

ACOL1002: Discrete Mathematical Structures

Ragesh Jaiswal, CSE, IIT Delhi

Induction and Recursion

Induction and Recursion

Recursion

- Recursively defined sets and structures
 - Basis step: An initial collection of elements is specified.
 - Recursive step: Rules for forming new elements in the set from those already known to be in the set are provided.
 - *Exclusion Rule*: A recursively defined set contains nothing other than those elements specified in the basis step or generated by applications of the recursive step.
- Examples:
 - Multiples of 3:
 - Basis step: $3 \in S$
 - Recursive step: If $x \in S$ and $y \in S$, then $x + y \in S$.
 - The set Σ^* of strings over the alphabet Σ :
 - Basis step: ?
 - Recursive step: ?

Induction and Recursion

Recursion

- Examples:
 - Multiples of 3:
 - Basis step: $3 \in S$
 - Recursive step: If $x \in S$ and $y \in S$, then $x + y \in S$.
 - The set Σ^* of strings over the alphabet Σ :
 - 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 $wx \in \Sigma^*$.
 - Well-formed formula in propositional logic:
 - Basis step: T , F , and s , where s is a propositional variable, are well-formed formulae.
 - Recursive step: If E and F are well-formed formulae, then $(\neg E)$, $(E \wedge F)$, $(E \vee F)$, $(E \rightarrow F)$, and $(E \leftrightarrow F)$ are well-formed formulae.

Induction and Recursion

Recursion

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

Induction and Recursion

Structural Induction

- A set S defined recursively:
 - Basis step: $3 \in S$
 - Recursive step: If $x \in S$ and $y \in S$, then $x + y \in S$.
- Let A be the set of all positive integers divisible by 3.
- Show that $S \subseteq A$.
- *Structural Induction*
 - Basis step: Show that the result holds for all elements specified in the basis step of the recursive definition to be in the set.
 - Recursive 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.
- Argue the validity of structural induction.

Induction and Recursion

Structural Induction: Examples

- Well-formed formula in propositional logic:
 - Basis step: T , F , and s , where s is a propositional variable, are well-formed formulae.
 - Recursive step: If E and F are well-formed formulae, then $(\neg E)$, $(E \wedge F)$, $(E \vee F)$, $(E \rightarrow F)$, and $(E \leftrightarrow F)$ are well-formed formulae.
- Show that every well-formed formula for compound propositions, as defined above, contains an equal number of left and right parentheses.

Induction and Recursion

Recursive algorithms

Definition

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

- Example:

Extended-Euclid-GCD(a, b)

 If($b = 0$), then return($a, 1, 0$)

 else

 Compute integers q, r such that $a = qb + r$ and $0 \leq r < b$.

 Let $(d, x, y) = \text{Extended-Euclid-GCD}(b, r)$

 return($d, y, x - yq$)

- Prove that the above algorithm returns the correct answer.

Counting

Counting

Basic counting principles

- What are we going to count?
 - Objects with certain properties.
- Basic Counting Principles:
 - The Product Rule: Suppose that a procedure can be broken down into a sequence of two tasks. If there are n_1 ways to do the first task and for each of these ways of doing the first task, there are n_2 ways to do the second task, then there are $n_1 \cdot n_2$ ways to do the procedure.

Counting

Basic counting principles

- Basic Counting Principles:
 - The Product Rule: Suppose that a procedure can be broken down into a sequence of two tasks. If there are n_1 ways to do the first task and for each of these ways of doing the first task, there are n_2 ways to do the second task, then there are $n_1 \cdot n_2$ ways to do the procedure.
 - Extended Product Rule: Suppose that a procedure is carried out by performing the tasks $T_1, T_2, \dots, T_m$ in sequence. If each task $T_i, i = 1, 2, \dots, m$, can be done in n_i ways, regardless of how the previous tasks were done, then there are $n_1 \cdot n_2 \dots n_m$ ways to carry out the procedure.
 - Can you argue the validity of the extended product rule using the product rule?

Counting

Basic counting principles

- Basic Counting Principles:
 - The Product Rule: Examples
 - How many functions are there from a set of m elements to a set of n elements?
 - How many such one-one functions are there?

End