CS 30: Discrete Math in CS (Winter 2020): Lecture 10

Date: 24th January, 2020 (Friday)
Topic: Combinatorics: Using functions to count, Division Principle
Disclaimer: These notes have not gone through scrutiny and in all probability contain errors.
Please discuss in Piazza/email errors to deeparnab@dartmouth.edu

---

1. **Maps to Count.** Often when faced with counting the number of elements of a certain set S, we see a correspondence/mapping to another set A whose cardinality we already know. In that case, this mapping can be used to count the number of elements in S.

2. **Bijective Maps.** The best kind of maps are *bijective* maps. Suppose we want to figure out $|S|$. If we can find a bijection $f : S \to A$, where A is a set for which we already know $|A|$, then we are done! Since f is a bijection we have $|S| = |A|$; the latter we already know! A very useful example is given below, and a few more are in the UGP. But we will see more shortly.

   (a) *The number of subsets of a finite set.* Let U be a finite set with $|U| = n$. How many subsets does U have? That is, if we define
   $$\mathcal{P}(U) := \{S : S \subseteq U\}$$
   then what is $|\mathcal{P}(U)|$? By the way, the set $\mathcal{P}(U)$ of subsets of U has a name; it is called the *power set* of U.
   The answer to the above question is 2^n. Why 2^n? The simplest proof is via a bijective map. What other set do you know which has 2^n elements; we saw last class that the set
   $$\{0,1\}^n := \{\vec{x} = (x_1, x_2, \ldots, x_n) : x_i \in \{0,1\}\}$$
   of length n bit strings is of size 2^n. Can we find a bijection from $\mathcal{P}(U)$ to $\{0,1\}^n$?
   Here is a bijection. First rename the elements of U to be $\{u_1, u_2, \ldots, u_n\}$. Given a subset $S \subseteq U$, consider the following n-bit string $\vec{x}$ where $x_i = 1$ if and only if $u_i \in S$. Note that the map takes every subset $S \subseteq U$ to an n-bit string.
   The map is surjective; given any bit string $\vec{x}$, consider the subset S which contains u_i if and only if $x_i = 1$. This set S maps to $\vec{x}$. The map is also injective; given two subsets $S \neq T$, there must be an element u_i which is in S but not in T, or vice-versa. Their maps also differ on the ith bit.
   Since the map is bijective, we get $|\mathcal{P}(U)| = |\{0,1\}^n|$ and the latter, we know from last time using the product principle, is 2^n.

   (b) *The number of **odd** subsets of a finite set.* How many *odd* subsets does U have? That is, if we define
   $$\mathcal{O} := \{S : S \subseteq U, \quad |S| \text{ odd}\}$$
   then what is $|\mathcal{O}|$?
   To answer this, we actually find a bijective mapping from all the odd sets to all the even sets. To this end, define
   $$\mathcal{E} := \{S : S \subseteq U, \quad |S| \text{ even}\}$$

We now define a bijective map from $\mathcal{O}$ to $\mathcal{E}$ when $n \geq 1$.

To this end, let's recall $U = \{u_1, \ldots, u_n\}$. Given an odd set $S \in \mathcal{O}$, we map it to

$$g(S) = \begin{cases} S \setminus \{u_1\} & \text{if } u_1 \in S \\ S \cup \{u_1\} & \text{if } u_1 \notin S \end{cases}$$

First observe g is valid. Indeed, for any S, $|g(S)| = |S| + 1$ or $|S| - 1$ which is even if $|S|$ is odd. Also observe $g(S) \subseteq U$. Thus, $g(S) \in \mathcal{E}$.

We claim that g is surjective. How will you do it? Given any *even* set $E \in \mathcal{E}$, you need to find a set $S \in \mathcal{O}$ which maps to it. Can you do it? Think of the two cases: $u_1 \in E$ and $u_1 \notin E$. Finish the details.

We claim that g is injective. To this end, fix two odd sets S and T which are unequal. If both contain u_1, then $S \neq T$ implies $S \setminus \{u_1\} \neq T \setminus \{u_1\}$, that is, $g(S) \neq g(T)$. If both *don't* contain u_1, then $S \neq T$ implies $S \cup \{u_1\} \neq T \cup \{u_1\}$, that is, $g(S) \neq g(T)$. If one of them contains u_1, and the other doesn't; so, for instance, suppose $u_1 \in S$ and $u_1 \notin T$, then note that $u_1 \notin g(S)$ and $u_1 \in g(T)$; this implies $g(S) \neq g(T)$.

Thus, g is a valid bijection from $\mathcal{O}$ to $\mathcal{E}$. This implies, $|\mathcal{O}| = |\mathcal{E}|$. How does it help? Well, we know that every subset is either odd or even, but not both. That is, $\mathcal{P}(U) = \mathcal{O} \cup \mathcal{E}$ and $\mathcal{O} \cap \mathcal{E} = \emptyset$. Thus, $2^n = |\mathcal{P}(U)| = |\mathcal{O}| + |\mathcal{E}| = 2 \cdot |\mathcal{O}|$. This implies, $|\mathcal{O}| = 2^{n-1}$. ✍

Exercise: *Can you try to generalize the above argument to see how many subsets of U have cardinality divisible by 3?*

3. **The Division Rule.** Sometimes we cannot find a bijection from the set S we want to count to a set A that we already know the count of. However, instead of finding an one-to-one, surjective mapping from A to S, we can find a k-to-one surjective mapping from A to S. This is also useful, for then $|S| = |A|/k$. The principle is encapsulated as follows.

   Suppose we can find a mapping $f : A \to S$ such that (a) f is surjective, and (b) for every $s \in S$, there is *exactly* k elements in A which map to s, then $|S| = |A|/k$.

   Examples.

   (a) *How many anagrams are there of the letters in GOOD?* There are 4 letters in the word, and so there are $4! = 24$ permutations of these letters. However, some of these permutations map to the same rearrangement. For example, if we mark the two O's as O_1 and O_2, then the two distinct permutations GO_1O_2D and GO_2O_1D map to the same rearrangement. Thus, there is a map from the set of all permutations to the set of valid anagrams where two distinct permutations (two since there are 2 O's) map to the same rearrangement. This implies there are $24/2 = 12$ rearrangements of the string GOOD. ✍

   Exercise: *How many ways can we rearrange the letters of TENNESSEE?*

   (b) *How many different ways can we line up 5 red balls, 4 blue balls, and 3 green balls?* This is exactly the same logic as before. There are a total of 12 balls, and *if* each of these balls were distinct, there would be $12!$ ways of arranging the balls in a line (sequence).

However, the balls are not distinct. The red balls are interchangeable, so are the blue balls, so are the green balls.

So if we name the red balls $R_1, R_2, \ldots, R_5$, and all the blue balls B_1, B_2, B_3, B_4, and the green balls G_1, G_2, G_3 thereby making them distinct, then there are $12!$ sequences using these characters. However we can map a bunch of these sequences to the *same* sequence *when the names are wiped out*. For example, the sequence $R_1B_1R_2G_1B_2R_3R_4R_5B_3B_4G_2G_3$ is the same as $R_2B_3R_4G_2B_2R_1R_5R_3B_1B_4G_1G_3$, when the names are wiped out: they both are $RBRGRRRBBGG$.

So how many sequences with names map to the same sequence without names? Well, once we fix a pattern of colors, the permutation of the red balls' names lead to the same pattern. Similarly, with the blue balls. And with the green balls. Now we can apply the product principle to see that the number of "collisions" equals the number of permutations of red balls times the number of permutations of blue balls times the number of permutations of green balls. This number is $5!4!3!$. Thus the answer is $12!/(5!4!3!)$ which is what it is...

(c) *How many arrangements of a string are there?* The above two arguments can be encapsulated in the following powerful "formula".

Theorem 1. Given a string s with k *distinct* characters where character i, $1 \le i \le k$, appears n_i times, where n_i is a positive integer. Then, the number of distinct rearrangements of s is

$$\frac{(n_1 + n_2 + \cdots + n_k)!}{n_1! \cdot n_2! \cdot \cdots \cdot n_k!}$$

✍

Exercise: *Use the above to solve the following:*

- *How many ways can 5 red balls, 4 yellow balls, and 1 green ball be arranged in a line?*
- *How many possible genome sequences of length 100 can be made of the characters $\{A, C, G, T\}$ if they all appear equally often?*
- *How many anagrams (not necessarily in the dictionary) are possible of your first name?*

(d) *How many n-length bit strings have exactly k ones?* A corollary to the above is the another very important identity. Note that a n length bit string with exactly k ones has exactly $n - k$ zeros. Thus, the above question is basically asking, how many distinct rearrangements are there of a string with k ones and $n - k$ zeros? Applying the theorem above, we see the answer is $\frac{n!}{k!(n-k)!}$. This has a special name – it is called $\binom{n}{k}$. We will meet this expression much more in the next two lectures.

Remark: Tattoo this in your brain: *The number of ways of choosing a set of k items from n distinct items is $\binom{n}{k} = \frac{n!}{k!(n-k)!}$. If someone wakes you up at 3 in the morning and asks you the above, you should be able to just spit the answer out.*

✍

Exercise: *Play with this formula. Write down the* actual values *of* $\binom{n}{k}$ *for all* $1 \le k \le n \le 5$. *Get a feel of how big these are. You should have a vague idea of the order of magnitude of* $\binom{10}{4}$, *or* $\binom{100}{10}$.