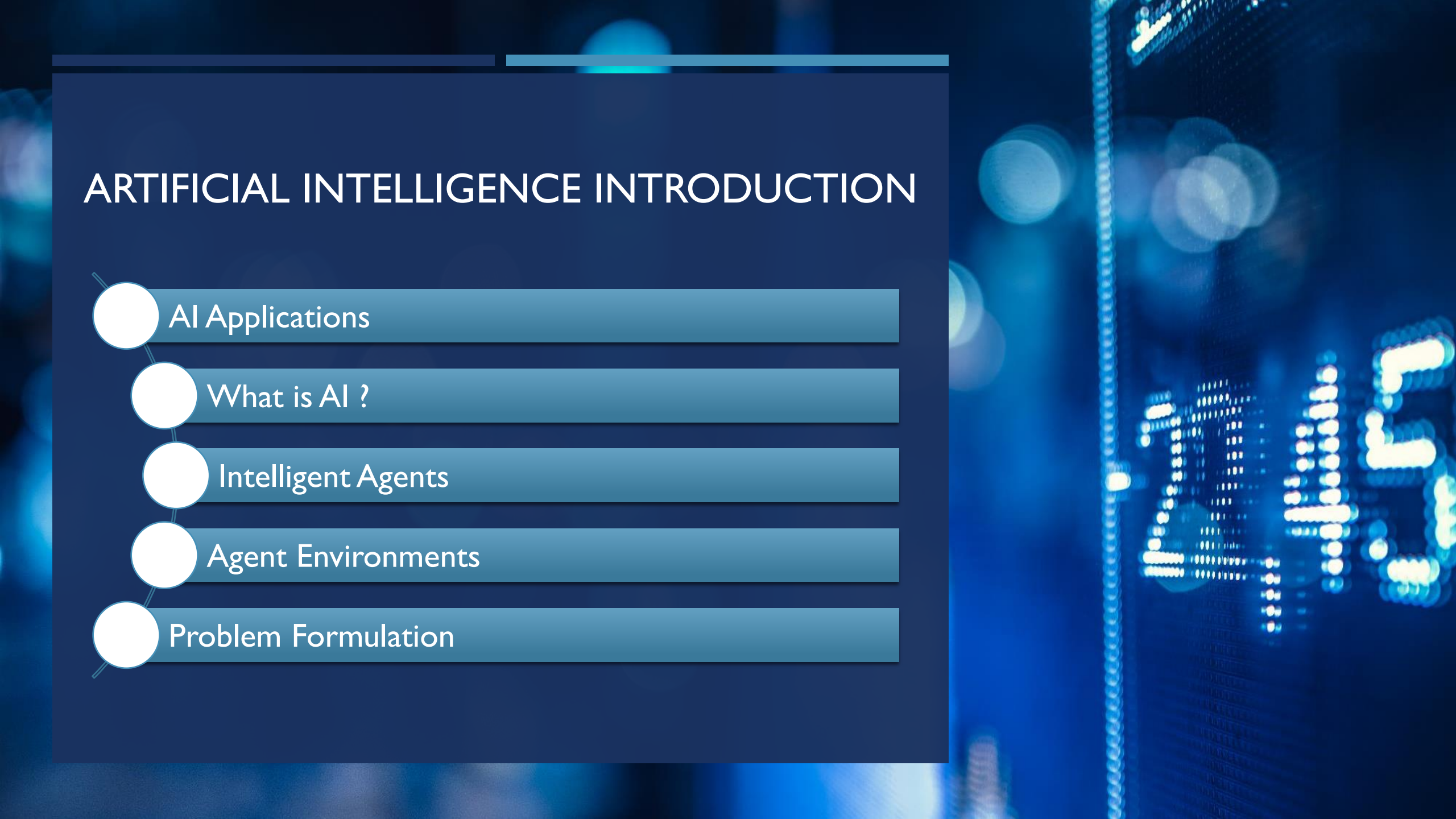
ARTIFICIAL INTELLIGENCE

PREPARED BY

-ANOOJA JOY

ARTIFICIAL INTELLIGENCE INTRODUCTION

- AI Applications
- What is AI ?
- Intelligent Agents
- Agent Environments
- Problem Formulation



COURSE OBJECTIVES

To introduce AI and key paradigms of AI

Understand core techniques and algorithms of AI: Deep Learning, Machine Learning, Natural Language Processing, Reinforcement Learning, Q-learning, Intelligent Agents, Various Search Algorithms

Understand the basic of knowledge presentation

Gain knowledge of problem solving techniques

How to build an intelligent system

TEXT BOOKS

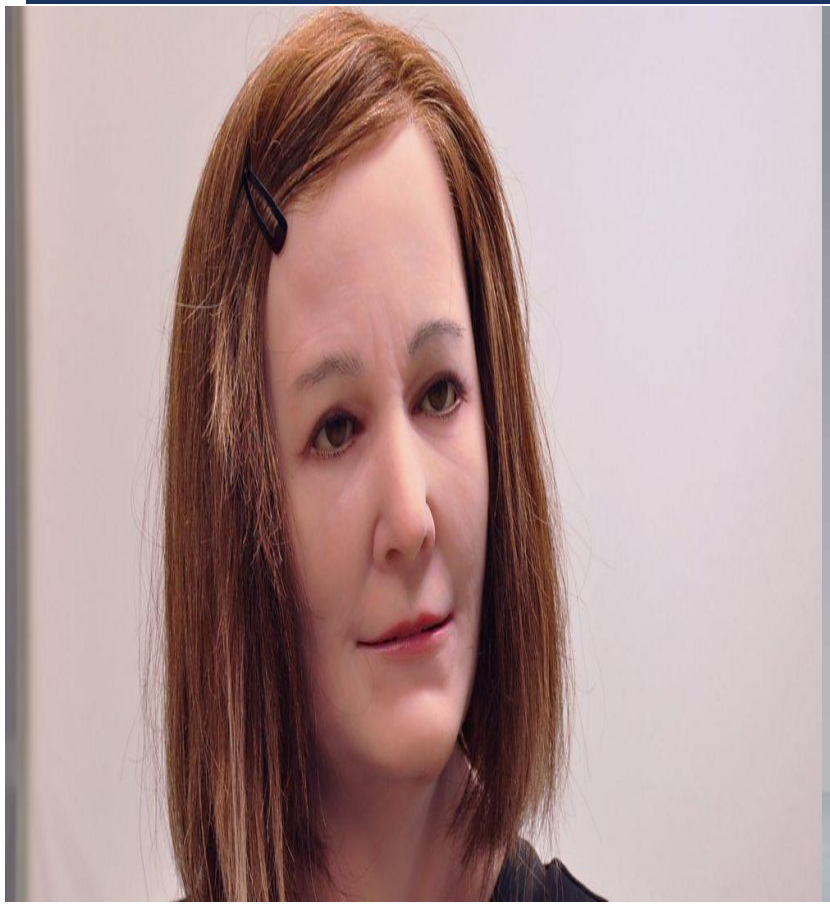
1. Artificial Intelligence A modern Approach, 3rd edition, Stuart Russel and Peter Norwig
2. *Artificial intelligence : structures and strategies for complex problem,* 6th Edition, Luger, George F, Pearson Education, 2009
3. *Master Machine Learning Algorithms*, Edition, v1.12, Jason Brownlee, eBook, 2017
4. *Artificial Intelligence,* Patrick H. Winston, 3rd edition, Pearson Education, 1992

AI APPLICATIONS

A GLIMPSE INTO THE FUTURE

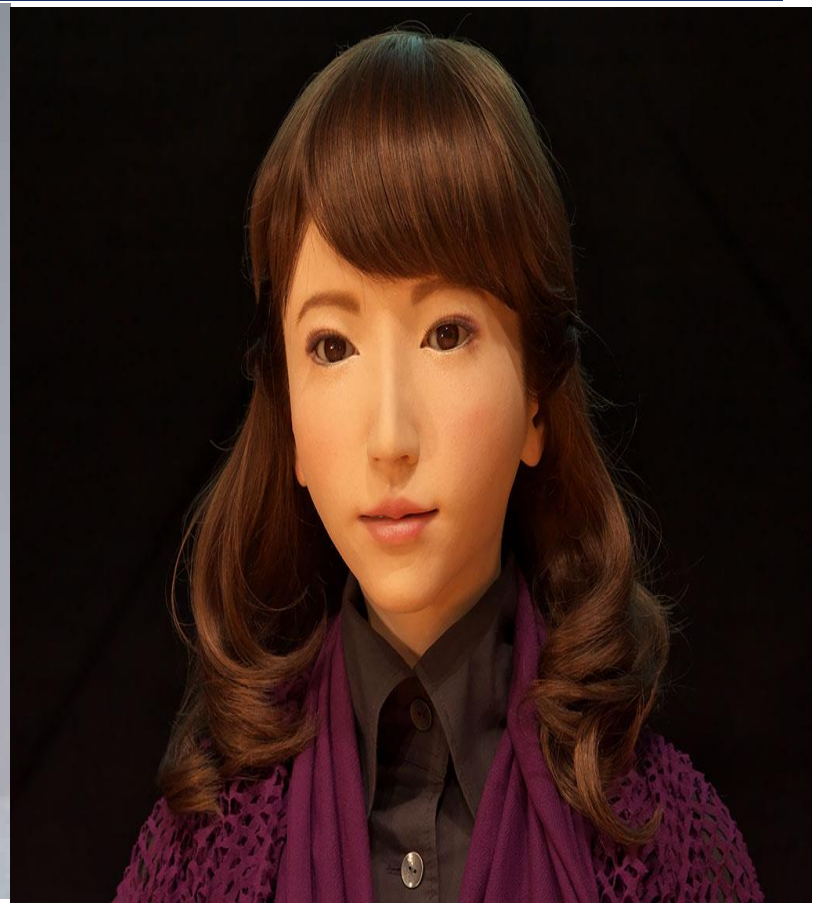
SOPHIA

HUMANOID ROBOTS



NADINE

ASIMO



ERICA

NON-HUMANOID ROBOTS

PARO

AIBO

MIRO

IS AI ONLY ABOUT ROBOTS?

- **HDFC Bank has developed an AI-based chatbot called EVA (Electronic Virtual Assistant), built by Bengaluru-based Senseforth AI Research.**
- **Apple's personal assistant, Siri.**
- **Amazon’s Alexa**
- **Tesla Vehicles**
- **Google Maps**
- **Ride Sharing apps like Uber, Lyft**
- **Jarvis by Facebook**
- **Photo recognition apps**

SOME DOMAINS WHERE AI IS INFUSED WITH

- **Steering a driver-less car, AI autopilots for commercial flights**
- **Spam Filters**
- **Beating Gary Kasparov in a chess match**
- **Understanding language**
- **Healthcare: Robotic assistants in surgery**
- **Automated customer support: chatbots**
- **Personalized shopping experience**
- **Finance: Stockbrkerage prediction**
- **Travel and navigation**
- **Social Media**
- **Smart home devices**
- **Creative arts: watson BEAT**
- **Security and surveillance**

INTRODUCTION TO AI

HOW TO DEFINE AI

WHAT IS ARTIFICIAL INTELLIGENCE?

- AI a term coined by McCortey
- **What is intelligence?**
- **Are humans intelligent?**
- ***"Can a machine think and behave like humans do?***

•**Artificial Intelligence** is a branch of computer science by which we can create intelligent machines which can behave like a human, think like humans, and able to make decisions

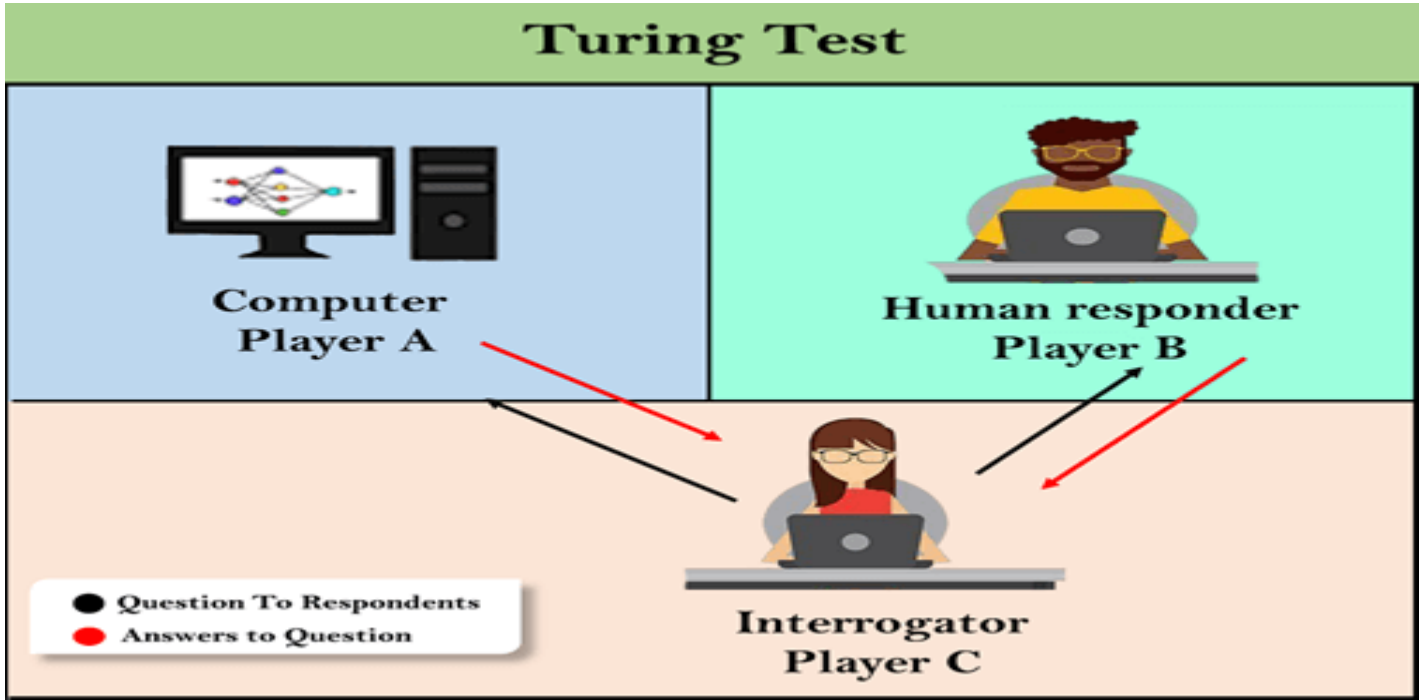
•Artificial Intelligence is the study of systems that

Think Humanly	**Think Rationally**
Acting/behaving Humanly	**Acting/behaving Rationally**

ACTING HUMANLY: TURING TEST

- Human beings are intelligent
- To be called intelligent, a machine must produce responses that are indistinguishable from those of a human

Alan Turing



- If interrogator cannot reliably distinguish human from computer, then computer possess intelligence.

AI TECHNIQUES: REQUIRED CAPABILITIES TO BEHAVE HUMANLY(TURING TEST)

Natural Language Processing: Enabling the computer to communicate successfully in English

Knowledge Presentation: Storing what the computer knows or hear

Automated Reasoning: Using the stored information to answer questions and to draw new conclusions

Machine Learning: Adapting to new circumstances and to detect and extrapolate patterns

Computer Vision: To perceive objects

Robotics: To manipulate objects and move about.

COGNITIVE MODELING APPROACH

To get inside the actual workings of human minds

Three ways to get inside the mind:

Introspection: Trying to catch our own thoughts as they go by	**Psychological Experiments:** observing a person in action	**Brain Imaging:** observing the brain in action

The interdisciplinary field of cognitive science brings together computer models from AI and experimental techniques from psychology to construct precise and testable theories of the human mind.

AI PROBLEMS


PLAYING CHESS
PROVING MATHEMATICAL THEOREMS
WRITING POETRY
DRIVING A CAR ON A CROWDED STREET
DIAGNOSING DISEASES

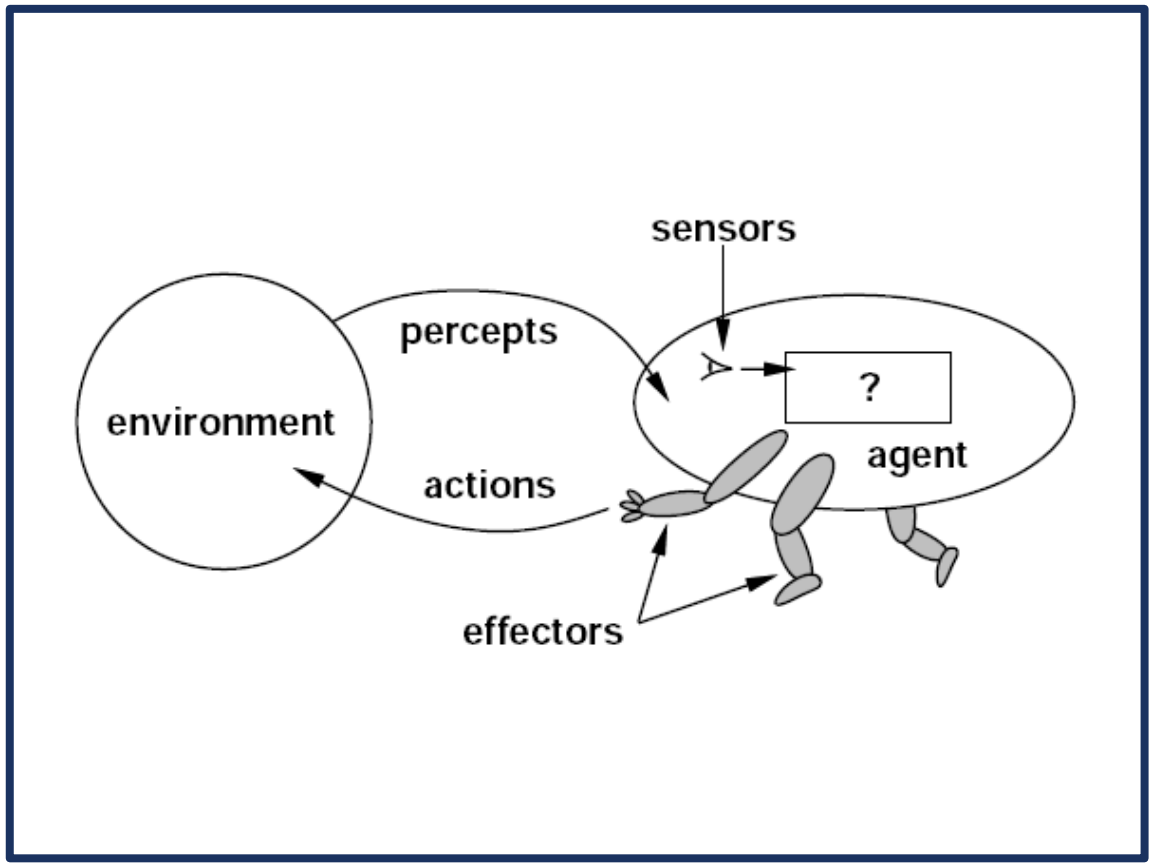
AI AGENTS

AGENTS

Agent = Architecture + Program

- An **agent** is just something that acts.
- Agents can be classified according to the environment in which they are like Intelligent Agents, Human Agent, Robotic agent, Biological Agents, Software/computer Agents, Hardware Agents, Interface Agents, Mobile Agents, Reactive Agents, Information Agents, Distributed Artificial Intelligence Agents …
- Computer Agents does:
  - operate autonomously
  - perceive their environment
  - persist over a prolonged time period
  - adapt to change
  - create and pursue goals

AI AGENTS



Agent-->perceives-->decides--> Acts

An agent is anything that can be viewed as perceiving its environment through sensors and acting upon that environment through actuators - *(Artificial Intelligence: A Modern Approach by Stuart Russell and Peter Norvig)*

Intelligent Agent must

- Must **sense/percept(current & memory)**
- Must **act/react with high performance and optimized result**
- Must **autonomous** (to some extend)
- Must take **rational action**

AI AGENT COMPONENTS

Sensors: An agent perceives its environment through sensors. Eg: cameras, infrared range finders

Actuators: Agent can change the environment through actuators/ effectors **Eg:** Motors

Percept: The complete set of inputs at a given time for an agent is called a percept

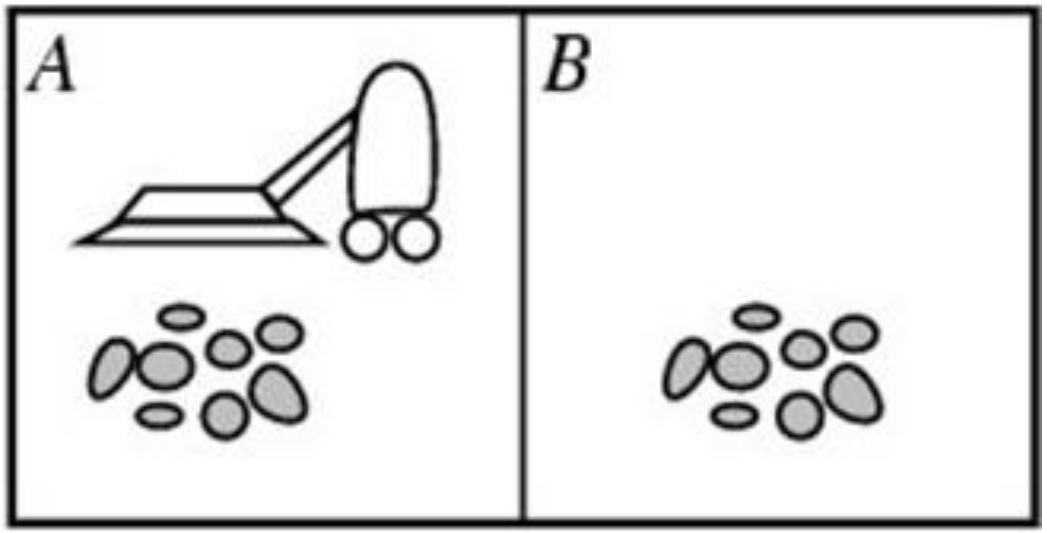
Percept sequence: The complete history of everything the agent has perceived

Agent function(agent's behaviour) maps from percept histories to actions: [f: p* --> A]

Agent program: runs on the physical architecture to produce f. It is a concrete implementation

Action: An operation involving an actuator is called an action

VACUUM-CLEANER WORLD

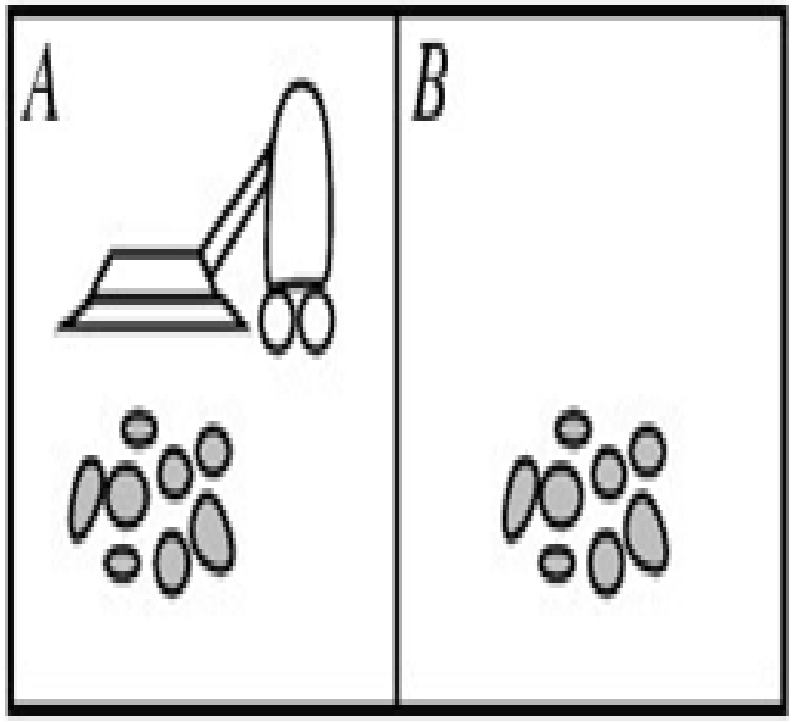


- **Percepts:** location and contents,

  e.g., [A, dirty]
- **Actions:** Left, Right, Suck, NoOp
- **Agents Function:** *look-up table*

A SIMPLE AGENT FUNCTION

Percept sequence	Action
[A, Clean]	Right
[A, Dirty]	Suck
[B, Clean]	Left
[B, Dirty]	Suck
[A, Clean], [A, Clean]	Right
[A, Clean], [A, Dirty]	Suck
…	
[A, Clean], [A, Clean], [A, Clean]	Right
[A, Clean], [A, Clean], [A, Dirty]	Suck
…	



PROPERTIES OF AGENT

- Rationality
- Autonomy
- Reactivity

RATIONALITY

An agent should "do the right thing", based on what it can perceive and the actions it can perform. The right action is the one that will cause the agent to be most successful

Definition Rationality: For each possible percept sequence, a rational agent should select an action that is expected to maximize its performance measure, given the evidence provided by the percept sequence and whatever built-in knowledge the agent has.

RATIONALITY

Rational is different from omniscience

Percepts may not supply all relevant information. **We can behave rationally even when faced with incomplete information.**

E.g., in card game, don't know cards of others.

Rational is different from being perfect

Rationality maximizes expected outcome while perfection maximizes actual outcome.

DEFINITION: The **autonomy** of an agent is the extent to which its behavior is determined by its own experience, rather than knowledge of designer.

- Agents can perform actions in order to modify future percepts so as to obtain useful information: information gathering, exploration.

Extremes of rationality

- No autonomy – ignores environment/data
- Complete autonomy – must act randomly/no program

Example: baby learning to crawl

Ideal Rationalism

- Idesign agents to have some autonomy
- Possibly become more autonomous with experience

AUTONOMY

PEAS ANALYSIS

How to design agent? Must first specify the setting for intelligent agent design

Agents can be described by their PEAS.

A rational agent maximizes the performance measure for their PEAS

PEAS

Specifying the task environment is always the first step in designing agent

PEAS:

Performance
Environment
Actuators
Sensors

PERFORMANCE MEASURES

Performance measure: An *objective criterion for success* of an agent's behavior. The performance measure depends on the agent function.

Performance measures of a vacuum-cleaner agent: amount of dirt cleaned up, amount of time taken, amount of electricity consumed, level of noise generated, etc.

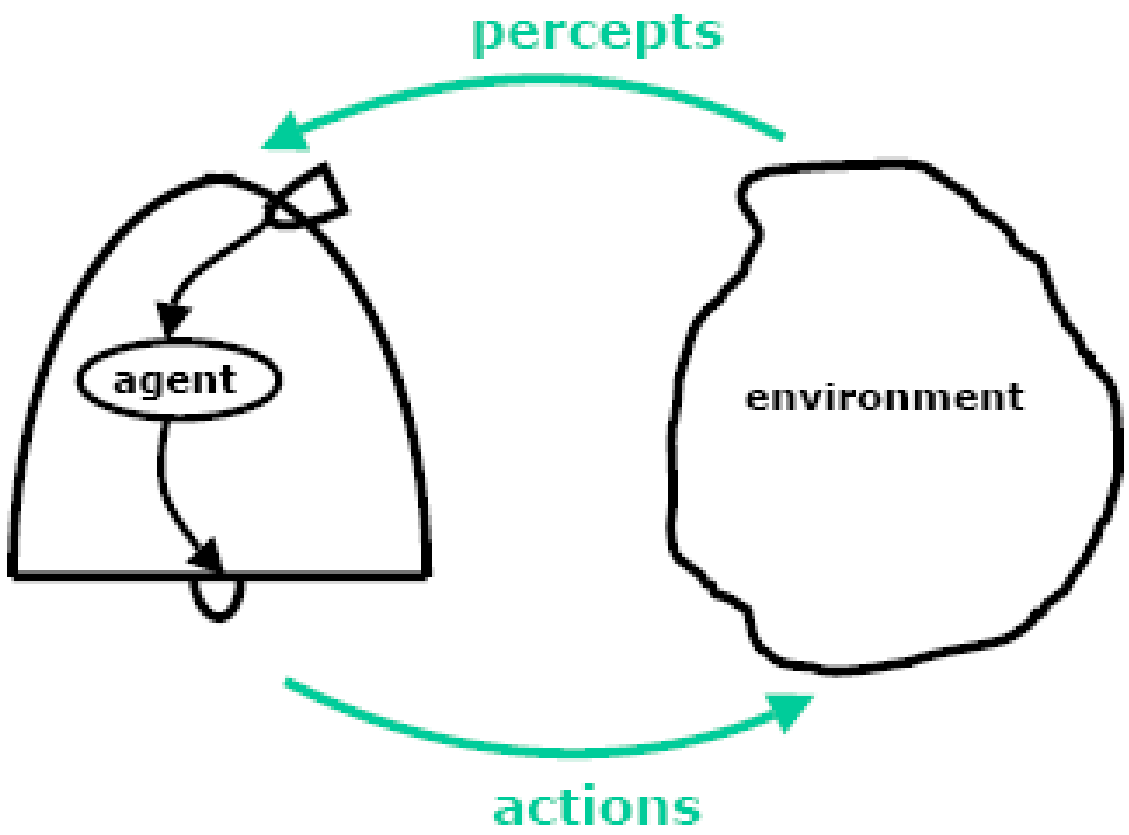
Performance measures self-driving car: time to reach destination (minimize), safety, predictability of behavior for other agents, reliability, etc.

Performance measure of game-playing agent: win/loss percentage (maximize), robustness, unpredictability (to "confuse" opponent), etc.

THE ENVIRONMENT

- What all do we need to specify?
  - The action space
  - The percept space

The environment is a string of mappings from the action space to the percept space



TAXI DRIVER EXAMPLE

Performance Measure	Environment	Actuators	Sensors
safe, fast, legal, comfortable trip, maximize profits	roads, other traffic, pedestrians, customers	steering, accelerator, brake, signal, horn, display	camera, sonar, speedometer, GPS, odometer, engine sensors, keyboard, accelerator

MEDICAL DIAGNOSIS SYSTEM

Performance Measure	Environment	Actuators	Sensors
Healthy patient, minimize costs, lawsuits	Patient, hospital, staff	Screen display (questions, tests, diagnoses, treatments, referrals)	Keyboard (entry of symptoms, findings, patient's answers)

PART-PICKING ROBOT

Performance Measure	Environment	Actuators	Sensors
Percentage of parts in correct bins	Conveyor belt with parts, bins	Jointed arm and hand	Camera, joint angle sensors

INTERACTIVE ENGLISH TUTOR

Performance Measure	Environment	Actuators	Sensors
Maximize student's score on test	Set of students	Screen display (exercises, suggestions, corrections)	Keyboard

AI AGENT ENVIRONMENT

WORLD OUTSIDE OR THAT SURROUNDS AGENT

AGENT ENVIRONMENT TYPES

- Fully Observable(vs. partially observable)
- Deterministic(vs. stochastic)
- Episodic (vs. sequential)
- Static Environment (vs. dynamic)
- Discrete(vs.continuous)
- Single Agent(vs. multiagent)

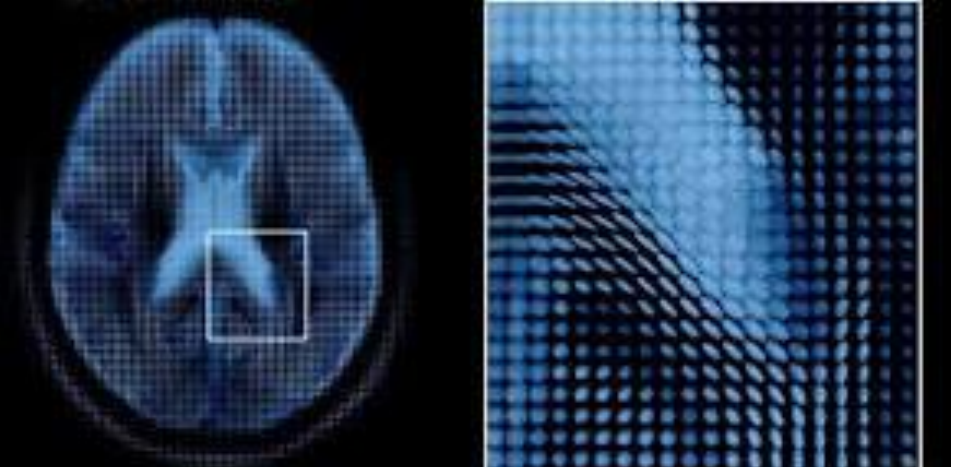
BASED ON OBSERVABILITY

- Is everything an agent requires to choose its actions available to it via its sensors? Perfect or Full information.
- **Fully observable :** An agent's sensors give it access to the complete state of the environment at each point in time. An accessible environment is one in which the agent can obtain complete, accurate, up-to-date information about the environment's state. The more accessible an environment is, the simpler it is to build agents to operate in it

Eg: Chess

- **Partially Observable:** The relevant features are partially observable. Need to track how to deliberate.

Eg: Poker



BASED ON DETERMINISM

- **Deterministic:** The next state of the environment is completely determined by the current state and the action executed by the agent.

Eg: Image analysis

  - **Stochastic:** If an element of uncertainty occurs then it is stochastic may be partially observable. Non-deterministic environments present greater problems for the agent designer. Stochastic is random in nature. Doe environment have aspects beyond the control of the agent?
  - Utility functions have to guess at changes in world

Eg: Ludo

- **Strategic:** If the environment is dependent on preceding state and action of other agents

Eg: Chess

BASED ON DYNAMISM

- **Static (vs. dynamic):** The environment is unchanged while an agent is deliberating. A static environment is unchanged while an agent is reflecting.

Eg: Speech Analysis, Crossword Puzzle

- **Dynamic :** A dynamic environment is one that has other processes operating on it, and which hence changes in ways beyond the agent's control. Other processes can interfere with the agent's actions (as in concurrent systems theory). The physical world is a highly dynamic environment. Semi-Dynamic: The environment is semi-dynamic if the environment itself does not change with the passage of time, but the agent's performance score does

Eg: Vision systems in drones

BASED ON CONTINUITY

- Discrete (vs. continuous): A limited number of distinct, clearly defined percepts and actions. An environment is discrete if there are a fixed, finite number of actions and percepts in it . Discrete environments could in principle be handled by a kind of "lookup table"

Eg: chess game, crossword

- Continuous: Continuous environments have a certain level of mismatch with computer systems

Eg: taxi driving.

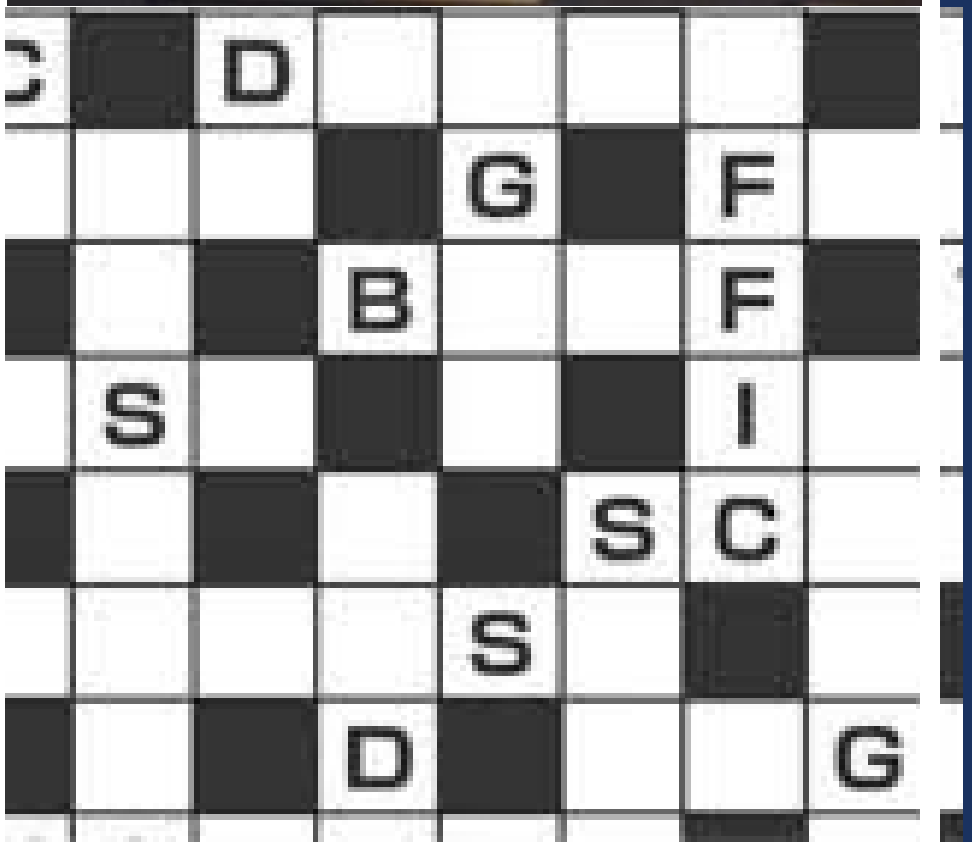
BASED ON AGENTS EXPERIENCE

- **Episodic :** The agent's experience is divided into discrete"episodes" (each episode consists of the agent perceiving and then performing a single action), and the choice of action in each episode depends only on the episode itself. In an episodic environment, the performance of an agent is dependent on a number of discrete episodes, with no link between the performance of an agent in different scenarios Episodic environments are simpler from the agent developer's perspective because the agent can decide what action to perform based only on the current episode — it need not reason about the interactions between this and future episodes

Eg: Part Picking Robot, Image Analysis

- **Sequential:** In a Sequential environment, the agent has to take decisions considering its previous decisions.

Eg: Chess, Automated car



BASED ON NUMBER OF AGENTS

- **Single agent:** An agent operating by itself in an environment.

Eg:Crossword

- **Multiagent:** A multi-agent system is a computerized system composed of multiple interacting intelligent

Eg: Chess

EXAMPLE

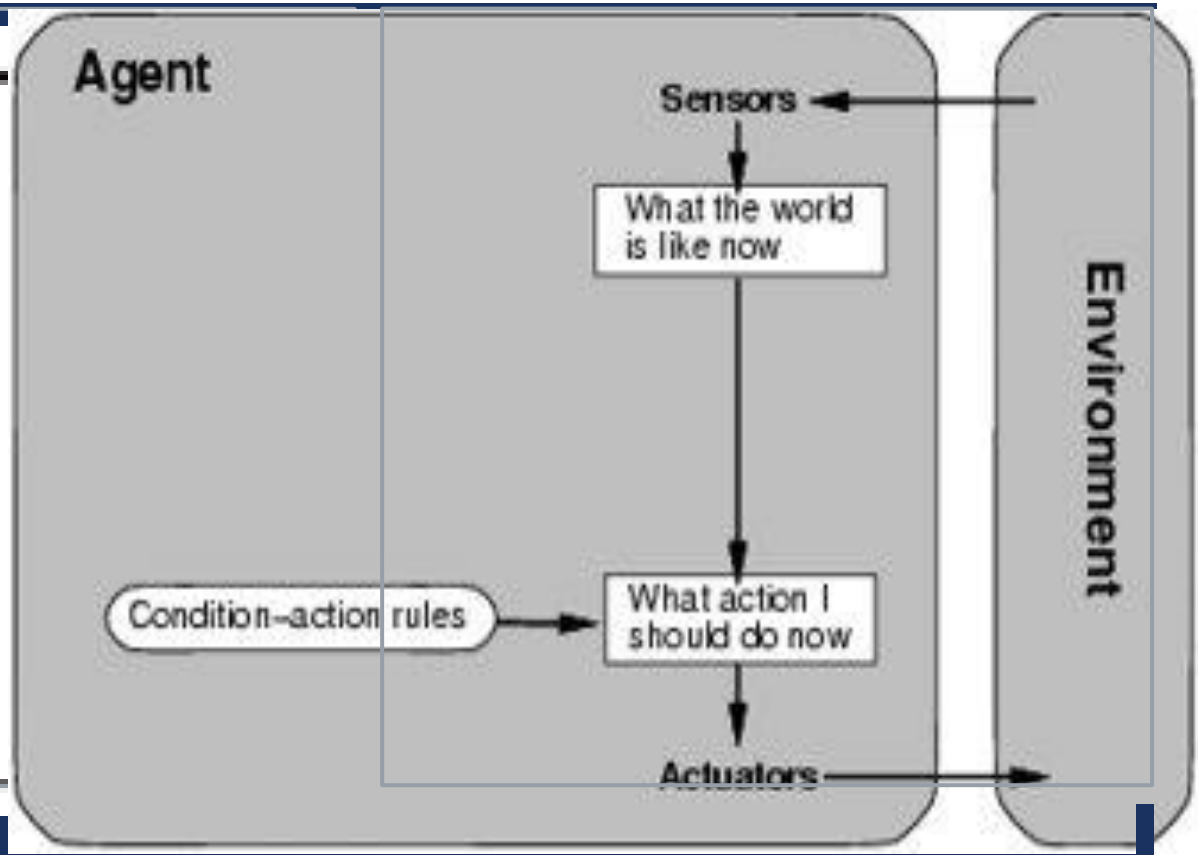
Environment	Accessible	Deterministic	Episodic	Static	Discrete
Chess with a clock	Yes	Yes	No	Semi	Yes
Chess without a clock	Yes	Yes	No	Yes	Yes
Poker	No	No	No	Yes	Yes
Backgammon	Yes	No	No	Yes	Yes
Taxi driving	No	No	No	No	No
Medical diagnosis system	No	No	No	No	No
Image-analysis system	Yes	Yes	Yes	Semi	No
Part-picking robot	No	No	Yes	No	No
Refinery controller	No	No	No	No	No
Interactive English tutor	No	No	No	No	Yes

AGENT TYPES

- Four basic types in order of increasing generality:
  1. **Simple reflex agents**
  2. **Reflex agents with state/model**
  3. **Goal-based agents**
  4. **Utility-based agents**
  5. **Learning Agents**

```
function REFLEX-VACUUM-AGENT([location,status]) returns an action

    if status = Dirty then return Suck
    else if location = A then return Right
    else if location = B then return Left
```



SIMPLE REFLEX AGENTS

SIMPLE REFLEX AGENTS

Action does not depend on percept history, only on current percept.

Environment should be **fully observable**

Agent function uses **Condition-Action-rule**.

Condition-Action Rule: It is a rule that maps a state (condition) to an action.

They are rational only if a correct decision is made only on the basis of current precept.

Eg: **Robotic Vaccum Cleaner, smart thermostat**

If each internal state includes all information relevant to information making, the state space is **Markovian**

SIMPLE REFLEX AGENTS

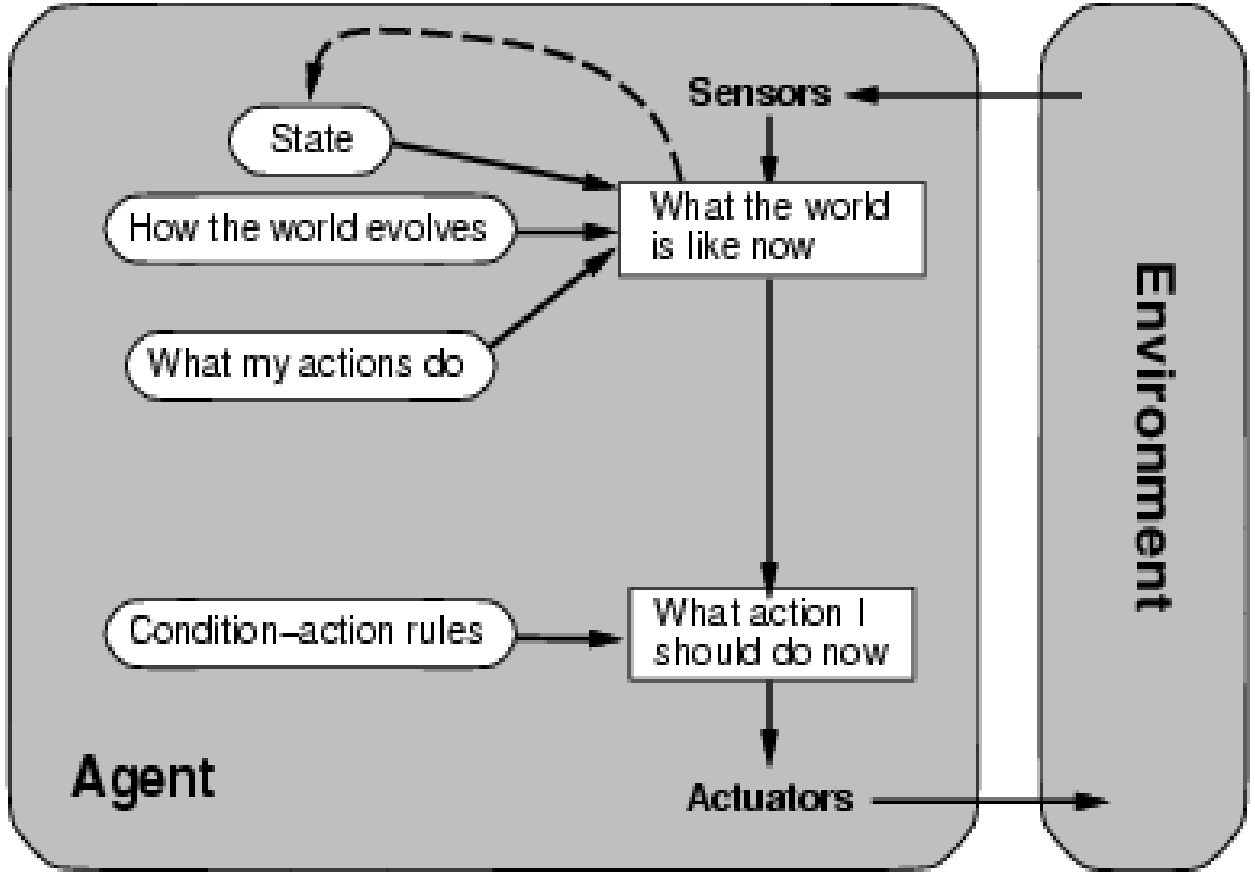
ADVANTAGES

- Simple but very limited intelligence.
- Therefore no memory requirements.

DISADVANTAGES

- They have very low intelligence capability as they don't have the ability to store past state.
- No knowledge of non-perceptual parts of state.
- No autonomy as actions are predetermined and everything is done by designer.
- Usually too big to generate and store.
  - Lookup table (not a good idea in general) as ^{35100} entries required for the entire game
- If there occurs any change in the environment, then the collection of rules need to be updated.
- Infinite loops due to partially observable environments
  - Suppose vacuum cleaner does not observe location. What do you do given location = clean? Left of A or right on B -> infinite loop.
  - Possible Solution: Randomize action.

MODEL-BASED REFLEX AGENTS




```
function REFLEX-AGENT-WITH-STATE(percept) returns action
  static: state, a description of the current world state
          rules, a set of condition-action rules

  state ← UPDATE-STATE(state, percept)
  rule ← RULE-MATCH(state, rules)
  action ← RULE-ACTION[rule]
  state ← UPDATE-STATE(state, action)
  return action
```

MODEL BASED AGENT

Can handle **Partially observable Environment**.

It also has the capability to **store the internal state (past information)** based on previous events. Model-Based Agents **updates the internal state at each step.**

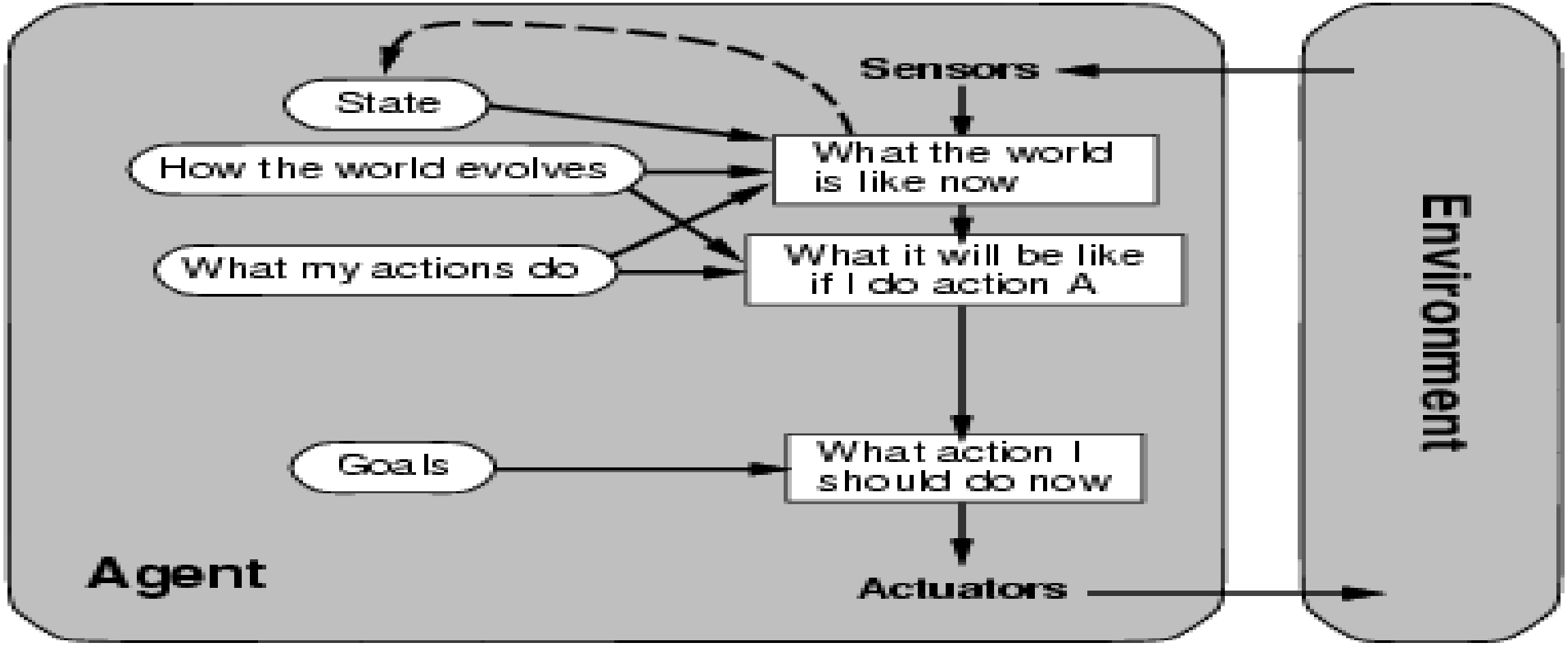
Agent perceives from history and has to keep **track of internal state** which is adjusted by each percept depending on percept history.

Knowledge is used to **build model and hence requires Knowledge Database** and becomes knowledge based agent

Condition action rules based on perception

Eg: **self-steering mobile vision**

GOAL-BASED AGENTS



GOAL BASED AGENTS

Depends on **current percept, previous percept & goal or desirable situation.**

It has a **goal and has a strategy to reach that goal**. A **goal** is a description of a desirable situation.

All actions are taken to reach this goal. From a set of possible actions, it selects the one that improves the progress towards the goal.

In order to attain its goal, it makes use of the **search and planning algorithm.**

The sequence of steps required to solve a problem is not known priori and is determined by **systematic exploration.**

Eg: **searching robot**

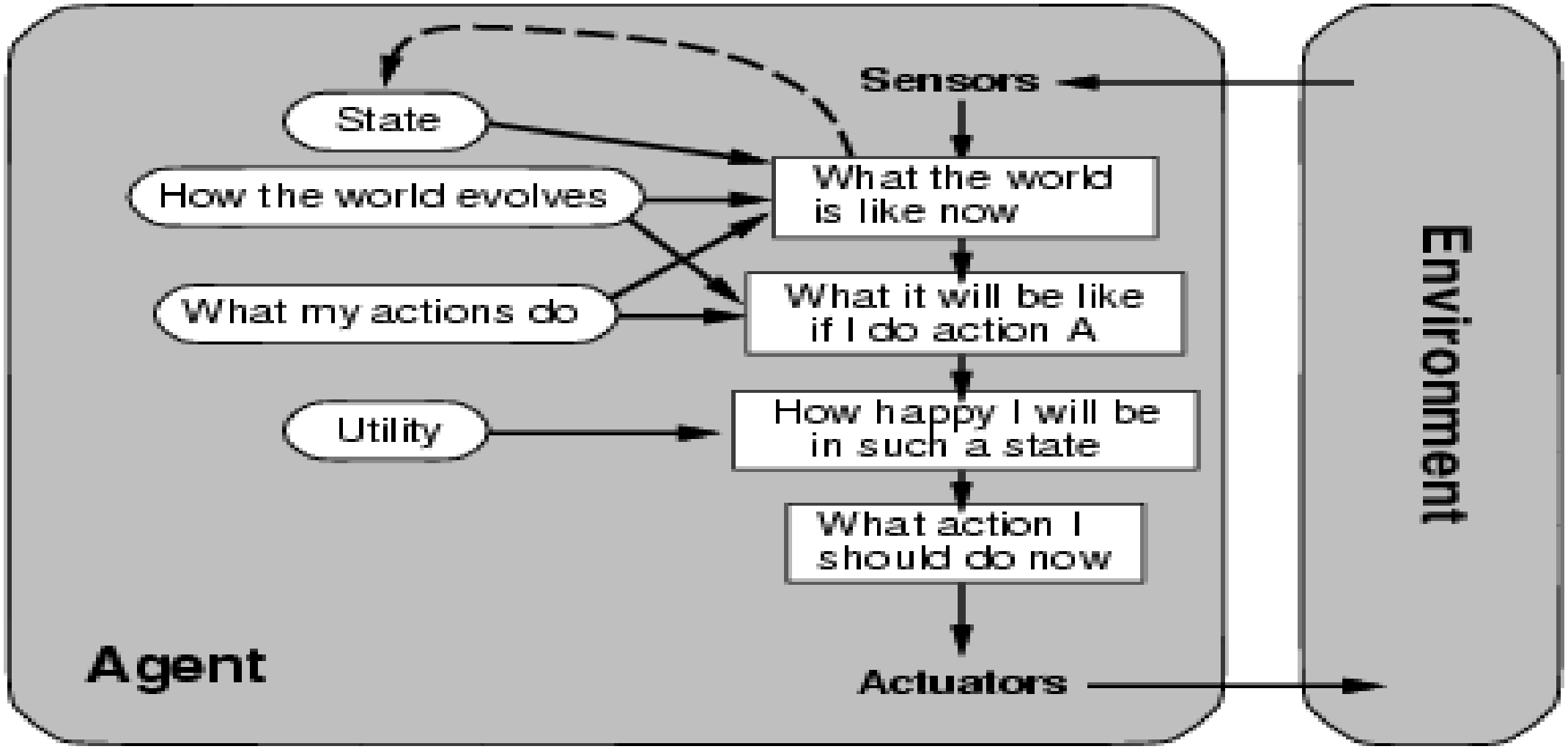
Keeping track of the current state is often not enough - need to add goals to decide which situations are good

Deliberative instead of **reactive**.

May have to consider long sequences of possible actions before deciding if goal is achieved – involves consideration of the future, *"what will happen if I do...?"*

One drawback of Goal-Based Agents is that **they don't always select the most optimized path to reach the final goal**

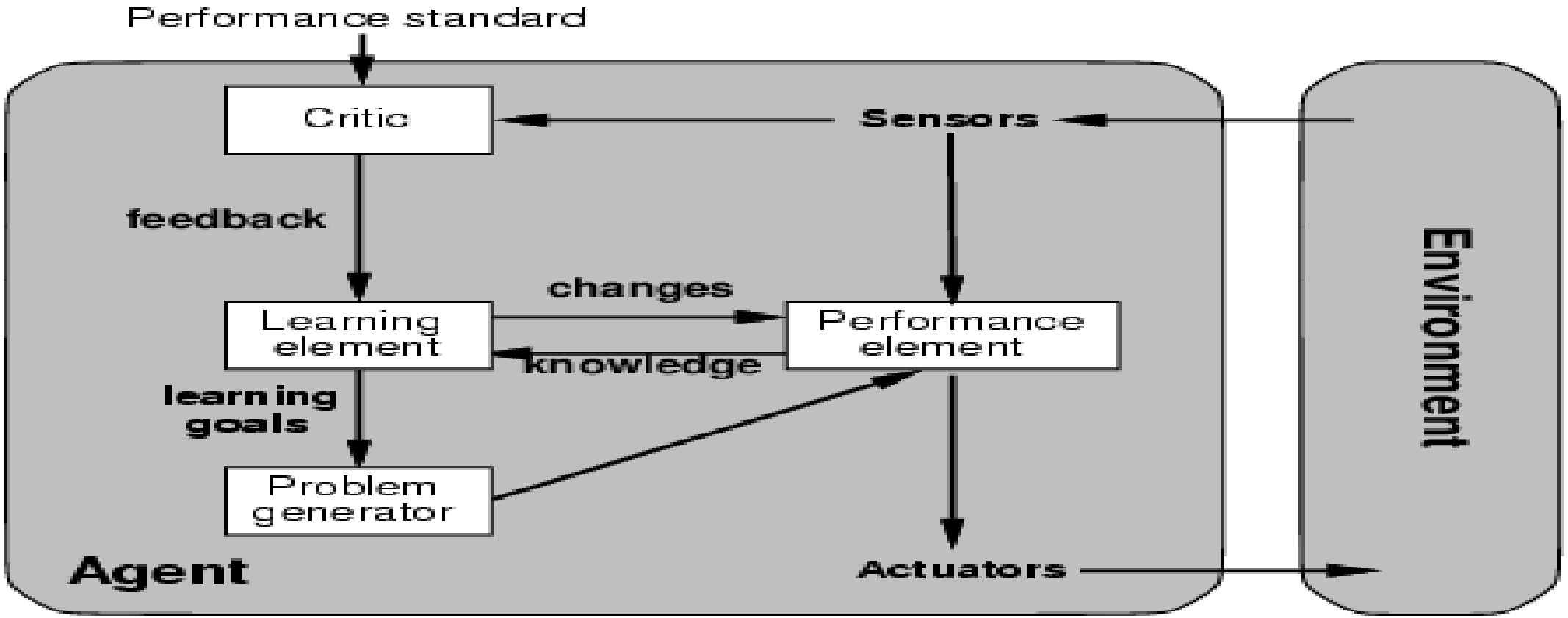
UTILITY-BASED AGENTS



UTILITY BASED AGENTS

- Utility Agents are used when there are **multiple solutions to a problem and the best possible alternative** has to be chosen.
- Action Depends on **utility function(state based)** and not goal.
- A **utility function** maps a state onto a real number which describes the associated degree of “happiness”, “goodness”, “success”.
- It is like the goal-based agent but with a measure of cost-benefit analysis of each solution and select the one which can achieve the goal in minimum cost.
- Status of each state is checked, and decision is made.
- Eg: **route recommendation system**

LEARNING AGENTS



LEARNING AGENTS

It is capable of **learning from past experience** which is achieved through **four major components**

It can operate on **unknown environments**

The learning agents have which enable to achieve it

- **Critic:** The Critic evaluates how well is the agent performing vis-à-vis the set performance benchmark.
- **Learning Elements:** It takes input from the Critic and helps Agent in performance improvement by learning from the environment.
- **Performance Element:** This component decides on the action to be taken to improve the performance.
- **Problem Generator:** Problem Generator takes input from other component and suggests actions which will result in a better experience.

It has the capability of **automatic information acquisition and integration** into the system.

These types of agents can start from scratch and over time can acquire significant knowledge from their environment.

AI PROBLEM FORMULATION

AI PROBLEM FORMULATION CONCERNS

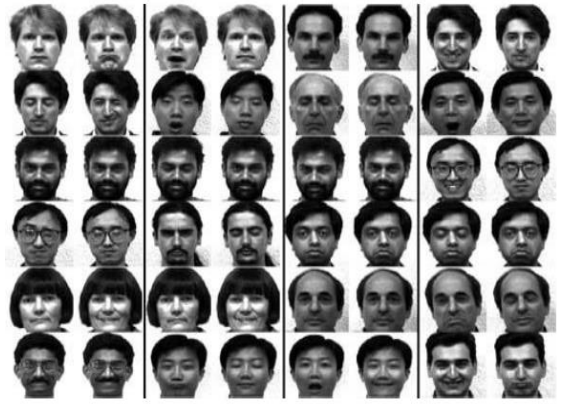
- How to **represent the problem precisely** so that can be easily analyzed?
- How to **represent knowledge which has been isolated particularly for solving problem**?
- Which **approach** to be used to solve a problem?
- **Problem-solving agents** in AI mostly uses **search strategies or algorithms** to solve a specific problem and provide the best result.
- AI agents should achieve **autonomity, adaptability, rationality, social abilty, coperativity...**

KNOWLEDGE REPRESENTATION ISSUES

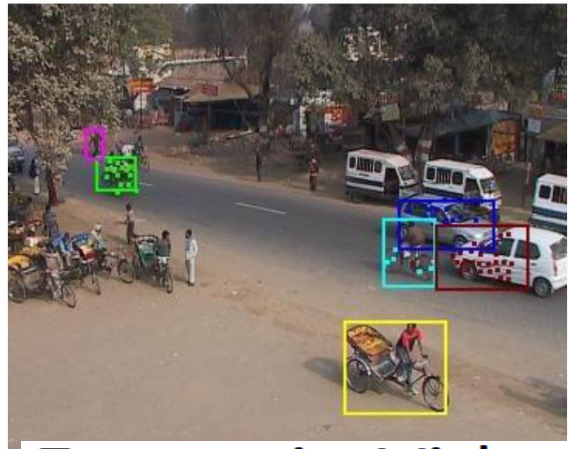
- **Structured Data**
- Features already extracted as Data + tags; (Relational Databases)
- e.g. Movie Preference matrix (Netflix)

- **Unstructured Data**
- **How to identify objects and their relations from unstructured data?**
- e.g. **Text:** Newspapers, blogs, technical papers

  Images: ImageNet,
- **Which features to use?**



Example : Face Recognition



Events in Video

AI PROBLEMS ASSOCIATED TYPES

- Reasoning and Problem Solving: How to work with incomplete and uncertain information? How to develop fast, intuitive algorithms for complex problems
- Knowledge Representation: To store what it knows or receives. AI can use **Logical Representation**(Propositional logic, first order logic), **Production Rule, Semantic Network:**(graphical networks), **Frame representation:**(records with slots and attributes)
- **Automated** reasoning: To use the stored information to answer questions and to draw new conclusions.
- Planning: How to set goals, achieve them, and visualize the future? Accessibility to its environment, make predictions, evaluate predictions, and adapt according to its assessment
- Learning: the ability to take a stream of input and find patterns in it. This includes classification and numerical regression.
- Natural Language Processing: How to read human language and understand it
- Perception: speech recognition, facial recognition, and object recognition
- Motion and Manipulation
- Social Intelligence

PROBLEM SOLVING AGENTS

- Intelligent agents adopt a goal and try to achieve it for maximizing performance measures.
- **Goal formulation**, based on the current situation and the agent's performance measure, is the **first step in problem solving**. A **goal is a description desirable state**. Goal directed agents need to achieve goal.
- **Goals of an agent** can be modelled using state **space search**. Each **state** is an **abstract representation of a physical configuration of agent's environment.**
- Agent moves from current sate to successor state through some actions. It is known as **plan/ path** which is a sequence of actions to reach its goal and is examined using goal test.
- **Problem formulation**: choosing relevant set of states together and feasible set of operators to move from one state to another state, given a goal. Problem formulation is the process of deciding what sorts of actions and states to consider, given a goal
- **Sate space:** problem is divided into intermediates states to reach goal state, set of all possible states reachable from initial state
- An **operator** is a function that expands a node

PROBLEM SOLVING AGENTS

- An agent with several immediate options of unknown value can decide what to do by first examining different possible sequences of actions that lead to states of known value, and then choosing the best sequence
- Looking for such a sequence is called **search**
- A search algorithm takes a problem as input and returns a solution in the form of action sequence
- Once a **solution** is found the actions it recommends can be carried out – execution phase
- **"formulate, search, execute"** design for the agent
- A **node** is a data structure constituting part of a search tree includes state, parent node, action, path cost g(x), depth

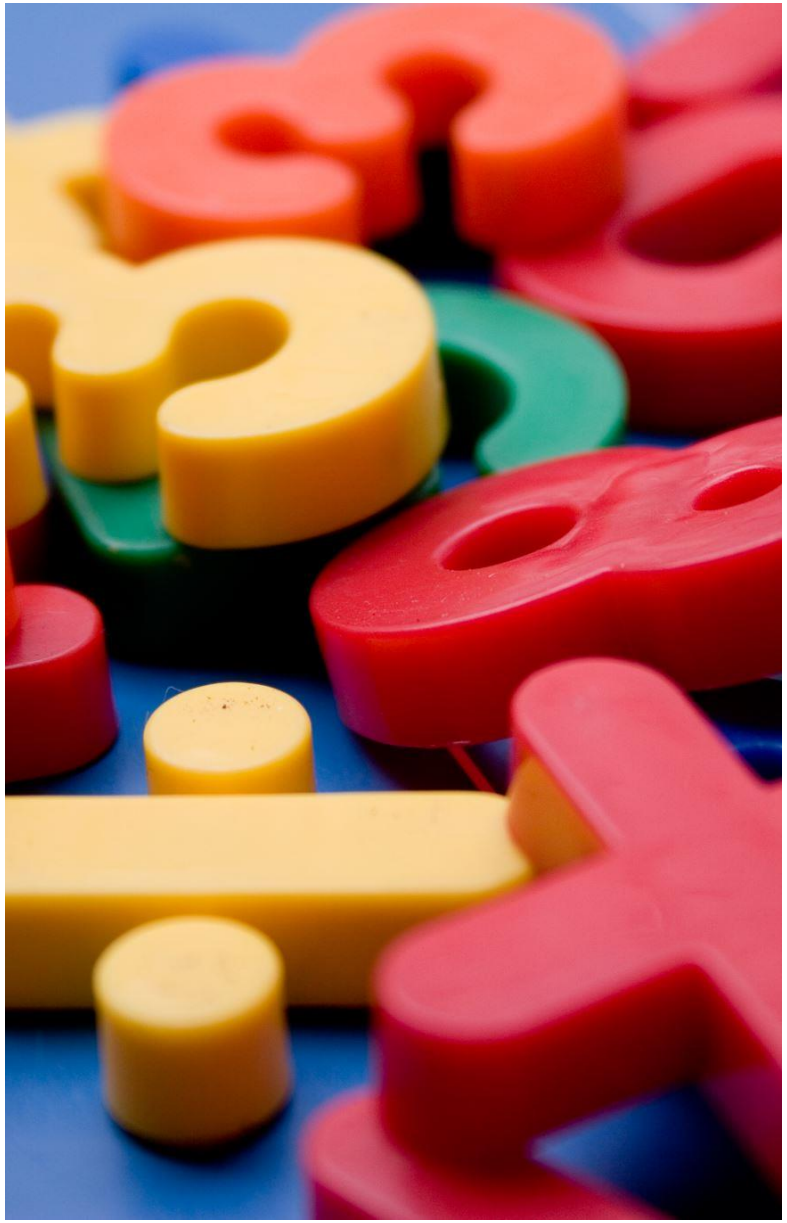
HOW TO FORMULATE A PROBLEM?

A problem can be defined formally by 5 components:

1. The **initial state** of the agent. In this case, the initial state can be described as *In: Arad*
2. The **possible actions** available to the agent, corresponding to each of the state the agent resides in.
3. The **transition model** describing what each action does. A successor function (transition model) Given a state, generates its successor states
4. The **goal test**, determining whether the current state is a goal state. Here, the goal state is {*In: Bucharest*}
5. The **path cost** function, which determine the cost of each path, which is reflecting in the performance measure.

REPRESENTING STATE

- The **size of a problem** is usually described in terms of the **number of states** that are possible.
  - The 8-puzzle has 181,440 states.
  - Tic-Tac-Toe has about 3^9 states.
  - Rubik's Cube has about 10^{19} states.
  - Checkers has about 10^{40} states.
  - Chess has about 10^{120} states in a typical game
- The number of actions / operators depends on the **representation** used in describing a state.
- **Closed World Assumption:** All necessary information about a problem domain is available in each percept so that each state is a complete description of the world.



AI problem can be represented as a state space inorder to solve it which can be either tree or graph. The state space is then searched to find a solution to the problem.

A sate space is set of all possible states where the problem can lead to from current state.

A search tree has following

Root = Start state	Set of nodes = representing each state of the problem	goal state.	Children = successor states	Edges = actions and costs. The legal moves from one state to another

HOW TO SOLVE A PROBLEM?

STATE SPACE TREE: *A state space tree is a tree constructed from all of the possible states of the problem as nodes, connected via state transitions from some initial state as root to some terminal state as leaf.*

FORMAL REPRESENTATION OF STATE SPACE

- a state space can be graphically represented using **tree or graph** and formally represented as a tuple **{S, A, Action(s), Result(s,a), Cost(s,a)},** in which:
  - **{ S}** is the set of all possible states;
  - **{A}** is the set of possible action, not related to a state but regarding all the state space;
  - **{Action(s)}** is the function that establish which action is possible to perform in a certain state;
  - **{Result(s,a)}** is the function that return the state reached performing action { a} in state {s}
  - **{ Cost(s,a)}** is the cost of performing an action {a} in state {s}. In many state spaces is a constant, but this is not true in general.

FEW TERMINOLOGIES

The initial state, the actions and the transition model together define the **state space** of the problem — the set of all states reachable by any sequence of actions

A **path** in the state space is a sequence of states connected by a sequence of actions.

The **solution** to the given problem is defined as the sequence of actions from the initial state to the goal states.

The quality of the solution is measured by the cost function of the path, and an **optimal solution** has the lowest path cost among all the solutions.

PERFORMANCE MEASURE OF PROBLEM-SOLVING AGENTS

Completeness: Is the algorithm guaranteed to find a solution if there exist one?

Optimality: Does the algorithm find the optimal solution?

Time complexity: How long does it take for the algorithm to find a solution?

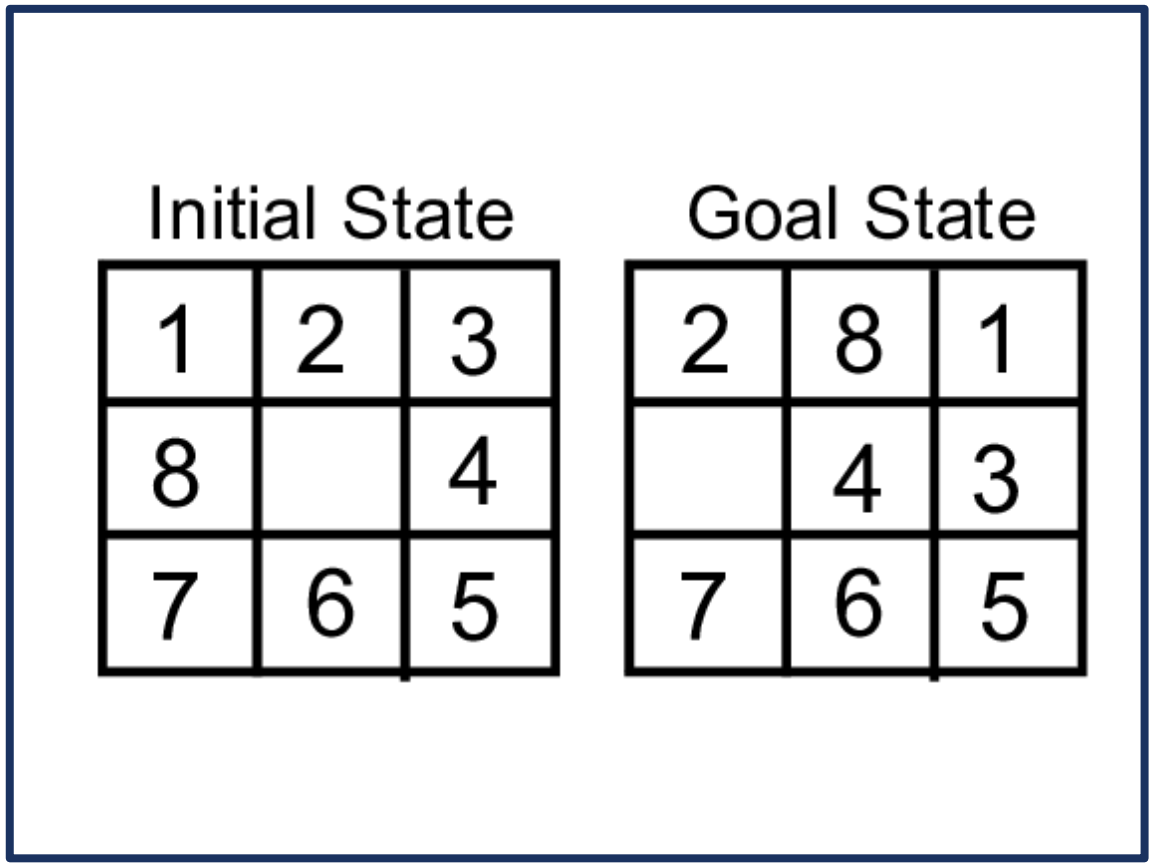
Space complexity: How much memory is consumed in finding the solution?

CLASSICAL AI PROBLEMS

Toy problems are well-defined problems that have concise, exact description which are intended to illustrate or exercise various problem-solving methods. Following are the classical Toy problems.

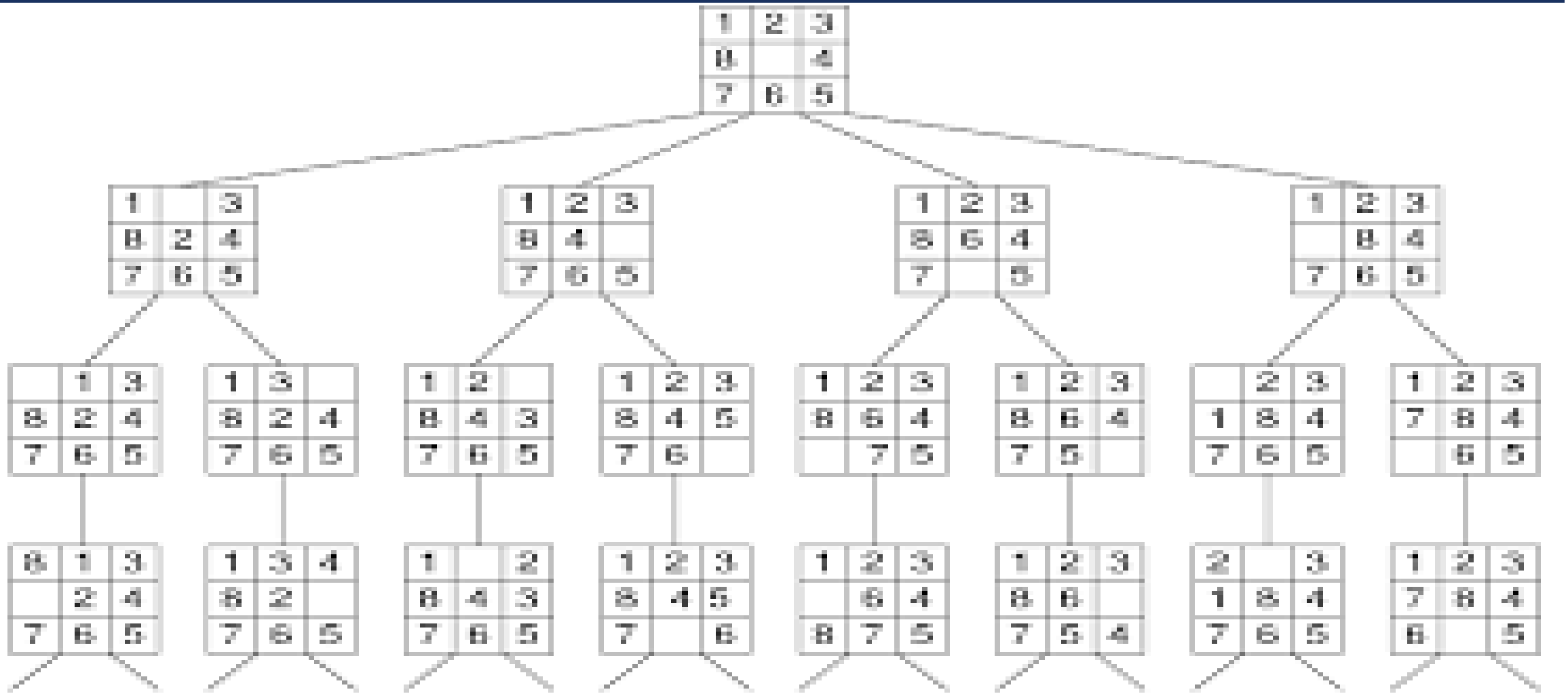
- **TRAVELLING SALESMAN PROBLEM**
- **CHESS**
- **TOWER OF HANOI**
- **8-PUZZLE PROBLEM**
- **WATER JUG PROBLEM**
- **MISSIONARY-CANNIBAL PROBLEM**
- **N-QUEENS PROBLEM**
- **CRYPTARITHMETIC**
- **REMOVE 5 STICKS**
- **BLOCK WORD PROBLEM**

EXAMPLE 1: 8 PUZZLE PROBLEM



- **PROBLEM DEFINITION:** Given a 3-by-3 grid with 8 square blocks labeled from 1 to 8 and a blank square. Goal is to rearrange the blocks to **produce a desired goal configuration of the tiles.** Player permitted to slide blocks horizontally or vertically into the blank square, if the blank square is in adjacent space.
- **State:** Specification of each of the eight tiles in the nine squares (the blank is in the remaining square).
- **Initial state:** Any state.
- **Successor function (Actions):** Blank moves *Left, Right, Up*, or *Down*.
- **Goal test:** Check whether the goal state has been reached.
- **Path cost:** Each move costs 1. The path cost = the number of moves.

STATE SPACE TREE FOR 8 PUZZLE PROBLEM

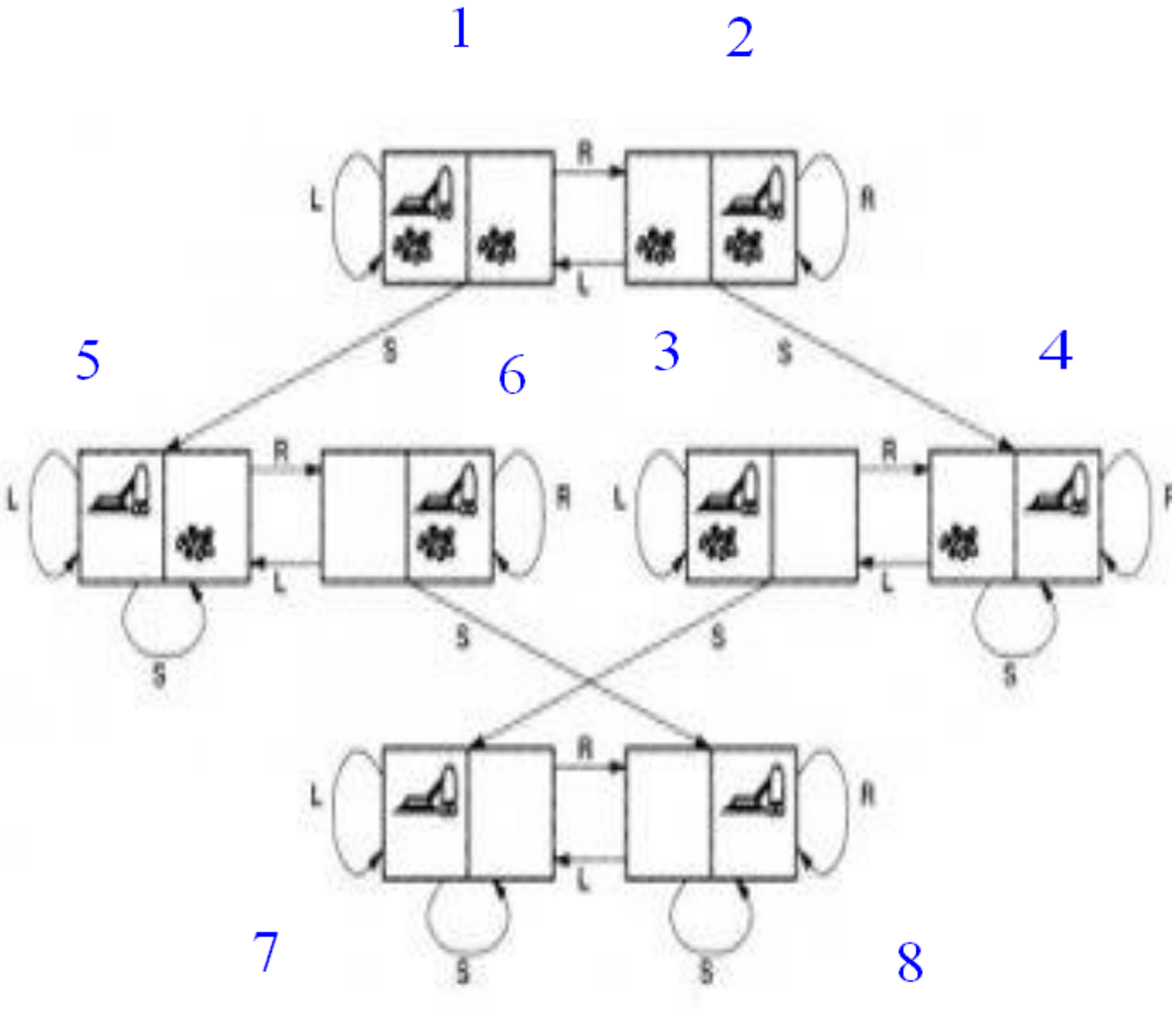


EXAMPLE 2: VACCUM CLEANER WORLD

- **States** integer dirt and robot locations. The agent is in one of two locations, each of which might or might not contain dirt – 8 possible states
- **Initial state:** any state
- **Actions:** left, right, suck, noOp
- **Goal:** test not dirty?
- **Path cost:** 1 per operation (0 for noOp)

STATE SPACE TREE

- Actions: Left, Right, Suck
- Goal state: 7, 8
- Initial state: one of {1,2,3,4,5,6,7,8}
- Right: one of {2,4,6,8}
- Solution: [Right, Suck, Left, Suck]



EXAMPLE 3: WATER JUG PROBLEM

- **PROBLEM DEFINITION:** given two jugs, a 4-gallon one and a 3-gallon one, Neither jug has any measuring markings on it. There is a pump that can be used to fill jugs with water. How can you get exactly 2 gallons of water in the 4-gallon jug?
- **State Representation:** Represent a state of the problem as a tuple (x, y) where x represents the amount of water in the 4-gallon jug and y represents the amount of water in the 3-gallon jug. where $0 \leq x \leq 4$, and $0 \leq y \leq 3$.
- **Initial state:** (0,0)
- **Goal state:** (2,y) where $0 \leq y \leq 3$.

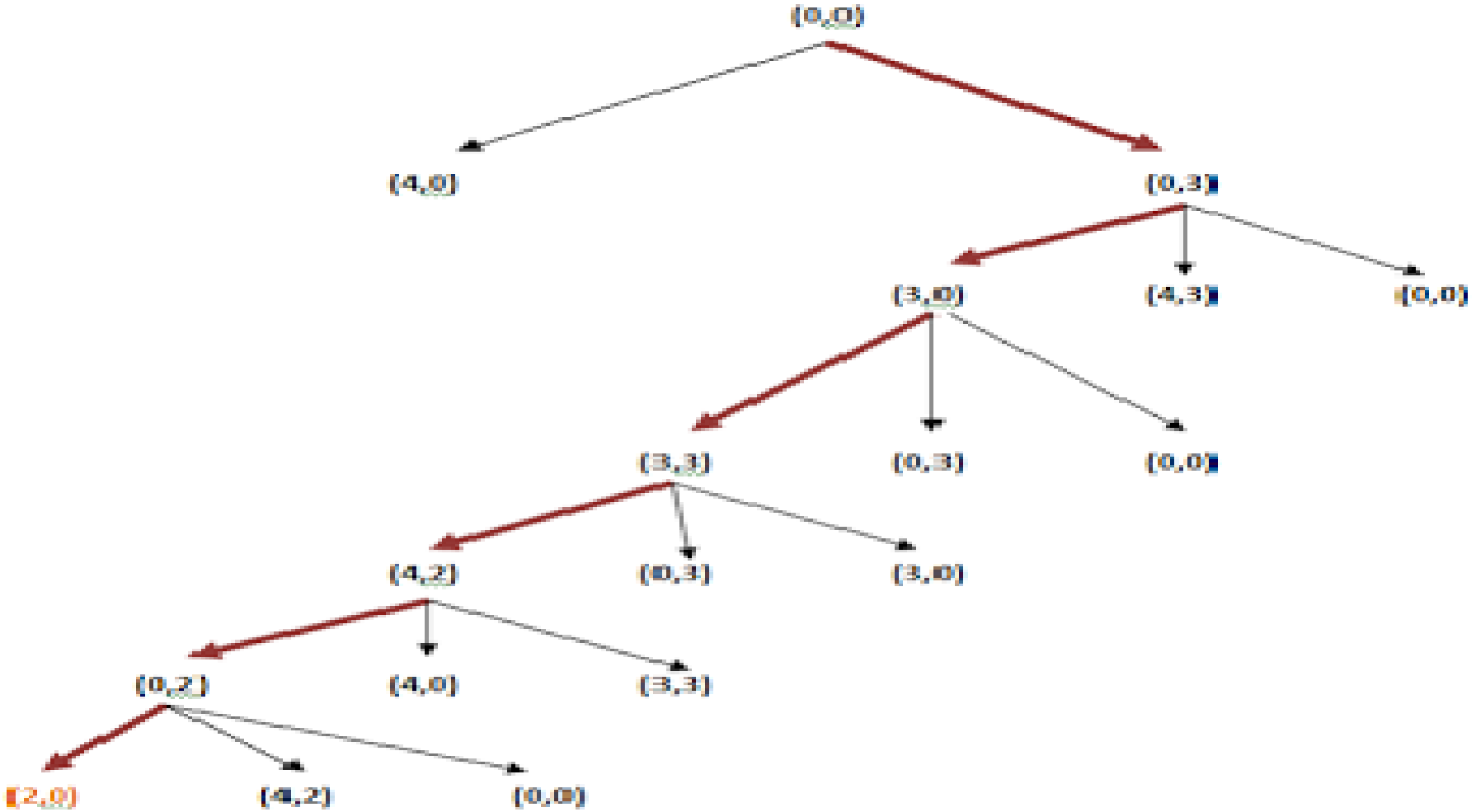
PRODUCTION RULES FOR SOLVING THE WATER JUG PROBLEM

No.	Initial State	Condition	Final state	Description of action taken
1.	(x,y)	If x<4	(4,y)	Fill the 4 gallon jug completely
2.	(x,y)	if y<3	(x,3)	Fill the 3 gallon jug completely
3.	(x,y)	If x>0	(x-d,y)	Pour/spill some part from the 4 gallon jug
4.	(x,y)	If y>0	(x,y-d)	Pour/spiil some part from the 3 gallon jug
5.	(x,y)	If x>0	(0,y)	Empty the 4 gallon jug
6.	(x,y)	If y>0	(x,0)	Empty the 3 gallon jug
7.	(x,y)	If (x+y)<7	(4, y-[4-x])	Pour some water from the 3 gallon jug to fill the four gallon jug
8.	(x,y)	If (x+y)<7	(x-[3-y],3)	Pour some water from the 4 gallon jug to fill the 3 gallon jug.
9.	(x,y)	If (x+y)<4	(x+y,0)	Pour all water from 3 gallon jug to the 4 gallon jug
10.	(x,y)	if (x+y)<3	(0, x+y)	Pour all water from the 4 gallon jug to the 3 gallon jug

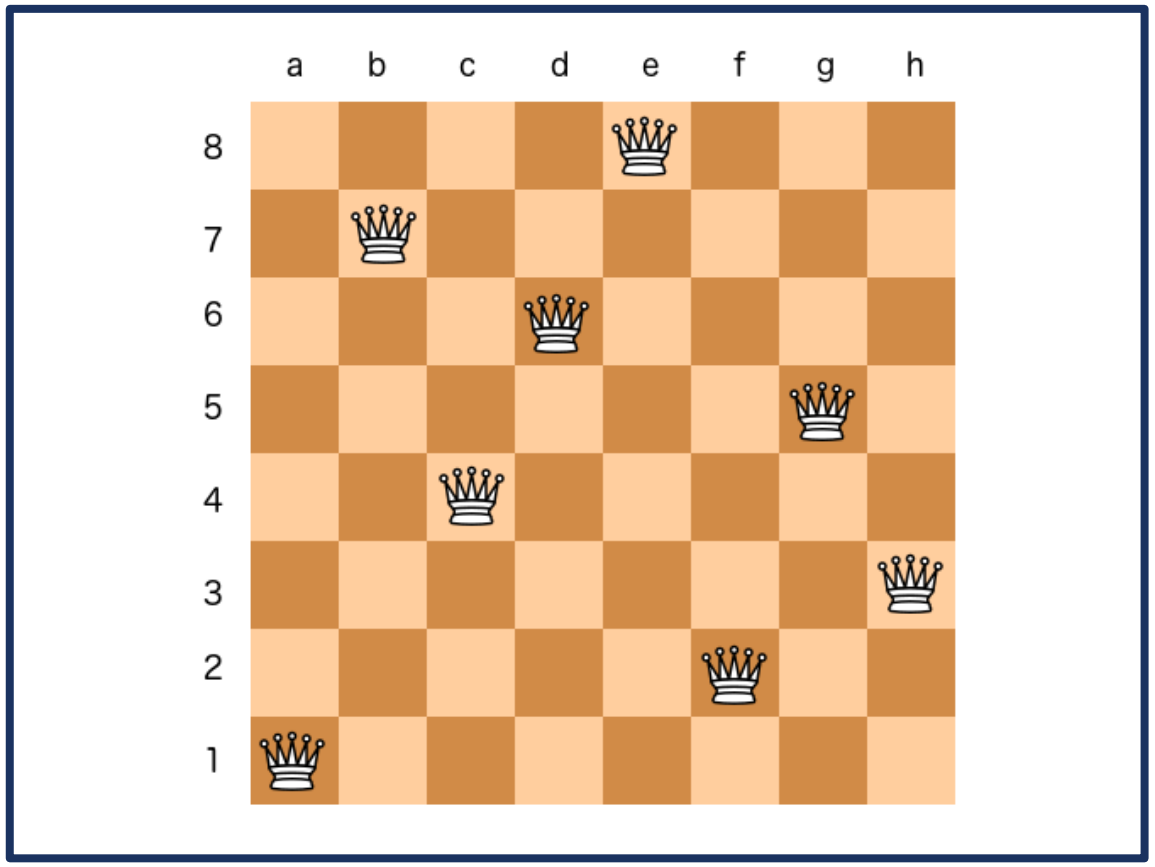
SOLUTION OF WATER JUG PROBLEM ACCORDING TO THE PRODUCTION RULES

S.No.	4 gallon jug contents	3 gallon jug contents	Rule followed
1.	0 gallon	0 gallon	Initial state
2.	0 gallon	3 gallons	Rule no.2
3.	3 gallons	0 gallon	Rule no. 9
4.	3 gallons	3 gallons	Rule no. 2
5.	4 gallons	2 gallons	Rule no. 7
6.	0 gallon	2 gallons	Rule no. 5
7.	2 gallons	0 gallon	Rule no. 9

STATE SPACE TREE FORWATER JUG PROBLEM



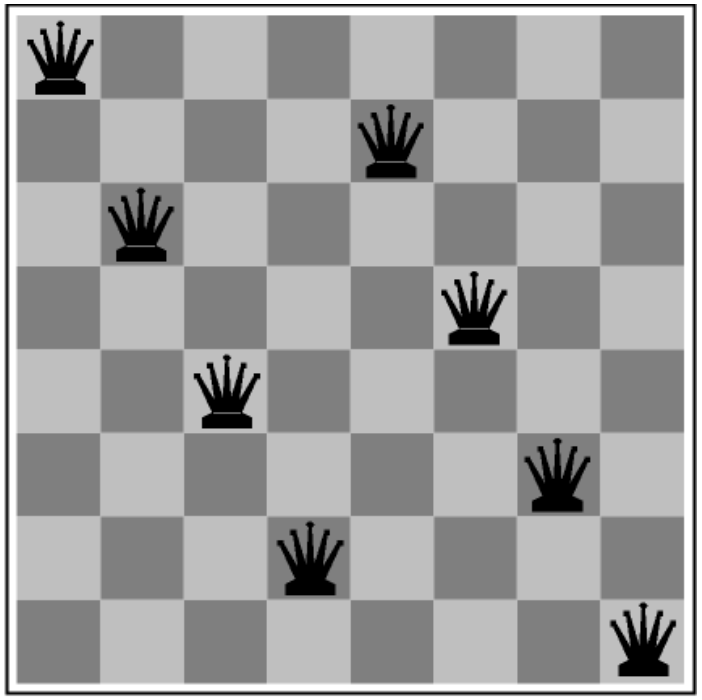
EXAMPLE 4: 8 QUEENS PROBLEM FORMULATION I



Naïve formulation/ Incremental formulation

- **PRBLEM DEFINITION:** Place 8 queens on an (8 by 8) chess board such that none of the queens attacks any of the others. This problem can be solved by searching for a solution.
- **states:** any arrangement of 0-8 queens on the board is a state
- **Initial state:** no queens on the board
- **actions:** add a queen to any empty square
- **goal test:** 8 queens are on the board, none attacked
- Problem: $64 \cdot 63 \cdot \ldots \cdot 57 \approx 10^{14}$ possible states

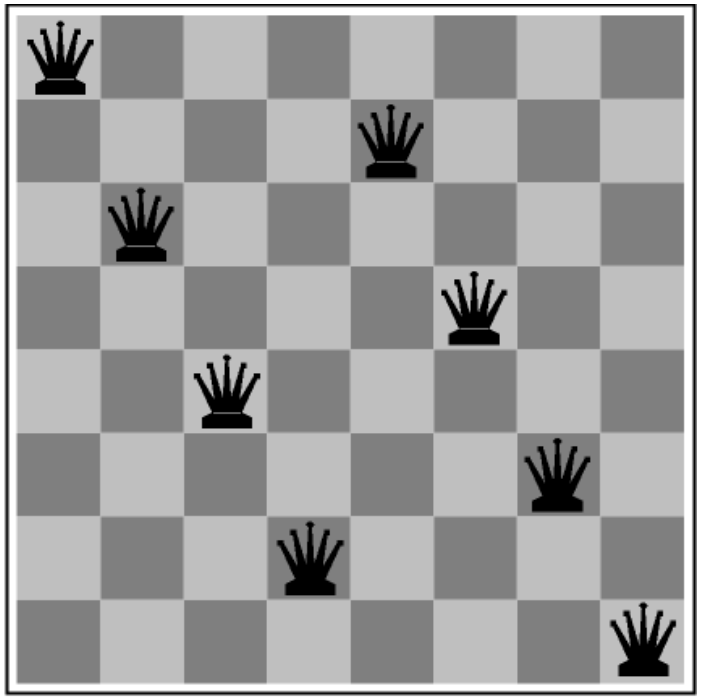
8 QUEENS PROBLEM ANOTHER FORMULATION



Complete Formulation

- **States:** Any arrangement of n queens (0 £ n £ 8) one per column in the leftmost n columns such that no queen attacks another. Any arrangement of 8 queens on the board.
- **Initial state:** All queens at coulmn 1.
- **Successor function:** Change the position of any one queen. Add a queen to any square in the leftmost empty column such that it is not attacked by any other queen.
- **Actions:** Change the position of any one queen
- **Problem:** 2,057 states. Sometimes no admissible states can be found.

8 QUEENS PROBLEM ANOTHER FORMULATION



Better formulation

- **States:** .Any arrangement of k queens on k rows so that none are attacked.
- **Successor function:** Add a queen to(k+1)the row so that none are attacked each other.
- **Initial state:** 0 Queens are on the board.

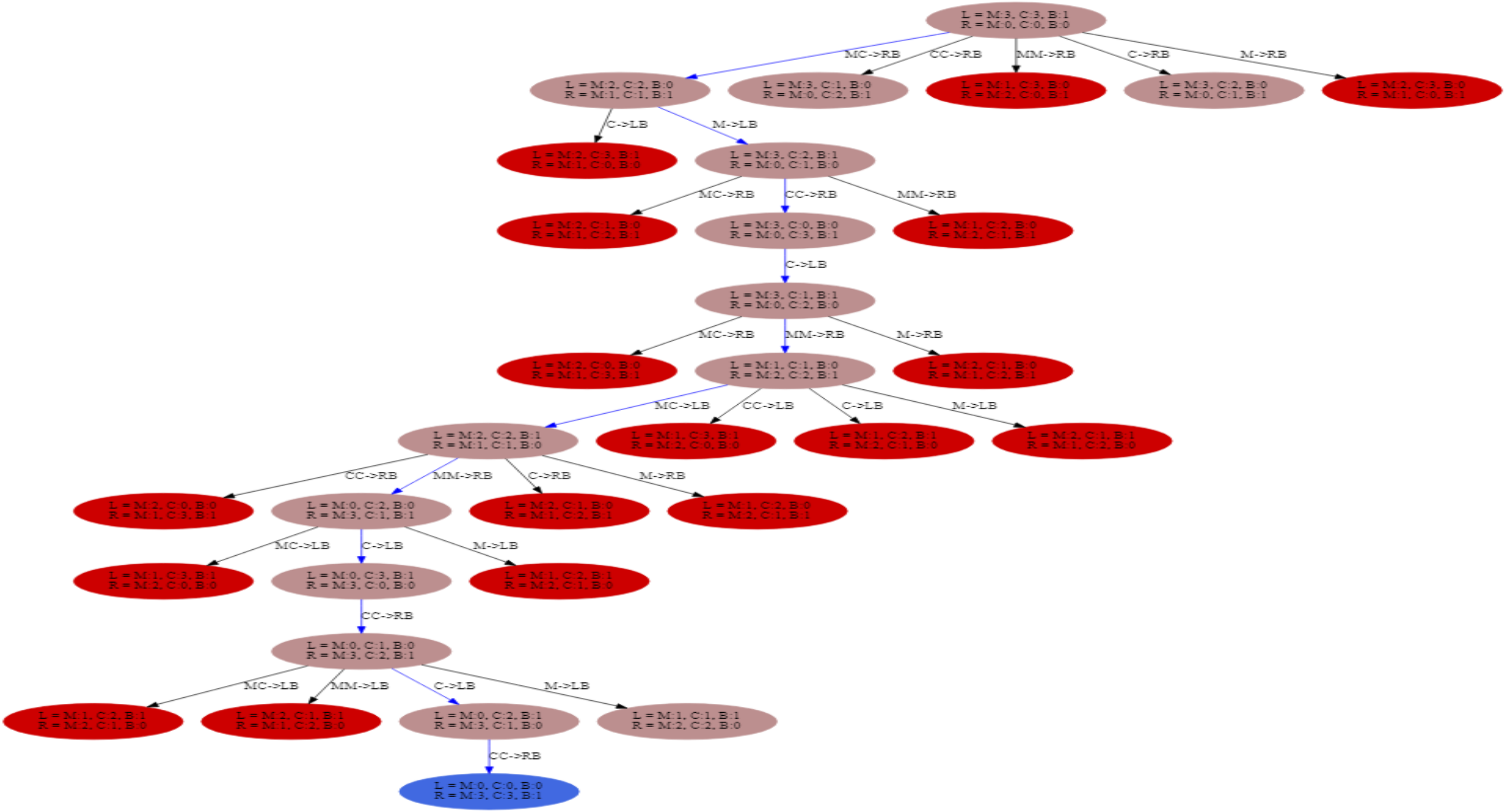
EXAMPLE 5: MISSIONARIES AND CANNIBALS

- **PROBLEM DEFINITION:** Three missionaries and three cannibals are on one side of a river, along with a boat that can hold one or two people. **Find the smallest number of crossings that will allow everyone to cross the river safely**, without ever leaving a group of missionaries outnumbered by cannibals.
- **State:** (m, c, 1/0) where m: number of missionaries in the first/left bank c: number of cannibals in the first/left bank The last bit indicates whether the boat t is in the first bank. is in the first bank.
- **Start state:** (3, 3, 1) (3, 3, 1)
- **Goal state:** (0, 0, 0) (0, 0, 0)
- **Operators:** Boat carries (1, 0) or (0, 1) or (1, 1) or (2, 0) or (0, 2)

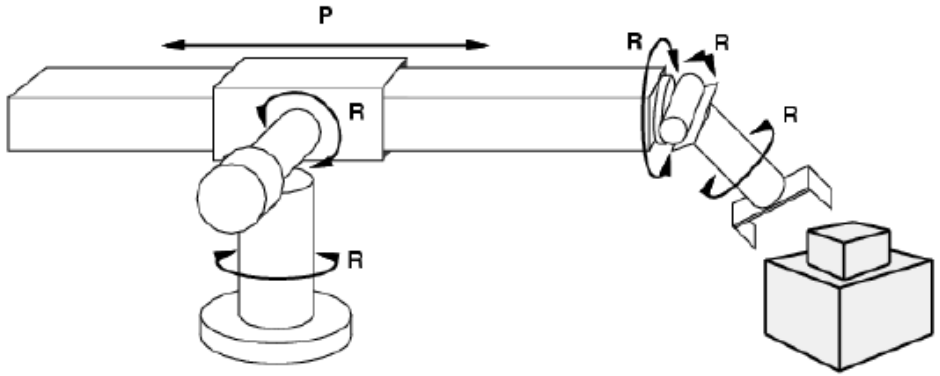
MISSIONARY AND CANNIBAL SOLUTION

	Near side		***Far side***
0 Initial setup:	MMMCCC	B	-
1 Two cannibals cross over:	MMMC	B	CC
2 One comes back:	MMMCC	B	C
3 Two cannibals go over again:	MMM	B	CCC
4 One comes back:	MMMC	B	CC
5 Two missionaries cross:	MC	B	MMCC
6 A missionary & cannibal return:	MMCC	B	MC
7 Two missionaries cross again:	CC	B	MMMC
8 A cannibal returns:	CCC	B	MMM
9 Two cannibals cross:	C	B	MMMCC
10 One returns:	CC	B	MMMC
11 And brings over the third:	-	B	MMMCCC

SOLUTION TREE



EXAMPLE 6: ROBOTIC ASSEMBLY



- **states:** real-valued coordinates of robot joint angles parts of the object to be assembled
- **actions:** continuous motions of robot joints
- **goal test:** complete assembly
- **path cost:** time to execute

EXAMPLE 7:CRYPTARITMETIC

- **PROBLEM DEFINITION:**Find an assignment of digits (0, ..., 9) to letters so that a given arithmetic expression is true.
- Examples:

1. Dudeney's puzzle reads: SEND + MORE = MONEY.
   - Cryptarithms are solved by deducing numerical values from the mathematical relationships indicated by the letter arrangements (i.e.) . S=9, E=5, N=6, M=1, O=0,….
   - The only solution to Dudeney's problem: 9567 + 1085 = 10,652.
2. **FORTY**

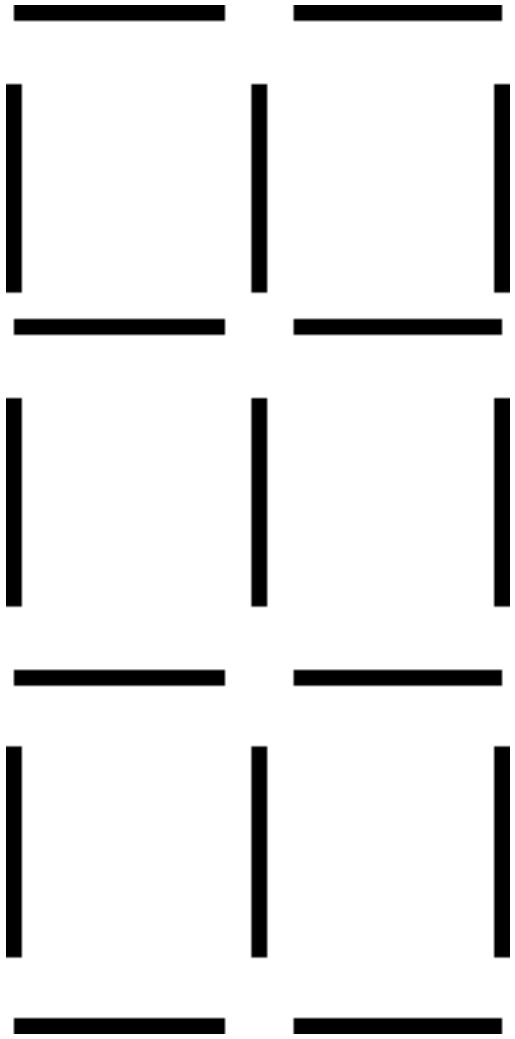
   + TEN

   + TEN

   SIXTY

STATE SPACE REPRESENTATION

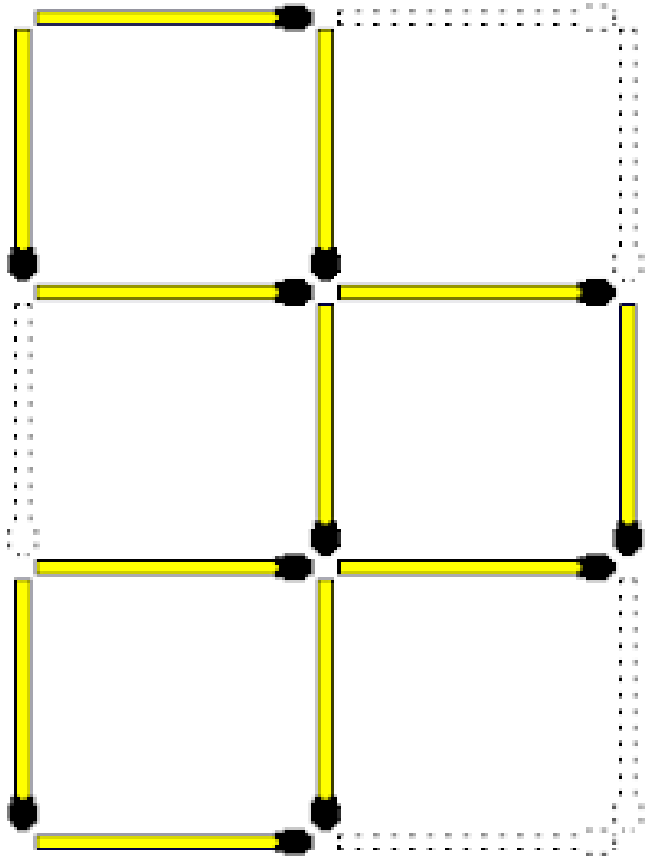
- **State:** mapping from letters to digits
- **Initial State:** empty mapping
- **Operators:** assign a digit to a letter ie no 2 letters have same digit
- **Goal Test:** whether the expression is true given the complete mapping
  ie it represents correct sum
- **Note:** In this problem, the solution is NOT a sequence of actions that transforms the initial state into the goal state; rather, the solution is a goal node that includes an assignment of a digit to each letter in the given problem.
- **APPROACH:** adopt a fixed order, e.g., alphabetical order. A better choice is to do whichever is the most constrained substitution, that is, the letter that has the fewest legal possibilities given the constraints of the puzzle.



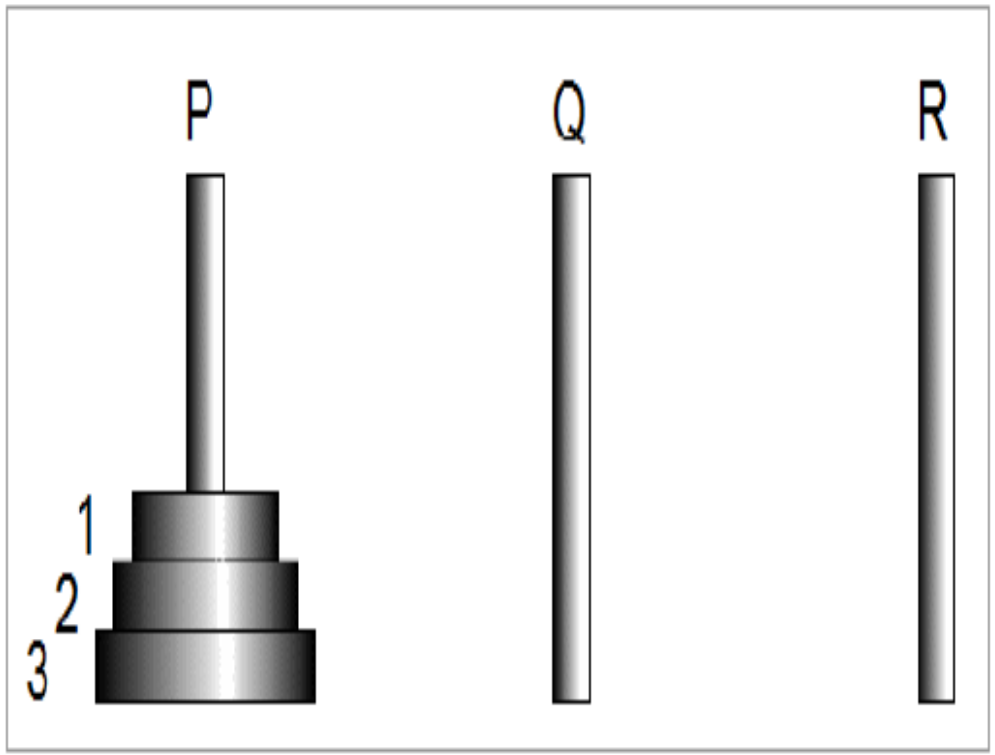
EXAMPLE 8: **REMOVE 5 STICKS**

- Given the following configuration of sticks, remove exactly 5 sticks in such a way that the remaining configuration forms exactly 3 squares.
- **State:** ?
- **Initial State:** ?
- **Operators:** ?
- **Goal Test:** ?

SOLUTION



EXAMPLE 9: TOWER OF HANOI PROBLEM



- **PROBLEM DEFINITION:** There are 3 discs with different diameters. We can slide these discs onto 3 perpendicular rods. It's important that if there is a disc under another one then it must be bigger in diameter. We denote the rods with „P", „Q", and „R", respectively. The discs are denoted by „1", „2", and „3", respectively, in ascending order of diameter. We can slide a disc onto another rod

1. if the disc is on the top of its current rod, and
2. the discs on the goal rod will be in ascending order by size after the replacing.

- Our goal is to move all the discs to rod R.

STATE SPACE REPRESENTATION

- **SATATES:** a state is a vector (a1 , a2 , a3) where ai is the position of disc i (i.e.,either P, Q, or R)
- **Initial state:** Initially, all the discs are on rod P, i. e.: k=(P , P ,P)
- **Goal state:** The goal is to move all the three discs to rod R. So, in this problem, we have only one goal state, namely: C={R, R , R}
- **Set of operators:**

STATE SPACE GRAPH

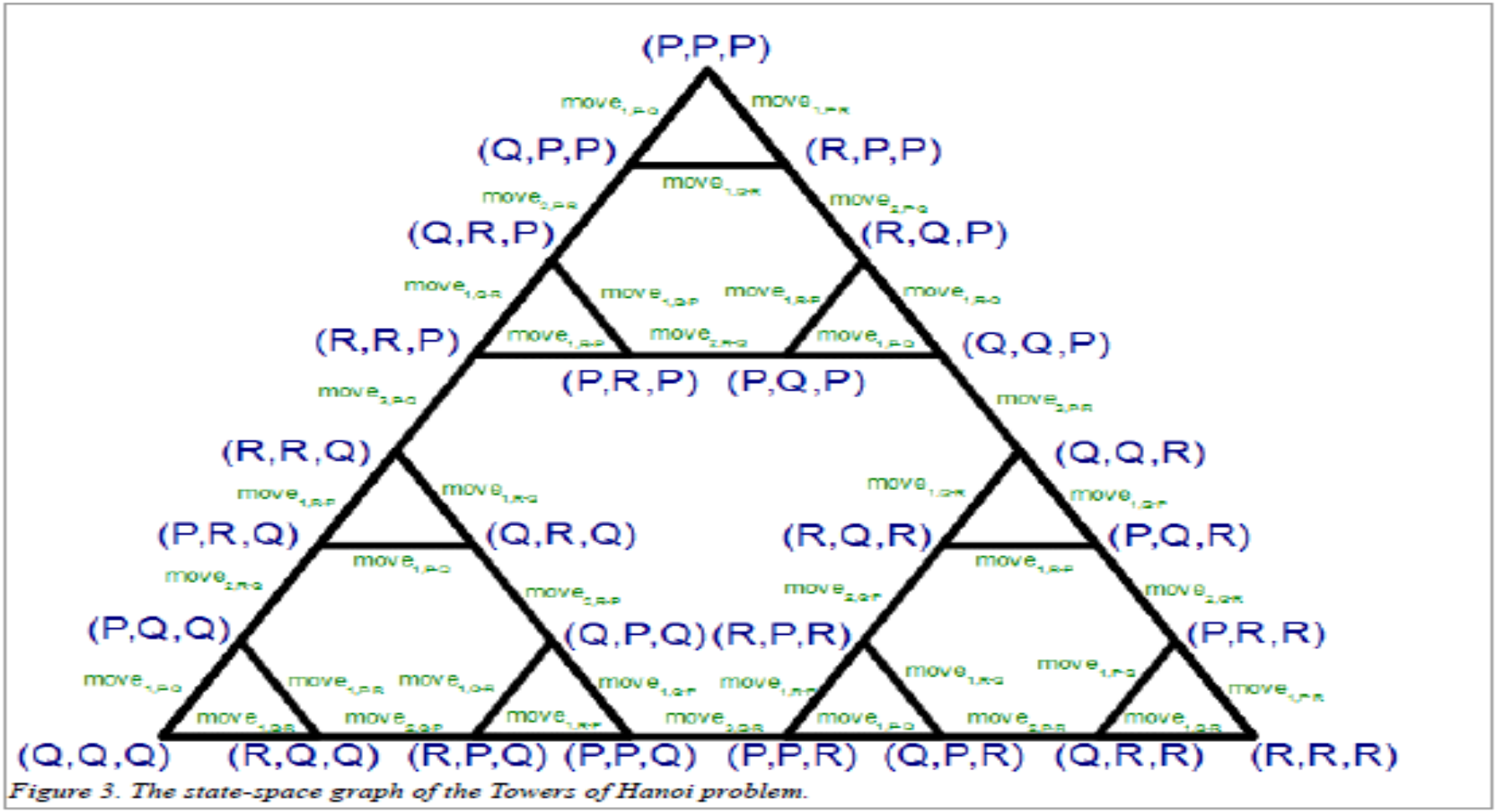
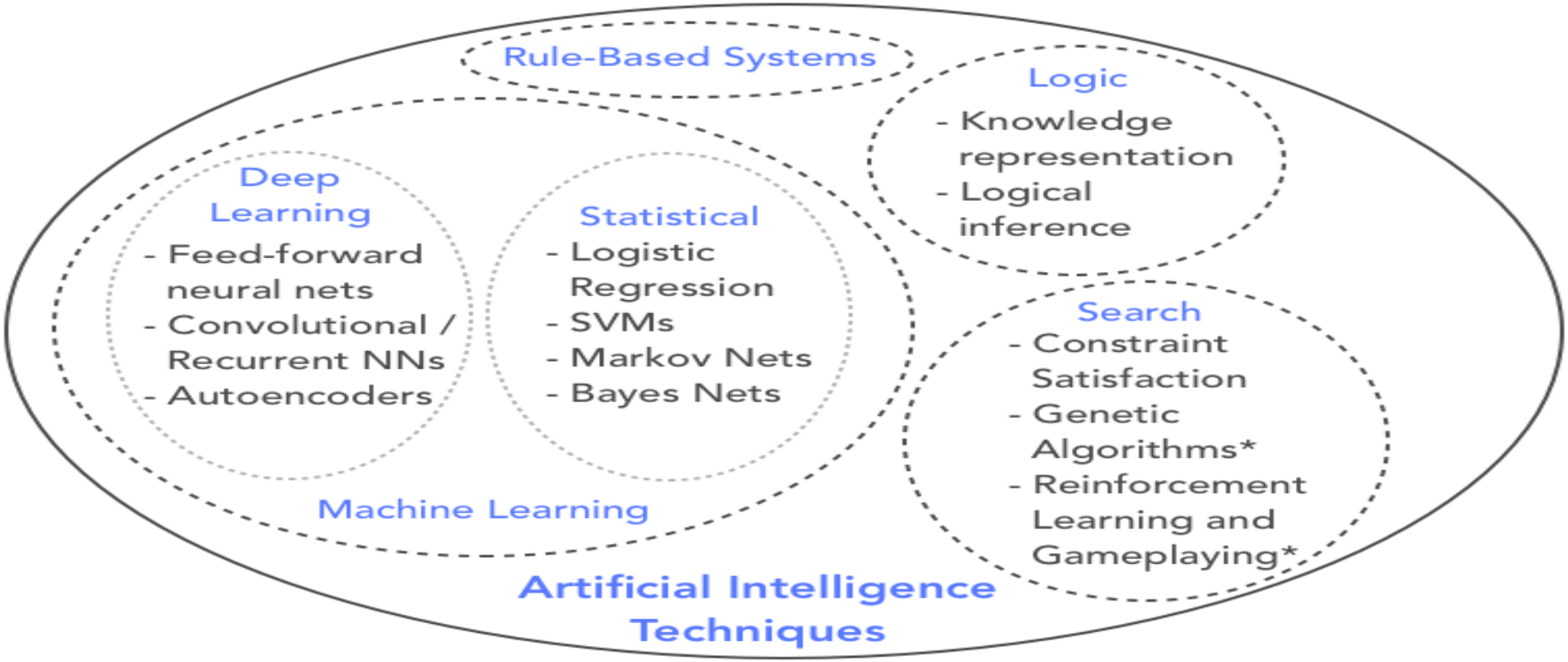


Figure 3. The state-space graph of the Towers of Hanoi problem.

AI TOOLS AND TECHNIQUES



AI TECHNIQUES

1. Machine Learning

- It is one of the applications of AI where machines are not explicitly programmed to perform certain tasks rather they **learn and improve from experience automatically**. Deep Learning is a subset of machine learning based on **artificial neural networks for predictive analysis**. There are various machine learning algorithms such as **Unsupervised Learning, Supervised Learning, and Reinforcement Learning**. In **Unsupervised Learning,** the algorithm does not use classified information to act on it without any guidance. In **Supervised Learning**, it deduces a function from the training data which consists of a set of an input object and the desired output. **Reinforcement learning** is used by machines to take suitable actions to increase the reward to find the best possibility which should be taken in to account.

2. NLP (Natural Language Processing)

- It is the interactions between computers and human language where the computers are programmed to process natural languages. **Machine Learning** is a reliable technology for Natural Language Processing to obtain meaning from human languages. In NLP, the audio of a human talk is captured by the machine. Then the audio to text conversation occurs and then the text is processed where the data is converted into audio. Then the machine uses the audio to respond to humans. Applications of Natural Language Processing can be found in **IVR (Interactive Voice Response)** applications used in call centers, language translation applications like **Google Translate and word processors such as Microsoft Word** to check the accuracy of grammar in text. However, the nature of human languages makes the Natural Language Processing difficult because of the rules which are involved in the passing of information using natural language and they are not easy for the computers to understand. So NLP uses algorithms to recognize and abstract the rules of the natural languages where the unstructured data from the human languages can be converted to a format that is understood by the computer.

AI TECHNIQUES

3. Automation and Robotics

- The purpose of Automation is to get the monotonous and repetitive tasks done by machines which also improve productivity and in receiving cost-effective and more efficient results. Many organizations use machine learning, neural networks, and graphs in automation. Such automation can prevent **fraud issues** while **financial transactions online by using CAPTCHA technology**. Robotic process automation is programmed to perform high volume repetitive tasks which can adapt to the change in different circumstances.

4. Machine Vision

- Machines can capture visual information and then analyze it. Here cameras are used to capture the visual information, the analog to digital conversion is used to convert the image to digital data and digital signal processing is employed to process the data. Then the resulting data is fed to a computer. In machine vision, two vital aspects are sensitivity, which is the ability of the machine to perceive impulses that are weak and resolution, the range to which the machine can distinguish the objects. The usage of machine vision can be found in signature identification, pattern recognition, and medical image analysis, etc.

PROBLEM SOLVING TYPES

Criteria	Supervised Learning	Unsupervised Learning	Reinforcement Learning
Definition	The machine learns by using labeled data	The machine is trained on unlabeled data without any guidance	An agent interacts with its environment by performing actions & learning from errors or rewards
Type of problems	Regression & classification	Association & clustering	Reward-based
Type of data	Labeled data	Unlabeled data	No predefined data
Training	External supervision	No supervision	No supervision
Approach	Maps the labeled inputs to the known outputs	Understands patterns & discovers the output	Follows the trial-and-error method

TASK DOMAINS OF ARTIFICIAL INTELLIGENCE

Task Domain of Artificial Intelligence		
Mundane (Ordinary) Tasks	**Formal Tasks**	**Expert Tasks**
Perception	Mathematics	Engineering
Computer Vision	Geometry	Fault Finding
Speech, Voice	Logic	Manufacturing
Speech, Voice	Integration and Differentiation	Monitoring
Natural Language Processing	Games	Scientific Analysis
Understanding	Go	
Language Generation	Chess (Deep Blue)	
Language Translation	Ckeckers	
Common Sense	Verification	Financial Analysis
Reasoning	Theorem Proving	Medical Diagnosis
Planing		**Creativity**
Robotics		
Locomotive		

HISTORY OF AI

- 1923
- Karel Čapek play named “Rossum's Universal Robots” (RUR) opens in London, first use of the word "robot" in English.
- 1943
- Foundations for neural networks laid.
- 1945
- Isaac Asimov, a Columbia University alumni, coined the term Robotics.
- 1950
- Alan Turing introduced Turing Test for evaluation of intelligence and published Computing Machinery and Intelligence.
- Claude Shannon published Detailed Analysis of Chess Playing as a search.
- 1956
- John McCarthy coined the term Artificial Intelligence. Demonstration of the first running AI program at Carnegie Mellon University.

HISTORY OF AI

- 1958
- John McCarthy invents LISP programming language for AI.
- 1964
- Danny Bobrow's dissertation at MIT showed that computers can understand natural language well enough to solve algebra word problems correctly.
- 1965
- Joseph Weizenbaum at MIT built ELIZA, an interactive problem that carries on a dialogue in English.
- 1969
- Scientists at Stanford Research Institute Developed Shakey, a robot, equipped with locomotion, perception, and problem solving.
- 1973
- The Assembly Robotics group at Edinburgh University built Freddy, the Famous Scottish Robot, capable of using vision to locate and assemble models.
- 1979
- The first computer-controlled autonomous vehicle, Stanford Cart, was built.

HISTORY OF AI

- 1985
- Harold Cohen created and demonstrated the drawing program, Aaron.
- 1990
- Major advances in all areas of AI −
- Significant demonstrations in machine learning
- Case-based reasoning
- Multi-agent planning
- Scheduling
- Data mining, Web Crawler
- natural language understanding and translation
- Vision, Virtual Reality
- Games

HISTORY OF AI

- 1997
- The Deep Blue Chess Program beats the then world chess champion, Garry Kasparov.
- 2000
- Interactive robot pets become commercially available.
- MIT displays Kismet, a robot with a face that expresses emotions.
- The robot Nomad explores remote regions of Antarctica and locates meteorites.

THANK YOU

anooja@somaiya.edu