Syllabus

Unit-I: Introduction to Artificial Intelligence.

- → Introduction
- → History
- → Intelligent Systems
- → Foundations of AI
- → Applications
- → Tic-Tac-Toe game playing.
- → Development of AI languages
- → Current Trends in AI

Problem Solving – State-space search and Control Strategies.

- → Introduction
- → General problem Solving
- → characteristics of problem.

Unit-II: Search Strategies.

- → Exhaustive Search
- → Heuristic Search Techniques
- → Iterative Deepening a*
- → Constraint Satisfaction
- → Two-player Games.

Unit-III: Logic Concepts

- → Introduction
- → Propositional Calculus
- → Propositional logic
- → Natural Deduction System.

→ Axiomatic System.
→ Semantic Tableau System in Propositional logic
→ Resolution Refutation in Propositional logic
→ Predicate Logic

Part-III Knowledge Representation.

→ Introduction
→ Approaches to knowledge Representation.
→ Knowledge Representation using Semantic network.
→ Extended Semantic networks for knowledge Representation.
→ knowledge Representation using Frames.

Unit-IV: Advanced knowledge Representation Techniques

I

→ Introduction
→ Conceptual Deperdency Theory
→ Script Structure.

Unit IV-II Excpert System and Applications.

→ Introduction. phases in building expert system.
→ Excpert System Vs Traditional Systems.
→ Rule based excpert Systems
→ Black Board Systems.
→ Truth Maintenance Systems
→ List of shells and tools.

Unit-V Uncertainty Measure - Probability Theory

Introduction

Probability Theory

Bayesian Belief networks

Certainty factor Theory

Dempster - Shafer Theory.

Unit-I Introduction to Artificial Intelligence

Introduction:-

* AI Foundation was laid with the development of BOOLEAN THEORY.
* Boolean theory operates on the principle that all statements are either "True" or "False".
* Building Blocks of Computing: Every computing system, at its core, operates on principles of Boolean logic. This is because computer h/w is built on binary states(0 and 1).
* Logic Programming in AI: Boolean logic forms the foundation for many AI algorithms, particularly in the field of logic programming. Programs can make decisi-ons based on logical rules, which are often express-ed in terms of Boolean operations.

For example, an AI might have a rule like "IF condition1 AND condition2 are True, THEN perform action1". Here, "condition1 AND condition2" is a Boolean expression.

So, in essence, Boolean logic is fundamental to both the hardware that AI systems run on; and the software algorithms that drive their decision-making processes.

* The goal of AI has always been to make machines smart, like trying to get a robot to think (or) make decisions like a human.
* There are two main ways people try to create AI:-

Bottom-up approach: It's like building a mini-brain with electronic parts. Imagine creating a toy brain and hoping it will think like us.

Top-down approach:-

Here, we don't built anything physically. Instead, we write computer programs to mimic our thinking. It's like creating a virtual brain inside the computer.

Intelligent Systems:-

Challenges of AI:

→ Difficult to understand.
→ Constantly changing
→ AI deals with problem which AI don't understand
→ AI makes mistakes (like Human Being).
→ Don't give perfect answers every time [try to give a good solution].

Two views on AI goals:-

1. To copy the human brain behaviour [Acting]
2. Doing things logically [Thinking].

~~Testing Techniques~~

Turing Test:-

Simple Test introduced by Alen Turing. It will test whether the system is behaving like a human (or) not is called.

ELIZA

A computer program which has a text based conversation with users.

ELIZA Communication

ELIZA	USER
Hello Iam ELIZA	Can you help me?
Don't you believe that I can help you?	Yes

I See	I feel depressed.
How long have you been feeling depressed?	since last month
This is quite interesting.	what is interesting in this?
Have you asked anyone else?	No
Are you saying no just to be negative?	No I did not ask anyone.
why no?	I do not believe other people.
Say, do you have any Psychological problems?	I do not know
Tell me more.	

Components of AI program

1) knowledge Base — A Memory stores like: rules, facts

characteristics:-
1. voluminous
2. Incompletes imprecise
3. Dynamic.

2, Control Strategy — Decision Making (Heuristics/Thumrules)

3, Inference Mechanism — thought processing (inference rules).

Foundations of AI:-

1. Mathematics
2. NeuroScience — study brain & functions.
3. Control Theory — change their behaviour based on situations. → decision making.
4. Linguistics — The study of language & how to communicate.

Applications:-

1. Business
2. Engineering
3. Education
4. Medicine
5. Manufacturing
6. Fraud Detection
7. Object Identification
8. Space Shuttle Scheduling
9. Information Retriever.

Unit - III

Logic Concepts

Symbolic logic

- propositional calculus
- Predicate calculus

Well formed formula:-

A well-formed formula is defined as a symbol (or) string of symbols generated by the formal grammar of a formal language.

1. Smallest unit is considered to be well-formed formula (WFF). Ex:- P, Q, R, - - -
2. If α is a well-formed formula, then $\neg\alpha$ is also WFF.
3. If α and β are well-formed formulas then $(\alpha \vee \beta)$, $(\alpha \wedge \beta)$, $(\alpha \xrightarrow{\text{implies}} \beta)$, $(\alpha \xleftrightarrow{\text{equivalence}} \beta)$.
4. A propositional expression is called a WFF, if and only if it satisfies the above properties.

Equivalent Formulas:-

Two formulas α and β are said to logically equivalent $(\alpha \cong \beta)$. if and only if truth values are same for all possible assignments of logical constants (T/F).

Equivalence Laws:-

Absorption law:- $A \vee (A \wedge B) \cong A$ | $A \vee (\neg A \wedge B) \cong A \vee B$

$A \wedge (A \vee B) \cong A$ | $A \wedge (\neg A \vee B) \cong A \wedge B$

Excluded Middle law:- $A \vee \neg A \cong T$ (True)

Contradiction law:- $A \wedge \neg A \cong F$ (False)

Commonly used equivalence relations:-

1. $A \vee F \cong A$
2. $A \vee T \cong T$
3. $A \wedge T \cong A$
4. $A \wedge F \cong F$
5. $A \to B \cong \neg A \vee B$
6. $(A \leftrightarrow B) \cong (A \to B) \wedge (B \to A) \cong (A \wedge B) \vee (\neg A \wedge \neg B)$
7.

* Truth values on each row ars called an "Interpretation" for formula.

Valid = Tautology.

Stmnt formula/group of Stmt Formula → Satisfiable/ unsatisfiable.

* A formula 'α' is said to be valid, if it is a tautology.
* A formula 'α' is said to be satisfiable, if their exist at least one interpretation for which 'α' is true.
* A formula 'α' is said to be unsatisfiable, if the value of 'α' is false under all interpretations.

1) Show that "If it is humid then it will rain and since it is humid today it will rain" is a valid argument.

Sol:- Let, A: It is humid and B: It will rain.

$\alpha: [[A \to B] \wedge A] \to B$

A	B	$A \to B$	$(A \to B) \wedge A$	$\alpha: [(A \to B) \wedge A] \to B$
T	T	T	T	T
T	F	F	F	T
F	T	T	F	T
F	F	T	F	T

} Tautology

Methods to check the condition is valid (or) not.

1. Natural Deduction System.
2. Axiomatic System
3. Semantic tableau Method
4. Resolution repitation method.

① → Importance of AS with Examp

★ imp

② Axiomatic System:-

3 axioms and 1 deduction rule

$\neg$ (~), $\rightarrow$ (condition). These two connectives are used.

Axiom:1 $\alpha \rightarrow (\beta \rightarrow \alpha)$ where α & β are WFF } Tautology.

Axiom:2 $[\alpha \rightarrow (\beta \rightarrow \gamma)] \rightarrow [(\alpha \rightarrow \beta) \rightarrow (\alpha \rightarrow \gamma)]$ } Tautology.

Axiom:3 $(\neg\alpha \rightarrow \neg\beta) \rightarrow (\beta \rightarrow \alpha)$

Axiom → means assumed to be true but their is no proof.

1 Deduction rule : Moden's Ponens i.e., $P, P \rightarrow Q \Leftrightarrow Q$.

(antecedents)

In this,

$\alpha \rightarrow \beta$ and α ∴ Hypothesis

β ∴ Conclusion | consequent

*Let $\Sigma = \{\alpha_1, \alpha_2, \ldots, \alpha_n\}$ be a set of hypothesis then,

α is said to be a deductive Consequence of Σ, if either α is an axiom (or) a hypothesis (or) is derived from α_j where $1 \leq j \leq n$ using Moden's Ponen Rule.

* It is Represented as $\{\alpha_1, \alpha_2, \ldots, \alpha_n\} \vdash \alpha$ (or)

$\Sigma \vdash \alpha$ ← infers.

* If α is empty then the above representation becomes $\vdash \alpha$, which means α is derived only from axioms and no hypothesis are involved and it is said to be theorem.

Example:- *

① Validate the inference $\{A \to B, B \to C\} \vdash (A \to C)$ using axiomatic System.

Solution:-

Description (How the formula obtained.)	Formula	Comments. (Line no / step no.)
Theorem	$\{A \to B, B \to C\} \vdash (A \to C)$	To be proved
Hypothesis1	$A \to B$	1
Hypothesis2	$B \to C$	2
Instance of axiom1 ↓ $\alpha \to (\beta \to \alpha)$ Here $\alpha = (B \to C)$, $\beta = A$	$(B \to C) \to [(A \to (B \to C)]$	3
Modens ponens Rule (2,3) ↓ $\alpha, \alpha \to \beta = \beta$	$A \to (B \to C)$	4
Instance of axiom 2. ⇓ $(\alpha \to (\beta \to \gamma)) \to [(\alpha \to \beta) \to (\alpha \to \gamma)]$	$[A \to (B \to C)] \to [(A \to B) \to (A \to C)]$	5

Modens Ponens Rule (4,5)	$[(A \to B) \to (A \to C)]$	6
Moden's ponens Rule (1,6)	$A \to C$	Proved

★ ② Validate the inference $\{\neg A\}$ infers $A \to B$.

Sol:-

Description	Formula	Comments
Theorem –	$\{\sim A\} \vdash (A \to B)$	– To be Proved
Hypothesis –	$\sim A$	– 1
Instance of axiom 1 ⇓ $\alpha \to (\beta \to \alpha)$ Here $\alpha = \sim A$, Let $\beta = \sim B$	$\sim A \to (\sim B \to \sim A)$	– 2
Modens ponens Rule (1)(2)	$\sim B \to \sim A$	– 3
Instance of axiom 3 – $(\alpha \to (\beta \to \gamma)) \to [(\alpha \to \beta) \to (\alpha \to \gamma)]$	$(\sim B \to \sim A) \to (A \to B)$	– 4
Moden's ponens Rule	$A \to B$	– Proved

Deduction Theorem

* Given that Σ is a set of hypothesis and α & β are well formed formulas.
* If β is proved from $\Sigma \cup \{\alpha\}$ then according to deduction theorem. $(\alpha \to \beta)$ is proved from 'Σ'.
* we can write $\Sigma \cup \{\alpha\} \vdash \beta$ implies $\Sigma \vdash (\alpha \to \beta)$.

Converse of Deduction Theorem.

* Given $\Sigma \vdash (\alpha \to \beta)$ then $\Sigma \cup \{\alpha\} \vdash \beta$.

3) Prove the theorem $\vdash$ infer $(\sim A) \to (A \to B)$ $\Sigma = \phi$

Sol:- Given, $\vdash (\sim A) \to (A \to B)$

By applying deduction theorem,

if $\Sigma \cup \{\alpha\} \vdash \beta$ then $\Sigma \vdash (\alpha \to \beta)$

Here, $\Sigma = \phi$, $\alpha = \sim A$, $\beta : (A \to B)$

Then

$$\phi \cup \{(\sim A)\} \vdash (A \to B)$$

$$\{(\sim A)\} \vdash (A \to B)$$

already proved in (2).

Hence By DT, $\vdash (\sim A) \to (A \to B)$ is proved.

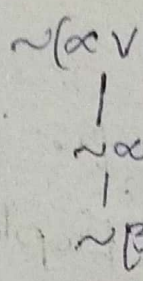
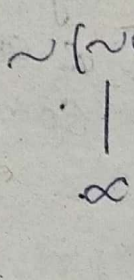
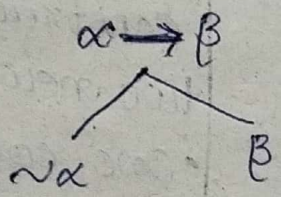
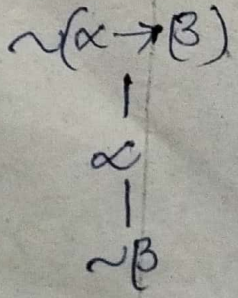
Semantic Tableau Method:-

(meaning) (Graphical Representation)

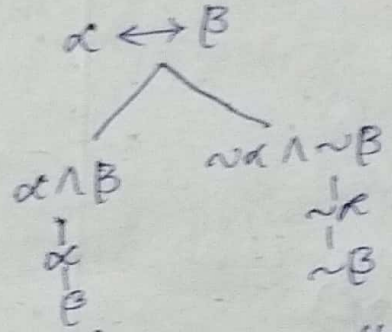
* It is a binary tree which is constructed by using Semantic Tableau Rules with a formula as a root.

Semantic Tableau Rules for α & β:-

Rule No.	Tableau Tree (It is a BT.)	Explanation
Rule1	$\alpha \wedge \beta$ is true if both α and β are true. Binary Tree:- $\alpha \wedge \beta$ / α / β (or) $\alpha \wedge \beta$ / β / α	A tableau for the formula $(\alpha \wedge \beta)$ is constructed by adding both α and β to the same path (branch)
Rule2	$\sim(\alpha \wedge \beta)$ is true if either $\sim\alpha$ is true (or) $\sim\beta$ is true. $\sim(\alpha \wedge \beta) \Leftrightarrow \sim\alpha \vee \sim\beta$. BT:- $\sim(\alpha \wedge \beta)$ / $\sim\alpha$ $\sim\beta$	A tableau for a formula $\sim(\alpha \wedge \beta)$ is constructed by adding two new paths:- • One containing $\sim\alpha$ and other containing $\sim\beta$.
Rule3	$\alpha \vee \beta$ is true if either α or β is true. $\alpha \vee \beta$ / α β	

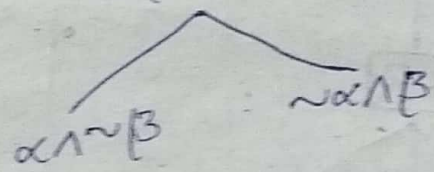
Rule 4	$\sim(\alpha \vee \beta)$ is true if both $\sim\alpha$ and $\sim\beta$ are true. $\sim(\alpha \vee \beta) \Leftrightarrow \sim\alpha \wedge \sim\beta$ 	
Rule 5	$\sim(\sim\alpha)$ is true then α is true. 	A tableau for $\sim(\sim\alpha)$ is constructed by adding α on the same path;
Rule 6	$\alpha \rightarrow \beta$ is true $\sim\alpha \vee \beta$ is true. 	
Rule 7	$\sim(\alpha \rightarrow \beta)$ is true then $\alpha \wedge \sim\beta$ is true 	

Rule 8 | $\alpha \leftrightarrow \beta$ is true then either $(\alpha \wedge \beta) \vee (\sim\alpha \wedge \sim\beta)$ is true.



$\alpha \leftrightarrow \beta$
$= (\alpha \rightarrow \beta) \wedge (\beta \rightarrow \alpha)$
(or)
$\checkmark (\alpha \wedge \beta) \vee (\sim\alpha \wedge \sim\beta)$

Rule 9 | $\sim(\alpha \leftrightarrow \beta)$ is true then $(\alpha \wedge \sim\beta) \vee (\sim\alpha \wedge \beta)$ is true.



$\sim(\alpha \leftrightarrow \beta) \Leftrightarrow$
$\sim[(\alpha \rightarrow \beta) \wedge (\beta \rightarrow \alpha)]$
$\Leftrightarrow \sim[(\sim\alpha \vee \beta) \wedge (\sim\beta \vee \alpha)]$
$\Leftrightarrow (\alpha \wedge \sim\beta) \vee (\sim\alpha \wedge \beta)$

*Q4
1) Explain the Construction of Semantic tableau for the formula $(A \wedge \sim B) \wedge (\sim B \rightarrow C)$

Sol:- The thumb rule to construct a semantic tableau is to apply non-branching rules before branching rules.

Description	Formula	line number
Tableau root	$(A \wedge \sim B) \wedge (\sim B \to C)$	1
Rule 1 (1)	$(A \wedge \sim B)$	2
	$(\sim B \to C)$	3
Rule 1 (2)	A	4
	$\sim B$	5
Rule 6 (3)	$\sim(\sim B)$ | C	6
	✓	7
Rule 5 (6)	B	
	X {B, ~B}	

* X-(closed) if their is complementary atoms elements. like B, ~B. in a path.
* ✓- (open) no complement elements
* A, ~B, C are the atoms in opened path. Prove that Truth values of A, ~B, C are true. {A=T; ~B=T; C=T} (or) {A=T; B=F; C=T} ~ Model.

* Parts in a tableau tree extend from root to the leaf nodes.
* There are 2 paths in the tree. starting from the root node to leaf nodes ending at B and C.
* It is observed that first path from root to B becomes closed. Because of the presence of complementary atoms B and ~B. while the other path remains open.
* The model of the formula is obtained by assigning 'T' to all atomic formulas appearing on the open path.

$\therefore \{A=T; B=F; C=T\}$ — Model

↳ is a model under which the given formula is true.

Model:-

For a given formula 'α' a model is the interpretation under which the formula is true, we assign 'T' to all atomic formulae appearing on the open path.

Note-1:- Path is said to be contradictory (or) closed, if complementary atoms appear on the same path.

Note-2:- If all paths are closed, it is contradictory tableau, such a formula 'α' is incosistent (or) unsatisfiable.

Note-3:- A formula 'α' is said to be consistent (or) satisfiable, if the tableau with root 'α' is not a contradictory tableau, that means there is atleast one open path.

Note-4:- If we obtained a contradictory tableau with root $\sim\alpha$. we say that, the formula 'α' is tableau provable.

Note-5:- If 'α' is tableau provable then it is a valid formula (Tautology).

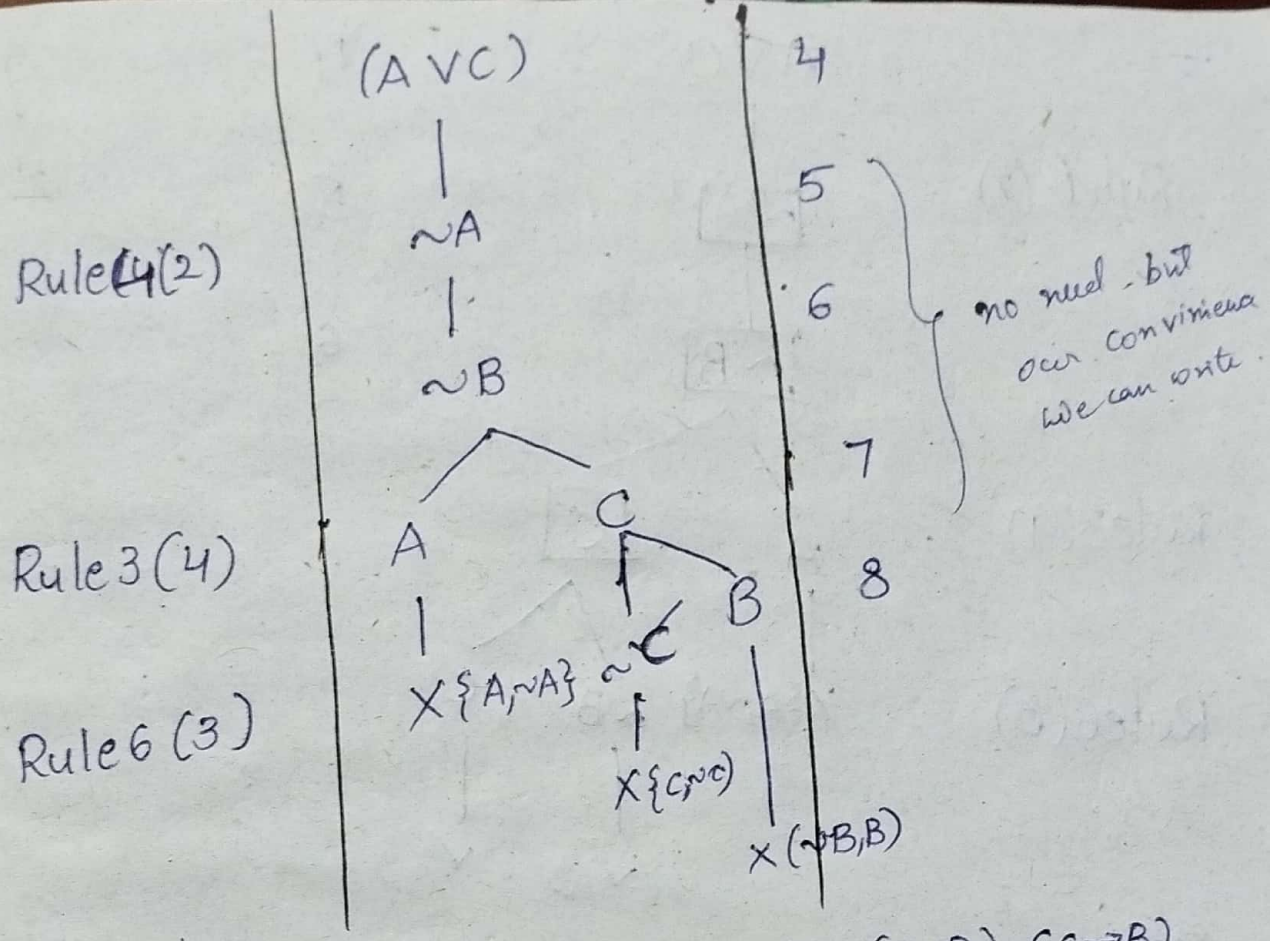
Note-6:- Let 'S' be a set of formulae.

Formula 'α' is said to be tableau provable from 'S'. if there is a contradictory tableau from S with $\sim\alpha$ as root.

2) Show that $S = \{\sim(A \vee B), (C \to B), (A \vee C)\}$ is unsatisfiable / inconsistent.

Sol:-

Description	Formula	Line number
Tableau Root	$\sim(A \vee B) \wedge (C \to B) \wedge (A \vee C)$	1
	|	
Rule1 (1)	$\sim(A \vee B)$	2
	|	
	$(C \to B)$	3
	|	
	$(A \vee C)$	4



∴ The set of formula $S=\{\sim(A \vee B), (C \rightarrow B), (A \vee C)\}$ is inconsistent / unsatisfiable.

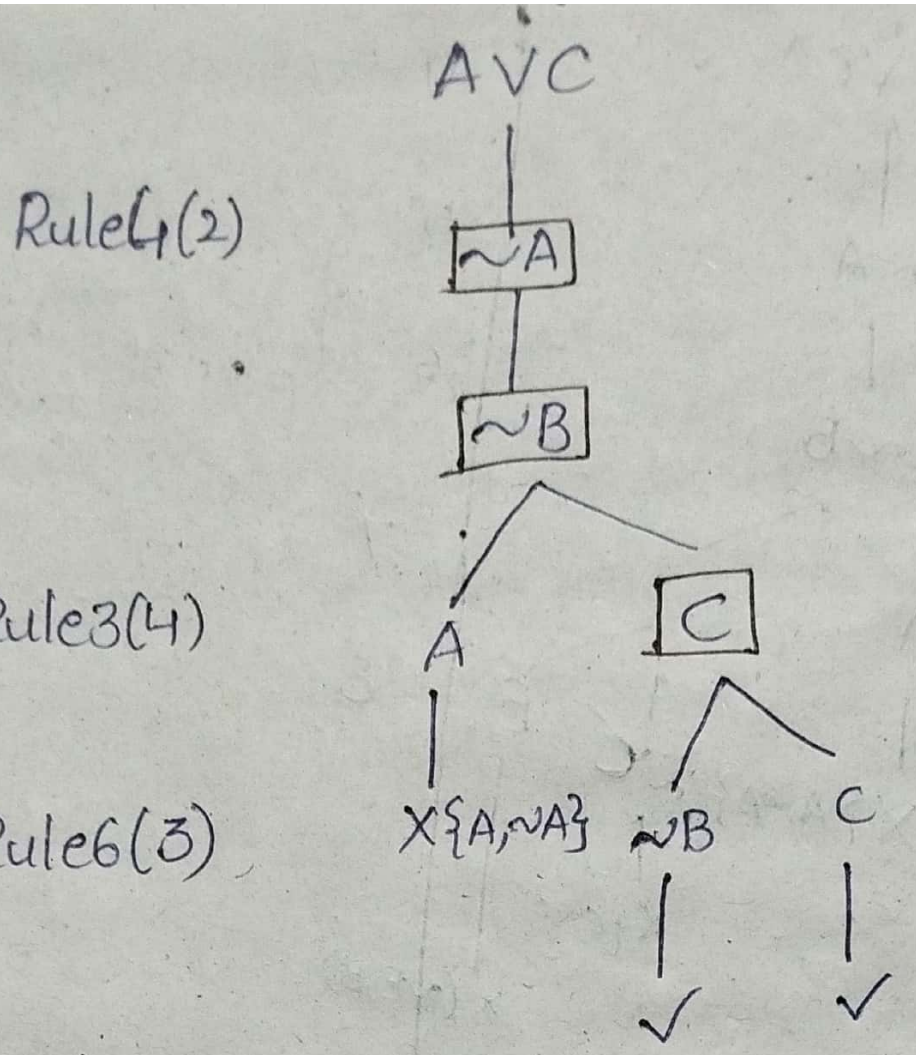
Since, all paths are closed, the obtained tableau is contradictory.

∴ The set S is unsatisfiable / consistent.

5) Show that $S=\{\sim(A \vee B), (B \rightarrow C), (A \vee C)\}$ is consistent / satisfiable.

Sol:-

Description	Formula	linenumber
Tableau root	$\{\sim(A \vee B) \wedge (B \rightarrow C) \wedge (A \vee C)$	1
	|	
rule 1 (1)	$\sim(A \vee B)$	2
	|	
	$(B \rightarrow C)$	3
	|	
	$A \vee C$	4
	|	



~~Model: {~A~~

Since, there is atleast one open path the tableau is not contradictory and therefore, the set S is satisfiable/ consistent.

. The Model for the formula $\sim(A \vee B) \wedge (B \rightarrow C)$ $\wedge(A \vee C)$ is $\{A=F, B=F, C=T\}$ (or) $\{\sim A=T, \sim B=T, C=T\}$

4) Show that B is a logical Consequence of $S = \{A \rightarrow B, A\}$.

Sol:- Description | Formula | Line number

Tableau root — $\sim B$ — 1

Premise 1 — $A \to B$ — 2

Premise 2 — A — 3

Rule 6(2) — $\sim A$ | B — 4

$X\{A, \sim A\}$ | $X\{B, \sim B\}$

From the above tableau, 'B' is tableau provable from S.

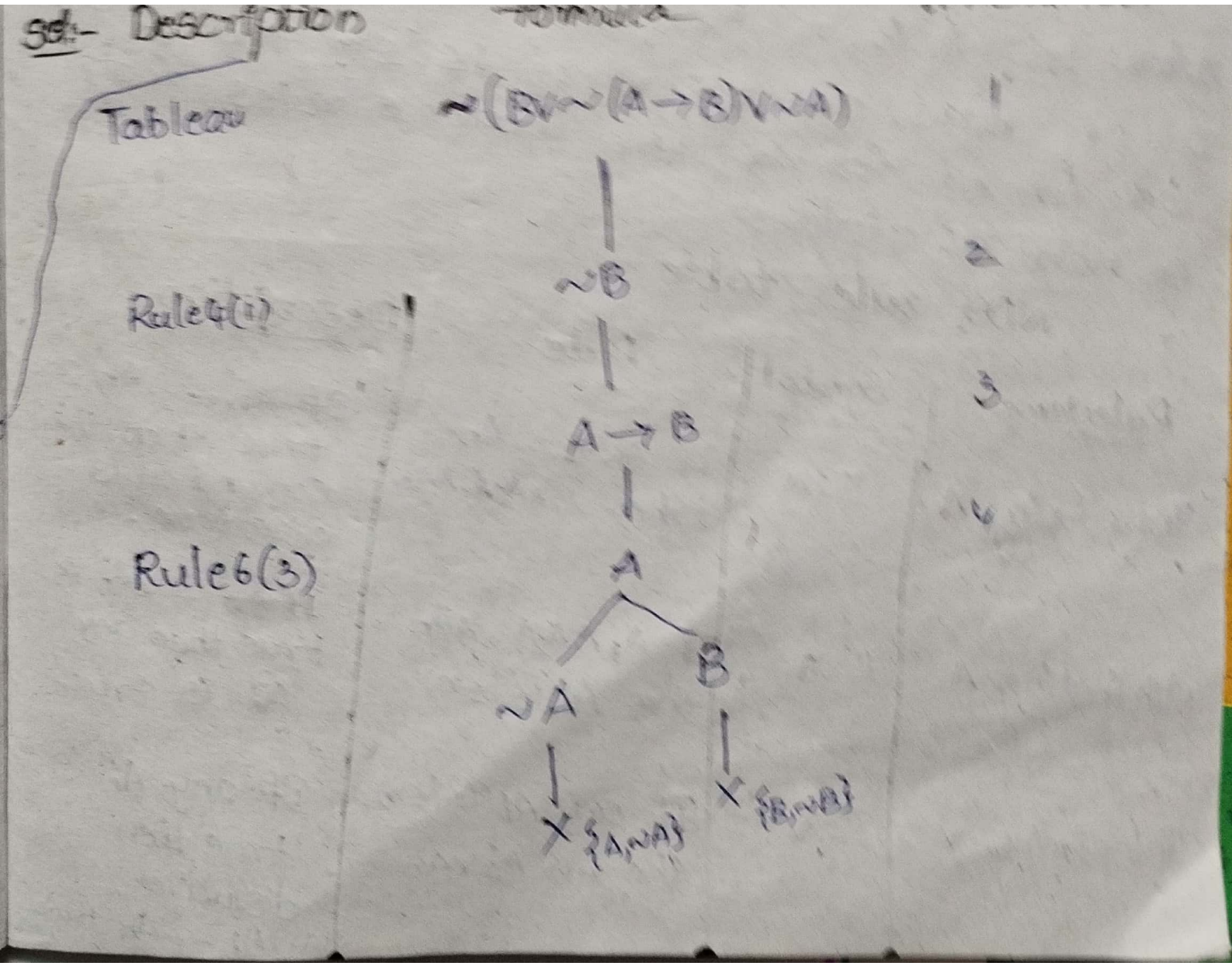
$\therefore$ B is a logical Consequence of S.

5. Show that $\alpha : B \vee \sim(A \rightarrow B) \vee \sim A$ is valid.

Inorder to show that α is valid, we have to show that α is tableau ~~provable~~ refutable
i.e, tableau tree with $\sim\alpha$ is contradictory.

Sol:- Description

Tableau

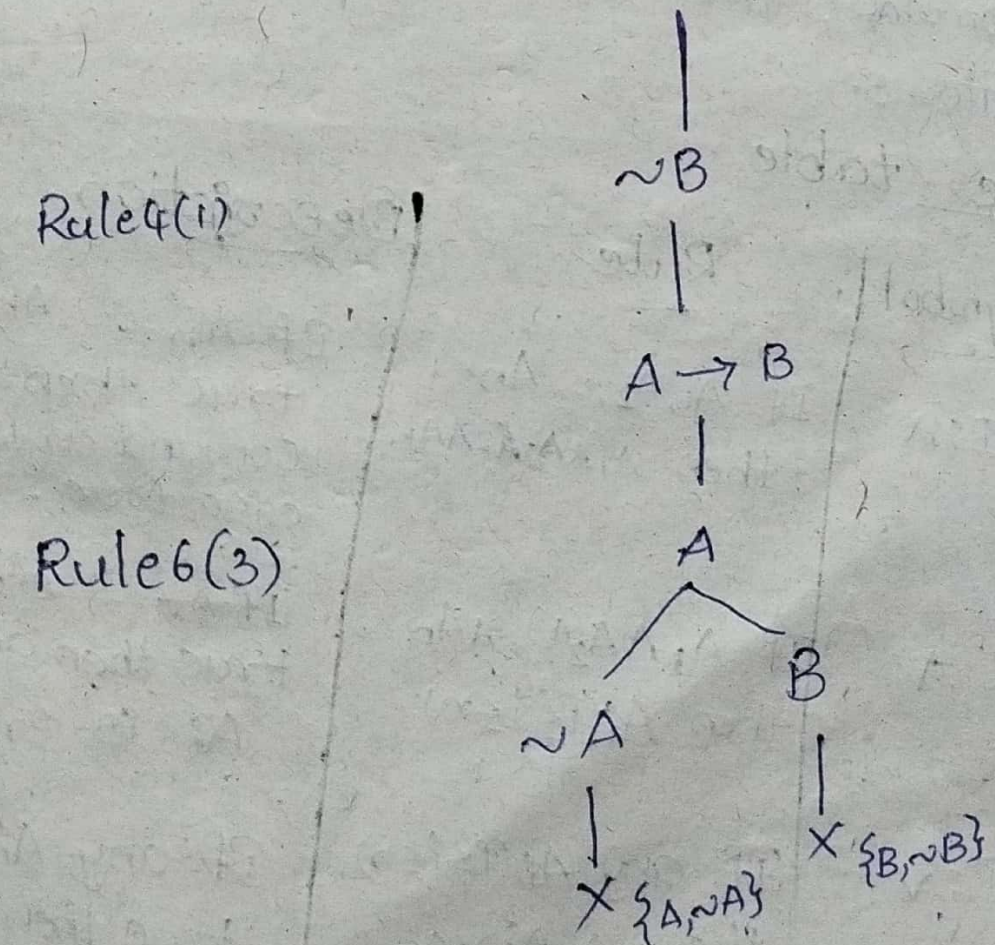


5. Show that $\alpha : B \vee \sim(A \to B) \vee \sim A$ is valid.

Sol:- Description — Formula — linenumber

Tableau

$\sim(B \vee \sim(A \to B) \vee \sim A)$ — 1



* Unit-I Current Trends in AI

1. Neural Networks & Deeplearning – Patterns, multiplelayer of analysis.
2. Transfer learning – Pre trained model eg:- cars, trucks.
3. Natural language Processing – Machine will understand and produce human language
4. ~~Reference~~ Reinforcement Learning – System learns "trail & error" approach
5. Explainable AI (XAI) – AI decision making → transparent & understandable to humans.

U-II → Explain significance of NDS with Example?

Natural Deduction System:-

It is based on a set of deductive inference rules. Let $A_1, A_2, \ldots, A_k$ where $1 \le k \le n$ are a set of atoms and α_j where $1 \le j \le m$ and β are well-formed formulae, the inference rules may be stated as follows:-

NDS rule table

RuleName	Symbol	Rule	Description.
Introducing $\wedge$	I:$\wedge$	If $A_1, A_2, \ldots A_n$ then $A_1 \wedge A_2 \wedge \ldots \wedge A_n$	If $A_1, \ldots, A_n$ are true then their conjunction is also true.
Eliminating $\wedge$	E:$\wedge$	If $A_1 \wedge A_2 \wedge \ldots \wedge A_n$ then $A_i (1 \le i \le n)$	If $A_1 \wedge \ldots \wedge A_n$ is true then any A_i is true
Introducing $\vee$	I:$\vee$	If any $A_i (1 \le i \le n)$, the $A_1 \vee A_2 \vee A_n$	If any A_i is true then disjunction of

Eliminating
Introducing
Eliminating
Introducing
Eliminating
Introduci
Eliminating

* Unit-I part-2
Character
Solved
1. Types

Unit-I Current Trends in AI

1. Neural Networks & Deeplearning – Patterns, multiple layer of analysis.
2. Transfer learning – Pre trained model eg:- cars, trucks.
3. Natural language Processing – Machine will understand and produce human language
4. ~~Reference~~ Reinforcement Learning – System learns "trail & error" approach
5. Explainable AI (XAI) – AI decision making ↓ transparent & understandable to humans.

U-II → Explain significance of NDS with Example?

Natural Deduction System:-

It is based on a set of deductive inference rules. Let $A_1, A_2, \ldots, A_k$ where $1 \le k \le n$ are a set of atoms and α_j where $1 \le j \le m$ and β are well-formed formulae, the inference rules may be stated as follows:-

NDS rule table

RuleName	Symbol	Rule	Description.
Introducing $\wedge$	I: $\wedge$	If $A_1, A_2, \ldots A_n$ then $A_1 \wedge A_2 \wedge \ldots \wedge A_n$	If $A_1, \ldots, A_n$ are true then their conjunction is also true.
Eliminating $\wedge$	E: $\wedge$	If $A_1 \wedge A_2 \wedge \ldots \wedge A_n$ then $A_i (1 \le i \le n)$	If $A_1 \wedge \ldots \wedge A_n$ is true then any A_i is true.
Introducing $\vee$	I: $\vee$	If any $A_i (1 \le i \le n)$, the $A_1 \vee A_2 \vee A_n$	If any A_i is true then disjuction of $A_1, A_2 \ldots A_n$ true

Eliminating $\vee$	E:$\vee$	If $A_1 \vee A_2 \vee \ldots \vee A_n$, $A_1 \to A, \ldots, A_n \to A$ then A.	If $A_1 \vee \ldots \vee A_n$, $A_1 \to A \ldots A_n \to A$ are true then A has to be true.
Introducing $\to$	I:$\to$	If from $\alpha_1, \ldots, \alpha_n$ infer β is proved, then $(\alpha_1 \wedge \alpha_2 \ldots \alpha_n) \to \beta$ is proved.	
Eliminating $\to$	E:$\to$	If $A_1 \to A$, A_1 then A	
Introducing $\leftrightarrow$	I:$\leftrightarrow$	If $A_1 \to A_2$, $A_2 \to A_1$ then $A_1 \leftrightarrow A_2$.	
Eliminating $\leftrightarrow$	E:$\leftrightarrow$	If $A_1 \leftrightarrow A_2$ then $A_1 \to A_2$, $A_2 \to A_1$	
Introducing $\sim$	I:$\sim$	If from A infer $A_1 \wedge \sim A_1$ is proved then $\sim A$ is proved.	
Eliminating $\sim$	E:$\sim$	If from $\sim A$ infer $A_1 \wedge \sim A_1$ is proved then A is proved.	

* Unit-1 / Part-2 Topics.

Characteristics of problems that can be solved by various AI techniques.

1. Types of problem – Ignorable (Theorem proving - lemmas - simple control strategies).
   – Recoverable (water Jug, Back tracking, push down stack).
   – Irrecoverable (chess, card controlled processing)

Eliminating V	E:V	If $A_1 \vee A_2 \vee \ldots \vee A_n$, $A_1 \rightarrow A, \ldots, A_n \rightarrow A$ then A	If $A_1 \vee \ldots \vee A_n$, $A_1 \rightarrow A \ldots A_n \rightarrow A$ are true then A has to be true.
Introducing →	I:→	If from $\alpha_1, \ldots$ then infer B is proved, then $(\alpha_1 \wedge \alpha_2 \ldots \alpha_n) \rightarrow B$ is proved	
Eliminating →	E:→	If $A_1 \rightarrow A$, A_1 then A	
Introducing ↔	I:↔	If $A_1 \rightarrow A_2$, $A_2 \rightarrow A_1$ then $A_1 \leftrightarrow A_2$	
Eliminating ↔	E:↔	If $A_1 \leftrightarrow A_2$ then $A_1 \rightarrow A_2$; $A_2 \rightarrow A_1$	
Introducing ~	I:~	If from A infer $A_1 \wedge \sim A_1$ is proved then $\sim A$ is proved.	
Eliminating ~	E:~	If from $\sim A$ infer $A_1 \wedge \sim A_1$ is proved then A is proved	

* Unit-I / Part-2 Topics.

Characteristics of problems that can be solved by various AI techniques.

1. Types of problem – Ignorable (Theorem proving - lemmas - simple control strategies).
   - Recoverable (water jug, Back tracking, push down stack).
   - Irrecoverable (chess, card controlled processing)

Eliminating $\vee$	E:$\vee$	If $A_1 \vee A_2 \vee \ldots \vee A_n$, $A_1 \rightarrow A, \ldots, A_n \rightarrow A$ then A.	If $A_1 \vee \ldots \vee A_n$, $A_1 \rightarrow A \ldots A_n \rightarrow A$ are true then A has to be true.
Introducing $\rightarrow$	I:$\rightarrow$	If from $\alpha_1, \ldots, \alpha_n$ infer β is proved, then $(\alpha_1 \wedge \alpha_2 \ldots \wedge \alpha_n) \rightarrow \beta$ is proved.	
Eliminating $\rightarrow$	E:$\rightarrow$	If $A_1 \rightarrow A$, A_1 then A	
Introducing $\leftrightarrow$	I:$\leftrightarrow$	If $A_1 \rightarrow A_2$, $A_2 \rightarrow A_1$ then $A_1 \leftrightarrow A_2$.	
Eliminating $\leftrightarrow$	E:$\leftrightarrow$	If $A_1 \leftrightarrow A_2$ then $A_1 \rightarrow A_2$, $A_2 \rightarrow A_1$	
Introducing $\sim$	I:$\sim$	If from A infer $A_1 \wedge \sim A_1$ is proved then $\sim A$ is proved.	
Eliminating $\sim$	E:$\sim$	If from $\sim A$ infer $A_1 \wedge \sim A_1$ is proved then A is proved.	

* Unit-I Part-2 Topics.

Characteristics of problems that can be solved by various AI techniques.

1. Types of problem – Ignorable (Theorem proving-lemmas-simple control strategies).
   – Recoverable (water Jug, Back tracking, Push down stack).
   – Irrecoverable (chess, card controlled processing)

2. Decomposibility of a problem

3. Role of knowledge - facts & rules - problem solving - state space.

4. Consistency of knowledge Base - It is daytime/ It is night time - inconsistent

5. Requirements of Solutions - Absolute (exact ans) (any path) / Relative (approximate ans) (or) Relatively good answer.

multiple/Any path prblm.

Travel Sales Person prblm.

Best path prblms.

* Example for NDS:-

* show that $A \wedge (B \vee C)$ is deduced from $A \wedge B$.

Sol:-

Description	Formula	Comments
Theorem	from $A \wedge B$ infer $A \wedge (B \vee C)$	To be proved
Hypothesis	$A \wedge B$	1
E: $\wedge$ (1)	A	2
E: $\wedge$ (1)	B	3
I: $\vee$ (3)	$B \vee C$	4
I: $\wedge$ (2)(4)	$A \wedge (B \vee C)$	proved.

Unit-II

Deduction Theorem:-

* To prove a formula $(\alpha_1 \wedge \alpha_2 \wedge \dots \wedge \alpha_n) \rightarrow \beta$.. It is sufficient to prove a theorem from $\alpha_1, \alpha_2, \dots, \alpha_n$ infer β. (or) $\alpha_1, \alpha_2, \dots, \alpha_n \vdash \beta$ (or) $\{\alpha_1, \alpha_2, \dots, \alpha_n\} \Rightarrow \beta$.

* Conversely, If $(\alpha_1 \wedge \alpha_2 \wedge \dots \wedge \alpha_n) \rightarrow \beta$. is proved then the theorem from $\alpha_1, \alpha_2, \dots \alpha_n \vdash \beta$ is assumed to be proved.

* Show that infer $[(A \xrightarrow{\alpha_1} B) \wedge (B \xrightarrow{\alpha_2} C)] \rightarrow (A \xrightarrow{B} C)$

Sol:-

Description	Formula	comments.
Theorem	from $[(A \rightarrow B), (B \rightarrow C)]$ infer $(A \rightarrow C)$	To be proved
Hypothesis1	$A \rightarrow B$	1
Hypothesis2	$B \rightarrow C$	2
Sub-theorem	from A. infer C	3
Hypothesis	A	3.1
Elimination/E:	B	3.2
E: $\rightarrow$(1, 3.1)	C	3.3
E: $\rightarrow$(2, 3.2)	$A \rightarrow C$	proved.
I: $\rightarrow$(3)		

* Problem statement:- The theorem infer $[(A \rightarrow B) \wedge (B \rightarrow C)] \rightarrow (A \rightarrow C)$ is reduced to the theorem from $(A \rightarrow B), (B \rightarrow C)$ infer $(A \rightarrow C)$ using Deduction theorem.

* Further, to prove $(A \rightarrow C)$. we will have to prove a sub-theorem from $A \vdash C$ / A Infer C.

The proof of the theorem is as follows:-

Development of AI languages.

- Knowledge Representation Schemes
- Pattern Recognition
- Flexible Search
- Program

} AI program language Data.

prolog - logic Programming lange
- Inorder to predicate logic

LISP — Functional language
- Lambda calculus (Formal System of mathematical logic).

pop-2 — Stack Based language
- high flexible (POP11 - embedded in AI environment).

ML — Machine Learning.

TIC-TAC-TOE Game Playing

Human Computer

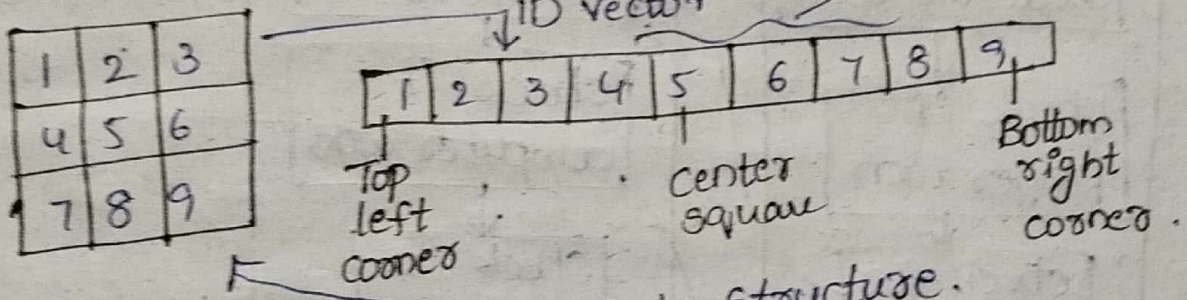
(1) - X , O - (2)

↓

First person/player choose 'X' (Mark) corner.

3×3 Grid.

corner	edge	corner
edge	Center square	edge
/	edge	corner



Approach 1: Board is a Data Structure.

0	0	0
0	0	0
0	0	0

0 - zero.

Blank initially.

After that fill the slots with 'x' and '0'.

* 'x' represented with '1'.

* '0' " '2'

0	0	0
1	0	2
0	0	0

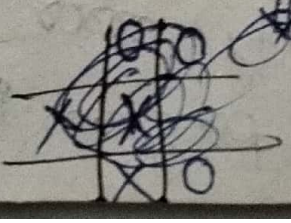
~~Approach 2~~:- Move Table DS → used by computer System not by HB

Ternary number — 0, 1, 2

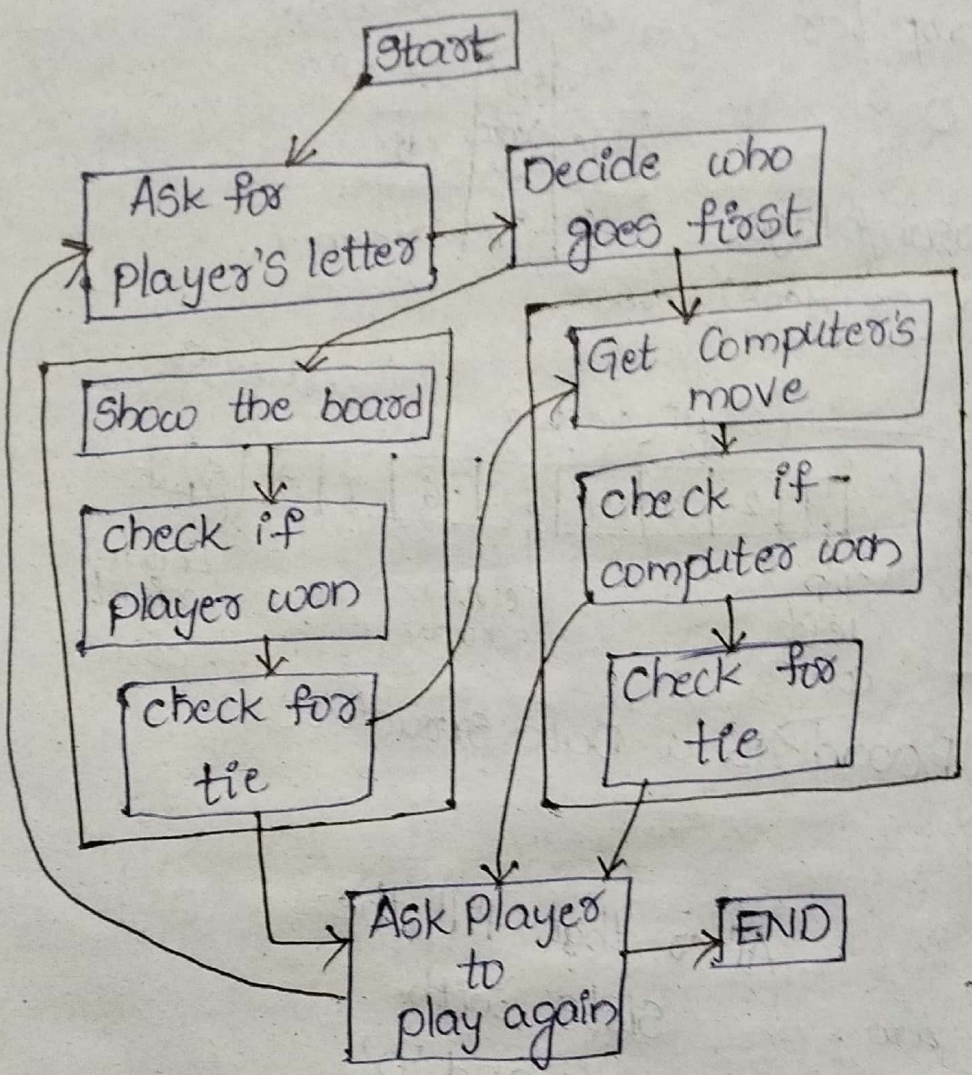
Base (3)

3^9 element vector.

Index	Current Board Position	New Board Position
0	000000000	000010000
1	000000001	020000000
2	000000002	000100002
3	000000010	002000010
4	000000011	
	So on	



Flowchart



Advantages:-

*The program is very efficient in time.

Disadvantages:-

*Lot of space is required to store the move table.

*Lot of work to specify all entries in the move table.

* Difficult to extend.

Highly error prone. as a data is voluminous

*The program is not at all intelligent.

II-Approach *

Data Structures

Board:- 9 element Vector, 2→blank, 3→X, 5→O

Turn: Integer → move to be played.

1→I; - - - - 9→ last move

Algorithm: 3 Sub procedures

1. Make2: Return 5 → Center Squares blank
otherwise, return any blank non-Corner
Square → (2,4,6 or 8).

2. Poss win(P): (possibility of winning) Returns 0, if P cannot move on next move. otherwise, returns no. of square that constitutes the winning move.
- checks each row, Column and diagonal, one at a time.
- Product is 18 (3x3x2), X can win, If it is 50(5x5x2), O can win.

3. Go(n): Makes a move in square 'n' sets Board to 3 if turn is odd and to 5 if turn is even.

Strategy of each turn:-

Turn=1 Go(1)

Turn(2): Board[5] = blank, Go(5) else Go(1)

Turn=3: If Board[9] is blank, Go(9) else Go(3).

Turn=4: If Posswin(x) is not zero, then Go(Posswin(x)), else Go(Make2);

Turn=5: If posswin(x) is not zero, then Go(Posswin(x)), else if posswin(o) is not zero, then Go(Posswin(o)), else if ~~posswin(o) is not zero, then~~ Board[7] is blank, Go(7) else Go(3);

Turn=6: If posswin(o) is not zero, then Go(posswin(x)), else if posswin(x) is not zero, then Go(Posswin(x)), else Go(Make2);

Turn=7:- If posswin(x) is not 0, then Go(posswin(x)), else if posswin(o) is not 0, then Go(posswin(o)), else go anywhere that is blank.

Turn=8:- If posswin(o) is not zero, Go(posswin(o)) else if posswin(x) is not 0, then Go (posswin(x)) else go anywhere that is blank,

Turn=9; is same as turn=7.

Turn=3: If Board[9] is blank, Go(9) else Go(3).

Turn=4: If Posswin(X) is not zero, then Go(Posswin(X)), else Go(Make2);

Turn=5: If posswin(X) is not zero, then Go(Posswin(X)), else if posswin(O) is not zero, then Go(Posswin(O)), else if ~~posswin(O) is not zero, then~~ Board[7] is blank, Go(7) else Go(3);

Turn=6: If posswin(O) is not zero, then Go(posswin(X)), else if posswin(X) is not zero, then Go(Posswin(X)), else Go(Make2);

Turn=7: - If posswin(X) is not 0, then Go(Posswin(X)), else if posswin(O) is not 0, then Go(Posswin(O)), else go anywhere that is blank.

Turn=8: - If posswin(O) is not zero, Go(posswin(O)), else if posswin(X) is not 0, then Go (posswin(X)) else go anywhere that is blank,

Turn=9; is same as turn=7,

IInd approach (Tic-Tac-Toe).

Advantages:-

* Program is effecient in terms of space.
* It is easy to understand the strategy.

Disadvantages:-

* It is not efficient interms of tie.
* It is not possible to generalize the programmers knowledge.

III Approach ★

Data Structures

Board: Magic square of order 3

List: Block list of each player is maintained.

Strategy :-

- Each pair of blocks a player owns is considered.
- Difference D between 15 and the sum of the two blocks is computed.
  - → If $D \leq 0$ (or) $D > 9$, then these two blocks are not collinear and so can be ignored; otherwise if the block representing the difference is blank, then player can make in that block.
  - → This strategy will produce a possible for the player.

Construction of Magic Square :-

3×3

8	1	6
3	5	7
4	9	2

$$\frac{n(n^2+1)}{2} = 15$$

State-Space

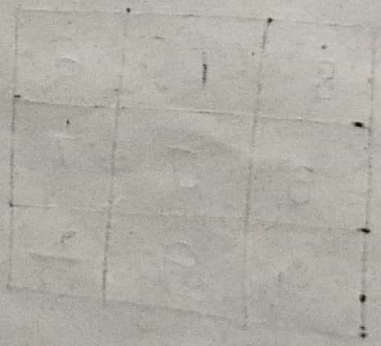
Set of all possible states from start state to goal state.

Production System - It is a formalism that helps AI programs to do search process more conviniently in state-space problem.

* It consists of start and goals state of the problem along with one (or) more databases consisting of suitable & necessary information for the particular task.

* It consists of no. of production Rules in which each production rule has left side that determines applicability of the rule and a right-side that describes the action to be performed if the rule is applied.

* It consists of Control strategy that specifies sequence in which the rules are applied when several rules match at once.



*Water Jug Problem

Problem Statement:- we have 2 jugs, a five (5-g gallon jug and the other 3-gallon (3-g) with low measuring marker on them.

* There is endless supply of water throug -h the tap.
* Our task is to get 4-gallon (4-g) of water in 5-gallon (5-g) Jug.

Solution:-

State-space for this problem can be descr -ibed as the set of ordered pairs of integers (x, y) such that x represents the number of gallons of water in 5-gallons (5-g) jug and y for (3-g) Jug.

Start state is (0,0)

goal state is (4, n)

For any Value $N \leq 3$ (0-3)

Production Rules for Water Jug problem.

Rule no.	Left of Rule	Right of Rule	Description
1.	$(x, y \mid x<5)$	$(5, y)$	Full-5g jug
2.	$(x, y \mid x>0)$	$(0, y)$	Empty 5-g jug
3.	$(x, y \mid y<3)$	$(x, 3)$	Fill 3-g jug
4.	$(x, y \mid y>0)$	$(x, 0)$	Empty 3-g jug
5.	$(x, y \mid x+y \leq 5 \wedge y>0)$	$(x+y, 0)$	Empty 3-g into 5-g jug.
6.	$(x, y \mid x+y \leq 3 \wedge x>0)$	$(0, x+y)$	Empty 5-g into 3-g jug.
7.	$(x, y \mid x+y \geq 5 \wedge y>0)$	$(5, y-(5-x))$	Pour water from 3-g jug into 5-g jug until 5-g jug is full
8.	$(x, y \mid x+y \geq 3 \wedge x>0)$	$(x-(3-y), 3)$	Pour water from 5-g jug into 3-g jug until 3-g jug is full

5 3
X Y

(0, 0)
↓
(5,0) — R(1)
↓
(2,3) — R(8)
↓
(2,0) — R(4)
↓
(0,2) — R(6)
↓
(5,2) — R(1)
↓
(4,3) — R(8)

Solution:- FS←(4,N)

Rule applied	5-gjug	3-gjug	Step No.
Start State	0	0	0
1	5	0	1
8	2	3	2
4	2	0	3
6	0	2	4
1	5	2	5
8	4	3	6
goal state	4	—	

* Missionaries and Cannibals Problem:-

Problem Statement:-

Three ~~Machineries~~ Missionaries and 3 cannibals wants to cross a river.

There is a boat on their side of the river, that can be used by either 1 (or) 2 persons.

How should they use this boat to cross the river in such a way that cannibals never out no. missionaries on either side of the river?

If the cannibals ever out number the mission-aries (on either bank). Then the missionaries will be eaten..

How can they all cross over without

anyone being eaten.

Solution:-

State-Space for this problem can be described as the set of ordered pairs of left and right banks of the river as (L,R) where each bank is represented as a list [nM, mC, B].

Here, n is the no. of missionaries 'M'
m is the no. of Cannibals 'C'.
and B represents the Boats

[([nM, mC, OB], [nM, mC, IB])]

1. Start-State:- ([3M, 3C, IB]), [0M, 0C, 0B])

•IB means when boat is present at OB means it is absent.

2. Any State:- ([n_1M, m_1C, -], [n_2M, m_2C, -])

With Constraints at any state as,

$n_1(\neq 0) \geq m_1$; $n_2(\neq 0) \geq m_2$;

$n_1 + n_2 = 3$; $m_1 + m_2 = 3$

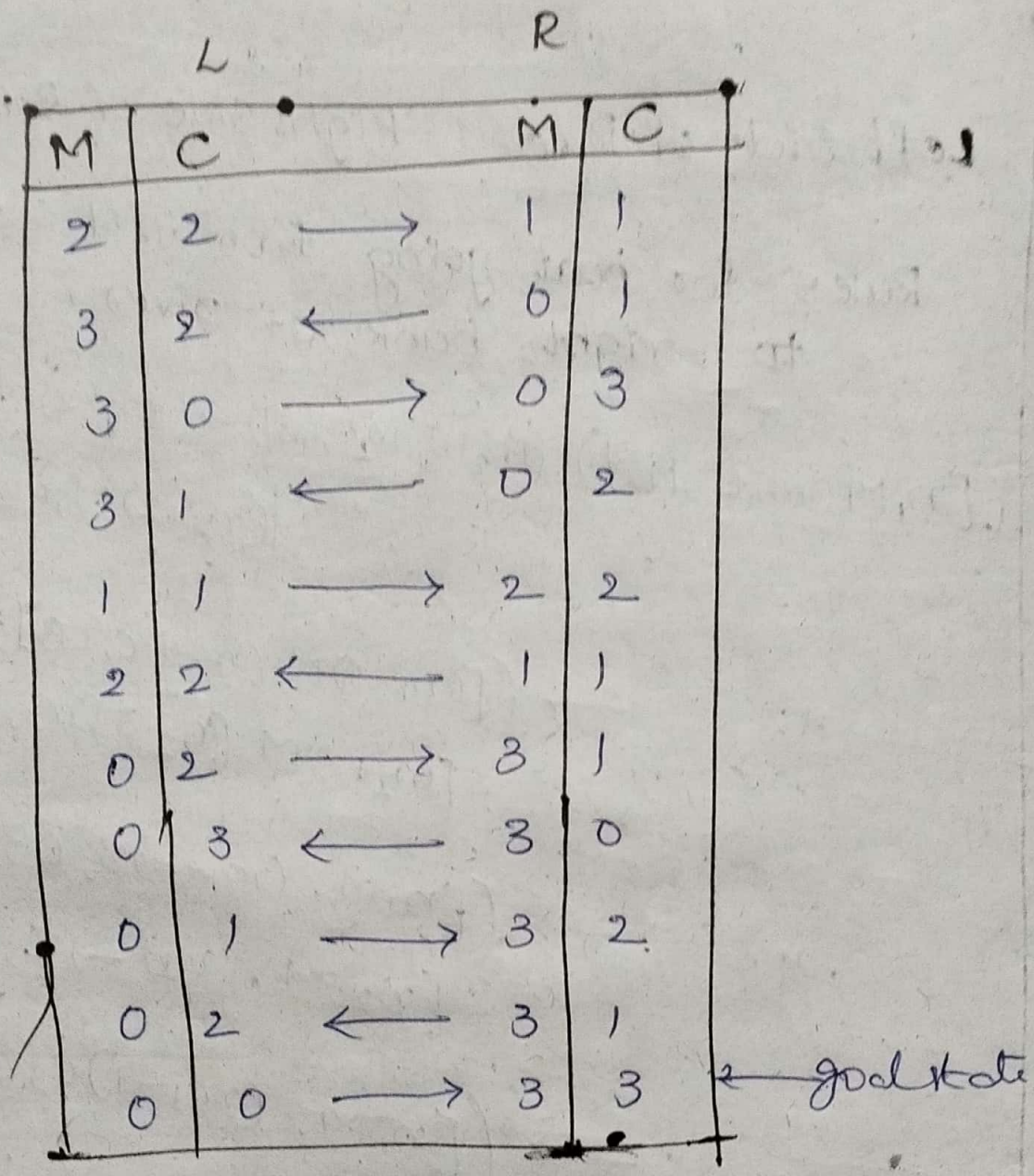
Boat can be on either side

3. Goal State:- ([0M, 0C, 0B], [3M, 3C, IB]).

Production Rules for missionaries cannibals problem.

Rule No	Left Side of Rule	Right Side of Rule
	Rules for boat going from left bank to right bank of river.	
L_1	$([n_1M, m_1C, 1B]), ([n_2M, m_2C, 0B])$	$\rightarrow ([(n_1-2)M, m_1C, 0B], [(n_2+2)M, m_2C, 1B])$
L_2	"	$\rightarrow ([(n_1-1)M, (m_1-1)C, 0B], [(n_2+1)M, (m_2+1)C, 1B]$
L_3	"	$\rightarrow ([n_1M, (m_1-2)C, 0B], [n_2M, (m_2+2)C, 1B]$.
L_4	"	$\rightarrow ([n_1M, (m_1-1)C, 0B], [n_2M, (m_2+1)C, 1B])$.
	Rules for boat going from R to L.	
R_1	$([n_1M, m_1C, 0B]), [n_2M, m_2C, 1B]$	$\rightarrow ([(n_1+2)M, m_2C, 1B], [(n_2+2)M, m_2C, 0B])$

Solution:-



Unit-II

Cryptarithmetic

```
  1  2          K  D
+ 9  4        + S  N
-------       -------
  1 0 6         K Z 0
-------       -------
```

0-9 can be replaced with A-Z

Ex:- C_2C_1
TO
+GO
OUT
↑
has to be '1' (msB)

letter digit

T → 2
O → 1
G → 8
U → 0

→ $C_2 = O = 1$

O+O=T
⇒ 1+1=2
T = 2

C_2 C_1

T	2		0	1
+	G	8	0	1
1	U	0	T	2

No carry
so, C_1 is '0'

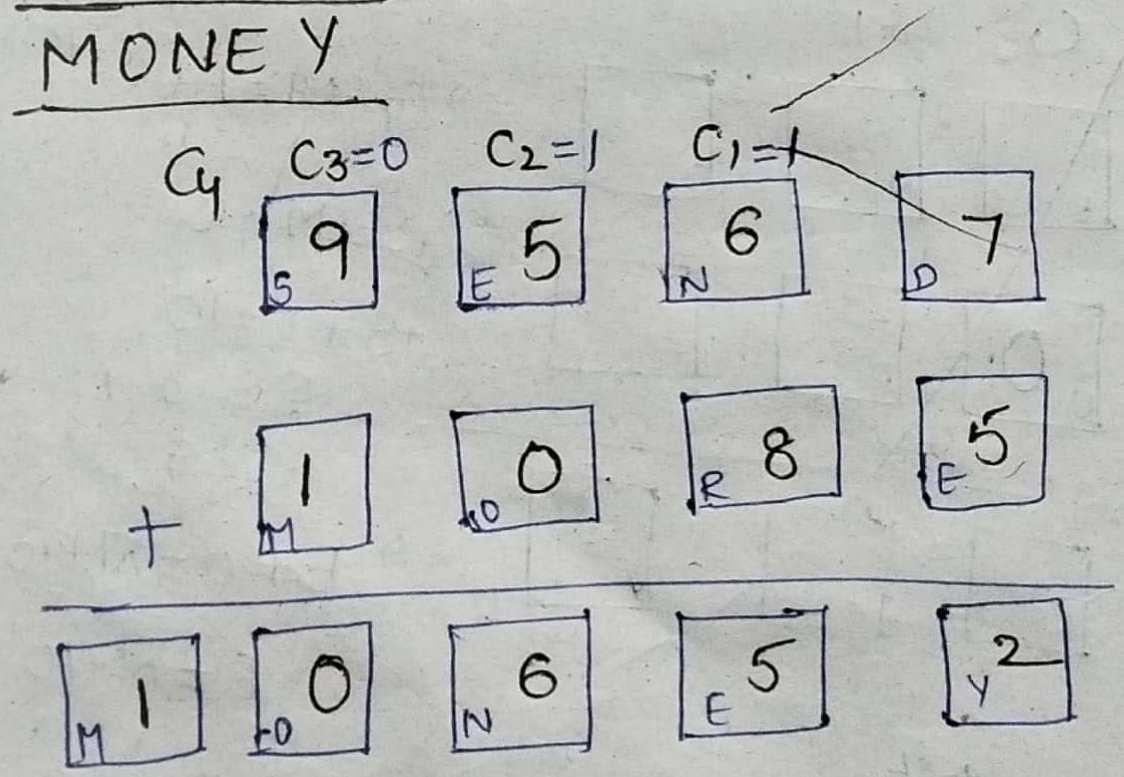
$C_1 + T + G = U + 10$
⇒ O + 2 + G = U + 10
⇒ G = U + 8

Let U
Then,
G = 0 + 8
= 8

* SEND [0-9]

+ MORE

MONEY



1) $\boxed{M=1}$ ($\because C_4=1$ & $10 \leq S+M \leq 19$)

2) $C_3+S+M=O+10$, Let $\boxed{C_3=0}$.

$\boxed{O=0}$

Then $S+M \geq 10$

i.e., It is possible only when $\boxed{S=9}$

3) $C_2+E+O=N \Rightarrow$ either $E=N$ (or) $E+1=N$ depending on the value of C_2 (=0 or 1).

$E \neq N$ $\therefore E+1=N$, only when $C_2=1$.

Let, $\boxed{\therefore E=5}$

Then $N=E+1=5+1=6$.

$\boxed{\therefore N=6}$

4) $C_1+N+R=E+10$.

Let $C_1=0$. Then $6+R=15 \Rightarrow R=9$ (conflict with S).

So, $C_1 \neq 0$. Let $\boxed{C_1=1}$

Then $1+6+R=15 \Rightarrow \boxed{R=8}$

5) $D+5=Y+10 \Rightarrow D=Y+5 \Rightarrow D \geq 5$

i.e, D can hold 5, 6, 7, 8 or 9.

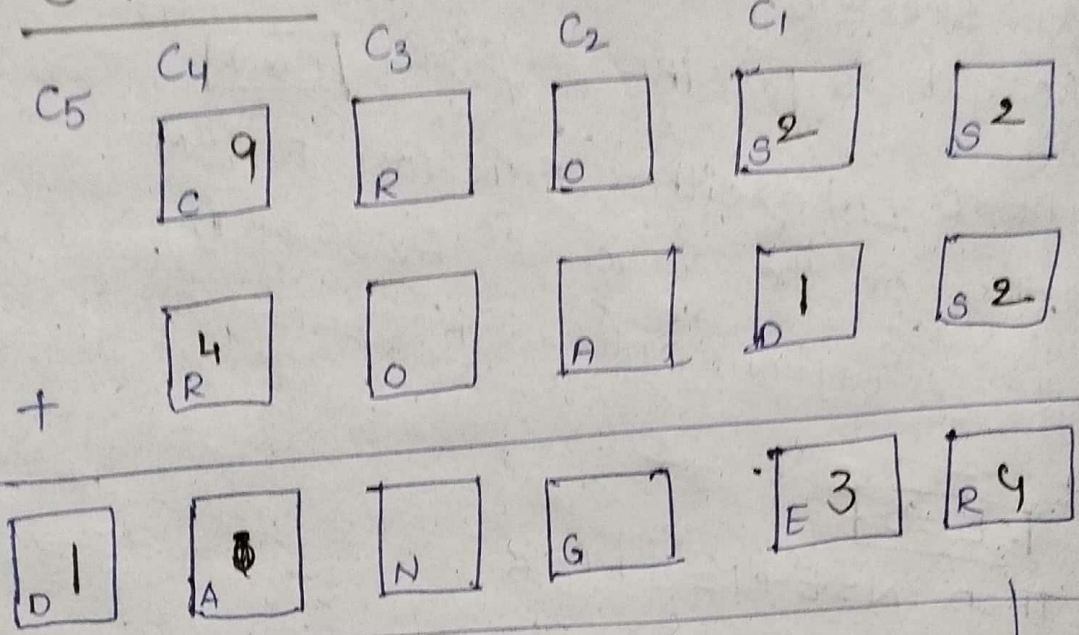
5, 6, 8 & 9 are already assigned.

Let, $\boxed{D=7}$. Then $Y=D-5$

$Y=7-5$

$\boxed{Y=2}$

* CROSS
+ ROADS
DANGER



1) $D = 1$ ($\because C_5 = 1$ & $10 \le 6 + R \le 19$)

2) $S + S = R \Rightarrow 2S = R$ ($\because R$ is even)

i.e., R can be 0, 2, 4, 6 (or) 8.

Let $R = 4$ Then $S = 2$

3) $C_4 + C + R = A + 10$. Let $C_4 = 0$.

Then $C + 4 = A + 10 \Rightarrow C = A + 6$

$\Rightarrow C \ge 6$. C can be 6, 7, 8 (or) 9

Let, $C = 9$.

Then $A = 3$ which has a conflict with $E = 3$.

Our assumption that R=4 is wrong.

Backtracking and assume

$\boxed{R=6}$ Then $\boxed{S=3}$

$C_5=1$	$C_4=0$	$C_3=0$	$C_2=0$	$C_1=0$
	9	6	2	3 3
	6	2	5	1 3
1	5	8	7	4 6

4) $C_4+C+6=A+10$. Let $\boxed{C_4=0}$

Then $C+6=A+10$. $\Rightarrow C=A+4$

$\Rightarrow C\geq 4$. C can hold 4, 5, 6, 7, 8 or 9.

Let C=9. Then $A=C-4$

$=9-4$

$\boxed{A=5}$

5) $C_3+R+O=N$. Let $C_3=0$

Then, $6+O=N$ $\because N\leq 9$

$\Rightarrow O+6\leq 9 \Rightarrow O\leq 3$

The possible Combinations for $N\leq 9$ are (6,0) (6,1) (6,2) and (6,3). The possible Combinations is (6,2)

$\boxed{O=2}$ and $\boxed{N=8}$.

Exchaustive Search (Uniformed Search)

* Breadth First Search
* Depth First Search.

Breadth First Search:- (BFS)

Algorithm:

1) Create a variable called NODE-LIST and set it to the initial state.

2) Until a goal state is formed, (or) NODE-LIST is empty:-

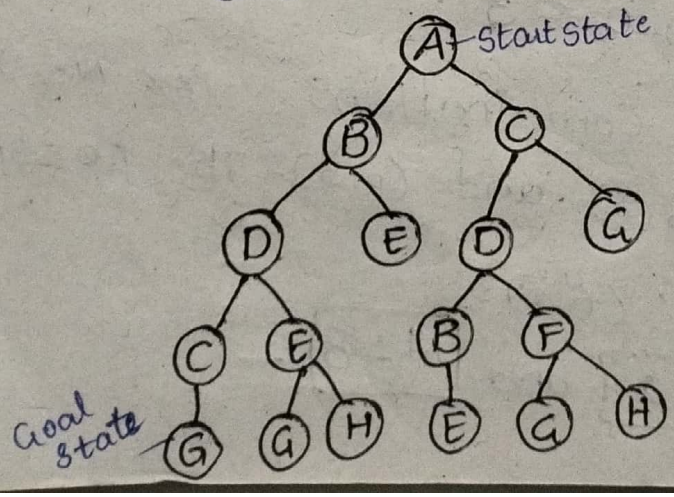
   a) Remove the first element from NODE-LIST and call it E. If NODE-LIST is empty, then quit.

   b) For element E, do the following:-

      (i) Apply the rule to generate a new state. (get the successors of E).

      (ii) If the new state is a goal state, quit and return this state.

      (iii) Otherwise, add the newstate to the end of NODE-LIST



1). NODE_LIST : A

2) E : A , NODE_LIST : B, C

3) E : B , NODE_LIST : C, D, E.

4) E : C , NODE_LIST : D, E, D, G.

* G, the new state that is generated is a goal state. Return G and quit.

* The algorithm returns the path A → C → G by following the parent pointers of the node corresponding to G.

Advantages:-

* It is one of the simplest search strategies
* It is Complete.
  If there is a solution BFS is guarantee to find it.
* If there are multiple solutions the minimal solution can be formed.

Disadvantages:-

It cannot be effectively used unless the search space is quite small.

Depth First Search (DFS):-

Algorithm:-

1) Create a variable called NODE_LIST and set it to the initial state.

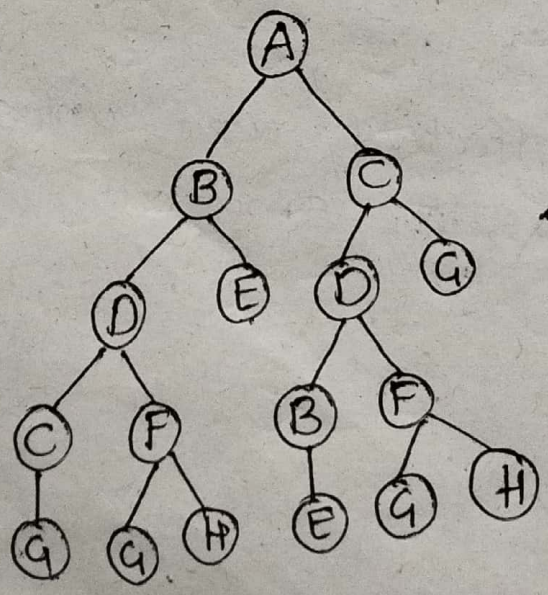
2) If the initial state is a goal state, quit and return SUCCESS.

3) Otherwise, do the following until SUCCESS (or) FAILURE is signaled:

   a) Generate a Successor, E, of the initial state. If there are no more successors, signal failure.

   b) Call depth first search with E as initial state

   c) If Success is returned, signal success. Otherwise, Continue in this loop.



1) NODE_LIST : A

2) E:A A is not the goal state.

NODE_LIST : BC

3) E:B B is not the goal state.

NODE_LIST : DEC

4) E:D , D is not the goalstate

NODE_LIST : CFEC.

5) E:C, C is not goalstate

NODE_LIST : GFEC.

6) E:G , G is a goal state

Return SUCCESS.

Algorithm will return the path

A -B-D-C-G.

Advantages:-

* It requires less memory. Since, only the nodes on current path are stored.
* It may find a solution without examining much of the search space at all.

Disadvantages:-

* It may find a sub-optimal solution.
* It is incomplete.

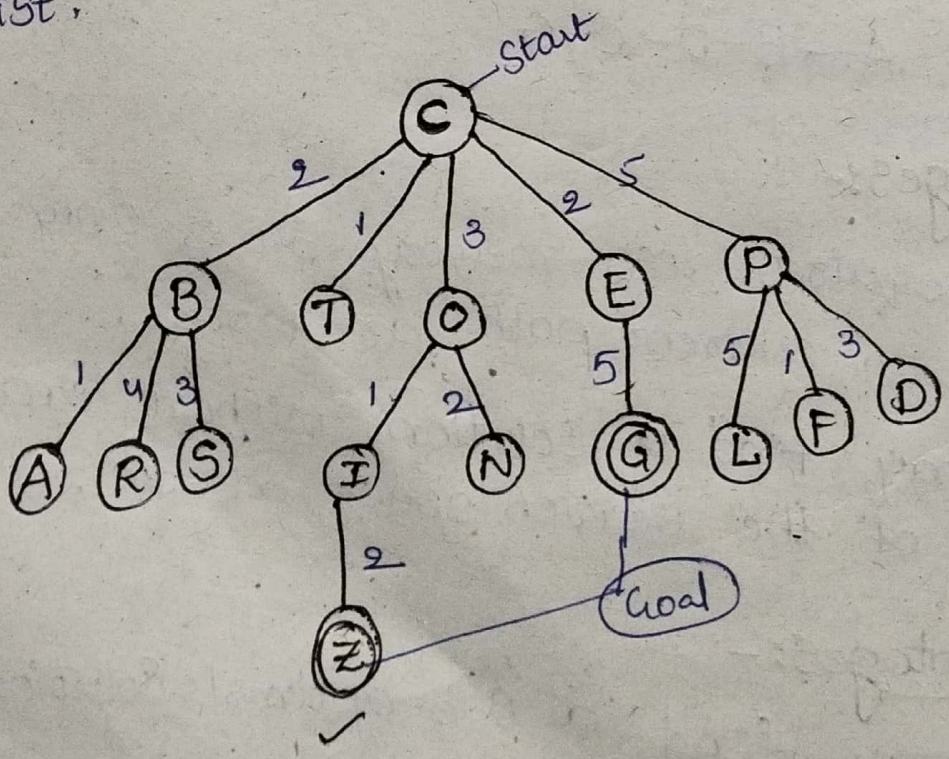
i.e, means a solution may not be found even if it exists.

Branch and Bound Search (Uniform Cost Search)

Algorithm:-

1) Insert root node into the priority queue.
2) Remove the element with highest priority. (Element with least cummulative cost.)
3) If the removed node is the goal node,
   - print total cost and stop the algorithm.

   else
   - Enqueue all the children of the current node to the priority queue, with the cummulative cost from the root as priority and the current node to the visited list.



Q: C(0)

Q: B(2), T(1), O(3), E(2), P(5)

⇓

Q: T(1), B(2), E(2), O(3), P(5).

Q: A(3), R(6), S(5), E(2), O(3), P(5)

⇓

Q: E(2), A(3), O(3), P(5), S(5), R(6)

Q: G(7), A(3), O(3), P(5), S(5), R(6)

⇓

Q: A(3), O(3), P(5), S(5), R(6), G(7)

Q: I(4), N(5), S(5), P(5) R(6) G(7)

Q: Z(6), N(5), S(5), P(5), R(6), G(7)

⇓

N(5), S(5), P(5), R(6), Z(6), G(7).

(no child).

Q: L(10), F(6), D(8), P(5), R(6), Z(6), G(7).

⇓

Q: P(5), R(6), Z(6), ~~R(6), D(8)~~, L

G(7), D(8), L(10).

∴ Z is the goal node with cummulative cost of 6. The path is C-O-I-Z.