iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 11 **Issue:** VI **Month of publication:** June 2023

DOI: https://doi.org/10.22214/ijraset.2023.53969

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 11 Issue VI Jun 2023- Available at www.ijraset.com

Elements of the Theory of Computation

Adarsh Salukhe[1], Ganesh Patil[2], Aniruddha Pawar[3], Gajanan Gangakhedkar[4], Prof. Ajay T. Sonawane[5]
[1, 2, 4]*Undergrad Student, Dept. of Information Technologies SKN Sinhgad Institute of Technology & Science, Lonavala, Maharashtra*
[3]*Undergrad Student, Dept. of Computer Engineering Sinhgad College of Engineering, Vadgaon, Pune, Maharashtra*
[5]*Asst.Professor, Dept. Of Information Tech, SKN Sinhgad Institute of Technology & Science, Lonavala, Maharashtra*

Abstract: ***On the undergraduate level, this book is an introduction to the Theory of Computation. A public purpose of terminology must protect a combination of different classifications. Natural languages, programming languages, mathematical languages, etc. The notation of a natural language is like English, Hindi, etc. Informal language can be defined as a system suitable for expressing specific ideas, facts, or concepts, including symbols and rules to manipulate these. The language we consider for our discussion is an abstraction of natural languages. The theory of Computation is based on Logic, Algorithms, and Theorem-proving. A machine is used to perform the user's essential tasks (requirements). So, it is equally important to know the computing models that compare the study. The model tells us how the machine will work together to do Computation. The theory of Computation abstractly provides computational models.***
Keywords: Finite Automata, Pushdown Automata, Post Machine, Turing Machine, Pumping Lemma, NP Problem

I. INTRODUCTION

The languages we consider for our discussion are an abstraction of natural languages. Our focus here is on formal languages that need precise and standard definitions. Programming languages belong to this category. Design and Compiler- The Theory of Computation (TOC) revolves entirely around the design and construction of a compiler. The compiler is a program that accepts a program written in high-level language as an input and generates a program at a low level which is going to execute by a microprocessor. The compiler is a machine that accepts some information, processes it, and generates some output. Computer Programs are written to solve problems; a problem accepts input and produces desired results. The outcome is caused by doing some processing of information.

A. Steps Involved In

1) Accepts input
2) Recognizes input
3) Process input
4) Generates output

Computer Science is now a much more mature and established discipline, which is essential in a world of ubiquitous computing. In this paper, we are going from Chapter 1 to Chapter 6 from all the various combinations of the sections of parsing and recursive functions. A course emphasizes computability and the foundation of a Computer Science logic mighty under-work through a Chapter 1, 2, 3, 4, 5, and 6. Mainly logic is concentrated in Chapters 4, 5, and 6.

However, it is used with a sincere hope that the computational problem will contribute to the intellectual development of the upcoming generation of Computer Scientists by introducing them to the early stage of the Theory of Computation. The Theory of Computation is exquisite and profound and can be studied with a greater connection to reality. The theory of Computation is intended for a comprehensive presentation of the computational problems or theory. The mathematical representation varies according to the topic. The production is based on mathematical notations. The central/essential focus is on closure properties (*) throughout the Theory of Computation.

II. CONTENT

A. Finite Automata and Language

Introduction / Symbols, Strings and Alphabets / Languages / Finite Automata (FA) / Concept of state transition diagram and transition table / Deterministic Finite Automata (DFA) / Non-Deterministic Finite Automata (NDFA/NFA) / Conversion of NFA to DFA / Finite State Machine (FSM) with output / Moore Machine / Mealy Machine.

B. Regular Expressions and Languages

Introduction / Regular Expressions / Identities of Regular Expression (RE) / Operators of RE / Equivalence of regular expressions and regular languages (RL) / RE to FA utilizing straightforward approach / Modification of FA to RE utilizing Arden's theorem, Pumping lemma for RL / Applications of Regular Expressions.

C. Context Free Grammar and Language

Introduction and representation / Regular Grammar (RG) / Context Free Grammar
(CFG) / Context Free Language (CFL) / Derivation Tree, Parse Tree / Ambiguous
Grammar and Unambiguous Grammar

1) Grammar Simplification: Simplification of Grammar / Unit Production Grammar / Epsilon Production Grammar

2) Normal Forms: Chomsky Normal Form / Greibach Normal Form / Closure properties of CFL / Pumping lemma for CFL.

D. Pushdown Automata and Post Machine

Introduction and Formal definition of Pushdown Automata (PDA) / Transition
Diagram and Table of PDA / Instantaneous Description of PDA / Deterministic PDA / Non-Deterministic PDA / Context Free Language and PDA / Conversion of CFG to PDA and PDA to CFG.
Post Machine (PM): Construction of Post Machine

E. Turing Machine

Introduction / Formal Definition of Turing Machine / Design of Turing machine /
Deterministic Turing machine (TM) / Non-Deterministic Turing machine (TM) /
Multi-Tape TM / Universal Turing Machine / Halting problem of TM / Church Turing thesis / Recursive Language and Recursively Enumerable Languages / Post Correspondence Problem.

F. Computational Complexity

Measuring Complexity / The Class P / The Class NP / Reducibility and Mapping Reducibility / Polynomial Period Squeezing and NP-Completeness / Satisfiability Problem / NP-Completeness of the SAT Problem / Asymptotic Notation.
Standard Forms: Boolean Expressions / Cook's Theorem / Node-C over Problem

III. FINITE AUTOMATA

A. Symbols

Characters are inseparable entities or commodities that cannot be clarified. That is, Logos are the particles of the world of vocabulary. Characters are the particles of the world of terminologies. A character is any single entity such as a, 0, 1, #, initiate, or do. Usually, characters from a standard keyboard are only utilized as characters.

B. Alphabets

An alphabet exists as a finite, non-empty grouping of emblems. The alphabet of the language is normally denoted by ∑. When more additional than one alphabet is supposed for conversation, then subscripts may be used (∑1, ∑2), or sometimes other symbols like G may also be introduced.

∑ = {0, 1}

∑ = {a, b, c} ∑ = {#,▼, ♠, β}

C. Strings or Words over Alphabet

A queue or observation over an alphabet ∑ is a delimited arrangement of attached symbols ∑ It is not the circumstance that a row over some alphabet should incorporate all the consistencies from the alphabet, for example {a, b, c} Prefixes: e, 0, 01, 011
Suffixes: e, 1, 11, 011
Substrings: e, 0, 1, 01, 11, 011

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 11 Issue VI Jun 2023- Available at www.ijraset.com

D. Formal Language

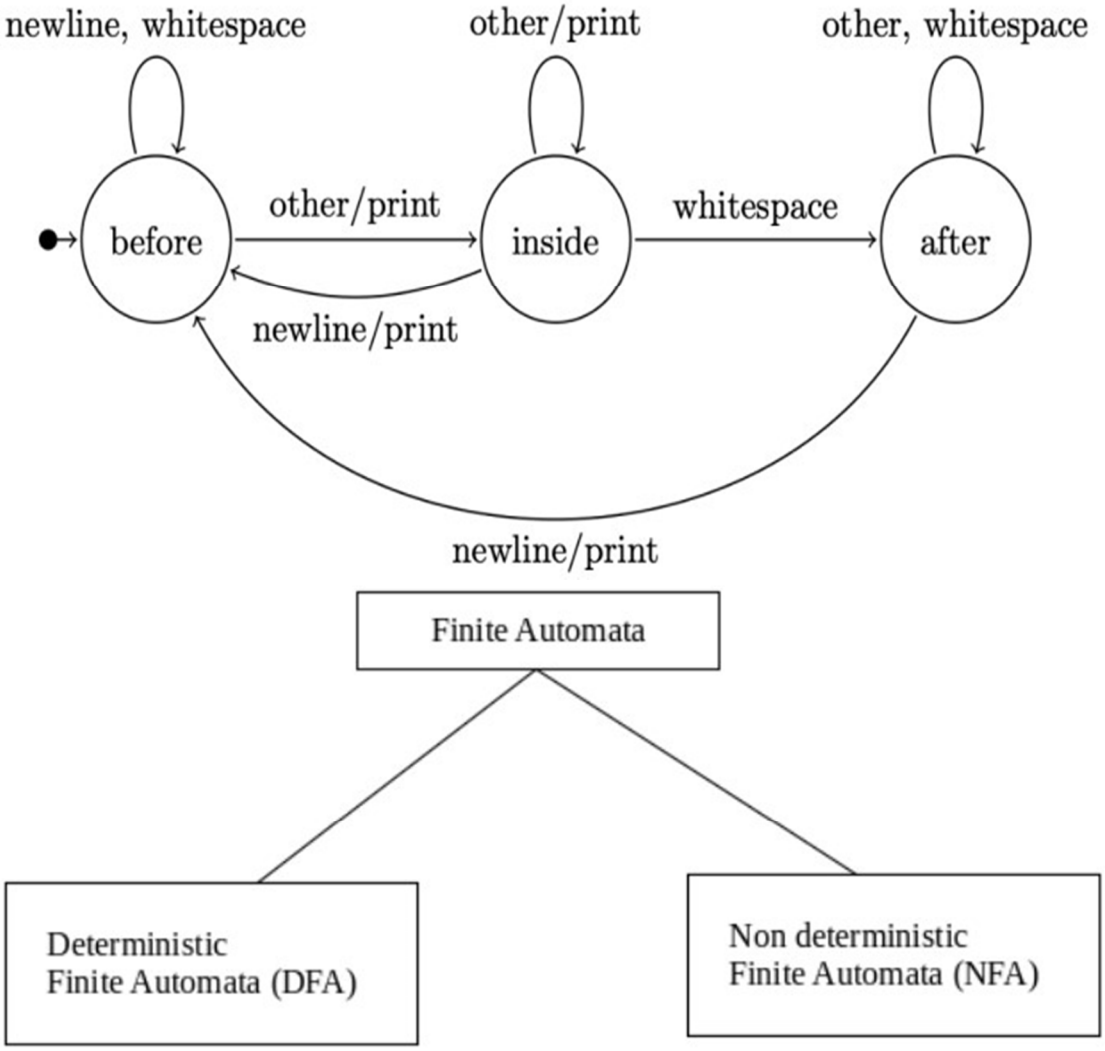
1) A language over an alphabet is a set of strings over that alphabet
2) We can define a language over an alphabet ∑ as a subset of ∑*
3) Therefore, a language L is any subset of ∑□
4) Languages which can be represented by automata machine is the formal language.

IV. THEORY OF COMPUTATION

1) Automata Theory: The robots hypothesis investigates conceptual machines or, more suitably, abstract 'mathematical' appliances or procedures and computational tribulations that can be interpreted employing these apparatuses. These conceptual machines are called robots. Robots come from Greek gossip, signifying something is done by itself.

2) Computability Theory: The computability hypothesis vends on the breadth to which a tribulation is explainable on a computer. The computability hypothesis is nearly associated with the association mathematical reasoning contact recursion hypothesis, which terminates the constraint of inspecting solely representatives of analyses that are reducible to the Turing representative.

3) Complexity Theory: The sophistication hypothesis regards whether a problem can be cracked on a computer. Two consequential characteristics are considered: time sophistication and space complexity, which are, respectively, how multiple steps it takes to perform a computation.



International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 11 Issue VI Jun 2023- Available at www.ijraset.com

V. REGULAR EXPRESSIONS

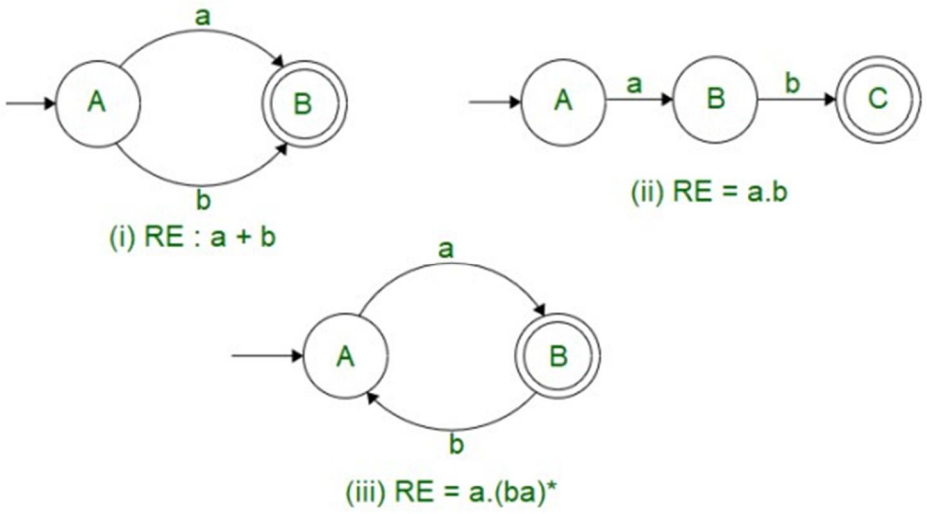
Regular Expression (RE) can be assembled utilizing three operators or a crossbreed of three operators that are cited

1) Union (+)
2) Concatenation (.)
3) Closure (*)

Priority * Higher . to + lower

A. *Arden's Theorem*

Arden's Theorem helps check the equivalence of two regular expressions and the conversion of DFA to the regular expression.



VI. PUMPING LEMMA FOR RL

1) It is a hypothetical rule applicable to non-regular languages. It is a power tool used to prove some sets are not regular.
2) The pumping lemma gives the necessary condition for an input string to belong to a standard group.
3) It provides a method of generating (pumping) many lines from the given series.
4) It states that given any sufficiently long string accepted by FSM, we can find the substring near the beginning of the string.

Arden's Theorem helps check the equivalence of two regular expressions and the conversion of DFA to the regular expression.

VII. CONTEXT FREE GRAMMAR

A. *Elements of Regular Grammar*

Grammar consists of mainly two types of essential elements

1) *Terminal Symbols:* Terminals are those which are part of the generated sentences. Example: "Car," "The," and "Moves" are terminals, collectively generating sentence
2) *Non-Terminal Symbols:* Non-terminals are part of the formation of sentences. An example is non-terminal. The rules of grammar are also called "production," "production rules," or "syntactical rule," which begins with the start symbol.

B. *Context Free Grammar (CFG)*

Mathematically context-free grammar is 4-tuple

G = (V, T, P, S) or G = (V, ∑ ,P, S)

V : finite set of non-terminals

T : finite set of terminals

P : finite set of rules or production in CFG

S : S is a non-terminal which is called as start symbol

The context in CFG is nothing but symbol substitution dependency, i.e., CFG is independent of substitution policy.

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 11 Issue VI Jun 2023- Available at www.ijraset.com

C. Context Free Language (CFL)

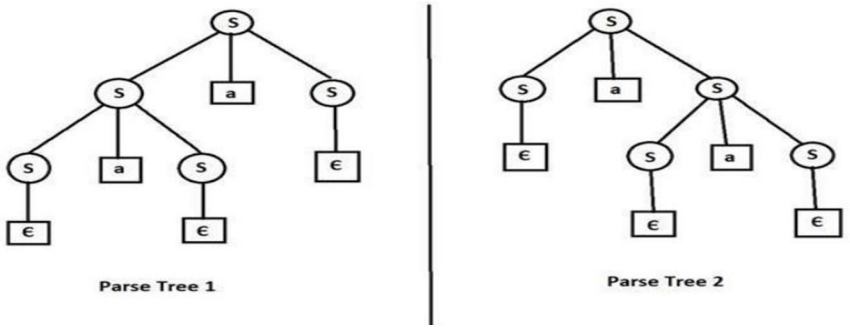
1) If a CFG is given, we try to specify the terminology to which it has corresponded. CFL is a language yielded from CFG.

2) The context-free terminologies are latched under some distinct process; closed means after doing that process on a context-free terminology, the consequential terminology will also be a context-free language

D. Ambiguous Grammar

1) As we know, grammar can be used to structure programs and documents. The assumption was that grammar uniquely determines a structure for each string in its language

2) However, not every grammar does provide a unique structure. That is, some grammar is used to give a unique structure. In some other grammar, the language puts more than one structure for some strings in the language.



E. Simplification of CFG

1) Removal of Useless Production

a) Here, we are going to identify those symbols which do not play any role in the derivation of any string and then eliminate the identified production.

b) A variable (production) is useless if it does not appear in the derivation of the string

2) Removal of Unit Production

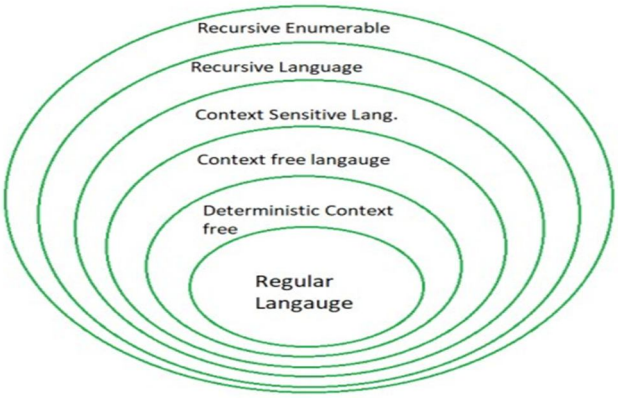
a) A production of the form Non-terminal → One non-terminal production where A→B (Where A and B both are nonterminal)

3) Removal of Ɛ-Production or Nullable Production

a) The production of the form A→ Ɛ is called Ɛ (epsilon) production and non-terminal. A is called a Nullable non-terminal.

b) Surely, if Ɛ is in L(G), then we can not eliminate all Ɛ production from G, but if Ɛ is not in L(G), we can eliminate all the Ɛ production from G.

F. Chomsky Normal Form



The Venn (diagram) representation of Chomsky Hierarchy

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 11 Issue VI Jun 2023- Available at www.ijraset.com

Chomsky Hierarchy	
Type 0	**Unrestricted Grammar (URG)**
Type 1	**Context Sensitive Grammar (CSG)**
Type 2	**Context Free Grammar (CFG)**
Type 3	**Regular Grammar (RE)**
LLG	**Left Linear Grammar**
RLG	**Right Linear Grammar**

Chomsky Hierarchy Layered representation

G. Pumping Lemma For CFL

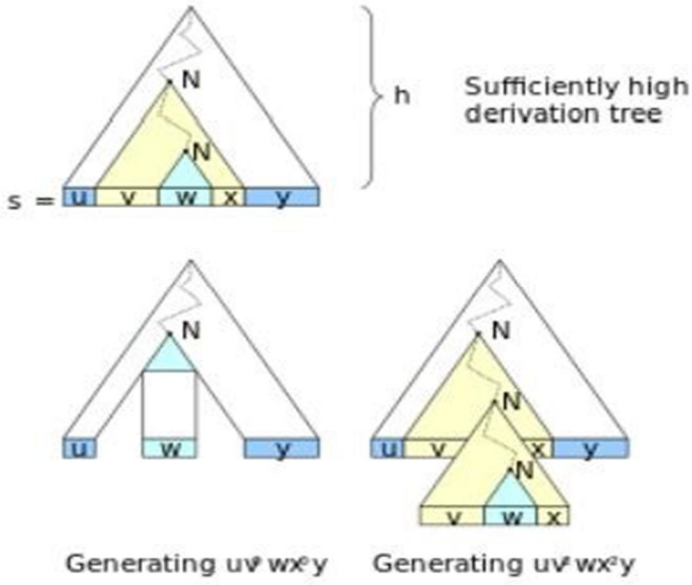
The pumping lemma gives us a technique to show that specific languages are not context-free. However, the pumping lemma for CFL is more interesting than the pumping lemma for regular language

Definition: The pumping lemma for CFLs states that for the sufficiently long string in CFL, we can find two short, nearby substrings that we can "Pump" in tandem and the resulting series must also be in the language.

Example:

Let L be a CFL. Then there exists a constant p such that if z is any string in L, Where $|z| \geq p$, then we can write z = uvwxyz subject to the condition

1) $|vwx| \leq p$, The middle portion is not more significant than p
2) $vx \neq \mathcal{E}$, We will pump v and x. In this case, One may be empty, but both cannot.
3) For all $I \geq 0$, uvi wxi y is also L. Thus we were going to pump both v and x



International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 11 Issue VI Jun 2023- Available at www.ijraset.com

VIII. PUSHDOWN AUTOMATA

A. *Pushdown Automata*

The pushdown automata are finite automata with control of both an input tape and auxiliary storage (stack) to what it has read. A stack is a "LIFO" [LAST IN FIRST OUT] structure where symbols can be inserted or removed from only one end, i.e., the top. Pushdown automata can be deterministic or non-deterministic. Hence PDA is DPDA [Deterministic Pushdown Automata] or NPDA [Non-Deterministic Pushdown Automata]. The difference between DPDA and NPDA is Ɛ [epsilon] transition. Ɛ [epsilon] transitions are allowed in case of Non-Deterministic PDA and work based on "GUESS." Hence NPDA, at all times, can have more than one more.

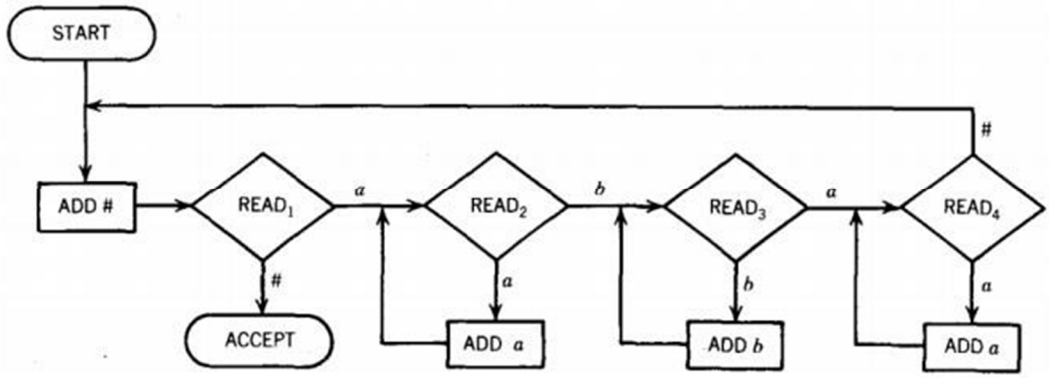
B. *Elements of PDA*

An input alphabet (∑)

1) An input tape
2) An alphabet of stack symbol (⌠)
3) A pushdown stack
4) Start state
5) Halt states: Accept and Reject
6) Non branching state: Push
7) Branching state: Read, Pop

C. *Post Machine*

The post machine was designed as a "Universal algorithm machine." It was invented by the scientist email Leon Post in 1936. By universal algorithm machine, we mean the device should accept any language that can be precisely defined by a human being



IX. TURING MACHINE

A. *Formal Definition of Turing Machine*

Mathematically Turing Machine is defined as

M = (Q, ∑, δ, q0, F, T, B/b)

Q : Non-empty finite set of states ∑: Non-empty finite set of inputs q0: Initial state of TM, ε Q F: Set of final states, ⊂ Q

⌠: Set of tape symbols { ∑, b } ⊂ ⌠

B/b: Blank symbol (Bounding symbol)

1) *Recursive Functions:* A function is recursive if it can be obtained from the initial operations by a finite number of applications of composition, recursion, and minimization over regular operations.
2) *Partial Recursive Functions:* A function is partially recursive if it can be obtained from the initial position by a finite number of compositions, recursion and minimization.

B. Variants of Turing Machine (TM)

Turing machines [TM] can perform fairly robust computation. To better understand their surprising power, we shall consider the effect of extending the Turing machine in various directions. We should subsequently use the additional feature when designing the Turing Machine to solve a particular problem.

Types of Turing Machine [TM]

1) Composite TM
2) Multi-tape TM
3) Iterative TM
4) Multi-head TM
5) Non-Deterministic TM
6) Multidimensional TM
7) Universal TM

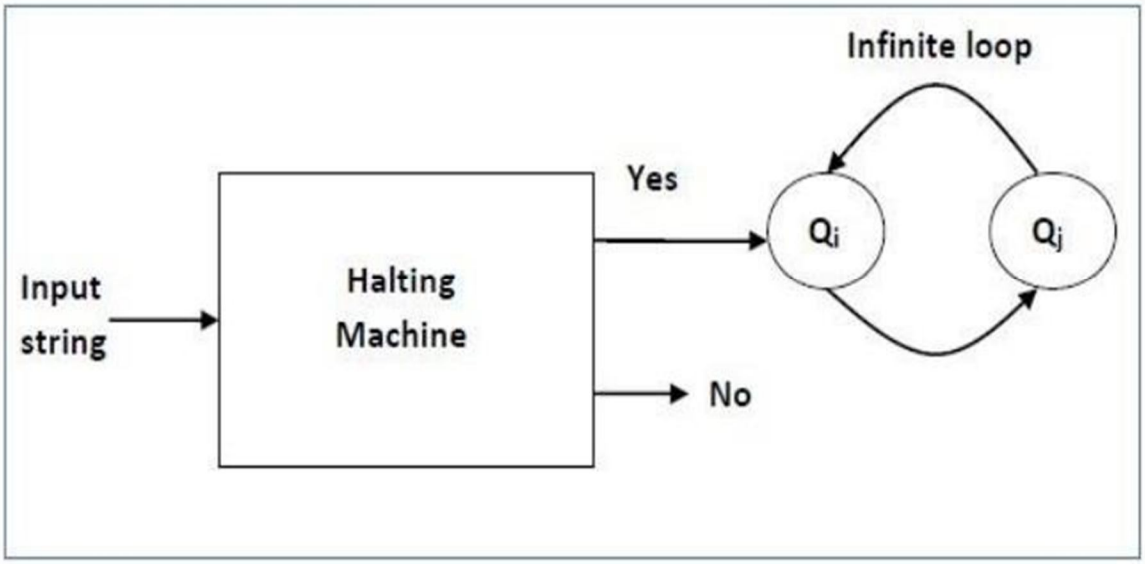
C. Church-Turing Thesis

1) The Church-Turing thesis states that any algorithmic procedure that can be carried out by human procedure that can be carried out by human beings/computer can be carried out by a Turing machine.
2) It has been universally accepted by computer scientists that the Turing machine provides an ideal theoretical model of a computer.
3) The Church-Turing thesis says that real-world computation can be translated into an equivalent computerization involving a Turing machine. In Church's original formulation (Church 1935, 1936), the thesis says that real-world calculation can be done using lambda calculus, using a general recursive function.
4) The Church-Turing thesis encompasses more kinds of computations than those intended initially, such as cellular automata, etc.

D. Turing Machine Halting problem

It is essential to determine whether the process of the Turing Machine will halt; this is known as the "Halting Problem of the Turing Machine."

1) Empty word problem of the Turing Machine is the famous halting problem that is unsolvable.
2) No algorithm exists that takes on any arbitrary Turing Machine and input alphabet and determines whether or not the machine will halt for every input.
3) Uniform halting problem of the Turing Machine is unsolvable, and the un-solvability of the problem implies the unsalability of many mathematical problems and problems from computer science.

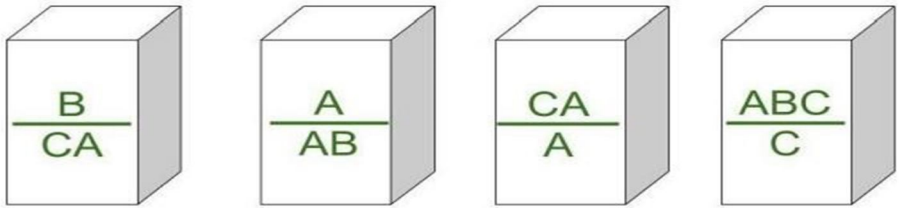


Halting Problem

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 11 Issue VI Jun 2023- Available at www.ijraset.com

E. Post Correspondence Problem

Email Post Later introduced the Post Correspondence Problem (PCP), which was found to have many applications in the theory of formal languages. It is possible to reduce the PCP to many classes of 2 outputs.



X. COMPUTATIONAL COMPLEXITY

Decidable problem concerning regular Language

A problem whose language is recursive then it is decidable; else, it is undecidable

1) If the problem is undecidable, then no algorithm takes input and finds output or answer.

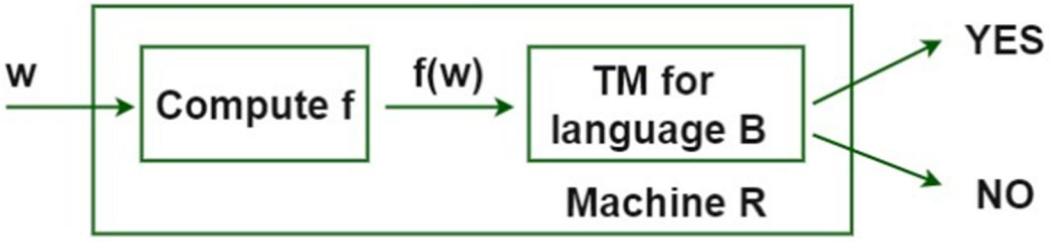
2) A problem is called decidable if it has a corresponding. The turning device terminates on every intake with an explanation. It is also necessary to find the Turing machine, which is termed decidable, cause the Turing machine takes a halt on every single input for accepting or rejecting.

A. Mapping Reducibility

Deduction means communicating that one tribulation is “more comfortable” than another.

1) A is more effortless than problem B. This also means that if we can solve A using the algorithm, that also solves B.

The entire idea behind reducibility is to transform the input of A into inputs of B



2) However, there are some implication problems of reductions.

3) We transform the input of A to the input of B. This is called an implication of reduction

B. Cook’s Theorem

Theorem:

SAT is NP-Complete

1) If the encoded expression E is of Length n, then the number of variables is n/2. Hence guessing a truth assignment, we can use a multi-tape Turing Machine for E.

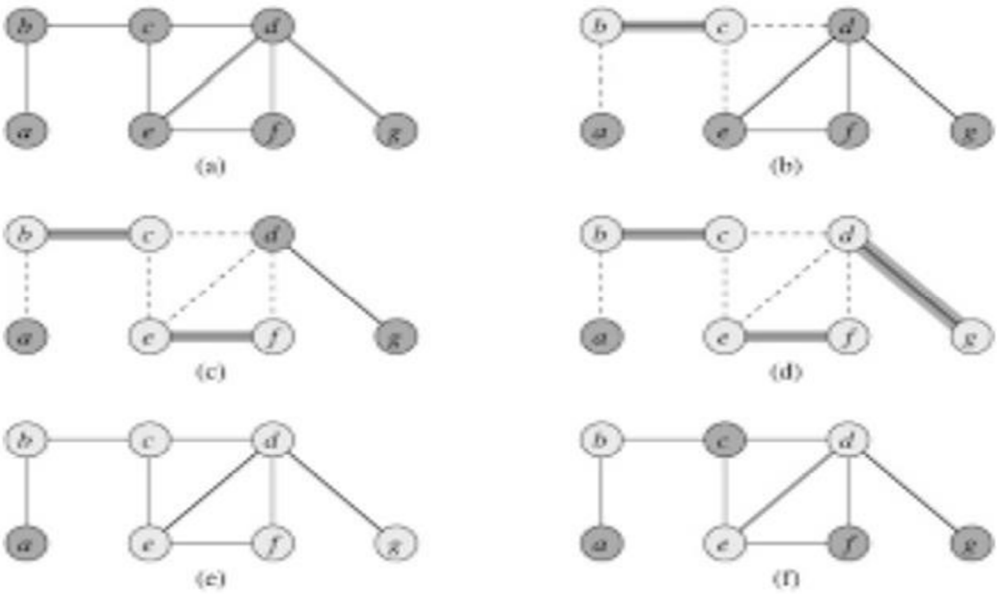
2) The time taken in multi-tape NTM M is O(n). Then M considers the significance of E for a fact undertaking t. This is accomplished in the O(n 2) period.

3) The equivalent single-tape Turing Machine acquire O(n 4) time. Once an accepting truth assignment is found, M accepts E, and M halts.

4) Thus, we have located a polynomial-time NTM for SAT. • Hence SAT ε NP.

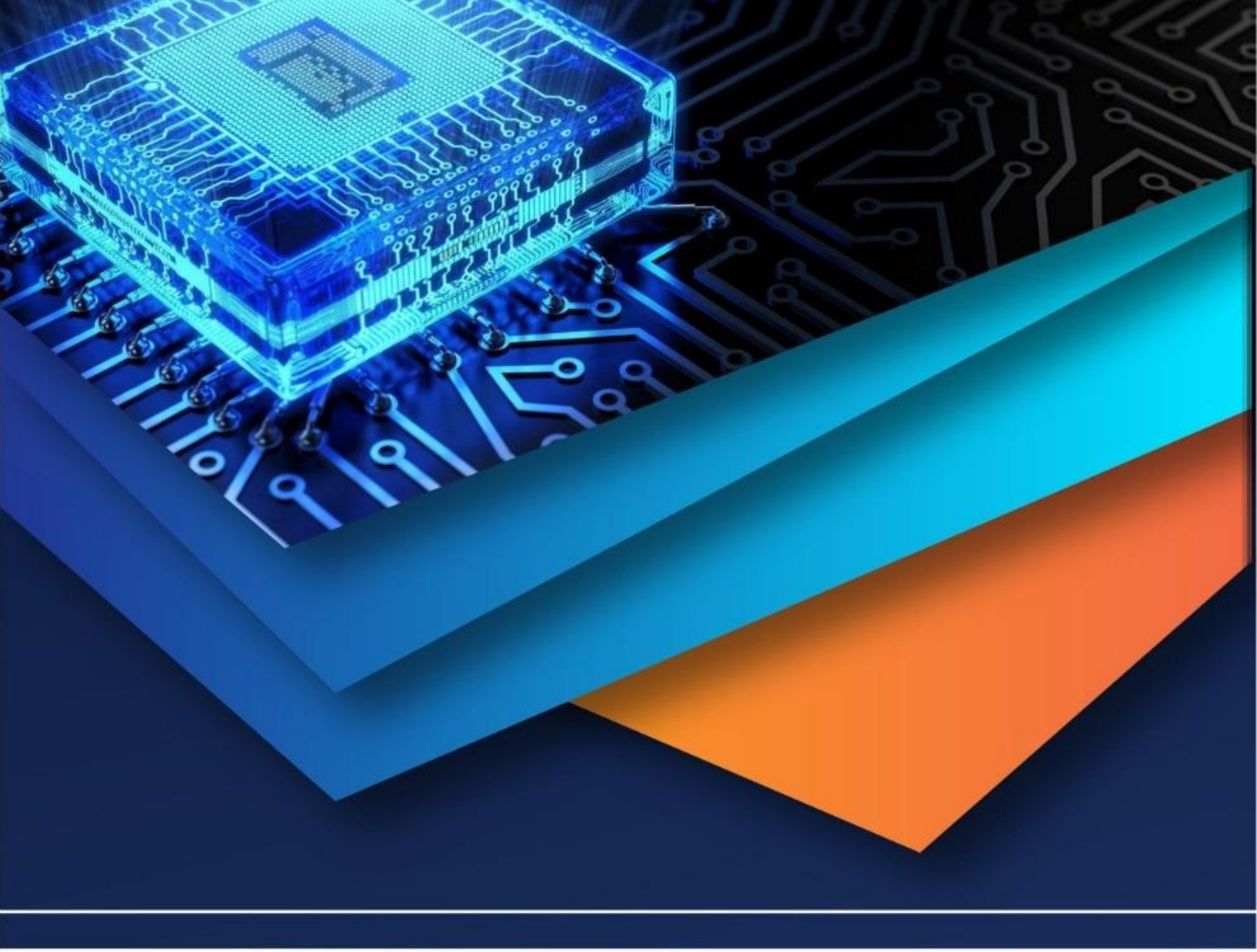
C. Node-Cover Problem

A vertex (node) exterior of a unidirectional diagram is a subset of its vertices such that for every edge (u, v) of the chart, either ‘u’ or ‘v’ is in the vertex cover. Although the representation is Vertex Exterior, the collection encircles all boundaries of the diagram.



REFERENCES

[1] Yan, Song Y. Co. Pte. Ltd. pp. 155–156, 1998.
[2] Chakraborty, P., Saxena, P. C., Katti, C. P. 2011. Fifty Years of Automata Simulation: A Review. ACM Inroads, 2(4):59–70. http://dl.acm.org/citation.cfm?id=2038893&dl=ACM&coll=DL&CFID=65021406&CFTOKEN=86634854 \
[3] Jirí Adámek and Vera Trnková. 1990. "Automata and Algebras in Categories." Kluwer Academic Publishers:Dordrecht and Prague
[4] S. Mac Lane, Categories for the Working Mathematician, Springer, New York 1971.
[5] Meseguer, J., Montanari, U.: 1990 Petri nets are monoids. Information and Computation 88:105–155 X. S. B. Cooper, 2004. Computability Theory, Chapman & Hall/CRC. ISBN 1-58488-237-9
[6] N. Cutland, 1980. Computability, An intro to recursive process hypothesis, Cambridge University Press. ISBN 0-52129465-7
[7] Matiyasevich, 1993. Hilbert's Tenth Problem, MIT Press. ISBN 0-262-13295-8



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY