AUTOMATA & COMPILER DESIGN

Course code: 13CS2110 **L P C**
4 0 3

Course outcomes:

At the end of the course, student will be able to

CO1: Summarize the working of a compiler and various phases involved in compilation and analyze the role of lexical.

CO2: Analyze the role of syntax analysis in a compiler and discriminate different parsing methods like top down and bottom up parsing.

CO3: Infer a semantic analyzer and able to implement semantic rules into a parser that performs attribution while parsing.

CO4: Infer various run time storage requirements for a compiler and code optimization techniques.

CO5: Classify different back end phases of a compiler like intermediate code generation and code generation.

UNIT – I

Formal Language and Regular Expressions: Languages, Definition Languages regular expressions, Finite Automata – DFA, NFA.Conversion of regular expression to NFA, NFA to DFA. Applications of Finite Automata to lexical analysis, lex tools.

UNIT – II

Context Free grammars and parsing : Context free grammars, derivation, parse trees, ambiguity LL(K) grammars and LL(1) parsing Bottom up parsing, handle pruning, LR Grammar Parsing, LALR parsing, parsing ambiguous grammars, YACC programming specification.

UNIT – III

Semantics : Syntax directed translation, S-attributed and L-attributed grammars, Intermediate code – abstract syntax tree, translation of simple statements and control flow statements.

Context Sensitive features – Chomsky hierarchy of languages and recognizers. Type checking, type conversions, equivalence of type expressions, overloading of functions and operations.

UNIT – IV

Symbol table, Storage organization, storage allocation strategies scope access to now local names, parameters, language facilities for dynamics storage allocation. Code optimization Principal sources of optimization, optimization of basic blocks, peephole optimization, flow graphs, optimization techniques.

UNIT – V

Code generation : Machine dependent code generation, object code forms, generic code generation algorithm, Register allocation and assignment. Using DAG representation of Block.

Text Books:

1. John E. Hopcroft, Rajeev M & J D Ullman: "Introduction to Automata Theory Languages &Computation", 3rd Edition, Pearson Education, 2007.
2. Aho, Ullman, Ravisethi: "Compilers Principles, Techniques and Tools", 2nd Edition, Pearson Education, 2009.

References:

1. Tremblay J P, Sorenson G P: "The Theory & Practice of Compiler writing", 1st Edition, BSP publication, 2010.
2. Appel W & Andrew G M: "Modern Compiler Implementation in C", 1st Edition, Cambridge University Press, 2003.
3. Louden: "Compiler Construction, Principles & Practice", 1st Edition, Thomson Press, 2006.
4. Sipser Michael: "Introduction to Theory of computation", 1st Edition, Thomson, 2009.