AI can be applied in various domains, including mundane tasks, formal tasks, and expert tasks. Here's an overview of how AI can be utilized in each of these domains:

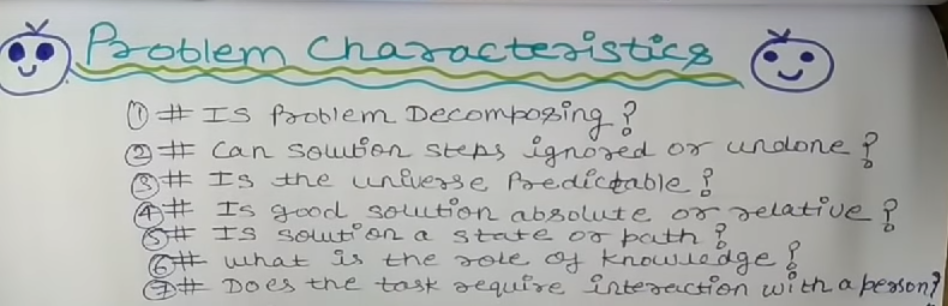
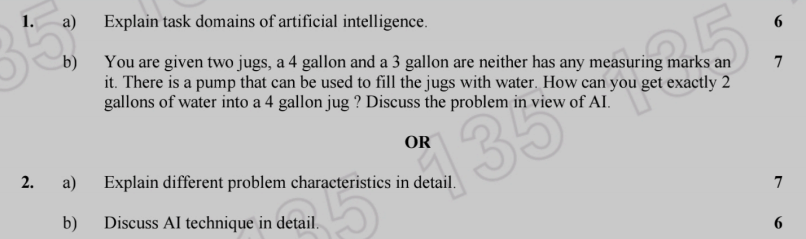
Mundane Tasks:

AI can be employed to automate routine and repetitive tasks, freeing up human resources for more complex and creative work. Mundane tasks include data entry, document processing, email filtering, scheduling, basic customer support, and other administrative tasks. AI-powered systems, such as chatbots or virtual assistants, can handle these tasks efficiently and accurately, minimizing human effort and error.

Formal Tasks:

Formal tasks refer to activities that involve structured and rule-based processes. AI can assist in automating these tasks by analyzing large volumes of data, identifying patterns, and making predictions. For example, in finance, AI algorithms can be used for fraud detection, risk assessment, and algorithmic trading. In legal domains, AI can aid in contract analysis, legal research, and document review. AI can also be utilized in scientific research for data analysis, simulations, and hypothesis testing.

Expert Tasks:

Expert tasks involve complex decision-making and problem-solving that typically require specialized knowledge and expertise. AI can be applied to assist human experts and professionals in these domains. For instance, in healthcare, AI systems can help with medical diagnosis, treatment planning, and personalized medicine. In engineering and design fields, AI can aid in optimization, simulation, and product development. AI-powered systems can also support experts in fields like cybersecurity, marketing strategy, and financial planning by providing insights, recommendations, and predictive analytics.1.Is the problem decomposable into set of sub problems? 2. Can the solution step be ignored or undone? 3. Is the problem universally predictable? 4.Is a good solution to the problem obvious without comparisontoall the possible solutions? 5. Is the desire solution a state of world or a path to a state? 6. Is a large amount of knowledge absolutely required to solvetheproblem? 7. Will the solution of the problem required interaction betweenthecomputer and the person?

**Is the problem decomposable into set of sub problems?** Decomposing a problem into a set of subproblems is a common techniqueinproblem-solving and software development. It involves breaking down a complex problem into smaller, more manageable parts. Whether a problemisdecomposable into subproblems depends on the nature of the problemitself. In many cases, problems can be effectively decomposed into subproblems. This approach allows for a modular and organized development process. Eachsubproblem can be tackled independently, making it easier to understand, analyze, and solve. Once the subproblems are solved, their solutions canbecombined to solve the overall problem.

**Can the solution step be ignored or undone?** the solution steps can be ignored or undone in recoverable, irrecoverable, and ignorable problems.  Recoverable problems: These are problems where the solutionsteps can be undone if they prove to be unwise. For example, inthe8-puzzle problem, if a wrong move is made, it can be undone bymoving the tile back to its original position.  Irrecoverable problems: These are problems where the solutionsteps cannot be undone. For example, in the game of chess, if awrong move is made, it cannot be undone.  Ignorable problems: These are problems where the solutionsteps can be ignored if they prove to be unwise. For example, inthetheorem proving problem, if a lemma is proved, but it is later found to be unnecessary, the lemma can be ignored. Is the problem universally predictable? The predictability of a problem's outcome depends on various factors, includingthe nature of the problem, available information, and the complexity of the system involved. In some cases, problems may have a predictable outcome, whilein others, the outcome may be uncertain. Here are some examples of predictable and uncertain problems: 

Predictable problems:

o The problem of finding the shortest path betweentwopoints in a graph.

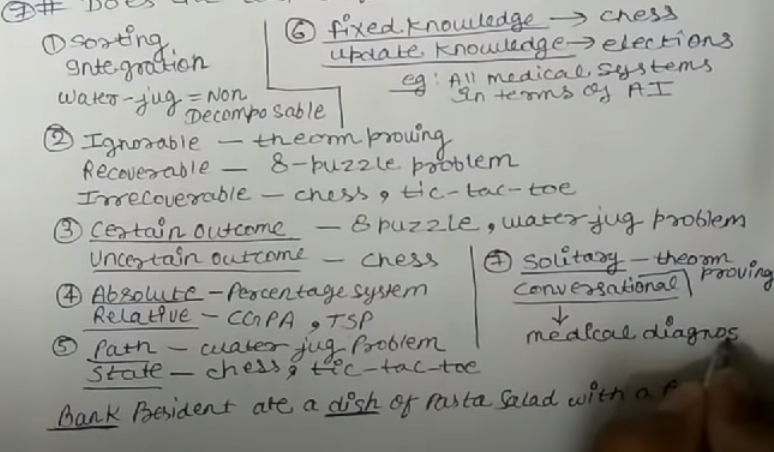
o The problem of sorting a list of numbers.

o The problem of solving a Sudoku puzzle. 

Uncertain problems:

o The problem of predicting the weather.

o The problem of predicting the stock market. o The problem of diagnosing a disease.

**Is a good solution to the problem obvious without comparison toall the possible solutions?** Whether a good solution to a problemis obvious withoutcomparison to all the possible solutions depends on the specific problem. Some problems have a single, obvious solution, whileothers have multiple possible solutions, some of which are betterthan others. Here are some examples of problems where a good solutionmaybeobvious without comparison to all the possible solutions:  Absolute examples: o The problem of finding the shortest path betweentwopoints in a graph. o The problem of sorting a list of numbers in ascendingorder. o The problem of solving a simple math equation.  Relative examples: o The problem of choosing the best route frompoint Atopoint B, given a map of the area. o The problem of choosing the best investment, givenaset of investment options. o The problem of choosing the best meal, given a menuof options. Is the desire solution a state of world or a path to a state? The desired solution can be understood as both a state and a path, depending on the context. Let's explore each concept with an example: Desired Solution as a State: In this interpretation, the desired solutionisseen as a specific end goal or outcome that needs to be achieved. It isafixed state that represents the desired state of the systemor problem. For example, if the problem is to reach a destination, the desiredsolution can be represented as the specific location or endpoint youwant to reach. Once you arrive at that location, you have achievedthedesired solution state. Desired Solution as a Path: In this interpretation, the desired solutionisviewed as a journey or a series of steps and actions that leadtothedesired outcome. It is not just the end state but also the process or pathtaken to reach that state. Using the previous example of reachingadestination, the desired solution as a path would involve planningtheroute, following directions, and making decisions along the waytonavigate towards the desired endpoint. Is a large amount of knowledge absolutely required to solve the problem?The requirement for a large amount of knowledge to solve a problemcanvary depending on the nature of the problem itself. Let's explore two scenarios: fixed knowledge and updated knowledge, each with five examples: Fixed Knowledge: In some cases, a problem may rely on well-establishedprinciples, theories, or concepts where the necessary knowledge is considered fixed and does not change significantly over time. Here are fiveexamples: a. Solving a mathematical equation: Solving an equation requires knowledgeof mathematical formulas and principles that are considered fixed anduniversally applicable. b. Building a basic website: Developing a simple website may only requireknowledge of HTML, CSS, and basic web design principles, which havewell-defined practices and standards. c. Repairing a bicycle: Fixing a bicycle may involve knowledge of the different bike components, tools required, and standard repair procedures, which can be learned and applied without frequent changes. Updated Knowledge: Other problems may require up-to-date or evolvingknowledge, where keeping up with the latest informationandadvancements is crucial for finding a solution. Here are five examples: a. Medical diagnosis: Accurate medical diagnosis often requiresknowledge of the latest research, medical guidelines, and diagnostic toolsto identify and treat conditions effectively. b. Cybersecurity: Addressing evolving cybersecurity threats requiresstaying updated on the latest vulnerabilities, attack vectors, and defensemechanisms to implement effective security measures. c. Climate change mitigation: Developing solutions to mitigate climatechange necessitates understanding current scientific knowledge, technological advancements, and policy frameworks in this rapidlyevolving field. Will the solution of the problem required interaction betweenthecomputer and the person? The requirement for interaction between a computer and a person tosolvea problem depends on the nature of the problem itself. Let's exploretwoscenarios: solitary task and conversation task. Solitary Task: In some problems, a solution can be achieved throughasolitary task where the computer performs computations or processes datawithout requiring direct interaction with a person. Here are a fewexamples: a. Sorting a list of numbers: A computer can performthe sortingalgorithm on its own without requiring interaction with a person. b. Running a simulation: If a problem involves running a simulationor mathematical model, the computer can execute the simulation basedonthe provided inputs and generate results without direct human interaction. c. Generating an automated report: If the problem involves generatingautomated reports based on predefined criteria or data inputs, thecomputer can generate the reports without requiring person-to-computer interaction. Conversation Task: In other problems, interaction between the computer and a person through conversation or interaction is essential to solvetheproblem effectively. This is particularly true for tasks that requireunderstanding natural language, context, or user preferences. Here areafew examples: a. Virtual assistant: Interacting with a virtual assistant like Siri or Alexaoften involves conversation or dialogue to provide instructions, askquestions, or receive responses. b. Customer support chatbot: Chatbots are designed to engageinconversations with users to provide support, answer queries, and resolveissues. c. Recommender systems: Systems that provide personalizedrecommendations, such as movie recommendations or product suggestions, often involve interaction with users to understandtheir preferences through conversations or feedbackTo solve the given problem using AI techniques, let's break it down step by step:

Initial State:

The 4-gallon jug is empty (0 gallons).

The 3-gallon jug is empty (0 gallons).

Goal State:

We want to have exactly 2 gallons of water in the 4-gallon jug.

Actions:

Fill the 4-gallon jug from the pump.

Fill the 3-gallon jug from the pump.

Empty the contents of the 4-gallon jug.

Empty the contents of the 3-gallon jug.

Pour water from one jug to another until either the receiving jug is full or the pouring jug is empty.

State Transition Rules:

Filling a jug adds the corresponding capacity of water to it.

Emptying a jug removes all the water from it.

Pouring water from one jug to another subtracts the amount poured from the pouring jug and adds it to the receiving jug, as long as the receiving jug is not full or the pouring jug is not empty.

Problem Solving:

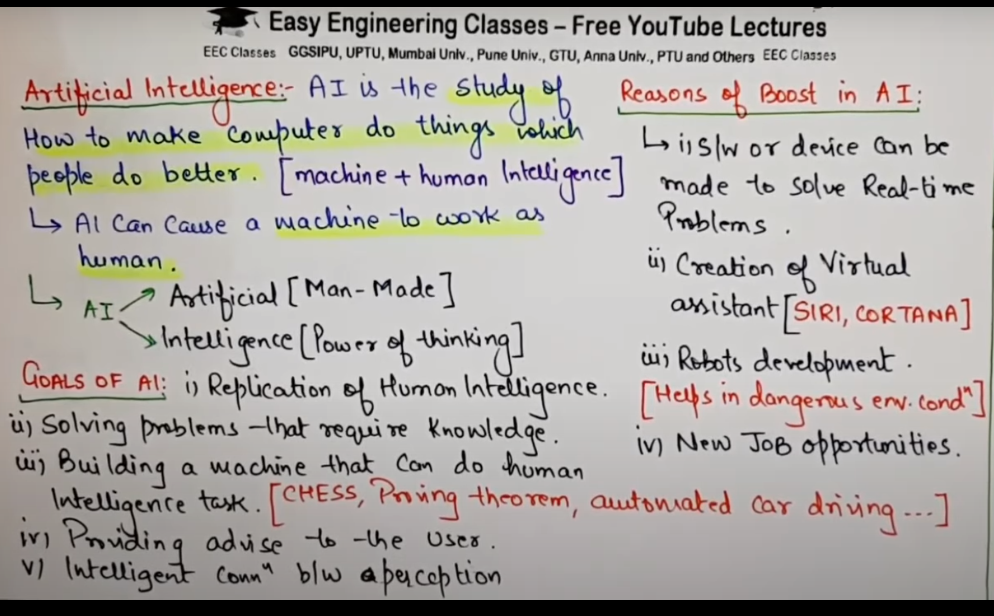
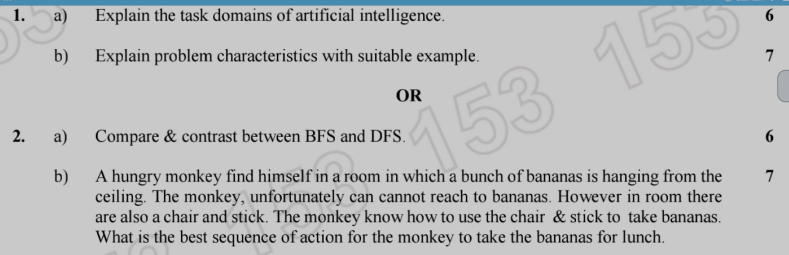
Using the above actions and state transition rules, we can follow these steps to achieve the goal of having exactly 2 gallons of water in the 4-gallon jug:

Fill the 3-gallon jug from the pump. (3 gallons in the 3-gallon jug)

Pour the water from the 3-gallon jug into the 4-gallon jug. (3 gallons in the 4-gallon jug, 0 gallons in the 3-gallon jug)

Fill the 3-gallon jug from the pump again. (3 gallons in the 3-gallon jug)

Pour water from the 3-gallon jug into the 4-gallon jug until the 4-gallon jug is full. (4 gallons in the 4-gallon jug, 2 gallons in the 3-gallon jug)

Now, the 4-gallon jug contains exactly 2 gallons of water.Various applications of AI span across different domains and industries. Here are some notable examples:

Natural Language Processing (NLP): NLP focuses on enabling machines to understand, interpret, and generate human language. Applications include language translation, sentiment analysis, chatbots, virtual assistants, and voice recognition.

Machine Learning (ML): ML involves training algorithms to learn patterns from data and make predictions or decisions without being explicitly programmed. Applications include image recognition, speech recognition, recommendation systems, fraud detection, and autonomous vehicles.

Computer Vision: Computer vision enables machines to analyze and interpret visual data, such as images and videos. Applications include object detection and recognition, facial recognition, gesture recognition, autonomous drones, and surveillance systems.

Robotics: AI is extensively used in robotics to develop intelligent machines capable of performing tasks in various industries. Applications range from industrial automation and manufacturing to healthcare and exploration.

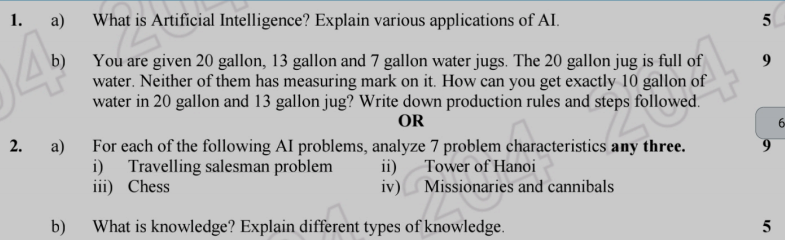
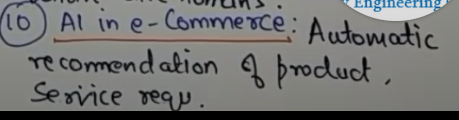
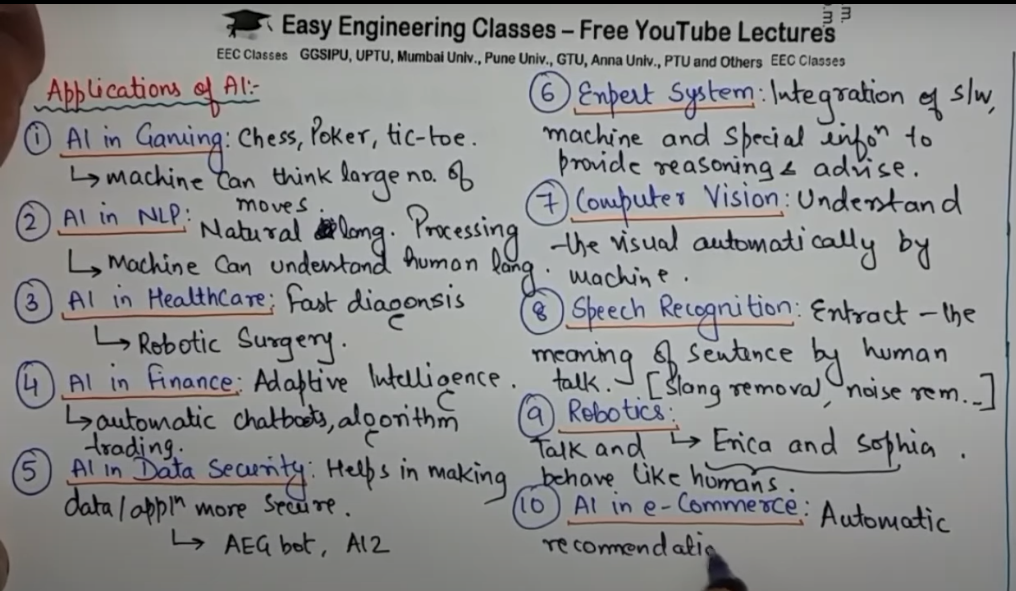
Healthcare: AI is transforming healthcare by aiding in disease diagnosis, medical imaging analysis, drug discovery, personalized medicine, patient monitoring, and virtual healthcare assistants.

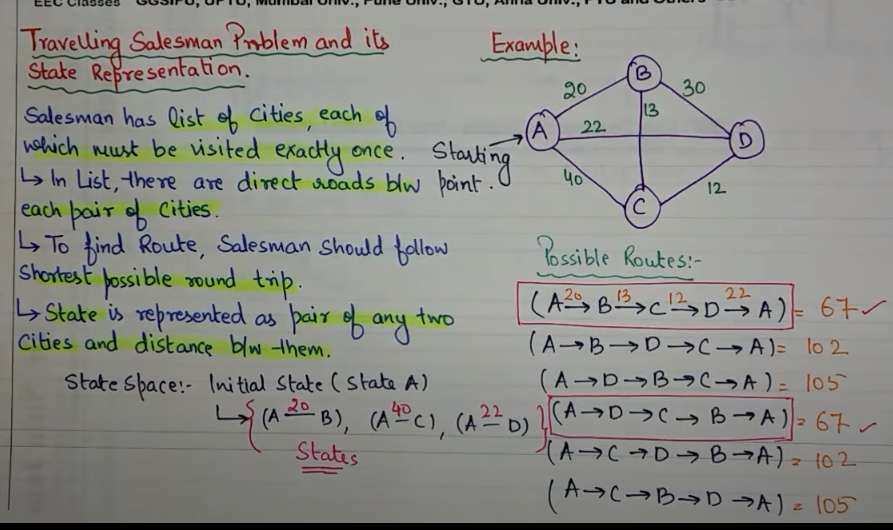
Finance: AI is used for algorithmic trading, fraud detection, risk assessment, credit scoring, customer service, and financial forecasting.

Smart Homes and Internet of Things (IoT): AI-powered systems and devices enable automation, energy management, security, and personalized experiences in smart homes and IoT applications.

Gaming and Entertainment: AI is employed in game development for creating intelligent non-player characters (NPCs), adaptive gameplay, procedural content generation, and realistic simulations.

Virtual Reality (VR) and Augmented Reality (AR): AI enhances VR and AR experiences through natural interaction, object recognition, and scene understanding.

Cybersecurity: AI assists in identifying and responding to cybersecurity threats, anomaly detection, network security, and user behavior analysis.

i) Travelling Salesman Problem:

**State Space**: The state space consists of all possible combinations of cities that the salesman can visit, with each state representing a particular order or sequence of cities.

**Initial State**: The initial state can be any city from which the salesman starts the journey.

**Goal State**: The goal state is to find the shortest possible route that visits each city exactly once and returns to the starting city.

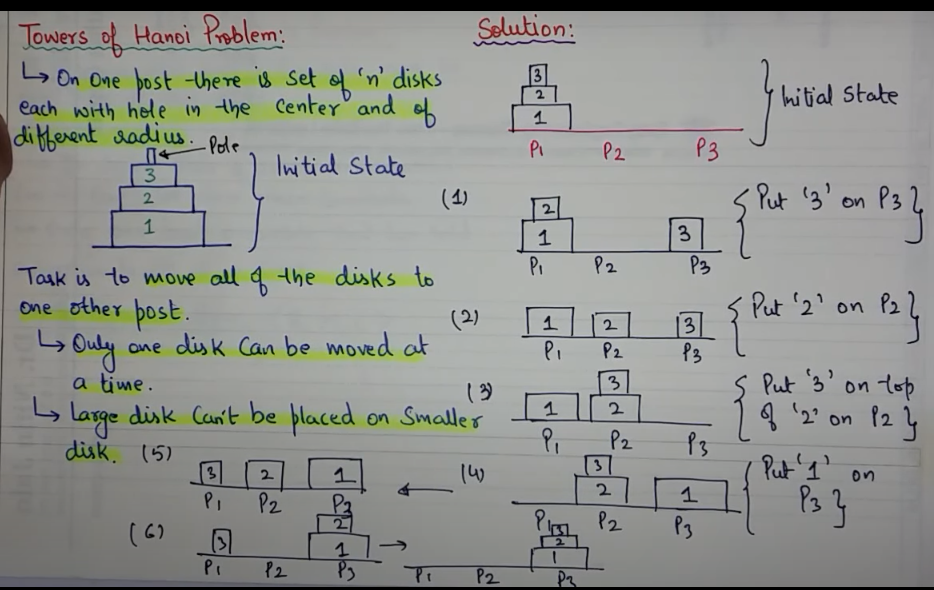
**Actions**: The actions involve selecting a city from the remaining unvisited cities and adding it to the current route.

**Transition Model**: The transition model determines the next state by adding the selected city to the current route and marking it as visited.

**Path Cost**: The path cost is the total distance or cost associated with the current route.

**Constraints**: The main constraint is that each city can be visited only once in the final solution.

ii) Tower of Hanoi:



**State Space:** The state space consists of the arrangement of disks on the three pegs, with each state representing a different configuration.

**Initial State**: The initial state involves all the disks stacked in ascending order of size on one peg, typically the leftmost peg.

**Goal State:** The goal state is to transfer all the disks from the initial peg to another peg, following the rules of the Tower of Hanoi puzzle.

**Actions**: The actions involve moving a single disk from one peg to another peg following the rules (moving one disk at a time and never placing a larger disk on top of a smaller disk).

**Transition Model**: The transition model determines the next state by moving a disk from one peg to another according to the chosen action.

**Path Cost**: The path cost is typically not considered in the Tower of Hanoi problem as the focus is on finding a valid sequence of moves rather than optimizing any cost metric.

**Constraints**: The main constraint is to follow the rules of the Tower of Hanoi puzzle, which prohibits placing a larger disk on top of a smaller disk.

iii) Chess:

**State Space**: The state space consists of the arrangement of chess pieces on the board, with each state representing a different configuration.

**Initial State:** The initial state represents the starting position of the chess pieces at the beginning of a game.

**Goal State:** The goal state depends on the specific chess problem being considered, such as checkmate, stalemate, or a specific position to achieve.

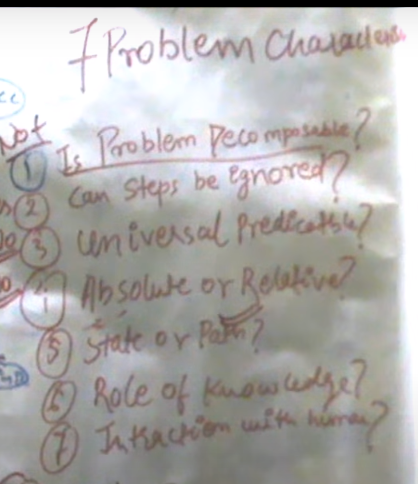
**Actions**: The actions involve legal moves that each chess piece can make, such as moving, capturing, castling, promoting a pawn, etc.

Transition Model: The transition model determines the next state by applying a valid move to the current state.

**Path Cost**: The path cost is typically not considered in chess problems, as the focus is on finding the best sequence of moves rather than optimizing any cost metric.

**Constraints**: The constraints include the rules and limitations of chess, such as the movement capabilities of each piece, the prohibition of moving through occupied squares, and avoiding moves that put the king in check.

iv) Missionaries and Cannibals:



**State Space**: The state space consists of the number and arrangement of missionaries, cannibals, and the boat on each side of the river, with each state representing a different configuration.

**Initial State:** The initial state represents the starting arrangement of missionaries, cannibals, and the boat on one side of the river.

**Goal State:** The goal state is to transfer all the missionaries and cannibals to the other side of the river while ensuring that the cannibals do not outnumber the missionaries on either side at any time.

Actions: The actions involve moving a certain number of missionaries, cannibals, or both from one side of the river to the other side.

**Transition Model:** The transition model determines the next state by applying a valid move to the current state, considering the constraints of not having more cannibals than missionaries on any side.

**Path Cost**: The path cost can be defined as the number of moves or actions taken to reach a particular state.

**Constraints**: The main constraints are to ensure that the number of cannibals does not exceed the number of missionaries on any side of the river, both during and after each move.

To obtain exactly 10 gallons of water in the 20-gallon and 13-gallon jugs using the given jugs, you can follow these production rules and steps:

Production Rules:

Fill a jug completely.

Empty a jug completely.

Pour the contents of one jug into another until either the receiving jug is full or the pouring jug is empty.

Steps:

Start with the 20-gallon jug full of water and the other two jugs (13-gallon and 7-gallon) empty.

Pour water from the 20-gallon jug into the 13-gallon jug until the 13-gallon jug is full. This step uses the production rule 3.

Current state: 20-gallon jug has 7 gallons, 13-gallon jug has 13 gallons, and 7-gallon jug is empty.

Empty the 13-gallon jug completely. This step uses the production rule 2.

Current state: 20-gallon jug has 7 gallons, 13-gallon jug is empty, and 7-gallon jug is still empty.

Pour the water from the 7-gallon jug into the 13-gallon jug. This step uses the production rule 3.

Current state: 20-gallon jug has 7 gallons, 13-gallon jug has 7 gallons, and 7-gallon jug is empty.

Fill the 7-gallon jug from the 20-gallon jug. This step uses the production rule 1.

Current state: 20-gallon jug has 0 gallons, 13-gallon jug has 7 gallons, and 7-gallon jug has 7 gallons.

Pour the 7 gallons from the 7-gallon jug into the 13-gallon jug until the 13-gallon jug is full.

Current state: 20-gallon jug has 0 gallons, 13-gallon jug has 13 gallons, and 7-gallon jug has 1 gallon.

Now, you have 10 gallons of water in the 13-gallon jug. Transfer this 10 gallons of water to the empty 20-gallon jug.

Current state: 20-gallon jug has 10 gallons, 13-gallon jug has 3 gallons, and 7-gallon jug has 1 gallon.

