CS8691
~~CS6659~~ - Artificial Intelligence

Unit - I Introduction to AI and Production systems

Introduction to AI:

Artificial Intelligence (AI) is the study of how to make computers do things which, at the moment, people do better.

Goal: Create system Function Intelligently & Independently

Computer science
/
AI.

1) Human can speak and communicate through language → speech Recognition.
(It is statistical based & it is known as statistical learning)

2) Humans can write & Read text → NLP (or) (Natural Language Processing)

3) Human can see with their eyes & Process what they see → computer vision.
(It falls under the symbolic way to process the information)

4) Human recognitions the scene around them → Image processing.

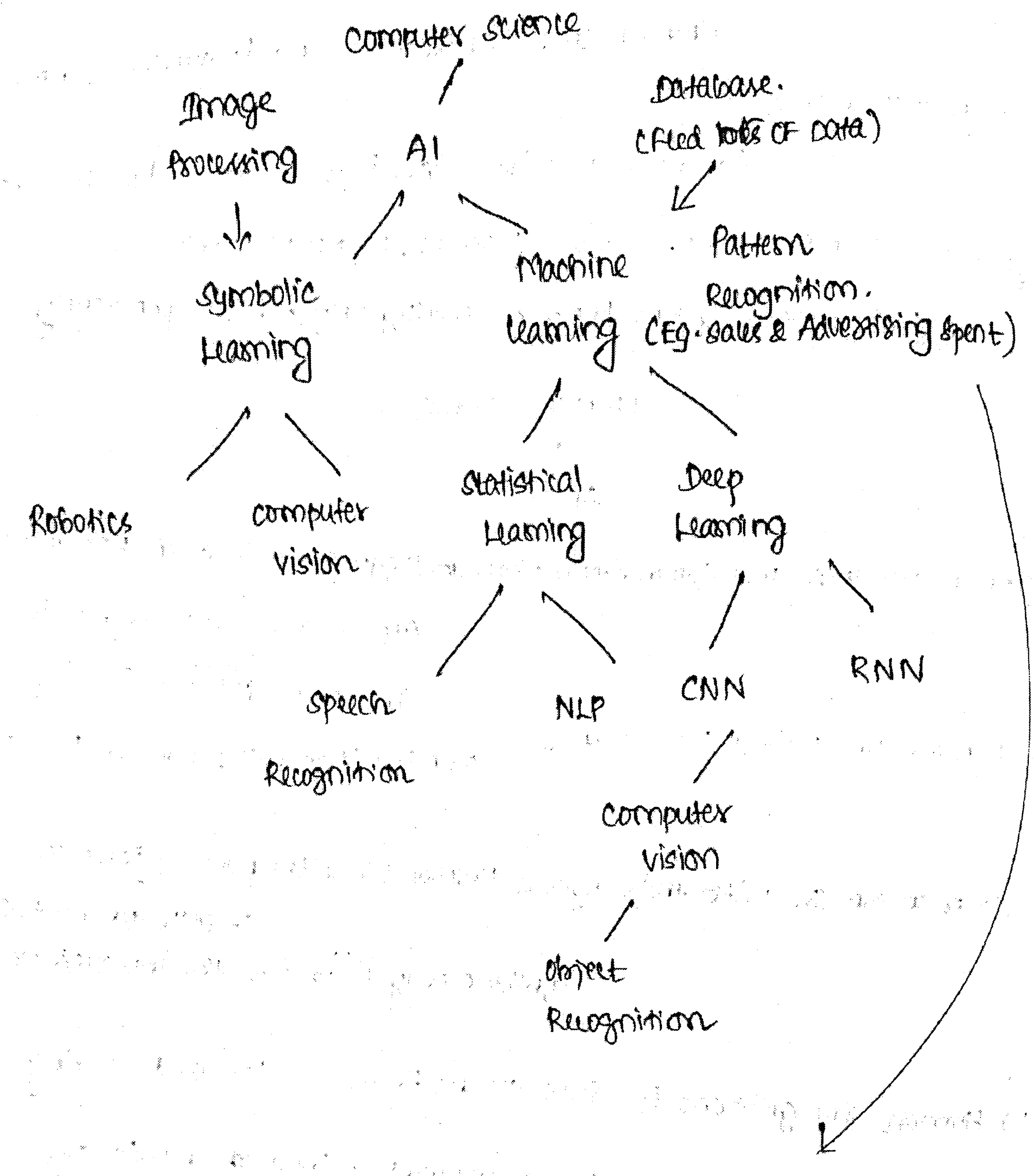
5) Humans can understand the environment & move around fluidly.
↓
Robotics.

6) Humans have the ability to see patterns such as grouping → Pattern Recognition.

7) Human brain is the network of neurons → Neural Networks.
(Complex).

8) Scan images left to right, top to bottom → Conventional neural n/w
(Recognizes Object & scenes)

9) To Remember past → RNN (Recurrent neural n/w).



Easy to Predict based on the data.

Classification
[Use some information about customers]

Prediction.
[Prediction based on data]

Learning. AI

supervised learning ⟶ Training an Algorithm with data [also contain answer]
[Eg. Train a m/c, to recognize ur friends by name, identify them]

unsupervised learning ⟶ Training an Algorithm based on pattern.
[Eg. based on data, figure out pattern [

Reinforcement learning ⟶ Algorithm by goal [Trial & Error]
[Eg: Robo attempts to climb a wall].

Acting Humanly: Turing Test Approach

Turing Test proposed by Alan Turing (1950) was designed to provide a satisfactory operational definition of intelligence. A computer passes the test if a human interrogator, after posing some written questions, cannot tell whether the written responses come from a ~~person~~ person or from a computer.

↳ Need to possess the following capabilities

→ Natural language processing
→ Knowledge Representation
→ Automated Reasoning
→ Machine Learning

Thinking Humanly: The cognitive Modeling Approach

Programs thinks like a human, [inside the actual workings of human minds]

Ways:

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.

Thinking Rationally: The "laws of thought" Approach

"Right thinking" set of rules.

His syllogisms provided patterns for argument structures "Socrates is a man; all men are mortal; therefore Socrates is mortal."

Laws of thought were supposed to govern the operation of the mind; their study initiated the field called logic

Artificial Intelligence definition is organized in four categories.

Thinking Humanly

"The exciting new effort to make computers think... machines with minds", in the full and literal sense" (Haugeland, 1985)

"[The automation of] activities that we associate with human thinking, activities such as decision-making, problem solving, learning" ... (Bellman 1978)

Thinking Rationally (Thinking Right way at the right time)

"The study of mental faculties through the use of computational models" (Charniak & McDermott, 1985)

"The study of computations that make it possible to perceive, reason and act" (Winston 1992).

Acting Humanly

"The art of creating machines that perform functions that require intelligence when performed by people" (Kurzweil, 1990)

"The study of how to make computers do things at which, at the moment, people are better" (Rich and Knight, 1991)

Acting Rationally

"Computational Intelligence is the study of the design of intelligent agents" (Poole et al., 1998)

"AI ... is concerned with intelligent behaviour in Artifacts (Nilsson 1998)

Acting Rationally: The Rational Agent Approach

An agent is just something that acts (agent comes from the Latin agere, to do). Of course, all computer programs do something, but computer agents are expected to do more: operate autonomously, perceive their environment, persist over a prolonged time period, adapt to change, & create and pursue goals.

A Rational Agent is one that acts so as to achieve the best outcome or, when there is uncertainty, the best expected outcome.

Agents & Environments

An agent is anything that can be viewed as perceiving its environment through sensors & acting upon that environment through actuators.

Human agent has Eyes, Ears & other organs for sensors
Hands, legs, vocal tract & so on for Actuators.

Robotics → cameras & infrared range finders for sensors
Various motors for Actuators.

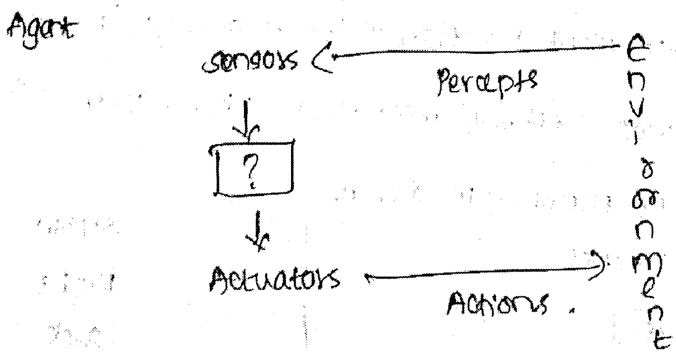


Fig. Agents interact with environments through sensors & Actuators.

Software Agent receives keystrokes, file contents, & n/w packets as sensory inputs & acts on the environment by displaying on the screen, writing files, & sending n/w packets.

FUTURE OF AI:

The Field of artificial Intelligence or AI, attempts not just to understand but also to build intelligent entities. Artificial Intelligence is a blooming technological advancement from software's like Alexa to basic human command system.

Every aspect of these technologies has been put to its best use. It is now turning a basic part of human lives. people need this technology in enterprises and even households. Artificial Intelligence has made its way through all sectors of society. Every person is using this technology in some or the other way. Moreover it is modified for different age groups and Industrial sectors.

The future of artificial Intelligence is more fascinating than ever. people seem to have indulged in this debate for years now. Some say robot's are the future and they will replicate humans completely. Another opinion is a human dependency on this sector will increase manifold. However, one thing is for sure, artificial intelligence is progressing faster than ever, and no one knows what migit come next.

Amongst all the debate, there are some predicitions that one can consider:

Prediction 1: Universal Usage

Artificial Intelligence is already a part of various everyday activities. The future might make humans completely dependent on this technology. For eg. self-driving cars are being developed with artificial intelligence. There is no limit to this technology, if it wants to make the human experience easier.

Prediction 2: AI and Robotics will Merge

Many have predicted that by applying this technology in robots multiple new inventions can be created. There are many effects this can have on humans. Like, robotics will completely alter the medical stream, Probably being able to cure anything easily.

Prediction 3: Simpler Work Environment

Unlike any other predictions, this one says that artificial intelligence will only enhance your work environment. It will make tedious tasks simpler and humans will be able to focus on key tasks. This will help in utilizing human resource to its best potential.

Prediction 4: Ongoing Maintenance

Ongoing maintenance of machinery is a huge expence for manufactures and the shift from reactive to predictive maintenance has become a must for all manufactures. By using advanced AI algorithms and artificial neural networks to formulate predictions regarding asset malfunction and briefing technicians ahead of time, AI has managed to save bussinesses valuable time and resources.

In addition, Predictive maintenance has helped extend the life of machines and has resulted in an overall reduction in labor costs.

Therefore, the future can be infinite and the changes can be unpredictable. Although, one thing is for sure this technology will change the dynamics of our lives.

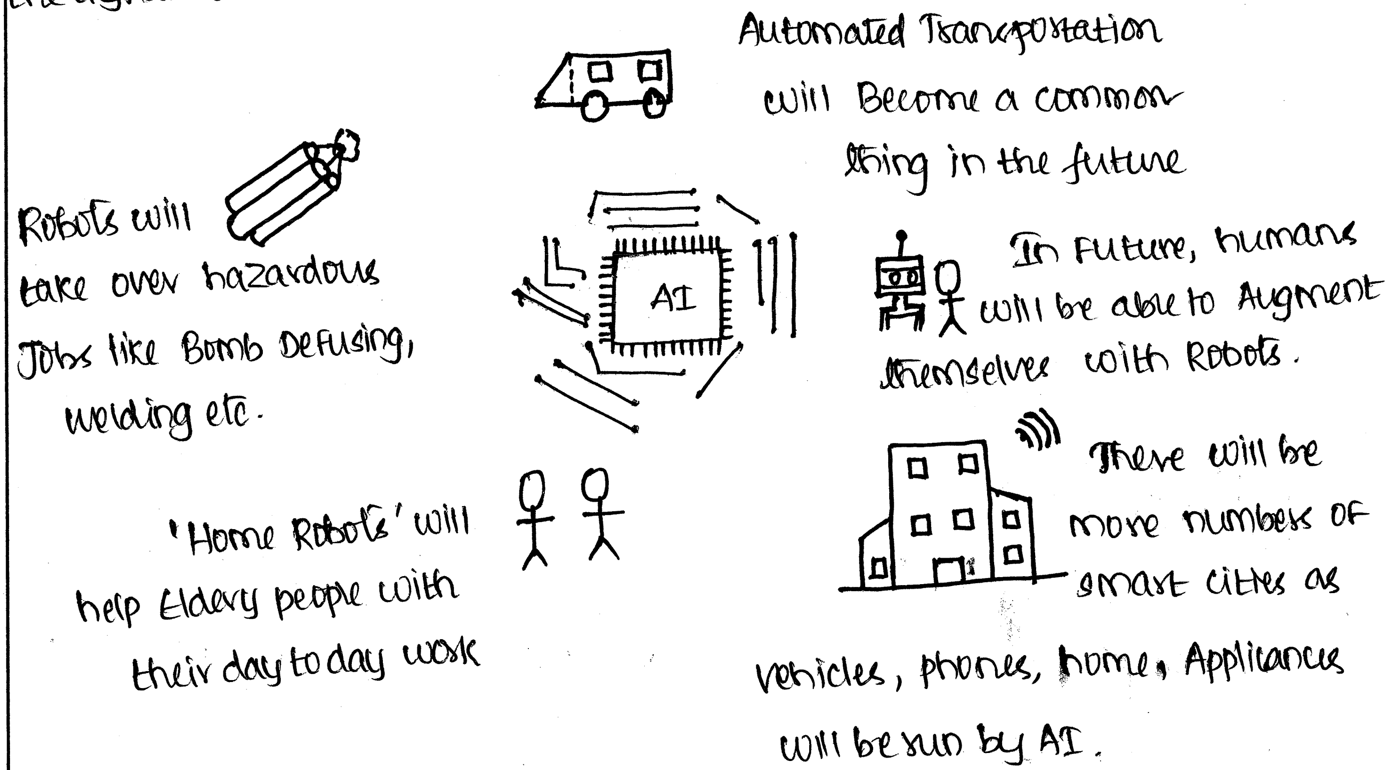


fig. Future of Artificial Intelligence.

Characteristics of Intelligent Agents:-

• An Agent is something that perceives and acts in an environment

An Agent can be viewed as perceiving its environment through sensors and acting upon that environment through actuators. The Agent Function for an agent specifies the action taken by the agent in response to any percept sequence.

The Agent Program implements the Agent Function. There exists a variety of basic agent-Program designs reflecting the kind of information made explicit and used in the decision process. The designs vary in efficiency, compactness and flexibility. The appropriate design of the agent program depends on the nature of the environments.

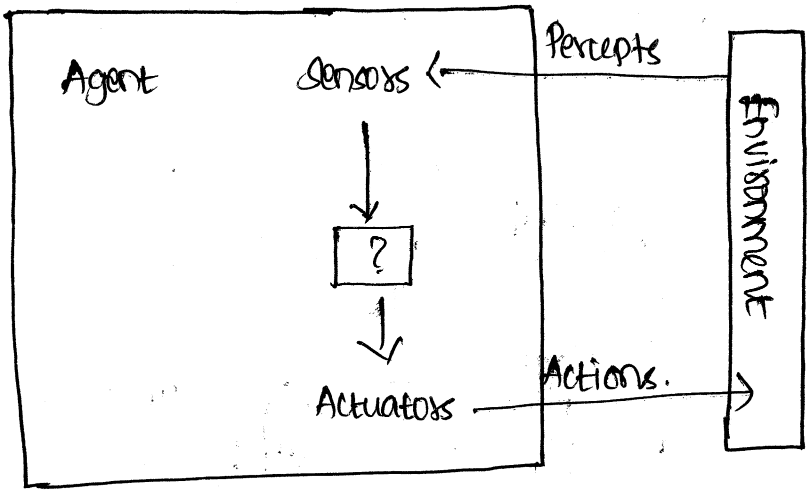


Fig. Agents Interact with Environments through sensors and Actuators

Intelligent Agent:

An Intelligent Agent is an autonomous entity which act upon an environment using sensors and actuators for achieving goals. An Intelligent Agent may learn from the environment to achieve their goals. A thermostat is an example of an intelligent Agent.

Characteristics

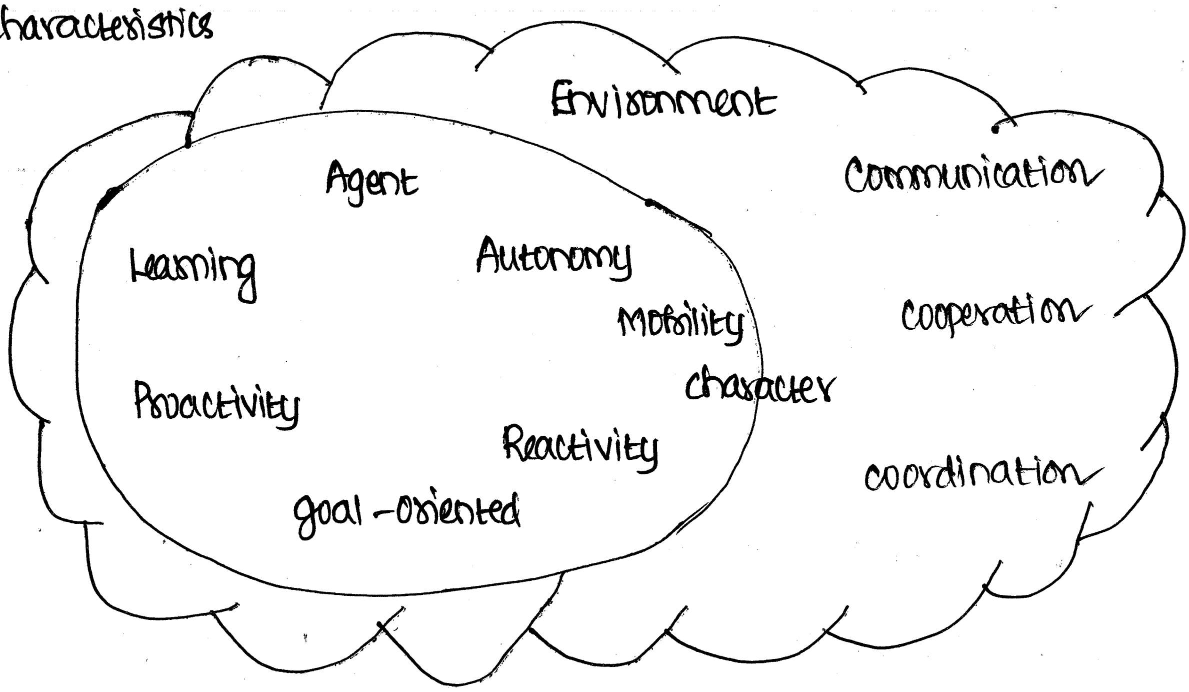


Fig. Characteristics of Intelligent Agent

1. Autonomy (Empowerment)

Agent takes initiative, exercises control over its actions. They are

* Goal-Oriented
* Collaborative
* Flexible
* Self-starting

An Agent must have both control over its actions and internal states. The degree of the agent's autonomy can be specified. There may need intervention from the user only for important decisions.

2. Learning/reasoning:

An Agent has the ability to learn from previous experience and to successively adapt its own behavior to the environment.

3. Reactivity:

An Agent must be capable of reacting appropriately to influences or information from its environment.

4. Goal-Orientedness:

An Agent has well-defined goals and gradually influence its environment and so achieve its own goals.

5. ProActivity:

In some cases agents are so programmed that they can take initiative by themselves in order to achieve their objectives or to move forward in that direction. (Agent exhibit goal-oriented behavior by acting on their own initiative)

6. Temporarily continous:

Agents constantly executing in Runtime Environment

7. Mobility: The ability of an agent to move around in an environment

8. Veracity: An Agent will not knowingly communicate false information

9. Benevolence: Agents do not have conflicting goals, and that every agent will therefore always try to do what is asked of it.

10. Social Ability: Agents must have mechanisms to communicate with other objects (humans or agents) if they need to. This property is of vital importance when we talk about collaborative Agents.

11. Agents also cooperate with other agents to carry out more complex tasks than they themselves can handle.

12. Agent is able to communicate with other entities (agents, systems, humans) for its tasks
→ Info Exchange, Task delegation, Negotiation.

Specifying the Task Environment

The Task Environment which are essentially the "Problems" to which rational agents are the "solutions". The performance measure, the environment, and the agent's actuators and sensors, can be grouped under the heading of the task Environment, it is known as PEAS (Performance, Environment, Actuators, Sensors).

Agent Type	Performance measure	Environment	Actuators	Sensors
Taxi Driver	Safe, fast, legal comfortable trip, maximize profits	Roads, other traffic, pedestrians, customers	Steering, accelerator, brake, signal, horn, display	Cameras, sonar, speedometer, GPS, odometer, accelerometer, engine sensors, keyboard
Medical Diagnosis System	Healthy patient, reduced costs	Patient, Hospital, staff	Display of questions, tests, diagnoses, treatments, referrals	Keyboard entry of symptoms, findings, patient's answers.
Satellite Image Analysis System	Correct Image categorization	Downlink from orbiting satellite	Display of scene categorization	Color pixel arrays.
Part-Picking Robot	Percentage of parts in correct bins	Conveyor belt with parts; bins	Jointed arm and hand	Camera, Joint Angle sensors
Refinery Controller	Purity, yield, safety	Refinery Operators	Valves, Pumps, heaters, displays	Temperature, Pressure, Chemical sensors
Interactive English Tutor	Student's score on Test	Set of students, Testing Agency	Display of exercises, suggestions, corrections	Keyboard Entry

Fig. Examples of Agent Types and their PEAS Description

Properties of Task Environments

The range of task environments that might arise in AI is obviously vast. However, to identify a fairly small number of dimensions along which task environments can be categorized.

1. Fully observable Vs. Partially observable

If an agent's sensors give it access to the complete state of the environment at each point in time, then we say that the task environment is Fully observable. A task environment is effectively fully observable if the sensors detect all aspects that are relevant to the choice of action.

An environment might be partially observable because of noisy and inaccurate sensors or because parts of the state are simply missing from the sensor data Eg. A vacuum Agent with only a local dirt sensor cannot tell whether there is dirt in other squares and an automated taxi cannot see what other drivers are thinking.

If the agent has no sensors at all then the environment is unobservable.

2. Single Agent Vs. Multi Agent

The distinction between single-agent and Multi-Agent environment may seem simple enough. For eg. An Agent solving a crossword puzzle by itself is clearly in a single-Agent Environment.

An Agent playing chess is in a two-Agent Environment. In Chess, the opponent entity B is trying to maximize its performance measure. Thus chess is a Competitive MultiAgent Environment.

In a taxi-driving environment, on the other hand, avoiding collisions maximizes the performance measures of all agents, so it is a Partially Cooperative MultiAgent Environment.

3. Deterministic Vs Stochastic

IF the next state of the environment is completely determined by the current state and the action executed by the agent, then the environment is deterministic. (Eg. Vacuum World)

IF the next state of the environment cannot be determined by the current state, then the environment is stochastic (Eg. Automated Taxi Driver)

A Environment is said to be uncertain if it is not fully observable or not deterministic. A Nondeterminstic environment is one in which actions are characterized by their possible outcomes, but no probabilities are attached to them.

4. Episodic Vs. Sequential

In an episodic task environment, the agent's experience is divided into atomic episodes. In each episode the agent receives a percept and then performs a single action. Crucially, the next episode does not depend on the actions taken in previous episodes.

<u>Eg</u>. An agent that has to spot defective parts on an assembly line bases each decision on the current part, regardless of previous

decisions;

In sequential environments, on the other hand, the current decision could affect all future decisions.

Eg: Chess and Taxi Driving are sequential;

Episodic Environments are much simpler then sequential environments because the agent does not need to think ahead.

5. Static Vs. Dynamic

If the environment can change while an agent is deliberating, then the environment is dynamic for that agent; otherwise it is static;

Static Environments are easy to deal with because the agent need not keep looking at the world while it is deciding on an action. Dynamic Environments on the other hand are continously asking the agent what it wants to do;

If the environment itself does not change with the passage of time but the agent's performance score does, then the environment is said to be semidynamic.

Eg: Taxi Driving is clearly Dynamic (other cars and the taxi itself keep moving while the driving algorithm dithers what to do next)

Chess, when played with a clock is semidynamic.

Crossword puzzles are static

6. Discrete Vs. Continous

The discrete/continous distinction applies to the state of the environment, to the way time is handled, and to the percepts and actions of the Agent.

Eg. Chess Environment has a finite number of distinct states (Excluding the clock) chess also has a discrete set of percepts and actions.

Taxi Driving is a continous state and continous-time problem.

Input from Digitial cameras is Discrete, strictly speaking but is typically treated as representing continously varying Intensities and locations.

7. Known Vs. unknown

This distinction refers not to the environment itself but to the agent's state of knowledge about the environment. In a known environment, the outcomes for all the actions are given. Obviously, if the environment is unknown the agent will have to learn how it works in order to make a good decisions.

Eg: In solitaire card games, the rules of the game is known but unable to see the cards that have not yet been turned over.

An unknown environment can be fully observable - in a new video game, the screen may show the entire game state but I still don't know what the buttons do until I try them.

Task Environment	Observable	Agents	Deterministic	Episodic	Static	Discrete
Crossword Puzzle	Fully	Single	Deterministic	Sequential	Static	Discrete
Chess with a clock	Fully	Multi	Deterministic	Sequential	Semi	Discrete
Poker	Partially	Multi	Stochastic	Sequential	Static	Discrete
Backgammon	Fully	Multi	Stochastic	Sequential	Static	Discrete
Taxi Driving	Partially	Multi	Stochastic	Sequential	Dynamic	Continous
Medical Diagnosis.	Partially	Single	Stochastic	Sequential	Dynamic	Continous
Image Analysis	Fully	Single	Deterministic	Episodic	Semi	Continous
Part-Picking Robot	Partially	Single	Stochastic	Episodic	Dynamic	Continous
Refinery Controller	Partially	Single	Stochastic	Sequential	Dynamic	Continous
Interactive English Tutor	Partially	Multi	Stochastic	Sequential	Dynamic	Discrete

Fig. Examples of Task Environments and their characteristics

Problem Characteristics:

Heuristic search is a very general method applicable to a large class of problem. It includes a variety of techniques. In order to choose an appropriate method, it is necessary to analyze the problem with respect to the following considerations.

1. Is the Problem Decomposable?

A very large and composite problem can be easily solved if it can be broken into smaller problems and recursion could be used.

<u>Eg</u>: $\int x^2 + 3x + \sin 2x \cos 2x \, dx$.

This can be done by breaking it into three smaller problems and solving each by applying specific rules.

2. Can solution steps be Ignored or undone?

Problem fall under three classes Ignorable, Recoverable and Irrecoverable. This classification is with reference to the steps of the solution to a problem.

<u>Eg</u>: 1. Ignorable Problems Eg: Theorem Proving
(In which solution steps can be Ignored

2. Recoverable Problems Eg: 8 Puzzle
(In which solution steps can be undone

3. Irrecoverable Problems Eg: Chess
(In which solution steps can't be undone)

3. Is the Universal Predictable?

Problems can be classified into those with certain outcome (Eight Puzzle and Water Jug) and those with uncertain outcome (Playing cards)

Incertain outcome Problems planning could be done to generate a sequence of operators that guarantees to a lead solution.

Uncertain outcome Problems planning can be at best to generate a sequence of operators that has a good probability of leading to a solution.

4. Is good solution Absolute or relative? (Is the solution a state or a Path?)

Two categories of problems. In one like the Water Jug and 8 Puzzle Problems ~~the~~ solutions ~~is~~ are satisfied, unmindful of the solution path taken, whereas in the other category not just any solution is acceptable (Eg. Travelling salesman)

5. The knowledge Base Consistent?

In some Problems the knowledge base is consistent and in some it is not. For eg: Evaluation of Boolean Expression

(knowledge Base now contains theorems and laws of Boolean Algebra which are always true)

6. What is the role of knowledge?

Though one could have unlimited computing power, the size of the knowledge base available for solving the problem does matter in arriving at a good solution. Eg: Playing chess, just the rules for determining legal moves and control, but additional knowledge will give good strategy and tactics to solve.

Eg: 1. Playing chess

2. Newspaper Understanding

7. Does the Task Requires Interaction with human?

The Problems can be categorized under two heads.

a) Solitary in which the computer will be given a problem description and will Produce an answer, with no intermediate communication

Eg: Theorem Proving (give basic Rules and laws to computer)

b) Conversational, in which there will be intermediate communication between a person and the computer, whether to provide additional assistance to the computer or to provide additional information to User.

Eg: Medical Diagnosis Problem

8. Problem classification?

Actual Problems are examined from the point of view, the task here is examine an input and decide what of a set of known classes.

Eg: Problems such as Medical Diagnosis Engineering Design

Analyse the following problems with respect to the seven Problem characteristics

1. Travelling Salesman Problem
2. Chess
3. Water Jug
4. 8-Puzzle
5. Missionaries and cannibals
6. Tower of Hanoi

Analyze each of them with respect to the seven problem characteristics

- **Chess**
- **Water jug**
- **8-puzzle**
- **Traveling salesman**
- **Missionaries and cannibals**
- **Tower of Hanoi**

1. **Chess**

Problem characteristic	**Satisfied**	**Reason**
Is the problem decomposable?	No	One game have Single solution
Can solution steps be ignored or undone?	No	In actual game(not in PC) we can't undo previous steps
Is the problem universe predictable?	No	Problem Universe is not predictable as we are not sure about move of other player(second player)
Is a good solution absolute or relative?	absolute	Absolute solution : once you get one solution you do need to bother about other possible solution. Relative Solution : once you get one solution you have to find another possible solution to check which solution is best(i.e low cost). By considering this **chess is absolute**
Is the solution a state or a path?	Path	Is the solution a state or a path to a state? – For natural language understanding, some of the words have different interpretations .therefore sentence may cause ambiguity. To solve the problem we need to find interpretation only , the workings are not necessary (i.e path to solution is not necessary) So In chess winning state(goal state) describe path to state
What is the role of knowledge?		lot of knowledge helps to constrain the search for a solution.
Does the task require human-interaction?	No	Conversational In which there is intermediate communication between a person and

Problem characteristic	satisfied	Reason
		the computer, either to provide additional assistance to the computer or ***to provide additional information to the user***, or both. In chess additional assistance is not required

2. Water jug

Problem characteristic	**Satisfied**	**Reason**
Is the problem decomposable?	No	One Single solution
Can solution steps be ignored or undone?	Yes	
Is the problem universe predictable?	Yes	Problem Universe is predictable bcz to slove this problem it require only one person .we can predict what will happen in next step
Is a good solution absolute or relative?	absolute	**Absolute solution** , water jug problem may have number of solution , bt once we found one solution,no need to bother about other solution **Bcz it doesn't effect on its cost**
Is the solution a state or a path?	Path	Path to solution
What is the role of knowledge?		lot of knowledge helps to constrain the search for a solution.
Does the task require human-interaction?	Yes	additional assistance is required. Additional assistance, like to get jugs or pump

3. 8 puzzle

Problem characteristic	**Satisfied**	**Reason**
Is the problem decomposable?	No	One game have Single solution
Can solution steps be ignored or undone?	Yes	We can undo the previous move

Problem characteristic	satisfied	Reason
Is the problem universe predictable?	Yes	Problem Universe is predictable bcz to slove this problem it require only one person .we can predict what will be **position of blocks** in next move
Is a good solution absolute or relative?	absolute	Absolute solution : once you get one solution you do need to bother about other possible solution. Relative Solution : once you get one solution you have to find another possible solution to check which solution is best(i.e low cost). By considering this 8 puzzle **is absolute**
Is the solution a state or a path?	Path	Is the solution a state or a path to a state? – For natural language understanding, some of the words have different interpretations .therefore sentence may cause ambiguity. To solve the problem we need to find interpretation only , the workings are not necessary (i.e path to solution is not necessary) So In 8 puzzle winning state(goal state) describe path to state
What is the role of knowledge?		lot of knowledge helps to constrain the search for a solution.
Does the task require human-interaction?	No	Conversational In which there is intermediate communication between a person and the computer, either to provide additional assistance to the computer or ***to provide additional information to the user***, or both. In 8 puzzle additional assistance is not required

4. **Travelling Salesman (TSP)**

Problem characteristic	**Satisfied**	**Reason**
Is the problem decomposable?	No	One game have Single solution
Can solution steps be ignored or undone?	Yes	
Is the problem universe	Yes	

Problem characteristic	satisfied	Reason
predictable?		
Is a good solution absolute or relative?	absolute	Absolute solution : once you get one solution you do need to bother about other possible solution. Relative Solution : once you get one solution you have to find another possible solution to check which solution is best(i.e low cost). By considering this TSP **is absolute**
Is the solution a state or a path?	Path	Is the solution a state or a path to a state? – For natural language understanding, some of the words have different interpretations .therefore sentence may cause ambiguity. To solve the problem we need to find interpretation only , the workings are not necessary (i.e path to solution is not necessary) So In TSP (goal state) describe path to state
What is the role of knowledge?		lot of knowledge helps to constrain the search for a solution.
Does the task require human-interaction?	No	Conversational In which there is intermediate communication between a person and the computer, either to provide additional assistance to the computer or ***to provide additional information to the user***, or both. In chess additional assistance is not required

5. Missionaries and cannibals

Problem characteristic	**Satisfied**	**Reason**
Is the problem decomposable?	No	One game have Single solution
Can solution steps be ignored or undone?	Yes	
Is the problem universe predictable?	Yes	Problem Universe is not predictable as we are not sure about move of other

Problem characteristic   satisfied   Reason

		player(second player)
Is a good solution absolute or relative?	absolute	Absolute solution : once you get one solution you do need to bother about other possible solution. Relative Solution : once you get one solution you have to find another possible solution to check which solution is best(i.e low cost). By considering this **is absolute**
Is the solution a state or a path?	Path	Is the solution a state or a path to a state? – For natural language understanding, some of the words have different interpretations .therefore sentence may cause ambiguity. To solve the problem we need to find interpretation only , the workings are not necessary (i.e path to solution is not necessary) So In winning state(goal state) describe path to state
What is the role of knowledge?		lot of knowledge helps to constrain the search for a solution.
Does the task require human-interaction?	Yes	Conversational In which there is intermediate communication between a person and the computer, either to provide additional assistance to the computer or ***to provide additional information to the user***, or both. In chess additional assistance is required to move **Missionaries to other side of river of other assistance is required**

6. Tower of Hanoi

Problem characteristic	**Satisfied**	**Reason**
Is the problem decomposable?	No	One game have Single solution
Can solution steps be ignored or undone?	Yes	
Is the problem universe predictable?	Yes	
Is a good solution absolute or	absolute	Absolute solution : once you get one

Problem characteristic	satisfied	Reason
relative?		solution you do need to bother about other possible solution. Relative Solution : once you get one solution you have to find another possible solution to check which solution is best(i.e low cost). By considering this Tower of Hanoi is **absolute**
Is the solution a state or a path?	Path	Is the solution a state or a path to a state? – For natural language understanding, some of the words have different interpretations .therefore sentence may cause ambiguity. To solve the problem we need to find interpretation only , the workings are not necessary (i.e path to solution is not necessary) So In tower of Hanoi winning state(goal state) describe path to state
What is the role of knowledge?		lot of knowledge helps to constrain the search for a solution.
Does the task require human-interaction?	No	Conversational In which there is intermediate communication between a person and the computer, either to provide additional assistance to the computer or ***to provide additional information to the user***, or both. In tower of Hanoi additional assistance is not required

Typical Intelligent Agents:

The Job of AI is to design an agent Program that implements the agent function - the mapping from percepts to actions. This program will run on some sort of computing device with physical sensors and actuators - we call this architecture

Agent = Architecture + Program

~~Fou~~ Five Basic kinds of Agent programs that embody the Principles underlying almost all Intelligent Systems:

1. Simple Reflex Agents
2. Model-Based Reflex Agents
3. Goal-Based Agents
4. Utility-Based Agents
5. Learning Agents

SIMPLE REFLEX AGENTS

The simplest kind of agent is the simple Reflex Agent. These agents select actions on the basis of the current percept, ignoring the rest of the percept history. Vacuum Agent is an Example of simple Reflex Agent, because its decision is based only on the current location and on whether that location contains dirt.

It follows a set of condition-action rules.

IF condition then action Rule:

<u>Eg:</u> IF car-in-front-is-braking then initiate-braking.

function SIMPLE-REFLEX-AGENT (percept) returns an action

Persistent : rules, a set of condition-action rules

state ← INTERPRET-INPUT (percept)

rule ← RULE-MATCH (state, rules)

action ← rule.ACTION

return action

Fig. A simple Reflex Agent - It acts according to a rule whose condition matches the current state, as defined by the Percept.

This Agent function only succeeds when the environment is fully observable. For simple Reflex Agents operating in partially observable environments, infinite loops are often unavoidable. It may be possible to escape from infinite loops if the agent can randomize its actions.

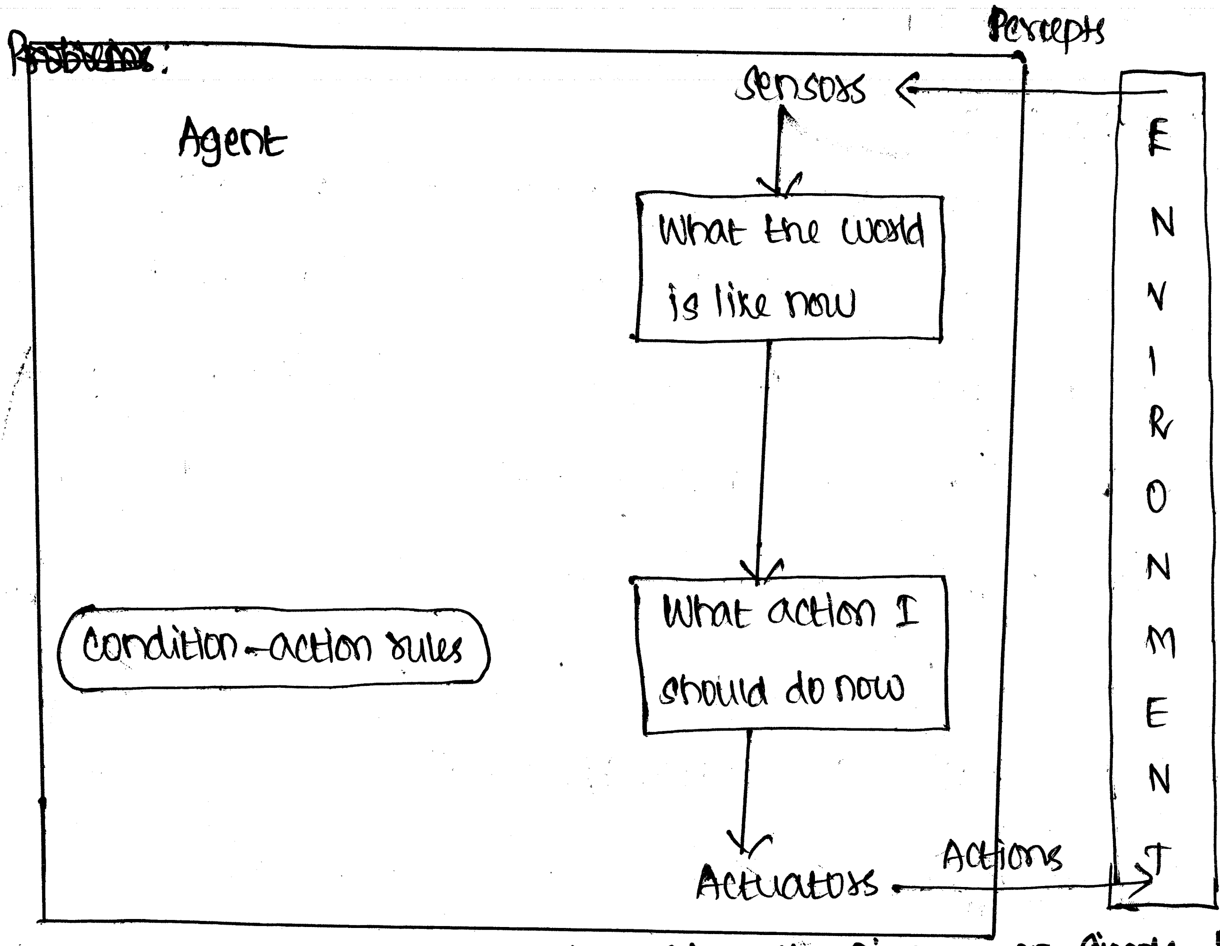


Fig. Schematic Diagram of Simple-Reflex Agent

Problems:

- → Very Limited Intelligence
- → No knowledge of non-perceptual parts of state
- → Usually too big to generate and store
- → If there occurs any changes in the environment, then the collection of rules need to be updated.

MODEL BASED REFLEX AGENTS

The most effective way to handle partial observability is for the agent to keep track of the part of the world it can't see now. (i.e) the agent should maintain some sort of Internal state that depends on the Percept history and thereby reflects at least some of the unobserved aspects of the current state.

```
function MODEL-BASED-REFLEX-AGENT(percept) returns an action
  Persistent: state, the agent's current conception of the world state
              model, a description of how the next state depends on
              current state and action
              rules, a set of condition-action rules
              action, the most recent action, initially none
  state ← UPDATE-STATE(state, action, percept, model)
  rule ← RULE-MATCH(state, rules)
  action ← rule.ACTION
  return Action
```

Fig. A model-based reflex agent. It keeps track of the current state of the world, using an internal model. It then chooses an action in the same way as the Reflex Agent.

It works by finding a rule whose condition matches the current situation. A model-based agent can handle partially observable environments by use of model about the world. The current state is stored inside the agent which maintains some kind of structure describing the part of the world which cannot be seen.

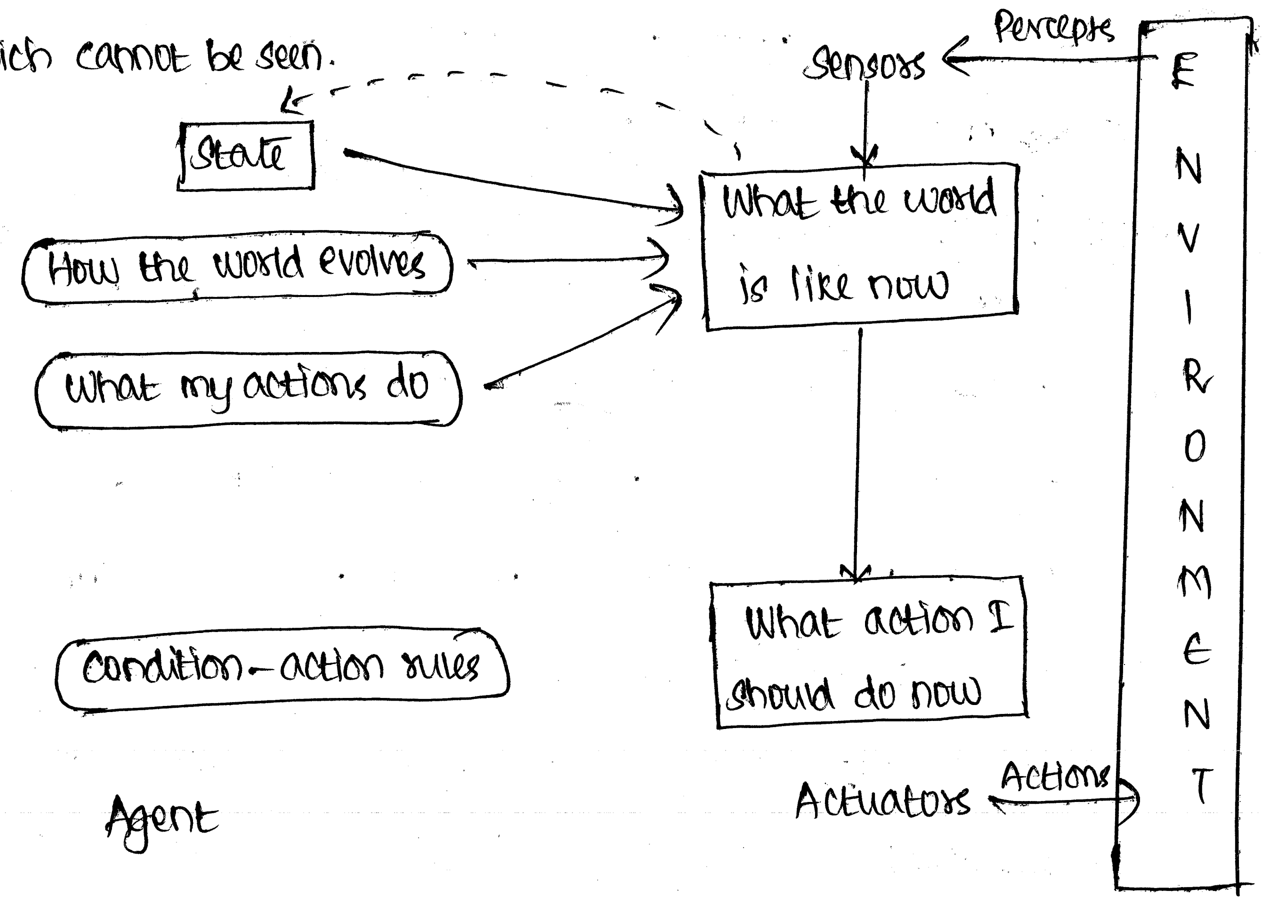


Fig. A Model Based Reflex Agents

Updating the state requires information about:

* how the world evolves in-dependently from the agent, and
* how the agent actions affects the world.

GOAL BASED AGENTS

Knowing something about the current state of the environment is not always enough to decide what to do. For eg. At a road junction, the taxi can turn left, turn right, or go straight on. The correct decision depends on where the taxi is trying to get to.

The Agent Program can combine this with the model (the same information as was used in the model-based reflex Agent) to choose actions that achieve the goal.

These kind of agents take decision based on how far they are currently from their goal (description of desirable situations). Their every action is intended to reduce its distance from the goal. This allows the agent a way to choose among multiple possibilities, selecting the one which reaches a goal state.

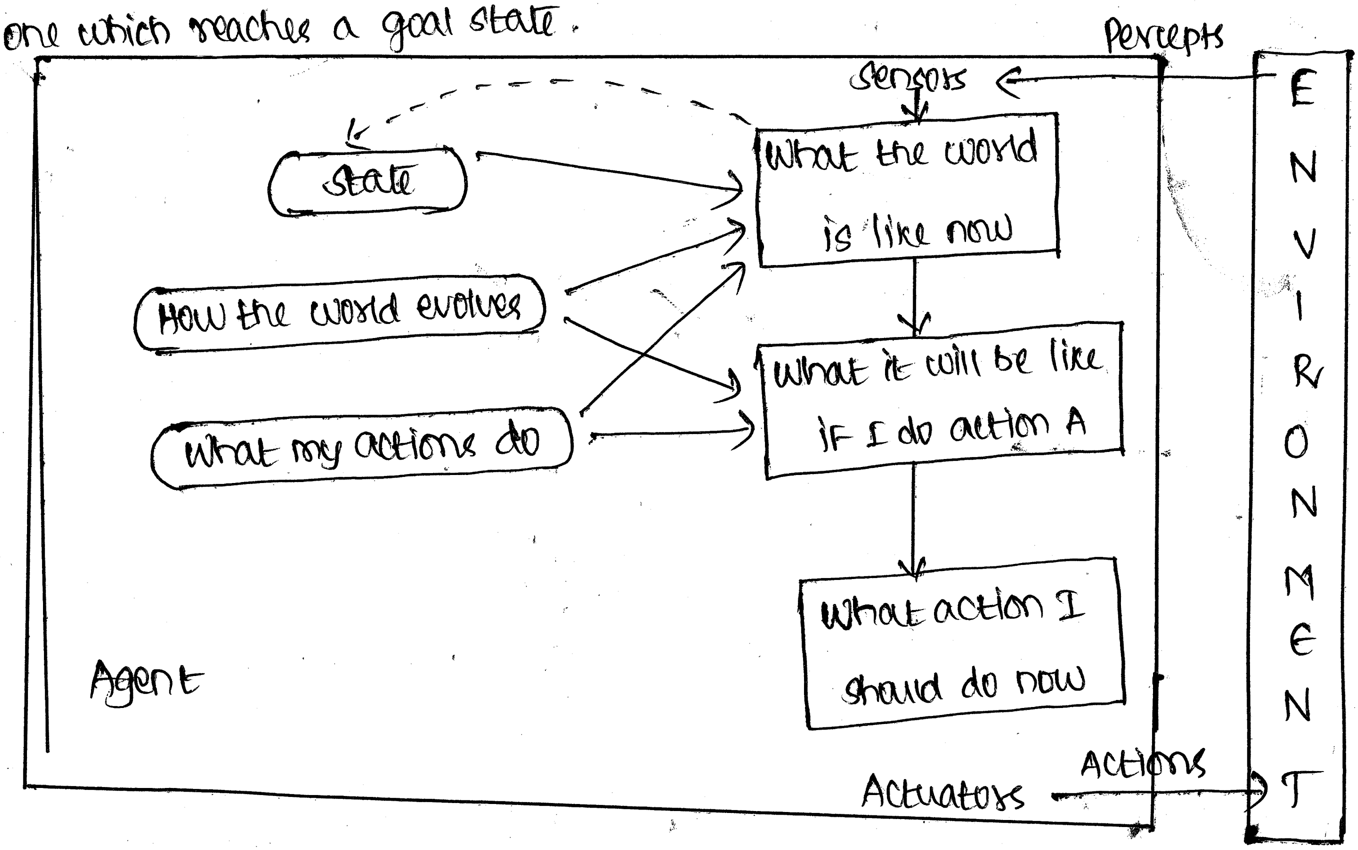


Fig. A Model-Based, goal-based Agent. It keeps track of the world state as well as a set of goals it is trying to achieve, and chooses an action that will (eventually) lead to the achievement of its goals.

The knowledge that supports its decisions is represented explicitly and can be modified, which makes these agents more flexible. They usually require <u>search</u> and <u>planning</u>. The goal-based agent's behavior can easily be changed.

UTILITY-BASED AGENTS

Goals alone are not enough to generate high-quality behavior in most environments. For eg. Many action sequences will get the taxi to its destination (thereby achieving the goal) but some are quicker, safer, more reliable or cheaper than others.

The agents which are developed having their end uses as building blocks are called **utility based agents**. When there are multiple possible alternatives, then to decide which one is best, utility-based agents are used. They choose actions based on a preference (utility) for each state.

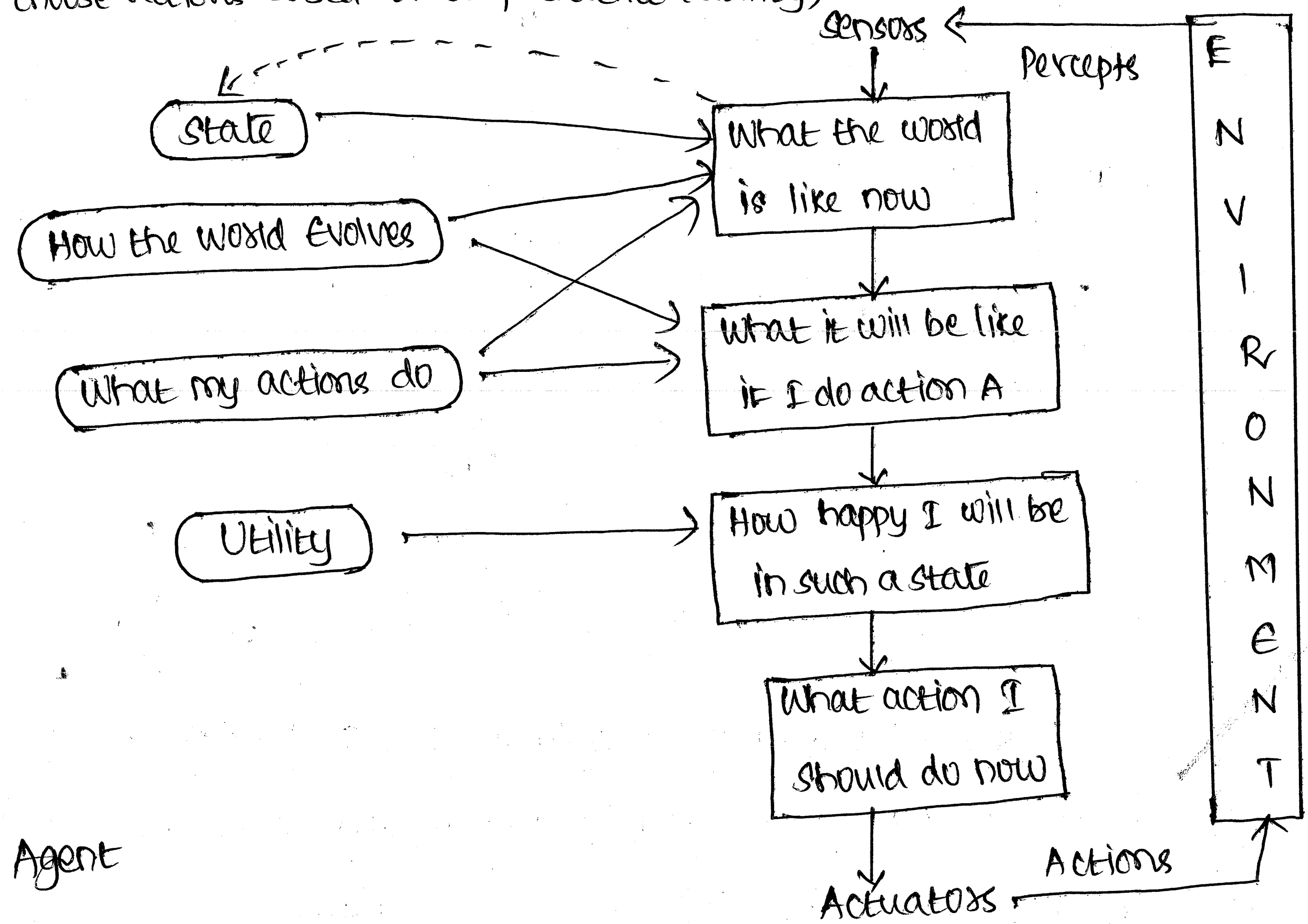


Fig. A Model Based, Utility-Based Agent. It uses a model of the world, along with a utility function that measures its preferences among states of the world. Then it chooses the action that leads to the best expected, utility where expected utility is computed by averaging over all possible outcome states, weighted by the probability of the outcome.

Sometimes achieving the desired goal is not enough. We may look for a quicker, safer, cheaper trip to reach a destination. Agent happiness should be taken into consideration. Utility describes how "happy" the agent is. Because of the uncertainty in the world, a utility agent chooses the action that maximizes the expected utility. A utility function maps a state onto a real number which describes the associated degree of happiness.

LEARNING AGENTS:

A learning Agent in AI is the type of agent which can learn from its past experiences or it has learning capabilities. It starts to act with basic knowledge and then able to act and adapt automatically through learning.

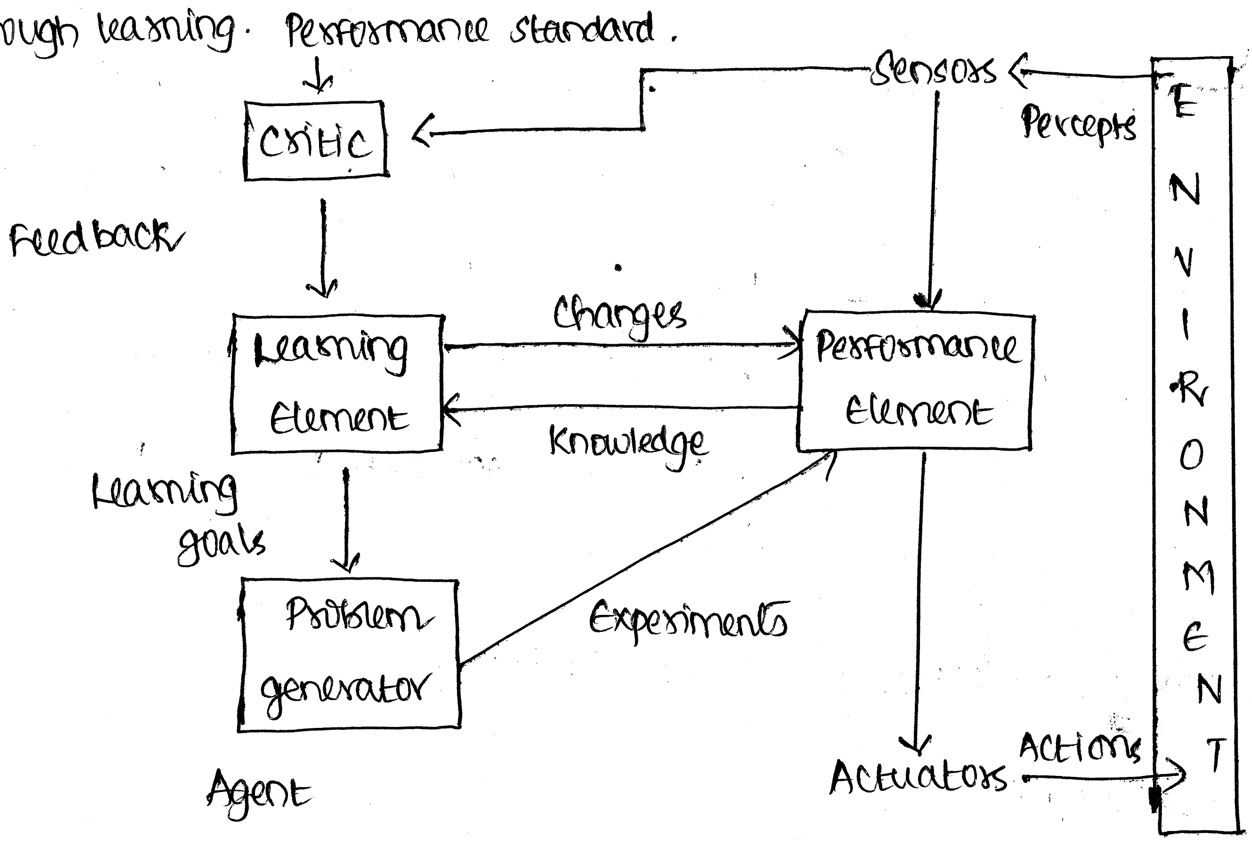


Fig. A General Learning Agent

A learning agent has mainly four conceptual components which are 1. Learning Element 2. Critic 3. Performance Element 4. Problem Generator

1. Learning Element:

It is responsible for making improvements by learning from the environment.

2. Critic:

Learning element takes feedback from critic which describes how well the agent is doing with respect to a fixed performance standard.

3. Performance Element:

It is responsible for selecting external action

4. Problem Generator:

This component is responsible for suggesting actions that will lead to new and informative experiences.

Note: Simple Reflex Agents respond directly to percepts, whereas model-based reflex Agents maintain internal state to track aspects of the world that are not evident in the current percept. Goal-based Agents act to achieve their goals and utility-based agents try to maximize their own expected "happiness". All Agents can improve their performance through learning.

PROBLEM SOLVING APPROACH TO TYPICAL AI Problems

Problem solving Agents:

Intelligent agents are supposed to maximize their performance measure, achieving this is sometimes simplified if the agent can adopt a goal and aim at satisfying it. An agent can find a sequence of actions that achieves its goals, when no single action will do.

Goal Formulation:

Goals helps organize behavior by limiting the objectives that the agent is trying to achieve and hence the actions it needs to consider. Goal formulation based on the current situation and the agent's performance measure, is the first step in problem solving.

Problem Formulation:

Problem formulation is the process of deciding what actions and states to consider, given a goal.

The process of looking for a sequence of actions that reaches a goal is called **search**. A search algorithm takes a problem as input and returns a solution in the form of an action sequence. Once a solution is found, the action it recommends can be carried out. This is called the **execution** phase.

Well-defined Problems and solutions

A Problem can be defined formally by **five components**

1. The <u>Initial state</u> that the agent starts in. For example, the initial state for our agent in Romania might be described as In(Arad)

2. A description of the possible <u>actions</u> available to the agent. Given a particular state s, ~~ACTION~~ ACTION(s) returns the set of actions that can be executed in s.

For <u>eg</u>. In(Arad), the applicable actions are

{Go(Sibiu), Go(Timisoara), Go(Zerind)}

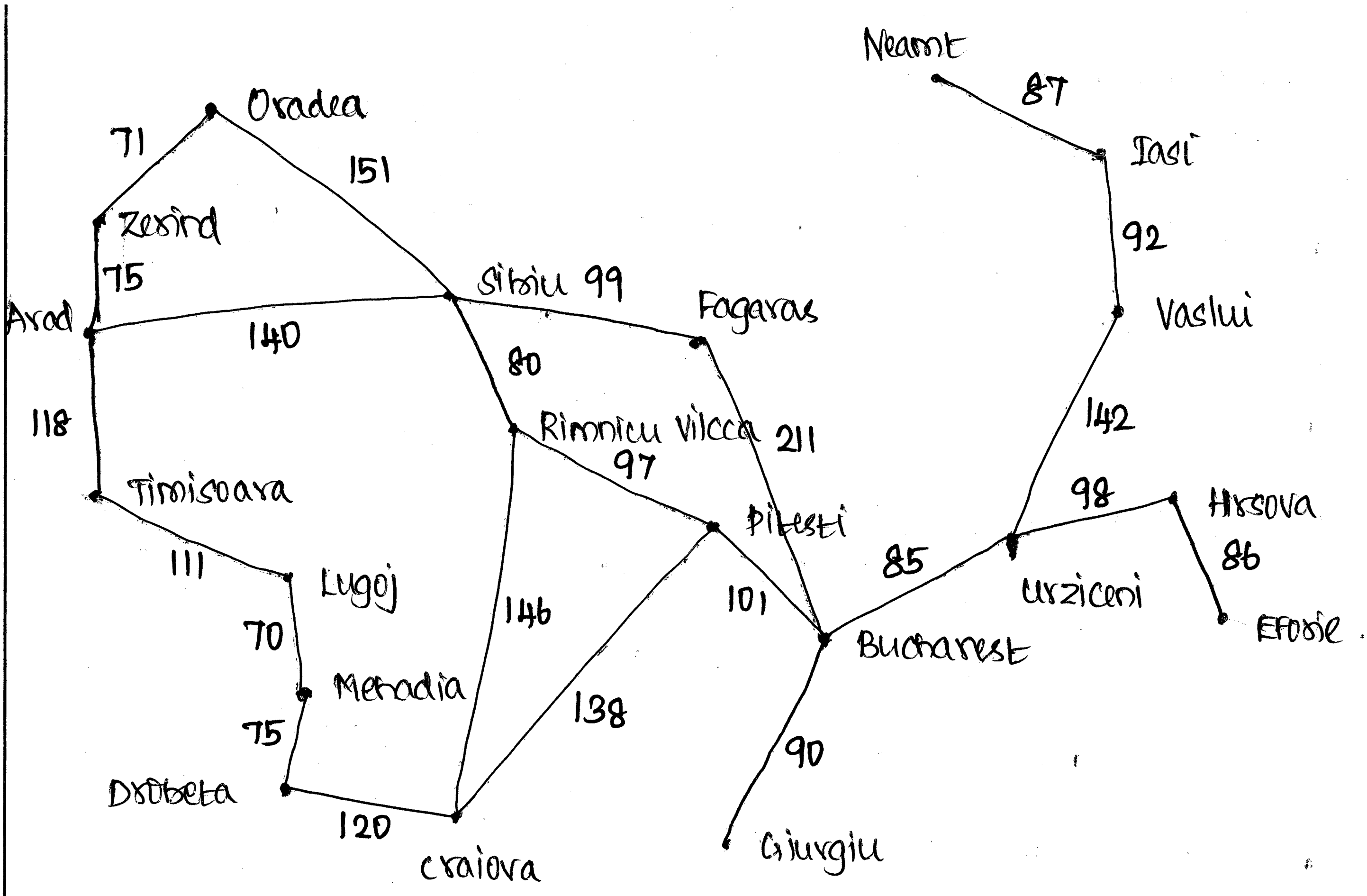


Fig. A simplified road map of part of Romania

3. A description of what each actions does; the formal name for this is the **transition model**, specified by a function RESULT (s, a) that returns the state that results from doing action a in state s. The term successor to refer to any state reachable from a given state by a single action.

Eg. RESULT (In (Arad), Go (Zerind)) = In (Zerind).

Note: The initial state, actions, transition model implicity define the state space, of the problem. The state space forms a directed network or graph in which the nodes are states and the links between nodes are actions. A path in the state space is a sequence of states connected by a sequence of actions.

function SIMPLE-PROBLEM-SOLVING-AGENT (percept) **returns** an action

Persistent : seq, an action sequence, initially empty

state, some description of the current world state

goal, a goal, initially null

Problem, a problem formulation

state ← UPDATE-STATE (state, percept)

if seq is empty then

goal ← FORMULATE - GOAL(state)

Problem ← FORMULATE - PROBLEM (state, goal)

seq ← SEARCH (problem)

if seq = failure **then return** a null action

action ← FIRST(seq)

seq ← REST(seq)

return action

Fig. A simple problem-solving agent. It first formulates a goal and a problem, searches for a sequence of actions that would solve the Problem, and then executes the actions one at a time. When this is complete, it formulates another goal and starts over.

4. The **goal test**, which determines whether a given state is a goal state. sometimes there is an explicit set of possible goal states, and the test simply checks whether the given state is one of them.

Eg: The agent's goal in Romania is the singleton set

{In (Bucharest)}

5. A path cost function that assigns a numeric cost to each path. The problem-solving agent chooses a cost function that reflects its own performance measure.

The step cost of taking action a in state s to reach state s' is denoted by c(s, a, s').

The process of looking for a sequence of actions that reaches a goal is called **search**. A search algorithm takes a problem as input and returns a **solution** in the form of an action ~~is called the~~ sequence. Once a solution is found, the actions it recommends can be carried out. This is called the **Execution** phase. A **solution** to a problem is an action sequence that leads from the initial state to a goal state. Solution quality is measured by the path cost function and an **optimal solution** has the lowest path cost among all solutions.

PROBLEMS:

The problem-solving approach has been applied to a vast array of task environments. ~~We list~~ Some of the best known here, distinguishing between toy and real-world problems.

A Toy Problem is intended to illustrate various problem-solving methods. It can be given a concise, exact description and hence is usable by different researchers to compare the performance of algorithms.

A real-world problem is one whose solutions people actually care about.

B TOY PROBLEM

1. VACUUM-WORLD The first example is the vacuum world.

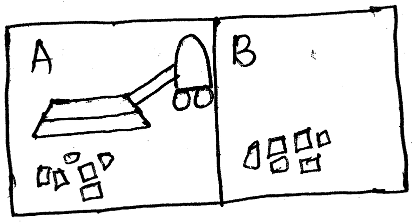


Fig. A Vacuum – cleaner world with 2 locations

This particular world has just two locations: squares A and B. The vacuum agent perceives which square it is in and whether there is dirt in the square. It can choose to move left, move right, suck up the dirt or do nothing. One very simple agent function is the following: if the current square is dirty then suck; otherwise, move to the other square.

This can be formulated as a problem as follows:

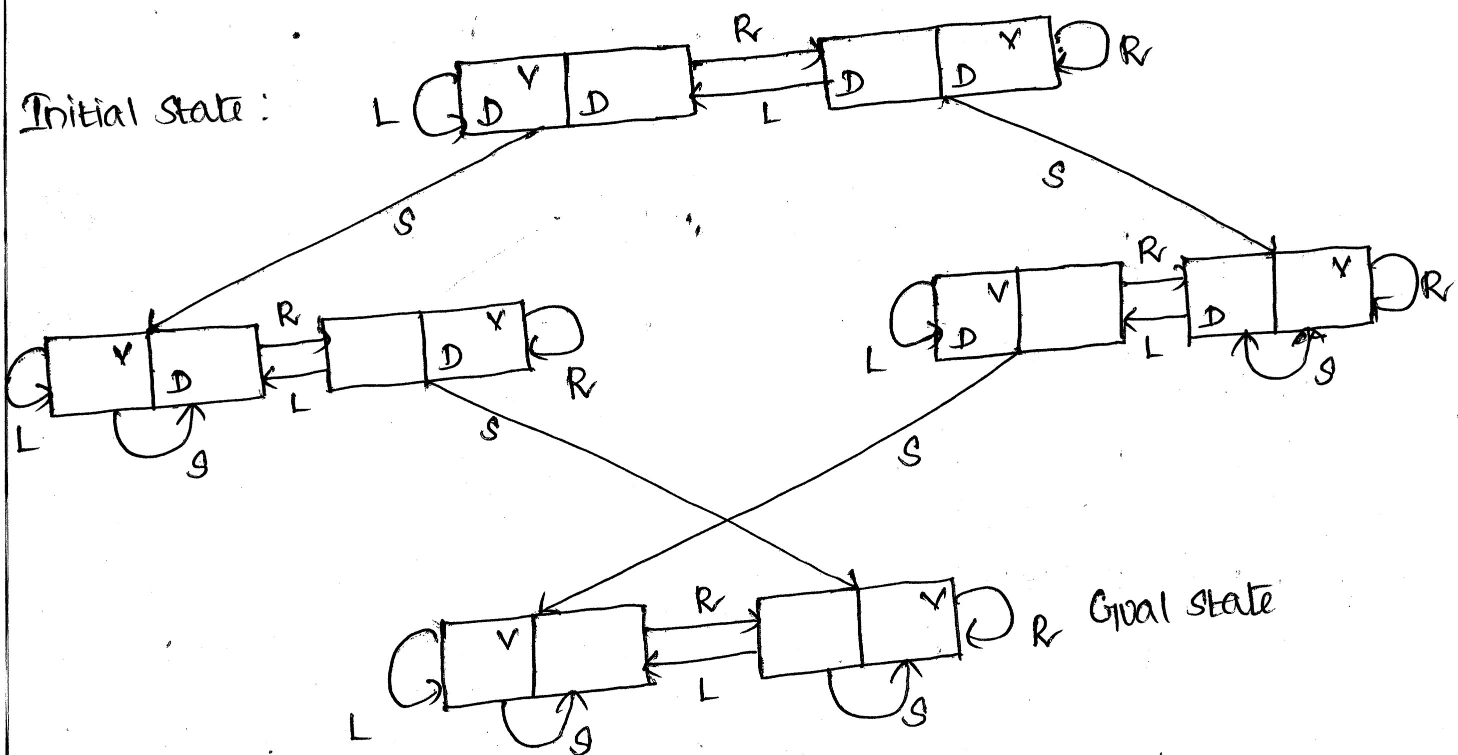


Fig. The state space for the vacuum world. Links denote actions:

L = left, R = Right, S = Suck.

States: The state is determined by both the agent location and the dirt locations. The agent is in one of two locations, each of which might or might not contain dirt. Thus, there are $2 \times 2^2 = 8$ possible world states. A larger environment with n locations has $n \cdot 2^n$ states.

Initial state: Any state can be designated as the Initial state

Actions: In this simple environment, each state has just three actions; Left, Right and Suck. Larger environments might also include Up and Down.

Transition Model: The actions have their expected effects, except that moving Left in the leftmost square, moving Right in the rightmost square and sucking in a clean square have no effect.

Goal Test: This checks whether all the squares are clean.

Path cost: Each step costs 1, so the path cost is the number of steps in the path.

2. 8-PUZZLE PROBLEM

The 8-puzzle, an instance of which is shown in the figure below, it consists of a 3x3 board with eight numbered tiles and a blank space. A tile adjacent to the blank space can slide into the space. The object is to reach a specified goal state.

7	2	4
5		6
8	3	1

Start State

	1	2
3	4	5
6	7	8

Goal State

Fig. A Typical Instance Of the 8-Puzzle

States: A state description specifies the location of each of the eight tiles and the blank in one of the nine squares.

Initial state: Any state can be designated as the Initial state. Note that any given goal can be reached from exactly half of the Possible Initial states

Actions: The simplest Formulation defines the actions as movements of the blank space Left, Right, Up or Down. Different subsets of these are possible depending on where the blank is.

Transition Model: Given a state and action, this returns the resulting state; for eg: If we apply Left to the start state in the above figure, the resulting state has the 5 and the blank switched.

Goal Test: This checks whether the state matches the goal configuration as shown (Other goal configurations are possible)

Path cost: Each step costs 1, so the path cost is the number of steps in the Path.

The 8-puzzle belongs to the family of sliding-block Puzzles, which are often used as test Problems for new search Algorithms.

3. 8-Queens PROBLEM

The goal of the 8-queens problem is to place eight queens on a chessboard such that no queen attacks any other. (A queen attacks any piece in the same row, column or diagonal)

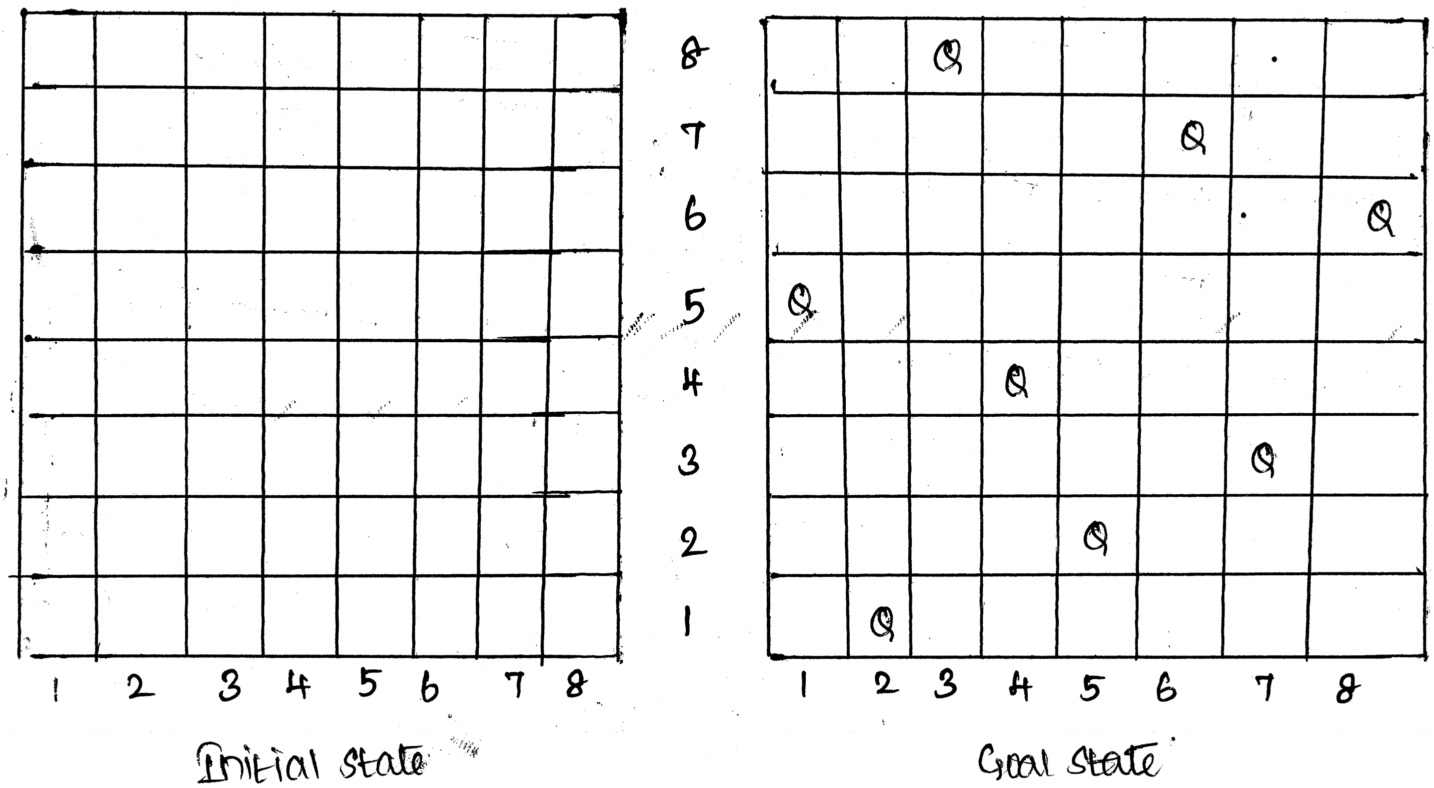


Fig. A solution to the 8-Queens Problem

<u>Note:</u> Although efficient special-purpose algorithms exist for this problem and for the whole n-queens family, it remains a useful test problem for search algorithms. There are two main kinds of formulation.

Incremental Formulation: It involves operators that augment the state description, starting with an empty state; for the 8-queens problem, this means that each action adds a queen to the state.

Complete-state Formulation: It starts with all 8 queens on the board and moves them around. In either case, the path cost is of no interest because only the final state counts.

States: Any arrangement of 0 to 8 queens on the board is a state.

Initial state: No queens on the board.

Actions: Add a queen to any empty square

Transition Model: Returns the board with a queen added to the specified square.

Goal Test: 8 queens are on the board, none attacked.

Path cost: Each step costs 1, so the path cost is the number of steps to obtain solution.

In the next formulation, it would prohibit placing a queen in any square that is already attacked.

States: All possible arrangements of n queens ($0 \leq n \leq 8$). one per column in the leftmost n columns, with no queen attacking another.

Actions: Add a queen to any square in the leftmost empty column such that it is not attacked by any other queen

Incremental → $64 \cdot 63 \ldots 57 \approx 1.8 \times 10^{14}$ possible sequences.

Complete - state → It reduces the 8-queens state space from 1.8×10^{14} to just 2,057 & solutions are easy to find.

4. INFINITE STATE SPACE

This problem was devised by Donald Knuth (1964) and illustrates how infinite state spaces can arise. Knuth conjectured that, starting with the number 4, a sequence of factorial, square root and floor operations will reach any desired positive integer.

For eg: We can reach 5 from 4 as follows:

$$\left\lfloor \sqrt{\sqrt{\sqrt{\sqrt{\sqrt{(4!)!}}}}} \right\rfloor = 5$$

States: Positive Numbers

Initial State: 4

Actions: Apply factorial, square root or floor operation (factorial for integers only)

Transition Model: As given by the mathematical definitions of the operations

Goal Test: State is the desired positive Integer.

There is no bound on how large a number might be constructed in the process of reaching a given target. For eg: the Number 620,448,401,733,239,439,360,000 is generated in the expression for 5, so the state space for this problem is infinite. Such state spaces arise frequently in tasks involving the generation of mathematical expressions, circuits, proofs, programs and other recursively defined objects.

REAL WORLD PROBLEMS

1. ROUTE-FINDING PROBLEM

It is defined in terms of specified locations and transitions along links between them. Route-finding algorithms are used in a variety of application such as web sites and in car system that provide direction. consider the airline travel problem that must be solved by a travel-planning web sites:

States: Each state obviously includes a location (e.g. an airport) and the current time. Furthermore, because the cost of an action (a flight segment) may depend on previous segments, their fare bases, and their status as domestic or international, the state must record extra information about these "historical" aspects.

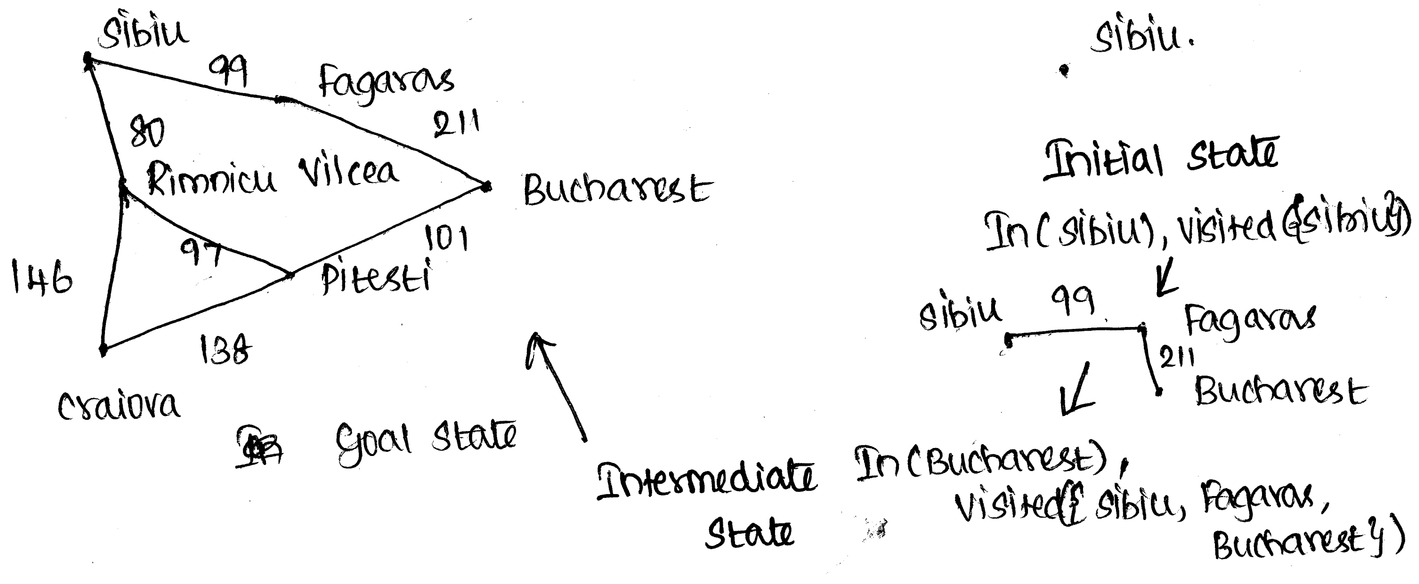
Initial state: This is specified by the user's query

Actions: Take any flight from the current location, in any seat class leaving after the current time, leaving enough time for within-airport transfer if needed.

Transition Model: The state resulting from taking a flight will have the flight's destination as the current location and the flight's arrival time as the current time.

Goal Test: Are we at the final destination specified by the user?

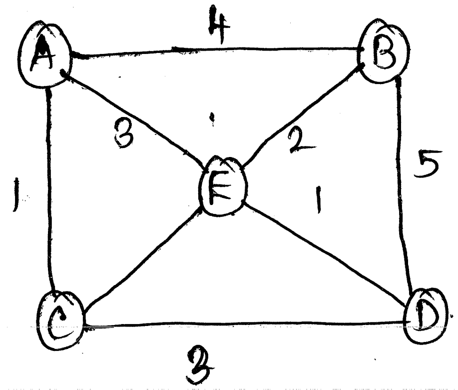
Path cost: This depends on monetary cost, waiting time, flight time, customs and immigration procedures, seat quality, time of day, type of airplane, frequent-flyer mileage awards and so on.



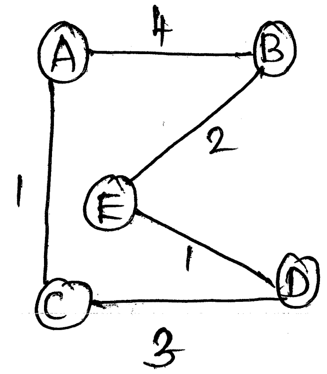
2. TRAVELING SALESMAN PROBLEM

Traveling salesman Problem (TSP) is a touring Problem in which each city must be visited exactly once. The aim is to find the shortest tour. The problem is known to be NP-hard, but an enormous amount of effort has been expanded to improve the capabilities of TSP Algorithms.

In addition to planning trips for traveling salesman, these algorithms have been used to tasks such as planning movements of automatic circuit-board drills and of stocking machine on shop floors.



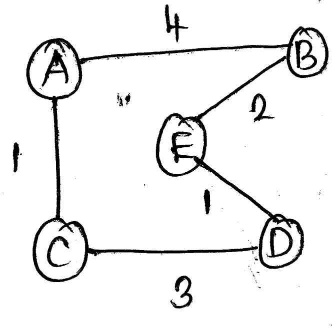
Initial State



(4+2+1+3+1 = 11)

A-B-E-D-C-A

(or)



(1+3+1+2+4 = 11)

A-C-D-E-B-A

Goal State

Fig. A Typical solution for TSP

Problem Formulation for TSP

States: Cities (A, B, C, D and E)

Initial state: A, in a complete state space All possible states and actions

Successor function: Travel from one city to another connected by a road

Goal Test: The trip visits each city only once that starts and ends at A

Path cost: Traveling time

```
function TSP (start_pos, C)
    current_pos ← start_pos
    V ← {}
    while |V| < |C| do
        next_pos ← min_c dist (current_pos, c), ∀c ∈ C\V
        V ← V ∪ next_pos
        current_pos ← next_pos
    end while
end function
```

Other ~~state~~ real world Problems

1. A VLSI layout problem
2. Robot Navigation
3. Automatic Assembly sequencing.

Sample Exercise Problem

The missionaries and cannibals problem is usually stated as follows. Three missionaries and three cannibals are on one side of a river, along with a boat that can hold one or two people. Find a way to get everyone to the other side without ever leaving a group of missionaries in one place outnumbered by the cannibals in that place. This problem is famous in AI

~~because it was the~~

a. Formulate the problem precisely, making only those distinctions necessary to ensure a valid solution. Draw a diagram of the complete state space.

b. Implement and solve the problem optimally using an appropriate search Algorithm. Is it a good idea to check for repeated states.

Solution

The given data are

1. There are three missionaries and three cannibals on the left bank of a river

2. They wish to cross over the right bank using a boat that can only carry two at a time.

3. The number of cannibals on either bank must never exceed the number of missionaries on the same bank, otherwise the missionaries will become the cannibals dinner.

4. Plan a sequence of crossings that will take everyone safely across.

State Representation

There are many possibilities. Represent the missionaries by M and the cannibals by C. Let the boat be B. Each state can be represented by the items on each side.

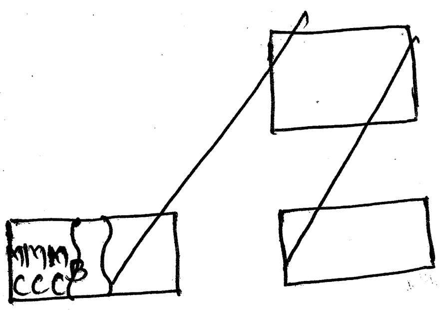
Eg. side$_1$ {M, M, C, C}, side$_2$ {M, C, B}

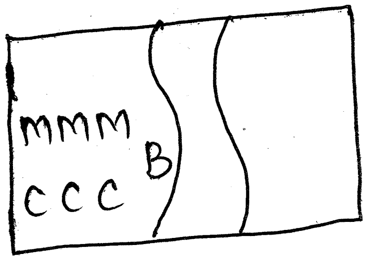
Initial state: side$_1$ {M, M, M, C, C, C, B}, side$_2$ {}

Goal state: side$_1$ {}, side$_2$ {M, M, M, C, C, C, B}

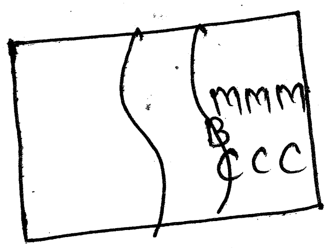
Cost Function: Each move has unit cost.

This Problem can be solved by searching for a solution, which is a sequence of actions that leads from the initial state to the goal state. The goal state is effectively a mirror image of the initial state. The complete search space is shown.





Initial State



Goal state.

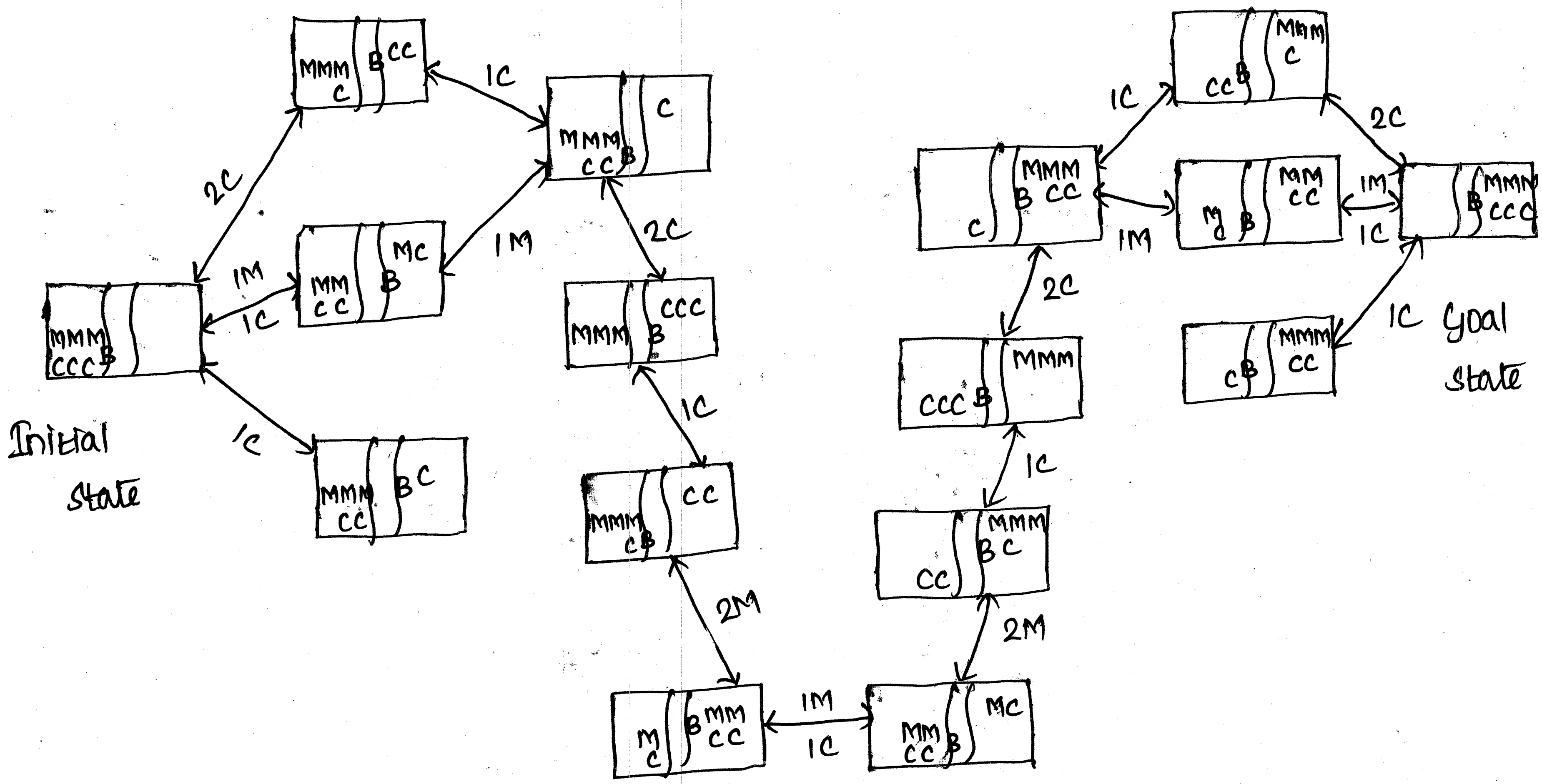


Fig. Search-space for the Missionaries and cannibals problem

b. Reachable states / successor function

side₁ {M, M, M, C, C, C, B}, side₂ {} → Initial state

side₁ {M, M, M, C}, side₂ {C, C, B}

side₁ {M, M, M, C, C, B}, side₂ {C}

side₁ {M, M, M}, side₂ {C, C, C, B}

side₁ {M, M, M, C, B}, side₂ {C, C}

side₁ {M, C}, side₂ {M, M, C, C, B}

side₁ {M, M, C, C, B}, side₂ {M, C}

side₁ {C, C}, side₂ {M, M, M, C, B}

side₁ {C, C, C, B}, side₂ {M, M, M}

side₁ {C}, side₂ {M, M, M, C, C, B}

side₁ {M, C, B}, side₂ {M, M, C, C}

side₁ {}, side₂ {M, M, M, C, C, C} → Goal state

Reachable states = {11}

The above states are the successor function, from each state either bring one missionary, one cannibal, two missionaries, two cannibals, or one of each type to the other bank.

Repeated states can cause incompleteness or enormous runtimes. Instead, the algorithm for search can maintain list of previously visited states to avoid this. By Russell and Norvig "Algorithms that forget their history are doomed to repeat it". It is suggested, that it is not a good idea to check for repeated states. A* search algorithm can be used to solve problem optimally.