

Solutions to In-Class Problems — Week 8, Wed

Problem 1. (a) Prove that at a party where there are at least two people, there are two people who know the same number of people there. Assume that “knowing” is a two-way relationship and that knowing yourself does not count.

Solution. Since there are n people, the number of other people a person could know is between 0 and $n - 1$ inclusive. However, both 0 and $n - 1$ cannot both occur. If there is a person who knows nobody, then there cannot be a person who knows everybody. As a result, there are only $n - 1$ choices for how many people a person knows. By the pigeonhole principle, since we have n people and $n - 1$ slots to put them in, two must end up in the same slot. ■

(b) Restate the problem from part (a) as a graph theorem.

Solution. A simple graph with n nodes must have at least two nodes with the same degree! ■

Problem 2. Use the generalized pigeonhole principle to prove the following statement:

Among any set of 150 natural numbers, there must be three numbers, a , b , and c , such that all of the pairwise differences, $(a - b)$, $(a - c)$, $(b - c)$, are multiples of 70.

(a) What are the pigeons?

Solution. The pigeons are The 150 numbers ■

(b) What are the holes?

Solution. the set of numbers $\{0, \dots, 69\}$ ■

(c) What is the function mapping the pigeons to the holes?

Solution. By taking their value mod 70 ■

(d) Carefully prove the statement using the pigeonhole principle.

Solution. With the pigeons as the 150 numbers, and the holes as the set of natural numbers mod 70, we see that there is some hole with $\lceil \frac{150}{70} \rceil = 3$ pigeons, that is, three numbers a, b, c that are equivalent modulo 70. Then their differences are equivalent to 0 modulo 70, therefore the differences are all divisible by 70. ■

Problem 3. Counting Numbered Trees.

A *numbered tree* is a tree whose vertex set is $\{1, 2, \dots, n\}$ for some $n \geq 2$. We define the *code* of the numbered tree to be a sequence of $n - 2$ integers from 1 to n obtained by the following recursive process:

If $n = 2$, stop—the code is the empty sequence. Otherwise, write down the *father* of the largest leaf, delete this *leaf*, and continue the process on the resulting smaller tree.

For example, the codes of a couple of numbered trees are shown in Figure (1).

(a) Describe a procedure for reconstructing a numbered tree from its code.

Solution. The key observation is that, given a code of length $n - 2$, the numbers between 1 and n which *do not appear* in the code must be leaves of the tree. Hence, the largest missing number is a leaf attached to the first number of the code. The rest of the tree can now be reconstructed by deleting the first number in the code, henceforth ignoring the largest leaf, and proceeding recursively on the rest of the code. (We're using the obvious fact that what's left after deleting a leaf from a tree is another tree.)

More precisely, the reconstruction procedure applies to any finite tree whose vertex set is totally ordered. The procedure takes *two* parameters: the vertex set, V , and a length $|V| - 2$ "code" sequence, S , of elements in V . If l is the largest element in V which does not appear in S , and f is the first element of S , then the reconstructed tree is obtained by adding edge (l, f) to the tree reconstructed by calling the procedure recursively with first argument $V - \{l\}$ and second argument equal to the code obtained by erasing the initial f from S . The procedure terminates when $|V| = 2$, returning the edge between the two numbers in V .

To justify the key observation, note that any vertex that gets deleted by the process and was not a leaf to begin with, must have been the father of a previously deleted leaf, which means it would appear in the code. So the missing integers must have been leaves to begin with or must be one of

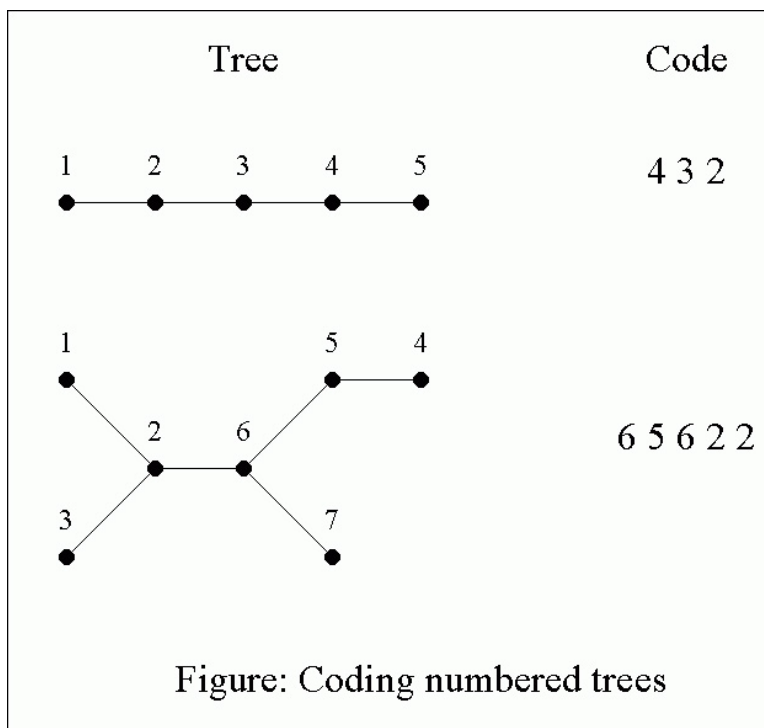


Figure 1: Coding Numbered Trees

the two undeleted vertices left when the coding process terminates. But by the end of the process the two remaining vertices are leaves, and if they weren't leaves to begin with, they must have become leaves by having their sons deleted, which means they would not have been missing from the code. So the two vertices remaining at the end must also have been leaves of the original tree. ■

(b) How many numbered trees with n vertices are there?

Solution. There are exactly as many n -vertex numbered trees as there are length $n - 2$ sequences of numbers between 1 and n . So there are n^{n-2} numbered trees.

The reason there are the same number is that the tree reconstruction procedure takes any such length $n - 2$ sequence and returns an n -vertex numbered tree whose code it is. But the explanation above of why the recursive reconstruction procedure is correct actually shows more: the reconstructed tree is the *unique* one with that code.

Now since every such length $n - 2$ sequence is the code of some n -vertex numbered tree, the coding procedure defines a surjection. Since the tree with that code is unique, the coding coding procedure also defines an injection. So the tree-coding function is a bijection. Therefore the number of numbered trees with n vertices is the same as the number of length $n - 2$ sequences of integers between 1 and n . ■