CS53 THEORY OF COMPUTATION

UNIT I AUTOMATA

1. What is deductive proof?
A deductive proof consists of a sequence of statements, which starts from a hypothesis, or a given statement to a conclusion. Each step is satisfying some logical
principle.

2.Give the examples/applications designed as finite state system.
Text editors and lexical analyzers are designed as finite state systems. A lexical analyzer scans the symbols of a program to locate strings corresponding to identifiers,constants etc, and it has to remember limited amount of information.

3.Define: (i) Finite Automaton(FA) (ii)Transition diagram

FA consists of a finite set of states and a set of transitions from state to state that occur on input symbols chosen from an alphabet $\sum$. Finite Automaton is denoted by a 5- tuple(Q, $\sum$, δ ,q0,F), where Q is the finite set of states , $\sum$ is a finite input alphabet, q0 in Q is the initial state, F is the set of final states and δ is the transition mapping function Q * _ to Q.
Transition diagram is a directed graph in which the vertices of the graph correspond to the states of FA. If there is a transition from state q to state p on input a, then there is an arc labeled ‘ a ‘ from q to p in the transition diagram.

4. What are the applications of automata theory?
_ In compiler construction.
_ In switching theory and design of digital circuits.
_ To verify the correctness of a program.
_ Design and analysis of complex software and hardware systems.
_ To design finite state machines such as Moore and mealy machines.

5. Define proof by contrapositive.
It is other form of if then statement. The contra positive of the statement “if H then C” is “if not C then not H”.

6.What are the components of Finite automaton model?
The components of FA model are Input tape, Read control and finite control.

(a)The input tape is divided into number of cells. Each cell can hold one i/p symbol.
(b)The read head reads one symbol at a time and moves ahead.
(c)Finite control acts like a CPU. Depending on the current state and input symbol read from the input tape it changes state.

7.Differentiate NFA and DFA

NFA or Non Deterministic Finite Automaton is the one in which there exists many paths for a specific input from current state to next state. NFA can be used in theory of computation because they are more flexible and easier to use than DFA. Deterministic Finite Automaton is a FA in which there is only one path for a specific input from current state to next state. There is a unique transition on each input symbol.(Write examples with diagrams).

8.What is ∈ -closure of a state q0?

∈ -closure(q0) denotes a set of all vertices p such that there is a path from q0 to p labeled∈ .

-∈ closure(q0)={q0,q1}

9.What is a : (a) String (b) Regular language

A string x is accepted by a Finite Automaton M=(Q, $\sum$, δ .q0,F) if _(q0,x)=p, for some p in F.FA accepts a string x if the sequence of transitions corresponding to the symbols of x leads from the start state to accepting state.
The language accepted by M is L(M) is the set {x | $\sum$ (q0,x) is in F}. A language is regular if it is accepted by some finite automaton.

10.Define Induction principle.

• Basis step:
P(1) is true.
• Assume p(k) is true.
• P(K+1) is shown to be true.

UNIT II REGULAR EXPRESSIONS AND LANGUAGES

1.What is a regular expression?

A regular expression is a string that describes the whole set of strings according to certain syntax rules. These expressions are used by many text editors and utilities to
search bodies of text for certain patterns etc. Definition is: Let $\sum$ be an alphabet. The
regular expression over $\sum$ and the sets they denote are:

i. ϕ is a r.e and denotes empty set.
ii. $\in$ is a r.e and denotes the set {_}
iii. For each 'a' in _ , a+ is a r.e and denotes the set {a}.
iv. If 'r' and 's' are r.e denoting the languages R and S respectively then (r+s),
q0 q1
(rs) and (r*) are r.e that denote the sets RUS, RS and R* respectively.

2. Differentiate L* and L+

_
L* denotes Kleene closure and is given by L* =U Li
i=0
example : 0* ={_ ,0,00,000,…………………………………}
Language includes empty words also.

_
L+ denotes Positive closure and is given by L+= U Li
i=1
example:0+={0,00,000,……………………………………..}

3.What is Arden's Theorem?

Arden's theorem helps in checking the equivalence of two regular expressions.
Let P and Q be the two regular expressions over the input alphabet _. The regular expression R is given as :
R=Q+RP
Which has a unique solution as R=QP*.

4.Write a r.e to denote a language L which accepts all the strings which begin or end with either 00 or 11.

The r.e consists of two parts:
L1=(00+11) (any no of 0's and 1's)
=(00+11)(0+1)*

L2=(any no of 0’s and 1’s)(00+11)
=(0+1)*(00+11)
Hence r.e R=L1+L2
=[(00+11)(0+1)*] + [(0+1)* (00+11)]

5.Construct a r.e for the language which accepts all strings with atleast two c’s over the set _={c,b}
(b+c)* c (b+c)* c (b+c)*
6.Construct a r.e for the language over the set _={a,b} in which total number of a’s are divisible by 3
(b* a b* a b* a b*)*

7.what is: (i) (0+1)* (ii)(01)* (iii)(0+1) (iv)(0+1)+

(0+1)*= { _ , 0 , 1 , 01 , 10 ,001 ,101 ,101001,…………………}
Any combinations of 0’s and 1’s.
(01)*={_ , 01 ,0101 ,010101 ,…………………………………..}
All combinations with the pattern 01.
(0+1)= 0 or 1,No other possibilities.
(0+1)+= {0,1,01,10,1000,0101,…………………………………}

8.Reg exp denoting a language over _ ={1} having
(i)even length of string (ii)odd length of a string

(i) Even length of string R=(11)*
(ii) Odd length of the string R=1(11)*

9.Reg exp for:
(i)All strings over {0,1} with the substring ‘0101’
(ii)All strings beginning with ’11 ‘ and ending with ‘ab’
(iii)Set of all strings over {a,b}with 3 consecutive b’s.
(iv)Set of all strings that end with ‘1’and has no substring ‘00’

(i)(0+1)* 0101(0+1)*
(ii)11(1+a+b)* ab
(iii)(a+b)* bbb (a+b)*
(iv)(1+01)* (10+11)* 1

10. What are the applications of Regular expressions and Finite automata
Lexical analyzers and Text editors are two applications.

Lexical analyzers:The tokens of the programming language can be expressed using regular expressions. The lexical analyzer scans the input program and separates the tokens.For eg identifier can be expressed as a regular expression as: (letter)(letter+digit)* If anything in the source language matches with this reg exp then it is recognized as an identifier.The letter is{A,B,C,………..Z,a,b,c….z} and digit is {0,1,…9}.Thus reg exp identifies token in a language.
Text editors: These are programs used for processing the text. For example UNIX text editors uses the reg exp for substituting the strings such as: S/bbb*/b/ Gives the substitute a single blank for the first string of two or more blanks in a given line. In UNIX text editors any reg exp is converted to an NFA with _ –transitions, this NFA can be then simulated directly.

11.Reg exp for the language that accepts all strings in which ‘a’ appears tripled overthe set $\sum$ ={a}

reg exp=(aaa)*

12.What are the applications of pumping lemma?

Pumping lemma is used to check if a language is regular or not.
(i) Assume that the language(L) is regular.
(ii) Select a constant ‘n’.
(iii) Select a string(z) in L, such that |z|>n.
(iv) Split the word z into u,v and w such that |uv|<=n and |v|>=1.
(v) You achieve a contradiction to pumping lemma that there exists an ‘i’
Such that uviw is not in L.Then L is not a regular language.

13. What is the closure property of regular sets?

The regular sets are closed under union, concatenation and Kleene closure.
r1Ur2= r1 +r2
r1.r2= r1r2
(r)*=r*
The class of regular sets are closed under complementation, substitution, homomorphism and inverse homomorphism.

14.Reg exp for the language such that every string will have atleast one ‘a’ followed by atleast one ‘b’.

$RE=a^+b^+$

15.Write the exp for the language starting with and has no consecutive b's

reg exp=(a+ab)*

UNIT III
CONTEXT FREE GRAMMAR AND LANGUAGES

1. What are the applications of Context free languages?

-
- Context free languages are used in:
- Defining programming languages.
- Formalizing the notion of parsing.
- Translation of programming languages.
- String processing applications.

2. What are the uses of Context free grammars?

- Construction of compilers.
- Simplified the definition of programming languages.
- Describes the arithmetic expressions with arbitrary nesting
- of balanced parenthesis { (,) }.
- Describes block structure in programming languages.
- Model neural nets.
- Regular Expression NFA with $\in$ - moves ,NFA without-$\in$ moves.
- Deterministic Finite Automata.

3.Define a context free grammar

A context free grammar (CFG) is denoted as G=(V,T,P,S) where V and T are finite set of variables and terminals respectively. V and T are disjoint. P is a finite set of productions each is of the form A->_ where A is a variable and _ is a string of symbols from (V U T)*.

3. What is the language generated by the grammar G=(V,T,P,S) where P={S->aSb, S->ab}?

S=> aSb=>aaSbb=>…………………………..=>anbn

Thus the language L(G)={anbn | n>=1}.The language has strings with equal number of a’s and b’s.

4. If S->aSb | aAb , A->bAa , A->ba .Find out the CFL

soln. S->aAb=>abab
S->aSb=>a aAb b =>a a ba b b(sub S->aAb)
S->aSb =>a aSb b =>a a aAb b b=>a a a ba b bb
Thus L={$a^n b^m a^m b^n$, where n,m>=1}

5. What is a ambiguous grammar?

A grammar is said to be ambiguous if it has more than one derivation trees for a sentence or in other words if it has more than one leftmost derivation or more than one rightmost derivation.

6.Consider the grammarP={S->aS | aSbS | _ } is ambiguous by constructing: (a) two leftmost derivation (b) rightmost derivation
Consider a string aab **:**

(a) (i)S=>aS **(ii)** S=>aSbS
=>aaSbS =>aaSbS
=>aabS =>aabS
=>aab =>aab
(b)(i)S=>aS **(ii)** S=>aSbS
=>aaSbS =>aSb
=>aaSb =>aaSbS
=>aab =>aaSb
=>aab

7. What are the three ways to simplify a context free grammar?

- By removing the useless symbols from the set of productions.
- By eliminating the empty productions.
- By eliminating the unit productions.

8. What are the properties of the CFL generated by a CFG?

- Each variable and each terminal of G appears in the derivation of some word in L
- There are no productions of the form A->B where A and B are

variables.

9.Find the language generated by :S->0S1 | 0A | 0 |1B | 1
A->0A | 0 , B->1B | 1

The minimum string is S-> 0 | 1
S->0S1=>001
S->0S1=>011
S->0S1=>00S11=>000S111=>0000A111=>00000111
Thus L={ 0^n 1^m | m not equal to n, and n,m >=1}

10.Construct the grammar for the language L={ an b an | n>=1}.

The grammar has the production P as:
S->aAa
A->aAa | b
The grammar is thus : G=({S,A} ,{a,b} ,P,S)

11. Construct a grammar for the language L which has all the strings which are all palindrome over _={a, b}.

G=({S}, {a,b} , P, S)
P:{ S -> aSa ,
S-> b S b,
S-> a,
S->b,
S->_ } which is in palindrome.

12. Differentiate sentences Vs sentential forms

- A sentence is a string of terminal symbols.
- A sentential form is a string containing a mix of variables and terminal symbols or
- all variables.This is an intermediate form in doing a derivation.

13. What is a formal language?

Language is a set of valid strings from some alphabet. The set may be empty,finite or infinite. L(M) is the language defined by machine M and L(G) is the language defined by Context free grammar. The two notations for specifying formal languages are:
Grammar or regular expression(Generative approach)
Automaton(Recognition approach)

14. Let G= ({S,C} ,{a,b},P,S) where P consists of S->aCa , C->aCa |b. Find L(G).

S-> aCa => aba
S->aCa=> a aCa a=>aabaa
S->aCa=> a aCa a=> a a aCa a a =>aaabaaa
Thus L(G)= { anban ,where n>=1 }

15.What is a parser?

A parser for grammar G is a program that takes as input a string w and produces as output either a parse tree for w ,if w is a sentence of G or an error message indicating that w is not a sentence of G.

16.Is it true that the language accepted by a PDA by empty stack and final states are different languages.

No, because the languages accepted by PDA ‘s by final state are exactly the languages accepted by PDA’s by empty stack.

17.What is the significance of PDA?

Finite Automata is used to model regular expression and cannot be used to represent non regular languages. Thus to model a context free language, a Pushdown Automata is used.

18.When is a string accepted by a PDA?

The input string is accepted by the PDA if:

- The final state is reached .
- The stack is empty.

19.Is NPDA (Nondeterministic PDA) and DPDA (Deterministic PDA)equivalent?

The languages accepted by NPDA and DPDA are not equivalent.
For example: wwR is accepted by NPDA and not by any DPDA.

20. State the equivalence of acceptance by final state and empty stack.

- If L = L(M2) for some PDA M2 , then L = N(M1) for some PDA M1.
- If L = N(M1) for some PDA M1 ,then L = L(M2) for some PDA M2.

where L(M) = language accepted by PDA by reaching a final state.

- N(M) = language accepted by PDA by empty stack.

UNIT IV

PROPERTIES OF CONTEXT FREE LANGUAGES

1. State the equivalence of PDA and CFL.

- If L is a context free language, then there exists a PDA M such that L=N(M).
- If L is N(M) for some PDA m, then L is a context free language.

2. What are the closure properties of CFL?

- CFL are closed under union, concatenation and Kleene closure.
- CFL are closed under substitution , homomorphism.
- CFL are not closed under intersection , complementation.
- Closure properties of CFL's are used to prove that certain languages are not context free.

3. State the pumping lemma for CFLs.

Let L be any CFL. Then there is a constant n, depending only on L, such that if z is in L and |z| >=n, then z=uvwxy such that :
(i) |vx| >=1
(ii) |vwx| <=n and
(iii) for all i>=0 uv^iwx^iy is in L.

4. What is the main application of pumping lemma in CFLs?

The pumping lemma can be used to prove a variety of languages are not context free . Some examples are:
L1 ={ $a^ib^ic^i$ | i>=1} is not a CFL.
L2= { $a^ib^jc^id^j$ | i>=1 and J>=1 } is not a CFL.

5. Give an example of Deterministic CFL.

The language L={a^nb^n : n>=0} is a deterministic CFL.

6. Compare NPDA and DPDA.

NPDA DPDA

1. NPDA is the standard PDA used in automata theory.
   The standard PDA in practical situation is DPDA.
2. Every PDA is NPDA unless otherwise specified.
   The PDA is deterministic in the sense ,that at most one move is possible from any ID.

7. What are the components of PDA ?

The PDA usually consists of four components:

- A control unit.
- A Read Unit.
- An input tape.
- A Memory unit.

8. What is the informal definition of PDA?

A PDA is a computational machine to recognize a Context free language. Computational power of PDA is between Finite automaton and Turing machines. The PDA has a finite control , and the memory is organized as a stack.

9.What is a turing machine?

Turing machine is a simple mathematical model of a computer. TM has unlimited and unrestricted memory and is a much more accurate model of a general purpose computer. The turing machine is a FA with a R/W Head. It has an infinite tape divided into cells ,each cell holding one symbol.

10.What are the special features of TM?

In one move ,TM depending upon the symbol scanned by the tape head and state of the finite control:

- Changes state.
- Prints a symbol on the tape cell scanned, replacing what was written there.
- Moves the R/w head left or right one cell.

11. What are the applications of TM?

TM can be used as:

- Recognizers of languages.
- Computers of functions on non negative integers.
- Generating devices.

12.What is the basic difference between 2-way FA and TM?

Turing machine can change symbols on its tape , whereas the FA cannot change symbols on tape. Also TM has a tape head that moves both left and right side ,whereas the FA doesn’t have such a tape head.

13. What are the various representation of TM?

TM can describe TM using:

- Instantaneous description.
- Transition table.
- Transition diagram.

14. What are the possibilities of a TM when processing an input string?

- TM can accept the string by entering accepting state.
- It can reject the string by entering non-accepting state.
- It can enter an infinite loop so that it never halts.

15. What are the techniques for Turing machine construction?

- Storage in finite control.
- Multiple tracks.
- Checking off symbols.
- Shifting over
- Subroutines.

16. What is the storage in FC?

The finite control(FC) stores a limited amount of information. The state of the Finite control represents the state and the second element represent a symbol scanned.

17. What is a multihead TM?

A k-head TM has some k heads. The heads are numbered 1 through k, and move of the TM depends on the state and on the symbol scanned by each head. In one move, the heads may each move independently left or right or remain stationary.

18.What is a 2-way infinite tape TM?

In 2-way infinite tape TM, the tape is infinite in both directions. The leftmost square is not distinguished. Any computation that can be done by 2-way infinite tape can also be done by standard TM.

19. What is a multi-tape Turing machine?

A multi-tape Turing machine consists of a finite control with k-tape heads and ktapes; each tape is infinite in both directions. On a single move depending on the state of finite control and symbol scanned by each of tape heads ,the machine can change state print a new symbol on each cells scanned by tape head, move each of its tape head independently one cell to the left or right or remain stationary.

20.What is a multidimensional TM?

The device has a finite control , but the tape consists of a k-dimensional array of cells infinite in all 2k directions, for some fixed k. Depending on the state and symbol scanned , the device changes state , prints a new symbol and moves its tapehead in one of the 2k directions, either positively or negatively ,along one of the k-axes

Unit V

1.When we say a problem is decidable? Give an example of undecidable problem?

A problem whose language is recursive is said to be decidable.
Otherwise the problem is said to be undecidable. Decidable problems have an algorithm that takes as input an instance of the problem and determines whether the answer to that instance is “yes” or “no”.
(eg) of undecidable problems are (1)Halting problem of the TM.

2.What are UTMs or Universal Turing machines?

Universal TMs are TMs that can be programmed to solve any problem, that can be solved by any Turing machine. A specific Universal Turing machine U is:
Input to U: The encoding “M “ of a Tm M and encoding “w” of a string w.
Behavior : U halts on input “M” “w” if and only if M halts on input w.

3. What is the crucial assumptions for encoding a TM?

There are no transitions from any of the halt states of any given TM .
Apart from the halt state , a given TM is total.

4. What properties of recursive enumerable seta are not decidable?

- Emptiness
- Finiteness
- Regularity
- Context-freedom.

5.What is a universal language Lu?

The universal language consists of a set of binary strings in the form of pairs (M,w) where M is TM encoded in binary and w is the binary input string.
Lu = { < M,w> | M accepts w }.

6.What is a Diagonalization language Ld?

The diagonalization language consists of all strings w such that the TM M whose code is w doesnot accept when w is given as input.

7. What are the different types of grammars/languages?

• Unrestricted or Phase structure grammar.(Type 0 grammar).(for TMs)
• Context sensitive grammar or context dependent grammar (Type1)(for Linear Bounded Automata)
• Context free grammar (Type 2) (for PDA)
• Regular grammar (Type 3) (for Finite Automata).
This hierarchy is called as Chomsky Hierarchy.

8.State a single tape TM started on blank tape scans any cell four or more times is decidable?

If the TM never scans any cell four or more times , then every crossing sequence is of length at most three. There is a finite number of distinct crossing sequence of length 3 or less. Thus either TM stays within a fixed bounded number of tape cells or some crossing sequence repeats.

9.Does the problem of " Given a TM M ,does M make more than 50 moves on input B "?

Given a TM M means given enough information to trace the processing of a fixed string for a certain fixed number of moves. So the given problem is decidable.

10. Show that AMBIGUITY problem is un-decidable.

Consider the ambiguity problem for CFGs. Use the "yes-no" version of AMB.An algorithm for FIND is used to solve AMB. FIND requires producing a word with two or more parses if one exists and answers "no" otherwise. By the reduction of AMB to FIND we conclude there is no algorithm for FIND and hence no algorithm for AMB.

11.State the halting problem of TMs.

The halting problem for TMs is:
Given any TM M and an input string w, does M halt on w?
This problem is undecidable as there is no algorithm to solve this problem.

.