Introduction to Formal Languages and Automata Theory

Dr. Mousumi Dutt

Lecture 1

Syllabus

Formal Language & Automata Theory
Code: PCC-CS403
Contacts: 3L

Name of the Course:	**Formal Language & Automata Theory**	
Course Code: PCC-CS403	Semester: IV	
Duration: 6 months	Maximum Marks:100	
Teaching Scheme		**Examination Scheme**
Theory: 3 hrs./week		Mid Semester exam: 15
Tutorial: NIL		Assignment and Quiz: 10 marks
		Attendance: 5 marks
Practical: NIL		End Semester Exam: 70 Marks
Credit Points:	3	
Objective:		
1	Be able to construct finite state machines and the equivalent regular expressions.	
2	Be able to prove the equivalence of languages described by finite state machines and regular expressions	
3	Be able to construct pushdown automata and the equivalent context free grammars. And Be able to prove the equivalence of languages described by pushdown automata and context free grammars.	
4	Be able to construct Turing machines and Post machines. Be able to prove the equivalence of languages described by Turing machines and Post machines	

Syllabus

Unit	Content	Hrs/Unit
1	Introduction: Alphabet, languages and grammars, productions and derivation, Chomsky hierarchy of languages.	6
2	Regular languages and finite automata: Regular expressions and languages, deterministic finite automata (DFA) and equivalence with regular expressions, nondeterministic finite automata (NFA) and equivalence with DFA, regular grammars and equivalence with finite automata, properties of regular languages, pumping lemma for regular languages, minimization of finite automata)	7
3	Context-free languages and pushdown automata: Context-free grammars (CFG) and languages (CFL), Chomsky and Greibach normal forms, nondeterministic pushdown automata (PDA) and equivalence with CFG, parse trees, ambiguity in CFG, pumping lemma for context-free languages, deterministic push down automata, closure properties of CFLs.	6
4.	Context-sensitive languages: Context-sensitive grammars (CSG) and languages, linear bounded automata and equivalence with CSG.	6
5	Turing machines: The basic model for Turing machines (TM), Turing recognizable(recursively enumerable) and Turing-decidable (recursive) languages and their closure properties, variants of Turing machines, nondeterministic TMs and equivalence with deterministic TMs, unrestricted grammars and equivalence with Turing machines, TMsas enumerators	6
6	Undecidability: Church-Turing thesis, universal Turing machine, the universal and diagonalization languages, reduction between languages and Rice s theorem, undecidable problems about languages	6

Pre Requisite

- Discrete Mathematics
- Data Structures and Algorithms
- Concepts of Programming Languages

Books

1. John E. Hopcroft, Rajeev Motwani and Jeffrey D. Ullman, Introduction to Automata Theory, Languages, and Computation, Pearson Education Asia.
2. Harry R. Lewis and Christos H. Papadimitriou, Elements of the Theory of Computation, Pearson Education Asia.
3. Dexter C. Kozen, Automata and Computability, Undergraduate Texts in Computer Science, Springer.
4. Michael Sipser, Introduction to the Theory of Computation, PWS Publishing.
5. John Martin, Introduction to Languages and The Theory of Computation, TataMcGraw Hill., PEARSON.
6. Dr. R.B.Patel, Theory of Computation, Khanna Publishing House
7. An Introduction to Formal Languages and Automata, Peter Linz
8. Theory of Computer Science: Automata, Languages and Computation, Mishra & Chandrasekaran

Course Outcome

The students will learn

CO1: To acquire a fundamental understanding of the core concepts in automata theory and formal languages.

CO2: To design finite automata to accept a set of strings of a language.

CO3: To design context free grammars to generate strings of context free language.

CO4: The designing concepts of Push Down Automata

CO5: The basics of Context Sensitive Grammars along with the equivalence relations among with the others.

CO6: The abstract model of Turing Machine and to distinguish between computability and non computability and decidability and undecidability.

Course Outcome & Bloom's Level

CO	Bloom Level
CO 1	Understanding
CO 2	Applying
CO 3	Applying
CO 4	Analyzing
CO 5	Analyzing
CO 6	Creating

Course Outcome & PO Mapping

CO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12
1	3	3	3	2	-	-	-	-	-	-	-	3
2	3	3	3	3	-	-	-	-	-	-	-	3
3	3	3	3	3	-	-	-	-	-	-	-	3
4	3	3	3	3	-	-	-	-	-	-	-	3
5	3	3	3	3	-	-	-	-	-	-	-	3
6	3	3	3	3	-	-	-	-	-	-	-	3
Total	3	3	3	2.83	-	-	-	-	-	-	-	3

Course Outcome & PSO Mapping

CO	PSO1	PSO2	PSO3
1	3	-	-
2	3	2	-
3	3	2	-
4	3	2	-
5	3	3	2
6	3	3	2
Total	3	2.4	2

Planning

- University rules will be followed
- Every week assignment or test will be conducted
- Viva voce will take place also
- Students performance will be based on the performance as stated above along with participation in class
- Planning for Mini Project and/ or Term Paper is also there

What is Automata

- **Automata** theory is important because it allows scientists to understand how machines solve problems.
- An **automaton** is any machine that uses a specific, repeatable process to convert information into different forms.
- Modern computers are a common example of an **automaton**.

What is Automata

- **Automata Theory** is an exciting, theoretical branch of computer science. It established its roots during the 20th Century, as mathematicians began developing - both theoretically and literally - machines which imitated certain features of man, completing calculations more quickly and reliably. The word **automaton** itself, closely related to the word "automation", denotes automatic processes carrying out the production of specific processes. Simply stated, automata theory deals with the logic of computation with respect to simple machines, referred to as **automata**. Through automata, computer scientists are able to understand how machines compute functions and solve problems and more importantly, what it means for a function to be defined as *computable* or for a question to be described as *decidable.*

What is Automata

- **Automatons** are abstract models of machines that perform computations on an input by moving through a series of states or configurations. At each state of the computation, a transition function determines the next configuration on the basis of a finite portion of the present configuration. As a result, once the computation reaches an accepting configuration, it accepts that input. The most general and powerful automata is the **Turing machine**.

What is Automata

- The **major objective** of automata theory is to develop methods by which computer scientists can describe and analyze the dynamic behavior of discrete systems, in which signals are sampled periodically. The behavior of these discrete systems is determined by the way that the system is constructed from storage and combinational elements. Characteristics of such machines include:
- **Inputs:** assumed to be sequences of symbols selected from a finite set *I* of input signals. Namely, set *I* is the set $\{x_1, x_{,2}, x_3 \ldots x_k\}$ where *k* is the number of inputs.
- **Outputs:** sequences of symbols selected from a finite set Z. Namely, set *Z* is the set $\{y_1, y_2, y_3 \ldots y_m\}$ where *m* is the number of outputs.
- **States:** finite set *Q*, whose definition depends on the type of automaton.
- There are **four major families of automaton** :
  - Finite-state machine
  - Pushdown automata
  - Linear-bounded automata
  - Turing machine

The families of automata above can be interpreted in a hierarchal form, where the finite-state machine is the simplest automata and the Turing machine is the most complex.

THANK YOU

More on next class…