Advanced Counting Techniques

Chapter 8

With Question/Answer Animations

1

Chapter Summary

Applications of Recurrence Relations
Solving Linear Recurrence Relations
- Homogeneous Recurrence Relations
- Nonhomogeneous Recurrence Relations

Divide-and-Conquer Algorithms and Recurrence Relations
Generating Functions
Inclusion-Exclusion
Applications of Inclusion-Exclusion

2

Applications of Recurrence Relations

Section 8.1

3

Section Summary

Applications of Recurrence Relations
- Fibonacci Numbers
- The Tower of Hanoi
- Counting Problems

4

Recurrence Relations (recalling definitions from Chapter 2)

Definition: A recurrence relation for the sequence $\{a_n\}$ is an equation that expresses a_n in terms of one or more of the previous terms of the sequence, namely,

$$a_0, a_1, \ldots, a_{n-1}$$

for all integers n with $n \geq n_0$, where n_0 is a nonnegative integer.

- A sequence is called a solution of a recurrence relation if its terms satisfy the recurrence relation.
- The initial conditions for a sequence specify the terms that precede the first term where the recurrence relation takes effect.

5

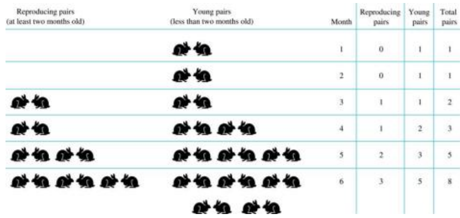
Rabbits and the Fibonacci Numbers

Example: A young pair of rabbits (one of each gender) is placed on an island. A pair of rabbits does not breed until they are 2 months old. After they are 2 months old, each pair of rabbits produces another pair each month. Find a recurrence relation for the number of pairs of rabbits on the island after n months, assuming that rabbits never die.

This is the original problem considered by Leonardo Pisano (Fibonacci) in the thirteenth century.

6

Rabbits and the Fibonacci Numbers



Month	Reproducing pairs	Young pairs	Total pairs
1	0	1	1
2	0	1	1
3	1	1	2
4	1	2	3
5	2	3	5
6	3	5	8

Modeling the Population Growth of Rabbits on an Island

Jump to long description

7

Rabbits and the Fibonacci Numbers

Solution: Let f_n be the number of pairs of rabbits after n months.

- There is $f_1 = 1$ pair of rabbits on the island at the end of the first month.
- Also $f_2 = 1$ since the pair does not breed during the first month.
- To find the number of pairs on the island after n months, add the number on the island after the previous month, f_{n-1}, and the number of newborn pairs, which equals f_{n-2}, because each newborn pair comes from a pair at least two months old.

Hence, the sequence $\{f_n\}$ satisfies the recurrence relation

$$f_n = f_{n-1} + f_{n-2} \quad \text{for } n \geq 3$$

with the initial conditions $f_1 = 1$ and $f_2 = 1$.

The number of pairs of rabbits on the island after n months is given by the n^{th} Fibonacci number.

8

The Tower of Hanoi

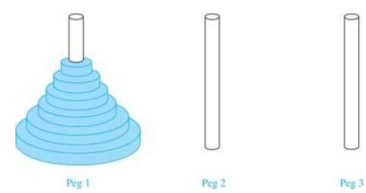
In the late nineteenth century, the French mathematician Édouard Lucas invented a puzzle consisting of three pegs on a board with disks of different sizes. Initially all of the disks are on the first peg in order of size, with the largest on the bottom.

Rules: You are allowed to move the disks one at a time from one peg to another as long as a larger disk is never placed on a smaller.

Goal: Using allowable moves, end up with all the disks on the second peg in order of size with largest on the bottom.

9

The Tower of Hanoi

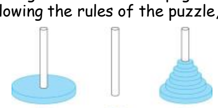


The Initial Position in the Tower of Hanoi Puzzle

10

The Tower of Hanoi

Solution: Let $\{H_n\}$ denote the number of moves needed to solve the Tower of Hanoi Puzzle with n disks. Set up a recurrence relation for the sequence $\{H_n\}$. Begin with n disks on peg 1. We can transfer the top n −1 disks, following the rules of the puzzle, to peg 3 using H_{n-1} moves.



Next, we use 1 move to transfer the largest disk to the second peg. Then we transfer the n −1 disks from peg 3 to peg 2 using H_{n-1} additional moves. This cannot be done in fewer steps. Hence,

$$H_n = 2H_{n-1} + 1.$$

The initial condition is $H_1 = 1$ since a single disk can be transferred from peg 1 to peg 2 in one move.

11

The Tower of Hanoi

We can use an iterative approach to solve this recurrence relation by repeatedly expressing H_n in terms of the previous terms of the sequence.

$$\begin{aligned} H_n &= 2H_{n-1} + 1 \\ &= 2(2H_{n-2} + 1) + 1 = 2^2 H_{n-2} + 2 + 1 \\ &= 2^2(2H_{n-3} + 1) + 2 + 1 = 2^3 H_{n-3} + 2^2 + 2 + 1 \\ &\vdots \\ &= 2^{n-1}H_1 + 2^{n-2} + 2^{n-3} + \ldots + 2 + 1 \\ &= 2^{n-1} + 2^{n-2} + 2^{n-3} + \ldots + 2 + 1 \quad \textit{because } H_1 = 1 \\ &= 2^n - 1 \quad \textit{usingthe formula for the sum of the terms of a geometricseries} \end{aligned}$$

- There was a myth created with the puzzle. Monks in a tower in Hanoi are transferring 64 gold disks from one peg to another following the rules of the puzzle. They move one disk each day. When the puzzle is finished, the world will end.
- Using this formula for the 64 gold disks of the myth,
  $2^{64} - 1$ = 18,446,744,073,709,551,615
  days are needed to solve the puzzle, which is more than 500 billion years.
- Reve's puzzle (proposed in 1907 by Henry Dudeney) is similar but has 4 pegs. There is a well-known unsettled conjecture for the minimum number of moves needed to solve this puzzle. (see Exercises 38-45)

12

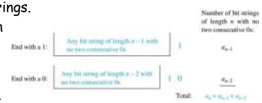
Counting Bit Strings

Example 3: Find a recurrence relation and give initial conditions for the number of bit strings of length n without two consecutive 0s. How many such bit strings are there of length five?

Solution: Let a_n denote the number of bit strings of length n without two consecutive 0s. To obtain a recurrence relation for $\{a_n\}$ note that the number of bit strings of length n that do not have two consecutive 0s is the number of bit strings ending with a 0 plus the number of such bit strings ending with a 1.

Now assume that $n \geq 3$.

- The bit strings of length n ending with 1 without two consecutive 0s are the bit strings of length n −1 with no two consecutive 0s with a 1 at the end. Hence, there are a_{n-1} such bit strings.
- The bit strings of length n ending with 0 without two consecutive 0s are the bit strings of length n − 2 with no two consecutive 0s with 10 at the end. Hence, there are a_{n-2} such bit strings.



We conclude that $a_n = a_{n-1} + a_{n-2}$ for $n \geq 3$.

Jump to long description

13

Bit Strings

The initial conditions are

- $a_1 = 2$, since both the bit strings 0 and 1 do not have consecutive 0s
- $a_2 = 3$, since the bit strings 01, 10, and 11 do not have consecutive 0s, while 00 does

To obtain a_5, we use the recurrence relation three times to find that

- $a_3 = a_2 + a_1 = 3 + 2 = 5$
- $a_4 = a_3 + a_2 = 5 + 3 = 8$
- $a_5 = a_4 + a_3 = 8 + 5 = 13$

Note that $\{a_n\}$ satisfies the same recurrence relation as the Fibonacci sequence. Since $a_1 = f_3$ and $a_2 = f_4$, we conclude that $a_n = f_{n+2}$

14

Counting the Ways to Parenthesize a Product

Example: Find a recurrence relation for C_n, the number of ways to parenthesize the product of n + 1 numbers, $x_0 \cdot x_1 \cdot x_2 \cdot \cdots \cdot x_n$, to specify the order of multiplication. For example, $C_3 = 5$, since all the possible ways to parenthesize 4 numbers are

$$((x_0 \cdot x_1) \cdot x_2) \cdot x_3,\ (x_0 \cdot (x_1 \cdot x_2)) \cdot x_3,\ (x_0 \cdot x_1) \cdot (x_2 \cdot x_3), x_0 \cdot ((x_1 \cdot x_2) \cdot x_3),\ \ x_0 \cdot (x_1 \cdot (x_2 \cdot x_3))$$

Solution: Note that however parentheses are inserted in $x_0 \cdot x_1 \cdot x_2 \cdot \cdots \cdot x_n$, one "·" operator remains outside all parentheses. This final operator appears between two of the n + 1 numbers, say x_k and x_{k+1}. Since there are C_k ways to insert parentheses in the product $x_0 \cdot x_1 \cdot x_2 \cdot \cdots \cdot x_k$ and C_{n-k-1} ways to insert parentheses in the product $x_{k+1} \cdot x_{k+2} \cdot \cdots \cdot x_n$, we have

$$C_n = C_0 C_{n-1} + C_1 C_{n-2} + \cdots + C_{n-2} C_1 + C_{n-1} C_0 = \sum_{k=0}^{n-1} C_k C_{n-k-1}$$

The initial conditions are $C_0 = 1$ and $C_1 = 1$.

The sequence $\{C_n\}$ is the sequence of **Catalan Numbers**. This recurrence relation can be solved using the method of generating functions.

15

Solving Linear Recurrence Relations

Section 8.2

16

Section Summary

Linear Homogeneous Recurrence Relations

Solving Linear Homogeneous Recurrence Relations with Constant Coefficients.

Solving Linear Nonhomogeneous Recurrence Relations with Constant Coefficients.

17

Linear Homogeneous Recurrence Relations

Definition: A linear homogeneous recurrence relation of degree k with constant coefficients is a recurrence relation of the form $a_n = c_1 a_{n-1} + c_2 a_{n-2} + \ldots + c_k a_{n-k}$, where $c_1, c_2, \ldots, c_k$ are real numbers, and $c_k \neq 0$

- it is linear because the right-hand side is a sum of the previous terms of the sequence each multiplied by a function of n
- it is homogeneous because no terms occur that are not multiples of the a_j's. Each coefficient is a constant
- the degree is k because a_n is expressed in terms of the previous k terms of the sequence

By strong induction, a sequence satisfying such a recurrence relation is uniquely determined by the recurrence relation and the k initial conditions $a_0 = C_1$, $a_0 = C_1, \ldots, a_{k-1} = C_{k-1}$.

18

Examples of Linear Homogeneous Recurrence Relations

$P_n = (1.11) P_{n-1}$ linear homogeneous recurrence relation of degree one

$f_n = f_{n-1} + f_{n-2}$ linear homogeneous recurrence relation of degree two

$a_n = a_{n-1} + a^2_{n-2}$ not linear

$H_n = 2H_{n-1} + 1$ not homogeneous

$B_n = nB_{n-1}$ coefficients are not constants

19

Solving Linear Homogeneous Recurrence Relations

The basic approach is to look for solutions of the form $a_n = r^n$, where r is a constant.

Note that $a_n = r^n$ is a solution to the recurrence relation

$a_n = c_1 a_{n-1} + c_2 a_{n-2} + \cdots + c_k a_{n-k}$ if and only if

$r^n = c_1 r^{n-1} + c_2 r^{n-2} + \cdots + c_k r_{n-k}$.

Algebraic manipulation yields the characteristic equation:

$$r^k - c_1 r^{k-1} - c_2 r^{k-2} - \cdots - c_{k-1} r - c_k = 0$$

The sequence $\{a_n\}$ with $a_n = r^n$ is a solution if and only if r is a solution to the characteristic equation.

The solutions to the characteristic equation are called the characteristic roots of the recurrence relation. The roots are used to give an explicit formula for all the solutions of the recurrence relation.

20

Solving Linear Homogeneous Recurrence Relations of Degree Two

Theorem 1: Let c_1 and c_2 be real numbers. Suppose that $r^2 - c_1 r - c_2 = 0$ has two distinct roots: r_1 and r_2. Then the sequence $\{a_n\}$ is a solution to the recurrence relation

$$a_n = c_1 a_{n-1} + c_2 a_{n-2}$$

if and only if

$$a_n = \alpha_1 r_1^n + \alpha_2 r_2^n$$

for n = 0,1,2,…, where α_1 and α_2 are constants.

21

Using Theorem 1

Example: What is the solution to the recurrence relation

$a_n = a_{n-1} + 2a_{n-2}$ with $a_0 = 2$ and $a_1 = 7$?

Solution: The characteristic equation is $r^2 - r - 2 = 0$.

Its roots are r = 2 and r = −1 . Therefore, $\{a_n\}$ is a solution to the recurrence relation if and only if

$$a_n = \alpha_1 2^n + \alpha_2 (-1)^n$$

for some constants α_1 and α_2

To find the constants α_1 and α_2, note that

$a_0 = 2 = \alpha_1 + \alpha_2$ and $a_1 = 7 = \alpha_1 2 + \alpha_2(-1)$

Solving these equations, we find that $\alpha_1 = 3$ and $\alpha_2 = -1$

Hence, the solution is the sequence $\{a_n\}$ with

$a_n = 3 \cdot 2^n - (-1)^n$

22

An Explicit Formula for the Fibonacci Numbers

We can use Theorem 1 to find an explicit formula for the Fibonacci numbers. The sequence of Fibonacci numbers satisfies the recurrence relation $f_n = f_{n-1} + f_{n-2}$ with the initial conditions: $f_0 = 0$ and $f_1 = 1$.

Solution: The roots of the characteristic equation $r^2 - r - 1 = 0$ are

$$r_1 = \frac{1+\sqrt{5}}{2}$$

$$r_2 = \frac{1-\sqrt{5}}{2}$$

23

Fibonacci Numbers

Therefore, by Theorem 1

$$f_n = \alpha_1 \left(\frac{1+\sqrt{5}}{2}\right)^n + \alpha_2 \left(\frac{1-\sqrt{5}}{2}\right)^n$$

for some constants α_1 and α_2.

Using the initial conditions $f_0 = 0$ and $f_1 = 1$, we have

$$f_0 = \alpha_1 + \alpha_2 = 0$$

$$f_1 = \alpha_1 \left(\frac{1+\sqrt{5}}{2}\right) + \alpha_2 \left(\frac{1-\sqrt{5}}{2}\right) = 1.$$

Solving, we obtain $\alpha_1 = \frac{1}{\sqrt{5}}$, $\alpha_2 - \frac{1}{\sqrt{5}}$.

Hence, $f_n = \frac{1}{\sqrt{5}} \left(\frac{1+\sqrt{5}}{2}\right)^n - \frac{1}{\sqrt{5}} \left(\frac{1+\sqrt{5}}{2}\right)^n$.

24

The Solution when there is a Repeated Root

Theorem 2: Let c_1 and c_2 be real numbers with $c_2 \neq 0$. Suppose that $r^2 - c_1r - c_2 = 0$ has one repeated root r_0. Then the sequence $\{a_n\}$ is a solution to the recurrence relation $a_n = c_1a_{n-1} + c_2a_{n-2}$ if and only if

$$a_n = \alpha r_0^n + \alpha_2 n r_0^n$$

for n = 0,1,2,… , where α_1 and α_2 are constants.

25

Using Theorem 2

Example: What is the solution to the recurrence relation $a_n = 6a_{n-1} - 9a_{n-2}$ with $a_0 = 1$ and $a_1 = 6$?

Solution: The characteristic equation is $r^2 - 6r + 9 = 0$. The only root is r = 3. Therefore, $\{a_n\}$ is a solution to the recurrence relation if and only if

$$a_n = \alpha_1 3^n + \alpha_2 n(3)^n$$

where α_1 and α_2 are constants.

To find the constants α_1 and α_2, note that

$a_0 = 1 = \alpha_1$ and $a_1 = 6 = \alpha_1 \cdot 3 + \alpha_2 \cdot 3$

Solving, we find that $\alpha_1 = 1$ and $\alpha_2 = 1$

Hence, $a_n = 3^n + n3^n$

26

Divide-and-Conquer Algorithms and Recurrence Relations

Section 8.3

27

Section Summary

Divide-and-Conquer Algorithms and Recurrence Relations

Examples

- Binary Search
- Merge Sort
- Fast Multiplication of Integers

Master Theorem

Closest Pair of Points (not covered yet in these slides)

28

Divide-and-Conquer Algorithmic Paradigm

Definition: A divide-and-conquer algorithm works by first dividing a problem into one or more instances of the same problem of smaller size and then conquering the problem using the solutions of the smaller problems to find a solution of the original problem.

Examples:

- Binary search, covered in Chapters 3 and 5: It works by comparing the element to be located to the middle element. The original list is then split into two lists and the search continues recursively in the appropriate sublist.
- Merge sort, covered in Chapter 5: A list is split into two approximately equal sized sublists, each recursively sorted by merge sort. Sorting is done by successively merging pairs of lists.

29

Divide-and-Conquer Recurrence Relations

Suppose that a recursive algorithm divides a problem of size n into a subproblems.

Assume each subproblem is of size n/b.

Suppose g(n) extra operations are needed in the conquer step.

Then f(n) represents the number of operations to solve a problem of size n satisfies the following recurrence relation:

$$f(n) = af(n/b) + g(n)$$

This is called a divide-and-conquer recurrence relation.

30

Example: Binary Search

Binary search reduces the search for an element in a sequence of size n to the search in a sequence of size n/2. Two comparisons are needed to implement this reduction:

- one to decide whether to search the upper or lower half of the sequence and
- the other to determine if the sequence has elements

Hence, if f(n) is the number of comparisons required to search for an element in a sequence of size n, then

$$f(n) = f(n/2) + 2$$

when n is even.

31

Example: Merge Sort

The merge sort algorithm splits a list of n (assuming n is even) items to be sorted into two lists with n/2 items. It uses fewer than n comparisons to merge the two sorted lists.

Hence, the number of comparisons required to sort a sequence of size n, is no more than M(n) where

$$M(n) = 2M(n/2) + n.$$

32

Example: Fast Multiplication of Integers

An algorithm for the fast multiplication of two 2n-bit integers (assuming n is even) first splits each of the 2n-bit integers into two blocks, each of n bits.

Suppose that a and b are integers with binary expansions of length 2n. Let

$a = (a_{2n-1}a_{2n-2} \ldots a_1a_0)_2$ and $b = (b_{2n-1}b_{2n-2} \ldots b_1b_0)_2$

Let $a = 2^nA_1 + A_0$, $b = 2^nB_1 + B_0$, where

$A_1 = (a_{2n-1} \ldots a_{n+1}a_n)_2$, $A_0 = (a_{n-1} \ldots a_1a_0)_2$,

$B_1 = (b_{2n-1} \ldots b_{n+1}b_n)_2$, $B_0 = (b_{n-1} \ldots b_1b_0)_2$.

The algorithm is based on the fact that ab can be rewritten as:

$ab = (2^{2n} + 2^n)A_1B_1 + 2^n(A_1 - A_0)(B_0 - B_1) + (2^n + 1)A_0B_0$.

This identity shows that the multiplication of two 2n-bit integers can be carried out using three multiplications of n-bit integers, together with additions, subtractions, and shifts.

Hence, if f(n) is the total number of operations needed to multiply two n-bit integers, then $f(2n) = 3f(n) + Cn$
where Cn represents the total number of bit operations; the additions, subtractions and shifts that are a constant multiple of n-bit operations.

33

Estimating the Size of Divide-and-conquer Functions

Theorem 2. Master Theorem: Let f be an increasing function that satisfies the recurrence relation

$$f(n) = af(n/b) + cn^d$$

whenever n = b^k, where k is a positive integer greater than 1, and c and d are real numbers with c positive and d nonnegative. Then

$$f(n) \text{ is } \begin{cases} O(n^d) & \text{if } a < b^d, \\ O(n^d \log n) & \text{if } a = b^d, \\ O(n^{\log_b a}) & \text{if } a > b^d. \end{cases}$$

34

Complexity of Binary Search

Binary Search Example: Give a big-*O* estimate for the number of comparisons used by a binary search.

Solution: Since the number of comparisons used by binary search is f(n) = f(n/2) + 2 where n is even, by the Master Theorem, it follows that f(n) is *O*(log n) since

a = 1, b = 2, and d = 0 and a = b^d ($1 = 2^0$)

35

Complexity of Merge Sort

Merge Sort Example: Give a big-*O* estimate for the number of comparisons used by merge sort.

Solution: Since the number of comparisons used by merge sort to sort a list of n elements is less than M(n) where M(n) = 2M(n/2) + n, by the Master theorem M(n) is O(n log n) since

a = 2, b = 2, and d = 1 and a = b^d ($2 = 2^1$)

36

Complexity of Fast Integer Multiplication Algorithm

Integer Multiplication Example: *Give a big-O estimate for the number of bit operations used needed to multiply two n-bit integers using the fast multiplication algorithm.*

Solution: We have shown that $f(n) = 3f(n/2) + Cn$, when n is even, where $f(n)$ is the number of bit operations needed to multiply two n-bit integers. Hence by the master theorem with $a = 3$, $b = 2$, $c = C$, and $d = 1$ (so that we have the case where $a > b^d$), it follows that $f(n)$ is $O(n^{\log 3})$.

Note that $\log 3 \approx 1.6$. Therefore, the fast multiplication algorithm is a substantial improvement over the conventional algorithm that uses $O(n^2)$ bit operations.

37

Inclusion-Exclusion

Section 8.5

38

Section Summary

The Principle of Inclusion-Exclusion

Examples

39

Principle of Inclusion-Exclusion

In Section 2.2, we developed the following formula for the number of elements in the union of two finite sets:

$$|A \cup B| = |A| + |B| - |A \cap B|$$

We will generalize this formula to finite sets of any size.

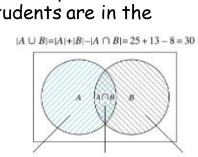
40

Two Finite Sets

Example: In a discrete mathematics class every student is a major in computer science or mathematics or both. The number of students having computer science as a major (possibly along with mathematics) is 25; the number of students having mathematics as a major (possibly along with computer science) is 13; and the number of students majoring in both computer science and mathematics is 8. How many students are in the class?

Solution: $|A \cup B| = |A| + |B| - |A \cap B|$

$= 25 + 13 - 8 = 30$

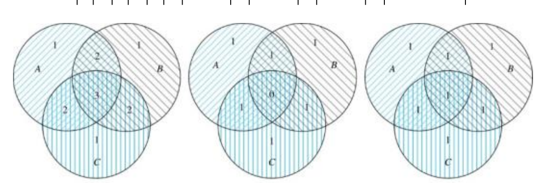


Jump to long description

41

Three Finite Sets

$$|A \cup B \cup C| =$$
$$|A| + |B| + |C| - |A \cap B| - |A \cap C| - |B \cap C| + |A \cap B \cap C|$$



(a) Count of elements by $|A|+|B|+|C|$

(b) Count of elements by $|A|+|B|+|C|-|A \cap B|-|A \cap C|-|B \cap C|$

(c) Count of elements by $|A|+|B|+|C|-|A \cap B|-|A \cap C|-|B \cap C|+|A \cap B \cap C|$

Jump to long description

42

Three Finite Sets

Example: A total of 1232 students have taken a course in Spanish, 879 have taken a course in French, and 114 have taken a course in Russian. Further, 103 have taken courses in both Spanish and French, 23 have taken courses in both Spanish and Russian, and 14 have taken courses in both French and Russian. If 2092 students have taken a course in at least one of Spanish French and Russian, how many students have taken a course in all 3 languages.

Solution: Let S be the set of students who have taken a course in Spanish, F the set of students who have taken a course in French, and R the set of students who have taken a course in Russian. Then,

$|S| = 1232$, $|F| = 879$, $|R| = 114$, $|S \cap F| = 103$, $|S \cap R| = 23$, $|F \cap R| = 14$, and $|S \cup F \cup R| = 2092$.

Using the equation

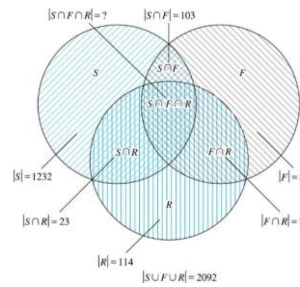
$$|S \cup F \cup R| = |S| + |F| + |R| - |S \cap F| - |S \cap R| - |F \cap R| + |S \cap F \cap R|,$$

we obtain $2092 = 1232 + 879 + 114 - 103 - 23 - 14 + |S \cap F \cap R|$.

Solving for $|S \cap F \cap R|$ yields 7.

43

Illustration of Three Finite Set Example



44

The Principle of Inclusion-Exclusion

Theorem 1. The Principle of Inclusion-Exclusion: Let $A_1, A_2, \ldots, A_n$ be finite sets. Then:

$$|A_1 \cup A_2 \cup \cdots \cup A_n| = \sum_{1 \le i \le j \le k \le n} |A_i| - \sum_{1 \le i \le j \le n} |A_i \cap A_j| + \sum_{1 \le i \le j \le k \le n} |A_i \cap A_j \cap A_k| - \ldots + (-1)^{n+1} |A_1 \cap A_2 \cap \cdots \cap A_n|$$

45

The Principle of Inclusion-Exclusion

Proof: An element in the union is counted exactly once in the right-hand side of the equation. Consider an element a that is a member of r of the sets $A_1, \ldots, A_n$ where $1 \le r \le n$.

- It is counted $C(r,1)$ times by $\sum |A_i|$
- It is counted $C(r,2)$ times by $\sum |A_i \cap A_j|$
- In general, it is counted $C(r,m)$ times by the summation of m of the sets A_i.

46

The Principle of Inclusion-Exclusion

Thus the element is counted exactly

$$C(r,1) - C(r,2) + C(r,3) - \cdots + (-1)^{r+1} C(r,r) = 0.$$

times by the right-hand side of the equation.

By Corollary 2 of Section 6.4, we have

$$C(r,0) - C(r,1) + C(r,2) - \cdots + (-1)^r C(r,r) = 0.$$

Hence,

$$1 = C(r,0) = C(r,1) - C(r,2) + \cdots + (-1)^{r+1} C(r,r).$$

47

Appendix of Image Long Descriptions

48

Rabbits and the Fiobonacci Numbers – Appendix

There is 0 reproducing pair and 1 young pair in the first month, the number of total pairs is 1. There is 0 reproducing pair and 1 young pair in the second month, the number of total pairs is 1. There is 1 reproducing pair and 1 young pair in the third month, the number of total pairs is 2. There is 1 reproducing pair and 2 young pairs in the fourth month, the number of total pairs is 3. There are 2 reproducing pairs and 3 young pairs in the fifth month, the number of total pairs is 5. There are 3 reproducing pairs and 5 young pairs in the sixth month, the number of total pairs is 8.

Jump to the image

49

Counting Bit Strings – Appendix

The first bit string starts with any bit string of length N minus 1 with no two consecutive zeros and ends with a 1. The number of strings of this type is A sub, N minus 1. The second bit strings starts with any bit string of length N minus 2 with no two consecutive zeros and ends with 1 0. The number of strings of this type is A sub, N minus 2. The total number of bit strings of length N with no two consecutive zeros is A sub, N minus 1, plus A sub, N minus 2.

Jump to the image

50

Two Finite Sets – Appendix

The number of elements in A is 25, the number of elements in B is 13. The number of elements in the intersection of A and B is 8.

Jump to the image

51

Three Finite Sets – Appendix

The first diagram shows count of elements by the formula number of elements in A plus the number of elements in B plus the number of elements in C. There is number 1 in exactly one of the three sets, number 2 in the intersection of any two of the sets, and number 3 in the intersection of all three sets. The second diagram shows count of elements by the formula number of elements in A plus the number of elements in B plus the number of elements in C minus the number of elements in intersection of A and B minus the number of elements in intersection of A and C minus the number of elements in intersection of B and C. There is number 1 in exactly one of the three sets and in the intersection of any two of the sets, and number 0 in the intersection of all three sets. The third diagram shows count of elements by the formula number of elements in A plus the number of elements in B plus the number of elements in C minus the number of elements in intersection of A and B minus the number of elements in intersection of A and C minus the number of elements in intersection of B and C plus the number of elements in intersection of A, B, and C. There is number 1 in exactly one of the three sets and in all intersections.

Jump to the image

52