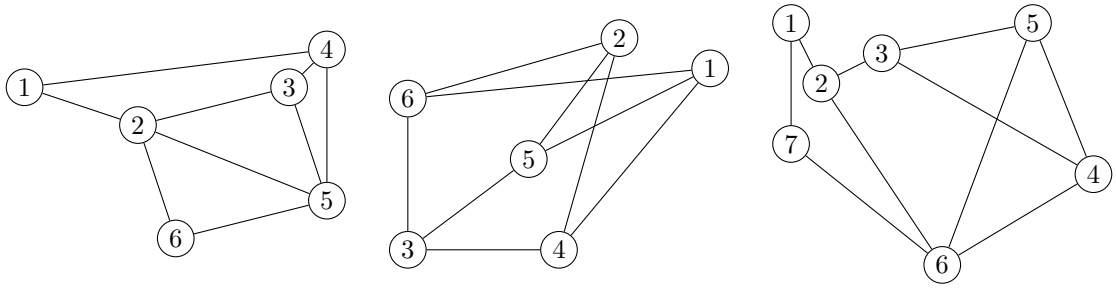
Trees and Tree Encodings

Berkeley Math Tournament

January 22, 2018

Introduction:

Today, we are going to be looking at a special class of graph theory called trees. These structures are an important discipline in mathematics and have wide applications in fields such as computer science and chemistry. Conceptually, a graph H is simply a collection of a set of vertices (denoted $V(H)$) and a set of edges (denoted $E(H)$) that connect them. In this power round, we will focus on *labeled graphs* which have distinct integers 1 to n (where n is the number of vertices) associated with each vertex. Examples are shown below:



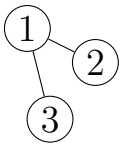
In the first example above $\{1, 2, 3, 4, 5, 6\}$ is the vertex set $V(H)$, and $\{(1, 2), (2, 3), (2, 6), (3, 5), (1, 4), (4, 5), (5, 6), (2, 5)\}$ is the edge set $E(H)$

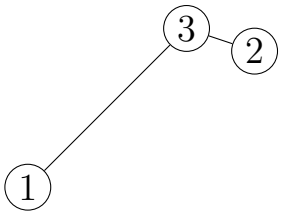
Definition. *Two labeled graphs, G and H are **isomorphic** if one can find a one-to-one and onto mapping f between the vertex sets of G and H such that if $(u, v) \in E(G)$ then $(f(u), f(v)) \in E(H)$ and the mapping is **label preserving** (i.e u and $f(u)$ have the same label).*

A simple way to check if two graphs are isomorphic is to see if you can move the vertices of one on top of the other such that the resulting graph preserves the labels and the edge structure. For instance, consider the examples below:

Ex 1.

Ex 2.



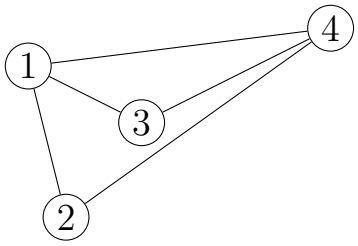


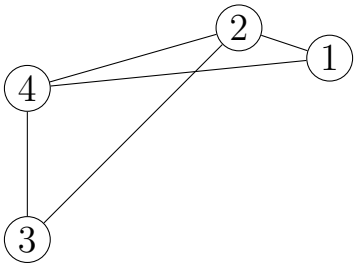
The first example is isomorphic since the graph contains the edges $(1, 3)$ and $(1, 2)$. However the second graph, while structurally the same, is not *label preserving*. In this case, 1 is not in the "middle" of the vertices 2 and 3 so it cannot be isomorphic.

Problem 1

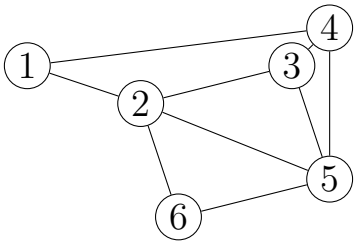
Determine if the following labeled graphs are isomorphic:

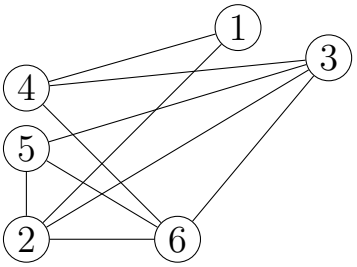
A.



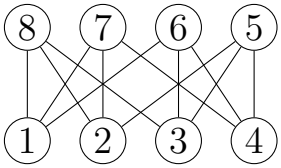


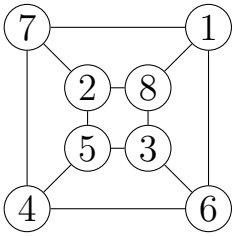
B.





C.





Trees and Tree Facts

Before we get to trees, we need to present a couple of definitions.

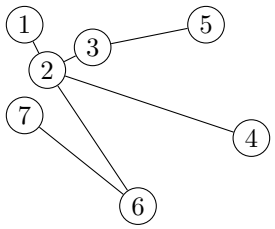
Definition. *Two vertices v_1, v_2 are **neighbors** if there exists an edge connecting them.*

Definition. *A **path** on n vertices is an ordered sequence of **distinct** vertices $v_0 ... v_n$ such that for any $i < n$ there exists an edge between v_i and v_{i+1}. (ie there is an edge between v_1 and v_2, v_2 and v_3...etc.).*

For instance the sequence of vertices corresponding to the labels $\{1, 2, 6, 5, 3\}$ in the first graph of problem $1B$ is a path, but the sequences $\{1, 3, 4, 5\}$ and $\{1, 4, 3, 2, 5, 3\}$ are not.

Problem 2

Find the longest path in the following graph



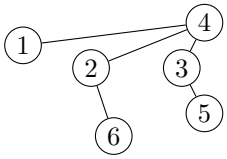
Definition. *A **cycle** on n vertices is a path $v_1 ... v_n$ such that there exists an edge between the starting vertex v_1 and v_n. Cycles must consist of at least 3 vertices.*

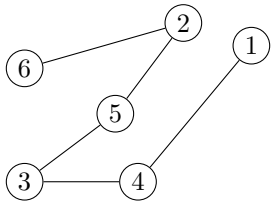
An example of a cycle in the first graph of problem $1B$ is the sequence of vertices corresponding to the labels $\{2, 3, 5\}$. However, the sequence $\{1, 2, 3, 5, 2\}$ is not a cycle since the vertex label 2 is repeated.

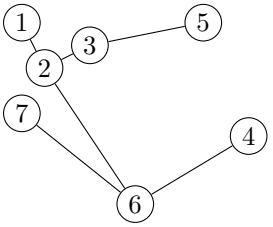
Definition. *A graph is **connected** if there exists a path between any two vertices in the graph*

Definition. *A **tree** is a graph T such that for any $u, v \in V(T)$, there exists exactly one path from u to v.*

Example. *Examples of trees can be seen below:*







Problem 3

Prove that

(***a***.) All trees are connected.

(***b***.) No tree contains a cycle.

Definition. *The **degree** of a vertex is the number of distinct neighbors it has. For instance, in the first tree above, the degree of vertex 4 is 3 and the degree of vertex 2 is 2.*

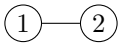
Definition. *A **leaf** (pl. leaves) of a tree is a vertex of degree 1. For instance, the leaves on the second graph above are vertices 6 and 1.*

Problem 4

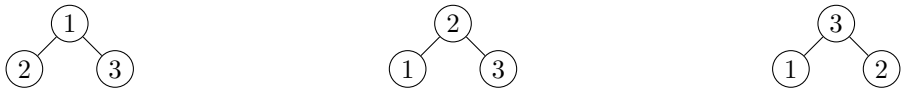
Prove that every tree contains at least two vertices of degree 1 (also called **leaves**).

Counting Labeled Trees

Now that we have the definitions, we want to be able to count all of the labeled trees on n vertices where we treat isomorphic trees the same. For instance, the number of labeled trees on 1 vertex is 1 since there is only 1 vertex. The number of labeled trees on two vertices is also 1, since it is just the edge below:



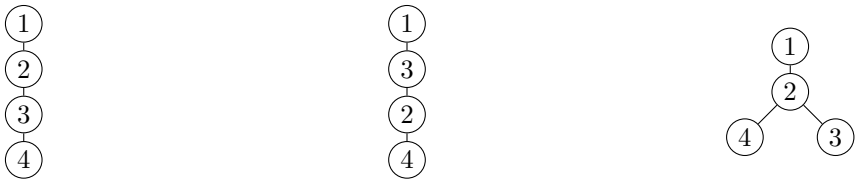
Likewise, the labeled trees up to isomorphism on 3 vertices are listed below:



We note that the label preserving distinction is important when considering labeled trees. These trees are all structurally similar (i.e all have a vertex connected to two others), but because there are different labels in the 'middle' they are counted as different. However, remember that two trees with different structure are also counted as different.

Problem 5

There are 16 distinct labeled trees on 4 vertices. Draw them out! Three trees have been given to you (you do not have to draw these out).



Prüfer Codes

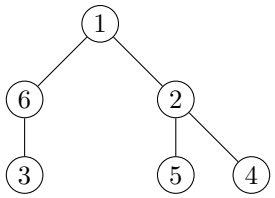
Remember how we proved that every tree must have at least 2 leaves? Well, Prüfer was able to use that to develop a recursive algorithm that equates a labeled tree on n vertices with a sequence of integers.

Algorithm. *Given a labeled tree T.*

1. *Find the value of the greatest labeled leaf.*
2. *Write down the integer label of the leaf's one neighbor.*
3. *Remove the leaf and the edge that goes with it.*
4. *Repeat until you reaches a tree with two vertices and a single edge between them.*

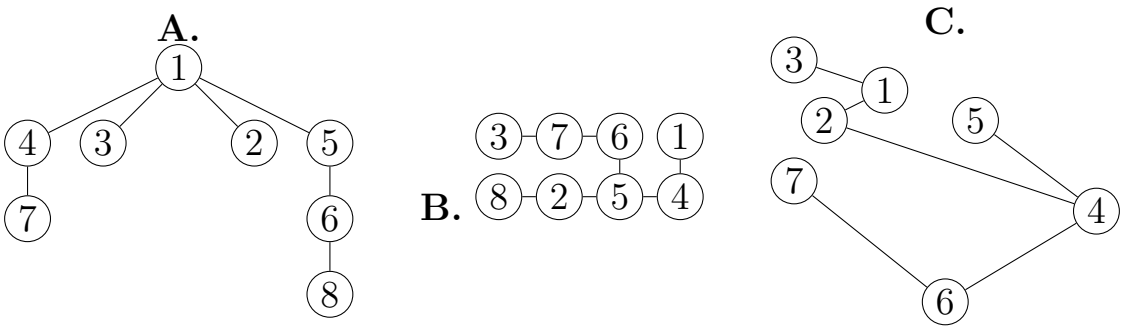
The sequence of written labels is the Prüfer code of the tree.

Example. *In the below example, the Prüfer code is* ***{2, 2, 6, 1}***



Problem 6

Compute the Prüfer codes for the following trees:



Problem 7

Prove that once the algorithm terminates, the remaining tree must contain a vertex with the label 1.

Problem 8

Given a tree on n vertices, determine (with proof) how many numbers are in that tree's Prüfer code.

Problem 9

Find an prove a graph and label structure on n vertices that has a constant Prüfer code.

Problem 10

Let j_k be the number of times the integer k ($\leq n$) appears in the Prüfer code. Prove that the degree of the node with the k label is $j_k + 1$

Problem 11

Draw out the trees corresponding to the following Prüfer codes.

(***a***.) $[1, 1, 3, 2, 4, 5]$

(***b***.) $[1, 1, 2, 2, 1, 1, 2, 2]$

(***c***.) $[6, 5, 4, 3, 1, 1, 7, 3, 1, 10]$

Problem 12

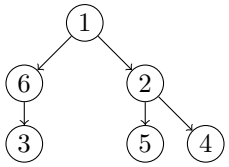
(***a***.) Find and prove a deterministic algorithm that takes a Prüfer sequence and produces the tree corresponding to the Prüfer code. This will establish a 1-1 correspondence between the set of Prüfer codes and the set of labeled trees.

(***b***.) Prove that the number of labeled trees on n vertices is n^{n-2}. This is known as Cayley's theorem.

Fleiner Codes

We now turn to a more recent paper that provided another proof of Cayley's theorem by constructing another graph encoding. This encoding takes advantage of another fact about trees called edge orientation. Because a tree has no cycles, we can orient edges away from a specific vertex. In our case, we will **always orient edges away from the "1" vertex**. An example is shown below:

Example. *In this example, all of the edges are oriented away from the "1" vertex.*



Because of this orientation, we can now define a function on the edges.

Definition. *Given an edge $e \in E(G)$, we let $c(e)$ be the label of the vertex at the tail of e. For example in the graph above, $c(\{1,2\}) = 1$ since the edge is oriented away from 1 and towards 2.*

Algorithm. *Given a labeled tree T with n vertices.*

1. *Start at the vertex with label 1.*
2. *Set $F = \{\}$ (make it an empty sequence). This will end up becoming our Fleiner Code*
3. *Repeat the following loop from $i = 2$ up to n.*
   (a) *Traverse the path from label 1 to label i*
   (b) *If you traverse an edge e that you have never traversed before, add $c(e)$ to F*
4. *After running the loop, the sequence F is our* ***Fleiner Code***

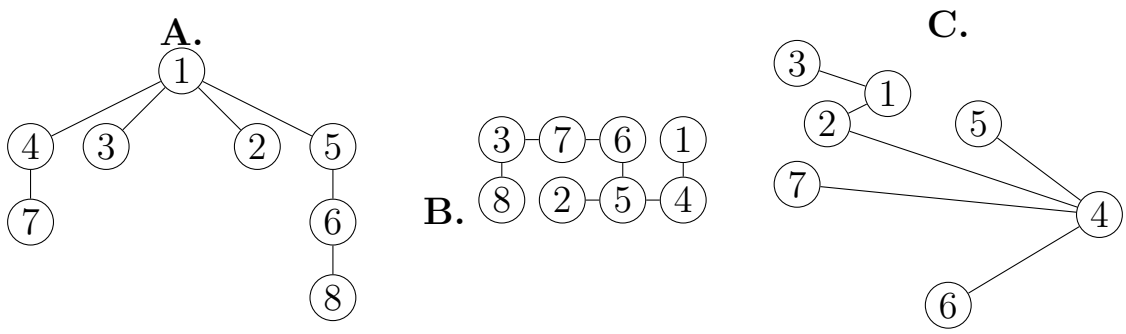
Example. *In the above tree example, the Fleiner code is {1, 1, 6, 2, 2}*

Problem 13

Find the Fleiner Codes of the following graphs:

Problem 14

Prove that the first number of any tree's Fleiner Code must be 1.



Problem 15

Prove that the reverse of a labeled tree T's Prüfer code is the last $n-2$ digits of that tree's Fleiner code.

Problem 16

Find an prove a formula for the number of labeled trees on n vertices such that the distance between vertices u and v with respective labels 1 and 2 is k. Here distance is defined to be the number of edges in the path from u to v. The formula should be in terms of n and k only.

Conclusion

Trees are one of the most interesting aspects of mathematics. Yet, the mathematics that was done in this power round is not only an integral part of Graph Theory, but also useful in real-world computer science applications!