Chris Piech
CS 109

Lecture Notes #1
Sept 24, 2018

Counting

Based on a handout by Mehran Sahami with examples by Peter Norvig

Although you may have thought you had a pretty good grasp on the notion of counting at the age of three, it turns out that you had to wait until now to learn how to really count. Aren't you glad you took this class now?! But seriously, below we present some properties related to counting which you may find helpful in the future.

Counting is important in the world of computer science for a few reasons. In order to understand probability on a fundamental level, it is useful to first understand counting. Moreover, while computers are fast, some problems require so much work that they would take an unreasonable amount of time to complete. We can use counting theory to reason about complexity.

1 Sum Rule

Sum Rule of Counting
If the outcome of an experiment can either be one of m outcomes or one of n outcomes, where none of the outcomes in the set of m outcomes is the same as the any of the outcomes in the set of n outcomes, then there are $m + n$ possible outcomes of the experiment.

Rewritten using set notation, the Sum Rule states that if the outcomes of an experiment can either be drawn from set A or set B, where $|A| = m$ and $|B| = n$, and $A \cap B = \emptyset$, then the number of outcomes of the experiment is $|A| + |B| = m + n$.

Example 1

Problem: You are running an on-line social networking application which has its distributed servers housed in two different data centers, one in San Francisco and the other in Boston. The San Francisco data center has 100 servers in it and the Boston data center has 50 servers in it. If a server request is sent to the application, how large is the set of servers it may get routed to?

Solution: Since the request can be sent to either of the two data centers and none of the machines in either data center are the same, the Sum Rule of Counting applies. Using this rule, we know that the request could potentially be routed to any of the 150 (= 100 + 50) servers.

2 Product Rule

Product Rule of Counting
If an experiment has two parts, where the first part can result in one of m outcomes and the second part can result in one of n outcomes regardless of the outcome of the first part, then the total number of outcomes for the experiment is $m \cdot n$.

Rewritten using set notation, the Product Rule states that if an experiment with two parts has an outcome from set A in the first part, where $|A| = m$, and an outcome from set B in the second part (regardless of the outcome of the first part), where $|B| = n$, then the total number of outcomes of the experiment is $|A||B| = mn$.

Example 2

Problem: Two 6-sided dice, with faces numbered 1 through 6, are rolled. How many possible outcomes of the roll are there?

Solution: Note that we are not concerned with the total value of the two dice, but rather the set of all explicit outcomes of the rolls. Since the first die[1] can come up with 6 possible values and the second die similarly can have 6 possible values (regardless of what appeared on the first die), the total number of potential outcomes is 36 (= 6 * 6). These possible outcomes are explicitly listed below as a series of pairs, denoting the values rolled on the pair of dice:

(1, 1)	(1, 2)	(1, 3)	(1, 4)	(1, 5)	(1, 6)
(2, 1)	(2, 2)	(2, 3)	(2, 4)	(2, 5)	(2, 6)
(3, 1)	(3, 2)	(3, 3)	(3, 4)	(3, 5)	(3, 6)
(4, 1)	(4, 2)	(4, 3)	(4, 4)	(4, 5)	(4, 6)
(5, 1)	(5, 2)	(5, 3)	(5, 4)	(5, 5)	(5, 6)
(6, 1)	(6, 2)	(6, 3)	(6, 4)	(6, 5)	(6, 6)

Example 3

Problem: Consider a hash table with 100 buckets. Two arbitrary strings are independently hashed and added to the table. How many possible ways are there for the strings to be stored in the table?

Solution: Each string can be hashed to one of 100 buckets. Since the results of hashing the first string do not impact the hash of the second, there are 100 * 100 = 10,000 ways that the two strings may be stored in the hash table.

[1] "die" is the singular form of the word "dice" (which is the plural form).

Example 4: Unique configurations of Go

The number of atoms in the observable universe is about 10 to the 80th power (10^{80}). This measure is frequently used by computer scientists as a canonical really big number. There certainly are a lot of atoms in the universe. As a leading expert said,

> “Space is big. Really big. You just won’t believe how vastly, hugely, mind-bogglingly big it is. I mean, you may think it’s a long way down the road to the chemist, but that’s just peanuts to space." - Douglas Adams

This number is often used to demonstrate tasks that computers will never be able to solve. Problems can quickly grow to such an absurd size through the product rule of counting. For example, lets say we wanted to write an AI algorithm to play the game of Go, and we need to store each possible board configuration. How many boards might we have to store?

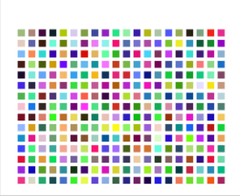
A Go board has 19×19 points where a user can place a stone. Each of the points can be in one of three states: empty, occupied by black or occupied by white. By the product rule of counting, we can compute the number of unique board configurations. Each board point is a unique choice where you can decide to have one of the three options in the set {Black, White, No Stone} so there are $3^{(19\times19)} \approx 10^{172}$ possible board configurations. It turns out “only" about 10^{170} of those positions are legal. That is about the square of the number of atoms in the universe. In other-words: if there was another universe of atoms for every single atom, only then would there be as many atoms in the universe as there are unique configurations of a Go board. Not even the snazziest datastructure can hold that many configurations.

Example: The Number of Digital Pictures

There is an art project to display every possible picture. Surely that would take a long time, because there must be many possible pictures. But how many? We will assume the color model known as True Color, in which each pixel can be one of $2^{24} \approx 17$ million distinct colors.

How many distinct pictures can you generate from (a) a digital camera shown with 12 million pixels, (b) a grid with 300 pixels, and (c) a grid with just 12 pixels?

(a) 12 million pixels



(b) 300 pixels

(c) 12 pixels

Answer: An array of n pixels produces $(17 \text{ million})^n$ different pictures. $(17 \text{ million})^{12} \approx 10^{86}$, so the tiny 12-pixel grid produces a million times more pictures than the number of atoms in the universe! How about the 300 pixel array? It can produce 10^{2167} pictures. You may think the number of atoms in the universe is big, but that's just peanuts to the number of pictures in a 300-pixel array. And 12M pixels? $10^{86696638}$ pictures.

So the number of possible pictures is really, really, really big. The crucial idea is, that as a number of physical things, 10^{80} is a really big number. But when you start applying the product rule many times, 10^{80} is a rather small number. It doesn't take a universe of product rules to get up to 10^{80} outcomes.

Example 5: Leveraging Exponential Growth

Let's take a moment to talk about how the product rule of counting can help! Most logrithmic time algorithms leverage this principle.

Problem: You need to simulate 10 million unique examples of student solutions to Breakout for a machine learning algorithm. You can generate a single example by composing a chain of independent decisions. How many binary decisions (two outcomes) do you have to encode in order to describe 10 million examples?

Solution: Using the product rule, the total number of unique outcomes is going to be 2 multiplied by itself n times where n is the number of binary decisions. to get 10 million outcomes we want to chose n such that $10,000,000 = 2^n$. We only need to encode $n = 24$ decisions.

3 The Inclusion-Exclusion Principle

Inclusion-Exclusion Principle:
If the outcome of an experiment can either be drawn from set A or set B, and sets A and B may potentially overlap (i.e., it is not guaranteed that $A \cap B = \varnothing$), then the number of outcomes of the experiment is $|A \cup B| = |A| + |B| - |A \cap B|$.

Note that the Inclusion-Exclusion Principle generalizes the Sum Rule of Counting for arbitrary sets A and B. In the case where $A \cap B = \varnothing$, the Inclusion-Exclusion Principle gives the same result as the Sum Rule of Counting since $|\varnothing| = 0$.

The Inclusion-Exclusion principle helps to make sure we aren't counting any element more than once. If you over-count, then you have to subtract off the number of elements that were double counted.

Example 6

Problem: An 8-bit string (one byte) is sent over a network. The valid set of strings recognized by the receiver must either start with 01 or end with 10. How many such strings are there?

Solution: The potential bit strings that match the receiver's criteria can either be the 64 strings that start with 01 (since that last 6 bits are left unspecified, allowing for $2^6 = 64$ possibilities) or the 64 strings that end with 10 (since the first 6 bits are unspecified). Of course, these two sets overlap, since strings that start with 01 and end with 10 are in both sets. There are $2^4 = 16$ such strings (since the middle 4 bits can be arbitrary). Casting this description into corresponding set notation, we have: $|A| = 64$, $|B| = 64$, and $|A \cap B| = 16$, so by the Inclusion-Exclusion Principle, there are $64 + 64 - 16 = 112$ strings that match the specified receiver's criteria.

4 The Pigeonhole Principle

Before we start putting pigeons in holes, let's go over *Floor* and *Ceiling*, two handy functions. Their names sound so much neater than "rounding down" and "rounding up", and they are well-defined on negative numbers too. Bonus.

Floor function: The floor function assigns to the real number x the largest integer that is less than or equal to x. The floor function applied to x is denoted $\lfloor x \rfloor$.

Ceiling function: The ceiling function assigns to the real number x the smallest integer that is greater than or equal to x. The floor function applied to x is denoted $\lceil x \rceil$.

Here are a few examples:

$$\begin{array}{llll} \lfloor 1/2 \rfloor = 0 & \lfloor -1/2 \rfloor = -1 & \lfloor 2.9 \rfloor = 2 & \lfloor 8.0 \rfloor = 8 \\ \lceil 1/2 \rceil = 1 & \lceil -1/2 \rceil = 0 & \lceil 2.9 \rceil = 3 & \lceil 8.0 \rceil = 8 \end{array}$$

Now that you know floors and ceilings, you are ready for the Pigeonhole Principle.

Basic Pigeonhole Principle: For positive integers m and n, if m objects are placed in n buckets, where $m > n$, then at least one bucket must contain at least two objects.

In a more general form, this principle can be stated as:

General Pigeonhole Principle: For positive integers m and n, if m objects are placed in n buckets, then at least one bucket must contain at least $\lceil m/n \rceil$ objects.

Note that the generalized form does not require the constraint that $m > n$, since in the case where $m \leq n$, we have $\lceil m/n \rceil = 1$, and it trivially holds that at least one bucket will contain at least one object.

Example 7

Problem: Consider a hash table with 100 buckets. 950 strings are hashed and added to the table.

a) Is it possible that a bucket in the table contains no entries?

b) Is it guaranteed that at least one bucket in the table contains at least two entries?

c) Is it guaranteed that at least one bucket in the table contains at least 10 entries?

d) Is it guaranteed that at least one bucket in the table contains at least 11 entries?

Solution:

a) Yes. As one example, it is possible (albeit very improbable) that all 950 strings get hashed to the same bucket (say bucket 0). In this case bucket 1 would have no entries.

b) Yes. Since, 950 objects are placed in 100 buckets and 950 > 100, by the Basic Pigeonhole Principle, it follows that at least one bucket must contain at least two entries.

c) Yes. Since, 950 objects are placed in 100 buckets and $\lceil 950/100 \rceil = \lceil 9.5 \rceil = 10$, by the General Pigeonhole Principle, it follows that at least one bucket must contain at least 10 entries.

d) No. As one example, consider the case where the first 50 bucket each contain 10 entries and the second 50 buckets each contain 9 entries. This accounts for all 950 entries (50 * 10 + 50 * 9 = 950), but there is no bucket that contains 11 entries in the hash table.

Bibliography

For additional information on counting, you can consult a good discrete mathematics or probability textbook. Some of the discussion above is based on the treatment in:

K. Rosen, *Discrete Mathematics and its Applications, 6th Ed.*, New York: McGraw-Hill, 2007.

The examples for the size of the universe are from Peter Norvig.