

only once. If we did not do this, the algorithm would have exponential worst-case complexity. The process of storing the values as each is computed is known as **memoization** and is an important technique for making recursive algorithms efficient.

ALGORITHM 1 Dynamic Programming Algorithm for Scheduling Talks.

```

procedure Maximum Attendees ( $s_1, s_2, \dots, s_n$ : start times of talks;
 $e_1, e_2, \dots, e_n$ : end times of talks;  $w_1, w_2, \dots, w_n$ : number of attendees to talks)
  sort talks by end time and relabel so that  $e_1 \leq e_2 \leq \dots \leq e_n$ 
for  $j := 1$  to  $n$ 
  if no job  $i$  with  $i < j$  is compatible with job  $j$ 
     $p(j) = 0$ 
  else  $p(j) := \max\{i \mid i < j \text{ and job } i \text{ is compatible with job } j\}$ 
   $T(0) := 0$ 
for  $j := 1$  to  $n$ 
   $T(j) := \max(w_j + T(p(j)), T(j - 1))$ 
return  $T(n)$  { $T(n)$  is the maximum number of attendees}

```

In Algorithm 1 we determine the maximum number of attendees that can be achieved by a schedule of talks, but we do not find a schedule that achieves this maximum. To find talks we need to schedule, we use the fact that talk j belongs to an optimal solution for the first j talks if and only if $w_j + T(p(j)) \geq T(j - 1)$. We leave it as Exercise 53 to construct an algorithm based on this observation that determines which talks should be scheduled to achieve the maximum total number of attendees.

Algorithm 1 is a good example of dynamic programming as the maximum total attendance is found using the optimal solutions of the overlapping subproblems, each of which determines the maximum total attendance of the first j talks for some j with $1 \leq j \leq n - 1$. See Exercises 56 and 57 and Supplementary Exercises 14 and 17 for other examples of dynamic programming.

Exercises

1. Use mathematical induction to verify the formula derived in Example 2 for the number of moves required to complete the Tower of Hanoi puzzle.
2. a) Find a recurrence relation for the number of permutations of a set with n elements.
b) Use this recurrence relation to find the number of permutations of a set with n elements using iteration.
3. A vending machine dispensing books of stamps accepts only one-dollar coins, \$1 bills, and \$5 bills.
a) Find a recurrence relation for the number of ways to deposit n dollars in the vending machine, where the order in which the coins and bills are deposited matters.
b) What are the initial conditions?
c) How many ways are there to deposit \$10 for a book of stamps?
4. A country uses as currency coins with values of 1 peso, 2 pesos, 5 pesos, and 10 pesos and bills with values of 5 pesos, 10 pesos, 20 pesos, 50 pesos, and 100 pesos. Find a recurrence relation for the number of ways to pay a bill of n pesos if the order in which the coins and bills are paid matters.
5. How many ways are there to pay a bill of 17 pesos using the currency described in Exercise 4, where the order in which coins and bills are paid matters?
- *6. a) Find a recurrence relation for the number of strictly increasing sequences of positive integers that have 1 as their first term and n as their last term, where n is a positive integer. That is, sequences a_1, a_2, \dots, a_k , where $a_1 = 1$, $a_k = n$, and $a_j < a_{j+1}$ for $j = 1, 2, \dots, k - 1$.
b) What are the initial conditions?
c) How many sequences of the type described in (a) are there when n is an integer with $n \geq 2$?
7. a) Find a recurrence relation for the number of bit strings of length n that contain a pair of consecutive 0s.

- b) What are the initial conditions?
 c) How many bit strings of length seven contain two consecutive 0s?
8. a) Find a recurrence relation for the number of bit strings of length n that contain three consecutive 0s.
 b) What are the initial conditions?
 c) How many bit strings of length seven contain three consecutive 0s?
9. a) Find a recurrence relation for the number of bit strings of length n that do not contain three consecutive 0s.
 b) What are the initial conditions?
 c) How many bit strings of length seven do not contain three consecutive 0s?
- *10. a) Find a recurrence relation for the number of bit strings of length n that contain the string 01.
 b) What are the initial conditions?
 c) How many bit strings of length seven contain the string 01?
11. a) Find a recurrence relation for the number of ways to climb n stairs if the person climbing the stairs can take one stair or two stairs at a time.
 b) What are the initial conditions?
 c) In how many ways can this person climb a flight of eight stairs?
12. a) Find a recurrence relation for the number of ways to climb n stairs if the person climbing the stairs can take one, two, or three stairs at a time.
 b) What are the initial conditions?
 c) In many ways can this person climb a flight of eight stairs?
- A string that contains only 0s, 1s, and 2s is called a **ternary string**.
13. a) Find a recurrence relation for the number of ternary strings of length n that do not contain two consecutive 0s.
 b) What are the initial conditions?
 c) How many ternary strings of length six do not contain two consecutive 0s?
14. a) Find a recurrence relation for the number of ternary strings of length n that contain two consecutive 0s.
 b) What are the initial conditions?
 c) How many ternary strings of length six contain two consecutive 0s?
- *15. a) Find a recurrence relation for the number of ternary strings of length n that do not contain two consecutive 0s or two consecutive 1s.
 b) What are the initial conditions?
 c) How many ternary strings of length six do not contain two consecutive 0s or two consecutive 1s?
- *16. a) Find a recurrence relation for the number of ternary strings of length n that contain either two consecutive 0s or two consecutive 1s.
 b) What are the initial conditions?
 c) How many ternary strings of length six contain two consecutive 0s or two consecutive 1s?
- *17. a) Find a recurrence relation for the number of ternary strings of length n that do not contain consecutive symbols that are the same.
 b) What are the initial conditions?
 c) How many ternary strings of length six do not contain consecutive symbols that are the same?
- **18. a) Find a recurrence relation for the number of ternary strings of length n that contain two consecutive symbols that are the same.
 b) What are the initial conditions?
 c) How many ternary strings of length six contain consecutive symbols that are the same?
19. Messages are transmitted over a communications channel using two signals. The transmittal of one signal requires 1 microsecond, and the transmittal of the other signal requires 2 microseconds.
 a) Find a recurrence relation for the number of different messages consisting of sequences of these two signals, where each signal in the message is immediately followed by the next signal, that can be sent in n microseconds.
 b) What are the initial conditions?
 c) How many different messages can be sent in 10 microseconds using these two signals?
20. A bus driver pays all tolls, using only nickels and dimes, by throwing one coin at a time into the mechanical toll collector.
 a) Find a recurrence relation for the number of different ways the bus driver can pay a toll of n cents (where the order in which the coins are used matters).
 b) In how many different ways can the driver pay a toll of 45 cents?
21. a) Find the recurrence relation satisfied by R_n , where R_n is the number of regions that a plane is divided into by n lines, if no two of the lines are parallel and no three of the lines go through the same point.
 b) Find R_n using iteration.
- *22. a) Find the recurrence relation satisfied by R_n , where R_n is the number of regions into which the surface of a sphere is divided by n great circles (which are the intersections of the sphere and planes passing through the center of the sphere), if no three of the great circles go through the same point.
 b) Find R_n using iteration.
- *23. a) Find the recurrence relation satisfied by S_n , where S_n is the number of regions into which three-dimensional space is divided by n planes if every three of the planes meet in one point, but no four of the planes go through the same point.
 b) Find S_n using iteration.
24. Find a recurrence relation for the number of bit sequences of length n with an even number of 0s.
25. How many bit sequences of length seven contain an even number of 0s?

26. a) Find a recurrence relation for the number of ways to completely cover a $2 \times n$ checkerboard with 1×2 dominoes. [Hint: Consider separately the coverings where the position in the top right corner of the checkerboard is covered by a domino positioned horizontally and where it is covered by a domino positioned vertically.]
 b) What are the initial conditions for the recurrence relation in part (a)?
 c) How many ways are there to completely cover a 2×17 checkerboard with 1×2 dominoes?
27. a) Find a recurrence relation for the number of ways to lay out a walkway with slate tiles if the tiles are red, green, or gray, so that no two red tiles are adjacent and tiles of the same color are considered indistinguishable.
 b) What are the initial conditions for the recurrence relation in part (a)?
 c) How many ways are there to lay out a path of seven tiles as described in part (a)?
28. Show that the Fibonacci numbers satisfy the recurrence relation $f_n = 5f_{n-4} + 3f_{n-5}$ for $n = 5, 6, 7, \dots$, together with the initial conditions $f_0 = 0, f_1 = 1, f_2 = 1, f_3 = 2$, and $f_4 = 3$. Use this recurrence relation to show that f_{5n} is divisible by 5, for $n = 1, 2, 3, \dots$.
- *29. Let $S(m, n)$ denote the number of onto functions from a set with m elements to a set with n elements. Show that $S(m, n)$ satisfies the recurrence relation

$$S(m, n) = n^m - \sum_{k=1}^{n-1} C(n, k)S(m, k)$$

whenever $m \geq n$ and $n > 1$, with the initial condition $S(m, 1) = 1$.

30. a) Write out all the ways the product $x_0 \cdot x_1 \cdot x_2 \cdot x_3 \cdot x_4$ can be parenthesized to determine the order of multiplication.
 b) Use the recurrence relation developed in Example 5 to calculate C_4 , the number of ways to parenthesize the product of five numbers so as to determine the order of multiplication. Verify that you listed the correct number of ways in part (a).
 c) Check your result in part (b) by finding C_4 , using the closed formula for C_n mentioned in the solution of Example 5.
31. a) Use the recurrence relation developed in Example 5 to determine C_5 , the number of ways to parenthesize the product of six numbers so as to determine the order of multiplication.
 b) Check your result with the closed formula for C_5 mentioned in the solution of Example 5.
- *32. In the Tower of Hanoi puzzle, suppose our goal is to transfer all n disks from peg 1 to peg 3, but we cannot move a disk directly between pegs 1 and 3. Each move of a disk must be a move involving peg 2. As usual, we cannot place a disk on top of a smaller disk.

- a) Find a recurrence relation for the number of moves required to solve the puzzle for n disks with this added restriction.
 b) Solve this recurrence relation to find a formula for the number of moves required to solve the puzzle for n disks.
 c) How many different arrangements are there of the n disks on three pegs so that no disk is on top of a smaller disk?
 d) Show that every allowable arrangement of the n disks occurs in the solution of this variation of the puzzle.



Exercises 33–37 deal with a variation of the **Josephus problem** described by Graham, Knuth, and Patashnik in [GrKnPa94]. This problem is based on an account by the historian Flavius Josephus, who was part of a band of 41 Jewish rebels trapped in a cave by the Romans during the Jewish-Roman war of the first century. The rebels preferred suicide to capture; they decided to form a circle and to repeatedly count off around the circle, killing every third rebel left alive. However, Josephus and another rebel did not want to be killed this way; they determined the positions where they should stand to be the last two rebels remaining alive. The variation we consider begins with n people, numbered 1 to n , standing around a circle. In each stage, every second person still left alive is eliminated until only one survives. We denote the number of the survivor by $J(n)$.

33. Determine the value of $J(n)$ for each integer n with $1 \leq n \leq 16$.
 34. Use the values you found in Exercise 33 to conjecture a formula for $J(n)$. [Hint: Write $n = 2^m + k$, where m is a nonnegative integer and k is a nonnegative integer less than 2^m .]
 35. Show that $J(n)$ satisfies the recurrence relation $J(2n) = 2J(n) - 1$ and $J(2n + 1) = 2J(n) + 1$, for $n \geq 1$, and $J(1) = 1$.
 36. Use mathematical induction to prove the formula you conjectured in Exercise 34, making use of the recurrence relation from Exercise 35.
 37. Determine $J(100)$, $J(1000)$, and $J(10,000)$ from your formula for $J(n)$.

Exercises 38–45 involve the Reve's puzzle, the variation of the Tower of Hanoi puzzle with four pegs and n disks. Before presenting these exercises, we describe the Frame–Stewart algorithm for moving the disks from peg 1 to peg 4 so that no disk is ever on top of a smaller one. This algorithm, given the number of disks n as input, depends on a choice of an integer k with $1 \leq k \leq n$. When there is only one disk, move it from peg 1 to peg 4 and stop. For $n > 1$, the algorithm proceeds recursively, using these three steps. Recursively move the stack of the $n - k$ smallest disks from peg 1 to peg 2, using all four pegs. Next move the stack of the k largest disks from peg 1 to peg 4, using the three-peg algorithm from the Tower of Hanoi puzzle without using the peg holding the $n - k$ smallest disks. Finally, recursively move the smallest $n - k$ disks to peg 4, using all four pegs. Frame and Stewart showed that to produce the fewest moves using their algorithm, k should be chosen to be the smallest integer

such that n does not exceed $t_k = k(k+1)/2$, the k th triangular number, that is, $t_{k-1} < n \leq t_k$. The unsettled conjecture, known as **Frame's conjecture**, is that this algorithm uses the fewest number of moves required to solve the puzzle, no matter how the disks are moved.

38. Show that the Reve's puzzle with three disks can be solved using five, and no fewer, moves.
39. Show that the Reve's puzzle with four disks can be solved using nine, and no fewer, moves.
40. Describe the moves made by the Frame–Stewart algorithm, with k chosen so that the fewest moves are required, for
 a) 5 disks. b) 6 disks. c) 7 disks. d) 8 disks.
- *41. Show that if $R(n)$ is the number of moves used by the Frame–Stewart algorithm to solve the Reve's puzzle with n disks, where k is chosen to be the smallest integer with $n \leq k(k+1)/2$, then $R(n)$ satisfies the recurrence relation $R(n) = 2R(n-k) + 2^k - 1$, with $R(0) = 0$ and $R(1) = 1$.

*42. Show that if k is as chosen in Exercise 41, then $R(n) - R(n-1) = 2^{k-1}$.

*43. Show that if k is as chosen in Exercise 41, then $R(n) = \sum_{i=1}^k i2^{i-1} - (t_k - n)2^{k-1}$.

*44. Use Exercise 43 to give an upper bound on the number of moves required to solve the Reve's puzzle for all integers n with $1 \leq n \leq 25$.

*45. Show that $R(n)$ is $O(\sqrt{n}2^{\sqrt{2n}})$.

Let $\{a_n\}$ be a sequence of real numbers. The **backward differences** of this sequence are defined recursively as shown next. The **first difference** ∇a_n is

$$\nabla a_n = a_n - a_{n-1}.$$

The $(k+1)$ st difference $\nabla^{k+1} a_n$ is obtained from $\nabla^k a_n$ by

$$\nabla^{k+1} a_n = \nabla^k a_n - \nabla^k a_{n-1}.$$

46. Find ∇a_n for the sequence $\{a_n\}$, where

- a) $a_n = 4$. b) $a_n = 2n$.
 c) $a_n = n^2$. d) $a_n = 2^n$.

47. Find $\nabla^2 a_n$ for the sequences in Exercise 46.

48. Show that $a_{n-1} = a_n - \nabla a_n$.

49. Show that $a_{n-2} = a_n - 2\nabla a_n + \nabla^2 a_n$.

*50. Prove that a_{n-k} can be expressed in terms of a_n , ∇a_n , $\nabla^2 a_n$, \dots , $\nabla^k a_n$.

51. Express the recurrence relation $a_n = a_{n-1} + a_{n-2}$ in terms of a_n , ∇a_n , and $\nabla^2 a_n$.

52. Show that any recurrence relation for the sequence $\{a_n\}$ can be written in terms of a_n , ∇a_n , $\nabla^2 a_n$, \dots . The resulting equation involving the sequences and its differences is called a **difference equation**.

*53. Construct the algorithm described in the text after Algorithm 1 for determining which talks should be scheduled to maximize the total number of attendees and not just the maximum total number of attendees determined by Algorithm 1.

54. Use Algorithm 1 to determine the maximum number of total attendees in the talks in Example 6 if w_i , the number of attendees of talk i , $i = 1, 2, \dots, 7$, is

- a) 20, 10, 50, 30, 15, 25, 40.
 b) 100, 5, 10, 20, 25, 40, 30.
 c) 2, 3, 8, 5, 4, 7, 10.
 d) 10, 8, 7, 25, 20, 30, 5.

55. For each part of Exercise 54, use your algorithm from Exercise 53 to find the optimal schedule for talks so that the total number of attendees is maximized.

56. In this exercise we will develop a dynamic programming algorithm for finding the maximum sum of consecutive terms of a sequence of real numbers. That is, given a sequence of real numbers a_1, a_2, \dots, a_n , the algorithm computes the maximum sum $\sum_{i=j}^k a_i$ where $1 \leq j \leq k \leq n$.

a) Show that if all terms of the sequence are nonnegative, this problem is solved by taking the sum of all terms. Then, give an example where the maximum sum of consecutive terms is not the sum of all terms.

b) Let $M(k)$ be the maximum of the sums of consecutive terms of the sequence ending at a_k . That is, $M(k) = \max_{1 \leq j \leq k} \sum_{i=j}^k a_i$. Explain why the recurrence relation $M(k) = \max(M(k-1) + a_k, a_k)$ holds for $k = 2, \dots, n$.

c) Use part (b) to develop a dynamic programming algorithm for solving this problem.

d) Show each step your algorithm from part (c) uses to find the maximum sum of consecutive terms of the sequence 2, -3, 4, 1, -2, 3.

e) Show that the worst-case complexity in terms of the number of additions and comparisons of your algorithm from part (c) is linear.

*57. Dynamic programming can be used to develop an algorithm for solving the matrix-chain multiplication problem introduced in Section 3.3. This is the problem of determining how the product $\mathbf{A}_1 \mathbf{A}_2 \cdots \mathbf{A}_n$ can be computed using the fewest integer multiplications, where $\mathbf{A}_1, \mathbf{A}_2, \dots, \mathbf{A}_n$ are $m_1 \times m_2, m_2 \times m_3, \dots, m_n \times m_{n+1}$ matrices, respectively, and each matrix has integer entries. Recall that by the associative law, the product does not depend on the order in which the matrices are multiplied.

a) Show that the brute-force method of determining the minimum number of integer multiplications needed to solve a matrix-chain multiplication problem has exponential worst-case complexity. [Hint: Do this by first showing that the order of multiplication of matrices is specified by parenthesizing the product. Then, use Example 5 and the result of part (c) of Exercise 41 in Section 8.4.]

- b) Denote by A_{ij} the product $A_i A_{i+1} \dots A_j$, and $M(i, j)$ the minimum number of integer multiplications required to find A_{ij} . Show that if the least number of integer multiplications are used to compute A_{ij} , where $i < j$, by splitting the product into the product of A_i through A_k and the product of A_{k+1} through A_j , then the first k terms must be parenthesized so that A_{ik} is computed in the optimal way using $M(i, k)$ integer multiplications and $A_{k+1,j}$ must be parenthesized so that $A_{k+1,j}$ is computed in the optimal way using $M(k+1, j)$ integer multiplications.
- c) Explain why part (b) leads to the recurrence relation $M(i, j) = \min_{i \leq k < j} (M(i, k) + M(k+1, j) + m_i m_{k+1} m_{j+1})$ if $1 \leq i \leq j < j \leq n$.
- d) Use the recurrence relation in part (c) to construct an efficient algorithm for determining the order the n matrices should be multiplied to use the minimum number of integer multiplications. Store the partial results $M(i, j)$ as you find them so that your algorithm will not have exponential complexity.
- e) Show that your algorithm from part (d) has $O(n^3)$ worst-case complexity in terms of multiplications of integers.

8.2 Solving Linear Recurrence Relations

Introduction



A wide variety of recurrence relations occur in models. Some of these recurrence relations can be solved using iteration or some other ad hoc technique. However, one important class of recurrence relations can be explicitly solved in a systematic way. These are recurrence relations that express the terms of a sequence as linear combinations of previous terms.

DEFINITION 1

A *linear homogeneous recurrence relation of degree k with constant coefficients* is a recurrence relation of the form

$$a_n = c_1 a_{n-1} + c_2 a_{n-2} + \dots + c_k a_{n-k},$$


where c_1, c_2, \dots, c_k are real numbers, and $c_k \neq 0$.

The recurrence relation in the definition is **linear** because the right-hand side is a sum of previous terms of the sequence each multiplied by a function of n . The recurrence relation is **homogeneous** because no terms occur that are not multiples of the a_j s. The coefficients of the terms of the sequence are all **constants**, rather than functions that depend on n . The **degree** is k because a_n is expressed in terms of the previous k terms of the sequence.

A consequence of the second principle of mathematical induction is that a sequence satisfying the recurrence relation in the definition is uniquely determined by this recurrence relation and the k initial conditions

$$a_0 = C_0, a_1 = C_1, \dots, a_{k-1} = C_{k-1}.$$

EXAMPLE 1

The recurrence relation $P_n = (1.11)P_{n-1}$ is a linear homogeneous recurrence relation of degree one. The recurrence relation $f_n = f_{n-1} + f_{n-2}$ is a linear homogeneous recurrence relation of degree two. The recurrence relation $a_n = a_{n-5}$ is a linear homogeneous recurrence relation of degree five. 

Example 2 presents some examples of recurrence relations that are not linear homogeneous recurrence relations with constant coefficients.

EXAMPLE 2

The recurrence relation $a_n = a_{n-1} + a_{n-2}^2$ is not linear. The recurrence relation $H_n = 2H_{n-1} + 1$ is not homogeneous. The recurrence relation $B_n = nB_{n-1}$ does not have constant coefficients. 