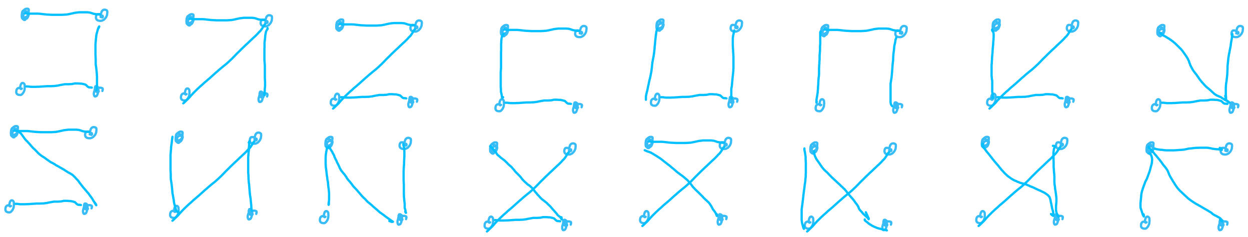
Math 38 – Graph Theory

Counting spanning trees

Nadia Lafrenière

04/24/2020

Question: How many spanning trees does the complete graph with n vertices have?



For the complete graph, there is an easy way of answering: This is the total number of trees with n vertices, as they are all subgraphs of the complete graph. Hence, it is n^{n-2}.

However, the following question is much harder:

Question: Given any graph G (simple or not), how many spanning trees are subgraphs of it?

Of course, finding a closed formula to count the number of spanning trees would be too much to ask for trees that don't have a specific structure.

We will in turn see an algorithm to answer this question easily.

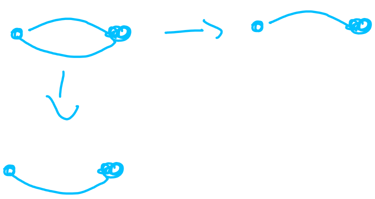
Proposition

There are as many spanning trees in a graph G as in the graph obtained from G by deleting all its loops.

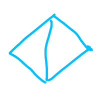
Proof

Loops cannot belong to any tree, as they are cycles. So deleting them won't remove any subtree.

However, the same cannot be said of graphs with multiple edges, as shown in the example below:



Example: Count the number of subgraphs of the kite ($K_4 - e$)



There are 4 trees passing through the outer cycle (of length 4). As for trees passing through the diagonal, we need to choose one edge from each triangle.

For each triangle, there are two edges to choose from, so there are in total 2*2=4 trees passing the diagonal, and 4 not using it. So there are 8 spanning trees in the kite.

To be able to count the number of trees, we distinguised two cases. These two cases span all the possibilities: either we use one specific edge or we don't use it. Of course, we cannot be in both situations. Combinatorially speaking, that means the total number of spanning trees is

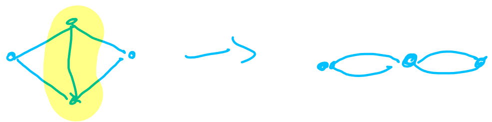
#(spanning trees using that edge) + #(spanning trees not using it).

If the latter seems easy to count in general, the former needs the introduction of the following operation.

In a graph G, contraction of edge e=uv is the replacement of both vertices u and v by a single vertex, by keeping all the edges incident to it, except e. The resulting graph, G·e, has one fewer edge than G, and one fewer vertex.

Example

In the kite, the contraction of the central edge gives the following:



Proposition

The number of spanning trees of G, noted $\tau(G)$, satisfies, for any single edge e, $\tau(G)=\tau(G-e)+\tau(G\cdot e)$.

Proof

We already noted above that the total number of spanning trees is the sum of the trees with and without edge e. The thing we need to prove is that $\tau(G\cdot e)$ is the number of spanning trees using edge e.

Start with T a spanning tree of G·e; T is connected to the new vertex created from the contraction of e, and has one fewer edge. Replacing back that vertex with the edge e (and distributing the edges among the two vertices like in the original graph) gives a spanning tree of G using e.

Also, from any spanning tree of G using e, we get a spanning tree of G·e by contracting vertex G (i.e. the spanning graph is still connected and still has no cycle).

This proposition will be the key to count, recursively, the number of spanning trees. To do so, we will also need the following:

Proposition

If G has no loop and does not have cycles of length at least 3, its number of spanning trees is the multiplicities of the edges.

Proof

Since G has no loops nor cycles of length at least 3, all the cycles have length 2, i.e. they are multiple edges. At most one of them can appear in a given spanning tree. But also, at least one of them must appear: otherwise the graph would be disconnected. This is because these edges are all not part of a cycle that uses other edges. Hence, we have to pick exactly one edge per pair of endpoint. These choices all being independent, we multiply their numbers.

Corollary

If there are k edges $\{e_1, e_2, \dots, e_k\}$ between endpoints u and v in G, the number of spanning trees of G is given by

$$\tau(G - \{e_1, e_2, \dots, e_k\}) + k\,\tau(G \cdot e)$$

where $G \cdot e$ is the graph obtained by merging u and v and deleting $\{e_1, e_2, \dots, e_k\}$.

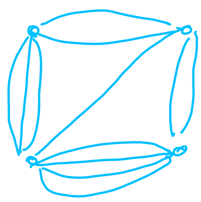
That yields an algorithm to count the number of spanning trees in G:

- If G is disconnected, it has no spanning tree; if G has a single vertex, it has only one spanning tree.
- Delete all loops in G.
- If G has no cycles of length at least 3:
  - The number of spanning trees is the product of the multiplicities of edges.
- Otherwise, choose a (multiple) edge e with multiplicity k, that is in a cycle of length at least 3. The number of spanning trees is $\tau(G-e)+k\,\tau(G \cdot e)$.

In the last step, both $G-e$ has fewer cycles than G, and $G \cdot e$ has shorter cycles. That means that the algorithm eventually terminates.

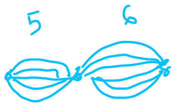
Example

We count the spanning trees in the graph below:

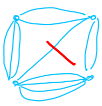
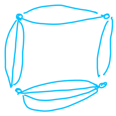


As we did earlier with the kite, we consider deleting or contracting the central edge.

Since it has multiplicity 1, the number of spanning trees can be counted in this way

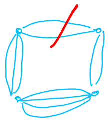
Contraction 

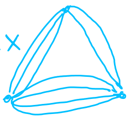
30 spanning trees using that edge.

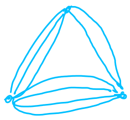
Deletion  ? spanning trees of 

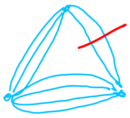
Total number of spanning trees

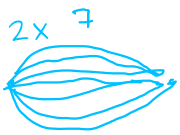
30
+1x(24+
2x(12+2x7))
= 106 spanning trees

Deletion  No cycle of length ≥ 3
There are 2x3x4=24 spanning trees not using top and diagonal edges.

Contraction 2x  We need to count the number of spanning trees of the multigraph on the left, and multiply it by 2.

? spanning trees of 

Deletion  There are 3x4=12 spanning trees not using the right edge.

Contraction 2x 

For each edge on the right, there are 7 edges in the contracted graph.

That process works for small trees, but the recursive procedure makes it very long to do for large connected graphs. We will see next class a theorem to make this computation efficient.

Reference: Douglas B. West. Introduction to graph theory, 2nd edition, 2001. Section 2.2