Discrete Mathematics

Recursively defined sets, functions, structures

Srinivasan, Assistant Professor

Amrita Vishwa Vidyapeetham, Coimbatore, INDIA

Recursive Definitions

- To define an object in terms of itself is called recursion
- We can use recursion to define sequences, functions, and sets

EXAMPLE

- The sequence of powers of 2 is given by $a_n = 2^n$ for n = 0, 1, 2, . . .
- Define a sequence recursively by specifying how terms of the sequence are found from previous terms.

Two Steps

1. Basis step: first term of the sequence: $a_0 = 1$
2. Recursive step: rule to find terms of the sequence from the previous one

$$a_{n+1} = 2a_n \text{ for } n = 0, 1, 2, 3, \cdots n$$

EXAMPLE

Give a recursive definition of the sequence $\{a_n\}$, n =1, 2, 3,… if

$a_n = 2n+1$

Basis step: a_0=1

Recursive step: $a_n = a_{n-1} + 2, n \geq 1$

EXAMPLE

Give a recursive definition for a^n for $a \in R$ and non-negative and is positive integer.

Give a recursive definition of the sequence $\{a_n\}$, n = 1, 2,3, … if $a_n = 5^n$

EXAMPLE

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

Solution: The recursive definition contains two parts. First a^0 is specified, namely, $a^0 = 1$. Then the rule for finding a^{n+1} from a^n, namely, $a^{n+1} = a \cdot a^n$, for $n = 0, 1, 2, 3, \ldots$, is given. These two equations uniquely define a^n for all nonnegative integers n. ◀

Recursive or inductive definition

Recursively Defined Functions

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

BASIS STEP: Specify the value of the function at zero.

RECURSIVE STEP: Give a rule for finding its value at an integer from its values at smaller integers.

EXAMPLE

Suppose that f is defined recursively by

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

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

Solution: From the recursive definition it follows that

$$f(1) = 2f(0) + 3 = 2 \cdot 3 + 3 = 9,$$
$$f(2) = 2f(1) + 3 = 2 \cdot 9 + 3 = 21,$$
$$f(3) = 2f(2) + 3 = 2 \cdot 21 + 3 = 45, \qquad f(4) = 2f(3) + 3 = 2 \cdot 45 + 3 = 93.$$

EXAMPLE

Give a recursive definition of

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

Solution: The first part of the recursive definition is

$$\sum_{k=0}^{0} a_k = a_0.$$

The second part is

$$\sum_{k=0}^{n+1} a_k = \left(\sum_{k=0}^{n} a_k\right) + a_{n+1}.$$

Fibonacci Numbers

Recall the set of Fibonacci numbers

0, 1, 1, 2, 3, 5, 8, 13, 21, …

How is each number related and found?

Lets give a recursive definition to this sequence.

Two steps

1. Basis step: $f_0 = 0, f_1 = 1$
2. Recursive step:

$$f_n = f_{n-1} + f_{n-2},\ n \geq 2$$

Recursively defined sets and structures

EXAMPLE

The set of natural numbers N = {0, 1, 2,….}

Basis step: $0 \in N$

Recursive step: how to find other elements using 0.

if $n \in N \text{ then } n+1 \in N$

EXAMPLE

Let S be a subset of the integers defined recursively by

Basis step: $7 \in S$

Recursively step: $\text{if } x \in S \text{ and } y \in S \text{ then } x+y \in S$

Recursive definitions play an important role in the study of strings

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

BASIS STEP: $\lambda \in \Sigma^*$ (where λ is the empty string containing no symbols).

RECURSIVE STEP: If $w \in \Sigma^*$ and $x \in \Sigma$, then $wx \in \Sigma^*$.

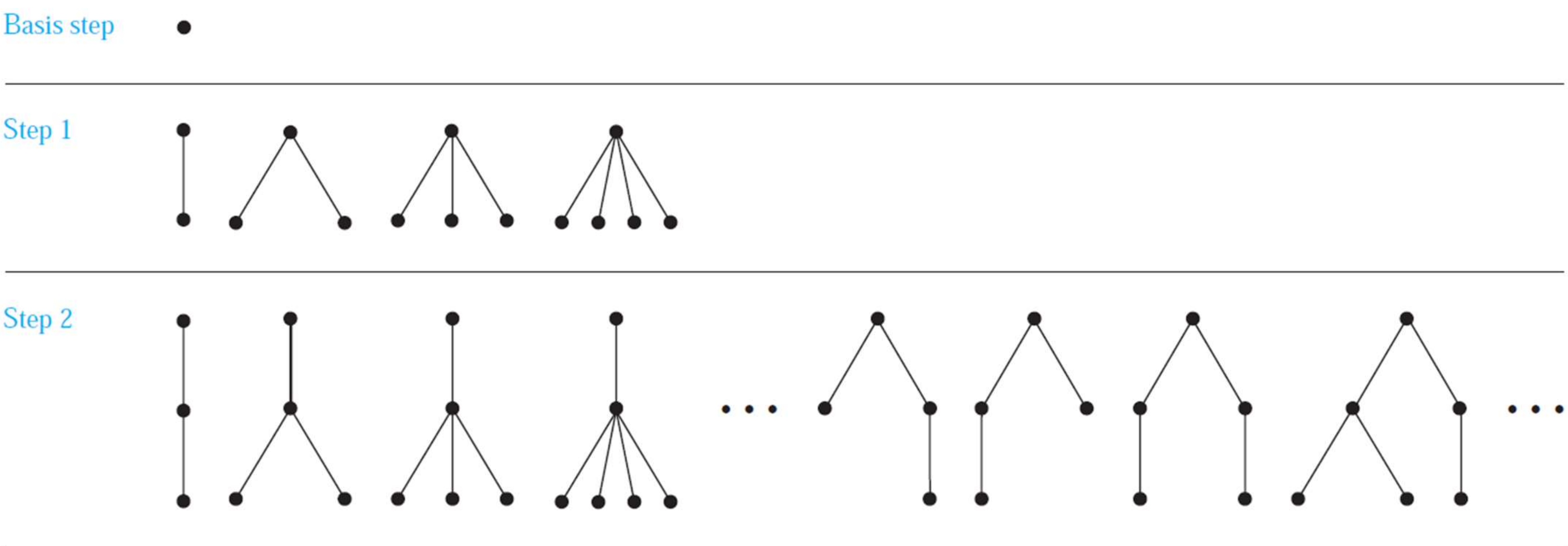
Rooted Trees recursively defined

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:

BASIS 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 a rooted tree.

Building Up Rooted Trees.



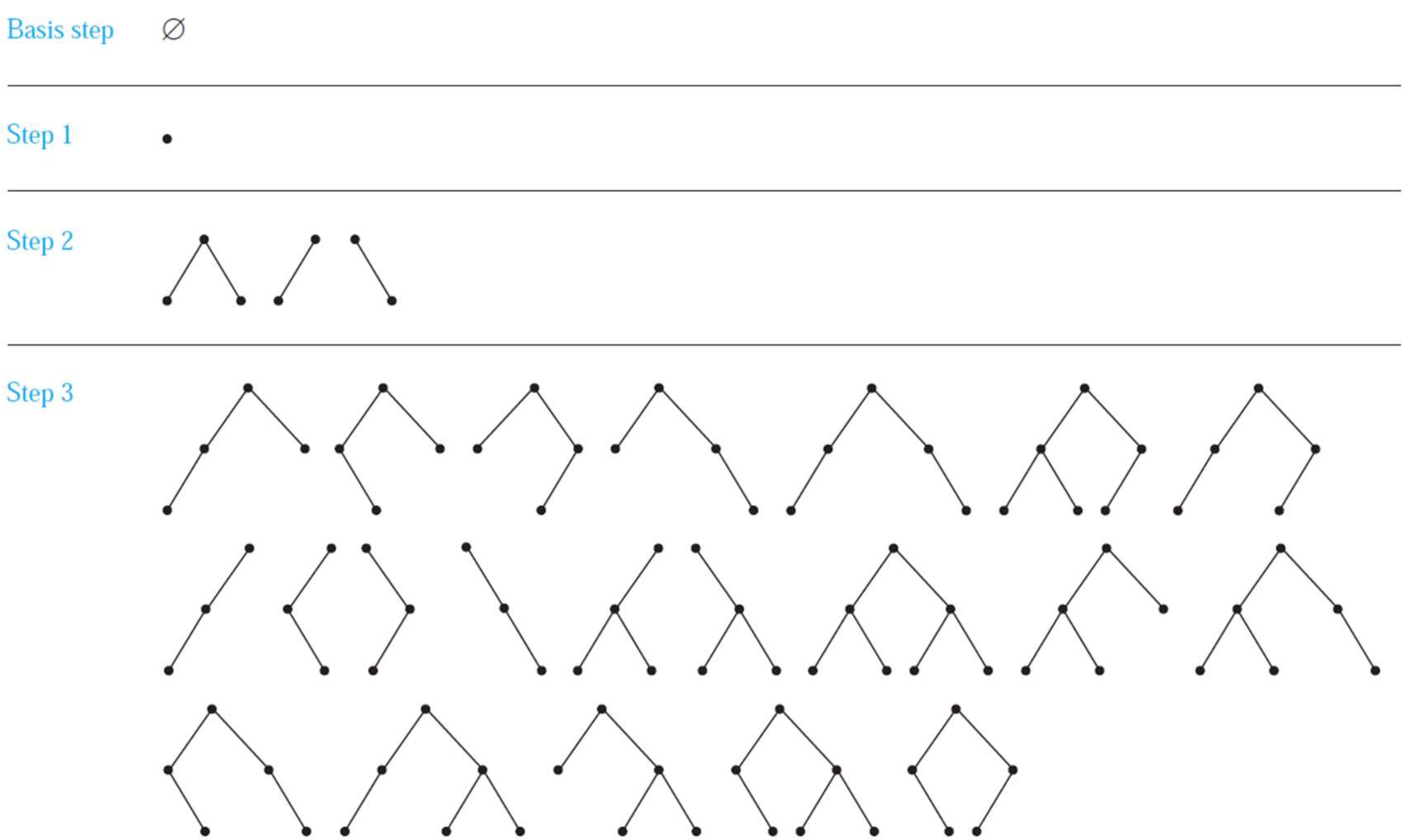
Binary trees are a special type of rooted trees.
We will provide recursive definitions of two types of binary trees – full binary trees and extended binary trees

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

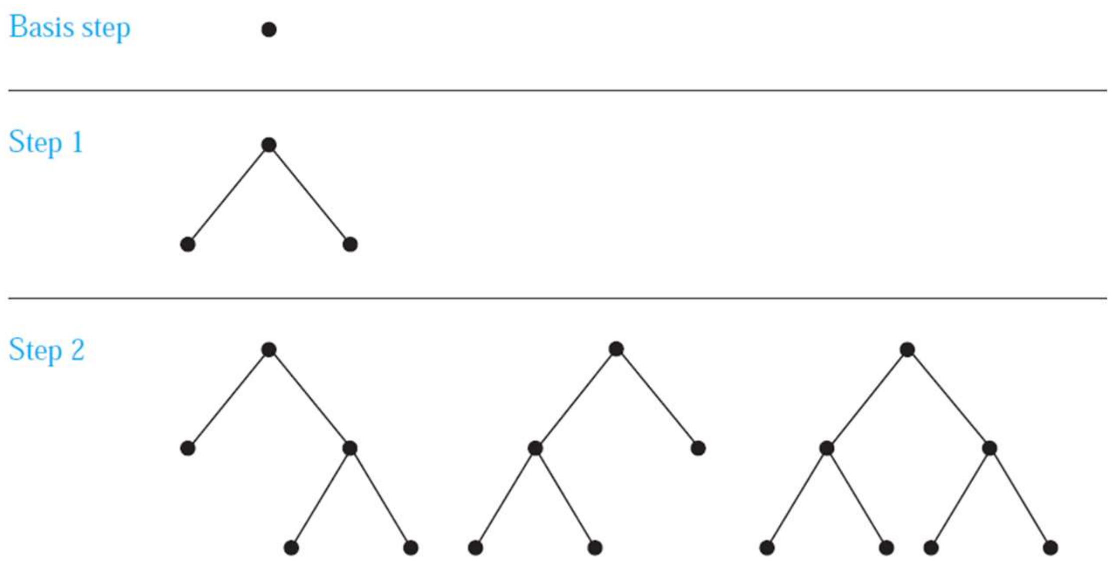
BASIS STEP: The empty set is an extended binary tree.

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 Extended Binary Trees



Building Up Full Binary Trees



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

BASIS STEP: There is a full binary tree consisting only of a single vertex r.

RECURSIVE STEP: If T_1 and T_2 are disjoint full binary trees, there is a full 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.