Tutorial 7

1 Comparing Infinities

In this section we will show that the cardinality of real numbers in the range $(0, 1)$ is the same as the cardinality of the set of all positive real numbers $R(0, \infty)$.

Theorem 1

$$|R(0,1)| = |R(0,\infty)|$$

Proof. We must prove that there exists a bijection $f : R(0,\infty) \mapsto R(0,1)$. Consider the function defined by $f(x) = e^{-1/x}$. This function has an inverse, $f^{-1}(x) = -\frac{1}{\ln x}$.

To prove that f is a bijection, we must show that f is a surjection and an injection. The function f is a surjection because for all $y \in R(0,1)$, there exists a $x \in R(0,\infty)$ such that $f(x) = y$; namely, $x = f^{-1}(y)$. The function f is also an injection. Suppose that $f(x) = f(x')$, then $f^{-1}(f(x)) = f^{-1}(f(x'))$, and so $x = x'$. ■

2 Induction on an Infinite Union

The purpose of this section is to point out a pitfall involving induction and infinite unions of sets. On your homework, you must prove that the union of a countable number of countable sets is a countable set. Avoid this pitfall on that homework problem! To illustrate the difficulty, we will give a false proof of a similar claim.

False Theorem 1 *The union of a countable number of finite sets is a finite set.*

As a counterexample, suppose that $S_i = \{i\}$ for all $i \geq 1$. Certainly, each S_i is finite! But then $\cup_{i=1}^{\infty} S_i$ is the set of natural numbers, which is countably infinite.

Proof Let $S_1, S_2, S_3, \ldots$ be finite sets. Let S be their union:

$$S = \bigcup_{i=1}^{\infty} S_i$$

Our goal is to prove that S is finite.

The proof is by induction. Let $P(n)$ be predicate that $\cup_{i=1}^{n} S_i$ is finite. In the base case, $P(1)$ is true because S_1 is finite. In the inductive step, assume that $\cup_{i=1}^{n} S_i$ is finite to prove that $\cup_{i=1}^{n+1} S_i$ is finite. The latter set is finite, because it is the union of $\cup_{i=1}^{n} S_i$ and S_{n+1}, both of which are finite. We conclude that that S is finite. ■

The problem here is in the final statement. The induction proof shows that the union of any number of finite sets is finite; the union of 7 finite sets is finite, the union of 10^{23} finite sets is finite, etc. But this does not imply that the union of a countable infinity of finite sets is finite!

3 A Stronger Pigeonhole Principle

The *Strong Pigeonhole Principle.* If $n(r-1)+1$ pigeons are placed in n holes then there is at least one hole with r or more pigeons in it.

Proof. By contradiction. Assume for the purpose of a contradiction that there is a way of placing the pigeons in the holes so that all the holes have $r-1$ or fewer pigeons. Then the total number of pigeons is at most $n(r-1)$. This is a contradiction since we assumed that there are at least $n(r-1)+1$ pigeons. ∎

An equivalent way of stating the principle is: If n pigeons are placed in r holes, then there is at least one hole with $\lceil n/r \rceil$ pigeons in it.

As a simple application, we will prove that there are at least 200 Americans with *exactly* the same number of hairs on their heads. First we estimate that the number of hairs on any person's head is at most a million. There are over 200 million Americans. To apply the strong pigeonhole principle, think of Americans as pigeons, and that there is a hole for each unique number of possible hairs. That is all Americans with the same number of hairs go into the same hole. Now by the pigeon hole principle there must be some hole that gets at least $200,000,000/1000,000 = 200$ pigeons. That is there are at least 200 Americans with exactly the same number of hairs on their heads.

As a second example, consider the following problem: we are to number the squares of a chessboard from 1 to 64. Is it possible to do this so that the difference between the numbers of any pair of adjacent squares (horizontally, vertically or diagonally) is no more than 8 ?

The answer is NO. Consider the square numbered 1 and the square numbered 64. We know that no matter where they are placed, there is a path of length at most 7 between them (i.e., in at most 7 hops to adjacent squares, we can go from the 1-square to the 64-square). To go from 1 to 64, there is a difference of 63 to be covered in 7 hops. Let us think of this difference as 63 pigeons and they have to each be placed in one of 7 holes. The holes represent which hop the step is taken in. Now by the strong PP there must be a hole with at least $63/7 = 9$ pigeons. This means that there must be a hop which covers a difference of 9 or greater. That is there are a pair of adjacent squares that differ by at least 9.

To see a more involved application, let us recall the definitions of a chain and an antichain in a DAG (directed acyclic graph).

A *chain* in a DAG is a sequence of vertices $a_1, a_2, \ldots, a_m$ such that there is a directed edge from a_1 to a_2, a directed edge from a_2 to $a_3, \ldots$, and a directed edge from a_{m-1} to a_m.

An *antichain* in a DAG is a set of vertices $b_1, b_2, \ldots b_m$ such that there is no directed path between any pair of distinct vertices in the set.

Recall also that if the longest chain in a DAG with n vertices is of length t, then the n vertices can be partitioned into t antichains.

Theorem 2 *(Dilworth) For any $t \geq 0$, every DAG with n vertices either has a chain of length greater than t or an antichain of size at least $\lceil n/t \rceil$.*

Proof. By the strong pigeonhole principle. We will prove this for any t. Consider some t. If there is a chain of length $> t$ then we are done. So assume that the longest chain has length $\leq t$. Then we know the n vertices can be partitioned into t antichains. Think of the antichains as the holes and the vertices as the pigeons. Then there must be some hole with at least $\lceil n/t \rceil$ pigeons in it. That is there is an antichain of size at least $\lceil n/t \rceil$. ∎

4 Two Weighing Problems

Suppose that we have n balls. All weigh the same except for one that is heavier. We have a funny balance scale that reports only whether two weights are equal or unequal, not which one is heavier. How many weighings are necessary to find the heavy ball?

This puzzle is like previous weighing problems in that we can represent any strategy with a tree. In this case, since the scale only reports two results at each stage, the tree is binary. If we use W weighings, then the tree has depth W and 2^W leaves. There are n possible outcomes, since any one of the n balls could be the heavy one. If the number of outcomes n is greater than the number of leaves 2^W, then by the Pigeonhole Principle, some leaf is associated with two or more outcomes. This means that the strategy can not distinguish these two situations, and so sometimes produces an incorrect answer. Therefore, we must have $2^W \geq n$, and so we need at least $W \geq \lceil \log_2 n \rceil$ weighings to always find the heavy ball. (This is only a lower bound; we have not given a strategy that actually works with this number of weighings!)

Now suppose the setup is the same, but instead of finding the heavy ball, our task is to find one of the normal-weight balls. There are still n possible outcomes, the strategy tree is still binary, so it appears we still need about $\log n$ weighings.

But what about this strategy? We weigh two balls. If they are equal, we pick one. If they are unequal, we pick a ball that we have not yet weighed. With this strategy, we can find a normal-weight ball in just 1 weighing! The preceding argument must contain a flaw!

The problem with the argument is that a strategy need not distinguish all n possible situations to be correct. For example, if we know that the heavy ball is among the first $n/2$ balls, then we can just pick any of the other $n/2$ to get a normal-weight ball. Unlike the previous weighing problems, we don't have to know precisely which ball is heavy in order to give a correct answer!

5 How Many Trees?

Bijection is a nice tool to determine the number of elements in a set. The basic idea is that we set up a one-to-one correspondence from the elements of the set whose size we want to find to the elements of a set whose size we already know or whose size is relatively easy to estimate.

A remarkable result one can obtain by this method is that the number of different *labelled* trees with n vertices is exactly n^{n-2}. A labelled tree on vertices is a tree whose vertices have been numbered $1, 2, \ldots, n$. E.g. there is 1 labelled tree with 2 vertices, there are 3 labelled trees with 3 vertices. One reason why this a surprising formula is that there doesn't seem to be any direct way to count the number of trees, and so no apparent reason for there to be such a compact formula.

Theorem 3 *(Cayley) The number of labelled trees with n vertices is n^{n-2}.*

Proof. The main idea of the proof is to set up a bijection from trees to sequences of length $n-2$ where each element of the sequence belongs to the set $\{1, 2, \ldots, n\}$. Suppose we succeed in setting up such a bijection. Then the number of different labelled trees on n vertices is the number of different sequences. The latter set is easy to count: there are n possibilities for the first position, n for the second, and so on for each of the $n-2$ positions, so the total number is n^{n-2}.

First let us describe a mapping from trees to sequences. We will refer to the sequence obtained from a tree as the *code* of the tree. The vertices of the tree are labelled $1, 2, \ldots, n$. Repeat for i going from 1 to $n-1$:

1. Let v be the least numbered leaf in the current tree.
2. Set the next element of the code, a_i, to be the number of the vertex that v is connected to.
3. Delete v.

Notice that we have a code $(a_1, \ldots, a_{n-1})$ of length $n-1$ from a tree of n vertices.

Lemma 1 $a_{n-1} = n$.

Proof. By contradiction. First note that every tree has at least 2 leaves. Then suppose for a contradiction that $a_{n-1} < n$. Then at the last step when there are two vertices remaining, both labels must be less than n. But this implies that the vertex with label n was deleted at an earlier step. This is impossible since at every stage the procedure had a choice of at least two leaves and always picked the smaller one. So the vertex with label n would never have been deleted. ■

It will be more convenient to deal with codes of length $n-1$. The previous lemma says that basically the code is of length $n-2$ (the last element, a_{n-1}, is fixed). Now we will prove the following statements which together imply that the above mapping is a bijection.

Lemma 2 *Let $(a_1, a_2, \ldots, a_{n-1})$ be the code obtained from a tree T. Then this code uniquely characterizes T.*

In other words each tree T is mapped to a unique code. So the mapping is an injection.
Proof. We will prove the lemma by showing that we can reconstruct T (uniquely) from its code $(a_1, a_2, \ldots, a_{n-1})$. This means that two different trees cannot map to the same code.

For this let $b_1, b_2, \ldots, b_{n-1}$ be the labels of the removed vertices in the order in which they were removed. So at step i of the procedure, b_i is a leaf with an edge to a_i and it is removed. Now b_i must be different from $b_1, b_2, \ldots, b_{i-1}$ since those have already been removed. Also b_i is different from a_i. And, $b_i \neq a_j$ for any $j > i$; because b_i is removed at the current step and it cannot be a neighbor of a vertex at a later step. So $b_i \notin (b_1, \ldots, b_{i-1}, a_i, \ldots, a_{n-1})$. Consider any vertex numbered k that is not in $(b_1, \ldots, b_{i-1}, a_i, \ldots, a_{n-1})$. Such a vertex cannot be the neighbor of a vertex removed at a later step, so it must be a leaf of the current tree $T - \{b_1, \ldots, b_{i-1}\}$. Thus, by the procedure,

$$b_i = \min \{k : k \notin (b_1, \ldots, b_{i-1}, a_i, \ldots, a_{n-1})$$

So the code uniquely determines the numbers $b_1, \ldots, b_{n-1}$. Since (b_i, a_i) are the edges of the tree, this uniquely determines T. ■

Lemma 3 *Let $(a_1, \ldots, a_{n-1})$ be any sequence of integers such that $1 \leq a_i \leq n$ and $a_{n-1} = n$. Then there is a tree T whose code is $(a_1, \ldots, a_{n-1})$.*

This lemma establishes that the mapping from Trees to sequences is a surjection.

Proof. Given a sequence $(a_1, \ldots, a_{n-1})$ we first show how to construct a tree from the sequence. For this let us define $b_1, \ldots, b_{n-1}$ as:

$$b_i = \min \{k : k \notin (b_1, \ldots, b_{i-1}, a_i, \ldots, a_{n-1})$$

Define the graph T on n vertices with edges (b_i, a_i) for $i = 1, \ldots, n-1$. So it has $n-1$ edges in all. We will show that

1. T is a tree and
2. The code of T is $(a_1, \ldots, a_{n-1})$.

Together these statements imply that every sequence has some tree that is mapped to it, hence our mapping from trees to sequences is a surjection.

Define the graph $T_i = T - \{b_1, \ldots, b_{i-1}\}$ for $i = 1, \ldots, n-1$. We will show that b_i is a leaf of T_i. For this notice that $a_i \in T_i$. This is because, by the way we defined the b_i's $a_i \neq b_1, \ldots, b_{i-1}$. Also, we know that b_i is adjacent to a_i. Next let us see that b_i cannot be adjacent to any other vertex of T_i. For the purpose of a contradiction, suppose it is adjacent to some other edge (b_j, a_j) as well. Then $j > i$ since b_j is still in T_i. Further either $b_j = b_i$, which is impossible, or $a_j = b_i$ which contradicts the definition of b_i. Hence b_i is a leaf of T_i. This implies that T and all the T_i's are trees [By contradiction: if there is a cycle in T then it remains a cycle because none of its vertices are ever deleted; but T_{n-1} is a single vertex and hence has no cycles].

Finally, we have to show that the code for T is $(a_1, \ldots, a_{n-1})$. To do this we have to show that for T_i, b_i is the leaf of smallest number. Suppose for a contradiction that there is some other leaf k such that $k < b_i$ in T_i. Since k was not picked when defining b_i, it must be that either k is one of $b_1, b_2, \ldots, b_{i-1}$ or one of $a_i, \ldots, a_{n-1}$. It cannot be the first possibility because those vertices have been deleted in T_i. So $k = a_j$ for some $j \geq i$. Since $a_{n-1} = n > b_i > k$ and we have that $j \leq n-2$ (k cannot be a_{n-1}). Consider T_j. The vertex b_j is a leaf of T_j, and its neighbor is a_j. But $a_j = k$, so it is also a leaf of T_j. This implies that the number of vertices in T_j is 2. However the number of vertices is $n - j + 1 \geq 3$ which is a contradiction. So b_i must be the smallest numbered leaf of T_i. This proves that the code for T is $(a_1, \ldots, a_n)$. ∎

From the above two lemmas we see that there is a bijection from labelled trees on n vertices to sequences of length $n-1$ on the set $\{1, 2, \ldots, n\}$ and the last element is constrained to be n. But the number of elements in the latter set is exactly n^{n-2} (all possible choices of the first $n-2$ elements). ∎

6 Team Problems

1. Let S be the set of numbers $7, 77, 777, \ldots$. Prove that there is an element of S that is divisible by 1997.

2. A boolean function from the naturals to $\{0,1\}$ is map $f : \mathcal{N} \rightarrow \{0,1\}$ i.e., it assigns 0 or 1 to each natural number. A computer program is a finite piece of code (in your favourite programming language). The code for a boolean function takes as input a natural number and outputs 0 or 1. Prove that not all boolean functions f are computable (i.e., not every boolean function can be programmed with a finite piece of code).

   (a) What is the cardinality of the set of finite programs ? Show that this is a countable set.

   (b) What is the cardinality of the set of boolean functions ? Show that this is equal to $|2^{\mathcal{N}}|$.

   (c) From the above, deduce that there are uncomputable boolean functions.