CSE211 – Formal Languages and Automata Theory

U4L9_Function and Other Models of Computation

Dr. P. Saravanan
School of Computing
SASTRA Deemed to be University

Integer Basic functions

- In the previous session we have discussed Turing machine as integer function
- Different integer functions such as addition, subtraction, multiplication, remainder finding, square, etc. are constructed using the Turing machine.
- These are the basic functions.
- By combining these basic functions, complex functions are constructed.
- As the basic functions are computable, the complex functions are also computable.

Function

- A function is a relation that associates each element of a set with an element of another set.
- Let a function f be defined between two sets A and B.
- There must be a relation between A and B such that for an element $a \in A$, there must be another element $b \in B$, such that (a, b) is in the relation.

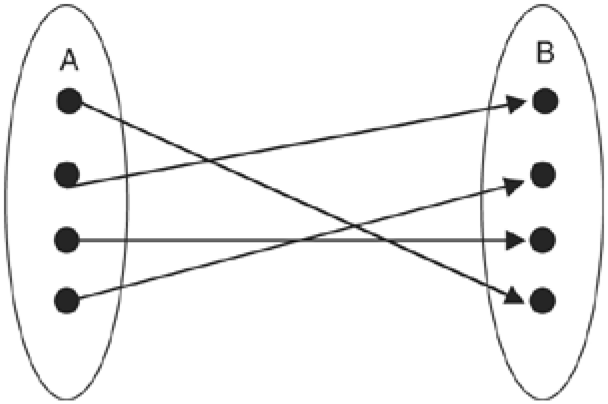
Different Types of Functions

- **1 One to One or Injective**
- A function from set A to set B is said to be one-to-one if no two elements of set A have exactly the same elements mapped in set B.
- In other words, it can be said that $f(x) = f(y)$ if and only if $x = y$.

Injective functions

- ***Example:*** $f(x) = x + 4$. Let $x = 1, 2, 3, \ldots\ldots.$ (set of all positive integer numbers).
- This is an injective function because

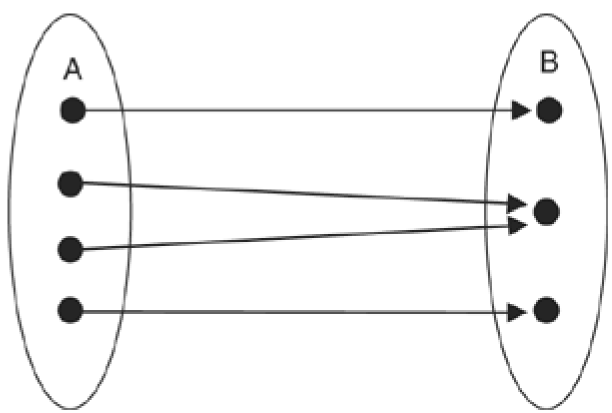
$f(1) = 5$, $f(2) = 6$, $f(3) = 7 \ldots\ldots\ldots$, for no two elements of set A there is exactly the same value in set B.



is a function $f(x) = x^2$, injective?

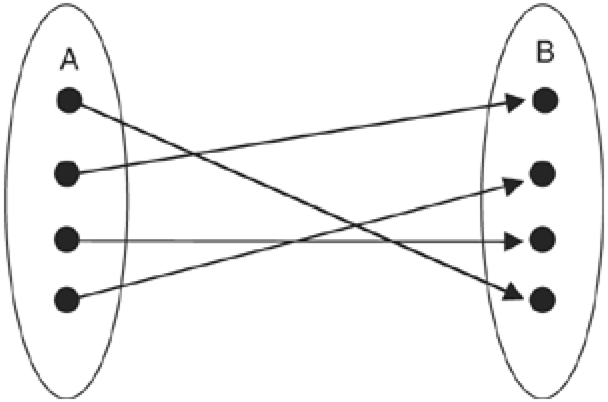
Different Types of Functions

- ## 2 Onto or Surjective
- A function f from set A to a set B is said to be surjective (or onto), if for every $y \in Y$, there is an $x \in X$ resulting in $f(x) = y$. In other words, each element of the set B can be obtained by applying the function f to some element of A.
- Example: $f(x) = 2x$ from $A \rightarrow B$, where A is the set of natural numbers and B is the set of non-negative even numbers. Here, f B, there is an element in A.



Different Types of Functions

- **3 Bijective**
- A function f is said to be bijective if it is both injective and surjective.
- Example: The example $f(x) = 2x$ of surjective is also bijective.
- Diagrammatically, it can be shown as follows.



Different Types of Functions

- **4 Inverse Function**
- Let us define a function f to be a bijection from a set A to set B.
- Suppose another function g is defined from B to A such that for every element y of B, $g(y) = x$, where $f(x) = y$.
- Then, the function g is called the inverse function of f, and it is denoted by $f-1$.
- Example: If $f(x) = 5x$ from the set of natural numbers A to the set of non-negative even numbers B, then $g(x) = f-1(x) = 1/5\ x$.

Different Types of Functions

- **5 Composite Function**
- Let f(x) be a function from a set A to set B, and let g be another function from set B to a set C.
- Then, the composition of the two functions f and g, denoted by fg, is the function from set A to set C that satisfies
- $fg(x) = f(g(x))$ for all x in A.
- Example: $f(x) = x2$ and $g(x) = (x + 2)$. Then, $fg(x) = f(g(x)) = (x + 2)2$

Church's Thesis:

All models of computation are equivalent

Turing's Thesis:

A computation is mechanical if and only if it can be performed by a Turing Machine

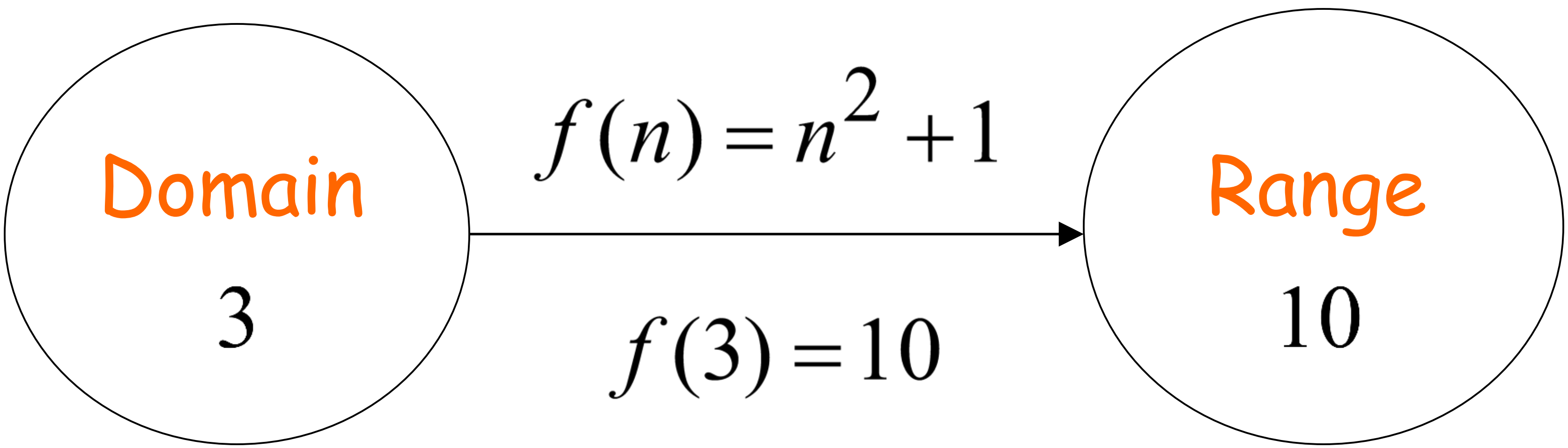
Models of computation:

- Turing Machines
- Recursive Functions
- Post Systems
- Rewriting Systems

Recursive Functions

General Recursive functions are partial functions that take finite tuples of natural numbers and return a single natural number

An example function:



We need a way to define functions

We need a set of basic functions

Basic Primitive Recursive Functions

Zero function: $z(x) = 0$

Successor function: $s(x) = x + 1$

Projection functions: $p_1(x_1, x_2) = x_1$

$$p_2(x_1, x_2) = x_2$$

Building complicated functions:

Composition: $$f(x,y) = h(g_1(x,y), g_2(x,y))$$

Primitive Recursion:

$$f(x,0) = g_1(x)$$

$$f(x,y+1) = h(g_2(x,y), f(x,y))$$

Any function built from
the basic primitive recursive functions
is called:

Primitive Recursive Function

A Primitive Recursive Function: $add(x, y)$

$$add(x,0) = x$$

(projection)

$$add(x, y+1) = add(x, y) + 1$$

(successor function)

$$add(x,0) = x$$

$$add(x, y+1) = add(x, y) + 1$$

$$\begin{aligned} add(3,2) &= add(3,1) + 1 \\ &= (add(3,0) + 1) + 1 \\ &= (3+1) + 1) \\ &= 4 + 1 \\ &= 5 \end{aligned}$$

Another Primitive Recursive Function:

$$mult(x, y)$$

$$mult(x,0) = 0$$

$$mult(x, y+1) = add(x, mult(x, y))$$

$$mult(x,0) = 0$$

$$mult(x, y+1) = add(x, mult(x, y))$$

$$\mathrm{mul}(3,2) = \mathrm{add}(3, \mathrm{mul}(3,1))$$

Theorem:

The set of primitive recursive functions is countable

Proof:

Each primitive recursive function can be encoded as a string

Enumerate all strings in proper order

Check if a string is a function

Post Systems

- Have Axioms

- Have Productions

Very similar with unrestricted grammars

Example: Unary Addition

Axiom: $$1+1=11$$

Productions:

$$V_1 + V_2 = V_3 \quad \rightarrow \quad V_1 1 + V_2 = V_3 1$$
$$V_1 + V_2 = V_3 \quad \rightarrow \quad V_1 + V_2 1 = V_3 1$$

A production:

$$V_1 + V_2 = V_3 \quad \to \quad V_1 1 + V_2 = V_3 1$$

$$1 + 1 = 11 \;\Rightarrow\; 11 + 1 = 111 \;\Rightarrow\; 11 + 11 = 1111$$

$$V_1 + V_2 = V_3 \quad \to \quad V_1 + V_2 1 = V_3 1$$

Post systems are good for proving mathematical statements from a set of Axioms

Theorem:

A language is recursively enumerable
if and only if
a Post system generates it

Thank you