one functions from a set of n elements to itself and two-line notation provides a clean way to represent compositions of permutations. To illustrate two-line notation, let us take the permutation 3421 in one-line notation as an example. In two-line notation, we write the original set in order on the first line, and the permutation of the set on the second line, like so:

$$\begin{pmatrix} 1 & 2 & 3 & 4 \\ 3 & 4 & 1 & 2 \end{pmatrix}.$$

The meaning of this notation is pretty intuitive. 1 is swapped with 3, 2 is swapped with 4, 3 is swapped with 1, and 4 is swapped with 2. (Yes, the last two swaps were redundant; we'll get to that later.) This is basically the same information as one-line notation, and so far doesn't seem to offer any advantages.

But as mentioned previously, the advantage of two-line notation is that it allows us to depict permutations as compositions of functions, and here's how this is done. Taking the same permutation as before, we now break it into a composition of cyclic permutations as follows: start at 1. 1 goes to 3. 3 goes to 1, which is a repeat. So the first cyclic permutation is

$$\begin{pmatrix} 1 & 3 \\ 3 & 1 \end{pmatrix}.$$

Now we move on to 2. 2 goes to 4. 4 goes to 2, which is a repeat. So the second cyclic permutation is

$$\begin{pmatrix} 2 & 4 \\ 4 & 2 \end{pmatrix}.$$

Having accounted for all numbers, we may represent our permutation as

$$\begin{pmatrix} 1 & 2 & 3 & 4 \\ 3 & 4 & 1 & 2 \end{pmatrix} = \begin{pmatrix} 1 & 3 \\ 3 & 1 \end{pmatrix} \begin{pmatrix} 2 & 4 \\ 4 & 2 \end{pmatrix}.$$

The meaning of this is that the permutation on the left-hand side is a composition of two cyclic permutations: swapping 1 and 3, followed by swapping 2 and 4. We often abbreviate this notation, called cycle notation, by writing

$$(3412) = (13)(24).$$

The redundancy in our wordy explanation of two-line notation is now understandable as the result of the permutation being a composition of 2-cycles. Using the "zig-zag" algorithm sketched in the example, it should not be difficult to verify that

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ 9 & 5 & 2 & 1 & 4 & 7 & 6 & 3 & 8 \end{pmatrix} = \begin{pmatrix} 1 & 9 & 8 & 3 & 2 & 5 & 4 \\ 9 & 8 & 3 & 2 & 5 & 4 & 1 \end{pmatrix} \begin{pmatrix} 6 & 7 \\ 7 & 6 \end{pmatrix}.$$

One thing to keep in mind is that a permutation as defined in group theory is a one-to-one function from the set $\{1, 2, \ldots, n\}$ to $\{1, 2, \ldots, n\}$. It is a basic result in elementary group theory that permutations can always be decomposed into compositions of cyclic permutations. A short argument can be provided here: consider building a permutation from $\{1, 2, \ldots, n\}$ to $\{1, 2, \ldots, n\}$. Then every number must go to a different number since a permutation is one-to-one. Suppose in our permutation, $n-1$ numbers are assigned without any cycles. Then no number goes to itself, so the n-th number cannot go to itself, as it is already associated with a number. Then it is guaranteed to go to one of the $n-1$ remaining numbers. Starting at that number and performing our "zig-zag" algorithm, we will eventually reach the n-th number and cycle back. (This is not a formal proof, but the pigeonholing idea is carried over.)

4.3 Graph representations of functions and permutations.

That a permutation is a composition of cyclic permutations is important because we can represent functions as graphs, and this property is reflected in the structure of the graph of a permutation. We are concerned with graphs of functions from $\{1, 2, \ldots, n\}$ to $\{1, 2, \ldots, n\}$. We will describe the process of drawing the graph of such functions via an example for $n = 6$. Suppose a function is represented in two-line notation by

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 2 & 1 & 4 & 2 & 3 & 6 \end{pmatrix}$$

(At this point we hope that the meaning of this notation, while unintroduced, is clear.) We will represent this function as a directed graph on 5 vertices, labeled from 1 to 5. Draw the 5 vertices and label them. Since 1 goes to 2, draw a directed edge from 1 to 2. Since 2 goes to 1, draw a directed edge from 2 to 1. Continue this process for all vertices, and our set of directed edges will be $\{12, 21, 34, 42, 53, 66\}$. Note that 6 points to itself. This algorithm can be extended to all functions on $\{1, 2, \ldots, n\}$. One thing to notice about this graph is that every vertex has exactly one directed edge coming out of it, i.e. every vertex has outdegree 1. Of course, from the definition of a function this isn't at all surprising: every element in the domain of a function must associate with exactly one element in its range.

Since the algorithm works for arbitrary functions, it works for permutations as well. Try applying the algorithm to the permutations

$$\begin{pmatrix} 1 & 2 & 3 & 4 \\ 4 & 1 & 2 & 3 \end{pmatrix} \tag{1}$$

$$\begin{pmatrix} 1 & 2 & 3 & 4 \\ 1 & 2 & 3 & 4 \end{pmatrix} \tag{2}$$

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 3 & 5 & 6 & 2 & 4 & 1 \end{pmatrix} \tag{3}$$

Notice anything? The graphs of these permutations are composed of collections of cycles. (1) is a single 4-cycle, (2) is a collection of 4 1-cycles, and (3) is composed of 2 3-cycles. This shouldn't be surprising at all, since we already saw that any permutation can be decomposed into cyclic permutations, and it is easy to see that a cyclic permutation corresponds to a cycle in a directed graph. (Try decomposing each of these permutations into cyclic permutations and compare the results to the graphs.) As with arbitrary functions, the vertices in the graph of a permutation have outdegree one; additionally, they have indegree 1 as well. (Why?)

5 Joyal's proof of Cayley's Formula

We are now equipped with all of the tools to prove Cayley's Formula via Joyal's ingenious combinatorial argument.

Theorem 2 (Cayley's Formula)**.** *The number of labeled trees on n vertices, denoted a_n, is given by $a_n = n^{n-2}$.*

Proof. First we note that the formula given is equivalent to the formula $n^2 a_n = n^n$. This is useful because n^n counts something very familiar: the number of functions from $\{1, 2, \ldots, n\}$ to $\{1, 2, \ldots, n\}$. (You may be more familiar with the idea of the number of permutations of n elements with replacement.) Joyal's idea was to use the graph representation of functions to create a bijective transformation between functions and certain special types of graphs, which we describe here.

We observed that the graph of a function on $\{1, 2, \ldots, n\}$ is guaranteed to contain a cycle. We can also observe that there is no directed path from one cycle in a graph to another: if there were, then the outdegree of one of the vertices in the first cycle would be 2, but the outdegree of all vertices must be 1, contradiction. Hence we can think of functions as collections of cycles with trees coming out of the vertices in the cycles; better yet, we can think of them as cycles with the trees as the vertices! (If a vertex has no tree coming out of it, think of it as the trivial tree with 0 vertices coming out of it.) Now we are looking at functions as disconnected cycles of trees.

Now we ask what the number $n^2 a_n$ counts. a_n counts the number of labeled trees on n vertices. Suppose we pick 2 of the vertices, which may be the same vertex, and call them the roots. Then there are n choices for the first root, and n choices for the second root (which may be the same vertex as the first root). Hence $n^2 a_n$ counts the number of doubly rooted labeled trees. Remember that labeled trees are undirected graphs.

Now we seek to establish a bijection between the set of doubly rooted labeled trees on n vertices and the set of functions on n elements. We do this by showing that we can transform every function into to unique double rooted labeled tree, and every doubly rooted labeled tree into a unique function.

Joyal's algorithm goes as follows:

First we start with a function as the input. The following algorithm, illustrated via an example function for $n = 9$, transforms this function into a doubly rooted labeled tree.

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ 1 & 1 & 5 & 7 & 6 & 3 & 8 & 7 & 4 \end{pmatrix}$$

Draw the graph representation of this function. Its set of directed edges will be $\{11, 21, 35, 47, 56, 63, 78, 87, 94\}$. As we observed previously, a function is a collection of cycles with trees as its vertices. Ignoring vertices in trees, take just the vertices that are part of a cycle and arrange them as a function:

$$\begin{pmatrix} 1 & 3 & 5 & 6 & 7 & 8 \\ 1 & 5 & 6 & 3 & 8 & 7 \end{pmatrix}$$

Pick the first and last correspondences to be the roots A and B. In this case A is 1 and B is 7. Now build a new double rooted labeled tree with roots A and B as follows. Connect A and B via a path going through the vertices in the second line. In this case, the path is $15-56-63-38-81$. This path is called the skeleton of the doubly rooted labeled tree. Now we reinstate the trees: add every vertex in a tree from the original directed graph, and add the edges from these vertices to the labeled tree, removing the directions. The set of undirected edges in our final graph should be $\{12, 15, 38, 36, 47, 49, 56, 78\}$, and its roots are 1 and 7. Thus we have built a doubly rooted labeled tree from a function on n elements.

We now do this in reverse. To illustrate that the transformation is invertible, we will perform it on our newly created doubly rooted labeled tree, aiming to get back our original function as output. Draw the graph of the doubly rooted labeled tree on $\{1, 2, ..., 9\}$ with edges $\{12, 15, 38, 36, 47, 49, 56, 78\}$ and roots 1 and 7. Now, remember the skeleton from our first algorithm? It is the sequence of vertices from one root to another - in this case, the edges $15-56-63-38-81$, or equivalently the sequence $1-5-6-3-8-7$. Now write this sequence of vertices in two-line notation, ordered on the first line and as it stands on the second line. You should get the familiar-seeming permutation

$$\begin{pmatrix} 1 & 3 & 5 & 6 & 7 & 8 \\ 1 & 5 & 6 & 3 & 8 & 7 \end{pmatrix}$$

Now generate the graph of this permutation: draw a directed edge from 1 to 1, from 3 to 5, and so on until all vertices are accounted for. This graph should start looking familiar: it is subgraph of our original function containing just the cycles and without the trees. What remains is to reinstate the trees: if vertex a is not part of the skeleton, it is part of a tree. Add all the vertices that are not in the skeleton to the graph, and add the edges coming into and out of these vertices. If b is in the skeleton and the undirected edge ab is in the doubly rooted labeled tree, add the directed edge from a to b to the graph of our function. The property that every vertex has outdegree exactly 1 will determine the rest of the edges. Our final set of directed edges will be $\{11, 21, 35, 47, 56, 63, 78, 87, 94\}$. From this we can get back the function, which in two-line notation reads

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ 1 & 1 & 5 & 7 & 6 & 3 & 8 & 7 & 4 \end{pmatrix}$$

as desired. Hence our transformation is invertible and we have a bijective transformation from the doubly rooted labeled trees on n vertices to the functions on n elements, meaning the sizes of these two sets are equivalent. This proves the claim that $n^2 a_n = n^n$, and the proof is complete. $\square$

6 Closing remarks.

The proof given was informal but can be easily made more formal, if desired. You may wish to try the algorithm on many other functions and doubly rooted labeled trees. If you want to test your mastery, try out the "quiz"

`http://www.math.rutgers.edu/~zeilberg/AJ12q.pdf`

and to check your answer

`http://www.math.rutgers.edu/~zeilberg/AJ12.pdf`

This paper was written by the first author, who transcribed (and improved) a guest lecture by the second author at Brian Nakamura's Rutgers University Summer 2012 undergraduate class on Combinatorics delivered on July 16, 2012.

[Added Aug. 17, 2012: we thank Octavio Agustin for corrections on an earlier version]