Discrete Mathematics

INDUCTION & RECURSION

Principles and Applications

FPT Education
FPT UNIVERSITY

Department of Mathematics

Võ Văn Nam, Ph.D.

Competency Goals

1. Prove mathematical statements using the principle of induction.
2. Manipulate sequences and sets defined through recursion.
3. Design and implement recursive algorithms, including the mergesort algorithm, and differentiate between iterative and recursive approaches.

Table of Contents

What's next?

Principle of Mathematical Induction

Problem. Prove that the statement $P(n)$ is true for all $n = 1, 2, \ldots$

Proof by Induction:

1. **Basis step**. Prove that $P(1)$ is true.
2. **Inductive hypothesis**. Assume that $P(k)$ is true for some positive integer k.
3. **Inductive step**. Show that $P(k+1)$ is true.
4. **Conclusion**. $P(n)$ is true for all positive integers n.

Illustration of Mathematical Induction

Example

Show that if n is a positive integer, then

$$1 + 2 + \cdots + n = \frac{n(n+1)}{2}.$$

Solution. Let $P(n)$ be the proposition that $1 + 2 + \cdots + n = \frac{n(n+1)}{2}$.

1. $P(1)$ is true since $1 = \frac{1(1+1)}{2}$.
2. Assume that $P(k)$ holds for an arbitrary positive integer k, namely
$$1 + 2 + \cdots + k = \frac{k(k+1)}{2}.$$
3. We now prove that $P(k+1)$ is true. Indeed,
$$1 + 2 + \cdots + k + (k+1) = \frac{k(k+1)}{2} + (k+1) = \frac{k(k+1)+2(k+1)}{2} = \frac{(k+1)(k+2)}{2}.$$
4. Hence, $P(n)$ is true for all positive integers n.

Questions

1. Show that for all nonnegative integers n,
$$1 + 2 + 2^2 + \cdots + 2^n = 2^{n+1} - 1.$$
2. Prove that $n^3 - n$ is divisible by 3 for all integers $n \geq 1$.
3. Show that $2^n > n^2$ for all integers $n > 4$.
4. The **harmonic numbers** H_n, $n = 1, 2, 3, \ldots$ are defined by
$$H_n = 1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{n}.$$
Prove that for n is a nonnegative integer, $H_{2^n} \geq 1 + \dfrac{n}{2}$.
5. Let n be a positive integer. Prove that every checkerboard of size $2^n \times 2^n$ with one square removed can be titled by L-shaped tiles.

What's next?

Problem. Prove that $P(n)$ is true for all $n = 1, 2, \ldots$

Proof by Strong Induction:

1. Prove that $P(1)$ is true.
2. Assume that $P(1), P(2), \ldots, P(k)$ are true for some $k \geq 1$.
3. Show that $P(k+1)$ is also true.
4. *Conclusion*: $P(n)$ is true for all positive integers n.

Example: Strong Induction

Prove that every integer greater than 1 can be written as a product of primes.

Solution. Let $P(n)$ be the proposition that n can be written as the product of primes.

1. $P(2)$ is true since $2 = 2$.
2. Assume that $P(j)$ is true for all integer j with $2 \leq j \leq k$.
3. We need to show that $P(k+1)$ is true under this assumption.
   Case 1. $k + 1$ is prime. Obviously, $P(n)$ is true.
   Case 2. $k + 1$ is composite and can be written as the product of two positive integers a and b with $2 \leq a \leq b < k + 1$. Because both a and b are integers at least 2 and not exceeding k, we can use inductive hypothesis to write both of them as the product of primes. Thus, $P(k+1)$ is true.
4. Hence, $P(n)$ is true for all integer greater than 1.

FPT Education
FPT UNIVERSITY

Questions

Question 1. You have coins of denominations 3 cents and 5 cents. Prove that you can form any amount of money greater than or equal to 8 cents using these coins.

Question 2. Prove that every positive integer n can be expressed as a sum of distinct non-consecutive Fibonacci numbers.

Question 3. You have a chocolate bar consisting of $m \times n$ squares. Each time you break it, you can only break along a straight line that divides the chocolate into two smaller pieces. Prove that it takes exactly $mn - 1$ breaks to split the chocolate into individual squares.

Using Strong Induction in Computational Geometry

A **polygon** is a closed geometric figure consisting of a sequence of line segments $s_1, s_2, \ldots, s_n$ is called **sides**.

A **diagonal** of a simple polygon is a line segment connecting two nonconsecutive vertices of the polygon, and a diagonal is called an **interior diagonal** if it lies inside the polygon, except for its endpoints.

Theorem

- A simple polygon with n sides, where n is an integer with $n \geq 3$, can triangulated into $n - 2$ triangles.
- (Lemma) Every simple polygon with at least four sides has an interior diagonal.

Well-Ordering

A set S is **well-ordered** if every non-empty subset of S has a **least element** under a given ordering. This means that for any non-empty subset T of S, there is an element m in T such that $m \leq x$ for all x in T.

The validity of the Principle of Mathematical Induction follows from the Well-Ordering property of the set of non-negative integers.

Example. The set of natural numbers $\mathbb{N} = \{0, 1, 2, 3, \ldots\}$ is well-ordered under the usual ordering $\leq$. For any non-empty subset of $\mathbb{N}$, there is always a smallest element. For example, in the subset $\{3, 7, 9\}$, the smallest element is 3.

FPT Education
FPT UNIVERSITY

What's next?

FPT Education
FPT UNIVERSITY

Recursive Definitions

We use two steps to define a function with the set of nonnegative integers in its domain:

1. **Basic step**: Specify the value of the function at zero.
2. **Recursive step**: Give a rule for finding its value at an integer from its values at smaller integers.

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

Solution.

1. a^0 is specified, $a^0 = 1$.
2. The rule for finding a^{n+1} from a^n is given by

$$a^{n+1} = a \cdot a^n, \ n = 0, 1, 2, \cdots$$

These two equations uniquely define a^n for all nonnegative integers n.

Recursively Defined Sets and Structures

Determine the set S defined by:

- **Basic step**: $3 \in S$.
- **Recursive step**: If $x, y \in S$ then $x + y \in S$.

Solution. We have

- the new elements found to be in S are 3 by the basic step,
- the first application of the recursive step $3 + 3 = 6$,
- the second application the recursive step $3 + 6 = 6 + 3 = 9$, and $6 + 6 = 12$,
- $\ldots$
- We will show that S is the set of all positive multiples of 3.

FPT Education
FPT UNIVERSITY

Questions

1. Give a recursive definition of the set of positive integers that are multiples of 5.
2. Give a recursive definition for the set of positive integers that are not divisible by 3.
3. Give a recursive definition of the set of positive integers congruent to 2 modulo 3.
4. Give a recursive definition for the set of integers that are not divisible by 3.

Example and Question

Example. The Fibonacci sequence $\{F_n\}$, where $n = 0, 1, 2, \ldots$, is defined as follows:

$$F_0 = 0, \quad F_1 = 1, \text{ and}$$
$$F_n = F_{n-1} + F_{n-2} \quad \text{for } n = 2, 3, \ldots$$

Question.

1. Find the n-th term of the sequence $\{x_n\}$ defined by the following recursive definitions:
   1. $x_1 = 5, \quad x_n = 3x_{n-1}$ for $n = 2, 3, \ldots$
   2. $x_0 = 2, \quad x_n = x_{n-1} + 1$ for $n = 1, 2, \ldots$
2. Give a recursive definition for the sequence $\{x_n\}, n = 1, 2, \ldots$ whose n-th term is:
   1. $x_n = 7 \cdot 5^{n+1}$.
   2. $x_n = n!$.
   3. $x_n = (-1)^n$.
   4. $x_n = 2n - 6$.

The set Σ^* of **strings** over the alphabet Σ is defined recursively by

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

Note.

- The basic step says that the empty string belongs to Σ^*.
- The recursive step states that new strings are produced by adding a symbol from Σ to the end of strings in Σ^*.
- At each application of the recursive step, strings containing one additional symbol are generated.

Example

Assume $\Sigma = \{0, 1\}$. Then

- the strings found to be in Σ^*, the set of all bit strings are λ, specified to be in Σ^* in the basic step.
- the first application of the recursive step, 0 and 1 are formed.
- the second application of the recursive step, $00, 01, 10, 11$ are formed.

FPT Education
FPT UNIVERSITY

Concatenation of two strings

Let Σ be a set of symbols and Σ^* be the set of strings formed from symbols in Σ. We define the **concatenation of two strings**, denoted as $\cdot$, recursively as follows:

- **Basic step**: If $w \in \Sigma^*$, then $w \cdot \lambda = w$ where λ is the empty string.
- **Recursive step**: If $w_1, w_2 \in \Sigma^*$ and $x \in \Sigma$, then $w_1 \cdot (w_2 x) = (w_1 \cdot w_2)x$.

Give a recursive definition of $l(w)$, the length of the string w. The **length of a string** can be recursively defined by

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

FPT Education
FPT UNIVERSITY

Building Up Rooted Trees

The set of **rooted trees**, where a rooted tree consists of a set of vertices containing a distinguished vertex called the root, and edges connecting these vertices, can be defined recursively by these steps:

- **Basics step**: A single vertex r is a rooted tree.
- **Recursive step**: Suppose that $T_1, T_2, \ldots, T_n$ are disjoint rooted trees with roots $r_1, r_2, \ldots, r_n$, respectively. Then the graph formed by starting with a root r, which is not in any of the rooted trees $T_1, T_2, \ldots, T_n$ and adding an edge from r to each of the vertices $r_1, r_2, \ldots, r_n$, is also rooted tree.

Extended binary trees

The set of **extended binary trees** can be defined recursively by these steps:

- **Basic step**: The empty set is an extended binary tree.
- **Recursive step**: If T_1 and T_2 are disjoint extended binary trees, there is an extended binary tree, denoted by $T_1 \cdot T_2$, consisting 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 when these trees are nonempty.

Building Up Full Binary Trees

Recursive definition for the set of **full binary trees**.

- **Basic step**: A single vertex is a full binary tree.
- **Recursive step**: If T_1 and T_2 are two full binary trees, then there is a full binary tree, denoted by $T_1.T_2$, consisting of a root r together with edges connecting this root to the root of the left subtree T_1 and the root of the right subtree T_2.

Question

Give a recursive definition for:

1. Leaves of full binary trees.
2. Height of full binary trees.

Structural Induction

Let S be a set defined recursively. To prove that a property P is true for all elements of S, we can use **structural induction**.

- **Basic step**: Prove that P is true for elements of S defined in the basic step.
- **Recursive step**: Show that if the property P is true for the elements used to construct new elements in the recursive step of the definition of S, then the property P is also true for these new elements.

Question.

1. Show that the set S where $3 \in S$ and if $x, y \in S$ implies $x + y \in S$, is the set of all positive integers that are multiples of 3.
2. Let T be a full binary tree with the number of vertices $n(T)$ and the number of leaves $\ell(T)$. Prove that $n(T) = 2\ell(T) - 1$.

We define the height $h(T)$ of a full binary tree T recursively.

- **Basic step**: The height of the full binary tree T consisting of only a root r is $h(T) = 0$.
- **Recursive step**: If T_1 and T_2 are full binary trees, then the full binary tree $T = T_1 \cdot T_2$ has height $h(T) = 1 + \max(h(T_1), h(T_2))$.

Theorem

Let T be a full binary tree with the number of vertices $n(T)$ and the height $h(T)$. Then, $n(T) \leq 2^{h(T)+1} - 1$.

Generalized Induction

Example. Given the sequence $\{a_{m,n}\}$ defined recursively as follows:

$$a_{0,0} = 0, \text{ and}$$

$$a_{m,n} = \begin{cases} a_{m-1,n} + 1 & \text{if } n = 0 \text{ and } m > 0 \\ a_{m,n-1} + n & \text{if } n > 0. \end{cases}$$

Prove that $a_{m,n} = m + \dfrac{n(n+1)}{2}$ for all $m, n \geq 0$.

What's next?

An algorithm is called **recursive** if it solves a problem by reducing it to an instance of the same problem with smaller input.

Example. A recursive algorithm that computes 5^n for $n \geq 0$.
Solution.

Procedure power (n: nonnegative)
if $n = 0$ then power$(0) := 1$
else power(n) :=power$(n-1) * 5$

Questions

1. Write a recursive algorithm to compute $n!$.
2. Write a recursive algorithm to compute the greatest common divisor of two nonnegative integers.
3. Express the linear search algorithm by a recursive procedure.
4. Express the binary search algorithm by a recursive procedure.

Recursion and Iteration

Problem. Write a recursive algorithm and an iteration algorithm to compute the nth Fibonacci number, and compare their complexity via the number of additions used.

Procedure Iterative Fib (n)
if $n = 0$ then $y := 0$
else
  $x := 0$
  $y := 1$
  for $i := 1$ to $n - 1$ do
    $z := x + y$
    $x := y$
    $y := z$
Print(y)

Procedure Fib (n)
if $n = 0$ then $\mathrm{Fib}(0) := 0$
else if $n = 1$ then $\mathrm{Fib}(1) := 1$
else
$\mathrm{Fib}(n) := \mathrm{Fib}(n-1) + \mathrm{Fib}(n-2)$

Merge Sort Algorithm

Procedure mergesort $(L = a_1, a_2, \ldots, a_n)$
if $n > 1$ **then**
  $m := \lfloor n/2 \rfloor$
  $L_1 = a_1, a_2, \ldots, a_m$
  $L_2 := a_{m+1}, a_{m+2}, \ldots, a_n$
  $L :=$ merge(mergesort(L_1), mergesort(L_2))
Print(L)

Theorem

The number of comparisons needed to merge sort a list with n elements is $O(n \log n)$.

Quiz

1. By induction hypothesis, for any positive integer n, the sum

$$1 * (1!) + 2 * (2!) + ... + n * (n!)$$

can be equivalent to:

A. $n^2 - 1$
B. $(n + 1)! - 1$
C. $(n + 2)! - 2$
D. $(n + 2)! - n$

2. TRUE or FALSE?
   - $1 + 2 + 3 + ... + n = 2n - 1$ for all integer $n > 0$
   - $1 + 3 + 5 + 7 + ... + (2n - 1) = n^2$ for all integers $n > 0$

FPT Education
FPT UNIVERSITY