CS109 assignment 1

andreea.a.musat@gmail.com

October 2018

1. A substitution cypher is derived from an ordering of the letters in the alphabet. How many ways can the 26 letters be ordered if each letter appears exactly once and:
a. There are no other restrictions?
b. The letters Q and U must be next to each other (but in any order)?

Solution:

a. There are 26! ways to permute the 26 letters of the alphabet.

b. We can assume that U does not exist in the alphabet, as its position always depends on Q in order to get a valid arrangement. Thus, we have 25! ways in which we can arrange all letters except U. Now, we can place U either before or after Q in any of these, so we get a total of $2 * 25!$ ways.

2. You are counting cards in a card game that uses four standard decks of cards. There are 208 cards total. Each deck has 52 cards (13 values each with 4 suits). Cards are only distinguishable based on their suit and value, not which deck they came from.
a. In how many distinct ways can the cards be ordered?
b. You are dealt two cards. How many distinct pairs of cards can you be dealt? Note: the order of the two cards you are dealt does not matter.
c. You are dealt two cards. Cards with values 10, Jack, Queen, King and Ace are considered “good” cards. How many ways can you get two “good” cards? Order does not matter.

Solution:

a. We are trying to order 208 elements, knowing that there are 13 * 4 = 52 unique elements and that each elements appears 4 times, as there ar 4 decks. Thus, there are: $\frac{208!}{4!4!...4!} = \frac{208!}{(4!)^{52}}$ ways of ordering them.

b. Because we have more than one deck, it is possible to have a pair consisting of the same card. There are 52^2 possible pairs of cards, but we count the pairs with 2 different elements twice ((1, 2) and (2, 1) for example). If we want to count all pairs twice, we have to add 52 to that number (the number of pairs having the same card), so we get: $\frac{52^2+52}{2} = 26 * 53$ total pairs.

c. There are 5 card values * 4 suits * 4 decks = 80 good cards, so one could get a pair of 2 good cards in $\binom{80}{2}$ ways.

3. In how many ways can n identical server requests (“identical balls”) be distributed among r servers (“urns”) so that the ith server receives at least m_i requests, for each i = 1, 2, ..., r ? You can assume that $n \geq \sum_{i=1}^{r} m_i$.

Solution

We first distribute the minimum number of requests for each server, so that the constraint is satisfied: server i gets m_i requests. Using the notation $m = \sum_{i=1}^{r} m_i$, we are left with $n - m$ requests to distribute on r servers, which can be done in $\binom{n-m+r-1}{r-1}$ ways, as the requests are assumed to be identical.

4. Determine the number of vectors $(x_1, x_2, ..., x_n)$ such that each x_i is a non-negative integer and $\sum_{i=1}^{n} x_i \leq k$ where k is some constant non-negative integer. Note that you can think of n (the size of the vector) as a constant that can be used in your answer.

Solution

We can split this problem into k + 1 more specific problems and solve each of them separately: *Given a constant j, determine the number of vectors n-dimensional vectors such that each component is a non-negative integer and the sum of all the components is exactly j.* This problem is equivalent to distributing j balls across n urns (and allowing some of the urns to be empty), which can be done in $\binom{j+n-1}{n-1}$ ways for each j. Now, because we are interested in vectors having the sum of their elements less than or equal to k, we get the total number of vectors: $\sum_{i=0}^{k} \binom{j+n-1}{n-1}$

5. Imagine you have a robot (Q) that lives on an n x m grid (it has n rows and m columns). The robot starts in cell (1, 1) (lower left corner) and can take steps either to the right or up (no left or down steps). How many distinct paths can the robot take to the destination in cell (n, m) (upper right corner):
a. If there are no additional constraints?
b. The robot must start by moving to the right?
c. If the robot changes direction exactly 3 times? As an example: moving up two times in a row is not changing directions but switching from moving up to moving right is. Moving [Up, Right, Right, Up] would count as having two direction switches.

Solution

a. Each path the robot can take can be encoded as a binary string of length n + m consisting of n 0s and m 1s, where 0 means the robot takes a step up and 1 means that the robot takes a step to the right. There are $\binom{n+m}{n} = \binom{n+m}{m}$ ways of choosing one such path.

b. If the robot starts by moving to the right, then we restrict our binary strings to start with 1. Now we have to count the ways in which we can construct a binary string of length n + m - 1 that consists of n 0s and m-1 1s. There are $\binom{n+m-1}{n} = \binom{n+m-1}{m-1}$ such strings.

c. If the robot changes direction 3 times, its possible paths would have the following form: 00...0 | 11...1 | 00...0 | 11...1 (or starting with a block of ones instead of zeros; we can count these and then multiply the result by 2 to account for the ones starting with 1s), where the first block of zeros should have length between 1 and n - 1 inclusive and the first block of ones should have length between 1 and m - 1 inclusive (and the length of the other 2 blocks is dependent on these, so they can be chosen just in one way). There are 2 * (n - 1) * (m - 1) ways to build a path in which the robot changes direction 3 times.

6. Given all the start-up activity going on in high-tech, you realize that applying combinatorics to investment strategies might be an interesting idea to pursue. Say you have $20 million that must be invested among 4 possible companies. Each investment must be in integral units of $1 million, and there are minimal investments that need to be made if one is to invest in these companies. The minimal investments are $1, $2, $3, and $4 million dollars, respectively for company 1, 2, 3, and 4. How many different investment strategies are available if:
a. an investment must be made in each company?
b. investments must be made in at least 3 of the 4 companies?

Solution

a. We first invest the minimum required amount in each company (it can be done in 1 way, as all million dollars are equal :-)), then distribute the rest of the money ($10 million) between the 4 companies, which can be done in $\binom{10+4-1}{4-1} = \binom{13}{3} = 286$ ways.

b. We will count the ways in which we can invest in exactly 3 companies and then add the result from the previous exercise to this number. Knowing that the minimum sum invested in company i is $i million dollars, investing in all companies except i requires a minimum investment of 10 - i million dollars (this is to satisfy the constraint; can be done in 1 way), which leaves us $10 + i million to invest in 3 companies. This can be done in $\binom{10+i+3-1}{3-1} = \binom{12+i}{2}$ ways. Now, knowing that i can be any of these 4 companies, we get a total number of investing in exactly 3 companies: $\sum_{i=1}^{4} \binom{12+i}{2}$, so the final result is: $286 + \sum_{i=1}^{4} \binom{12+i}{2}$.

TODO: exercises 7-8

9. To get good performance when working binary search trees (BST), we must consider the probability of producing completely degenerate BSTs (where each node in the BST has at most one child).
a. If the integers 1 through n are inserted in arbitrary order into a BST (where each possible order is equally likely), what is the probability (as an expression in terms of n) that the resulting BST will have completely degenerate structure?
b. Using your expression from part (a), determine the smallest value of n for which the probability of forming a completely degenerate BST is less than 0.01 (i.e., 1%).

Solution

a. If $(x_1, x_2, \ldots x_n)$ is a permutation of the numbers 1 through n, we denote by $D(x_1, x_2, \ldots x_n)$ the event that the BST generated by $x_1, x_2, \ldots x_n$ in this order is a degenerate BST. We know that a tree is degenerate when each node has at most one child. We know that if an element x is inserted into a BST, all the elements smaller that x will be found on the left subtree of node x, while all the elements greater than x will be found on the right subtree of the BST. Thus, a node will have at most one child only when we know that after inserting it, we are left with numbers that are either smaller or larger than it, but not a mix of them. We can write $P(D(x_1, x_2, \ldots x_n))$ as: $P(D(x_1, x_2, \ldots x_n)) = P((x_1 = \max(x_1, x_2, \ldots x_n) \vee x_1 = \min(x_1, x_2, \ldots x_n)) \wedge D(x_2, x_3, \ldots x_n))$.
Because those are independent events, we rewrite it as: $P(D(x_1, x_2, \ldots x_n)) = (P(x_1 = \max(x_1, x_2, \ldots x_n)) + P(x_1 = \min(x_1, x_2, \ldots x_n)))P(D(x_2, x_3, \ldots x_n))$.
Because the numbers x_i, i $\in 1, 2, \ldots n$ are all different,
$P(x_1 = \max(x_1, x_2, \ldots x_n)) = \frac{1}{n}$ and $P(x_1 = \min(x_1, x_2, \ldots x_n)) = \frac{1}{n}$, we get that: $P(D(x_1, x_2, \ldots x_n)) = \frac{2}{n} P(D(x_2, x_3, \ldots x_n))$.
Writing $P(D(x_2, x_3, \ldots x_n))$ in terms of $P(D(x_3, x_3, \ldots x_n))$ and so on, we obtain: $P(D(x_1, x_2, \ldots x_n)) = \frac{2}{n} \frac{2}{n-1} \frac{2}{n-2} \ldots \frac{2}{3} \frac{2}{2} = \frac{2^{n-1}}{n!}$

b. We want to find the smallest value of n such that: $\frac{2^{n-1}}{n!} < 0.01$ The left-hand side of the inequality is decreasing. We use an inequality to find an upper bound for our minimum n: $\frac{2^{n-1}}{n!} = \frac{2}{3} \frac{2}{4} \ldots \frac{2}{n} <= \frac{2}{3}^{n-2} < 0.01$, which is equivalent to $n > 13$. We can now binary search for n and obtain n = 8.

10. Say a hacker has a list of n distinct password candidates, only one of which will successfully log her into a secure system.
a. If she tries passwords from the list at random, deleting those passwords that do not work, what is the probability that her first successful login will be (exactly) on her k-th try?
b. Now say the hacker tries passwords from the list at random, but does not delete previously tried passwords from the list. She stops after her first successful login attempt. What is the probability that her first successful login will be (exactly) on her k-th try?

Solution

a. If the hacker tries the password at random and deletes them after trying them, each try of all the n passwords generates one of the n! possible permutations. There are (n-1)! ways in which she can guess the correct password on the k-th try (that is, all the permutations that have the correct password on the k-th position), so the probability is $\frac{(n-1)!}{n!} = \frac{1}{n}$

b. The probability that the first successful login will be exactly on the k-th try is equal to the probability that she tries a wrong password on each of the first k-1 tries multiplied by the probability that she tries the correct one on the k-th try, which is: $(\frac{n-1}{n})^{k-1}\frac{1}{n}$.

11. Say a university is offering 3 programming classes: one in Java, one in C++, and one in Python. The classes are open to any of the 100 students at the university. There are:
• a total of 27 students in the Java class
• a total of 26 students in the C++ class
• a total of 18 students in the Python class
• 12 students in both the Java and C++ classes
• 5 students in both the Java and Python classes
• 7 students in both the C++ and Python classes
• 3 students in all three classes (note: these students are also counted as being in each pair of classes in the numbers above).
a. If a student is chosen randomly at the university, what is the probability that he or she is not in any of the 3 programming classes?
b. If a student is chosen randomly at the university, what is the probability that he or she is taking exactly one of the three programming classes?
c. If two students are chosen randomly at the university, what is the probability that at least one of the chosen students is taking at least one programming class?

Solution

a. Using the notation —C— = number of students in the C++ class, —J— = number of students in the Java class and —P— = number of students in the Python class, the total number of students enrolled in programming classes is: $|C|+|J|+|P|-|C\cap J|-|C\cap P|-|P\cap J|+|P\cap J\cap C| = 50$, so the probability that a random student is not in any of the 3 programming classes in 0.5.

b. We denote by $P(\#Cl = 1)$ the probability that the number of programming classes taken by a student is exactly one. $P(\#Cl = 1) = P(Cl = C \vee Cl = P \vee Cl = J) = P(Cl = C) + P(Cl = P) + P(Cl = J)$
The probability that a student is only taking the Java class is $P(Cl = J) = \frac{|J|-|J\cap P|-|J\cap C|+|J\cap C\cap P|}{100} = 0.13$ Similarily, we compute $P(Cl = P) = 0.09$ and $P(Cl = C) = 0.1$, so the probability that a student is taking exactly one programming class is 0.32.

c. The event of choosing a pair of students so that at least one of them is taking at least one programming class is complementary to the event of choosing a pair of students in which none of them takes any programming class. There are 50 students not taking any programming class, so $\binom{50}{2}$ such pairs out of

$\binom{100}{2}$ total pairs, so the probability is $1 - \frac{\binom{50}{2}}{\binom{100}{2}} = 0.7525$.

12. A binary string containing M 0's and N 1's (in arbitrary order, where all orderings are equally likely) is sent over a network. What is the probability that the first r bits of the received message contain exactly k 1's?

Solution

There are $\binom{(M+N)}{N}$ possible strings containing M 0's and N 1's. We assume that $r \geq k$, otherwise the probability is 0. There are $\binom{r}{k}$ ways to form the first r bits of the string so that it contains k 1's and there are $\binom{M+N-r}{N-k}$ ways to form the rest of (M+N-r) bits so that they contain N-k 1's. Thus, the probability is: $\frac{\binom{r}{k}\binom{M+N-r}{N-k}}{\binom{M+N}{N}}$

13. Suppose that m strings are hashed (randomly) into N buckets, assuming that all N^m arrangements are equally likely. Find the probability that exactly k strings are hashed to the first bucket.

Solution

There are $\binom{m}{k}$ ways to choose the k strings that will be hashed into the first bucket. After that, we are left with m - k strings to be hashed in N - 1 buckets, which can be done in $(N-1)^(m-k)$ ways, so the probability of having exactly k strings hashed to the first bucket is: $\frac{\binom{m}{k}(N-1)^{m-k}}{N^m}$

14. A computer generates two random integers in the range 1 to 12, inclusive, where each value in the range 1 to 12 is equally likely to be generated. What is the probability that the second randomly generated integer has a value that is greater than the first?

Solution

First, as each number is equally likely to be generated, $p(n) = \frac{1}{12}, \forall n \in \{1, 2, ..., 12\}$. We use the notation $p(I_2 > I_1)$ for the probability that the second generated integer is greater than the first one. We can write this as:

$$p(I_2 > I_1) = \sum_{i_1, i_2 \in \{1,2,...,12\}, i_2 > i_1} p(i_1)p(i_2|i_1) = \\ \frac{1}{12} \sum_{i_1, i_2 \in \{1,2,...,12\}, i_2 > i_1} p(i_2|i_1),$$

because the probability of each integer i_1 being generated is $\frac{1}{12}$. Now, the probability to generate a second integer greater than the first one i_1 is $\frac{12-i_1}{12}$, so the final result is:

$$p(I_2 > I_1) = \frac{1}{12} \sum_{i_1, \in \{1,2,...,12\}} \frac{12-i_1}{12} = \frac{1}{12^2} \sum_{i=1}^{i=11} i = \frac{11*12}{2*12^2} = \frac{11}{24}$$