Induction, Recursion, and Recurrence Relations

Dit-Yan Yeung

Department of Computer Science and Engineering
Hong Kong University of Science and Technology

COMP 2711: Discrete Mathematical Tools for Computer Science

Contents

Principle of Mathematical Induction

Understanding how to read and construct proofs by mathematical induction is a key goal of learning discrete mathematics.

Principle (Principle of mathematical induction)

To prove that $P(n)$ is true for all integers $n \geq b$, where $P(n)$ is a propositional function and b is an integer, we complete two steps:

Basis step:	Verify that $P(b)$ is true.
Inductive step:	Show that $P(k) \rightarrow P(k+1)$ is true for all integers $k \geq b$.

Remark

Remark

The principle of mathematical induction is usually described in two forms. The one above, called the "weak form", applies to statements about integers n. The principle is also known as the **weak principle of mathematical induction**.

Remarks

Remark

Expressed as a rule of inference, the proof technique based on the principle of mathematical induction can be stated as

$$\big[P(b) \wedge \forall k\,(P(k) \to P(k+1))\big] \to \forall n\,P(n),$$

where the domain is the set of integers greater than or equal to b.

Remark

To show that $P(k) \to P(k+1)$ is true for every integer $k \geq b$, we take the **inductive hypothesis** which assumes that $P(k)$ is true. We then proceed to show that, under this hypothesis, $P(k+1)$ must also be true.

Proving Summation Formulae

Remark

Mathematical induction is not a tool for finding theorems about all positive integers. Rather, it is a proof method for proving such results once they have been conjectured.

Example 1

Conjecture a formula for the sum of the first n positive odd integers. Then use mathematical induction to prove the conjecture.

Example

Example 2

Use mathematical induction to show that

$$\sum_{j=0}^{n} 2^j = 2^{n+1} - 1$$

for all nonnegative integers n.

Example

Example 3

Use mathematical induction to prove the following formula for the sum of a finite number of terms of a geometric progression:

$$\sum_{j=0}^{n} ar^j = \frac{a(r^{n+1} - 1)}{r - 1} \quad \text{when } r \neq 1,$$

where n is a nonnegative integer.

Proving Inequalities

Example 4

Use mathematical induction to prove the inequality $n < 2^n$ for all positive integers n.

Example 5

Find the smallest integer b for which $2^n < n!$ true for all integers $n \geq b$. Use mathematical induction to show that your answer is correct.

Example

Example 6

The **harmonic numbers** H_j, $j = 1, 2, 3, \ldots$, are defined by

$$H_j = 1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{j}.$$

Use mathematical induction to show that

$$H_{2^n} \geq 1 + \frac{n}{2},$$

where n is a nonnegative integer.

Remark

Remark

The inequality established above shows that the **harmonic series**

$$1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{n} + \cdots$$

is a divergent infinite series.

Proving Divisibility Results

Example 7

Use mathematical induction to prove that $n^3 - n$ is divisible by 3 for all positive integers n.

Proving Results about Sets

Example 8

Use mathematical induction to show that if S is a finite set with n elements where n is a nonnegative integer, then S has 2^n subsets.

Example 9

Use mathematical induction to prove the following generalization of one of De Morgan's laws:

$$\overline{\bigcap_{j=1}^{n} A_j} = \bigcup_{j=1}^{n} \overline{A_j},$$

where $A_1, A_2, \ldots, A_n$ are subsets of a universal set U and $n \geq 2$.

Some Subtle Examples

Example 10

An odd number of people stand in a yard at mutually distinct distances. At the same time each person throws a pie at their nearest neighbor, hitting this person. Use mathematical induction to show that there is at least one survivor, i.e., at least one person who is not hit by a pie.

Some Subtle Examples (cont'd)

Example 11

Find the error in this "proof" of the clearly false claim that every set of lines in the plane, no two of which are parallel, meet in a common point.

"Proof": Let $P(n)$ be the proposition that every set of n lines in the plane, no two of which are parallel, meet in a common point.

- Basis step: $P(2)$ is true, because any two lines in the plane that are not parallel meet in a common point.
- Inductive step: For the inductive hypothesis, we assume that $P(k)$ is true, i.e., every set of k lines in the plane, no two of which are parallel, meet in a common point. To show that $P(k+1)$ is true, consider a set of $k+1$ distinct lines in the plane. By the inductive hypothesis, the first k of these lines meet in a common point p_1. Moreover, by the inductive hypothesis, the last k of these lines meet in a common point p_2. We will show that p_1 and p_2 must be the same point. If p_1 and p_2 were different points, all lines containing both of them must be the same line because two points determine a line. This contradicts our assumption that all these lines are distinct. Thus, p_1 and p_2 are the same point and hence the point $p_1 = p_2$ lies on all $k+1$ lines. This shows that $P(k+1)$ is also true.

Consequently, by the principle of mathematical induction, we can conclude that $P(n)$ is true for all integers $n \geq 2$.

Strong Principle of Mathematical Induction

Sometimes it is not easy to prove a result using the weak principle of mathematical induction. The second form of the principle of mathematical principle may then be applicable.

Principle (Strong principle of mathematical induction)

To prove that $P(n)$ is true for all integers $n \geq b$, where $P(n)$ is a propositional function and b is an integer, we complete two steps:

Basis step:	Verify that $P(b)$ is true.
Inductive step:	Show that $\big[P(b) \wedge P(b+1) \wedge \cdots \wedge P(k)\big] \rightarrow P(k+1)$ is true for all integers $k \geq b$.

Remarks

Remark

The terms *weak* and *strong* arise from what is assumed in the inductive hypothesis. Adding more restrictions *strengthens* an assertion, while removing restrictions *weakens* the assertion.

Remark

The strong principle of mathematical induction is also known as **strong induction**. In a proof by strong induction, the inductive hypothesis is the assumption that $P(j)$ is true for $j = b, b+1, \ldots, k$. Because we can use all statements $P(b), P(b+1), \ldots, P(k)$ to prove $P(k+1)$, rather than just the statement $P(k)$ as in a proof by mathematical induction, strong induction is a more flexible proof technique.

Remark

Remark

Although strong induction is more flexible, mathematical induction and strong induction are in fact equivalent. That is, each can be shown to be a valid proof technique assuming that the other is valid. Moreover, a proof based on one principle can be converted into one by the other principle.

Examples

Example 12

Show that if n is an integer greater than 1, then n can be written as the product of primes.

Remark

This example is an important result from the Fundamental Theorem of Arithmetic.

Example 13

Consider a game in which two players take turns removing any positive number of matches they want from one of two piles of matches. The player who removes the last match wins the game. Show that if the two piles contain the same number of matches initially, the second player can always guarantee a win.

Alternative Form of Strong Principle

Principle (Alternative form of strong principle of mathematical induction)

To prove that $P(n)$ is true for all integers $n \geq b$, where $P(n)$ is a propositional function and b is an integer, we complete two steps:

Basis step:	Verify that $P(b), P(b+1), \ldots, P(b+j)$ are true for some fixed positive integer j.
Inductive step:	Show that $\big[P(b) \wedge P(b+1) \wedge \cdots \wedge P(k)\big] \rightarrow P(k+1)$ is true for all integers $k \geq b+j$.

Remark

The proof that this form is equivalent to strong induction is skipped. Note that multiple base cases are needed here.

Example

Example 14

Prove that every amount of postage of 12 cents or more can be formed using just 4-cent and 5-cent stamps.

Remark

This example can also be proved using the principle of mathematical induction.

Recursively Defined Functions

Definition

A **recursive definition** (a.k.a. **inductive definition**) of a function f with the set of nonnegative integers as its domain involves two steps:

Basis step:	Specify $f(0)$.
Recursive step:	Give a rule for finding $f(n+1)$, for $n \geq 0$, based on $f(k)$ for all k with $0 \leq k \leq n$.

Examples

Example 15

Suppose a function f is defined recursively as follows:

$$f(0) = 3$$
$$f(n+1) = 2f(n) + 3, \quad n \geq 0.$$

Find $f(1)$, $f(2)$, $f(3)$, and $f(4)$.

Example 16

Give a recursive definition of the factorial function $F(n) = n!$.

Examples

Example 17

Using the recursive definition of the factorial function $F(n)$ above, find $F(5)$.

Example 18

Give a recursive definition of a^n, where a is a nonzero real number and n is a nonnegative integer.

Example 19

Give a recursive definition of

$$\sum_{k=0}^{n} a_k.$$

Examples

Example 20

The **Fibonacci numbers** $f_0, f_1, f_2, \ldots$ can be defined recursively as follows:

$$f_0 = 0$$
$$f_1 = 1$$
$$f_n = f_{n-1} + f_{n-2}, \quad n \geq 2.$$

Find the Fibonacci numbers f_2, f_3, f_4, f_5, and f_6.

Example 21

Let the Fibonacci numbers be denoted by f_n. Show that whenever $n \geq 3$, $f_n > \alpha^{n-2}$, where $\alpha = (1 + \sqrt{5})/2$.

Recursively Defined Sets and Structures

Example 22

Consider the subset S of the set of integers defined by

Basis step:	$3 \in S$
Recursive step:	If $x \in S$ and $y \in S$, then $x + y \in S$.

We will prove later that S is the set of all positive multiples of 3.

Example

Example 23

A **string** defined over an alphabet Σ is a finite sequence of symbols from Σ. The set of strings over Σ, denoted by Σ^*, can be defined recursively as follows:

Basis step:	$\lambda \in \Sigma^*$, where λ denotes the empty string containing no symbols
Recursive step:	If $w \in \Sigma^*$ and $x \in \Sigma$, then $wx \in \Sigma^*$.

Examples

Example 24

Two strings can be combined via the operation of **concatenation**. The concatenation of two strings, denoted by $\cdot$, can be defined recursively as follows:

Basis step:	If $w \in \Sigma^*$, then $w \cdot \lambda = w$.
Recursive step:	If $w_1, w_2 \in \Sigma^*$ and $x \in \Sigma$, then $w_1 \cdot (w_2 x) = (w_1 \cdot w_2)x$.

Example 25

The length of a string w, denoted by $l(w)$, can be defined recursively as follows:

$$l(\lambda) = 0$$
$$l(wx) = l(w) + 1, \quad \text{if } w \in \Sigma^* \text{ and } x \in \Sigma.$$

Example

Example 26

The set of binary trees can be defined recursively as follows:

- Basis step:
  The empty set is a binary tree.
- Recursive step:
  If T_1 and T_2 are disjoint binary trees, there is a binary tree, denoted by $T_1 \cdot T_2$, that consists of a root r together with edges connecting the root to each of the roots of the left subtree T_1 and the right subtree T_2 if these trees are nonempty.

Structural Induction

To prove results about recursively defined sets and structures we generally use some form of mathematical induction.

Example 27

Prove that the set S defined in Example 22 is the set of all positive integers divisible by 3.

Remark

The proof above uses mathematical induction over the set of positive integers and a recursive definition to prove a result about a recursively defined set. Alternatively, we can use a more convenient form of induction called **structural induction**.

Structural Induction (cont'd)

Principle (Structural induction)

A proof by structural induction consists of two parts:

Basis step:	Show that the result holds for all elements specified in the basis step of the recursive definition to be in the set.
Inductive step:	Show that if the statement is true for each of the elements used to construct new elements in the recursive step of the definition, the result holds for these new elements.

Example

Example 28

Use structural induction to prove that $l(xy) = l(x) + l(y)$, where x and y are strings in Σ^*, the set of strings over an alphabet Σ.

Recurrence Relations

In a recursive definition, the recursive step specifies a rule for determining subsequent terms from those that precede them. Such a rule is an example of a **recurrence relation**.

Recurrence relations can also be used in studying and solving counting problems.

Definition

A **recurrence relation** (or simply **recurrence**) 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.

Examples

Example 29

Let $\{a_n\}$ be a sequence that satisfies the recurrence relation $a_n = a_{n-1} - a_{n-2}$ for $n = 2, 3, 4, \ldots$, and suppose that $a_0 = 3$ and $a_1 = 5$. What are a_2 and a_3?

Example 30

Determine whether the sequence $\{a_n\}$, where $a_n = 3n$ for every nonnegative integer n, is a solution of the recurrence relation $a_n = 2a_{n-1} - a_{n-2}$ for $n = 2, 3, 4, \ldots$. Answer the same question where $a_n = 2^n$ and where $a_n = 5$.

Remark

Remark

Multiple solutions may exist for a given recurrence relation. However, when the **initial conditions** for a sequence are given, i.e., the terms that precede the first term where the recurrence relation takes effect are specified, the solution is unique. That is, the recurrence relation and initial conditions uniquely determine a sequence.

Modeling with Recurrence Relations

Example 31

When paying off a loan with initial amount A and monthly payment M at an interest rate of p percent, the total amount T_n of the loan after n months is computed by adding $p/12$ percent to the amount due after $n-1$ months and then subtracting the monthly payment M. Convert this description into a recurrence relation for T_n.

Example 32

Suppose that a person deposits \$10,000 in a savings account at a bank yielding 11% per year with interest compounded annually. How much will be in the account after 30 years?

Example

Example 33

A young pair of rabbits (one of each sex) 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 no rabbits ever die.

Remark

This problem was originally posed by Leonardo Pisano, also known as Fibonacci, in the thirteenth century in his book *Liber abaci*. The sequence is often known as the **Fibonacci sequence**.

Example

Example 34

The Tower of Hanoi puzzle consists of three pegs mounted on a board together with disks of different sizes. Initially these disks are placed on the first peg in order of size, with the largest on the bottom. The rules of the puzzle allow disks to be moved one at a time from one peg to another as long as a disk is never placed on top of a smaller disk. The goal of the puzzle is to have all the disks on the second peg in order of size, with the largest on the bottom. Let H_n denote the number of moves needed to solve the problem with n disks. Set up a recurrence relation for the sequence $\{H_n\}$.

Remark

This popular puzzle was invented by the French mathematician Édouard Lucas in the late nineteenth century.

Example

Example 35

Find a recurrence relation and give initial conditions for the number of bit strings of length n that do not have two consecutive 0s. How many such bit strings are there of length five?

Remark

The sequence $\{a_n\}$ satisfies the same recurrence relation as the Fibonacci sequence. Because $a_1 = f_3$ and $a_2 = f_4$, it follows that $a_n = f_{n+2}$.

Example

Example 36

A computer system considers a string of decimal digits a valid codeword if it contains an even number of 0 digits. For instance, 1230407869 is valid, whereas 120987045608 is not valid. Let a_n be the number of valid n-digit codewords. Find a recurrence relation for a_n.

Linear Homogeneous Recurrence Relation

Definition

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

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

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

Remark

The recurrence relation is **linear** because the right-hand side is a sum of the 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_js. 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.

Examples

Example 37

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 38

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.

Characteristic Equation

Definition

To solve linear homogeneous recurrence relations, we look for solutions of the form $a_n = r^n$, where r is a constant. $a_n = r^n$ is a solution of 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 $r^n = c_1 r^{n-1} + c_2 r^{n-2} + \cdots + c_k r^{n-k}$. Dividing both sides of the equation by r^{n-k} and re-arranging, we get the **characteristic equation** of the recurrence relation

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

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

Theorem

Theorem 4.1

Let c_1 and c_2 be real numbers with $c_2 \neq 0$. 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 of 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 $n = 0, 1, 2, \ldots$, where α_1 and α_2 are constants.

Proof.

There are two steps in the proof. First, we must show that if $a_n = \alpha_1 r_1^n + \alpha_2 r_2^n$ where r_1 and r_2 are the roots of the characteristic equation and α_1 and α_2 are constants, then the sequence $\{a_n\}$ is a solution of the recurrence relation. Second, we must show that if the sequence $\{a_n\}$ is a solution of the recurrence relation, then $a_n = \alpha_1 r_1^n + \alpha_2 r_2^n$. (The rest of the proof is omitted here.) □

Examples

Remark

This theorem does not hold when there are two equal characteristic roots. We need another theorem which will not be covered in this course.

Example 39

What is the solution of the recurrence relation $a_n = a_{n-1} + 2a_{n-2}$ with $a_0 = 2$ and $a_1 = 7$?

Example 40

Find an explicit formula for the Fibonacci numbers.

More Special Cases

Example 41

Find a general formula for the solution to the recurrence relation $T_n = rT_{n-1} + c$, with $T_0 = a$, where r, a, and c are constants.

Example 42

Find a general formula for the solution to the recurrence relation $T_n = rT_{n-1} + F(n)$, with $T_0 = a$, where r and a are constants and $F(n)$ is a function of n.

Example 43

Solve the recurrence relation $T_n = 4T_{n-1} + 2^n$, with $T_0 = 6$.

Example 44

Solve the recurrence relation $T_n = 3T_{n-1} + n$, with $T_0 = 10$.

Remark

Remark

We now prove the result

$$\sum_{j=1}^{n} ja^j = \frac{na^{n+2} - (n+1)a^{n+1} + a}{(1-a)^2},$$

for any real number $a \neq 1$.
Let us consider a function $f(a)$ defined as follows:

$$f(a) = \sum_{j=0}^{n} a^j = \frac{1 - a^{n+1}}{1 - a}.$$

Remark (cont'd)

Remark (cont'd)

We differentiate $f(a)$ with respect to a to get

$$\frac{df}{da} = \sum_{j=0}^{n} ja^{j-1} = \frac{-(1-a)(n+1)a^n + (1-a^{n+1})}{(1-a)^2}$$
$$= \sum_{j=1}^{n} ja^{j-1} = \frac{na^{n+1} - (n+1)a^n + 1}{(1-a)^2}.$$

Multiplying both sides by a, we get

$$\sum_{j=1}^{n} ja^j = \frac{na^{n+2} - (n+1)a^{n+1} + a}{(1-a)^2}.$$

Divide-and-Conquer Algorithm

Definition

A **divide-and-conquer algorithm** first *divides* a problem into one or more instances of the same problem of smaller size. Then it *conquers* the problem by using the solutions of the smaller problems to find a solution of the original problem, possibly with some additional work.

Some examples of divide-and-conquer algorithms:

- Binary search
- Mergesort
- Fast multiplication of integers or matrices

Big-O Notation

Big-O notation is used extensively to estimate the number of operations that an algorithm uses as its input size grows.

Definition

Let f and g be functions from the set of integers or the set of real numbers to the set of real numbers. We say that $f(x)$ is $O(g(x))$, read as "$f(x)$ is big-oh of $g(x)$", if there exist positive constants C and k such that

$$|f(x)| \leq C|g(x)|$$

whenever $x > k$.

Remark

In subsequent discussions, we will almost always deal with functions that take on only positive values. All references to absolute values can be dropped when working with big-O estimates for such functions.

Examples

Example 45

Show that $f(x) = x^2 + 2x + 1$ is $O(x^2)$.

Example 46

Show that $f(x) = x^2 + 2x + 1$ is $O(x^3)$.

Remark

In fact, we can replace x^2 by any function with larger values than x^2. When big-O notation is used in practice, the function g in the relationship $f(x)$ is $O(g(x))$ is chosen to be as small as possible.

Example

Example 47

Let $f(x) = x^2 + 2x + 1$ and $g(x) = x^2$. We have shown above that $f(x)$ is $O(g(x))$. Show that $g(x)$ is $O(f(x))$.

Remark

If two functions $f(x)$ and $g(x)$ are such that $f(x)$ is $O(g(x))$ and $g(x)$ is $O(f(x))$, we say that $f(x)$ and $g(x)$ are of the **same order**.

Big-Ω Notation

Definition

Let f and g be functions from the set of integers or the set of real numbers to the set of real numbers. We say that $f(x)$ is $\Omega(g(x))$, read as "$f(x)$ is big-Omega of $g(x)$", if there exist positive constants C and k such that

$$|f(x)| \geq C|g(x)|$$

whenever $x > k$.

Remark

While big-O notation provides an **upper bound** for the size of $f(x)$ for large values of x, big-Ω notation provides a **lower bound** for the size of $f(x)$ for large x.

Big-Θ Notation

Definition

Let f and g be functions from the set of integers or the set of real numbers to the set of real numbers. We say that $f(x)$ is $\Theta(g(x))$, read as "$f(x)$ is big-Theta of $g(x)$" or "$f(x)$ is of order $g(x)$", if $f(x)$ is $O(g(x))$ and $f(x)$ is $\Omega(g(x))$.

Remark

When $f(x)$ is $\Theta(g(x))$, it is also the case that $g(x)$ is $\Theta(f(x))$. Also note that $f(x)$ is $\Theta(g(x))$ if and only if $f(x)$ is $O(g(x))$ and $g(x)$ is $O(f(x))$.

Divide-and-Conquer Recurrence Relation

Definition

Suppose that a recursive algorithm divides a problem of size n into a subproblems, where each subproblem is of size n/b. Also, suppose that a total of $g(n)$ extra operations are required in the conquer step of the algorithm to combine the solutions of the subproblems into a solution of the original problem. Then, if $T(n)$ represents the number of operations required to solve the problem of size n, it follows that $T(n)$ satisfies the recurrence relation

$$T(n) = a\,T(n/b) + g(n),$$

which is called a **divide-and-conquer recurrence relation**.

Remark

Remark

We assume for simplicity that n is a multiple of b so that n/b is an integer. In reality, the smaller problems are often of size equal to the nearest integers either less than or equal to, or greater than or equal to, n/b.

Example

Example 48 (Binary search)

The binary search algorithm reduces the search for an element in a search sequence of size n to the binary search for this element in a search sequence of size $n/2$, when n is even. Two comparisons are needed to implement this reduction (one to determine which half of the list to use and the other to determine whether any terms of the list remain). Hence, if $T(n)$ is the number of comparisons required to search for an element in a search sequence of size n, then

$$T(n) = T(n/2) + 2,$$

when n is even.

Example

Example 49 (Finding maximum and minimum)

Consider the following algorithm for locating the maximum and minimum elements of a sequence $a_1, a_2, \ldots, a_n$. If $n = 1$, then a_1 is the maximum and the minimum. If $n > 1$, split the sequence into two sequences of the same length or of lengths that differ by one. The problem is reduced to finding the maximum and minimum of each of the two smaller sequences. The solution to the original problem results from the comparison of the separate maxima and minima of the two smaller sequences to obtain the overall maximum and minimum.

Let $T(n)$ be the total number of comparisons needed to find the maximum and minimum elements of the sequence with n elements. We have shown that a problem of size n can be reduced into two problems of size $n/2$, when n is even, using two comparisons, one to compare the maxima of the two sequences and the other to compare the minima of the two sequences. This gives the recurrence relation $T(n) = 2\,T(n/2) + 2$, when n is even.

Example

Example 50 (Mergesort)

The mergesort algorithm splits a list to be sorted with n items, where n is even, into two lists with $n/2$ elements each, and uses fewer than n comparisons to merge the two sorted lists of $n/2$ items each into one sorted list. Consequently, the number of comparisons used by mergesort to sort a list of n elements is less than $T(n)$, where $T(n)$ satisfies the divide-and-conquer recurrence relation

$$T(n) = 2\,T(n/2) + n.$$

Special Case of Master Theorem

Theorem 5.1

Let T be an increasing function that satisfies the recurrence relation

$$T(n) = a\,T(n/b) + c$$

whenever n is divisible by b, where $a \geq 1$, $b > 1$ is an integer, and c is a positive real number. Then

$$T(n) = \begin{cases} \Theta(n^{\log_b a}) & \text{if } a > 1 \\ \Theta(\log n) & \text{if } a = 1. \end{cases}$$

Furthermore, when $n = b^k$, where k is a positive integer,

$$T(n) = C_1 n^{\log_b a} + C_2,$$

where $C_1 = T(1) + c/(a-1)$ and $C_2 = -c/(a-1)$.

Proof

Proof.

We first consider the case when $n = b^k$. Iterating the recurrence relation, we get

$$\begin{aligned} T(n) &= a\,T(n/b) + c \\ &= a^2\,T(n/b^2) + ac + c \\ &= a^3\,T(n/b^3) + a^2c + ac + c \\ &\vdots \\ &= a^k\,T(1) + c\sum_{j=0}^{k-1} a^j. \end{aligned}$$

Proof (cont'd)

Proof (cont'd).

If $a = 1$, then $T(n) = T(1) + ck = T(1) + c\log_b n$. When n is not a power of b, we have $b^k < n < b^{k+1}$, where k is a nonnegative integer. Because T is increasing, it follows that $T(n) \le T(b^{k+1}) = T(1) + c(k+1) = (T(1) + c) + ck \le (T(1) + c) + c\log_b n$. Therefore, in both cases, $T(n)$ is $\Theta(\log n)$ when $a = 1$.

Proof (cont'd)

Proof (cont'd).

Now suppose $a > 1$. First consider $n = b^k$, where k is a positive integer. Iterating the recurrence relation, we get

$$\begin{aligned} T(n) &= a^k\,T(1) + c\sum_{j=0}^{k-1} a^j \\ &= a^k\,T(1) + \frac{c(a^k - 1)}{a - 1} \\ &= a^k\left[T(1) + \frac{c}{a-1}\right] - \frac{c}{a-1} \\ &= C_1 n^{\log_b a} + C_2, \end{aligned}$$

because $a^k = a^{\log_b n} = n^{\log_b a}$, where $C_1 = T(1) + c/(a-1)$ and $C_2 = -c/(a-1)$.

Proof (cont'd)

Proof (cont'd).

Now suppose n is not a power of b. Then $b^k < n < b^{k+1}$, where k is a nonnegative integer. Because T is increasing,

$$T(n) \le T(b^{k+1}) = C_1 a^{k+1} + C_2 = (C_1 a)a^k + C_2 = (C_1 a)n^{\log_b a} + C_2.$$

Hence, for both cases, $T(n)$ is $\Theta(n^{\log_b a})$. □

Examples

Example 51

Let $T(n) = 5\,T(n/2) + 3$ and $T(1) = 7$. Find $T(2^k)$, where k is a positive integer. Also, estimate $T(n)$ if T is an increasing function.

Example 52

Estimate the number of comparisons used by a binary search.

Example 53

Estimate the number of comparisons used to locate the maximum and minimum elements in a sequence using the algorithm given above.

Master Theorem

Theorem 5.2

Let T be an increasing function that satisfies the recurrence relation

$$T(n) = a\,T(n/b) + cn^d$$

whenever $n = b^k$, where k is a positive integer, $a \geq 1$, $b > 1$ is an integer, and c and d are real numbers with c positive and d nonnegative. Then

$$T(n) = \begin{cases} \Theta(n^d) & \text{if } a < b^d \\ \Theta(n^d \log n) & \text{if } a = b^d \\ \Theta(n^{\log_b a}) & \text{if } a > b^d. \end{cases}$$

Proof

Proof.

For simplicity, we only prove the special case when $n = b^k$, where k is a positive integer. Extension to the general case can be done like that for Theorem 5.1.
By iterating the recurrence relation, we have

$$\begin{aligned} T(n) &= a\,T(n/b) + cn^d \\ &= a^2\,T(n/b^2) + (a/b^d)cn^d + cn^d \\ &= a^3\,T(n/b^3) + (a/b^d)^2 cn^d + (a/b^d)cn^d + cn^d \\ &\;\;\vdots \\ &= a^k\,T(1) + (a/b^d)^{k-1} cn^d + (a/b^d)^{k-2} cn^d + \cdots + (a/b^d)cn^d + cn^d. \end{aligned}$$

Proof (cont'd)

Proof (cont'd).

If $a = b^d$, then

$$\begin{aligned} T(n) &= a^k\,T(1) + ckn^d \\ &= T(1)n^{\log_b a} + cn^d \log_b n \\ &= T(1)n^d + cn^d \log_b n. \end{aligned}$$

It follows that $T(n)$ is $\Theta(n^d \log n)$.

Proof (cont'd)

Proof (cont'd).

When $a \neq b^d$,

$$\begin{aligned} T(n) &= a^k\, T(1) + \frac{1-(a/b^d)^k}{1-a/b^d} cn^d \\ &= T(1)a^k + \frac{b^d c}{b^d - a} n^d - \frac{b^d c}{b^d - a} a^k \\ &= \frac{b^d c}{b^d - a} n^d + \left[T(1) - \frac{b^d c}{b^d - a} \right] n^{\log_b a}. \end{aligned}$$

If $a < b^d$, then $\log_b a < d$ and so $T(n)$ is $\Theta(n^d)$. If $a > b^d$, then $\log_b a > d$ and so $T(n)$ is $\Theta(n^{\log_b a})$. □

Examples

Example 54

What is the complexity of mergesort?

Recurrence Inequalities

Sometimes the recurrence relations cannot be expressed as an equation, but an inequality, e.g.

$$T(n) \leq 2\, T(n/2) + cn.$$

Master Theorem for Recurrence Inequalities

Theorem 6.1

Let S be an increasing function that satisfies the recurrence relation

$$S(n) = \begin{cases} a\, S(n/b) + cn^d & \text{if } n > 1 \\ h & \text{if } n = 1 \end{cases}$$

whenever $n = b^k$, where k is a nonnegative integer, $a \geq 1$, $b > 1$ is an integer, and c, d, and h are real numbers with c positive and d and h nonnegative. Let T be a solution to the following recurrence inequality

$$T(n) \leq \begin{cases} a\, T(n/b) + cn^d & \text{if } n > 1 \\ h & \text{if } n = 1. \end{cases}$$

Then $T(n) \leq S(n)$ for all $n = b^k \geq 1$.

Proof

Proof.

We use a proof by the principle of mathematical induction. Let $P(b^k)$ be the proposition that $T(b^k) \leq S(b^k)$.

- Basis step:
  $P(b^0) = P(1)$ is true, because it is given that $T(1) \leq h = S(1)$.
- Inductive step:
  For the inductive hypothesis, we assume that $P(b^k)$ is true for an arbitrary nonnegative integer k, i.e., $T(b^k) \leq S(b^k)$. To prove that $P(b^{k+1})$ is true, we note that

$$T(b^{k+1}) \leq a\,T(b^k) + c(b^{k+1})^d \leq a\,S(b^k) + c(b^{k+1})^d = S(b^{k+1}).$$

  This shows that $P(k+1)$ is true.

By the principle of mathematical induction, we can conclude that $T(b^k) \leq S(b^k)$ for all nonnegative integers k. □

Remark

Remark

The result still holds even if we change cn^d to a more general function $g(n)$.

Corollary

Corollary 6.2 (Master theorem for recurrence inequalities)

Let T be an increasing function that satisfies the following recurrence inequality

$$T(n) \leq \begin{cases} a\,T(n/b) + cn^d & \text{if } n > 1 \\ h & \text{if } n = 1 \end{cases}$$

whenever $n = b^k$, where k is a nonnegative integer, $a \geq 1$, $b > 1$ is an integer, and c, d, and h are real numbers with c positive and d and h nonnegative. Then the conclusions of the master theorem for recurrence relations hold for T with Θ replaced by O, i.e.

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

Proof

Proof.

Define S as in Theorem 6.1. Then S satisfies the conclusions of the master theorem for recurrence relations. By Theorem 6.1, $T(n) \leq S(n)$. So the conclusions of the master theorem also hold for T. □

Example

Example 55

What is the complexity of any solution to the recurrence inequality

$$T(n) \leq 2\,T(n/2) + cn.$$

Induction Proofs for Recurrence Inequalities

Example 56

Prove by induction that for any function T defined on the nonnegative integral powers of 2, if

$$T(n) \leq 2\,T(n/2) + cn,$$

for some constant c, then $T(n)$ is $O(n \log n)$.

Solution. We want to show that there exist positive constants C and k such that $T(n) \leq Cn \log n$ whenever $n > k$.
We cannot expect $T(n) \leq Cn \log n$ to hold for $n = 1$, because $\log 1 = 0$.
To have $T(2) \leq C \cdot 2 \log 2 = 2C$, we must choose $C \geq T(2)/2$. This is the first assumption we must make about C.

Solution (cont'd)

For our inductive hypothesis, we assume that $T(n) \leq Cn \log n$ holds for an arbitrary positive integer j such that $n = 2^j$, i.e., $T(2^j) \leq C \cdot 2^j \cdot j$. We want to show that $T(2^{j+1}) \leq C \cdot 2^{j+1} \cdot (j+1)$. From the recurrence inequality and inductive hypothesis, we have

$$\begin{aligned} T(2^{j+1}) &\leq 2\,T(2^j) + c \cdot 2^{j+1} \\ &\leq 2C \cdot 2^j \cdot j + c \cdot 2^{j+1} \\ &= C \cdot 2^{j+1} \cdot j + c \cdot 2^{j+1}. \end{aligned}$$

To show that $T(2^{j+1}) \leq C \cdot 2^{j+1} \cdot (j+1)$, we need to make sure that

$$\begin{aligned} C \cdot 2^{j+1} \cdot j + c \cdot 2^{j+1} &\leq C \cdot 2^{j+1} \cdot (j+1) \\ c \cdot 2^{j+1} &\leq C \cdot 2^{j+1} \\ c &\leq C. \end{aligned}$$

Therefore, if $C \geq \max(T(2)/2, c)$, then $T(n) \leq Cn \log n$ whenever $n > k = 1$. Consequently, $T(n)$ is $O(n \log n)$.

Example

Example 57

Suppose that c is a positive real number. Prove by induction that any solution $T(n)$ to the recurrence inequality

$$T(n) \leq T(n/3) + cn,$$

with n restricted to integer powers of 3, has $T(n) = O(n)$.

Solution. We want to show that there exist positive constants C and k such that $T(n) \leq Cn$ whenever $n > k$.
For the base case, we choose $n = 1$ (i.e., $k = 0$). This requires that $C \geq T(1)$.

Solution (cont'd)

For the inductive hypothesis, we assume that $T(n) \leq Cn$ holds for an arbitrary nonnegative integer j such that $n = 3^j$, i.e., $T(3^j) \leq C \cdot 3^j$. We want to show that $T(3^{j+1}) \leq C \cdot 3^{j+1}$. From the recurrence inequality and inductive hypothesis, we have

$$\begin{aligned} T(3^{j+1}) &\leq T(3^j) + c \cdot 3^{j+1} \\ &\leq C \cdot 3^j + c \cdot 3^{j+1}. \end{aligned}$$

To show that $T(3^{j+1}) \leq C \cdot 3^{j+1}$, we need to make sure that

$$\begin{aligned} C \cdot 3^j + c \cdot 3^{j+1} &\leq C \cdot 3^{j+1} \\ 3c \cdot 3^j &\leq 2C \cdot 3^j \\ \frac{3c}{2} &\leq C. \end{aligned}$$

Therefore, if $C \geq \max(T(1), 3c/2)$, then $T(n) \leq Cn$ whenever $n > k = 0$. Consequently, $T(n)$ is $O(n)$.

Example

Example 58

Suppose that c is a positive real number. Prove by induction that any solution $T(n)$ to the recurrence inequality

$$T(n) \leq 4\,T(n/2) + cn,$$

with n restricted to integer powers of 2, has $T(n) = O(n^2)$.

Solution. We want to show that there exist positive constants C and k such that $T(n) \leq Cn^2$ whenever $n > k$.
For the base case, we choose $n = 1$ (i.e., $k = 0$). This requires that $C \geq T(1)$.

Solution (cont'd)

For the inductive hypothesis, we assume that $T(n) \leq Cn^2$ holds for an arbitrary nonnegative integer j such that $n = 2^j$, i.e., $T(2^j) \leq C(2^j)^2$. We want to show that $T(2^{j+1}) \leq C(2^{j+1})^2$. From the recurrence inequality and inductive hypothesis, we have

$$\begin{aligned} T(2^{j+1}) &\leq 4\,T(2^j) + c \cdot 2^{j+1} \\ &\leq 4C \cdot (2^j)^2 + c \cdot 2^{j+1}. \end{aligned}$$

To show that $T(2^{j+1}) \leq C(2^{j+1})^2$, we need to make sure that

$$\begin{aligned} 4C \cdot (2^j)^2 + c \cdot 2^{j+1} &\leq C(2^{j+1})^2 \\ c \cdot 2^{j+1} &\leq 0. \end{aligned}$$

Unfortunately we cannot choose a value for C to make this possible.

Solution (cont'd)

We need a stronger inductive hypothesis. To do this, we want to show that there exist constants C_1, C_2, and k such that $T(n) \leq C_1 n^2 - C_2 n$ whenever $n > k$.
For the base case, we choose $n = 1$ (i.e., $k = 0$). This implies that $T(1) \leq C_1 - C_2$.
For the inductive hypothesis, we assume that $T(n) \leq C_1 n^2 - C_2 n$ holds for an arbitrary nonnegative integer j such that $n = 2^j$, i.e., $T(2^j) \leq C_1(2^j)^2 - C_2 \cdot 2^j$. We want to show that $T(2^{j+1}) \leq C_1(2^{j+1})^2 - C_2 \cdot 2^{j+1}$. From the recurrence inequality and inductive hypothesis, we have

$$\begin{aligned} T(2^{j+1}) &\leq 4\,T(2^j) + c \cdot 2^{j+1} \\ &\leq 4C_1 \cdot (2^j)^2 - 4C_2 \cdot 2^j + c \cdot 2^{j+1}. \end{aligned}$$

Solution (cont'd)

To show that $T(2^{j+1}) \leq C_1(2^{j+1})^2 - C_2 \cdot 2^{j+1}$, we need to make sure that

$$\begin{aligned} 4C_1 \cdot (2^j)^2 - 4C_2 \cdot 2^j + c \cdot 2^{j+1} &\leq C_1(2^{j+1})^2 - C_2 \cdot 2^{j+1} \\ c \cdot 2^{j+1} &\leq 2C_2 \cdot 2^j \\ c &\leq C_2. \end{aligned}$$

Suppose we choose $C_2 = c$. Then we can choose C_1 such that $C_1 \geq T(1) + C_2 = T(1) + c$. Consequently, $T(n)$ is $O(n^2)$.