Lecture

Chapter 4. Induction and Recursion

4.5 Program Correctness

Chapter 5. Counting

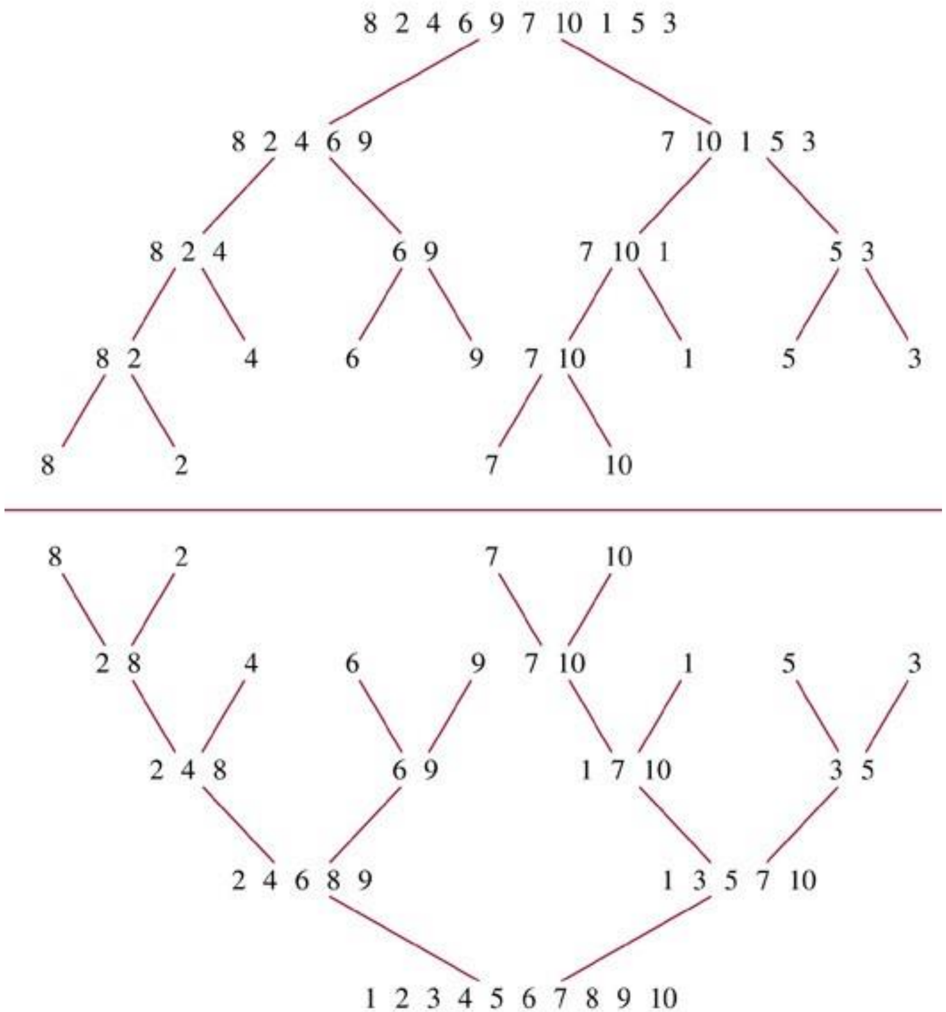
5.1 The Basics of Counting

Quiz

- Develop a recursive procedure for computing the minimum item in a list of integer numbers.

- Given is the recursive definition:
- f(0) = f(1) = 2
- f(n+1) = f(n) * f(n-1)
- Develop a recursive procedure for this definition
- What is your most time-efficient way to compute f(n)?
- What are the complexities of the recursive method and of yours?

Recursive Merge Sort



Split

Merge

Recursive Merge Sort

procedure *mergesort*($L = \ell_1, \ldots, \ell_n$)

if $n > 1$ **then**

$m := \leq\lfloor n/2 f \rfloor$ {this is rough ½-way point}

$L_1 := \ell_1, \ldots, \ell_m$

$L_2 := \ell_{m+1}, \ldots, \ell_n$

$L := merge(mergesort(L_1), mergesort(L_2))$

return L

- The merge takes $\Theta(n)$ steps, and merge-sort takes $\Theta(n \log n)$.

Merging Two Sorted Lists

TABLE 1 Merging the Two Sorted Lists 2, 3, 5, 6 and 1, 4.

First List	*Second List*	*Merged List*	*Comparison*
2 3 5 6	1 4		1 < 2
2 3 5 6	4	1	2 < 4
3 5 6	4	1 2	3 < 4
5 6	4	1 2 3	4 < 5
5 6		1 2 3 4	
		1 2 3 4 5 6	

Recursive Merge Method

procedure *merge*(A, B: sorted lists)

{Given two sorted lists $A = (a_1,\ldots, a_{|A|})$, $B = (b_1,\ldots, b_{|B|})$, return a sorted list of all.}

if A = empty **return** B {If A is empty, it's B.}

if B = empty **return** A {If B is empty, it's A.}

if $a_1 < b_1$ **then**

$L := (a_1, merge((a_2,\ldots, a_{|A|}), B))$

else

$L := (b_1, merge(A, (b_2,\ldots, b_{|B|})))$

return L

Merge Routine

procedure *merge*(A, B: sorted lists)

L = empty list

i:=0, j:=0, k:=0

while $i < |A| \wedge j < |B|$ {$|A|$ is length of A}

if $i=|A|$ **then** $L_k := B_j$; $j := j + 1$

else if $j=|B|$ **then** $L_k := A_i$; $i := i + 1$

else if $A_i < B_j$ **then** $L_k := A_i$; $i := i + 1$

else $L_k := B_j$; $j := j + 1$

$k := k+1$

return L

Takes $\Theta(|A|+|B|)$ time

Program Correctness

- We want to be able to ***prove*** that a given program meets the intended specifications.
- This can often be done manually, or even by *automated program verification* tools.

- A program is ***correct*** if it produces the correct output for every possible input.
- A program is ***partially correct*** if it produces the correct output for every input for which the program eventually halts.

Initial & Final Assertions

- A program's I/O specification can be given using ***initial*** and ***final*** assertions.
  - **The *initial assertion p*** is the condition that the program's input (its initial state) is guaranteed to satisfy (by its user).
  - **The *final assertion q*** is the condition that the output produced by the program (in its final state) is required to satisfy.
- ***Hoare triple*** notation:
  - The notation ***p{S}q*** means that, for all inputs I such that $p(I)$ is true, if program S (given input I) halts and produces output $O = S(I)$, then $q(O)$ is true.
- That is, S is partially correct with respect to specification p, q.

A Trivial Example

- Let S be the program fragment "$y := 2; z := x + y$"
- Let p be the initial assertion "$x = 1$".
- The variable x will hold 1 in all initial states.
- Let q be the final assertion "$z = 3$".
- The variable z must hold 3 in all final states.
- Prove $p\{S\}q$.
- **Proof:** If $x = 1$ in the program's input state, then after running $y := 2$ and $z := x + y$, z will be $1 + 2 = 3$.

Hoare Triple Inference Rules

- Deduction rules for Hoare Triple statements.
- A simple example: the ***composition rule***: $p\{S_1\}q$

$q\{S_2\}r$

$\therefore\ \underline{p\{S_1;\ S_2\}r}$

- **It says:** If program S_1 given condition p produces condition q, and S_2 given q produces r, then the program "S_1 followed by S_2", if given p, yields r.

Inference Rule for *if* Statements

- Program segment that is the conditional statement

    if *condition* **then**

    $\quad S$

- Rule of inference

$$\frac{\begin{array}{l}(p \wedge condition)\{S\}q \\ (p \wedge \neg condition) \rightarrow q\end{array}}{\therefore\ p\{\textbf{if } condition \textbf{ then } S\}q}$$

Initial assertion

Final assertion

- Example: Show that **T** {**if** $x > y$ **then** $y := x$} $y \geq x$.

Proof: When the initial assertion is true and if $x > y$, then the **if** body is executed, which sets $y = x$,
and so afterwards $y \geq x$ is true.
Otherwise, $x \leq y$ and so $y \geq x$. In either case $y \geq x$ is true. So the fragment meets the specification.

if-then-else Rule

- Program segment that is the conditional statement **if** *condition* **then**

S_1
else
S_2

- Rule of inference

$(p \wedge condition)\{S_1\}q$
$(p \wedge \neg condition)\{S_2\}q$

$\therefore\ p\{\textbf{if } condition \textbf{ then } S_1 \textbf{ else } S_2\}q$

- Example: Show that

$\mathbf{T}\ \{\textbf{if } x < 0 \textbf{ then } abs := -x \textbf{ else } abs := x\}\ abs = |x|$

- If $x < 0$ then after the **if** body, abs = -x = $|x|$.

If $\neg(x < 0)$, *i.e.*, $x \geq 0$, then after the **else** body, $abs = x = |x|$. So the rule applies and the program segment is correct.

Loop Invariants

- For a while loop "**while** *condition S*", we say that p is a ***loop invariant*** of this loop if $(p \wedge condition)\{S\}p$.
- If p (and the continuation condition *condition*) is true before executing the body, then p remains true afterwards.
- And so p stays true through *all* subsequent iterations.

- This leads to the inference rule:

p is a loop invariant

$$\frac{(p \wedge condition)\{S\}p}{\therefore\ p\{\textbf{while}\ condition\ S\}(\neg condition \wedge p)}$$

Loop Invariant Example

S $\left\{ \begin{array}{l} i := 1 \\ fact := 1 \\ \textbf{while } i < n \\ \qquad i := i + 1; \\ \qquad fact := fact \cdot i \\ \textbf{end while} \end{array} \right.$

- Prove that the following Hoare triple holds when n is a positive integer: **T** $\{S\}$ $(fact = n!)$
  - **Proof.** Note that p: "$fact = i! \land i \leq n$" is a loop invariant, and is true before the loop. Thus, after the loop we have $(\neg condition \land p) \Leftrightarrow \neg(i < n) \land (fact = i! \land i \leq n) \Leftrightarrow i = n \land fact = i! \Leftrightarrow fact = n!$. ■

Big Example

- S **=** S_1**;** S_2**;** S_3**;** S_4 (compute the product of two integers m, n)

procedure *multiply*(m, n: integers) — p: $m, n \in \mathbf{Z}$

S_1 **if** $n < 0$ **then** $a := -n$ **else** $a := n$ — q: $p \wedge (a = |n|)$

S_2 $k := 0$; $x := 0$ — r: $q \wedge (k = 0) \wedge (x = 0)$

Loop invariant $x = mk \wedge k \leq a$

S_3 **while** $k < a$ {

$x = x + m$; $k = k + 1$

}

Maintains loop invariant:

$x = mk \wedge k \leq a$ $x = mk \wedge k = a$

$\therefore x = ma = m|n|$ — s

$\therefore (n < 0 \wedge x = -mn) \vee (n \geq 0 \wedge x = mn)$

S_4 **if** $n < 0$ **then** *prod* $:= -x$ **else** *prod* $:= x$ — t: $prod = mn$

Chapter 5: Counting

- Combinatorics
- The study of the number of ways to put things together into various combinations.

- *E.g.* In a contest entered by 100 people,
- how many different top-10 outcomes could occur?
- *E.g.* If a password is 6~8 letters and/or digits,
- how many passwords can there be?

Sum and Product Rules

- Let m be the number of ways to do task 1 and n the number of ways to do task 2,
- with each number independent of how the other task is done,
- and also assume that no way to do task 1 simultaneously also accomplishes task 2.
- Then, we have the following rules:
- The ***sum rule***: The task "do either task 1 or task 2, but not both" can be done in $m + n$ ways.
- The ***product rule***: The task "do both task 1 and task 2" can be done in mn ways.

The Sum Rule

- If a task can be done in one of n_1 ways, or in one of n_2 ways, …, or in one of n_m ways, where none of the set of n_i ways of doing the task is the same as any of the set of n_j ways, for all pairs i and j with $1 \leq i < j \leq m$.
- Then the number of ways to do the task is

$n_1 + n_2 + \cdots + n_m$.

The Sum Rule: Example 1

- A student can choose a computer project from one of three lists A, B, and C:
- List A: 23 possible projects
- List B: 15 possible projects
- List C: 19 possible projects
- No project is on more than one list
- How many possible projects are there to choose from?

$$23 + 15 + 19 = 57$$

The Sum Rule: Example 2

- What is the value of k after the following code has been executed?

$k := 0$

for $i_1 := 1$ **to** n_1 $k := k + 1$

for $i_2 := 1$ **to** n_2 $\qquad n_1 + n_2 + \cdots + n_m$

$k := k + 1$

…

for $i_m := 1$ **to** n_m

$k := k + 1$

The Product Rule

- Suppose that a procedure can be broken down into a sequence of *m* successive tasks.

If the task T_1 can be done in n_1 ways;
the task T_2 can then be done in n_2 ways; ...; and
the task T_m can be done in n_m ways, then there are $n_1 \cdot n_2 \cdots n_m$ ways to do the procedure.

The Product Rule: Example

- Show that a set $\{x_1, \ldots, x_n\}$ containing n elements has 2^n subsets.
- A subset can be constructed in

n successive steps:

- Pick or do not pick x_1, pick or do not pick x_2, …, pick or do not pick x_n .
  - Each step can be done in two ways.
- Thus the number of possible subsets is $\underbrace{2 \cdot 2 \cdots 2}_{n \text{ factors}} = 2^n$.

Lecture

Chapter 5. Counting

1. The Basics of Counting
2. The Pigeonhole Principle
3. Permutations and Combinations

Review

- **Sum Rule:** If a task can be done in one of n_1 ways, or in one of n_2 ways, …, or in one of n_m ways, where none of the set of n_i ways of doing the task is the same as any of the set of n_j ways, for all pairs i and j with $1 \le i < j \le m$. Then the number of ways to do the task is $n_1 + n_2 + \cdots + n_m$.
- **Product Rule:** Suppose that a procedure can be broken down into a sequence of m successive tasks. If the task T_1 can be done in n_1 ways; the task T_2 can then be done in n_2 ways; ...; and the task T_m can be done in n_m ways, then there are $n_1 \cdot n_2 \cdots n_m$ ways to do the procedure.

The Product Rule: Example

- Show that a set $\{x_1, \ldots, x_n\}$ containing n elements has 2^n subsets.
  - A subset can be constructed in n successive steps:
    - Pick or do not pick x_1, pick or do not pick x_2, …, pick or do not pick x_n .
  - Each step can be done in two ways.
  - Thus the number of possible subsets is
    $$\underbrace{2 \cdot 2 \cdots 2}_{n \text{ factors}} = 2^n.$$

The Product Rule: Example

- What is the value of k after the following code has been executed?

$k := 0$

for $i_1 := 1$ **to** n_1

  for $i_2 := 1$ **to** n_2

    …

    for $i_m := 1$ **to** n_m

      $k := k + 1$

$n_1 \cdot n_2 \cdots n_m$

The Product Rule: Example

- How many functions are there from a set with *m* elements to one with *n* elements?

$$n^m$$

- How many one-to-one functions are there from a set with *m* elements to one with *n* elements?

$$n \cdot (n-1)(n-2) \cdots (n-m+1)$$

- More examples in the textbook

IP Address Example

- In version 4 of the Internet Protocol (IPv4)
  - Internet address is a string of 32 bits
  - Network number (*netid*) + host number (*hostid*)
  - Valid computer addresses are in one of 3 types:
    - A ***class A*** IP address consists of 0, followed by a 7-bit “netid” $\neq \mathbf{1}^7$, and a 24-bit “hostid”
    - A ***class B*** address has 10, followed by a 14-bit netid and a 16-bit hostid.
    - A ***class C*** address has 110, followed by a 21-bit netid and an 8-bit hostid.
  - The 3 classes have distinct headers (0, 10, 110)
  - Hostids that are all 0s or all 1s are not allowed.

128.171.224.100

IP Address Example

Bit Number	0	1	2	3	4	8	16	24	31
Class A	0	netid				hostid			
Class B	1	0	netid				hostid		
Class C	1	1	0	netid				hostid	
Class D	1	1	1	0	Multicast Address				
Class E	1	1	1	1	0	Address			

- How many valid computer addresses are there?

IP Address Solution

- \# of addresses

  = (# class A) + (# class B) + (# class C)

  (by sum rule)
- \# class A = (# valid netids)$\times$(# valid hostids)

  (by product rule)
- \# valid class A netids = $2^7 - 1 = 127$.
- \# valid class A hostids = $2^{24} - 2 = 16,777,214$.
- Continuing in this fashion we find the answer is:

  3,737,091,842 (3.7 billion IP addresses)

Set Theoretic Version

- If A is the set of ways to do task 1, and B the set of ways to do task 2, and if A and B are disjoint, then:
  - The ways to do either task 1 or 2 are $A \cup B$, and $|A \cup B| = |A| + |B|$
  - The ways to do both task 1 and 2 can be represented as $A \times B$, and $|A \times B| = |A| \cdot |B|$

Inclusion-Exclusion Principle

- Suppose that k out of m ways of doing task 1 also simultaneously accomplish task 2.
  - And there are also n ways of doing task 2.
- Then, the number of ways to accomplish "Do either task 1 or task 2" is $m + n - k$.
- Set theory: If A and B are not disjoint, then $|A \cup B| = |A| + |B| - |A \cap B|$.
  - If they are disjoint, this simplifies to $|A| + |B|$.

Inclusion-Exclusion Example

- Some hypothetical rules for passwords:
  - Passwords must be 2 characters long
  - Each character must be a letter a ~ z, a digit 0 ~ 9, or one of the 10 punctuation characters ! @ # $ % ^ & * ( )
  - Each password must contain ***at least one*** digit or punctuation character

Setup of Problem

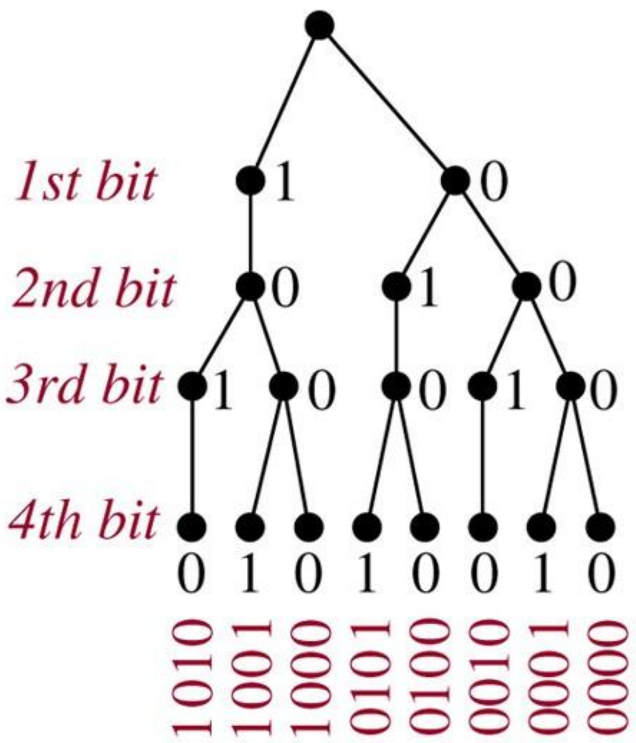
- A legal password has a digit or punctuation character in position 1 **or** position 2.
  - These cases overlap, so the principle applies.
- # of passwords with OK symbol in position #1 $= (10 + 10) \times (10 + 10 + 26) = 20 \times 46 = 920$
- # with OK symbol in pos. #2 $= 46 \times 20 = 920$
- # with OK symbol both places $= 20 \times 20 = 400$
- Answer: $920 + 920 - 400 = 1{,}440$

Tree Diagrams

- A tree diagram can be used in many different counting problems.
- To use trees in counting, we use a branch to represent each possible choice.
- We represent the possible outcomes by the leaves, which are the endpoints of branches not having other branches starting at them.

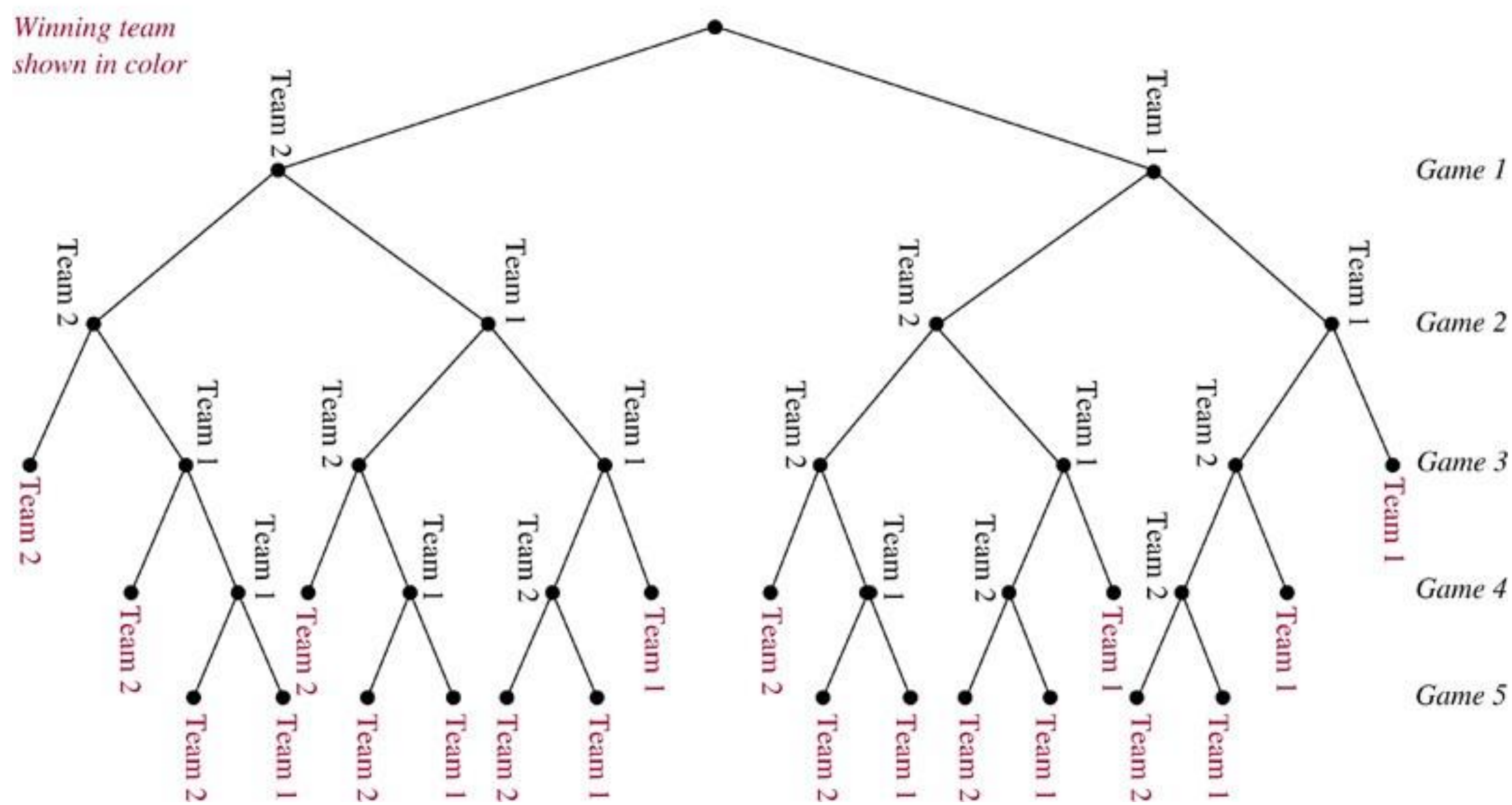
Tree Diagrams: Example

How many bit strings of length four do not have two consecutive 1s?



Tree Diagrams: Example

- A playoff between two teams consists of at most five games. The first team that wins three games wins the playoff. In how many different ways can the playoff occur?

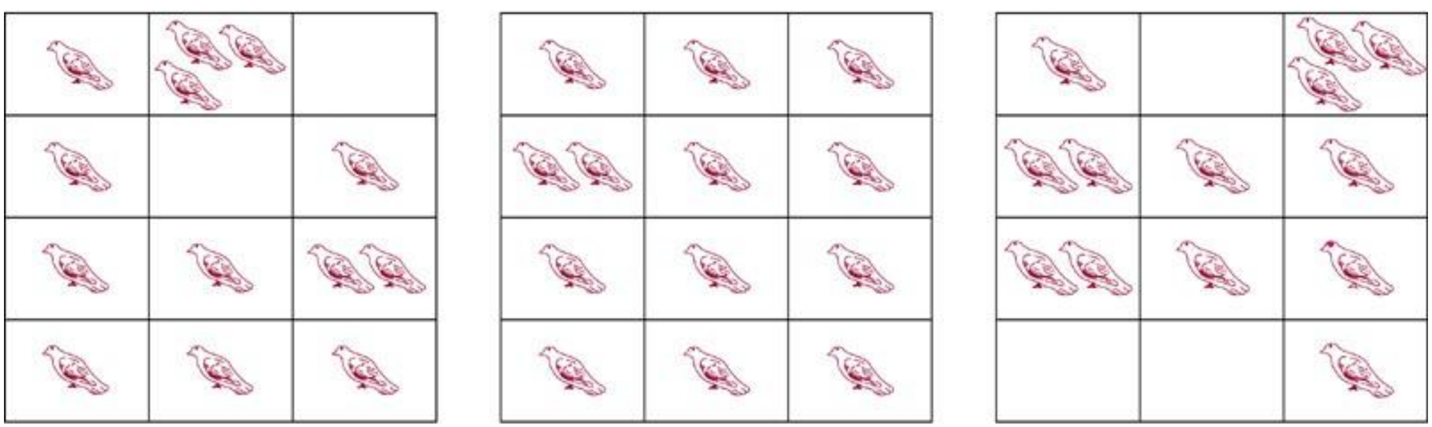


The Pigeonhole Principle

- A.k.a. the "Dirichlet drawer principle" or the "Shoe Box Principle".
- If $k + 1$ or more objects are assigned to k places, then at least 1 place must be assigned 2 or more objects.
- In terms of the assignment function:
  - If $f: A \rightarrow B$ and $|A| \geq |B| + 1$, then some element of B has more than two preimages under f.
    - I.e., f is not one-to-one.

The Pigeonhole Principle

- Proof by contradiction:
  - If the conclusion is false, each pigeonhole contains at most one pigeon and in this time, we can account for at most *k* pigeons.
  - Since there are $k + 1$ pigeons, we have a contradiction.



(a) (b) (c)

Pigeonhole Principle: Example

- There are 101 possible numeric grades (0% ~ 100%) rounded to the nearest integer.
  - Also, there are $>$101 students in a class.
- Therefore, there must be at least one (rounded) grade that will be shared by at least 2 students at the end of the semester.
  - i.e., the function from students to rounded grades is *not* a one-to-one function.

Another Example of P.P.

- 10 persons have first names as Alice, Bernare, and Charles, and last names as Lee, McDuff, and Ng. Show that at least two persons have the same first and last names.
- Solution:
  - 9 possible names for the 10 persons → 10 pigeons and 9 pigeonholes.
  - Assignment of names to people = assignment of pigeonholes to the pigeons
  - By the Pigeonhole Principle, some name (pigeonhole) is assigned to at least two persons (pigeons).

Generalized Pigeonhole Principle

- If *N* objects are assigned to *k* places, then at least one place must be assigned at least $\lceil N/k \rceil$ objects.

- *E.g.*, there are *N* = 280 students in a class. There are *k* = 52 weeks in the year.
  - Therefore, there must be at least 1 week during which at least $\lceil 280/52 \rceil = \lceil 5.38 \rceil = 6$ students in the class have a birthday.

Proof of G.P.P.

- By contradiction.

  Suppose every place has $< \lceil N/k \rceil$ objects,

  thus $\leq \lceil N/k \rceil - 1$.

- Then the total number of objects is at most

$$k\left(\left\lceil \frac{N}{k} \right\rceil - 1\right) < k\left(\left(\frac{N}{k} + 1\right) - 1\right) = k\left(\frac{N}{k}\right) = N$$

- So, there are less than N objects, which contradicts our assumption of N objects! ■

G.P.P. Example I

- Given: There are 280 students in a class.
  - Without knowing anybody's birthday, what is the largest value of *n* for which we can prove using the G.P.P. that at least *n* students must have been born in the same month?
- Answer:

$$\lceil 280/12 \rceil = \lceil 23.3 \rceil = 24$$

G.P.P. Example II

- What is the least number of area codes needed to guarantee that the 25 million phones in a state can be assigned distinct 10-digit telephone numbers?
  - Phone #: NXX-NXX-XXXX
    - N: 2 ~ 9 and X: any digit
- Solution
  - NXX-XXXX: $(8 \cdot 10 \cdot 10) \cdot (10 \cdot 10 \cdot 10 \cdot 10) = 8$ million
  - By G.P.P. at least $\lceil 25{,}000{,}000/8{,}000{,}000 \rceil = 4$ phones have the identical numbers
  - Hence, at least 4 area codes are required

Fun Pigeonhole Proof

- **Example 4:** $\forall n \in \mathbf{N}$, $\exists$ a multiple $m > 0$ of n such that m has only 0's and 1's in its decimal expansion!

- **Proof:** Consider the n+1 decimal integers 1, 11, 111, …, $\underbrace{1\cdots1}_{n+1}$. They have only n possible remainders mod n.

So, take the difference of two that have the same remainder. The result is the answer! □

A Specific Case

- Let n=3. Consider 1, 11, 111, 1111.
  - 1 mod 3 = 1
  - 11 mod 3 = 2
  - 111 mod 3 = 0 ⇇ Lucky extra solution.
  - 1,111 mod 3 = 1

Note same residue

- 1,111 − 1 = 1,110 = 3·370.
  - It has only 0's and 1's in its expansion.
  - Its remainder mod 3 = 0, so it's a multiple of 3.

Baseball Example

- Suppose that next June, the Marlins baseball team plays at least 1 game a day, but ≤ 45 games total. Show there must be some sequence of consecutive days in June during which they play *exactly* 14 games.
  - **Proof:** Let a_j be the number of games played on or before day j. Then, $a_1,\ldots,a_{30} \in \mathbf{Z}^+$ is a sequence of 30 distinct integers with $1 \le a_j \le 45$.
    Therefore $a_1+14,\ldots,a_{30}+14$ is a sequence of 30 distinct integers with $15 \le a_j+14 \le 59$.
    Thus, $(a_1,\ldots,a_{30},a_1+14,\ldots,a_{30}+14)$ is a sequence of 60 integers from the set {1,..,59}.
    By the Pigeonhole Principle, two of them must be equal, but $a_i \neq a_j$ for $i \neq j$. So, $\exists ij$: $a_i = a_j+14$.
    Thus, 14 games were played on days $a_j+1, \ldots, a_i$.

Baseball Problem Illustrated

- Example of $\{a_i\}$: Note all elements are distinct.
  - 1, 2, 4, 5, 7, 8, 10, 11, 13, 14, 16, 17, 19, 21, 22, 23, 25, 27, 29, 30, 31, 33, 34, 36, 37, 39, 40, 41, 43, 45
  - Then $\{a_i+14\}$ is the following sequence: 15, 16, 18, 19, 21, 22, 24, 25, 27, 28, 30, 32, 33, 35, 36, 37, 39, 41, 43, 44, 45, 47, 48, 50, 51, 53, 54, 55, 57, 59
- In any 60 integers from 1-59 there must be some duplicates, indeed we find the following ones:
  - 16, 19, 21, 22, 25, 27, 30, 33, 36, 37, 39, 41, 43, 45

Thus, for example, exactly 14 games were played during days

3 to 11:
2+1+2+1+2+1+2+1+2

Lecture

Chapter 5. Counting

3. Permutations and Combinations
4. Binomial Coefficients
5. Generalized Permutations and Combinations

Permutations

- A ***permutation*** of a set *S* of distinct elements is an **ordered** sequence that contains each element in *S* exactly once.
- E.g. {*A*, *B*, *C*} → six permutations:

ABC, *ACB*, *BAC*, *BCA*, *CAB*, *CBA*

- An **ordered** arrangement of *r* distinct elements of *S* is called an ***r-permutation** of S*.
- The number of *r*-permutations of a set with *n* = |*S*| elements is

$$\boxed{P(n,r) = n(n-1)(n-2)\cdots(n-r+1) = \frac{n!}{(n-r)!}, \quad 0 \le r \le n.}$$

- $P(n,n) = n!/(n-n)! = n!/0! = n!$ (Note: $0! = 1$)

Permutation Examples

- **Example**: Let $S = \{1, 2, 3\}$.
  - The arrangement 3, 1, 2 is a permutation of S ($3! = 6$ ways)
  - The arrangement 3, 2 is a 2-permutation of S ($3 \cdot 2 = 3!/1! = 6$ ways)

- **Example**: There is an armed nuclear bomb planted in your city, and it is your job to disable it by cutting wires to the trigger device. There are 10 wires to the device.

  If you cut exactly the right three wires, in exactly the right order, you will disable the bomb, otherwise it will explode!

  If the wires all look the same, what are your chances of survival?

$P(10,3) = 10 \times 9 \times 8 = 720$,
so there is a 1 in 720 chance that you'll survive!

More Permutation Examples

- **Example 6**: Suppose that a sales woman has to visit eight different cities. She must begin her trip in a specified city, but she can visit the other seven cities in any order she wishes. How many possible orders can the saleswoman use when visiting these cities?

First city is determined, and the remaining seven can be ordered arbitrarily: $7! = 7 \cdot 6 \cdot 5 \cdot 4 \cdot 3 \cdot 2 \cdot 1 = 5040$

- **Example 7**: How many permutations of the letters ABCDEFGH contain the string ABC?

ABC must occur as a block, i.e. consider it as one object
Then, it'll be the number of permutations of six objects (ABC, D, E, F, G, H), which is $6! = 720$

Another Example

- How many ways are there to pick a set of 3 people from a group of 6?
- There are 6 choices for the first person, 5 for the second one, and 4 for the third one, so there are $6 \cdot 5 \cdot 4 = 120$ ways to do this.
- This is not the correct result!
- For example, picking person C, then person A, and then person E leads to the same group as first picking E, then C, and then A.
- However, these cases are counted separately in the above equation.
- So how can we compute how many different subsets of people can be picked (that is, we want to disregard the order of picking)?

Combinations

- **How many different committees of three students can be formed from a group of four students?**
- An ***r-combination*** of elements of a set *S* is an **unordered** selection of *r* elements from the set. Thus, an *r*-combination is simply a subset $T \subseteq S$ with *r* members, $|T| = r$.
- **Example**: S = {1, 2, 3, 4}, then {1, 3, 4} is a 3-combination from S
- **Example**: How many distinct 7-card hands can be drawn from a standard 52-card deck?
- The order of cards in a hand doesn't matter.

Calculate $C(n, r)$

- Consider that we can obtain the *r*-permutation of a set in the following way:
- First, we form all the *r*-combinations of the set (there are $C(n, r)$ such *r*-combinations)
- Then, we generate all possible orderings in each of these *r*-combinations (there are $P(r, r)$ such orderings in each case).
- Therefore, we have:

$$P(n, r) = C(n, r) \cdot P(r, r)$$

$$C(n, r) = \frac{P(n, r)}{P(r, r)} = \frac{n(n-1)\cdots(n-r+1)}{r!}$$

$$= \frac{n!}{(n-r)!} \cdot \frac{1}{r!} = \frac{n!}{r!(n-r)!}$$

Combinations

- The number of *r*-combinations of a set with *n* = |*S*| elements is

$$C(n, r) = \binom{n}{r} = \frac{P(n, r)}{P(r, r)} = \frac{n!/(n-r)!}{r!} = \frac{n!}{r!(n-r)!}$$

- Note that $C(n, r) = C(n, n-r)$
  - Because choosing the *r* members of *T* is the same thing as choosing the (*n* − *r*) non-members of *T*.

$$C(n, n-r) = \binom{n}{n-r} = \frac{n!}{(n-r)![n-(n-r)]!} = \frac{n!}{(n-r)!r!}$$

Combination Example I

- How many distinct 7-card hands can be drawn from a standard 52-card deck?
- The order of cards in a hand doesn't matter.

- Answer:

$C(52, 7) = P(52, 7) \,/\, P(7, 7) = 52! \,/\, (7!\cdot 45!)$

$= (52\cdot 51\cdot 50\cdot 49\cdot 48\cdot 47\cdot 46) \,/\, (7\cdot 6\cdot 5\cdot 4\cdot 3\cdot 2\cdot 1)$

17 10 7 $\cancel{8}$

$\cancel{2}$

$52\cdot 17\cdot 10\cdot 7\cdot 47\cdot 46 = 133{,}784{,}560$

Combination Example II

- $C(4, 3) = 4$, since, for example, the 3-combinations of a set {1, 2, 3, 4} are {1, 2, 3}, {1, 3, 4}, {2, 3, 4}, {1, 2, 4}.
- $C(4, 3) = P(4, 3) / P(3, 3) = 4! / (3! \times 1!)$

$= (4 \times 3 \times 2) / (3 \times 2 \times 1) = 4$

- How many ways are there to pick a set of 3 people from a group of 6 (disregarding the order of picking)?
- $C(6, 3) = 6! / (3! \times 3!)$

$= (6 \times 5 \times 4) / (3 \times 2 \times 1) = 20$

- There are 20 different groups to be picked

Combination Example III

- A soccer club has 8 female and 7 male members. For today's match, the coach wants to have 6 female and 5 male players on the grass. How many possible configurations are there?
- $C(8, 6) \times C(7, 5)$

$= \{P(8, 6) / P(6, 6)\} \times \{P(7, 5) / P(5, 5)\}$

$= \{8!/ (2! \times 6!)\} \times \{7!/ (2! \times 5!)\}$

$= \{(8 \times 7) / 2!\} \times \{(7 \times 6) / 2!\}$

$= 28 \times 21$

$= 588$

Binomial Coefficients

- Expressions of the form $C(n, r)$ are also called ***binomial coefficients***
- Coefficients of the expansion of powers of binomial expressions
- Binomial expression is a simply the sum of two terms such as $x + y$
- Example:

$$(x+y)^3 = (x+y)(x+y)(x+y)$$

$$= (xx + xy + yx + yy)(x+y)$$

$$= xxx + xxy + xyx + xyy + yxx + yxy + yyx + yyy$$

$$= C(3,0)x^3 + C(3,1)x^2 y + C(3,2)xy^2 + C(3,3)\, y^3$$

$$= x^3 + 3x^2 y + 3xy^2 + y^3$$

The Binomial Theorem

THE BINOMIAL THEOREM Let x and y be variables, and let n be a nonnegative integer. Then

$$(x+y)^n = \sum_{j=0}^{n} \binom{n}{j} x^{n-j} y^j = \binom{n}{0} x^n + \binom{n}{1} x^{n-1} y + \cdots + \binom{n}{n-1} x y^{n-1} + \binom{n}{n} y^n.$$

Proof: We use a combinatorial proof. The terms in the product when it is expanded are of the form $x^{n-j}y^j$ for $j = 0, 1, 2, \ldots, n$. To count the number of terms of the form $x^{n-j}y^j$, note that to obtain such a term it is necessary to choose $n - j$ xs from the n sums (so that the other j terms in the product are ys). Therefore, the coefficient of $x^{n-j}y^j$ is $\binom{n}{n-j}$, which is equal to $\binom{n}{j}$. This proves the theorem. ◁

Examples

- $(a + b)^9 \rightarrow$ the coefficient of $a^5b^4 = C(9, 4)$
- The coefficient of $x^{12}y^{13}$ in the expansion of $(2x - 3y)^{25}$
- By binomial theorem

$$(2x + (-3y))^{25} = \sum_{j=0}^{25} \binom{25}{j} (2x)^{25-j} (-3y)^j$$

- The coefficient of $x^{12}y^{13}$ is obtained when $j = 13$

$$C(25,13) \cdot 2^{12} \cdot (-3)^{13} = -\frac{25!}{13! \cdot 12!} 2^{12} 3^{13}$$

- $(x + y + z)^9 \rightarrow$ the coefficient of $x^2y^3z^4 = C(9, 2) \cdot C(7, 3).C(9,4)$.

COROLLARY 1

COROLLARY 1

Let n be a nonnegative integer. Then

$$\sum_{k=0}^{n} \binom{n}{k} = 2^n.$$

Proof: Using the binomial theorem with $x = 1$ and $y = 1$, we see that

$$2^n = (1+1)^n = \sum_{k=0}^{n} \binom{n}{k} 1^k 1^{n-k} = \sum_{k=0}^{n} \binom{n}{k}.$$

This is the desired result. ◁

There is also a nice combinatorial proof of Corollary 1, which we now present.

Proof: A set with n elements has a total of 2^n different subsets. Each subset has zero elements, one element, two elements, . . . , or n elements in it. There are $\binom{n}{0}$ subsets with zero elements, $\binom{n}{1}$ subsets with one element, $\binom{n}{2}$ subsets with two elements, . . . , and $\binom{n}{n}$ subsets with n elements. Therefore,

$$\sum_{k=0}^{n} \binom{n}{k}$$

counts the total number of subsets of a set with n elements. By equating the two formulas we have for the number of subsets of a set with n elements, we see that

$$\sum_{k=0}^{n} \binom{n}{k} = 2^n.$$

◁

COROLLARY 2

COROLLARY 2

Let n be a positive integer. Then

$$\sum_{k=0}^{n}(-1)^k\binom{n}{k} = 0.$$

Proof: When we use the binomial theorem with $x = -1$ and $y = 1$, we see that

$$0 = 0^n = ((-1)+1)^n = \sum_{k=0}^{n}\binom{n}{k}(-1)^k 1^{n-k} = \sum_{k=0}^{n}\binom{n}{k}(-1)^k.$$

This proves the corollary.

Remark: Corollary 2 implies that

$$\binom{n}{0}+\binom{n}{2}+\binom{n}{4}+\cdots=\binom{n}{1}+\binom{n}{3}+\binom{n}{5}+\cdots.$$

COROLLARY 3

Let n be a nonnegative integer. Then

$$\sum_{k=0}^{n} 2^k \binom{n}{k} = 3^n.$$

Proof: We recognize that the left-hand side of this formula is the expansion of $(1+2)^n$ provided by the binomial theorem. Therefore, by the binomial theorem, we see that

$$(1+2)^n = \sum_{k=0}^{n} \binom{n}{k} 1^{n-k} 2^k = \sum_{k=0}^{n} \binom{n}{k} 2^k.$$

Hence

$$\sum_{k=0}^{n} 2^k \binom{n}{k} = 3^n.$$

◁

Generalized Permutations Combinations

- Permutations and combinations allowing repetitions.
  - How many strings of length *r* can be formed from the English alphabet?
  - How many different ways are possible when we select a dozen donuts from a box that contains four different kinds of donuts?
- Permutations where not all objects are distinguishable.
  - The number of ways we can rearrange the letters of the word *MISSISSIPPI*

Permutations with Repetitions

- **<u>Theorem 1</u>**: The number of *r*-permutations of n objects with repetition allowed is n^r.
- Proof: There are *n* ways to select an element of the set for each of *r* positions with repetition allowed. By the product rule, the answer is given as *r* multiples of *n*.

- **Example**: How many strings of length *r* can be formed from the English alphabet?
- Answer: 26^r

Combinations with Repetitions

- An example
- How many ways are there to select four pieces of fruit from a bowl containing apples, oranges, and pears if there are at least four pieces of each type of fruit in the bowl?
- In this case, the order in which the pieces are selected does not matter, only the types of fruit, not the individual piece, matter.

Combinations with Repetitions

- Example Rephrased: The number of 4-combinations with repetition allowed from a 3-element set {apple, orange, pear}
  - All four in same type: 4 apples, 4 oranges, 4 pears [3 ways]
  - Three in same type: two cases for each of 3 apples, 3 oranges, 3 pears [2*3=6 ways]
  - Two diff. pairs with each pair in same type [3 ways]
  - Only one pair in same type [3 ways]
  - <u>Total 15 ways</u>
- Can be generalized:
  - The number of ways to fill 4 slots from 3 categories with repetition allowed

Combinations with Repetitions

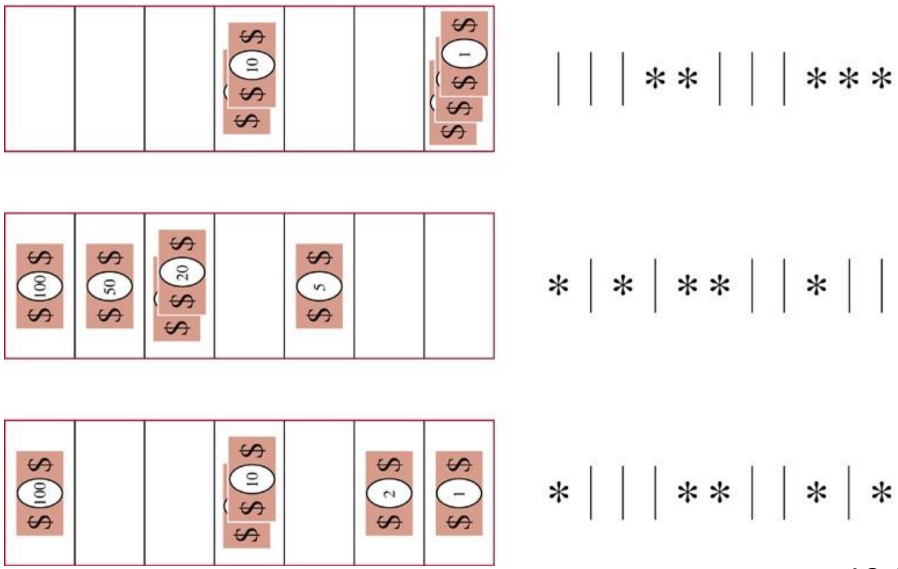
4 apples
3 apples, 1 orange
3 oranges, 1 pear
2 apples, 2 oranges
2 apples, 1 orange, 1 pear

4 oranges
3 apples, 1 pear
3pears, 1 apple
2 apples, 2 pears
2 oranges, 1apple, 1 pear

4 pears
3 oranges, 1 apple
3 pears, 1 orange
2 pears, 2 oranges
1apple, 1 orange, 2 pears

Example

- How many ways are there to select five bills from a cash box containing $1 bills, $2 bills, $5 bills, $10 bills, $20 bills, $50 bills, and $100 bills?
- The order in which the bills are chosen doesn't matter
- The bills of each denomination are indistinguishable
- At least five bills of each type



- $C(7–1+5, 5)$

$= C(11, 5)$

$= 11! / (5!·6!)$

$= 462$

Combinations with Repetitions

- **<u>Theorem 2</u>**: The number of *r*-combinations from a set with *n* elements with repetition allowed is:

$$\boxed{C(n+r-1,r)=C(n+r-1,n-1)}$$

- Other representations with the same meaning
- # of ways to fill *r* slots from *n* categories with repetition allowed
- # of ways to select *r* elements from *n* categories of elements with repetition allowed

Proof of Theorem 2

- Represent each *r*-combinations from a set with *n* elements with repetition allowed by <u>a list of $n - 1$ bars and r stars</u>.
- $n - 1$ bars: used to mark off n different cells (categories)
- r stars: each star in *i-th* cell (if any) represents an element that is selected for the *i-th* category

- # of different lists that containing $n - 1$ bars and r stars
- = # of ways to chose the r positions to place the r stars from $n + r - 1$ positions [$\mathbf{C(n + r - 1, r)}$]
- = # of ways to chose the $n - 1$ positions to place the $n - 1$ bars from $n + r - 1$ positions [$\mathbf{C(n + r - 1, n - 1)}$]

More Examples

- How many ways can I fill a box holding 100 pieces of candy from 30 different types of candy?
  - <u>Solution:</u> Here #stars = 100, #bars = 30 – 1, so there are $C(100+29,100) = 129! / (100! \cdot 29!)$ different ways to fill the box.

- How many ways if I must have at least 1 piece of each type?
  - <u>Solution:</u> Now, we are reducing the #stars to choose over to (100 – 30) stars, so there are $C(70+29, 70) = 99! / (70!29!)$

When to Use Generalized Combinations

- Besides categorizing a problem based on its order and repetition requirements as a generalized combination, there are a couple of other characteristics which help us sort:
  - In generalized combinations, having all the slots filled in by only selections from one category is allowed;
  - It is possible to have more slots than categories.

More Integer Solutions & Restrictions

- How many integer solutions are there to:

$$a + b + c + d = 15,$$

when $a \geq -3$, $b \geq 0$, $c \geq -2$ and $d \geq -1$?

- In this case, we alter the restrictions and equation so that the restrictions "go away." To do this, we need each restriction ≥ 0 and balance the number of slots accordingly.
- Hence $a \geq -3+3$, $b \geq 0$, $c \geq -2+2$ and $d \geq -1+1$, yields $a + b + c + d = 15+3+2+1 = 21$
- So, there are C(21+4–1,21) = C(24,21) = C(24,3) = (24x23x22)/(3x2) = 2024 solutions.

Distributing Objects into Distinguishable Boxes

- Distinguishable (or labeled) objects to distinguishable boxes
- How many ways are there to distribute hands of 5 cards to each of four players from the standard deck of 52 cards?

C(52,5)C(47,5)C(42,5)C(37,5)

- Indistinguishable (or unlabeled) objects to distinguishable boxes
- How many ways are there to place 10 indistinguishable balls into 8 distinguishable bins?

C(8+10-1, 10) = C(17,10) = 17! / (10!7!)

Distributing Distinguishable Objects into Indistinguishable Boxes

- How many ways are there to put <u>4 different employees</u> into <u>3 indistinguishable offices</u>, when each office can contain any number of employees?
- all four in one office: C(4,4) = 1
- three + one: C(4,3) = 4
- two + two: C(4,2)/2 = 3
- two + one + one: C(4,2) = 6

Distributing Indistinguishable Objects into Indistinguishable Boxes

- How many ways are there to pack 6 copies of same book into 4 identical boxes, where each box can contain as many as six books?
- List # of books in each box with the largest # of books, followed by #s of books in each box containing at least 1 book, in order of decreasing # of books in a box.

6

5,1

4,2 4,1,1

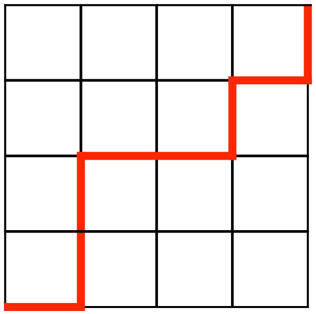
3,3 3,2,1 3,1,1,1

2,2,2 2,2,1,1

Another Combination Example

- How many routes are there from the lower left corner of an $n \times n$ square grid to the upper right corner if we are restricted to traveling only to the right or upward.
- Solution

R: right
U: up



- route $\rightarrow$ *RUURRURU* : a string of *n R*'s and *n U*'s
- Any such string can be obtained by selecting *n* positions for the *R*'s, without regard to the order of selection, from among the 2*n* available positions in the string and then filling the remaining position with *U*'s.
- Thus there are C(2*n*,*n*) possible routes.