$\boxed{\textbf{Basics of Counting}}$

Notes.
Main concepts

- Sum rule
- Product rule
- Bijection principle

Product rule

Often when counting something, it is useful to think of what we are counting as a selection from one set **AND** then a selection from another set. In such situations the product rule is useful.

The product rule stems from the concept of Cartesian products. $|A \times B| = |A||B|$.

Example, a customer goes to Dunkin Donuts and orders a small coffee. They are then asked if they want cream and sugar. How many different types of coffee are possible.

A simple way of dealing with this problem is to say that the customer has to first choose from a set that has the elements $\{cream, nocream\}$ and then from another set that has the elements $\{sugar, nosugar\}$.

So there are 4 possibilities.

Important clarification - We are not distinguishing between option1 -adding cream first and then adding sugar) and option2 - (adding sugar first and then adding cream). That, hopefully, is a reasonable assumption!

In general, if you are choosing from a set A_1, then choosing from a set A_2, then from a set A_3 and so on until A_n, the total number of choices you have is $|A_1| \cdot |A_2| \cdots |A_n|$. But, you do need to be careful that the number of choices you have in subsequent steps does not depend on things you do in the previous steps.

Another way this gets stated (a more formal way!) is

If an operation has k steps and

the first step can be performed in n_1 ways

the second step can be performed in n_2 ways (regardless of how the first step was performed)

the kth step can be performed in n_k (regardless of how the previous steps were performed)

then the entire operation can be performed in $n_1 n_2 \cdots n_k$ ways

Example - How many possible 3 digit numbers are there? The first digit cannot be 0 so 9 possibilities. The next two digits can be anything. Hence - 9 * 10 * 10 = 900. Also easily done as we have to count all the numbers from 100 to 999, both inclusive. 999-100+1 elements.

Fun example - How many possible drinks can you get at Starbucks? We'll figure out the answer in class.

Sum rule

If instead of making choices one after the other, you are counting something where you choose from one set OR from another set, then the sum rule is applied.

The keyword to look out for over here is **OR**.

A database has two tables. One of them has information about faculty - F, the other has information about administrative staff - S. The payroll program is trying to figure out how many people to pay. Can it add the number of people in faculty and the number of people in staff? Yes - because these two sets are disjoint.

Consider n sets, A_1, A_2, ..., A_n. If the sets are mutually disjoint, $A_i \cap A_j = \emptyset$, then $|A_1 \cup A_2 \cup \ldots \cup A_n| = |A_1| + |A_2| + \ldots + |A_n|$.

Consider two sets U - united states citizens. C - canadian citizens. If I want to count the number of people who are either United States citizens OR Canadian citizens, can we do it just by saying $|U| + |C|$. Why or why not?

What about her? - http://en.wikipedia.org/wiki/Alanis_Morissette.

When applying the sum rule, be careful that your sets truly are disjoint.

Applying the sum and product rule together

How many passwords can be made using just upper and lower case letters that are between 4 and 8 characters long?

Let us use some notation first

P_4 - passwords of length 4.

P_5 - passwords of length 5.

P_6 - passwords of length 6

and so on.

Then consider $P = P_4 \cup P_5 \cup P_6 \cup P_7 \cup P_8$. What we want to find is $|P|$. The sum rule can be applied since you cannot possibly be of length 5 and of length 8 for instance. The sets are mutually disjoint (nothing in common between any pairs of them!).

So $|P| = |P_4| + |P_5| + \ldots + |P_8|$. Now what is $|P_4|$?

Assume A is the set of all the upper case and lower case alphabet in English. $|A| = 26 \cdot 2 = 52$. For a 4 letter password, we can think of it as a 4-tuple. In particular $|P_4| = |A \times A \times A \times A|$. That means $|P_4| = 52^4$.

That logic extends to every P_i, so the total number of passwords becomes

$|P| = 52^4 + 52^5 + 52^6 + 52^7 + 52^8$.

Tricky Example (can be found in zybook as well)

Three officers - a president, a treasurer and a secretary - are to be chosen from among four people. A, B, C and D. Suppose for whatever reason A cannot be president and either C or D must be secretary. How many ways can these officers be chosen.

This question is tricky because the order of our choosing operations becomes more critical.

If we pick the president, then pick the treasurer and then pick the secretary, we run into an issue because the number of choices for secretary depends on who was picked for president and treasurer. If C or D were picked in either of the first two operations, the number of choices becomes less than 2. If we want to do this via the product principle we need to realize that we just need to pick these 3 different officials. We can reorder the order of choosing the officials so that it looks more like a situation where the product principle can be applied.

Reorder to do the choosing in the following manner - pick secretary first, then pick the president, then pick the treasurer. Now there are 2 people that can be chosen from for the secretary (C and D). Regardless of who gets chosen, when it comes to picking the president next, we realize that there are only 2 choices. B and whoever was not picked out of C and D. Also regardless of who we pick as president, there are 2 choices for the treasurer - A and whoever was not picked as either the president or the secretary.

So it is $2 \times 2 \times 2 = 8$

It is also interesting to try and do this same question in a different manner

1. Pick a secretary - you can do this in 2 ways
2. Pick a treasurer - You cannot pick the person who was picked as secretary, so this can be done in 3 ways.
3. Pick a president - But this now depends on who was picked as treasurer. If A gets picked as treasurer we have 2 choices. If anyone else gets picked as treasurer, we have only 1 choice since A cannot be president. Since we are depending upon the result of the previous step, we cannot use the product principle.

However since we are making a choice in the second stage we can look at it in this manner. Either we pick A or do not. Since the set of choices that include A are obviously disjoint from the choices that do not include A, we can use the sum rule and break this up into 2 case

- Number of ways to choose the officers, if A is to be picked - 2 ways to pick the secretary, A is the treasurer and 2 ways to pick the president. So 4 ways.
- Number of ways to choose the officers, if A is not to be picked - Still 2 ways to pick the secretary, 2 ways to pick the treasurer (either it will be B or the person who was

not picked to be the secretary) and now there is only 1 way to pick the president since A is not being picked at all. So 4 ways again.

So, the number of ways in total just becomes $4 + 4 = 8$ ways.

Example

How many different 4-letter radio station call letters can be made if the first letter can be a K or a W and no letters can be repeated. A radio station call letter is something like WKYP or KXPN etc.

Either the radio station begins with a

K - $25 \cdot 24 \cdot 23$ possibilities (cannot repeat the K hence begins with 25).

OR

it begins with a W - $25 \cdot 24 \cdot 23$ possibilities

So a total of $2 \cdot 25 \cdot 24 \cdot 23$ possibilities.

Bijection Principle

A set X and a set Y have the same number of elements if and only if, there is some bijection between the two sets.

Remember that a bijection means one to one and onto.

This principle is surprisingly useful at times when it is tough to count the elements of a set directly, but the elements of the set can be mapped to an easier set.

Example - 30 teams qualify for a new form of the World Cup where every single game is a knockout. If you lose, you are out. How many games need to be played before we declare a winner?

The initial and obvious approach would be to start constructing a format of the World Cup where in the first stage, you have 15 matches and then drawing out a full bracket (or tree if you prefer). The problem is once the first stage is done, we have 15 winners. How exactly do we play them. Does it matter which order the knockouts are held in?

Instead of counting this by looking at the matches, let us see if there is any interesting mapping that can be done. In every game, there is a winner and a loser. So let us map the set of games to the set of losers. Clearly, each game can be identified uniquely by its loser. The game to loser mapping is one-one and onto and therefore a bijection.

Since we have a bijection, we know that the number of losers is the same as the number of games. So how many losers are there? There are 29 losers! Therefore there are 29 games.

This result should feel a little surprising because regardless of how you arrange the matches, the counting is done so very simply.

Applications

So why is it important to know how to count?

1. Discrete probability, which we will cover in this course, is essentially an exercise in counting twice. Count all possible scenarios and then count the specific event you are interested in. Divide the two and get the probability.

2. There are lots of applications of computer science (or programming if you will) to sequencing problems. One of the more famous ones is the Traveling Salesperson Problem. Given a list of cities and the distances between each pair of cities, what is the shortest possible route that visits each city exactly once and returns to the origin city.

   Every possible solution can be mapped to a sequence of cities. But how many such sequences are there? That is a counting problem. For n cities, you have n choices first, then $n-1$, then $n-2$ and so on. This quantity $n \times (n-1) \times (n-2) \cdots 1$ is called n factorial and represented as $n!$.

   We will talk more about factorials and arrangements in upcoming lectures.

3. How long will programs take? That depends on how many operations they are executing. For instance, here is a Python program that sorts a list. How many comparison operations (between pairs of elements in the list) are being made?

   No need to read this if you have not covered lists in 591 yet

```
def selectionsort( aList ):
  for i in range( len( aList ) ):
    least = i
    for k in range( i + 1 , len( aList ) ):
      if aList[k] < aList[least]:
        least = k

    swap( aList, least, i )


def swap( A, x, y ):
  tmp = A[x]
  A[x] = A[y]
  A[y] = tmp
```

   In each iteration, the element in position i is compared with everything to its right. Let S_i be the set of pairs of elements that get compared in the ith iteration. For instance, S_1 will contain the pairs (1st element, 2nd element), (1st element, 3rd element) and so on. S_2 will contain the pairs (2nd element, 3rd element), (2nd element, 4th element) and so on. The interesting property of these sets is that they are mutually disjoint. So for computing the total number of comparisons we just need to add up the cardinalities of all the S_is.

Total number of comparisons $= \displaystyle\sum_{i=1}^{n} |S_i|$.

Now what is $|S_1|$? It is $n-1$.

What is $|S_2|$? Elements from the 3rd to the nth position get compared to the 2nd element. So it is $n-2$.

In general, the pattern is $|S_i| = n-i$.

The total number of comparisons $= (n-1)+(n-2)+\ldots+1$. That can be shown to be $\frac{n(n-1)}{2}$