Topic 1: Graph Coloring using Backtracking (State Space Tree)

Graph Coloring Problem: Assign colors to each vertex of a graph such that no two adjacent vertices share the same color.

Backtracking Approach:

- Try assigning colors one by one.
- If assigning a color violates the constraint, backtrack.
- Use recursion to try all possibilities.

State Space Tree Structure:

- Root: No node is colored.
- Each level: Represents one vertex.
- Each branch: Represents a color choice.
- Leaf node: A solution if all vertices are colored legally.

Example (3-Coloring, 4 Nodes):

```
          []
        /  |  \
     R    G    B     <- Color choices for Node 1
    /|\  /|\  /|\
   R G B R G B R G B    <- Choices for Node 2 (checking adjacency)
```

Follow-up Question: How many nodes in the worst-case state space tree for n vertices and k colors?
Answer: k^n nodes.

Topic 2: Why is Rational AI Rational?

Definition: A rational agent does the right thing based on its knowledge and percepts to maximize performance.

Why Rational?:

- Acts to achieve best outcome or best expected outcome.
- Based on performance measure.
- Operates under perceptual and environmental constraints.

Example Analogy: Like a chess player planning best moves based on known board state.

Critical Exam Points:

- Difference between rational and omniscient.
- Importance of performance measure.

Follow-up: Can a rational agent make a mistake?
Answer: Yes, due to incomplete information.

Topic 3: Cricket Riddle: 2 Batsmen on 94, 2 Balls, Both Make 100

Answer Strategy: Use a combination of scoring and strike rotation.

1. Ball 1: Batsman A hits 6 (now on 100), retains strike due to odd runs (assume hit is a 3 with overthrows).
2. Ball 2: Batsman B hits 6 (now also on 100).

Exam Insight: This tests logical problem solving and interpretation of edge cases.

Topic 4: Time and Space Complexity of Search Algorithms

Algorithm	Time Complexity	Space Complexity	Complete?	Optimal?
BFS	O(b^d)	O(b^d)	Yes	Yes (for unit cost)
DFS	O(b^m)	O(bm)	No	No
IDFS	O(b^d)	O(bd)	Yes	Yes
A*	O(b^d)	O(b^d)	Yes	Yes (if h is admissible)

Where:

- **b** = branching factor
- **d** = depth of optimal solution
- **m** = maximum depth

Topic 5: Blocks World: BFS vs DFS

- **BFS:** Finds shortest solution (fewer moves).
- **DFS:** May go deep without finding goal.
- BFS uses more memory.

Analogy: Solving a Rubik's cube using layered solution vs random experimentation.

Topic 6: ***A vs Greedy Search****

Feature	A* Search	Greedy Search
Formula	f(n) = g(n) + h(n)	f(n) = h(n)
Uses cost?	Yes	No
Optimal?	Yes (if h is admissible)	No

Memory Trick: "A* is a *star* student—it does homework (g) and thinks ahead (h). Greedy just guesses!"

Topic 7: Problem Formulation

Includes:

1. Initial State
2. Actions

3. Transition Model
4. Goal Test
5. Path Cost

Example: In chess, initial state = board setup; goal = checkmate.

Follow-up: Why is a good problem formulation important? **Answer:** It simplifies search and ensures correctness.

Topic 8: Rationality in AI

Refer to Topic 2.

Topic 9: Turing Test

Goal: Test a machine's ability to exhibit intelligent behavior equivalent to human.

Setup: Human judge talks to both a machine and a human via text. If it can't tell which is which, AI passes.

Criticism: Doesn't measure understanding or consciousness.

Current View: More a philosophical benchmark than practical test.

Follow-up: What replaced Turing Test today?
Answer: Benchmarks like Winograd Schema, BLEU/NLP metrics.

Topic 10: FOL (First Order Logic)

Meaning: Expresses knowledge using quantifiers and predicates.

Benefits:

- More expressive than propositional logic.
- Can represent relationships and quantify over objects.

Example: "Every cat chases a mouse" → $\forall x(Cat(x) \rightarrow \exists y(Mouse(y) \land Chases(x,y)))$

Topic 11: STRIPS vs ADL

Feature	STRIPS	ADL (Action Description Language)
Simplicity	Simpler, limited	More expressive
Negation	Not supported	Supported
Conditions	No quantifiers	Allows conditional effects

Topic 12: Utility Agent Components

1. Sensor
2. Percept History
3. Knowledge Base
4. Utility Function
5. Decision-making module

Goal: Maximize utility over time.

Follow-up: How is this different from goal-based agents? **Answer:** Goal-based achieves target; utility-based ranks preferences.

Topic 13: Reinforcement Learning (RL)

Core Idea: Learn by trial and error using rewards.

Components:

- Agent
- Environment
- Reward signal

- Policy (π)
- Value function (V)

Example: Training a dog using treats.

Popular Algorithms: Q-Learning, Deep Q-Network (DQN), SARSA

Topic 14: Informed vs Uninformed Search

Type	Uses Heuristics?	Example Algorithms
Uninformed	No	BFS, DFS, IDDFS
Informed	Yes	A*, Greedy, Best-First

Memory Trick: Uninformed = "blind", Informed = "guided"

Topic 15: Local Search

- Doesn’t keep full path.
- Focuses on current state and neighbors.
- Examples: Hill Climbing, Simulated Annealing.

Useful in: Optimization problems (e.g., N-Queens, TSP).

Topic 16: Bayesian Networks

- Graphical models representing probabilistic relationships.
- Nodes = variables; Edges = dependencies.

Applications:

- Medical diagnosis

- Spam filtering

Follow-up: What is conditional independence? **Answer:** A variable is independent of others given its parents in the graph.

Topic 17: Steps to Design an Expert System

1. Identify Problem Domain
2. Knowledge Acquisition
3. Knowledge Representation
4. Inference Engine
5. User Interface
6. Testing & Evaluation

Topic 18: Block Diagram of Expert System

[User] ↔ [User Interface] ↔ [Inference Engine] ↔ [Knowledge Base] ↓ [Explanation Facility] [Knowledge Acquisition]

Topic 19: Genetic Algorithm

Inspired by: Natural selection and evolution.

Steps:

1. Initialize population
2. Evaluate fitness
3. Select parents
4. Crossover
5. Mutation
6. Replace population

Applications: Scheduling, optimization, game AI

Follow-up: What's the role of mutation? **Answer:** Maintains diversity in the population.

---

1. Genetic Algorithm (GA)

Definition: Genetic Algorithm is a search heuristic inspired by Charles Darwin's theory of natural selection. It mimics the process of natural evolution, employing techniques such as selection, crossover, and mutation.

Key Concepts:

- **Population**: A set of candidate solutions.
- **Chromosome**: A representation of a solution.
- **Fitness Function**: Measures how good a solution is.
- **Selection**: Choosing the best-fit individuals for reproduction.
- **Crossover**: Combining two parents to produce offspring.
- **Mutation**: Introducing small random changes to individuals.

Applications:

- Feature selection
- Game playing
- Optimization problems

Critical Points for Oral Exam:

- Emphasize natural selection analogy
- Explain genetic operators with examples
- Discuss convergence and diversity

Follow-Up Q: How is mutation rate chosen? **A**: Low mutation maintains structure; high mutation increases diversity but may disrupt convergence. Usually set empirically.

---

2. Heuristics of 8-Puzzle

Popular Heuristics:

- **Misplaced Tile Heuristic (h1)**: Number of tiles not in goal position
- **Manhattan Distance (h2)**: Sum of distances of each tile from its goal position

Critical Insight: Manhattan distance is more informed as it captures relative position.

Follow-Up Q: Can Manhattan distance overestimate the cost? **A**: No. It's admissible, i.e., it never overestimates.

---

3. Alpha-Beta Pruning: Role of Alpha & Beta

Alpha: Best (highest) score for MAX so far. **Beta**: Best (lowest) score for MIN so far.

Usage: Alpha-Beta pruning cuts branches in minimax trees that cannot influence the final decision.

Critical Point: Reduces time complexity from O(b^d) to O(b^{d/2}) in the best case.

Follow-Up Q: What affects pruning efficiency? **A**: Node ordering; better ordering → more pruning.

---

4. Uninformed vs Informed Search

Uninformed Search: No domain knowledge (e.g., BFS, DFS) **Informed Search**: Uses heuristics (e.g., A*, Greedy Best-First)

Comparison Table:

Feature	Uninformed	Informed
Knowledge Used	Only structure	Uses heuristics
Efficiency	Often slower	Faster if heuristic is good

---

5. Choosing Between Two Heuristic Functions

Criteria:

- **Admissibility**: Never overestimates cost
- **Dominance**: If $h1(n) \geq h2(n)$ for all n and both are admissible, h1 is better

Follow-Up Q: What if both heuristics are admissible and non-dominating? **A**: Take the max: $h(n) = \max(h1(n), h2(n))$ → still admissible.

---

6. State Space Tree: Graph Coloring (Backtracking)

Graph Coloring Problem: Color vertices such that no two adjacent nodes have same color.

State Space Tree:

- Root: No vertex colored
- Level 1: Color vertex 1 with color1, color2...
- Each branch: Color next vertex considering constraints

(Provide tree diagram in notebook for visual reference)

---

7. Architecture of a Learning Agent

Components:

- **Learning Element**: Improves with experience
- **Performance Element**: Chooses actions
- **Critic**: Gives feedback
- **Problem Generator**: Suggests exploratory actions

Mnemonic: Learn People Can Perform

---

8. Drawbacks of Expert Systems

- Lack of learning ability
- Hard to update rules
- Brittleness: Fails outside knowledge domain

- No common sense reasoning

---

9. Alpha-Beta Pruning: Sample Graph

(Sketch a minimax tree with example values, prune unnecessary branches using Alpha & Beta update rules)

---

10. Determining a Good Heuristic

Characteristics:

- Admissible
- Consistent (Triangle inequality: $h(n) \leq c(n, a, n') + h(n')$)

Testing: Compare estimated cost vs actual cost over sample runs

---

11. Real Life Example of CSP

Example: Exam Timetabling

- Variables: Courses
- Domains: Time slots
- Constraints: No overlapping exams, room availability

---

12. Drawbacks of A*

- High memory usage (stores all generated nodes)
- Performance heavily depends on heuristic
- Can be slow for large search spaces

---

13. Time & Space Complexity Table

Algorithm	Time	Space	Complete?	Optimal?
BFS	O(b^d)	O(b^d)	Yes	Yes
DFS	O(b^m)	O(bm)	No	No
IDDFS	O(b^d)	O(bd)	Yes	Yes
A*	O(b^d)	O(b^d)	Yes	Yes (if h admissible)
AO*	Depends	High	Yes	Yes

---

14. Wumpus World Environment

Type: Discrete, partially observable, stochastic, sequential, dynamic

Agent's goal: Avoid pit and Wumpus, grab gold, exit safely

Used For: Knowledge representation, inference, and planning

---

15. IDDFS: Depth Increase Strategy

Better Strategy: Exponentially (1, 2, 4...)

- Pros: Faster convergence to deeper goal
- Cons: May miss shallow solutions initially

---

16. Problem Formulation: Self-Driving Car

- **States**: Position, speed, sensor input
- **Actions**: Accelerate, brake, steer
- **Transition Model**: Physics + Environment rules

- **Goal Test**: Reach destination safely
- **Path Cost**: Time, fuel, safety, legality

---

17. STRIPS vs ADL

STRIPS:

- Simple, uses add and delete lists

ADL:

- Advanced, supports:
  - Conditional effects
  - Quantifiers
  - Negative preconditions

---

18. Making an Expert System

Steps:

1. Knowledge acquisition (from experts)
2. Knowledge representation (rules/frames)
3. Inference engine (forward/backward chaining)
4. Explanation module
5. User interface

---

19. Resolution

Definition: Rule of inference used in propositional and first-order logic

Steps:

1. Convert to CNF
2. Apply resolution rule to derive new clauses
3. Repeat until empty clause or no new info

---

20. Sentence to FOL (First-Order Logic)

Example: Sentence: "Every human is mortal." FOL: $\forall x \, (Human(x) \rightarrow Mortal(x))$

Tips:

- Use quantifiers smartly
- Translate negations carefully

---

21. Cryptarithmetic Sum (Non-Carry Example)

Example: SEND + MORE = MONEY Use constraints: unique digits, no carry (simplified for exam)

Apply CSP with backtracking

---

22. Turing Test

Definition: A test to check if a machine can exhibit intelligent behavior indistinguishable from a human

Conducted by: Alan Turing (1950)

Involves: Human judge interacting with unseen machine and human

Criticism: Doesn’t measure understanding, just mimicry

Memory Tips:

- Use acronyms (e.g., PEAS = Performance, Environment, Actuators, Sensors)
- Visualize trees/graphs for search
- Mnemonics for algorithms (e.g., "Best Friends Are Always Informed" for BFS, DFS, A*, IDDFS)

Concept Interconnections:

- Search algorithms → Used in problem-solving
- Heuristics → Guide informed search (e.g., A*)
- CSP → Special kind of problem formulation
- Knowledge Representation → Basis of reasoning in Wumpus World, Expert Systems
- Learning → Builds intelligent agents that adapt

---

1. Constraint Satisfaction Problem (CSP)

✨ Definition:

A **Constraint Satisfaction Problem** consists of:

- A set of **variables**: X = {X1, X2, ..., Xn}
- A set of **domains**: D = {D1, D2, ..., Dn} where each Di is the set of possible values for Xi
- A set of **constraints** that specify allowable combinations of values.

🧲 Example:

Sudoku, Map Coloring, N-Queens, Scheduling

🔹 Key Concepts:

- **Unary constraints**: Affect a single variable (e.g., $X \neq 3$)
- **Binary constraints**: Involve pairs of variables (e.g., $X \neq Y$)
- **Global constraints**: Involve multiple variables (e.g., all-different)

💡 Techniques:

- **Backtracking** (DFS with constraint checks)
- **Forward Checking** (eliminates inconsistent values from future domains)
- **Constraint Propagation** (e.g., Arc Consistency)
- **Heuristics**:
  - MRV (Minimum Remaining Values)
  - Degree Heuristic

- ○ Least Constraining Value

🎯 **Examiner Focus:**

- Efficient backtracking
- Variable & value ordering heuristics
- Arc Consistency (AC-3 Algorithm)

🧠 **Follow-up Qs:**

- Q: Why is CSP preferred in certain search problems?
  - A: It decouples the problem representation from the solution strategy, reducing complexity.

🔗 **Connected Topics:**

- N-Queens
- Search Trees
- Knowledge Representation

---

2. Knowledge-Based Agent

🔎 **Definition:**

A **Knowledge-Based Agent** is an intelligent agent that uses a **knowledge base** to make decisions by logical inference.

🔹 **Components:**

- **Knowledge Base (KB)**: Set of sentences in a formal language
- **Inference Engine**: Derives conclusions from the KB
- **Tell, Ask, Perform** operations

📚 **Types of Knowledge:**

- Declarative
- Procedural
- Meta-knowledge

💡 **Analogy:**

Like Sherlock Holmes reasoning from facts to deduce the truth.

🎯 **Examiner Focus:**

- How KB agents update beliefs
- How inference is used in decision-making

🧠 **Follow-up:**

- Q: How does a KB agent differ from a reflex agent?
    - A: KB agents reason with internal models; reflex agents rely only on condition-action rules.

---

3. Best Informed Search Algorithm

✔ **Answer:** ***A Search****

🔹 **Why?**

- Combines **greedy best-first** and **uniform cost search**
- Uses evaluation function: $f(n) = g(n) + h(n)$
    - $g(n)$: cost so far
    - $h(n)$: estimated cost to goal

✅ **Conditions for Optimality:**

- $h(n)$ must be **admissible** (never overestimates)
- $h(n)$ must be **consistent** (monotonic)

👉 **Practical Use:**

- GPS Navigation, Robot Path Planning

🧠 **Follow-up:**

- Q: What's the drawback of A*?
    - A: High memory usage due to storing all nodes in memory.

---

4. Quantifiers in First-Order Logic (FOL)

📒 **Two Quantifiers:**

- **Universal (∀)**: For all x, P(x)
- **Existential (∃)**: There exists an x such that P(x)

✨ **Example:**

- $\forall x\ Human(x) \rightarrow Mortal(x)$
- $\exists x\ Prime(x) \land x > 10$

🧠 **Follow-up:**

- Q: How do quantifiers affect inference?
  - A: They change the scope and applicability of rules during deduction.

---

5. Applications of AI in Education

🏫 **Use Cases:**

- **Personalized Learning** (e.g., Duolingo, Khan Academy)
- **AI Tutors**: Adaptive feedback
- **Predictive Analytics**: Student performance prediction
- **Automation**: Grading, attendance

💡 **Trend:**

- ChatGPT-like models being integrated into tutoring systems

🧠 **Follow-up:**

- Q: Risks of AI in education?
  - A: Bias in data, lack of human empathy, privacy concerns

---

6. Four Approaches of AI

1. **Thinking Humanly** (Cognitive Modeling)
2. **Acting Humanly** (Turing Test)
3. **Thinking Rationally** (Laws of Thought)
4. **Acting Rationally** (Rational Agent)

🧠 **Follow-up:**

- Q: Which is most practical today?
  - A: Acting rationally (Agent-based systems)

---

7. Search Tree vs Search Graph

Feature	Search Tree	Search Graph
Structure	Tree (no cycles)	Graph (may contain cycles)
Node Expansion	May revisit states	Avoids duplicate states
Memory Use	Higher due to redundancy	Efficient storage

---

8. Alpha-Beta Pruning

🧬 **Definition:**

Optimization for minimax algorithm, eliminates branches that don't influence final decision.

✅ **Example:**

If a MAX node has a better option already explored, no need to explore further branches under MIN.

📈 **Efficiency:**

Best case: Reduces time complexity from O(b^d) to O(b^(d/2))

---

9. Cryptarithmetic Sum (with carry)

📄 **Example:**

```
Unset
  SEND
+ MORE
= MONEY
```

-
  Constraint: Each letter represents a unique digit
- Solve via CSP or backtracking with carry propagation

---

10. Task Environment Properties

1. **Observable**: Fully or partially
2. **Agents**: Single or Multi-agent
3. **Deterministic**: Outcome predictable?
4. **Episodic**: Independent or sequential
5. **Static**: Environment changes during action?
6. **Discrete/Continuous**
7. **Known/Unknown**

---

11. Vacuum Cleaner Problem

💡 **Problem:**

- Two rooms: A and B
- Each can be clean or dirty
- Agent can move left, right, suck

👨‍💻 Agent Type:

- **Goal-Based**
- **Rational**: Chooses actions to achieve goal (clean all rooms)

🖊️ Draw Goal-Based Agent:

Percept ➔ State ➔ Goal ➔ Search ➔ Action

---

12. N-Queen Problem (N=4)

🔹 Formulation:

Place 4 queens on 4x4 board such that none attack each other

✍️ Strategies:

- Backtracking (best)
- CSP
- Min-Conflicts heuristic

---

13. Rational Agent

🚀 Definition:

An agent that selects actions to maximize performance based on percepts and knowledge.

🧠 Follow-up:

- Q: Is a reflex agent rational?
  - A: It can be, if its rules are well-designed for the environment.

---

14. Wumpus World

🔮 Environment:

- 4x4 grid world
- Hazards: Wumpus, pits
- Goal: Find gold and escape

🔹 **Key Features:**

- Partial observability
- Logical reasoning required (KB agent)

---

15. Genetic Algorithms (GA)

See previous detailed explanation. Highlights:

- Mimics evolution
- Uses population, selection, crossover, mutation
- Solves optimization problems

---

16. NLP - All Steps

1. **Lexical Analysis**: Tokenizing text
2. **Syntactic Analysis**: Parsing grammar
3. **Semantic Analysis**: Understanding meaning
4. **Discourse Integration**: Meaning from context
5. **Pragmatic Analysis**: Interpretation in real situations

🧠 **Follow-up:**

- Q: Which step involves Named Entity Recognition?
    - A: Semantic Analysis

Water Jug Problem Heuristic

The Water Jug problem involves finding a solution to measure a specific amount of water using two jugs of different capacities, where you can only fill, empty, or transfer water between the two jugs. The **heuristic** for this problem typically involves considering:

1. **State Representation:** The state is represented by the amount of water in each jug.
2. **Goal State:** The goal is to achieve a specific amount in one of the jugs.
3. **Heuristic Approach:** A possible heuristic could be the **number of steps left** to reach the goal, or **how close** the current state is to the goal (i.e., the difference between the current amount in a jug and the target amount).

AO Algorithm*

AO* (And-Or Search) is used in problem-solving where decisions need to be made in a tree-like structure, involving both **AND** nodes (where all child nodes must be satisfied) and **OR** nodes (where any one child node must be satisfied). AO* is typically used in **game theory** or **planning problems**, like scheduling or route planning.

1. **And nodes:** All child nodes must be solved for the parent node to be considered solved.
2. **Or nodes:** Only one child node needs to be solved for the parent node to be solved.
3. **Q for solving:** It refers to the **evaluation function** that estimates the cost of reaching the goal from the current state, which helps prioritize searches in the AO* search process.

Learning Agent Diagram

A learning agent in AI typically consists of **four main components**:

1. **Performance Element:** Decides what actions to take based on the current environment state.
2. **Critic:** Provides feedback to evaluate the performance.
3. **Learning Element:** Uses the feedback to improve future performance.
4. **Problem Generator:** Suggests new experiences to help the agent learn more about its environment. This cycle helps the agent continually improve through experience.

Expert Systems

An **Expert System** is a software designed to simulate the decision-making ability of a human expert in a specific field. It has two main components:

- **Knowledge Base**: Contains domain-specific knowledge.
- **Inference Engine**: Applies logical rules to the knowledge base to derive conclusions.

Applications: Expert systems are used in fields like medical diagnosis, troubleshooting, and financial forecasting.

Limitations of Expert Systems

1. **Inflexibility**: They can't adapt to new problems or situations not covered in their knowledge base.
2. **Costly**: Building and maintaining an expert system can be expensive.
3. **Dependence on human knowledge**: They can only perform as well as the information given by experts.
4. **Lack of common sense**: They often miss out on common-sense reasoning.

Water Jug Problem - Detailed Explanation

The water jug problem typically involves two jugs with capacities, say 4 liters and 3 liters. The challenge is to measure exactly 2 liters using only these two jugs. Actions available include:

- **Fill a jug** to its maximum capacity.
- **Empty a jug** completely.
- **Transfer water** between jugs until one is full or empty. This problem can be solved using **breadth-first search (BFS)** or **Euclidean algorithms** (if capacities have a greatest common divisor).

Omniscient Agent

An **omniscient agent** is one that knows everything about the environment and its future states. This is a theoretical concept because in practical AI applications, agents can never be truly omniscient. They have to make decisions based on limited information.

Rationality of an Agent

A rational agent is one that acts to **maximize its expected performance** according to some predefined performance measure, based on its current knowledge and perceptions. It chooses actions that **best fulfill its goals** in its environment, even if it doesn't have complete information.

Uniform Cost Search Algorithm

The **Uniform Cost Search (UCS)** is a search algorithm that expands the least costly path first. It is similar to Dijkstra's algorithm but applies to non-negative edge-weighted graphs. The algorithm ensures that the shortest path is always found, but its time complexity is high in larger graphs.

UCS Properties:

- **Complete**: Yes, if there's a solution.
- **Optimal**: Yes, it always finds the least-cost path.
- **Time Complexity**: O(b^d) where b is the branching factor, and d is the depth of the shallowest solution.

Task Environments in AI

A **task environment** defines the context and constraints under which an agent operates. It includes the **set of states**, **actions**, and **performance measure**.

Examples:

- **Medical diagnosis**: An environment where the agent helps doctors in diagnosing diseases.
- **Self-driving cars**: The environment consists of roads, other vehicles, pedestrians, traffic signals, etc.

Reasons: Task environments help define what an agent must do, which impacts its design and learning process.

Write FOL for a Sentence

- **Sentence:** "John is a doctor." **FOL Representation:** Doctor(John)

DFS Path Sequence

Given a tree with nodes labeled from 1 to 6, performing **DFS** starting from node 1 would yield a traversal path like:

- **Path:** 1 → 2 → 4 → 5 → 3 → 6

Cost depends on the problem, but it's often associated with the number of steps or edge weights.

Reinforcement Learning

Reinforcement Learning (RL) is a type of machine learning where an agent learns to make decisions by receiving **rewards** or **punishments** based on its actions in an environment. The goal is to maximize cumulative rewards over time.

Example: Teaching a robot to walk by rewarding it when it takes steps forward and penalizing it for falling.

Backtracking

Backtracking is a search algorithm for solving problems like puzzles, where it tries all possible solutions and **backtracks**as soon as it detects that a path will not lead to a solution (e.g., N-Queens problem).

Heuristic for 8-Puzzle Problem

The **8-Puzzle problem** involves arranging tiles in a 3x3 grid. A common heuristic is the **Manhattan distance**, which is the sum of the horizontal and vertical distances of tiles from their goal positions.

Turing Test

The **Turing Test**, proposed by Alan Turing, measures whether a machine can exhibit **intelligent behavior indistinguishable from that of a human**. It involves a human evaluator who interacts with both a machine and another human, without knowing which is which, and determines if the machine can fool the evaluator.

Can AI Do Everything?

AI can't handle **common sense reasoning** effectively or tasks that require a deep understanding of emotions, abstract reasoning, or ethical judgment. AI is limited by its **data** and the algorithms it uses.

Turing Test Score for ChatGPT

ChatGPT has performed well in Turing-like evaluations, fooling some evaluators into thinking it's a human. However, it still falls short in tasks requiring deep, emotional, or highly context-specific understanding.

Rationality

A rational agent's actions are **goal-directed** and aim at maximizing the expected **performance measure**, given its **knowledge and limitations**. It may act irrationally due to imperfect information or limitations in its decision-making algorithm.

Types of Environments for Medical Diagnosis Application

Medical diagnosis environments are typically **partially observable**, **dynamic**, and **sequential**. The environment may involve incomplete data (e.g., patient history) and requires an agent that makes decisions based on uncertain information.

STRIPS vs ADL

- **STRIPS** (Stanford Research Institute Problem Solver) is a classical planning formalism that uses a **set of actions**with specific preconditions and effects.
- **ADL** (Action Description Language) extends STRIPS by including **more complex representations** like existential and universal quantifiers and allowing for **incomplete information** in actions.

Time and Space Complexities of Search Algorithms

1. **Breadth-First Search (BFS)**:
    - **Time Complexity**: O(b^d), where b is the branching factor and d is the depth of the solution.
    - **Space Complexity**: O(b^d), as it stores all the nodes in the current level at once.
2. **Depth-First Search (DFS)**:
    - **Time Complexity**: O(b^d), where b is the branching factor and d is the depth of the solution.
    - **Space Complexity**: O(d), as it only stores the nodes along the current path.
3. **Uniform Cost Search (UCS)**:

    - **Time Complexity**: O(b^d) for a uniform graph (like BFS).
    - **Space Complexity**: O(b^d), because UCS keeps all nodes in the frontier.
4. **Greedy Best-First Search**:
    - **Time Complexity**: O(b^d), as it evaluates nodes based on heuristics.
    - **Space Complexity**: O(b^d), as it stores all expanded nodes in memory.
5. *A Search**:
    - **Time Complexity**: O(b^d), where d is the depth and b is the branching factor.
    - **Space Complexity**: O(b^d), similar to UCS but with additional storage for the heuristic.
6. **Iterative Deepening Search (IDS)**:
    - **Time Complexity**: O(b^d), as it performs DFS repeatedly for increasing depths.
    - **Space Complexity**: O(d), as only the current depth is stored in memory.

Alpha-Beta Pruning Problem

Alpha-Beta Pruning is a search algorithm that seeks to **eliminate branches** in a minimax tree that will not be explored because they cannot influence the final decision. It improves **minimax search** by cutting off unnecessary exploration.

Problem: In some cases, the alpha-beta pruning may fail to prune the branches optimally if the evaluation function or ordering of moves is poor, leading to **inefficiency**.

Constraint Satisfaction Problem (CSP)

A **CSP** involves finding values for variables under certain constraints. It has the form of:

- **Variables**: A set of variables to be assigned values.
- **Domains**: Each variable has a set of possible values.
- **Constraints**: Rules that restrict the values variables can take.

Example Problem: The **Map Coloring Problem**, where you must assign colors to a map's regions, subject to constraints that adjacent regions must have different colors.

Properties of Task Environments

1. **Accessibility**: Whether the agent has full or partial access to the environment.
2. **Determinism**: Whether the environment is predictable or stochastic.
3. **Static/Dynamic**: Whether the environment changes when the agent is not acting.

4. **Discrete/Continuous**: The granularity of the state space (discrete means finite states; continuous means infinite).
5. **Episodic/Sequential**: Whether the outcome of one action influences subsequent actions.

For an Online Math Tutor:

- **Partially Observable**: The tutor can only observe the student's inputs and responses, not their full thought process.
- **Dynamic**: The tutor adjusts questions based on student responses.
- **Sequential**: The agent's actions (questions) depend on prior interactions.
- **Discrete**: The math problems are discrete, with clear answers.

Vacuum Cleaner Problem Formulation

The **vacuum cleaner problem** can be modeled as an agent that cleans a grid with two locations. It must decide on actions based on its environment: **move left, move right, or clean**.

- **State Space**: The vacuum's position (left or right) and whether the locations are dirty.
- **Actions**: Move, Clean.
- **Goal**: Clean both locations.

Search Tree vs Search Graph

- **Search Tree**: Represents all possible states, even if some states are repeated.
- **Search Graph**: Reuses previously explored states, avoiding redundant paths and reducing memory usage.

Cross + Roads = Danger (Crypting)

This phrase refers to a **simple encryption scheme** using a **substitution cipher** where each symbol is replaced by another. It could symbolize the idea of **cryptography** in AI where secret communication is necessary, like encoding and decoding messages.

Real-Life Example of Expert Systems

An **Expert System** used in **medical diagnosis**: Systems like **MYCIN** help diagnose bacterial infections and recommend antibiotics based on patient symptoms and medical history.

Write FOL for 2 Sentences

- **Sentence 1**: "John is a doctor."
  - **FOL**: Doctor(John)
- **Sentence 2**: "John treats Mary."
  - **FOL**: Treats(John, Mary)

What is Percept Sequence?

A **percept sequence** refers to the sequence of perceptions that an agent receives from its environment. It's a record of everything the agent has sensed over time, which informs its decision-making.

Heuristic and Its Importance

A **heuristic** is a problem-solving strategy that uses **rules of thumb** or **approximations** to make decisions faster. It helps reduce the search space and improve efficiency, though it may not always guarantee an optimal solution.

Importance: Heuristics are critical in real-time problems (e.g., navigation) where finding the optimal solution is too slow.

Difference Between Greedy Best-First Search and Hill Climbing

- **Greedy Best-First Search**: Chooses the path based on the heuristic that seems to lead closest to the goal.
- **Hill Climbing**: A local search algorithm that iteratively moves to the neighbor with the best evaluation, but it can get stuck in local optima.

Parts of NLP

1. **Syntax**: Analyzes the structure of sentences.
2. **Semantics**: Focuses on the meaning of words and sentences.
3. **Pragmatics**: Considers context and real-world usage.
4. **Discourse**: Focuses on how sentences form coherent stories.
5. **Morphology**: Studies word formation.

Disadvantages of Expert Systems

1. **Limited Scope**: Expert systems can only work in predefined domains.
2. **High Maintenance**: They require frequent updates as new knowledge is added.
3. **No Common Sense**: They don't use real-world knowledge, just expert input.

Laws of Thought in AI

The **Laws of Thought** refer to traditional rules of logic, such as **identity, non-contradiction, and the excluded middle**, which form the basis of **logical reasoning** in AI. These laws are used in **knowledge representation**.

Why Study the Wumpus World Problem?

The **Wumpus World** problem is used to teach concepts of **agent design, reasoning, and decision-making** in a partially observable environment. It illustrates key AI concepts like **logical reasoning**, **sensors**, and **action planning**.

Which Agent is Suitable for Google Maps and Why?

A **Model-Based Reflex Agent** is ideal for **Google Maps** because:

- It uses a **map (model)** of the environment and **sensors (real-time data)** to make decisions.
- The agent's decisions are based on **current and past locations**, ensuring optimal pathfinding.

Difference Between DLS and IDLS

- **DLS (Depth-Limited Search)**: Performs a depth-first search up to a specified depth limit.
- **IDLS (Iterative Deepening Depth-First Search)**: Repeatedly performs DLS with increasing depth limits, ensuring completeness and optimality.

Turing Test

The **Turing Test** is a method of determining if a machine can exhibit **intelligent behavior** that is indistinguishable from that of a human. A human evaluator interacts with both a machine and a human without knowing which is which and assesses if the machine can "fool" the evaluator.

1. Genetic Algorithm (GA)

A **Genetic Algorithm (GA)** is an optimization technique inspired by the process of natural selection. It is used to find approximate solutions to optimization and search problems by mimicking the principles of evolution.

Key Concepts:

- **Population**: A set of potential solutions (individuals) to the problem.
- **Chromosomes**: The representation of solutions (usually encoded as strings).
- **Fitness Function**: A function that evaluates how good a solution is.
- **Selection**: Individuals are selected based on their fitness to form a new generation.
- **Crossover**: Two individuals (parents) combine their genetic material to create offspring.
- **Mutation**: Random changes are introduced to an individual's genetic material to maintain diversity.

Steps of GA:

1. **Initialization**: Generate an initial population randomly.
2. **Selection**: Select individuals based on fitness (e.g., roulette wheel, tournament selection).
3. **Crossover**: Perform crossover (recombination) between selected individuals to generate offspring.
4. **Mutation**: Introduce mutations in the offspring to ensure genetic diversity.
5. **Evaluation**: Evaluate the fitness of offspring using a fitness function.
6. **Termination**: Repeat the above steps until the stopping condition is met (e.g., reaching maximum iterations or a desired fitness level).

Applications:

- **Optimization Problems**: Traveling Salesman Problem (TSP), scheduling problems.
- **Machine Learning**: Hyperparameter tuning, feature selection.

---

2. BFS vs. GBFS (Breadth-First Search vs. Greedy Best-First Search)

- **Breadth-First Search (BFS)**:

    - **Search Strategy**: Explores all nodes at the present depth before moving on to nodes at the next level.
    - **Completeness**: Always finds the shortest path in an unweighted graph.
    - **Time Complexity**: O(b^d), where b is the branching factor and d is the depth.
    - **Space Complexity**: O(b^d), as it stores all nodes in the current frontier.
    - **Drawback**: High memory consumption and slow in large graphs.
- **Greedy Best-First Search (GBFS)**:
    - **Search Strategy**: Uses a heuristic function to select the node that seems to be closest to the goal.
    - **Completeness**: Not guaranteed to find the optimal solution.
    - **Time Complexity**: O(b^d) but may be faster than BFS due to heuristic guidance.
    - **Space Complexity**: O(b^d).
    - **Drawback**: Can get stuck in local minima and is not optimal.

---

3. Steps of NLP (Natural Language Processing)

NLP involves processing and understanding human language using computers. The typical steps in NLP are:

1. **Tokenization**: Breaking the text into words, sentences, or phrases.
2. **Part-of-Speech Tagging**: Assigning parts of speech (e.g., noun, verb) to each word.
3. **Named Entity Recognition (NER)**: Identifying entities like names, dates, and locations.
4. **Lemmatization**: Reducing words to their base form (e.g., "running" to "run").
5. **Parsing**: Analyzing the syntactic structure of a sentence.
6. **Semantic Analysis**: Understanding the meaning of words in context.
7. **Text Classification**: Categorizing text into predefined labels (e.g., spam vs. not spam).

Applications: Speech recognition, sentiment analysis, machine translation.

---

4. Steps of Expert Systems (ES)

An **Expert System (ES)** is a computer system that simulates the decision-making ability of a human expert. The steps in building an expert system are:

1. **Knowledge Acquisition**: Collecting knowledge from human experts.
2. **Knowledge Representation**: Structuring the acquired knowledge in a machine-readable form (e.g., rules, frames).
3. **Inference Engine**: Uses logical reasoning to apply the knowledge to solve problems.
4. **Explanation**: Justifies the reasoning process to the user.
5. **Knowledge Base**: The collection of rules, facts, and heuristics used for decision-making.

Applications: Medical diagnosis, troubleshooting systems, legal advice systems.

---

5. Goal-Based Agent vs. Utility-Based Agent

- **Goal-Based Agent**:
  - **Behavior**: Takes actions to achieve specific goals.
  - **Example**: A robot trying to reach a destination.
  - **Rationality**: Achieves goals without considering the cost or satisfaction level.
- **Utility-Based Agent**:
  - **Behavior**: Takes actions to maximize its **utility** function, which quantifies how satisfied it is with the environment.
  - **Example**: A self-driving car optimizing speed and safety.
  - **Rationality**: Chooses actions that maximize its overall utility, not just reaching a goal.

---

6. PEAS (Performance measure, Environment, Actuators, Sensors)

PEAS is a framework for describing an intelligent agent's operation:

1. **Performance Measure**: How the success of an agent’s actions is evaluated.
2. **Environment**: The external context in which the agent operates.
3. **Actuators**: The physical components that execute the agent’s actions.
4. **Sensors**: The input devices that allow the agent to perceive its environment.

---

7. PEAS for Chess

- **Performance Measure**: Win the game, control the center, checkmate the opponent.
- **Environment**: Chessboard, opponent's pieces.
- **Actuators**: Move pieces on the board.
- **Sensors**: Board status (piece positions, legal moves).

---

8. Learning Agent

A **Learning Agent** improves its performance over time through interactions with the environment. It consists of:

1. **Learning Element**: Responsible for making improvements based on feedback.
2. **Performance Element**: The part that takes actions based on the learned information.
3. **Critic**: Evaluates the performance of the agent.
4. **Problem Generator**: Suggests new actions to explore.

---

9. Problem Generator vs. Performance Element

- **Problem Generator**: Suggests new actions and exploration paths to the agent to improve its learning.
- **Performance Element**: Executes the actions suggested by the problem generator or learned behavior.

---

10. Approaches of AI

AI can be approached in multiple ways:

1. **Symbolic AI**: Involves explicit reasoning and knowledge representation (e.g., expert systems).

2. **Machine Learning**: Uses data and algorithms to learn from experience (e.g., neural networks, reinforcement learning).
3. **Connectionist Approach**: Involves networks of simple units (e.g., neural networks).
4. **Evolutionary Algorithms**: Inspired by biological evolution (e.g., genetic algorithms).

---

11. Steps of Bayesian Belief Algorithm

1. **Define the Prior**: The initial belief before any evidence is observed.
2. **Obtain New Evidence**: Gather new data that will influence beliefs.
3. **Apply Bayes' Theorem**: Update beliefs using Bayes' Theorem to calculate the posterior probability.
4. **Compute the Posterior**: The updated belief after considering new evidence.
5. **Decision-Making**: Use the posterior probabilities to make decisions or predictions.

---

12. PEAS for Mars Rover

- **Performance Measure**: Exploration of Mars, avoiding obstacles, collecting data, safe landing.
- **Environment**: Mars surface, terrain.
- **Actuators**: Wheels, cameras, robotic arms.
- **Sensors**: Cameras, temperature sensors, distance sensors.

---

13. What is a Bayesian Belief Network (BBN)?

A **Bayesian Belief Network** is a probabilistic graphical model representing a set of variables and their conditional dependencies via a directed acyclic graph (DAG).

Applications: Medical diagnosis, decision support systems.

---

14. Components of NLP (Not Steps)

1. **Syntax**: Structure of sentences.
2. **Semantics**: Meaning of words and sentences.
3. **Pragmatics**: Context of language use.
4. **Discourse**: Connections between sentences.
5. **Morphology**: Word formation and structure.

---

15. Agent Architecture (Simple Reflex Agent)

A **Simple Reflex Agent** reacts to the current percept using a set of condition-action rules. It doesn't consider the future and may get stuck in loops if not carefully designed.

---

16. Use of Genetic Algorithms

Genetic Algorithms (GAs) are used to solve optimization problems where the solution space is large and complex. They can handle **nonlinear, high-dimensional, and multimodal problems**, such as scheduling, vehicle routing, or evolving neural networks.

---

17. Alpha-Beta Pruning Conditions

Alpha-Beta Pruning improves the **minimax algorithm** by cutting off branches that cannot affect the final decision.

- **Alpha**: The best value found so far for the maximizer.
- **Beta**: The best value found so far for the minimizer.

Pruning occurs when:

- If the current node's score is worse than alpha, prune the branch.
- If the current node's score is better than beta, prune the branch.

---

18. Real-Life Use of Wumpus World

The **Wumpus World** is a simple problem used to teach AI concepts. Its real-life application is in **robotics and autonomous agents**, especially for decision-making in partially observable environments (like Mars rovers or search-and-rescue robots).

---

19. Named Entity Recognition (NER)

NER is the process of identifying and classifying entities in text into predefined categories, such as names of people, organizations, locations, dates, etc.

Example: "Barack Obama was born in Hawaii in 1961."

- **NER output**: Person = "Barack Obama", Location = "Hawaii", Date = "1961".

---

20. Real-Life Example of CSP (Constraint Satisfaction Problem)

A **CSP** can be seen in real life when **scheduling** appointments for doctors. The constraints could be:

- No two appointments can occur at the same time.
- A doctor can only be in one room at a time.
- Patients have preferred time slots.

---

21. A vs AO **Difference****

- **A***: A search algorithm that finds the shortest path from a start node to a goal node by using both **cost to reach the node** and **heuristic** (estimated cost to the goal).
- **AO***: Similar to A*, but works for **AND-OR** graphs (where nodes can represent multiple sub-goals and paths).

---

22. Expert System Diagram

Diagram:

- **Knowledge Base**: Holds the domain-specific knowledge.

- **Inference Engine**: Processes the knowledge to make decisions.
- **User Interface**: Communicates with the user.
- **Explanation System**: Justifies decisions to the user.

---

23. STRIPS vs. ADL

- **STRIPS** (Stanford Research Institute Problem Solver) is a classical planning formalism that focuses on actions and their preconditions/effects.
- **ADL** (Action Description Language) is an extension of STRIPS, which allows for more expressive planning, including the use of **negation** and **quantified variables**.

---

24. Bayesian Network Graph Solving

To solve a **Bayesian Network**, apply **Bayes' Theorem** iteratively, updating beliefs based on evidence. This involves **forward propagation** (from causes to effects) or **inference** to compute posterior probabilities.

---

25. Heuristic Function for Water Jug Problem

The **Water Jug Problem** involves measuring an exact amount of water using two jugs of different sizes. A possible **heuristic function** could be the **absolute difference** between the target amount and the current water volume in the jugs.

---

26. NLP Applications

1. **Machine Translation**: Converting text from one language to another (e.g., Google Translate).
2. **Sentiment Analysis**: Analyzing text to determine the sentiment (positive, negative, neutral).
3. **Speech Recognition**: Converting spoken language into text (e.g., Siri, Alexa).

---

27. BFS and DFS Solving

- **BFS** solves by exploring all nodes at the current depth before going deeper.
- **DFS** explores as far down a branch as possible before backtracking.

---

28. How BFS is Better Than DFS

- **BFS** guarantees the shortest path in an unweighted graph, whereas DFS can get stuck in a deep branch and may miss the optimal solution.
- **BFS** is more memory-intensive than DFS but offers completeness and optimality for shortest-path problems.

---

29. Environment Properties of Medical Diagnosis System

Properties:

- **Dynamic**: Medical conditions can change over time.
- **Stochastic**: Uncertainty due to incomplete data.
- **Continuous**: Medical variables can have continuous values (e.g., blood pressure).
- **Partially Observable**: Not all symptoms or conditions are visible.

---

30. Cricket Scenario - 94 and 7 Runs to Win from 2 Balls

You need to get **6 runs per ball** to reach 100 for both batsmen, which is unlikely unless they hit boundaries (e.g., sixes). Therefore, the scenario could be solved by them trying to hit sixes or perfect shots.

---

31. IDFS (Iterative Deepening Depth-First Search)

IDFS combines the benefits of both DFS and BFS by conducting DFS up to a certain depth limit and then increasing the limit iteratively. It provides a space-efficient solution while ensuring completeness and optimality (similar to BFS)