3. Prove that the problem that a regular language L accepted by a pushdown automata P is decidable.

4. Prove that the problem that a regular string w is generated by a CFG G is decidable.

5. Prove that the problem that the same string w accepted by two DFA, M_1 and M_2, is decidable.

6. Prove that the problem whether a DFA and a regular expression is equivalent is decidable.

7. Prove that the problem whether two regular expressions R_1 and R_2 are equivalent is decidable.

8. Prove that the problem that an arbitrary Turing machine M ever prints a string 101 on its tape is undecidable.

9. Prove that the problem whether a string accepted by an arbitrary Turing machine M is regular is undecidable.

10. Prove that the problem whether an arbitrary Turing machine accepts a single string is undecidable.

11. Find a solution for the following Post correspondence problem.

$$\text{List } A = \{01, 10, 01, 1\}$$
$$\text{List } B = \{011, 0, 11, 0\}$$

1 picture

Recursive Function 11

Introduction
In the ‘Turing machine as integer function’ section of the chapter ‘Extension of the Turing Machine’, different integer functions such as addition, subtraction, multiplication, remainder fi nding, 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.

Like the theory of the Turing machines, the recursive function theory is a way to make formal and precise the intuitive, informal, and imprecise notion of an effective method. Like the theory of the Turing machines, the recursive function theory also obeys the converse of the Church’s thesis. The base of the recursive function is some very elementary functions that are naturally effective. Then, it provides a few methods for constructing more complex functions from those simpler functions. The initial functions and the steps for building complex functions are very few in number, and very simple in character.

11.1 Function

A function is a relation that associates each element of a set with an element of another set. Let a function f be defi ned 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.
The relation given by f between a and b is represented by the ordered pair $< a, b >$ and denoted as $f(a) = b$, and b is called the image of a under f.

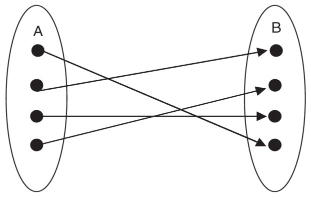
11.1.1 Different Types of Functions

11.1.1.1 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.

Example: $f(x) = x + 4. Let x = 1, 2, 3, ...$ (set of all positive integer numbers). This is an injective function because $f(1) = 5, f(2) = 6, f(3) = 7......$, for no two elements of set A there is exactly the same value in set B.
Diagrammatically, it can be shown as follows.

2 picture



But if there is a function $f(x) = x^2$, where x is the set of real numbers then f(x) is not injective because for $x = 2$ and -2; in both cases, f(x) results in 4 which violates the condition of the injective function.

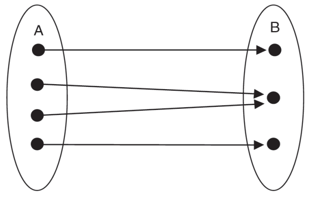
11.1.1.2 Onto or Surjective

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, for every element in B, there is an element in A.

Diagrammatically, it can be shown as follows.

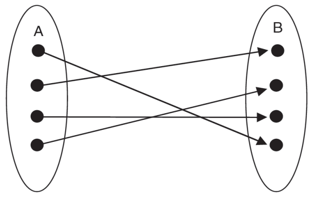
3 picture



11.1.1.3 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.

4 picture



11.1.1.4 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 $g(x) = f^-1(x) = 1/5x$.

11.1.1.5 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: $f(x) = x^2$ and $g(x) = (x+2)$. Then, $fg(x) = f(g(x)) = (x + 2)^2$

11.2 Initial Functions

For a set of natural numbers N, three types of functions can be defined.

1. **Zero function:** A zero function returns 0 for all values. It is denoted as

   $Z(x) = 0$, for all $x \in N$, where N is the set of natural numbers

   The zero function is also called constant function.

2. **Successor function:** A successor function takes one number as argument and returns the succeeding number. It can be de-

noted as

$$S(x) = x + 1, \text{ for all } x \in N, \text{ where N is the set of natural numbers}$$

3. **Projection function:** A projection function is an n-ary projection function denoted by P_i^n , where $n \geq 1$ and $1 \leq i \leq n$, takes n arguments, and returns the ith element.
   It can be denoted as $P_i^n(x_1, x_2, x_3, \ldots..x_n) = x_i$.

11.3 Recursive Function

We are familiar with the term 'recursion' at the time of doing computer programming using some middle level or high level programming languages such as C, C++, Java, etc. A function which calls itself and terminates after a finite number of steps is called a recursive function.

For every recursive, there must be a base condition using which the function terminates.

Example: $fact(x) = x \times fact(x-1)$, where x is a positive integer number and fact(0) = 1. If $x = 4$, then

$$\begin{aligned} fact(4) &= 4 \times fact(3) \\ &= 4(3 \times fact(2)) \\ &= 4 \times (3 \times (2 \times fact(1))) \\ &= 4 \times (3 \times (2 \times (1 \times fact(0)))) \\ &= 4 \times 3 \times 2 \times 1 \times 1 \\ &= 24 \end{aligned}$$