Artificial Inteligence

ECAP653

Edited by:
Dr. Rishi Chopra

LOVELY PROFESSIONAL UNIVERSITY

LOVELY PROFESSIONAL UNIVERSITY

Artificial Inteligence

Edited By

Dr. Rishi Chopra

Title: Artificial Inteligence

Author's Name: Dr. V Devenderan

Published By : Lovely Professional University

Publisher Address: Lovely Professional University, Jalandhar Delhi GT road, Phagwara - 144411

Printer Detail: Lovely Professional University

Edition Detail: (I)

ISBN:

Content

Unit 01: Introduction

CONTENTS

Objectives

- Understanding the basic concepts of Artificial Intelligence.
- Understanding the types of intelligence with examples.
- Understanding the difference between hard problems and easy problems in AI.
- Understanding the various approaches of artificial intelligence in detail.
- Understanding the applications with real time examples in various fields.

Introduction

In this unit, we will learn the basic concepts of Artificial Intelligence. This topic discusses the types of intelligence i.e., Natural intelligence and Artificial intelligence. Also,it discusses the difference between Artificial Intelligence and Natural Intelligences with examples. Every problem is not easy for artificial intelligence to implement and use. Hence, the classification of problems is very important to start working with artificial intelligence. In addition to the discussion, few examples were given for hard problems and easy problemsfor better understanding.Realtime applications are alsogivenin this unit.This unit also covers the different approaches used for developing artificial intelligence applications. At the end of this unit, the foundation of artificial intelligence will be understood.

1.1 What is Artificial Intelligence?

First of all, before start discussing about Artificial Intelligence we will try to understand what is Intelligence. We do believe that we have intelligence. Yes, at the same time, don't forget that we also have the skills. Then the question comes as what is the difference between skills and intelligence? We try to understand these slowly for more clarity.

Intelligencecan be defined, as it is the computational part of the ability to achieve goals in the world. Every human being is having varying kinds and degrees of intelligence to solve their day-to-day problems. Intelligence is a kind of heuristic approach or futuristic or thinking in all the directions for the optimal solution. On the other hand, the skill – is the knowledge such as driving, swimming, skating, reading, writing and so on. The intelligence what human being is having, is known as Natural Intelligence.

Now, we are trying to develop similar kind of intelligence artificially for the machines, is known as Artificial Intelligence. It is the science and engineering of making intelligent machines, especially intelligent computer programs. It is related to the similar task of using computers to understand human intelligence, but AI does not have to confine itself to methods that are biologically observable.

1.2 Hard Problems

We are trying to differentiate the problems into two categories such as hard problems and easy problems. It is better to know how the natural intelligence is applied and getting solved in those problems. Here are the few examples of Hard Problems.One of the simple examples is that to mechanize tasks that animals can do easily like catching prey and avoiding predators, interpreting complex sensory information (visual, aural), modeling the internal states of other animals from their behavior and working as a team (ants, bees). In fact, Artificial Intelligence is a collection of hard and easy problems which can be solved by humans and other living things. The reason being hard is creating the algorithms or procedures are really difficult. Let me give few more commonly used problems below.List of commonly known hard problems is given below.

Expert Consulting Systems

This is the key problem in the development ofAutomated Consulting System is how to represent and use the knowledge that human experts in these subjects obviously posses and use. It tries to identify the problems of the human being and tries to give optimal solution in a specific domain. It will be working based on the question-answering mode usually. There is no limit in the collection of knowledge base. That makes this problem tough. Also, this problem is more difficult by the fact that the expert knowledge in any important field is imprecise and uncertain.

Automatic Programming

In automatic programming, a system takes in a high level description of what program is to be accomplished and produce a program.Presently, we have much more human power (software engineers) to code. Consider the situation where you havemore computing power than the manpower. Computers should have knowledge / intelligence by whichthe task can be divided and given to multiple computers to execute fast. This can be decided based on the number of computers having less workload. This is known as automatic programming, which is comes under hard problems.

Natural Language Processor

This problem focuses on the natural language processing. Natural language means the language what we are using to speak and write. Different regions or countries have their own natural languages for the communication. If we go to them for any business or personal trip, then we will find very difficult to talk to them and get the work done. This problem addresses this and tries to make automation helping the translation between different languages. First of all, the computers need to understand the spoken and written languages. It is necessary to know a lot about the vocabulary and grammar in each language. Translation should be perfect without changing the original meaning, isreason for hard problem.

Perceptional Problems

Now days, we have been using lot of Internet of Things for the surveillance activities and health care especially. Hence, we can easily understand that computers are made to listen their surroundings from many input (audio, video) devices. There will be lot of information will be coming from audio as well as video camera devices. The problem is how the computer extracts the information from the audio and visual things. There should be some preliminary knowledge to understand them. It is applicable to human being too. You can experience this while reading the newspapers. You can’t understand today’s news if you are missing few weeks. Our actual focus is the perceptional problem. That is, different people understand the same information differently, is exactly known as perceptional problem. Here, the figure 1 is depicting clearly on the number. It is visible as 6 to one person but it is looking like 9 to another person. This is known as perceptions. This is because of their previous experience, the way of approach towards the problem and the level of knowledge, they posses. This cannot be same for sure at any point of time. These things should not be occurring in the computers. So, it requires processing of large base knowledge about the things being perceived, which is making complicated.

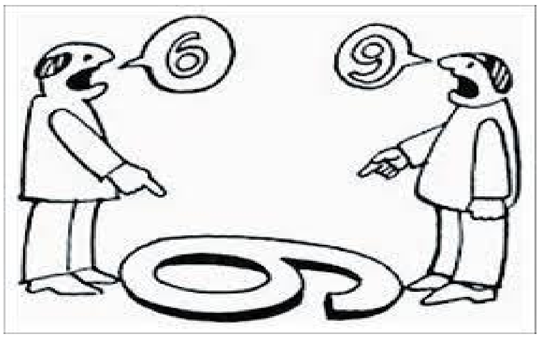


Fig. 1 Perception of human

Robotics

There is no need to explain much for this kind of problem. Everyone can easily understand what robotics and their nature of task are. Actually, the kind of robot given in figure 2 is mobile robot. It is having multiple functionalities. It finds the obstacles on its way for the smooth movement. It can do physical actions like picking up some object from the floor and store it if needed. It can do surveillance from the video camera and record unusual activities. It deals with the problems of controlling the physical actions of a mobile Robot. All these need intelligences, which makes the problem very hard.

Fig. 2 Mobile Robot

1.3 Easy Problems in AI

The second category of the problems is discussed here i.e., easy problems. Easy problems means that the problems where writing the procedures or algorithms is simple and not much difficult. There are few examples is given for your better understanding like symbolic integration, proving theorems, playing chess, etc. Similarity between all the problems is that they are having certain procedures or rules to solve the problem. Hence, these problems can be solved by following given set of rules. We should know which rule is to be selected or used on the particular situation. Hence, we are considering this as easy problems. Now, have a look at the given examples.

Theorem Proving

If you are familiar with mathematics, then you must know the term 'theorem'. In simple, theorem can be understood as a statement. This statement should be proved as true with required evidences called proofs. Mathematicians spend lot of their time proving the existing theorems. Sometimes, it will come nicely. Otherwise, they should repeat the whole thing with different approach. They also want to redo the proving in a better way. Hence, automation comes to reduce the time. Computer knows all the basic principles and knowledge about related theorems / lemmas. Computer should have the ability to make deductions from hypothesis. It demands intuitive scales such as guessing which path should be proved first in order to help proving the theorem.It also requires judgments to guess accurately about which previously proven theorems in a subject area willbe useful in the present proof. Also sometimes it is needed to break the main problem into sub-problems to work on independently. Lot of trial and error will help to solve the problem.

Playing Chess

Chess game becomes the common board game with us and many of us know how to play the game. Generally speaking, it is a two-player game. It is having 16 coins on each side as in Figure 3. Those coins are moved with predefined set of rules / procedures. Hence, coins need to follow the predefined rules on certain conditions. Meaning that it need to satisfy some rules before moving any coins. The moves are also fixed for the said coins. Random moves are not allowed at any cost. Success of player is clearly mentioned already. So, this problem is completely based on following the rules, is hence understood as simple problem.

Fig. 3 Chess Game

1.4 Types of Intelligence

In this context, we know that we are having two types of intelligences. First type of intelligence is Natural Intelligence, and the second type of intelligence is Artificial Intelligence. These types were already introduced in the previous sections, and we know the differences too. In addition to that, we like to refer again with general definitions as given below.

Natural Intelligence

Natural or Human Intelligence is defined as the quality of the mind that is made up of capabilities to learn from past experience, adaptation to new situations, handling of abstract ideas and the ability to change his/her own environment using the gained knowledge.

Artificial Intelligence

It is a way to make machines think and behave intelligently. These machines are controlled by software inside them. It has a lot to do with intelligent software programs that control these machines. It is a science of finding theories and methodologies that can help machines understand the world and accordingly react to situations in the same way that humans do.

But, actually there are in general three kinds of intelligence exist. First kind of people understands things for itself. The second kind of people appreciates what other can understand easily. Third kind of people, neither understands for itself nor from others. The first kind is excellent and the second kind is good. But, the third kind of intelligence is useless.

Generally, intelligences were categorized into many. Figure 4 gives us the clear idea about the general classification of intelligences.

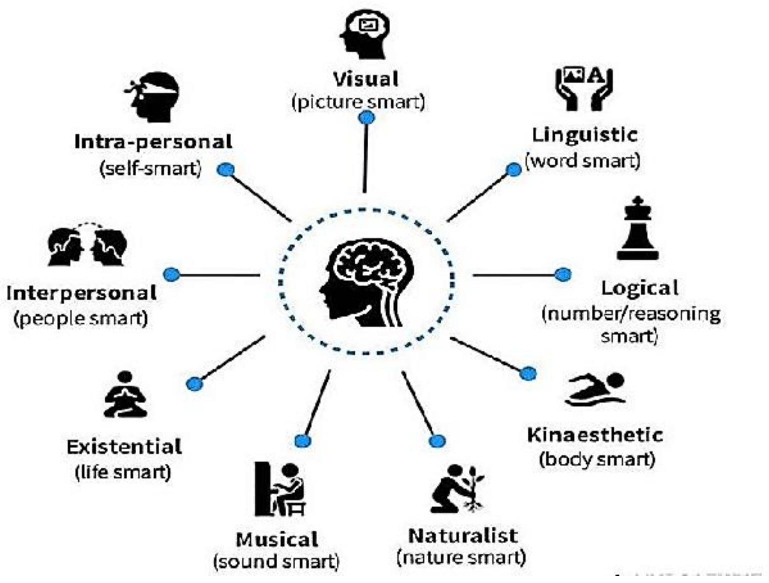


Fig 4. Types of Intelligence

1.5 Applications of Artificial Intelligence

There are plenty of real time applications that use Artificial Intelligence. Some of the most popular areas include the followings.

Computer Vision

These are systems that deal with visual data such as images and videos. These systems understand the content and extract insights based on the use case.

Speech Recognition

These systems are capable of hearing and understanding spoken words. For example, there are intelligent personal assistants on our smart phones that can understand what we are saying and give relevant information or perform an action based on that.

Games

Artificial Intelligence is used extensively in the gaming industry. It is used to design intelligent agents that can complete with humans.

Robotics

Robotic systems are able to perform many different tasks depending on the situations. The robots have sensors and actuators that can do things of receiving input and giving the output. They have processors on board to compute the inputs in respect with the assigned task in the real time. They also learn to adapt to the new environments.

Expert Systems

This kind of systems is developed to provide advice or make decisions like an expert in a particular field or domain.They usually use databases of expert knowledge in the areas like finance, medicine, marketing, law, education and so on.

1.6 Artificial IntelligenceApproaches

There are four approaches used to make Artificial Intelligence machines. They are Think Well, Act Well, Think like human and Act like human. The figure 5 given below depicts this clearly.

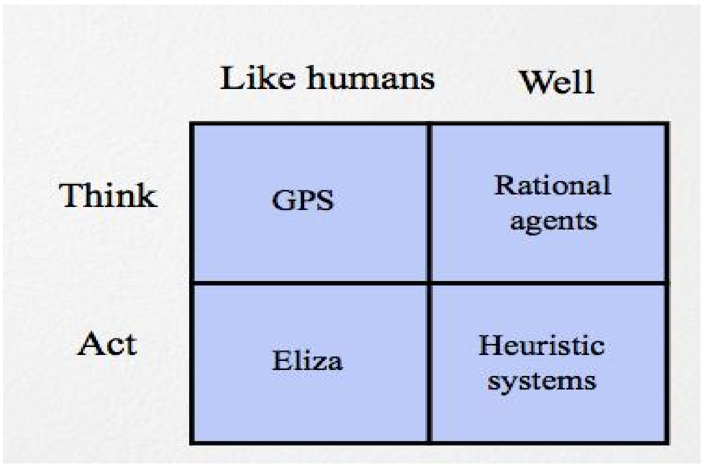


Fig. 5Artificial Intelligence Approaches

Think like humans

First row is depicting the term 'Think'. The term 'Think' refers to machines where thinking happens by some means. Thinking can be like human being or it can be imitating like human being. The imitating like human being is understood by the term 'Like. The example given is GPS for thinking like humans. GPS stands for General Problem Solver, the other name given for the human being. We are the one who can think and finds optimal solution for maximum of the problems in maximum number of disciplines. Majorly, we will manage any kind of situations. It is very difficult to replace us.

We can also say that this otherwise focusing on a reasoning process as same as we do thinking about something usually. The goal is to produce a sequence of steps of the reasoning process that was similar to the steps followed by a person in solving that task. This machine is known as General Problem Solver as in Figure 6.

Fig. 6ThinkingRobot

Think well

The next one is the Think Well. The example shows that the machine is imitating that it is thinking, which is called rational agents. This rational agent represents a decision-making machine on a given situation like a human being. You can consider an example of traffic controller. Thinking and deciding as in Figure 7 when to give green light or yellow light or red light for the smooth traffic. Yes, it is focusing on Decision Making. It can be anything that makes decisions like human at certain situations. This machine is otherwise known as rational agent. It will be having the expertise on only one field or domain.

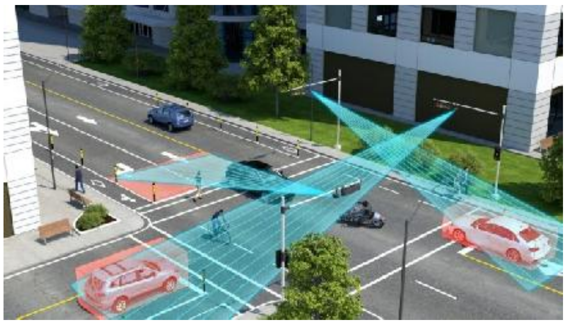


Fig. 7Traffic Controller

Act like humans

This is focusing on the behavior of the human being. This kind of problem is trying to imitate the human behavior and hence it is known as behavioral approach. The popular example is the robot named as SOPHIA as in Figure 8. This robot is latest and highly talented in acting like human being with emotions. Also it is having the skill in understanding the questions and answers just like the human being. You can see the interview given by the robot in YouTube. The weblink is given in end of the unit.Any robot, which imitates the body language how the human being acts and reacts, will come in this category.

Fig. 8SensingRobot

Act Well

This is the very basic approach among the artificial intelligence approaches. The idea is to help in finding optimal solution from a large problem space as in Figure 9.Means; it takes lot of time if you search in a usual way with standard searching algorithms. Here, some kind of intelligence is applied / used to reduce the searching time. This kind of intelligence is known as heuristics. There is no standard formula to frame the heuristics. Even you are using your own heuristics to solve your day-to-day problems. You know how to solve it but you may not know how you got that idea at that situation suddenly. That's what we try to mean by heuristics. We try to standardize the heuristics, which works well for the given problems. In specific, we just trying to imitate what others are solving the problemsare known as Act Well. This can be achieved using some algorithms such as optimization algorithms and searching algorithms. The genetic algorithm is the popular searching algorithm in the large spaces using its own heuristics. The ant colony optimization algorithm and bee colony optimization are few well-known algorithms in optimization problems. Optimization problem can be understood via travelling salesmen problem or job scheduling problems. Hence, these algorithms are coming under the “Act Well” approach.

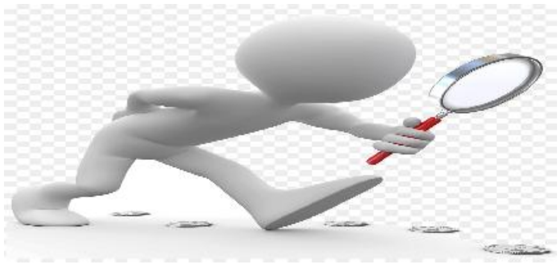


Fig. 9Searching Engine

1.7 <u>Turing Test</u>

Turing test is the test conducted for the machines, which is claiming that it is having artificial intelligence. Proposed by the English mathematician Alan M. Turing during 1950. The test is conducted using the following experimental setup as given in figure 10. There are three players are involved. First person on the left is called interrogator. He is the evaluator. On the right hand side, there are two players are present. The top one is our artificial intelligence system and the one on the down is the actual person who is having the similar knowledge as of the knowledge possessed by the artificial intelligence machine.Here, the interrogator and evaluator can’t see the things that exist on the right side, as the thick wall divides them.The test is a method of inquiry used for determining whether a computer is capable of thinking like a human being or not.Interrogator will be asking some questions to both of them at the same time and he will not down all the answers given by them. He will have few more questions till he satisfied. Finally, the test is passed only if he is not able to identify which side the machine existsand which side the actual person is sitting.

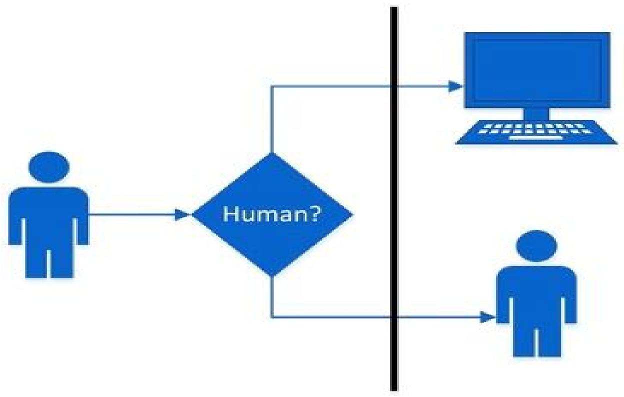


Fig. 10Turing Test

Summary

- Introduced with the basic concepts of Artificial Intelligence.
- Understood the difference between natural intelligence and artificial intelligence.
- Understood the purpose of Turing-test.
- Understood the applications of Artificial Intelligence.
- Studied the various approaches used in Artificial Intelligence.

Keywords

- Artificial Intelligence
- Natural Intelligence
- Expert System
- Robotics

Self Assessment

1. Who is known as the -Father of AI"?

A. Fisher Ada

B. Alan Turing

C. John McCarthy

D. Allen Newell

2. Which of the given language is not commonly used for AI?

A. Perl

B. LISP

C. PROLOG

D. Python

3. The component of an Expert system is_________.

A. Knowledge Base

B. Inference Engine

C. User Interface

D. All of the above

4. An AI agent perceives and acts upon the environment using______.

A. Sensors

B. Perceiver

C. Actuators

D. Both a and c

5. Justify the given statement. "Rational agent always does the right things".

A. True

B. False

6. In state-space, the set of actions for a given problem is expressed by the________.

A. Intermediate States

B. Successor function that takes current action and returns next state

C. Initial States

D. None of the above

7. In the Travelling salesman problem of 'n' cities, the time taken for traversing all cities, without having prior knowledge of the length of the minimum tour will be_______.

A. (n)

B. (n2)

C. (n!)

D. (n/2)

8. The main tasks of an AI agent are_______.

A. Input and Output

B. Moment and Humanly Actions

C. Perceiving, thinking, and acting on the environment

D. None of the above

9. Which of the following is an application of AI?

A. It helps to exploit vulnerabilities to secure the firm

B. Language understanding and problem-solving (Text analytics and NLP)

C. Easy to create a website

D. It helps to deploy applications on the cloud

10. Which of the following is the not an Easy Problem?

A. Natural Language Understanding

B. Chess Playing

C. Theorem Proving

D. All the above

11. Which of the following approach will be used for problem like expert systems.
A. Act like human
B. Act Well
C. Think like Human
D. Think Well

12. What is a state space?
A. It is your Definition to a problem.
B. It is the Problem you design.
C. A space where you know the solution.
D. It is representing your problem with variable and parameter.

13. A production rule consists of _________ .
A. A set of Rule
B. A sequence of steps
C. Both (a) and (b)
D. Dead-end states and goal states.

14. A heuristic is a way of trying ___________.
A. To discover something or an idea embedded in a program
B. To search and measure how far a node in a search tree seems to be from a goal
C. To compare two nodes in a search tree to see if one is better than the other
D. All of the above

15. Which of the following is anexample for the constraint satisfaction problem?
A. Graph-coloring problem
B. Surveillance Problem
C. Tic-Tac-Toe Problem
D. 8-Puzzle Problem

Answers for Self Assessment

1.	C	2.	A	3	D	4.	D	5.	A
6.	B	7.	C	8.	C	9.	B	10.	A
11.	B	12.	C	13.	C	14.	D	15.	A

Review Questions

1. List all the approaches used for Artificial Intelligence.
2. Differentiate artificial intelligence and natural intelligence.
3. Give any two real time applications that use artificial intelligence.

4. Mention the few names of the robots, which are available in the markets.
5. What is the format of defining the production rules?

Further Readings

- Elaine Rich, Kevin Knight and Shivashankar B Nair, "Artificial Intelligence", Third Edition, Tata McGraw Hill Publishing Company Ltd., 2009.
- Stuart Russell and Peter Norvig, "Artificial Intelligence: A Modern Approach", Global Edition, Pearson Prentice Hall, 2016.

Web Links

- https://www.youtube.com/watch?v=Bg_tJvCA8zw
- https://www.youtube.com/watch?v=XuE0GqYHPqQ
- https://www.youtube.com/watch?v=sXx-PpEBR7k

Unit 02: Formulating Problems

CONTENTS

Objectives

1. Understanding the basic concepts of problem-solving methodology.
2. Understanding the importance of state space and different states.
3. Understanding and Solving the Water Jug problem and 8-puzzle problem.
4. Understanding types of problems in Artificial Intelligence.
5. Solving missionaries and cannibal problem with state space representation.

Introduction

In this unit, we will learn the basic concepts of problem-solving methodology or an approach in Artificial Intelligence. This topic discusses the state space representation, which is very important component for problem solving. We will try to understand how the problem statements are consumed, will be understood with three different problem statements such as Water Jug Problem, 8-puzzle problem and Missionaries-Cannibal Problem. Problem Formulation is the important element in developing an artificial intelligent agent, is also introduced in this unit.This unit also discusses the concepts of production system along with their components, and types of problems. You will understand better how the production system works to solve Artificial Intelligence Problems.

2.1 Problem Solving

First of all, before start discussing aboutsolving the problems in Artificial Intelligence we will try to understand how to represent the knowledge in Artificial Intelligence.Intelligenceis the computational part of the ability to achieve goals in the world. Intelligence is a kind of heuristic approach or futuristic or thinking in all the directions for the optimal solution. In short, it can be understood that finding a short cut for a given problem and finds a meaningful solution.

This unit discusses the problem solving methodologies for three problems, that is water jug problem, 8-puzzle problem and missionaries-cannibal problem. There are different types of agents that can be developed. They are Simple Reflex Agent, Model-based reflex agent, Goal-based agents, Utility-based agent and Learning agent. But, here we are going to concentrate on Goal-based agents.

2.2 Formulating Problems

The problems should be analyzed and formulated as per the mentioned components. The components are Goal Formulation, Problem Formulation, Initial State, Actions, Transition Model, Goal State and Cost of the path. These should be clearly mentioned before making artificial intelligence agent for solving the problem.

Goal State : It is the first and simplest step in problem-solving. It organizes the steps/sequence of steps that are required to formulate one goal out of multiple goals as well as actions to achieve that goal.

Problem Formulation: This decides what actions should be taken to achieve the goal state, depending upon the possible moves at the current state.

Initial State: It is the starting state or initial step of the agent towards its goal.

Actions: It is the description of all the possible actions available to the agent on all the possible situations.

Transition Model: It describes what each action does. Describes in terms of expression or equations as a mathematical model.

Goal Test: It determines how to check whether we reached the goal state or not.

Path cost: The problem-solving agent selects a cost function, which reflects its performance measure. Remember, an optimal solution has the lowest path cost among all the solutions.

2.3 Water Jug Problem

Problem Statement

- Given two water jugs as in Figure 1.
- First jug is having the capacity of 4 liters.
- Second jug is having the capacity of 3 liters.
- There is a pump/pipe, you can fill any amount of water and you can waste any amount of water.
- No measuring markers on the Jugs.
- How can you get only 2 liters of water in the 4 liter jug ?

Figure 1 Water Jug Problem

Solving with Natural Intelligence

This is the first problem, we are trying to solve in artificial intelligence. Just go with our natural intelligence. Prepare your mind and set that you are going to watch and understand how our brain is going to see this given problem, what are all the different possibilities it watches, and so on. This is the first step in understanding the natural intelligence. And, if you are not solving any problem with your natural intelligence, then you cannot solve that problem using artificial intelligence for sure. This is because; we are the one who is going to teach the machine what to do step by step to solve the problem. Here are few examples are given below for different approaches to begin solving. Check whether it is matching your thinking.

- Case 1 : Assume both the jugs are empty.
- Case 2 : Assume both the jugs are full.
- Case 3 : Assume First jug has some water. Second jug is empty.
- Case 4 : Assume First jug is empty. Second jug has some water.

Solving with Artificial Intelligence

Here are the steps to follow in the artificial intelligence. Very first step is to create the state space for the said problem. You need to define how you are going to mention your state at any given point of time. Yes, the state can be understood and can be identified as (x, y), where x represents the content of 4-liter jug and y represents the content of 3-liter jug.Here are some production rules, which is having all the possible actions, which can be applicable on satisfying some preconditions. These rules are formed by understanding how your brain think and we should write them all as rules without missing anything from any states.

Rule 1 : Fill the 4 liter jug.

if $(x < 4)$ then $(x, y) \rightarrow (4, y)$.

Rule 2 : Fill the 3 liter jug.

if $y < 3$ then $(x, y) \rightarrow (x, 3)$.

Rule 3 : Pour some water out of the 4 liter jug.

if $(x > 0)$ then $(x, y) \rightarrow (x - d, y)$.

Rule 4 : Pour some water out of the 3 liter jug.

if $(y > 0)$ then $(x, y) \rightarrow (x, y - d)$.

Rule 5 : Empty the 4 literjug on the ground.

if $(x > 0)$ then $(x, y) \rightarrow (0, y)$.

Rule 6 : Empty the 3 liter jug on the ground.

if $(y > 0)$ then $(x, y) \rightarrow (x, 0)$.

Rule 7 : Pour water from 3L jug into 4L jug until the 4L jug is full.

if $(x + y = 4$ and $y > 0)$ then $(x, y) \rightarrow (4, y - (4 - x))$.

Rule 8 : Pour water from 4L jug into 3L jug until the 3L jug is full.

if $(x + y = 3$ and $x > 0)$ then $(x, y) \rightarrow (x - (3 - y), 3)$.

Rule 9 : Pour all the water from 3L jug to 4L jug.

if $(x + y <= 4$ and $y > 0)$ then $(x, y) \rightarrow (x + y, 0)$.

Rule 10 : Pour all the water from 4L jug to 3L jug.

if $(x + y <= 3$ and $x > 0)$ then $(x, y) \rightarrow (0, x + y)$.

Rule 11 : Pour the 2L from 3L jug to 4L jug.

if $(x = 0$ and $y = 2)$ then $(0, 2) \rightarrow (2, 0)$.

Rule 12 : Empty the 2L from the 4 liter jug on the ground.

if $(x = 2)$ then $(2, y) \rightarrow (0, y)$.

Actions

This will be helping in changing the current state into the next state. There are some specific conditions that should be satisfied before selecting the conditions / rules. The below given is the sample for better understanding.

- Action : Pour all the water from second jug.

- Precondition : There should be water in the second jug.
- Effect : New state after the action is performed.
- Current State is (4, 3)➔ Next State : (4, 0)

Solution

The final outcome is the sequence of states from the initial state to the goal state as shown in Figure 2.The transition between the sequences of states will be done as per the possible actions from the respective current state.

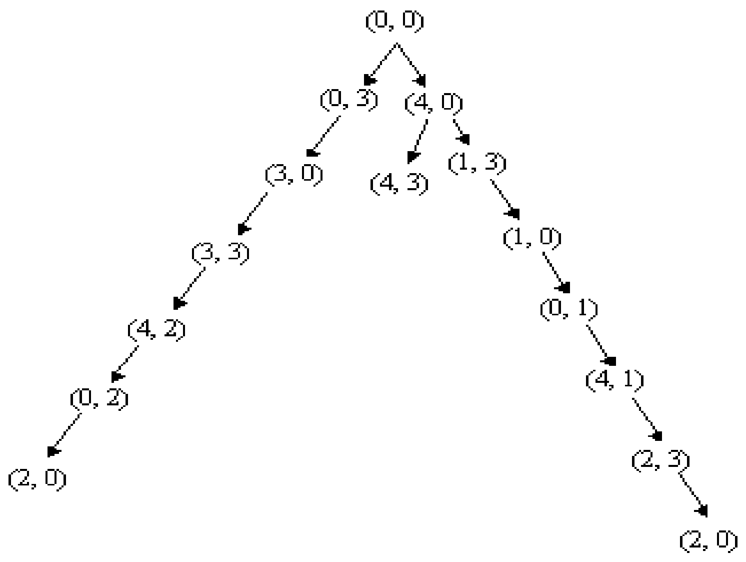


Figure 2State Space Representation for Water Jug Problem

2.4 8-Puzzle Problem

The 8-puzzle problem is a puzzle invented and popularized by Noyes Palmer Chapman in the 1870s. The 8-puzzle problem is actually a board game as shown in Figure 3. The board is described as "It has set off a 3x3 board having 9 block spaces out of which 8 blocks having tiles bearing number from 1 to 8. One space is left blank. The tile adjacent to blank space can move into it. We have to arrange the tiles in a sequence for getting the goal state".

We also know the eight-puzzle problem by the name of N puzzle problem or sliding puzzle problem. N-puzzle that consists of N tiles (N+1 titles with an empty tile) where N can be 8, 15, 24 and so on. In our example N = 8. (that is square root of (8+1) = 3 rows and 3 columns).In the same way, if we have N = 15, 24 in this way, then they have Row and columns as follow (square root of (N+1) rows and square root of (N+1) columns). That is if N=15 than number of rows and columns= 4, and if N= 24 number of rows and columns= 5.

So, basically in these types of problems we have given an initial state or initial configuration (Start state) and a Goal state or Goal Configuration.

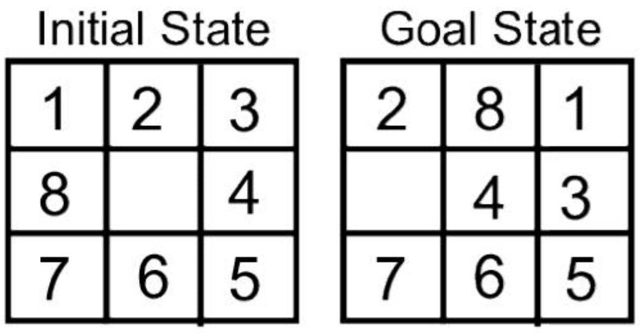


Figure 3 8-Puzzle Problem

The 8-puzzle problem belongs to the category of "sliding block puzzle" type of problem. The 8-puzzle i s a square tray in which eight square tiles are placed. The remaining ninth square is uncovered. Each tile in the tray has a number on it.A tile that is adjacent to blank space can be slide into that space. The game consists of a starting position and a specified goal position. The goal is to transform the starting position into the goal position by sliding the tiles around.

The control mechanisms for an 8-puzzle solver must keep track of the order in which operations are performed, so that the operations can be undone one at a time if necessary.The empty space can only move in four directions (Movement of empty space)

- Up
- Down
- Right or
- Left

State space representation is given in Figure 4.

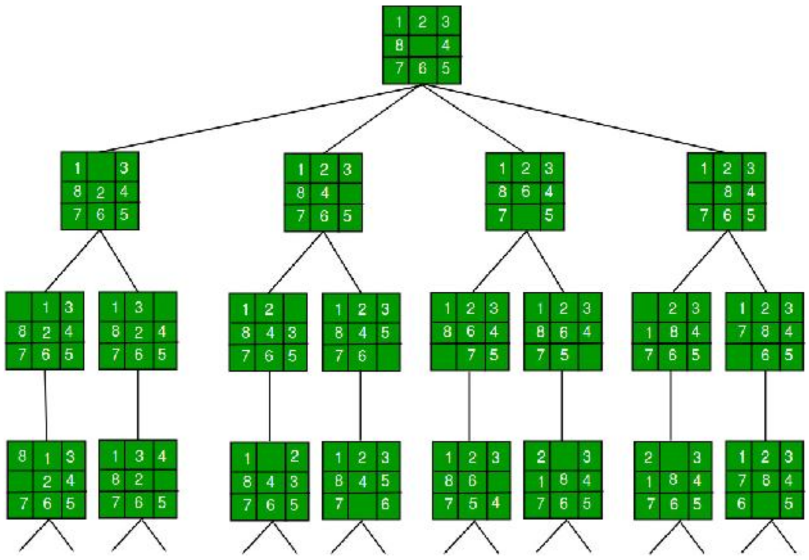


Figure 4 State space representation of 8-Puzzle Problem

Production rules should be formed and used for finding the solution or sequence of states from initial state to the goal state.

2.5 Missionaries and Cannibal Problem

Problem Statement

Three missionaries and three cannibals want to cross a river using a boat as in Figure 5.Boat can carry at most two people at a time.Ifthere are missionaries present on any of the bank then they can not be outnumbered by cannibals.Ifthe missionaries were outnumbered then the cannibals would eat the missionaries. The boat cannot cross the river by itself with nobody on board.How do you help them to reach the destination?

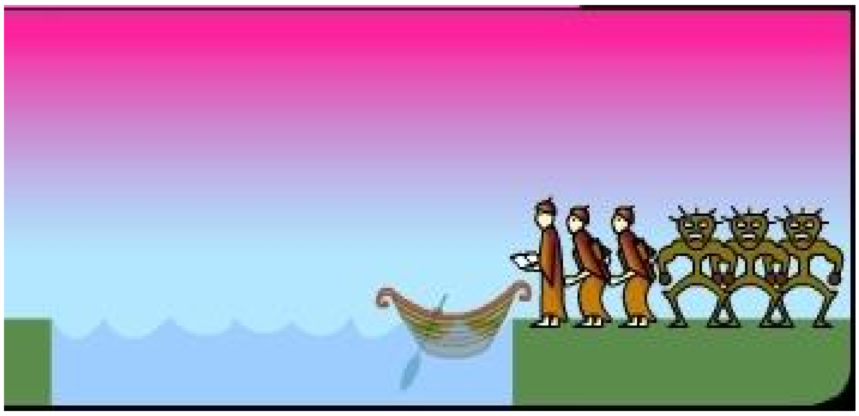


Figure 5 Problem Descriptions of Missionaries and Cannibal Problem

Goal

Move all the missionaries and cannibals across the river.

Constraint

Cannibals on either side of river can never outnumber missionaries, or else the cannibals kill the missionaries and eat them.

Operators

Move boat containing some set of occupants across the river (in either direction) to the other side.

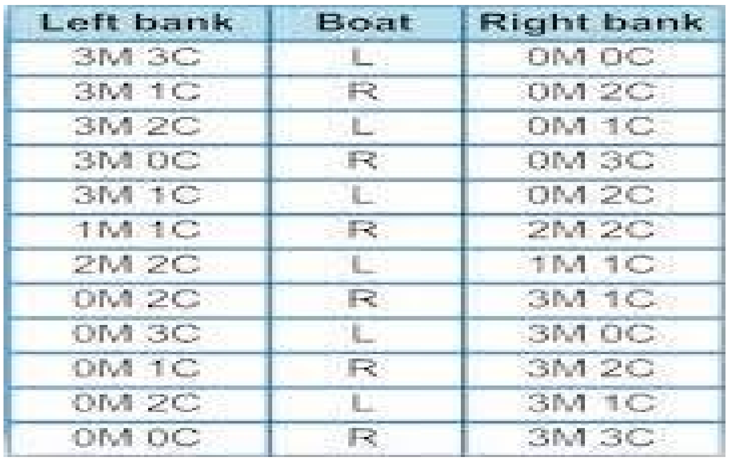
State – How to represent?

There are 3 missionaries, 3 cannibals, and 1 boat. They may be either side of the riverbanks. State should represent the present position of missionaries, cannibals and boat on each side of river.

Production Rules

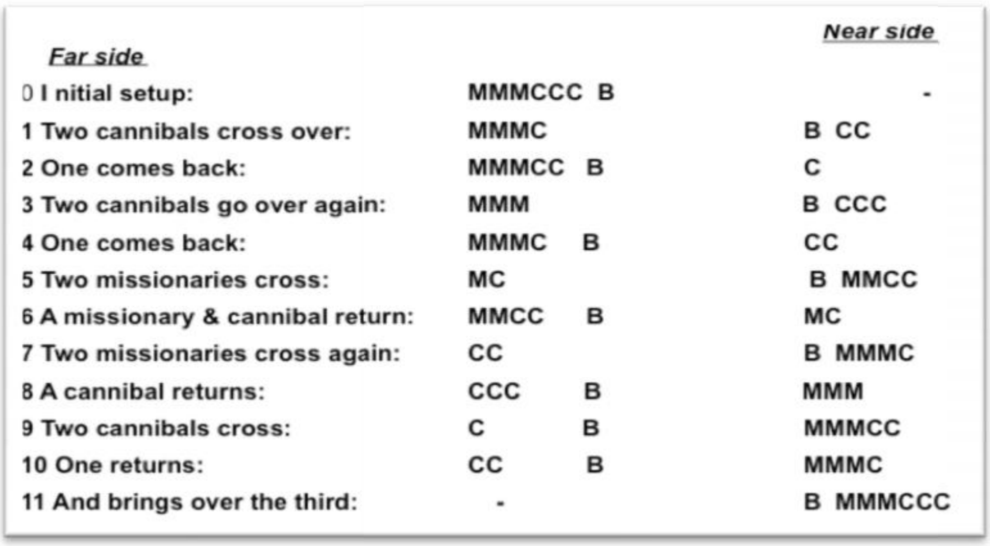
- Mathematical equations need to be developed to satisfy the following constraints.
- Cannibals on either side of river can never outnumber missionaries.
- Boat always has someone in crossing the river.

State Representationand the solution is tabulated in Figure 6 and Figure 7.



Left bank	Boat	Right bank
3M 3C	L	0M 0C
3M 1C	R	0M 2C
3M 2C	L	0M 1C
3M 0C	R	0M 3C
3M 1C	L	0M 2C
1M 1C	R	2M 2C
2M 2C	L	1M 1C
0M 2C	R	3M 1C
0M 3C	L	3M 0C
0M 1C	R	3M 2C
0M 2C	L	3M 1C
0M 0C	R	3M 3C

Figure 6Solutions for Cannibal and Missionaries



Far side		Near side
0 I nitial 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

Figure 7Solutions for Cannibal and Missionaries

2.6 StateSpace

State space is the space where you have all the possible way of solving the given problem is available in some specific form of notations. Separate state space is to be developed for differentproblem individually. Here, we like to introduce few terminologies. The first one is the term ‘state’. State is the current situation or how it looks presently. Try to observe from Figure 8, what is their state or present state.Yes, the first person is trying to kick the second person on his back. It is okay then, what are all the things that can be expected from the current state?

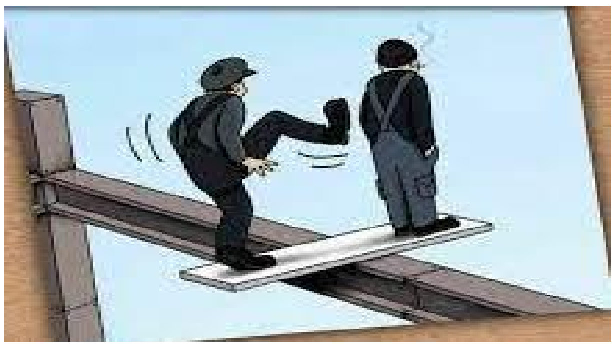


Figure8State, Action and Reaction

Now, let us jump into understanding the state space. We need to have all the options of possible actions and their changes into different states.

Option 1 : First person kicks the second person and the second person falls down and also the first person too falls down. Here, some action you are performing from the current states and subsequently it is leading into other states and go on.

Option 2 : First person don’t kick the second person. Then, both of them are standing on the wood safely.

Option 3 : Second person goes for suicidal attempt. Then, second person falls down and the first person also falls down.

Option 4 : First person goes for suicidal attempt. Then, first person falls down and the second person also falls down.

The state space starts with the initial state and wandering through all the set of states in search of final / goal state. I have given here the idea how to consider all the possibilities from the initial state.It has all the possible states and their consequences. The state space can be understood better in the subsequent discussions on the example problems.

There are few more terminologies on the name of the states, which are very important in developing the state space. They are initial state or start state, current state, goal state and dead end state. Start state or initial state represents the state exists at the beginning of the problem. The next is the current state, which represents the state at a given point of time. The leaf nodes of the state space will be known as goal state. That is otherwise the solution of the problem. There may be more than one solution for some problems. Hence, it is expected to have more than one goal states. Lastly, the state which is in the middle, not having any possible options / actions to move further is called Dead-end state.

Summary

- Understood the problem formulation and kind of solving methodology.
- Solved few example problems step by step to understand the concepts of making AI.
- Understood various types of states and the state space representation.
- Water Jug problem, 8-Puzzle Problem and Missionaries-Cannibal Problem were taken for discussion.
- Importance of state space representation is

Keywords

- Intelligence
- State Space
- Goal State
- Initial State
- Dead State
- Production System

Self Assessment

Q1) Who is known as the -Father of AI"?

A. Fisher Ada
B. Alan Turing
C. John McCarthy
D. Allen Newell

Q2)Which of the given language is not commonly used for AI?

A. Perl
B. LISP
C. PROLOG
D. Python

Q3) The component of an Expert system is_________.

A. Knowledge Base
B. Inference Engine
C. User Interface
D. All of the above

Q4) An AI agent perceives and acts upon the environment using______.

A. Sensors
B. Perceiver
C. Actuators
D. Both a and c

Q5) Justify the given statement. "Rational agent always does the right things".

A. True
B. False

Q6) In state-space, the set of actions for a given problem is expressed by the________.

A. Intermediate States
B. Successor function that takes current action and returns next state
C. Initial States
D. None of the above

Q7) In the TSP problem of n cities, the time taken for traversing all cities, without having prior knowledge of the length of the minimum tour will be_______.

A. O(n)
B. O(n2)
C. O(n!)
D. O(n/2)

Q8) The main tasks of an AI agent are_______.

A. Input and Output
B. Moment and Humanly Actions
C. Perceiving, thinking, and acting on the environment
D. None of the above

Q9). Which of the following is an application of AI?

A. It helps to exploit vulnerabilities to secure the firm
B. Language understanding and problem-solving (Text analytics and NLP)
C. Easy to create a website
D. It helps to deploy applications on the cloud

Q10) Which of the following is the not Easy Problem?

A. Natural Language Understanding
B. Chess Playing
C. Theorem Proving
D. All the above

Q11) Which of the following approach will be used for problem like expert systems.

A. Act like human

B. Act Well

C. Think like Human

D. Think Well

Q12) What is a state space?

A. It is your Definition to a problem.

B. It is the Problem you design.

C. A space where you know the solution.

D. It is representing your problem with variable and parameter.

Q13) A production rule consists of _________ .

A. A set of Rule

B. A sequence of steps

C. Both (a) and (b)

D. Dead-end states and goal states.

Q14) A heuristic is a way of trying ___________.

A. To discover something or an idea embedded in a program

B. To search and measure how far a node in a search tree seems to be from a goal

C. To compare two nodes in a search tree to see if one is better than the other

D. All of the above

Q15) Which of the following is the constraint satisfaction problem?

A. Graph-coloring problem

B. Surveillance Problem

C. Tic-Tac-Toe Problem

D. 8-Puzzle Problem

Answers for Self Assessment

1.	C	2.	A	3	D	4.	D	5.	A
6.	B	7.	C	8.	C	9.	B	10.	A
11.	B	12.	C	13.	C	14.	D	15.	A

Review Questions

1. What do you mean by state space? Why is it important?
2. Describe the types of problems with suitable examples.
3. Explain the problem formulation of 8-Puzzle Problem.
4. List the different states in the missionaries and cannibal problem.
5. What is the format of defining the production rules?

Further readings

- Elaine Rich, Kevin Knight and Shivashankar B Nair, "Artificial Intelligence", Third Edition, Tata McGraw Hill Publishing Company Ltd., 2009.
- Stuart Russell and Peter Norvig, "Artificial Intelligence: A Modern Approach", Global Edition, Pearson Prentice Hall, 2016.

Web Links

- https://www.youtube.com/watch?v=Bg_tJvCA8zw
- https://www.geeksforgeeks.org/problem-solving-in-artificial-intelligence/
- https://www.tutorialandexample.com/problem-solving-in-artificial-intelligence
- https://studyresearch.in/2018/03/11/state-space-representation-consist-of-defining-an-initial-state/
- https://studyresearch.in/2018/03/11/ai_prod/

Unit 03: Uninformed SearchStrategies

CONTENTS

Objectives

1. Understanding the use of searching algorithms in Artificial Intelligence.
2. Understanding the difference between informed and uninformed searches.
3. Understanding the concepts of heuristics functions.
4. Working withuninformed search algorithms with examples.
5. Understanding the merits and demerits of BFS and DFS.

Introduction

In this unit, searching algorithms are discussed in details. It is important in solving the Artificial Intelligence problems. Searching is used mainly selectingthe right production rule to change the states in order to go to the goal state. The travel as usual starts from the initial state. You need to find out the path / solution which will be reaching to the goal state. Hence, you need to have the state space first. This is the pre-requisite to apply the searching algorithms. You can understand how do you travel in the graph or state space. We are discussing all the algorithms and their working principles with examples. You can understand all of them easily. You can also understand their applicability, so that the selection of algorithm will be good if any problem is given. Let us discuss one by one.

Let us have an overview about the production rule a little. The rules in a production system are determined by LHS (left-hand side) and RHS (right-hand side) equations, where LHS denotes the specific condition to be applied, and RHS shows the output of the applied condition. Any production rule will be having the format of IF – THEN Rule.

Generally speaking, any real world problems, must have many options of actions for any possible inputs / situations. According to our approach. Selection of possible actions (we feel right.) will be happened using searching algorithms.Let us get introduced with the term "Production System" too now. Production system is a computer program, which is used to provide a solution for a given problem. Production system is having many components such as database, control system and production rule.

Database

The primary database, which contain all the information necessary to successfully complete the given task.

Set of Production Rules

A set of rules that will be operatingfrom the databases. Each rule consists of a pre-condition and post-condition that the global database either meets or not.

Control System

A control system that acts as the decision-maker, decides which production rule should be applied. The Control system stops computation or processing when a termination condition is met on the database.

Here, searching algorithms plays a vital role to search for a specific production rule in the given situation / given state of condition. Best Searching algorithm should be having less time and with less memory utilization.Now let us start discussing with our first category of searching, i.e., uninformed searching.

3.1 Introduction to Uninformed Search

The searching algorithms is divided into two categories i.e., informed search and uninformed search. We are going to study all the algorithms along with their categories. The first category is uninformed search is discussed below along with their list of algorithms. In this category of search, the machine blindly follows the algorithm regardless of whether right or wrong. We don't consider whether it is efficient or inefficient. Hence, this search is also known as blind search. It will just follow the said procedures.The list of the algorithms on uninformed search category is given below.

- Generate and Test
- Hill Climbing
- Breadth First Search
- Depth First Search

Informed Searchis the second category of searching. This is a technique that has additional information about the search. We can understand like this that we know some clue before start our searching. Expecting that this clue will reduce our searching time. Clue can be anything but depending upon the problem. In fact, we are going to identify and check which clue / hits will work on what problems. This hints / clue can be known as heuristics. This term will refer something as futuristic approach. For example, you are travelling to some place; you are going for the first time. The sign-boards or the distance from your place and the destination is written on the board in many places on the road along with the directions. In particular, the estimate distance from the current state to the goal state is acting as clue / hints. This additional information is obtained using the concept called heuristic approach. The function used to calculate the heuristic value is known as heuristic function.The below is given the list of all the algorithms under informed search.

- Heuristic Approach
- Best First Search
- Bi-Directional Search
- OR Graph
- A* Algorithm

3.2 Generate and Test

This is the first algorithm in uninformed search. The concept is very simple. The figure 1 is showing you the number lock, which is very good example to understand this technique. Assume that you have a number lock and unfortunately you forgot the key. But, you want to open the lock. It’s an emergency. Now, you are in the searching mode in search of the forgotten key. What do you do now?. You will think many options like this or that. You keep on trying. How many options / how many times you will try? When will you stop the trying / searching. The number lock given here is having three digits. Your options are too much. All the options can not be checked one by one. It will take lot of our time, and it may take few days too. So, we don't prefer the serial searching. Instead, we go for random search. This is known as generate and test. This is having only two steps.

- Step 1 : Generate a possible solution.
- Step 2 : Test the solution whether it is correct.

If it is correct then, no more try is needed. If it is not correct then, repeat the steps again and again till you want try.

Figure 1 Generate and Test

3.3 Hill Climbing

Hill climbing algorithm is a local search algorithm and it is one of the uninformed search algorithms. We should apply the search repeatedly and hence, is known as Iterative approach. This technique helps us to find out the better solution than the present solution. Expecting that next solution is better than the present. That is reason why this technique is called as hill climbing. This search will not be smooth always. There are few challenges to be addressed during the search, which can be understood from reference materials given. The simple example is shown in Figure 2.(a). The improvement in the solution is shown in the figure and there are two different solutions such as local maximum and global maximum. Some solution is better at the present environment among its neighboring solutions, is known as local maximum. And, the final optimal solution is known as global maximum as there are no better solutions available for the given problem. We always wish to obtain the global maximum from our searching algorithm. It terminates when it reaches a peak value where no neighbor has a higher value.

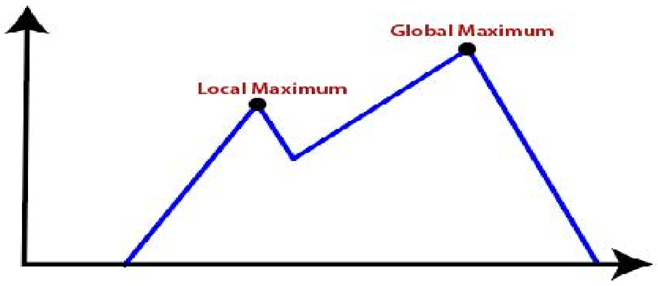


Figure 2 (a) Hill Climbing

This can be explained using an example i.e., 8 – Puzzle Game as given in figure 2 (b). The game is to find out the path the path / steps to reach goal state from the given initial state.

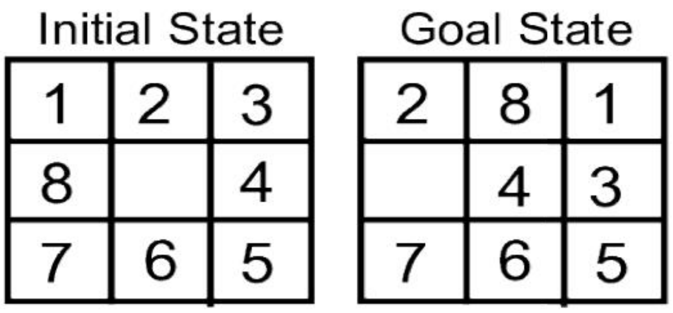


Figure 2 (b) 8-Puzzle Game

From each state, you have the choices of moves / possible next states. Select any move randomly and compare the next state with the current state. If next state is better than the current state, then the next state becomes your current state. And, the above steps will be repeated until you find no more states that are better than the current state and you are reaching the goal state as given above in the figure 2(b). The steps can be understood in simple way as below.

Step 1: Generate the solution along with the direction to move which is used in checking current state and next state whether is better or not.

Step 2: Continuously move in the direction of increasing elevation/value to find the peak of the mountain or best solution to the problem.

There are few techniques to compute the value of the states usingHamming Distance, Euclidean Distance, Manhattan Distance and Minkowski Distance. The following figure 3 describes how to calculate euclidean and manhattan distances which are very often used now a days.

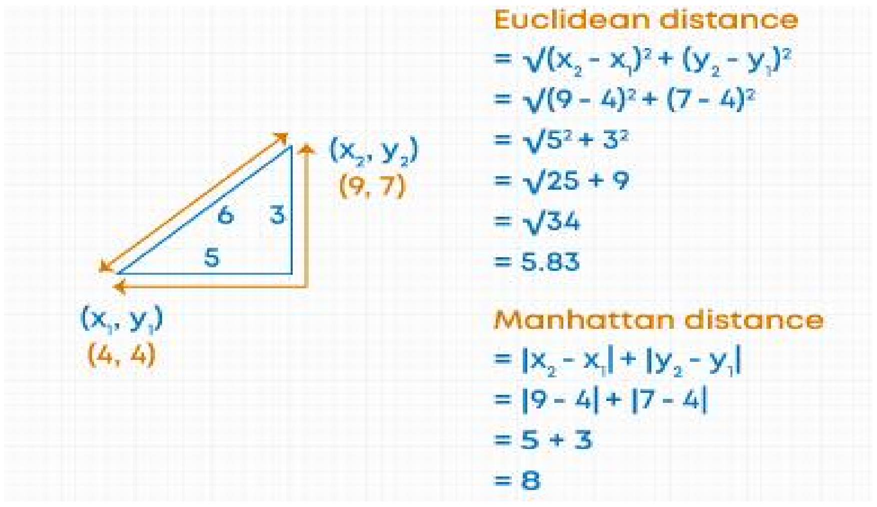


Figure 3 Distance Metrics – Euclidean and Manhattan

Distances will be calculated between the each respective tile in the goal state and the same tile position in the current state. Summation of all the distances will be the current value of the given state. This value can be calculated in any of the distance metrics as mentioned above in Figure 3.

Block world problem - Problem statement

This is one of the most famous problems in artificial intelligence. We will be having few uniform sized blocks i.e., cubes. All the cubes are placed on the table as in Figure 3(a). These blocks are not in order and kept just like that. Consider the Figure 3(a) and our cubes are kept as Marked as Initial State. Each block is placed on another block or on the table. This is not having any order.But, we need to arrange these blocks as shown in the Goal State.

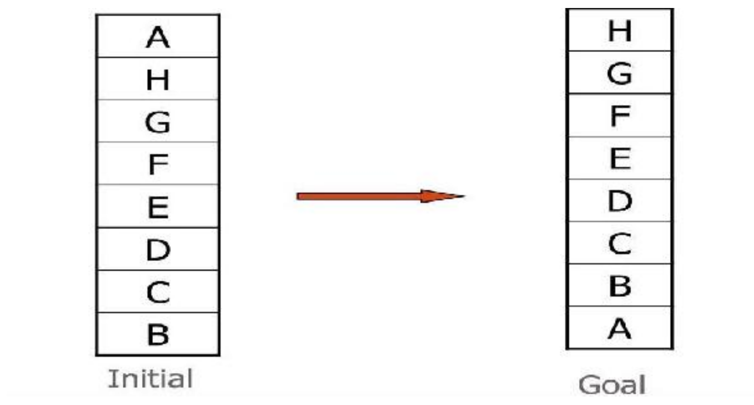
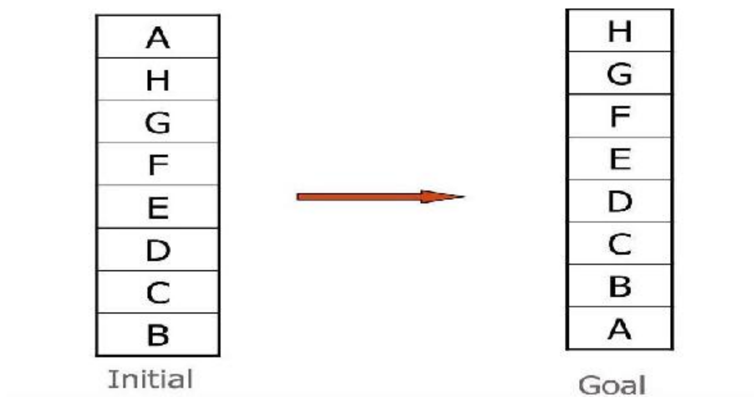


Figure 3 (a) Block World Problem

There is only one action available for the agent to solve the problem. Either place a block on the top ofanother block or place the block on the table. The initial and the goal state are described by the exact position of each block as in Figure 3(a). Assume the cost of each action is 1.

A solution to the blocks world problem is a sequence of actions that allows us to start from an initial state and end up to a goal state. Therefore, the optimal solution is the solution with the minimum cost.



Any heuristics can be used as per the programmers idea and can be found with minimum number of steps to solve the problems or finds the less cost to solve the problems. This problem can be complicated with the different shapes of the blocks and different colors of the blocks.

3.4 Breadth First Search

Breadth first search is also uninformed search algorithms. This search is carried out in the side ways, known as breadth. If it you are travelling from top to bottom straightly, will be known as depth. A simple tree is given in figure 4 to explain the breadth first search. The tree is having level 0, level 1 and level 2. This is understood, as the depth of the tree is 2. We are always starting searching from Level 0. Let us fix value to be searched. That is our Goal State / node. In case, you want to search whether the value 5 is exist or not, we do the following steps.

Step 1: Go to level 0. We have only one node. Check whether 5 exist or not. If exist, report and terminate the search. Otherwise continue to step 2.

Step 2: Go to level 1. We have three nodes. Check one by one from left to right. If does not exist, continue to step 3.

Step 3: Go to level 2. We have four nodes. Again, check one by one from left to right. The number 5 exists in the second node. Report and stop the searching.

Similarly, you can carry out the search in any tree size. This takes quite amount of time for searching as it is happening serially.

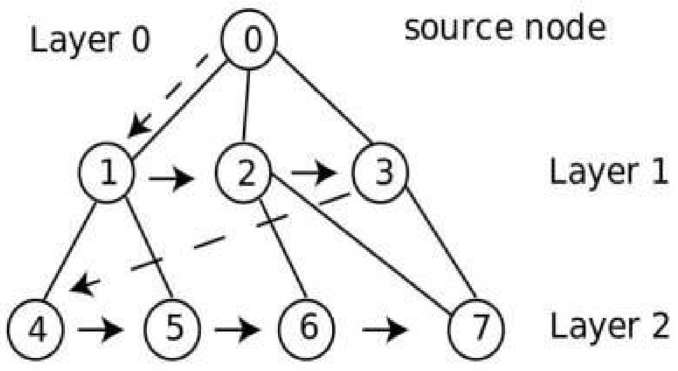


Figure 4Breadth First Search

Example

Consider this example with the mentioned tree structure as in Figure 5. We just try to implement Breadth First Search algorithm and trace what could be the outcome.

```
        A
       / \
      B   C
     /   / \
    D   E   F
```

Figure 5Example for Breadth First Search

The output will be as mentioned below as in Figure 6.

```
A, B, C, D, E, F
```

Figure 6Example for Breadth First Search - Output

3.5 Depth First Search

Depth first search is also uninformed search algorithms. Here the search is carried out like travelling from the top to bottom mannerknown as depth first search. The tree given below in figure 7 is having level 0, level 1, level 2 and level 3. This is understood, as the depth of the tree is 4.

The algorithm starts at the root (top) node of a tree. It goes as far as it can down a given branch (path). Then backtracks until it finds an unexplored path, and then explores it. The algorithm does this until the entire graph has been explored. The travelling can be understood from the figure 7.Let us fix value to be searched.

In case, you want to search, whether the letter 'e' exist or not in the given diagram. Then it is going to be our Goal State / node.We do the following steps.

Step 1: Go to initial node (node 'a'). Check whether the letter 'e' exists or not. If it is the goal node, report and terminate the search. Otherwise continue to step 2.

Step 2: Go to node 'b'. If it is not the goal node, continue to step 3.

Step 3: Go to node 'd'. If it is not the goal node, continue to step 4.

Step 4: Go to node 'h'. If it is not the goal node, continue to step 5.

Step 5: Go back (backtracking) and check whether any node is having some other branches. We find node 'b' is having a branch.

Step 6: Go to node 'e'. Yes. It is our goal node. Report and Stop the search.

The search always goes left side to the depth and then starts backtracking. Once, it finds any branch during backtracking the same procedure is applied. Go depth in the left side and backtracking. Hence, we are giving priority on the depth rather than breadth.

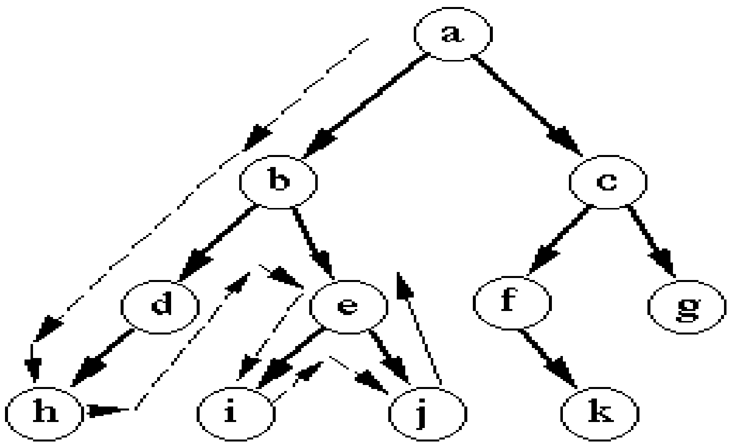


Figure 7Depth First Search

Example

Consider this example with the mentioned tree structure as in Figure 8. We just try to implement and Breadth First Search algorithm and trace what could be the outcome.

```
      A
     / \
    B   C
   /   / \
  D   E   F
```

Figure 8 Example - Depth First Search

The output will be as mentioned below in Figure 9.

```
A, B, D, C, E, F
```

Figure 9 Example - Depth First Search - Output

3.6 Heuristics

We have already discussed in previous section about heuristics. Still, it can be clear from the following figure 10. In short, a heuristic function or simply a heuristic is a shortcut to solving a problem when there are no exact solutions for it or the time to obtain the solution is too long.

The figure is the map of cities such as A, B, C, D, E, F and G. The distances between the two cities are given, as per the road availability. Assume the goal state as G and start state as A. We have to find out the shortest route / path from city A to G.

Here, the heuristics / clue which is going to help you in searching the path comfortably is the distance measures. In fact, well-designed heuristic function will play an important role in guiding a search process toward a solution.

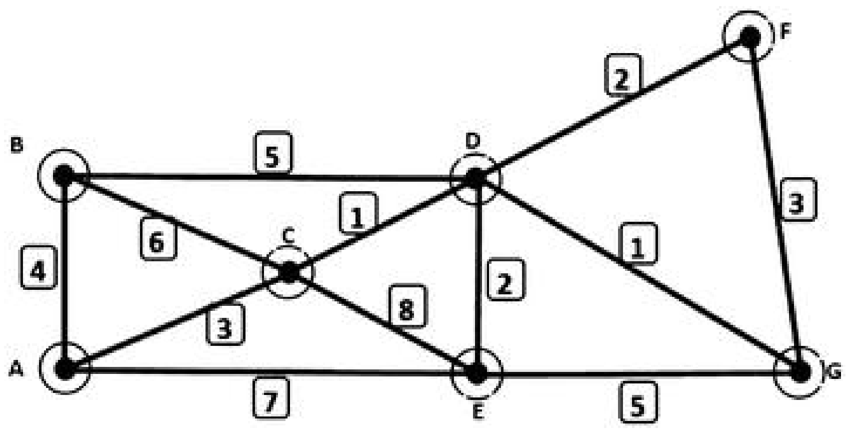


Figure 10Heuristics

Heuristic function estimates how close a state is to the goal. It is represented by h(n), and it calculates the cost of an optimal path between the pair of states. The value of the heuristic function is always positive. Admissibility of the heuristic function is given as: h(n) <= h*(n).

Summary

The use of searching algorithms in Artificial Intelligence is discussed along with their types. Understood the difference between informed and uninformed searches.The concepts of heuristics and heuristic functions are elaborated. Various algorithms under uninformed searching algorithm are discussed. Principles and workings are explained with examples in step-by-step. By this time, selection of suitable algorithm for a given problem is also to be known. The differences between the Breadth first search and depth first search is understood. The hill-climbing algorithm is also discussed.

Keywords

- Informed search
- Uninformedsearch
- Generate and test
- Breadthfirst search
- Depthfirst search
- Heuristics

Self Assessment

Q1) The Data structure used in standard implementation of Breadth First Search is?

A. Stack
B. Queue
C. Linked List
D. Tree

Q2) A person wants to visit some places. He starts from a vertex and then wants to visit every place connected to this vertex and so on. What algorithm is most suitable for him to use?

A. Depth First Search
B. Generate and Test Search
C. Breadth First Search
D. A* algorithm

Q3) In BFS, how many times a node is visited?

A. Once
B. Twice
C. Equivalent to number of in-degree of the node
D. Thrice

Q4) What is the other name of informed search strategy?

A. Simple search
B. Heuristic search
C. Online search
D. None of the mentioned

Q5) Which is used to improve the performance of heuristic search?

A. Quality of nodes
B. Quality of heuristic function
C. Simple form of nodes
D. None of the mentioned

Q6) Which search strategy is also called as blind search?

A. Uninformed search
B. Informed search
C. Simple reflex search
D. All of the mentioned

Q7) How many successors are generated in backtracking search?

A. 1
B. 2
C. 3
D. 4

Q8) What is the space complexity of Depth-first search?

A. O(b)
B. O(bl)
C. O(m)
D. O(bm)

Q9) Which search method takes less memory?

A. Depth-First Search
B. Breadth-First search
C. Optimal search
D. Linear Search

Q10) ________ search algorithm is a very simple algorithm that guarantees to find a solution if done systematically and there exists a solution.

A. Generate-and-Test
B. Simple Hill Climbing
C. Steepest-Ascent Hill Climbing
D. Simulated Annealing

Q11) Consider the following statement:

"The search first begins from the root node and the first one of the child node's sub-tree is completely traversed. That is, first all the one-sided nodes are checked, and then the other sided nodes are checked."

Which search algorithm is described in the above definition?

A. The Breadth First Search (BFS)
B. The Depth First Search (DFS)
C. The A* search
D. None of the above

Q12) What is Time Complexity of Breadth First search algorithm?

A. b
B. b^d
C. b^2
D. b^b

Q13) Which of the following are the drawbacks of hill climbing.

(i) Local maximum

(ii) Plateau

(iii) Ridge

A. (i) and (ii) only

B. (ii) and (iii) only

C. (i) and (iii) only

D. All (i), (ii) and (iii)

Q14) Hill Climbing algorithm terminates when ___________ .

A. Stopping criterion met
B. Global min / max is achieved
C. No neighbor has higher value
D. All of the mentioned.

Q15) What is Time Complexity of Depth First search algorithm?

A. b
B. b^d
C. b^2
D. b^b

Answers for Self-Assessment

1.	B	2.	C	3	C	4.	B	5.	B
6.	A	7.	A	8.	D	9.	A	10.	A
11.	B	12.	B	13	D	14	C	15	B

Review Questions

1. Differentiate informed and uninformed search strategies.
2. What do you understand by the concept of heuristics and heuristic function?
3. Explain theworking principles of BFS and DFS with an example.
4. Listthe algorithms under uninformed search.
5. Give the merits and demerits of BFS and DFS.

Further readings

- Elaine Rich, Kevin Knight and Shivashankar B Nair, "Artificial Intelligence", Third Edition, Tata McGraw Hill Publishing Company Ltd., 2009.
- Stuart Russell and Peter Norvig, "Artificial Intelligence: A Modern Approach", Global Edition, Pearson Prentice Hall, 2016.

Web Links

- https://www.javatpoint.com/search-algorithms-in-ai
- https://www.tutorialspoint.com/artificial_intelligence/artificial_intelligence_popular_search_algorithms.htm
- https://www.youtube.com/watch?v=XCPZBD9lbVo&list=PLbMVogVj5nJSFZoiF6RDqyz_m6Srjx_MY

Unit 04: Informed Search Strategies

CONTENTS

Objectives

- Understanding the use of searching algorithms in Artificial Intelligence.
- Understanding the iterative deepening depth first search with an example.
- Understanding the concepts of Best First Searchalgorithm.
- Understanding the working principles ofA* Algorithm with examples.
- Understanding the IDA and SMA*with examples.

Introduction

In this unit, searching algorithms are discussed in detail. This is very important in solving the Artificial Intelligence problems. Searching is used mainlyfor selectingthe right production rule to change one state to next state in order to go to the goal state. The traversal as usual starts from the initial state. You need to find out the path / solution which will be reaching to the goal state. Hence, you need to have the state space first. This is the pre-requisite to apply the searching algorithms. You can understand how you travel in the graph or state space. There are many methods or approaches to travel the graphs. This graph may be ofAO graph or AND graph which are discussed at the end of the unit. We are discussing all the algorithms under informed search category and their working principles with examples. You can understand all of them easily. You can also understand their applicability, so that the selection of algorithm will be good if any real time scenario is given. Let us discuss one by one.

4.1 Introduction to Informed Search

The searching algorithms is divided into two categories i.e., informed search and uninformed search. The uninformed search algorithms are already discussed in the previous unit. Here, we are going to study the informed search algorithms. This is a technique that has some additional information about the search, and hence it makes the searching process easy. We can also understand that what if we know some clue before start our searching process. This will definitely make us feel that this clue will reduce our searching time. Clue can be anything but depending upon the problem. In fact, we are going to identify and check which clue / hits will work on what problems. This hints / clue can be known as heuristics. This term will refer something as futuristic approach. For example, you are travelling to some place; you are going for the first time. The sign-boards or the distance from your place and the destination is written on the board in many places on the road along with the directions. In particular, the estimate distance from the current state to the goal state is acting as clue / hints. This additional information is obtained using the concept called heuristic approach. The function used is known as heuristic function.The below is given the list of all the algorithms under informed search.

- Best First Search
- Bi-Directional Search
- AND / OR Graph
- A* Algorithm
- Iterative Deepening Depth First Algorithm
- SMA*

These algorithms will be discussed in detail in the following sections.

4.2 Best First Search

This is the first algorithm in informed search category. Here, the searching is little advanced. We knowhow to select the better node among the given possible nodes.We are selecting the best node among the possible nodes is called best first search algorithm. This is done at every step to movefrom the current state to the next state.Node is selected for expansion based on an evaluation function f(n).This is explained in the figure 1.This uses the concept of a Priority queue and heuristic search. Combining the two is to follow a single path at a time, but switch paths whenever some competing paths look more promising than the current one.

Let us assume our initial state is A and the goal state is J. Let us start the search in the given tree below. Every node is given some cost value (distance measure) and it can be assumed as our heuristics. In our problem, we assume that better node is the one, which is having minimum heuristic value. As usual, start from the root node, check whether it is the goal node or not. Then, expand the node and check their children.

Step 1: Start from root node. It is not the goal node. If the best node is not the goal state, then, Go to step 2.

Step 2: Expand the root node and you have three children B, C and D. Select the best node among three nodes which is having minimum heuristic value. So, node D is the best node. If the best node is not the goal state, then, go to step 3.

Step 3: Expand the node D. You have two children E and F. In this case, you have other options also such as select the best node among nodes B, C, E and F. Here, node B is the best node as it is having minimum heuristic of value 3. If the best node is not the goal state, then, go to step 4.

Step 4. Expand the node B. You have children G and H. In this case, your options are G, H, C, E and F. Best node is E as it is having minimum value of 4. If the best node is not the goal state, then, Go to step 5.

Step 5. Expand the node E. You have children I and J. Your options are G, H, I, J and F. The minimum heuristic is 1. The best node is J. The best node is our goal state. Hence, our searching is completed.

Step 6. The path we travelled is the solution for the problem.

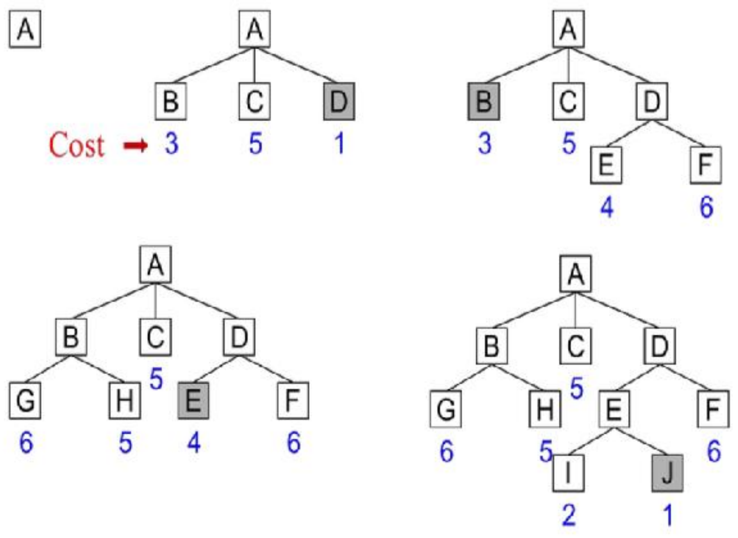


Figure 1Working with Best First Search - Example 1

In hill climbing algorithm, one move is selected,and all the others are rejected. Those nodes neverbe reconsidered again. But, in Best First Search, one move is selected, but the other moves are kept around so that they can be revisited later if the selected path becomes less promising. You can try with another example given below in the Figure 2. Find the path to reach Node (L) from (A) using best first search algorithm like explained above.

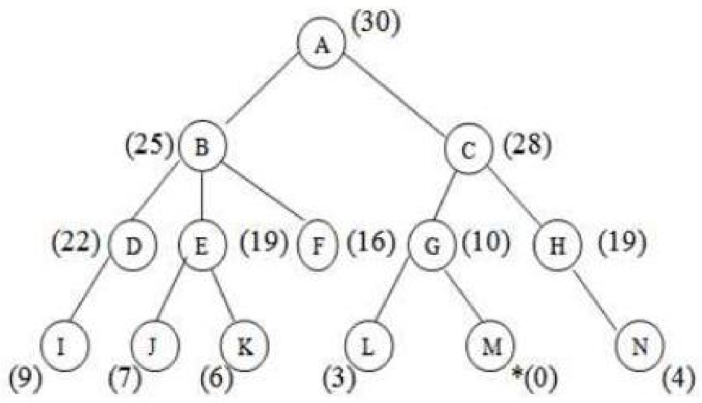


Figure 2Working with Best First Search–Example 2

4.3 Bi-Directional Search

This is also one of the informed search algorithms. This bi-directional search encourages to performin two different directions.One search will go from top to bottom and another search will go from bottom to top. As it is doing two directions, it is known as bi-directional search. In fact, first graph starts from initial state and the second graph starts from goal state. These searching will be performed at the same time simultaneously. The two different searches will find a common node to connect the paths. This will reduce the execution time by half. The search will terminate when two graphs intersect each other with a common node. Let us have an example as shown in the figure 3. Let the start state is 1 and goal state is 16.

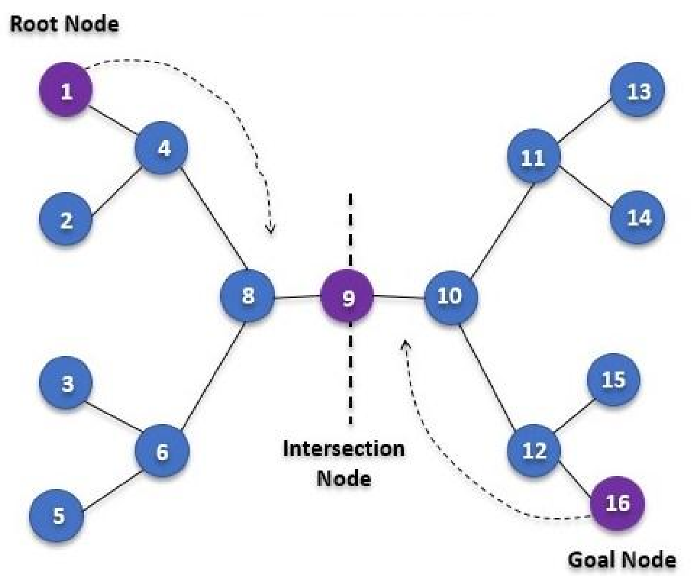


Figure 3Bi-directional Search

We can use any searching algorithm for searching such as DFS, BFS and so on. There is no restriction, but we should do this in two directions (top to bottom and bottom to top). Hence, it is called bi-directional search.

4.4 AND / OR Graph

This section discusses the type of graphs / trees that are used in the previous sections for searching. AND / OR is a form of graph, which used to decompose the problem into a set of smaller problems, which helps in problem solving.The default graphs as shown in the figure 4 are called OR Graph. We have been working with this kind of graphs from the beginning of this section. The node selection is done among the possible nodes such as this node or that node. Every time, only OR is used for node selection.

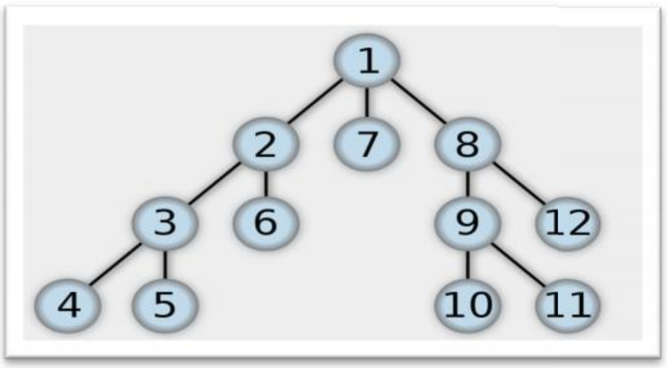


Figure 4OR Graph

Let us look at the level 1. The nodes are 2, 7 and 8. These nodes are joined in OR condition. We can select any one from the three nodes i.e., do select: 2 or 7 or 8. All the nodes are individually responsible states. All are independent in specific. And the other graph as shown in figure 5 is called AND Graph. Few nodes are not independent. Dependent nodes are connected with a curve symbol. This is the symbol for AND to represent the dependency. You cannot select any one by chance. If you want to select, you need to consider them both for any decision-making. This is clearly visible in figure 10.

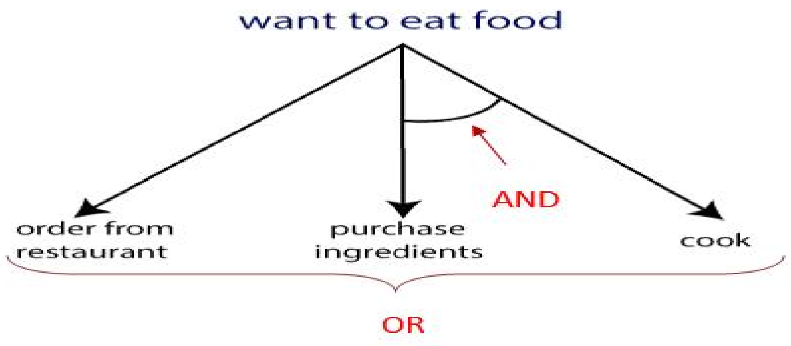


Figure 5AND Graph

We try to understand the options given from the figure 5. The root node is "Want to eat food?" . There are only two options. The first is "order from restaurant", is independent. The second node and third nodes are dependent, as "Cook" is not performed without "purchase ingredients". Hence, the second options becomes like "purchase ingredients " AND "cook". Hence, this graph is called AND Graph.

Working with AND Graph

We are already familiar with working with OR graphs. So, we will focus on the working of AND Graph in detail. The example is given in the figure 6. Let us try to select the best node fromthe following example.

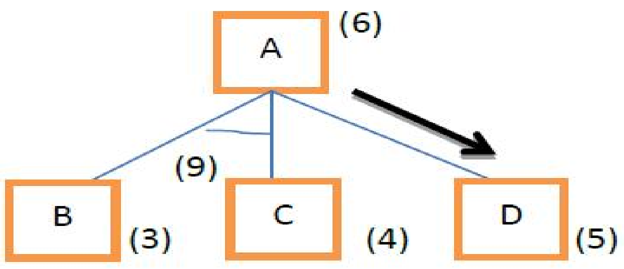


Figure 6AND Graph - Example

Yes, the example is having the heuristic values. The heuristic value of nodes B is 3, C is 4 and D is 5. The minimum is to be selected. In this case, node B and node C are dependent. Hence, heuristic values should be added together for them. And, node D has the same heuristic, as it is independent. The calculation is depicted below. The cost between two nodes are assumed as 1, if not mentioned.

A to (B and C) : f (B and C) = (1 + 3) + (1 + 4) = 9

A to D : f (D) = 1 + 5 = 6

The minimum is 6. Hence, Node D is selected.

4.5 A* Algorithm

A* is the best-known form of best-first search.This is, one of the best and popular techniques used in graph traversals and path-finding. Path finding can be understood easily from the figure 7 given below.

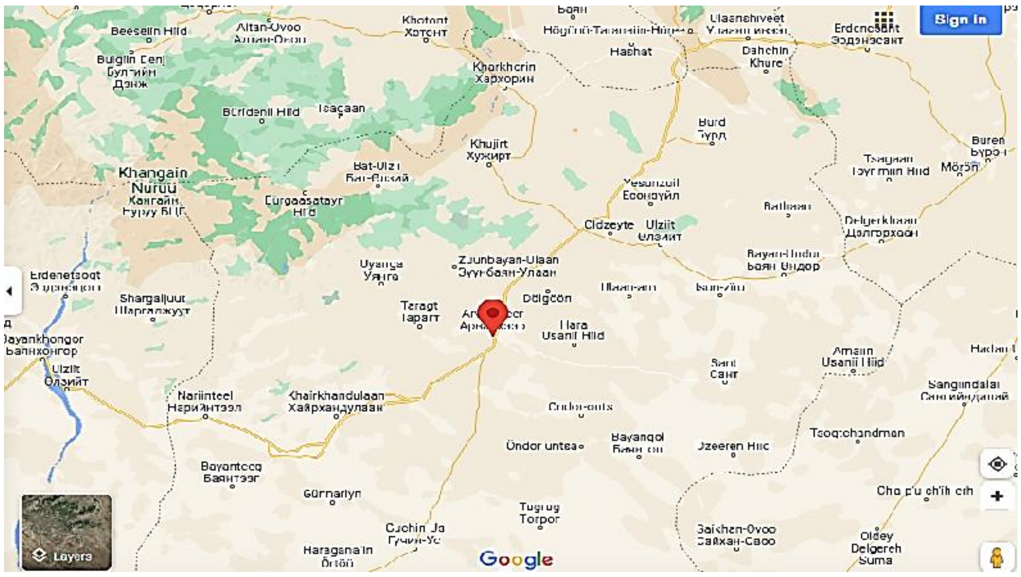


Figure 7Google Map

Also, in specific, you can understand the path finding from a given graph as in figure 8.Let we start from node ‘a’ and wish to go node ‘z’.Try to traverse the graph for the shortest route.

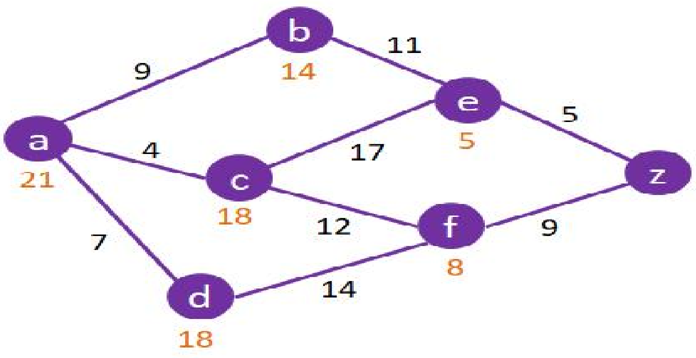


Figure 8Graph – An Example

A* Algorithm is using two parameters which are written as g and h. The distance from the current node to the goal node, noted by ‘h’.The distance travelled so far to reach the current node, noted by ‘g’. The final heuristics is calculated using a function known as heuristic function. The heuristic function of A* algorithm is given below.

f (n) = g (n) + h (n)

Let us have an example from the figure 9. In this case, try to select the best node from A or (B and C).

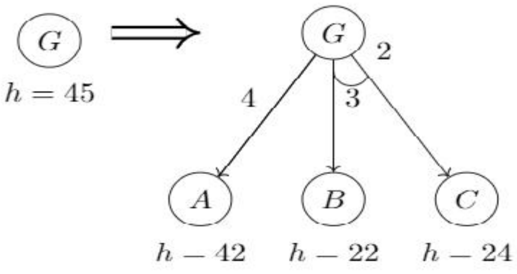


Figure 9AND Graph – Example

- G to A : f (A) = 4 + 42 = 46
- G to (B and C) : f (B and C) = (1 + 22) + (2 + 24) = 49

The minimum is 46. Hence, Node A is selected.

Little complex example is given below as in figure 10. Let us try to give a try selecting the best node again from the root node A.

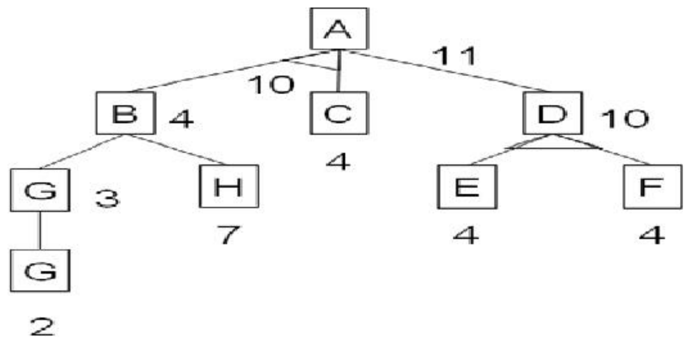


Figure 10AND Graph – Example

- A to (B and C) = f(B and C) = 4 + 1 + 4 + 1 = 10
- A to D = f(D) = 10 + 1 = 11

Hence, the node (B and C) is selected.

Please refer the graph given in figure 8. Let we start from (initial node) node 'a' and wish to go (goal node) node 'z'. Try to traverse the graph for the shortest route. Follow the steps one by one.

Step 1:Now the current node is 'a'. Calculate the heuristics. The details are given in the Table 1.

Table 1 Heuristic Calculation from node 'a'

Path	Value of 'g'	Value of 'h'	Value of 'f'
Node a to b	9	14	23
Node a to c	4	18	22
Node a to d	7	18	25

The value of 'f' is minimum in 'a to c'. So, Path 'a to c' is selected.Hence, next node is 'c'.

Step 2:Now the current node is 'c'. Calculate the heuristics. The details are given in the Table 2.

Table 2 Heuristic Calculation from node 'c'

Path	Value of 'g'	Value of 'h'	Value of 'f'
Node c to e	17	5	22
Node c to f	12	8	20

- The value of 'f' is minimum in 'c to f'. So, Path 'c to f' is selected.
- Hence, the next node is 'f'.

Step 3:Now, the current node is 'f'.There is no multiple options from node 'f'. Has only one route. So, there is no need to calculate the heuristics. The default next node is 'z'.

Table 3 Heuristic Calculation from node 'f'

Nodes	Value of 'g'	Value of 'h'	Value of 'f'

Node f to z	9	0	9

- The value of ‘z’ for the path ‘f to z’ is 9. Of course, the next node is ‘z’.

Step 4: The optimal path (solution) for the given problem is given below.

- The final path is :‘a’→‘c’→‘f’→‘z’.
- Total cost = 4 + 12 + 9 = 31.

4.6 Iterative Deepening Algorithm

The iterative deepening algorithm is a combination of DFS and BFS algorithms. This search algorithm finds out the best depth limit and does it by gradually increasing the limit until a goal is found. This algorithm performs depth-first search up to a certain "depth limit", and it keeps increasing the depth limit after each iteration until the goal node is found.This Search algorithm combines the benefits of Breadth-first search's fast search and depth-first search's memory efficiency.The iterative search algorithm is useful uninformed search when search space is large, and depth of goal node is unknown. Let us assume a graph as given in the figure 11.

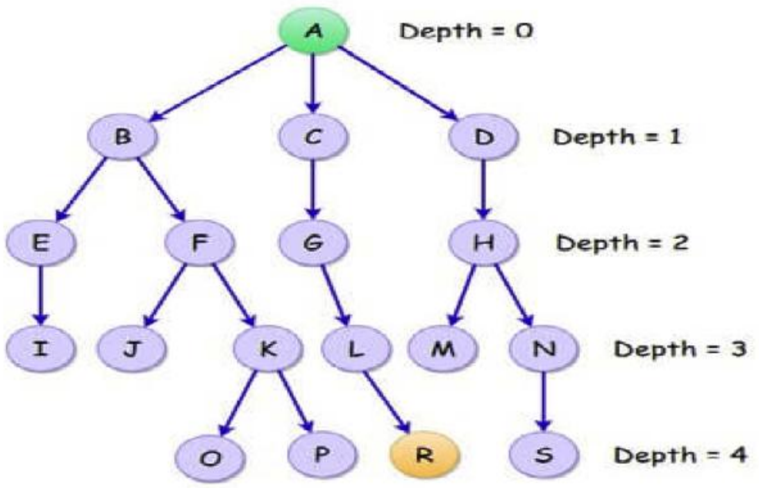
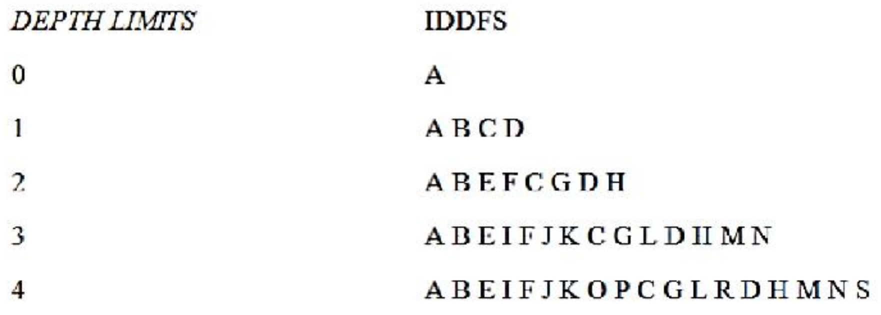


Figure 11 IDDFS – Example

Starting node is Node A. Let us examine the searching path for each depth levels starting from 0, 1, 2, and so on. Maximum depth is only 4 as mentioned in the figure 11. The following figure 12 is showing the searching and how the node is traversed. Those nodes are given against each depth limits.



DEPTH LIMITS	IDDFS
0	A
1	A B C D
2	A B E F C G D H
3	A B E I F J K C G L D H M N
4	A B E I F J K O P C G L R D H M N S

Figure12 Path Traversal of IDDFS

Few points to remember as merits:

- Combine the advantage of both DFS and BFS
- Required less amount of memory

- It is Complete & Optimal
- Will not go in infinite loop

Few points to remember as demerits:

- Regressive Recursion is required
- The main drawback of IDDFS is the time and wasted calculations that take place at each depth.

4.7 Simplified MemoryBounded A* (SMA*)

SMA* (Simplified Memory Bounded A*) is a shortest path algorithm that is based on the A* algorithm. The difference between SMA* and A* is that SMA* uses a bounded memory, while the A* algorithm might need exponential memory.All other characteristics of SMA* are inherited from A*.Like the A*, it expands the most promising branches according to the heuristic. What sets SMA* apart is that it prunes nodes whose expansion has revealed less promising than expected.SMA*, just like A*, evaluates nodes byf(n) = g(n) + h(n).

Since g(n) gives the path cost from the start node to node n, and h(n) is the estimated cost of the cheapest path from n to the goal, we have f(n) = estimated cost of the cheapest solution through n. The lower the f value is, the higher priority the node will have.The difference from A* is that the f value of the parent node will be updated to reflect changes to this estimate when its children are expanded. A fully expanded node will have an f value at least as high as that of its successors.In addition, the node stores the f value of the best-forgotten successor (or best forgotten child). This value is restored if the forgotten successor is revealed to be the most promising successor. We have an example how to work with SMA* using the given figure 13.

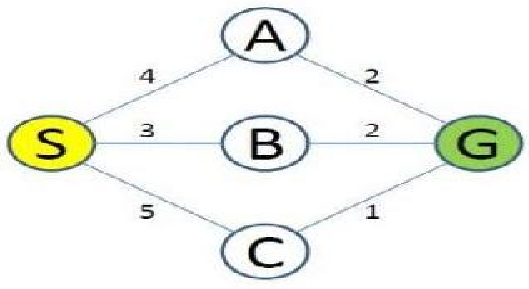


Figure 13 Example for SMA*

Assume the heuristics values 3, 0, 2, 1 and 0 for the nodes S, A, B, C and G respectively. Let the memory is availableto store only 3 nodes. Let us begin how to apply searching process using simplified bounded memory of 3 nodes.Our requirement is to find the shortest path starting from node "S" to node "G". We can see the graph as in Figure 13 and how the nodes are connected. Also, we can remember the heuristic values for each of the nodes mentioned above.

Step 1:Start with expanding the Figure 13 from the nodes S and then A and B. Also, mention their corresponding f(n) values. Remembers the restriction that we have only bounded memory of 3 nodes. We will get as Figure 14.

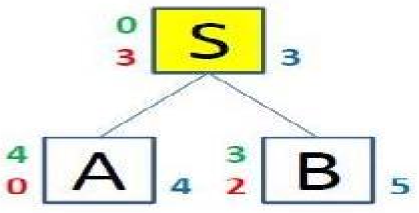


Figure 14 SMA* Implementation - Step 1

Step 2:Try expanding the next node that is C.We can see now what happens when we want to evaluate more nodes than what the memory allows. The already evaluated children with the highest f value get removed, that is, the node B is removed with f(n) value of 5, but its value is remembered in its parentnode S. Refer the Figure 15.

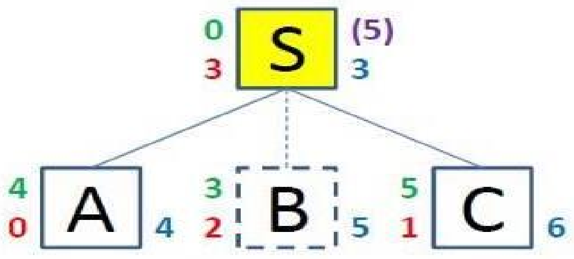


Figure 15. SMA* Implementation - Step 2

Step 3:Similarly, we can see that if all accessible children nodes are visited or explored we adjust the parent's f value to the value of the children with the lowest f value. Refer the Figure 16.

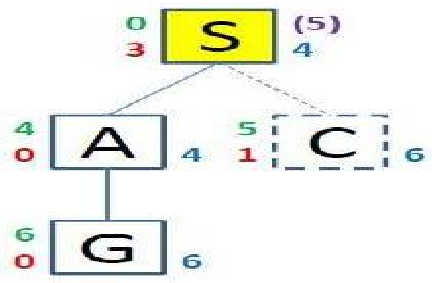


Figure 16 SMA* Implementation - Step 3

Step 4:We can see how the algorithm evaluates the children of the "A" node. The only child is the "G" node that is the goal, which means there is now a way for exploring further. Now we remove the "C" node, since it has the highest f value but we don't remember it, since the "B" node has a lower value. Now we update the f value of the "A" node to 6 because all the children accessible are explored as in Figure 17.

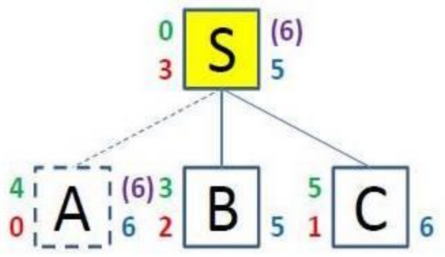


Figure 17 SMA* Implementation - Step 4

Here, we will remove the "A" node since it has already discovered the goal and has the same f value as the "C" node so we will remember this node in the parent "S" node.

Step 5: Now, we are expanding the nodes from node B as in Figure 18.

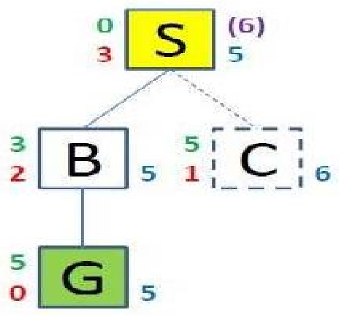


Figure 18 SMA* Implementation - Step 5

We can see that we've reached the goal again, this time through the "B" node. This time cost is lower than through the "A" node. The memory is full, which means we need to remove the "C" node, because it has the highest f value. At this stage, the algorithm terminates, and we have found the shortest path from the "S" node to the "G" node.

Summary

The use of searching algorithms in Artificial Intelligence is discussed along with their types. Understood the difference between informed and uninformed searches.The concepts of heuristics and heuristic functions are elaborated. Various algorithms under each type of searching algorithm are discussed. Principles and workings are explained with examples in step-by-step. And, we have seen what AO and AND are graphs and how to traverse in those graphs. By this time, selection of suitable algorithm for a given problem is known.

Keywords

- Informed search
- best first search
- bi-directional search
- OR / AND graph
- A* Algorithm
- Iterative Deepening Depth First Search
- Simplified Memory Bounded A*

Self Assessment

1. A person wants to visit some places. He starts from a vertex and then wants to visit every place connected to this vertex and so on. What algorithm is most suitable for him to use?

A. Depth First Search
B. Generate and Test Search
C. Breadth First Search
D. A* algorithm

2. In BFS, how many times a node is visited?

A. Once
B. Twice
C. Equivalent to number of in-degree of the node
D. Thrice

3. What is the other name of informed search strategy?
A. Simple search
B. Heuristic search
C. Online search
D. None of the mentioned

4. Which is used to improve the performance of heuristic search?
A. Quality of nodes
B. Quality of heuristic function
C. Simple form of nodes
D. None of the mentioned

5. Which search strategy is also called as blind search?
A. Uninformed search
B. Informed search
C. Simple reflex search
D. All of the mentioned

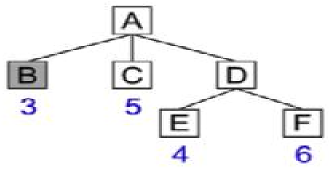
6. How many successors are generated in backtracking search?
A. 1
B. 2
C. 3
D. 4

7. Which search method takes less memory?
A. Depth-First Search
B. Breadth-First search
C. Optimal search
D. Linear Search

8. A* algorithm is based on ________ .
A. Breadth First Search
B. Depth First Search
C. Best First Search
D. Hill Climbing

9. ___________ algorithm uses the priority queue for the best node selection.
A. Best First Search
B. A* algorithm
C. Iterative Deepening
D. None of the above

10. What is the best node in the given diagram ?



A. 6
B. 5
C. 4
D. 3

11. _______________ algorithm does the searching from top to bottom and also from bottom to top at a time to search the path from initial node to goal node.
A. A* algorithm
B. Bi-directional search
C. Iterative Deepening
D. None of the above

12. In A* algorithm, what the termg(n) refers in the heuristics calculation ?
A. The value corresponds to initial node to goal node
B. The value corresponds to current node to goal node
C. The value corresponds to initial node to the current node
D. None of the above

13. Justify the given statement.

“Iterative deepening algorithm is the combination of Breadth First Search and Depth First Search algorithms.”

A. Yes
B. No

14. Which of the existing node is eliminated when a new node comes for evaluation in SMA* Algorithm?
A. The node which was recently evaluated
B. The node which was evaluated first
C. The node with lowest f(n) value
D. The node with highest f(n) value

15. Which searching algorithm is applied level by level in an increasing order until goal node is found?
A. Best First Algorithm
B. Iterative Deepening
C. Simplified Memory Bounded A*
D. Bi-directional algorithm

Answers for SelfAssessment

1.	C	2.	C	3	B	4.	B	5.	A
6.	A	7.	A	8.	C	9.	A	10.	D
11.	B	12.	C	13.	A	14.	D	15.	B

Review Questions

- Explain the Iterative Deepening with example.
- What do you understand by the concept of Best First Search?
- Giveexamples to demonstrate the real timeapplicationfor bi-directional algorithm.
- Listthe algorithms under informed and uninformed search.
- Explain the difference between the A* and SMA*.

Further readings

- Elaine Rich, Kevin Knight and Shivashankar B Nair, “Artificial Intelligence”, Third Edition, Tata McGraw Hill Publishing Company Ltd., 2009.
- Stuart Russell and Peter Norvig, “Artificial Intelligence: A Modern Approach”, Global Edition, Pearson Prentice Hall, 2016.

Web Links

- https://www.javatpoint.com/search-algorithms-in-ai
- https://www.tutorialspoint.com/artificial_intelligence/artificial_intelligence_popular_search_algorithms.htm
- https://www.javatpoint.com/ai-uninformed-search-algorithms

Unit 05 : Game Playing

CONTENTS

Objectives

1. Understanding the concepts of game theory.
2. Understanding the difference between MinMax and NegaMax algorithms.
3. Understanding the searching in one player and two player scenarios.
4. Understanding the real word applicationsofgame theory.
5. Understanding how pruning helps searching algorithms.

Introduction

In this unit, we will study to solve the problems using gaming approach. Here, we talk about the multi agent scenarios to solve a particular problem. Games can be known for their well-defined rules. It is otherwise understood that games don't require much knowledge to remember. It is enough to follow the rules, legal moves and the conditions of winning or losing the game. Multi-agent can be like having more than one players involved in a game. We are familiar with two player games such as board games. Here, both players try to win the game. So, both of them try to make the best move possible at each turn.The theory derived from this gaming approach is called game theory. It is now used in searching algorithms. Now, this game theory is applied for selecting the best move or searching for best move at a given state. We are going to discuss about two popular algorithms. The very first algorithm isknown as MinMax algorithm and the second is called as NegaMax algorithm. These algorithms are used in Artificial Intelligence for searching for next best state from the current state. The algorithms will be discussed in detail with some examples. Also, as the tree structures / state space are involved, there is a need to remove the unwanted branches from the tree, which will be done by an algorithm called Alpha-Beta Pruning.

The working principles of how pruning is done, will be discussed step by step in detail.Let us discuss all these one by one.

Game Playing

Let us begin with few simple two-player games, i.e.,Hexapawn game, tic-tac-toe game and last coin game. It is important thing to know that the winning of the game lies in selecting the right move. As the game theory stands for having well defined rules, both the players know what are all the possible moves going to be at a given state from each other's move. So, choice of moves is fixed and known to both of the players before any play. But, only one player win and other have to lose the game. There are also some rules to define the winning or losing the game. Hence, there is no confusing element in game theory. The different parts of information that is very much important for any game is depicted in the figure 1. The first is players who is playing the games, second is the rules which are being followed during the play by the players, third is consequences, which will be the outcome of the current moves and the last is payoffs which tells us how to decide the win, lose or draw of the game. These four parts will be clearly known to both the players – which in short understood as well defined games.

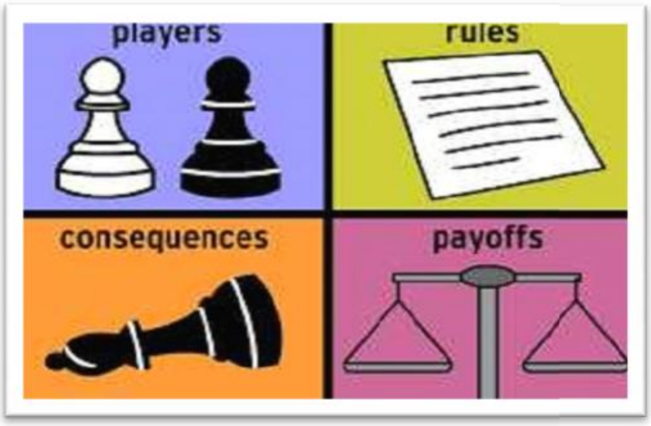


Figure 1 Parts of Information in Game theory

The tree structure is used to mention all the possible moves for both of them after every possible move that are played by the player. For example, let us assume there are two players as Player A and Player B. Player A is starting from Root. He is having only two choices initially. He selects any one from two choices. Now, Player B has to play. Player B checks the availableoptions from the current state and has to select anyone. Again, the Player A gets its turn. This goes repeatedly till some one win or lose or draw. The basic terminologies of the tree structure are well understood from the figure 2.

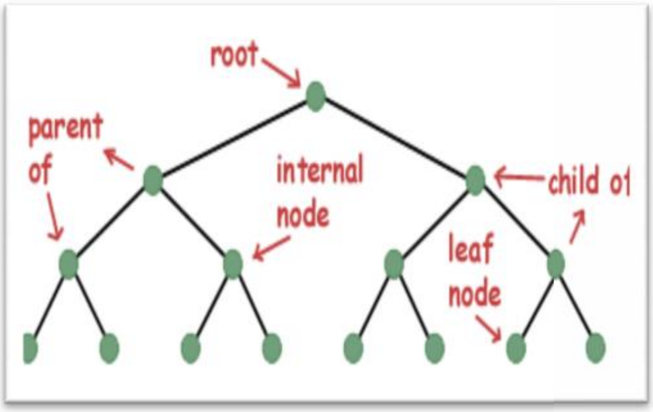


Figure 2Tree structure

The tree in the above figure is also called as binary tree because it is having just two children for each node. The terminologies discussed here, are root node, parent nodes, child nodes, internal nodes and leaf nodes. The nodes, which are available at the bottom line, are known as leaf nodes. The node exists at the top of the tree is known as root node. All the other nodes between root and leaf are called internal nodes. Nodes, which are having any branches of nodes, are called as parent node. The nodes in the branches are called child nodes of that parent node.

This tree structure is going to be used to mention all the possible moves according to the current state. The understanding of the moves and the nature of two player games, will be understood very easily with some simple examples such as board games. As we know, board games or any games is having well-defined rules and it will be known to all the players before the start of the game. Also, each player will know what are other options will go for the other player, in response to the current move. Hence, all the possible moves for each possible move can easily be represented by the tree structures for two player games. That is the reason, why we tried to understand better about tree structure before discussing the specific algorithms, which uses the trees.

5.1 Hexapawn Game

- Hexapawn is a deterministic two-player game invented by Martin Gardner.
- Played on a rectangular 3x3 board as shown in figure 3.
- The goal of each player is to advance one of their pawns to the opposite end of the board or to prevent the other player from moving.

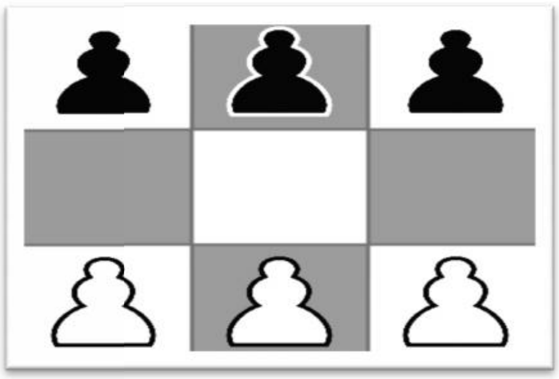


Figure 3 Hexapawn game

- Each pawn may be moved in two different ways.Itmay be moved one square forward, or it may capture a pawn one square diagonally ahead of it. A pawn may not be moved forward if there is a pawn in the next square.
- The first move of a pawn may not advance it by two spaces.
- A player loses if they have no legal moves or the other player reaches the end of the board with a pawn.

5.2 Tic-Tac-Toe Game

- This is also a two-player game.
- Tic-tac-toe board will be looking as shown in figure 4.
- Different players have their own coins. In this case, let us assume that Coin X is with Player A and Coin O is with Player B.
- The player who succeeds in placing three of their marks in a diagonal, horizontal, or vertical row is the winner. Here, from the figure, let us guess who has won the game. It is exactly the Player A because coin X is placed diagonally.
- Players will be getting one chance at a time; hence you can place only one coin at a time. Then, next turn will be given for next player. Subsequently the first player will place his / her second coin after other's done with their turn.

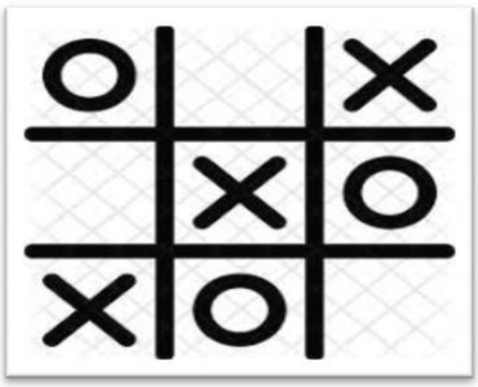


Figure 4 Tic-Tac-Toe board games

The tree structure will be look like the figure-5 depicted below. You can understand the how the nodes are defined for this kind of problems. Each node presents each state that is the each possible move that can be played by each player and vice versa. The root node is the empty board as usual becausethere are no coins present before the first player starts playing. Similarly, think how many different ways the first player put his / her first coin in the empty board / root node. Yes, nine will be right answer. There will be nine different ways for the first coin to be placed. Hence, the root node can have nine children nodes having all the possible moves. Now, let us think about the second player. Assume the first player has put his first coin as the given in the figure 5. This is now, the turn of the second player. Decide, how many different moves are possible if the first player already places the coin in the centre of the board.Those will be the children nodes for this parent node. Finish all the children nodes for this level of nodes. Repeat the steps for both the players alternatively until you reach leaf nodes or until no more moves are possible.

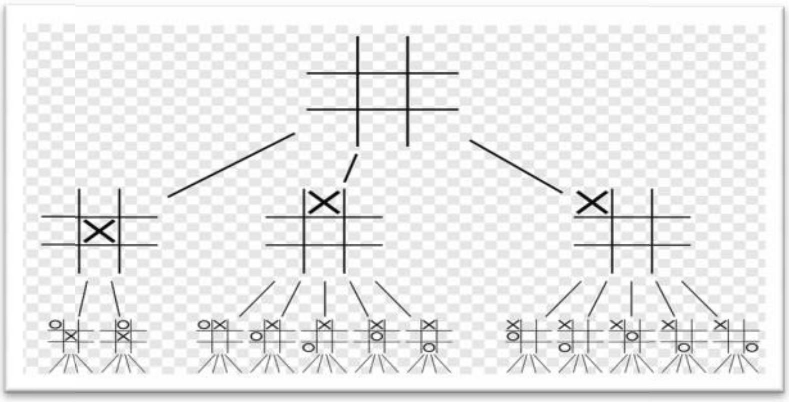


Figure 5 Sample state spaces for tic-tac-toe

5.3 <u>Last Coin Game</u>

- Coin game is a game in which each player picks coins from the given N coins in such a way that he can pick coins (ranging from 1 to 5 coins, for example) in one turn and the game continues for both the players.
- Limit the maximum number of coins that can be taken for each turn. Player can pick the coins within that specified limit.
- Both the players do their turn repeatedly until some one picks the last coin.
- The player who picks the last coin loses the game.

5.4 One Ply Search

- Ply represents the player. This is one player model.
- Nodes are mentioned as A, B, C and D.
- Node A is the root node and the remaining B, C and D are children of the node A.
- Player starts playing. There are three options. What is the selection criteria to be adopted?

Here, the player always wants to win the game. Hence, he chooses the option which will give him the highest winning probability. The figure 6 is giving one level state space to understand the process. Recall the idea of heuristics what we used for searching algorithms. Try to identify some approach / methods to measure the options. Let us assume, that the node B is having the value 8, node C is having the value 3 and node D is having the value -2. Selection will be in respect to our definition such as maximum likelihood of winning if the value is less or the maximum likelihood of winning if the value is high. It will go with our idea of describing the problem and the heuristics.

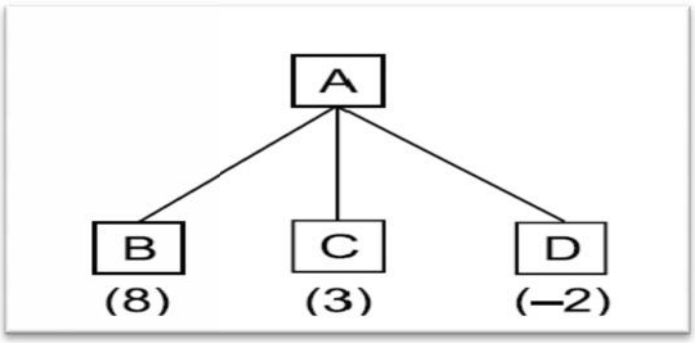


Figure 6 One-Ply Search

5.5 Two Ply Search

- Ply represents the player. This is two-player model.
- The first player is trying to win his game, so always he selects the moves, which will give him the highest probability to win the game.
- The second player is always trying to reduce the winning probability of the first player. So, the second player will select the moves, which will the highest probability to lose the game.
- Hence, first player will lose the game and automatically the second player will end up with win or draw.

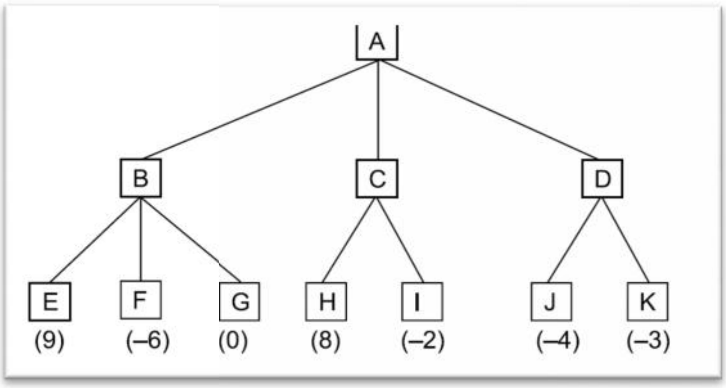


Figure 7 Two-Ply Search

Figure 7 helps in explaining the process of selecting the nodes. What will be choice of the first player who starts from node A? He wants to win. Okay, then the second player who plays from nodes B, C and D – will be interested to move towards lose, otherwise moves towards draw. Now, let us solve the above problem given in figure 7. Look at the node B, what will be selected? Definitelynode B will select node F because it is having low value -6. Then, look at the node C, selection will be the node I as it only -2, very low. Lastly, node D will select the node J. Subsequently, now node A is having options such as node B (-6), node C (-2) and node D (-4). Hence, the first player, for sure, will select the node C because it is having highest probability of winning than the other nodes. You can understand this way also, such as the decision is taken after analyzing all the moves of the opposition player, so that your winning probability is higher. The second player will do the same, but pulling you down, never allowing you to win by selecting the options which reduces the probability of your winning. This alternative way of selection is held in the two-ply search.

5.6 MinMax / MaxMin Algorithm

- This algorithm will be used to handle two player games.
- This algorithm may be called as MinMax or MaxMin algorithms, as there are two players namely Max and Min. If Max starts first, then we can refer it as MaxMin algorithm. Sometimes, if needed, we can start with Min and hence, we can refer it as MinMax algorithm.
- Max player wants to increase the possibility of winning. Min player wants to minimize the possible of winning. That's how they are called so.
- Perfect information is shared between the players because we are working on the well-defined games.
- The style of selecting the moves will be as same as what we discussed in the previous section 5.6 as given in the two-ply search scenario.
- The structure will be looking as in the Figure 8.

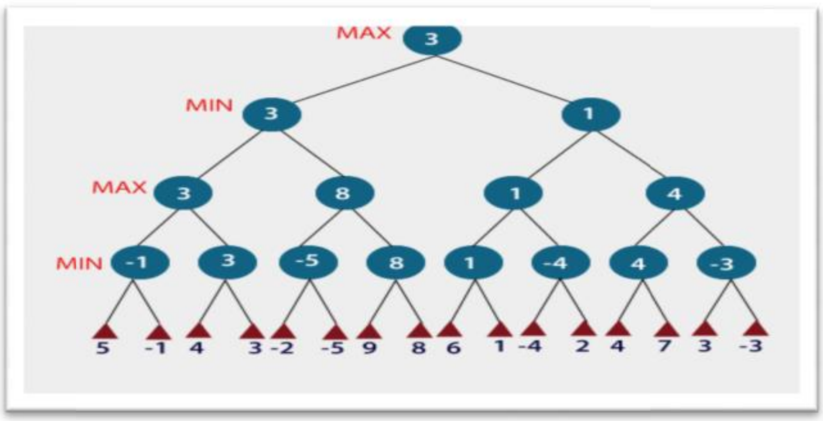


Figure 8 MaxMin Algorithms

I am leaving to you the process of selecting the nodes in favor of both of the players. Please refer the Two-ply search and try to solve the problem. The above problem is having four levels. Exactly speaking, player Max is playing two times and player Min is playing two times. On every turn, what Max will select and what min will select? Discuss and finalize the solution.

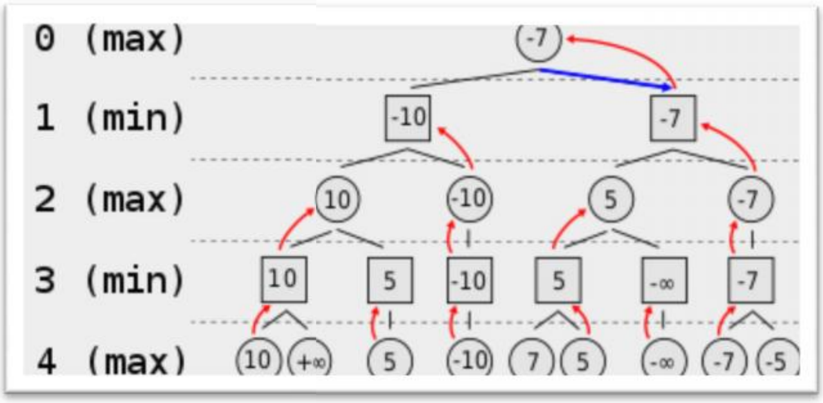


Figure 9 MaxMin Algorithms – An Example

Let us explore in detail how MaxMin Works using Figure 9. This is going to be step-step or turn after turn of the players. The selection of the nodes is pictorially represented in the figure given below. Player Max stars first and player Min then continues.Every player's turn will be deciding after processing the heuristics values given in the leaf nodes. Those values, how they are processed from leaf to the top are given clearly in the figure 9.

5.7 NegaMax Algorithm

Negamax search is a variant form of minimax search that relies on the zero-sum property of a two-player game. In simple words, NegaMax algorithm is focusing on simplifying the computations. This section describes how is it done? andwhat kind of mathematics involved in NegaMax. Why MaxMin is assumed that it is having more computations. All these will be explained in an easy way for better understanding.

MinMax algorithm performs two different computational methods such as Player A is playing for maximizing his win and at the same time Player B is playing for minimizing the win of Player A. Hence, two different approaches are happening which causes increasing in computations. Now, NegaMax is coming with a novel idea that reduces two approaches into one approach. Two players still can play using only one mathematical model, which reduces the computational time. The novel idea comes true with the zero-sum property. Let us understand what is zero-sum property before moving to mathematical model.

Zero-Sum Property

- Win for the First Player makes a Loss for the Second Player and vice versa.
- If the total gains of the participants are added up, and the total losses are subtracted, they will sum to zero.

The mathematical model

We know that the MinMax algorithm have two different functions such as Max and Min.Negamax simplifies the operation and does with comparatively less time. The mathematical model of NegaMax is given below as in the figure 10.

$$\max(a, b) = -\min(-a, -b)$$

Figure 10 NegaMax Function

In contrast to MinMax, NegaMax always selects maximum value out of possible values.When NegaMax selects any value then that value is multiplied by -1.This can be experienced with an example as given in figure 11.

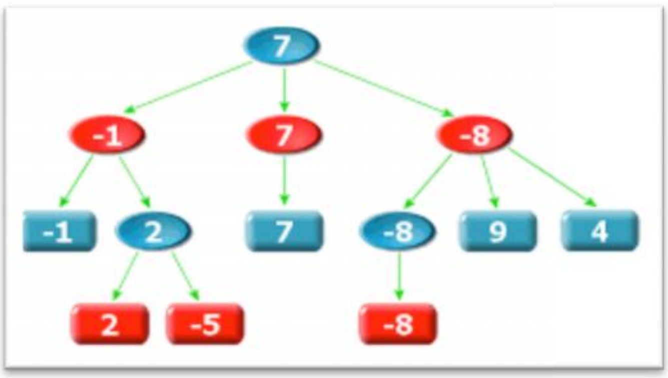


Figure 11 NegaMax Example

From the above mathematical model, one thing is very clear i.e., both the players are going to do the minimization function. If we recall our MAX and MIN players, it will be easy to make still clear on this. MAX player will do minimization function as given in figure 10. MIN player is also doing the minimization function as usual. Here, instead of having both maximization function and minimization functions, we are using only the minimization function for both the players.Concentrate on figure 11. The blue color shaded nodes are belonging to player MAX and the red colored nodes are belonging to player MIN.We are finding the minimum value as usual whenever player MIN turn comes. Level-1 tree is explaining this very well from how the data -1, 7, -8 is derived. The values -1 is derived from minimum (-1, 2), 7 is derived from minimum (7) and -8 is derived from minimum(-8, 9, 4). Let us go to Level-0 and play for player MAX. The values are -1, 7 and 8. Apply the mathematical model as in figure 10. The details of calculation are given below.

Max (-1, 7, 8) is equivalent to –Min (-(-1), -(7), -(8)). Do the calculation and let us see whether both the outputs are same or not.Max (-1, 7, 8) = 8. –Min (1, -7, -8) = -(-8) = 8. The minimization function is used to get the final output. All the selection of nodes for two player games will be done alternatively using the NegaMax algorithm.

5.8 Alpha Beta Pruning

The process of eliminating branchesfrom the tree is known as pruning. The alpha-beta pruning algorithm identifies the branches, which are not useful in searching,and then they are pruned.

Merits of Pruning

- To eliminate searching nodes that is potentially unreachable.
- To speedup the search process.

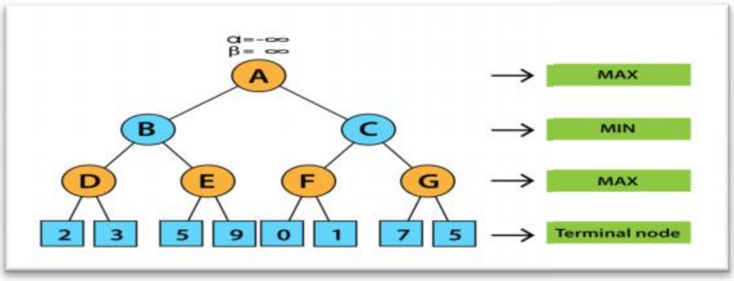
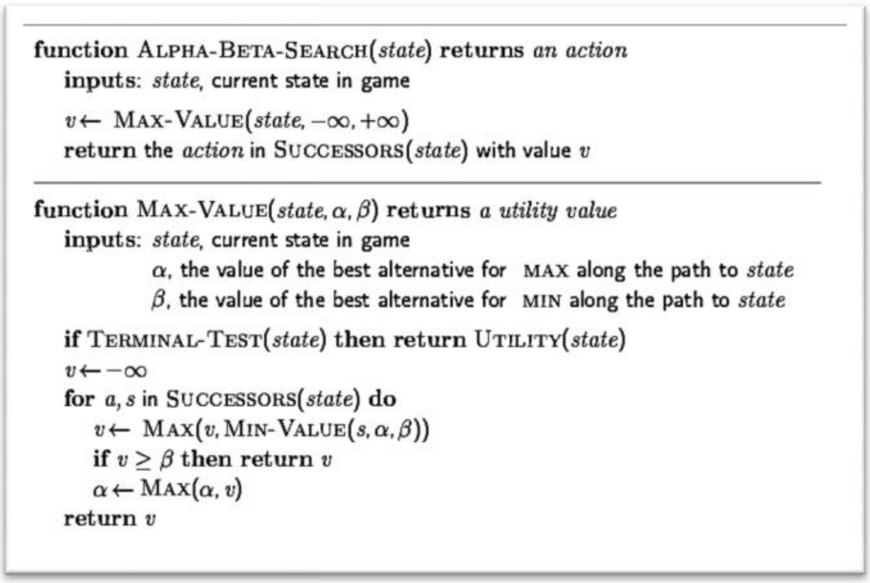


Figure 12 Alpha-Beta Pruning Example

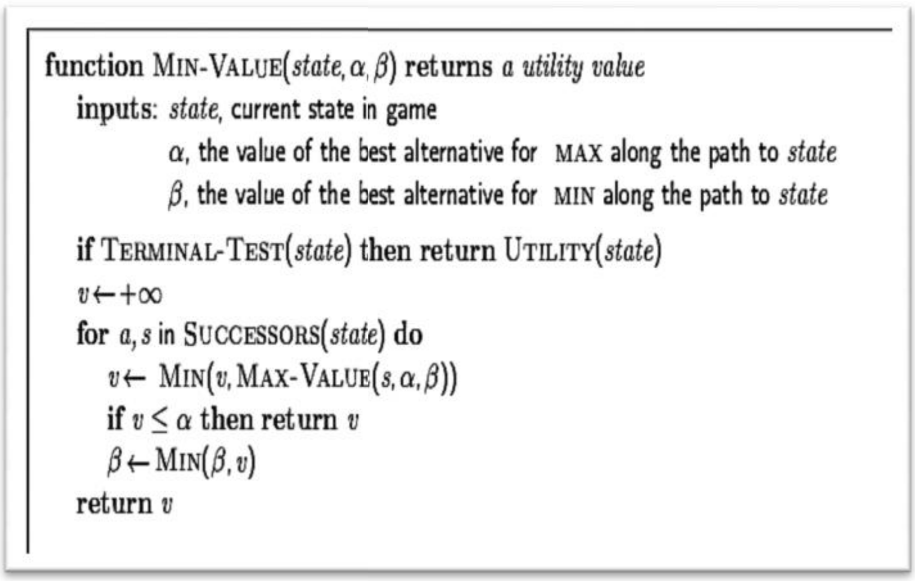
Let us understand the terminologies from the figure 12. The first term is Alpha. This refers to a lower bound on the value that a max node may ultimately be assigned.The next term is Beta.This refers to an upper bound on the value that a minimizing node may ultimately be assigned.These are symbolized as$\propto$ andβ. The initial values assigned are negative infinity and positive infinity for alpha and beta respectively. The process of alpha-beta pruning is explained in pseudo codes as given in the figures 13 and 14. Process in player MAX and the process in player MIN are explained separately. The same will also be explained with an example in the later part of this section.



```
function ALPHA-BETA-SEARCH(state) returns an action
    inputs: state, current state in game
    v ← MAX-VALUE(state, −∞, +∞)
    return the action in SUCCESSORS(state) with value v

function MAX-VALUE(state, α, β) returns a utility value
    inputs: state, current state in game
            α, the value of the best alternative for MAX along the path to state
            β, the value of the best alternative for MIN along the path to state
    if TERMINAL-TEST(state) then return UTILITY(state)
    v ← −∞
    for a, s in SUCCESSORS(state) do
        v ← MAX(v, MIN-VALUE(s, α, β))
        if v ≥ β then return v
        α ← MAX(α, v)
    return v
```

Figure 13 Pseudo Code of Alpha-Beta Pruning (MAX)



```
function MIN-VALUE(state, α, β) returns a utility value
    inputs: state, current state in game
            α, the value of the best alternative for MAX along the path to state
            β, the value of the best alternative for MIN along the path to state
    if TERMINAL-TEST(state) then return UTILITY(state)
    v ← +∞
    for a, s in SUCCESSORS(state) do
        v ← MIN(v, MAX-VALUE(s, α, β))
        if v ≤ α then return v
        β ← MIN(β, v)
    return v
```

Figure 14Pseudo Code of Alpha-Beta Pruning (MIN)

The initial values and partial steps are explained just to begin with in the figure 15. All the nodes are assigned with initial values of alpha and beta. Let us start with node D. Max player will adjust the value of symbol alpha. Min player will adjust the value of symbol beta. In the level-2, node D selects the maximum of 2 and 3. And, it assigns to alpha. It means that is the lower bound for alpha as it can take more than this in times to come. Then it passes to node B, as it is the Min player, the value of 3, becomes the upper bound for node B. because, it finds the minimum value. And hence, the value of 3, becomes the upper bound for its process. Now, coming to node E. Maximum of 5 and 9, is 9. Hold for a minute here. This is the point of deciding whether we need to process node E or we need to prune the node E. Node B is checking the minimum values. The upper bound already had as 3. If anything we are getting less than 3, it really gives meaning in proceeding further. But, in the current scenario, as we have the value 9 from node E, it is not going to change the decision of node B. So, can we prune the Node E completely? Yes, that's it. This has to be repeated for all the nodes till the end. The remaining portions of pruning are left to the reader to complete. You can refer the next step if you have any doubts in the process.

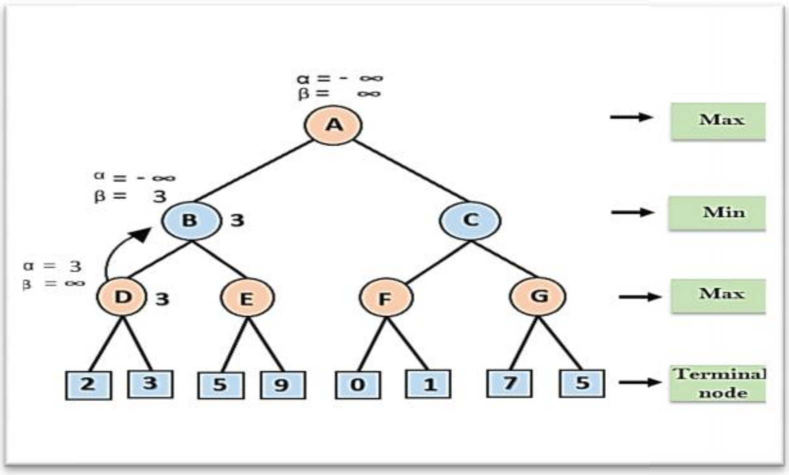


Figure 15Process in Alpha-Beta Pruning

5.9 Working with Alpha-Beta Pruning

We have another example, where we are going to explain all the processes completely step by step along with necessary figures.The tree to be pruned is given in the figure 16. It's a big structure having three levels and fourteen leaf nodes.

Step 1: Initially, the structure is looking like as in figure 16. Initialize all the nodes with alpha and beta values by positive infinity and negative infinity values. Assume, the player MAX starts first and the player MIN starts next. We repeat till some one wins / loses / draws.

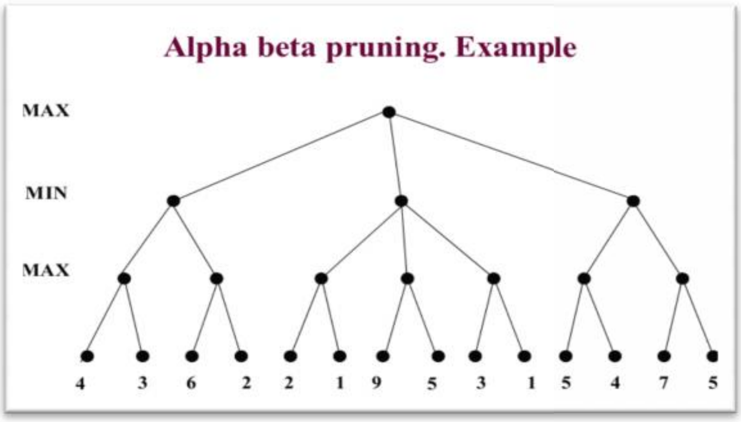


Figure 16 Step1 - Alpha-Beta Pruning

Step 2: Recall that alpha represents for MAX and beta represents for MIN.At the first, choosing the maximum value, i.e., 4. Hence, this value becomes the lower bound for the alpha. We write alpha >= 4.Any value more than 4, will replace alpha and becomes new lower bound for alpha. No change in beta value at the current node. This is given in figure 17.

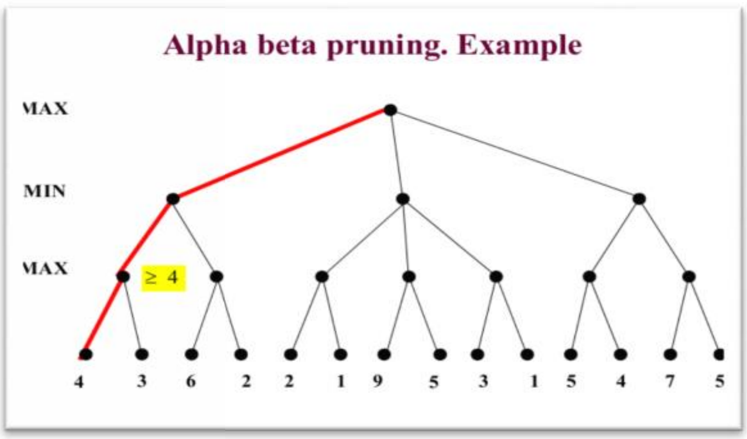


Figure 17 Step2 - Alpha-Beta Pruning

Step 3: Carrying towards its parent node. There, as given in figure 18, the value 4 becomes the upper bound for MIN. Hence, we write beta <= 4. Alpha remains no change.

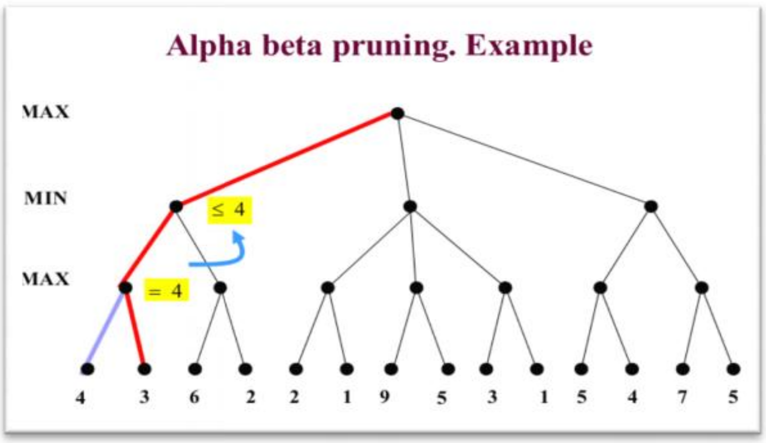


Figure 18 Step3 - Alpha-Beta Pruning

Step 4: Now, looking at another branch of the same parent node as in figure 19. Coming to the third and fourth leaf node, again the turn comes to player MAX. Maximum of 6 and 2 is 6. So, the lower bound of MAX is set as alpha >= 6. No change in Beta for the current node.

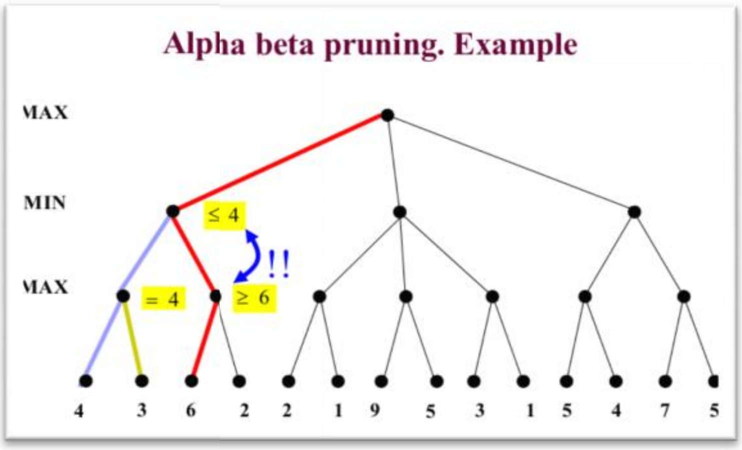


Figure 19 Step 4 - Alpha-Beta Pruning

Step 5: Now, it is the point to hold on a little bit. We are going to discuss the first pruning in our tree structure. At present, we working on one parent and two child nodes – having four leaf nodes.

Second child is having the value alpha >= 6, still you have not expanded the next leaf node(2). We need to decide, whether you need to expand or not. Go to the parent node. Beta <= 4. You are searching for minimum value. First child is having value 4 and the second child is having value 6 as per the present scenario. Minimum value is 4 as for as now. Even though, you expand the node (2), there will not be any change in the decision, meaning that, the minimum value 4 does not change. So, the node (2) is not needed and it can be pruned as shown in figure 20.

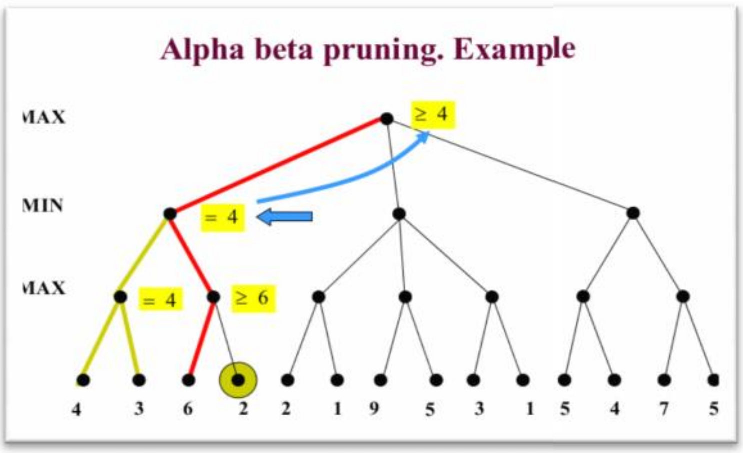


Figure 20 Step 5 - Alpha-Beta Pruning

Step 6: Now, after the step 5, after the pruning, the values are pushed to level-0 of the tree. That's the turn of Player MAX. We can write alpha >=4. Subsequently, we should process the next leaf nodes. So, we are reaching at fifth leaf node i.e., node 2 as in the figure 21. Finding the maximum of 2 and 1. The alpha becomes >= 2.

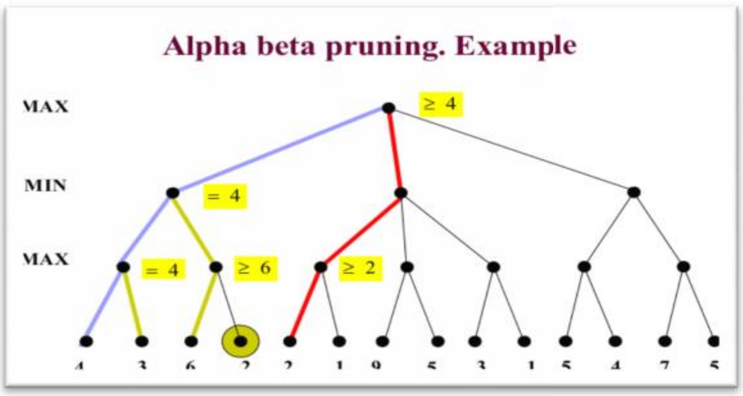


Figure 21 Step 6 - Alpha-Beta Pruning

Step 7: We are moving to the parent node now. As it is the turn of player MIN. The value of beta becomes <=2 as shown in figure 22.Now, have a break and hold on. We are going to have our next pruning. If we continue to process the remaining nodes under this node, we will be end up with searching still less than the value of 2. At any cost, there is no chance of having the value more than 2 from this node. Look at the root node, meaning that it is the parent of the current node. In the root, that is the turn of player MAX. It needs to find the maximum value from the available nodes. Already, it is having value 4. In the second child node, processing of remaining nodes (which are shaded in green color), does not change the maximum value of root node. So, we are pruning these nodes from the given tree.

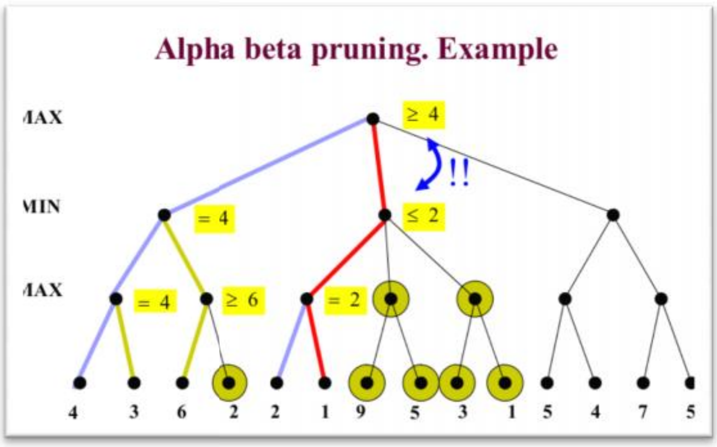


Figure 22 Step 7 - Alpha-Beta Pruning

Step 8: After pruning, we are going to next leaf nodes from the third child node as in the figure 23. The maximum value is 5. Hence, the alpha >= 5.

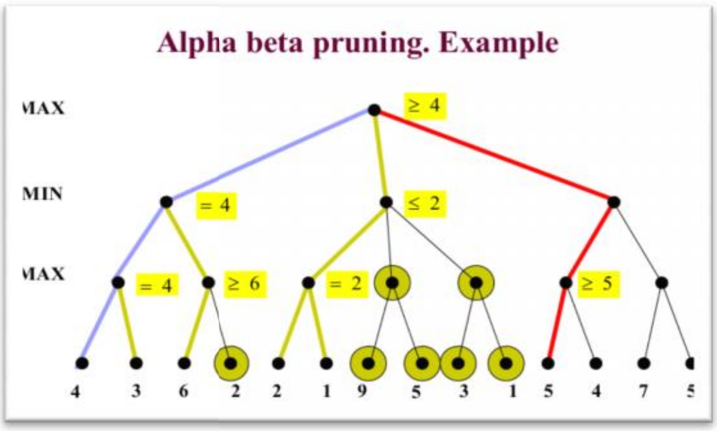


Figure 23 Step 8 - Alpha-Beta Pruning

Step 9:Next, the values of alpha are passed to parent node. Hence, the value of Beta becomes<= 5 as in figure 24.

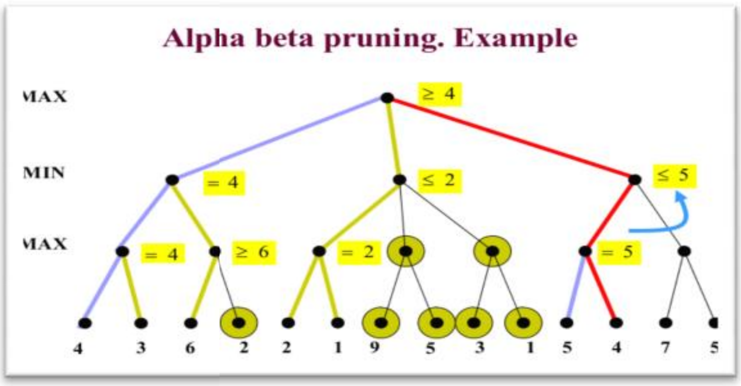


Figure 24 Step 9 - Alpha-Beta Pruning

Step 10: Going to the last leaf nodes, the need for the pruning arises. Check for the situation whether the expansion of node 5, affecting the value of the parent of parent node or not going to change the minimum value of its parent node. Last child node is having the value 7 at present, which is assumed to be maximum value from the last but one leaf node. Parent is checking for the minimum value, having already the value 5. Hence, the last leaf node, which is green-shaded, is not going to be needed for process. So, we are pruning it as in figure 25.

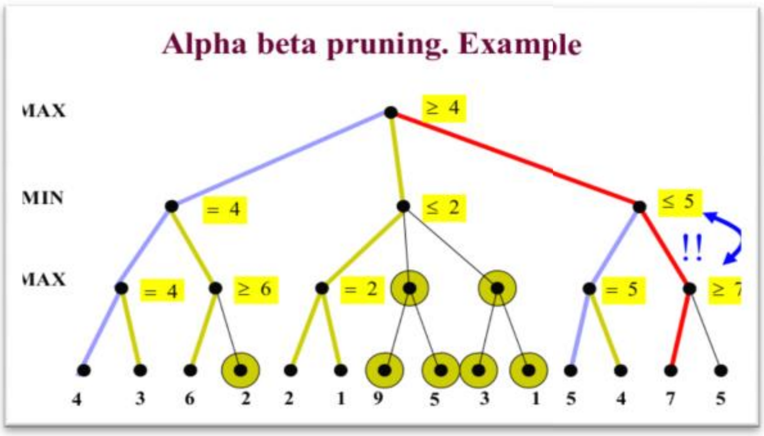


Figure 25 Step 10 - Alpha-Beta Pruning

Step 11: Now, the minimum value of 5 should be checked with the root node. It is finding the maximum value from 4, 2 and 5. Hence, the value is updated as 5 in the root node as in the figure 26.

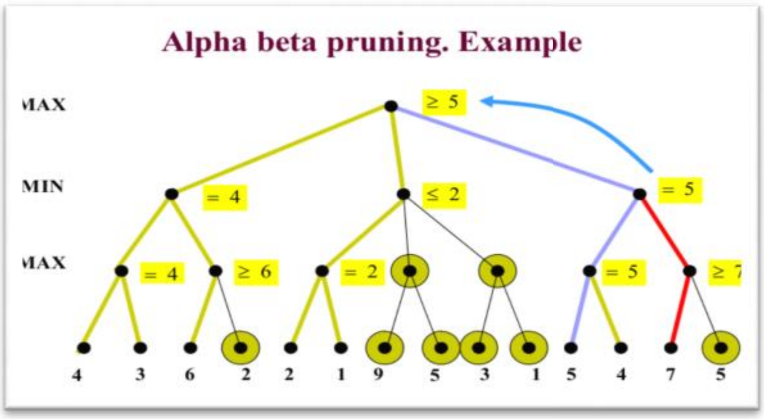


Figure 26 Step 11 - Alpha-Beta Pruning

Summary

- This unit discussed the concepts of game theory and their applications in the real world.
- Megamax algorithm is discussed in detail.
- Differences between the Megamax and Negamax algorithm was understood.
- All the algorithms were discussed with examples.
- The concept of pruning is explained and Alpha-Beta alpha pruning algorithm is illustrated with example.

Keywords

- MegaMax
- NegaMax
- Alpha-Beta Pruning
- Game Theory

Self Assessment

Q1) General games involves ____________

A. Single-agent
B. Multi-agent
C. Neither Single-agent nor Multi-agent
D. Only Single-agent and Multi-agent

Q2) Zero sum game has to be a ______ game.

A. Single player
B. Two player
C. Multiplayer
D. Three player

Q3) What is the complexity of minimax algorithm?

A. Same as of DFS
B. Space – bm and time – bm
C. Time – bm and space – bm
D. Same as BFS

Q4) To which depth does the alpha-beta pruning can be applied?

A. 10 states
B. 8 States
C. 6 States
D. Any depth

Q5) Which search is similar to minimax search?

A. Hill-climbing search
B. Depth-first search
C. Breadth-first search
D. All of the mentioned

Q6) How the effectiveness of the alpha-beta pruning gets increased?

A. Depends on the nodes
B. Depends on the order in which they are executed
C. All of the mentioned
D. None of the mentioned

Q7) Which values are independent in minimax search algorithm?

A. Pruned leaves x and y
B. Every states are dependent
C. Root is independent
D. None of the mentioned

Q8) Which value is assigned to alpha and beta in the alpha-beta pruning?

A. Alpha = max

B. Beta = min

C. Beta = max

D. Both Alpha = max & Beta = min

Q9) The initial value of alpha is ____________.

A. Negative Infinity

B. 0

C. Positive Infinity

D. 1

Q10) The initial value of beta is _______.

A. Negative Infinity

B. 0

C. Positive Infinity

D. -1

Q11) While backtracking the tree, the node values will be passed to upper nodes instead of values of alpha and beta.

A. TRUE

B. FALSE

C. Can be true or false

D. Can not say

Q12) The MIN player will only update the value of alpha.

A. TRUE

B. FALSE

C. Can be true or false

D. Can not say

Q13) The main condition that required for alpha-beta pruning is ______.

A. alpha<=beta

B. alpha>=beta

C. alpha=beta

D. alpha!=beta

Q14) Zero sum games are the one in which there are two agents whose actions must alternate and in which the utility values at the end of the game are always the same.

A. True

B. False

Q15) The one-ply and search represents the architecture for _______-.

A. Game with one player.

B. Game with two players.

C. Game with one player and two players both.

D. None of the above.

Answers for SelfAssessment

1.	D	2.	C	3	A	4.	D	5.	B
6.	A	7.	A	8.	D	9.	A	10.	C
11.	A	12.	B	13.	B	14.	B	15.	A

Review Questions

1. Differentiateone ply searchand two ply searchwith examples.
2. What do you understand by the concept of game theory?
3. Illustrate the process of pruning using alpha beta pruning algorithm in detail.
4. Explain the MegaMax algorithm with an example.
5. Describe the NegaMax algorithm with an example.

Further readings

- Elaine Rich, Kevin Knight and Shivashankar B Nair, "Artificial Intelligence", Third Edition, Tata McGraw Hill Publishing Company Ltd., 2009.
- Stuart Russell and Peter Norvig, "Artificial Intelligence: A Modern Approach", Global Edition, Pearson Prentice Hall, 2016.

Web Links

- https://www.geeksforgeeks.org/game-theory-in-ai/
- https://www.analyticsvidhya.com/blog/2019/11/game-theory-ai/
- https://www.youtube.com/watch?v=h0bdo06qNVw
- https://www.youtube.com/watch?v=RuWFxh9aRmc
- https://www.youtube.com/watch?v=0oqhN5tvLgA
- https://www.youtube.com/watch?v=a2tqR2eUlek

Unit 06 : Reasoning - Representation

CONTENTS

Objectives

1. Understanding the basic concepts of inference and inference systems.
2. Illustrating the propositional logic and rules of inferencing.
3. Understand the predicate logic and its method of solving the problems.
4. Differentiate the predicate and propositional logic.
5. Exploring forward and backward chaining with examples.

Introduction

In this unit, we will start discussing about what is inferencing and reasoning. Subsequently, the different techniques such as propositional logic and predicate logic will be introduced with few examples to show how inferencing works. Then, the kind of problem solving will be understood with respect to propositional logic and predicate logic separately. We will try to understand the differences between the propositional logic and predicate logic clearly in this section. Finally, Logical reasoning is discussed along with six types of reasoning such as deductive reasoning, inductive reasoning, abductive reasoning, common sense reasoning, monotonic reasoning and non-monotonic reasoning. Thereasoningusing forward chaining approach and backward chaining approach isalso discussed clearly.

6.1 Propositional Logic

This is a technique of knowledge representation in logical and mathematical form.A proposition is a declarative statement, which is either true or false.Propositional logic is the simplest form of logic where all the statements are made by propositions. Let us have two declarative statements for example,"It is hot" and "It is humid". We assume that the statements are true. Let us use P and Q to represent the given statements.Hence, P and Q are known as Propositional symbols.There are two types of Propositions. They are Atomic Propositions and Compound propositions, which are discussed below.

Atomic Proposition

Atomic propositions are the simple propositions. It consists of a single proposition symbol. These are the sentences, which must be either true or false. P, ¬P are atomic propositions, which means "It is hot" and "It is not hot".Negated symbol is representing the false statement.Hence, P can be true or false. Similarly, Q can be either true or false at a time.

Compound proposition

Compound propositions are constructed by combining simpler or atomic propositions, using parenthesis and logical connectives. For example, $(P\vee Q)$, $(P\wedge Q)$, $(P\rightarrow Q)$, and $(P\leftrightarrow Q)$ are compound propositions.

$(P\vee Q)$ means True if atleast any one statement is true, otherwise false. $(P\wedge Q)$ means True if both the statements are true, otherwise false. $(P\rightarrow Q)$ means Q is true if P is true or Q is false if P is false. $(P\rightarrow Q)$ means Q is true if and only if P is true, otherwise false.

In addition to the above, true or false are known as **Logical constants** and a proposition can be having Wrapping **parentheses**(…) also.

In the previous discussion, we are using few connectives to join the propositions.There are mainly five connectives, which are given as follows.They are Negation, Conjunction, Disjunction, Implication and Biconditional.

Negation

A sentence such as ¬P is called negation of P. A literal can be either Positive literal or negative literal.

Conjunction

A sentence which has ∧connective such as, $P\wedge Q$ is called a conjunctionwhere P and Q are the propositions. Let us assumeP= "Rohan is intelligent" and Q= "Rohan is hardworking". $P\wedge Q$ means Rohan is intelligent andhardworking.

Disjunction

A sentence which has ∨ connective, such as $P\vee Q$ is called disjunctionwhere P and Q are the propositions.Let us asume P = "Ritika is Doctor" and Q= "Ritika is Engineer". $P\vee Q$ means "Ritika is a doctor or Engineer".

Implication

A sentence such as $P\rightarrow Q$, is called an implication. Implications are also known as if-then rules. It is understood as Q becomes True if P is true, or Q becomes false if P is false. Let us have two propositions likeP= "It is raining"and Q= "Street is wet". Hence, if we write like $P\rightarrow Q$, it means "If it is raining, then the street is wet".

Biconditional

A sentence such as $P\leftrightarrow Q$ is a Biconditional sentence. This connective is known as "if and only if". $P\leftrightarrow Q$ can be read as P is true if and only if Q is true. Or, Q is true if and only if P is true. Let us assume P= "I am breathing", Q= "I am alive". $P\leftrightarrow Q$ means Q is true if P is true and also P is true if Q is true. Both the directions are considered. "If I am breathing, then I am alive" and "If I am alive then I am breathing" are valid propositions.

Let us have some more examples for better understanding. Assume P = "It is hot", Q = "It is humid" and R = "It is raining".

- $(P\wedge Q)\rightarrow R$ means "If it is hot and humid, then it is raining"
- $Q\rightarrow P$ means "If it is humid, then it is hot"
- Q means "It is humid."
- $(\neg P\vee\neg Q)\rightarrow R$ means "If it not hot or not humid then it is raining".

Well-formed formula

A sentence is also known as well formed formula. This can be defined as follows:

- A symbol is a sentence.
- If S is a sentence, then ¬S is a sentence.
- If S is a sentence, then (S) is a sentence.
- If S and T are sentences, then $(S \vee T)$, $(S \wedge T)$, $(S \rightarrow T)$, and $(S \leftrightarrow T)$ are sentences.

Inference

Inference is the process of deriving new sentences from old / given sentences.A **valid sentence** is true in all worlds under all interpretations.**Sound** inference derives true conclusions given true premises. **Complete** inference derives all true conclusions from a set of premises.It is also worth noted that the **Propositional logic** commits only to the existence of facts that may or may not be the case in the world being represented.

Let us try to solve a problem. Here, there are four statements as given below.We have to prove R is true.The given statements are also called premises.

P

$(P \wedge Q) \rightarrow R$

$(S \vee T) \rightarrow Q$

T

The above four statements are expanded as given below. These statements are individually true which are similar to the above statements.

$$
\begin{array}{ll}
P & (1) \\
\neg P \vee \neg Q \vee R & (2) \\
\neg S \vee Q & (3) \\
\neg T \vee Q & (4) \\
T & (5)
\end{array}
$$

In order to prove R is true, we are going to prove that ¬R is true. This is called Negated Assumption. If we can't prove, then we can say our assumption is wrong and hence R is true. Now, We have to prove that R is true as shown in the given figure 1.

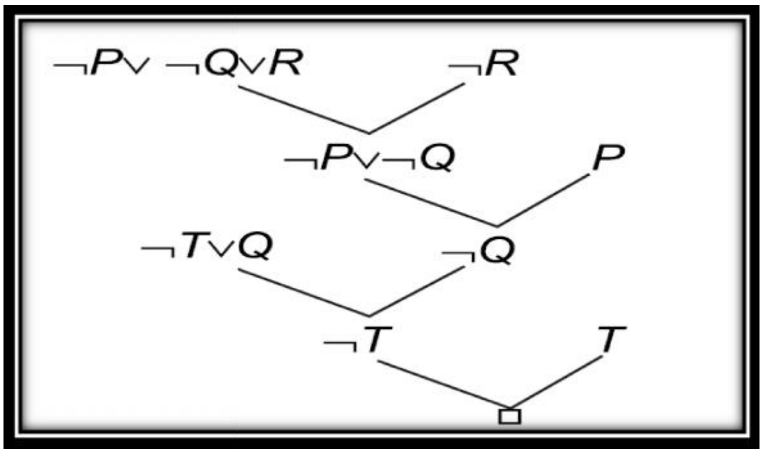


Figure 1 The proof using propositional logic

In the first matching of sentences, R and negated R cant be true at the same time. So, the remaining will be true for sure. That will be the implication of the first matching. Subsequent matching follows it. Every time, try to eliminate which are not supposed to be true. At the last, we have T and the negated T, which are not supposed to be true at the same time. This is known as contradiction. Hence, the final conclusion is the contradiction and hence it is false. This concludes that our assumption is proved as wrong / false. Hence, R is true.

Consider the following knowledge base.Premises are given.We should prove that "It will rain" is true.

1. The humidity is high or the sky is cloudy.
2. If the sky is cloudy then it will rain.
3. If the humidity is high then it is hot.
4. It is not hot.

Let us first denote the above clauses by the following symbols.

p = the humidity is high

q = the sky is cloudy

r = it will rain.

s = it is hot.

The propositions are formed for the given statements using the above mentioned symbols.

1 . $p \vee q$

2 . $\neg q \vee r$

3 . $\neg p \vee s$

4 . $\neg s$

Goal statement :¬ r

If we conclude with contradiction then ¬ r is false otherwise it is true. Proof is given below in Figure 2.

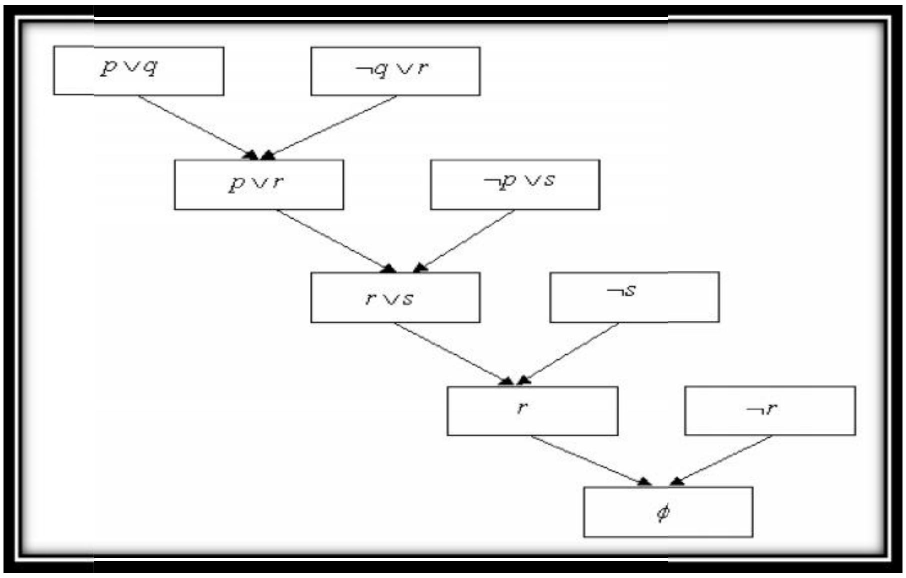


Figure 2 The proof using propositional logic

Now, we got the conclusion with a contradiction. Hence, ¬ r is false.

Limitations of Propositional logic:

- We cannot represent relations like ALL, some, or none with propositional logic.
- Example:All the girls are intelligent.Some apples are sweet.
- Propositional logic has limited expressive power.

- In propositional logic, we cannot describe statements in terms of their properties or logical relationships.

6.2 Predicate Logic

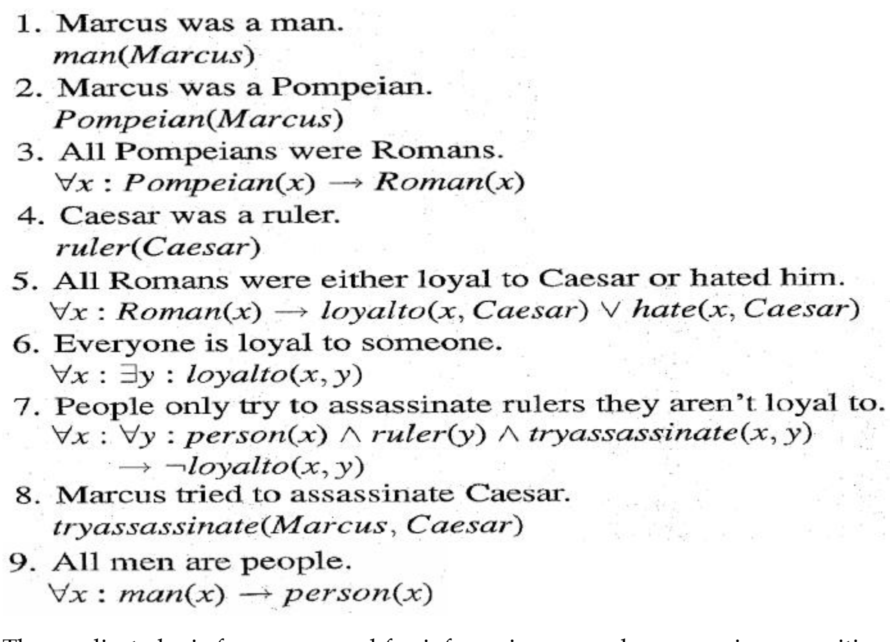
Predicate Logic deals with predicates, which are propositions containing variables.A predicate is an expression of one or more variables defined on some specific domain.A predicate with variables can be made a proposition by either assigning a value to the variable or by quantifying the variable.The following are some few examples of predicates.Let E(x, y) denote "x = y". Let X(a, b, c) denote "a + b + c = 0". Let M(x, y) denote "x is married to y" Propositions E, X and M may represent any statements like mentioned above. The variables are participating in the propositions, which is our interest in this section.

We have seen the disadvantages of propositional logic; hence we need to justify how predicate logic is going to overcome those demerits. Predicate logic is using two types of quantifiers. They are Universal Quantifier and Existential Quantifier.The variables of predicate are quantified by quantifiers. Universal quantifier states that the statements within its scope are true for every value of the specific variable. It is denoted by the symbol . Existential quantifier states that the statements within its scope are true for some values of the specific variable. It is denoted by the symbol .If we use a quantifier that appears within the scope of another quantifier, it is called nested quantifier.

Let us have an example like "All the dogs have tails". Here, let us assume the symbol D for "Dog" and T for "Tail".We can write the predicate logic as, x D(x) → T(x)which means that for all x, if x is dog then it x must have tails.

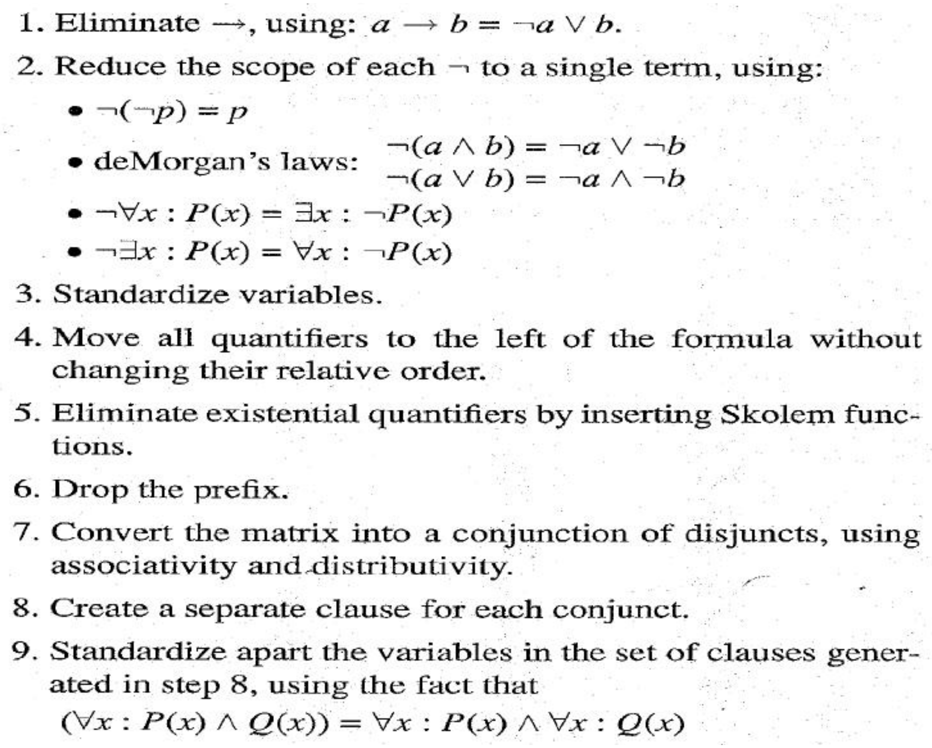
Try for another example like "Some dogs don't have tails". Here too, let us assume the symbol D for "Dog" and T for "Tail". We can write the predicate logic as, x D(x) →¬ T(x), which means that there exists x, if x is dog then it x don't have tails."There exists" refers for some dogs.

Try for few more examples. The statements and the predicates are given here for your better understanding.



1. Marcus was a man.
   $man(Marcus)$
2. Marcus was a Pompeian.
   $Pompeian(Marcus)$
3. All Pompeians were Romans.
   $\forall x : Pompeian(x) \rightarrow Roman(x)$
4. Caesar was a ruler.
   $ruler(Caesar)$
5. All Romans were either loyal to Caesar or hated him.
   $\forall x : Roman(x) \rightarrow loyalto(x, Caesar) \vee hate(x, Caesar)$
6. Everyone is loyal to someone.
   $\forall x : \exists y : loyalto(x, y)$
7. People only try to assassinate rulers they aren't loyal to.
   $\forall x : \forall y : person(x) \wedge ruler(y) \wedge tryassassinate(x, y)$
   $\rightarrow \neg loyalto(x, y)$
8. Marcus tried to assassinate Caesar.
   $tryassassinate(Marcus, Caesar)$
9. All men are people.
   $\forall x : man(x) \rightarrow person(x)$

The predicate logic forms are used for inferencing as we have seen in propositional logic. But it is difficult to use as it is given in original form. Hence, they have to be converted into clause forms. Because,clause forms are very much comfortable in inferencing / problem solving.The following steps should be used for conversion into clause form.



1. Eliminate $\rightarrow$, using: $a \rightarrow b = \neg a \vee b$.
2. Reduce the scope of each $\neg$ to a single term, using:
   - $\neg(\neg p) = p$
   - deMorgan's laws: $\neg(a \wedge b) = \neg a \vee \neg b$; $\neg(a \vee b) = \neg a \wedge \neg b$
   - $\neg \forall x : P(x) = \exists x : \neg P(x)$
   - $\neg \exists x : P(x) = \forall x : \neg P(x)$
3. Standardize variables.
4. Move all quantifiers to the left of the formula without changing their relative order.
5. Eliminate existential quantifiers by inserting Skolem functions.
6. Drop the prefix.
7. Convert the matrix into a conjunction of disjuncts, using associativity and distributivity.
8. Create a separate clause for each conjunct.
9. Standardize apart the variables in the set of clauses generated in step 8, using the fact that $(\forall x : P(x) \wedge Q(x)) = \forall x : P(x) \wedge \forall x : Q(x)$

6.3 Logical Reasoning

Reasoning is the mental process of deriving logical conclusions and making predictions from available knowledge, facts, and beliefs. It is a general process of thinking rationally, to find valid conclusions. Or we can say, "Reasoning is a way to infer facts from existing data." In artificial intelligence, reasoning is essential so that the machine can also think rationally as a human brain, and can perform like a human. There are various types of Reasoning as given below.

Deductive reasoning

Deductive reasoning is deducing new information from logically related known information. It is the form of valid reasoning, which means the argument's conclusion must be true when the premises are true.Deductive reasoning is a type of propositional logic in AI, and it requires various rules and facts. It is sometimes referred to as top-down reasoning, and contradictory to inductive reasoning.

In deductive reasoning, the truth of the premises guarantees the truth of the conclusion.Deductive reasoning mostly starts from the general premises to the specific conclusion, which can be explained as below example.

Example:

Premise 1: All the human eats vegetables.

Premise 2: Suresh is human.

Conclusion:Suresh eats vegetables.

Inductive reasoning

Inductive reasoning is a form of reasoning to arrive at a conclusion using limited sets of facts by the process of generalization. It starts with the series of specific facts or data and reaches to a general statement or conclusion.Inductive reasoning is a type of propositional logic, which is also known as cause-effect reasoning or bottom-up reasoning.In inductive reasoning, we use historical data or various premises to generate a generic rule, for which premises support the conclusion.

In inductive reasoning, premises provide probable supports to the conclusion, so the truth of premises does not guarantee the truth of the conclusion.

Example:

Premise	: All of the pigeons that we have seen in the zoo are white.
Conclusion	: Therefore, we can expect all the pigeons to be white.

Abductive reasoning

Abductive reasoning is a form of logical reasoning, which starts with single or multiple observations then seeks to find the most likely explanation or conclusion for the observation. Abductive reasoning is an extension of deductive reasoning, but in abductive reasoning, the premises do not guarantee the conclusion.

Example:

Implication	: Cricket ground is wet if it is raining
Axiom	: Cricket ground is wet.
Conclusion	: It is raining.

Common Sense Reasoning

Common sense reasoning is an informal form of reasoning, which can be gained through experiences. Common Sense reasoning simulates the human ability to make presumptions about events, which occurs on every day. It relies on good judgment rather than exact logic and operates on heuristic knowledge and heuristic rules.

Example:

Statement 1	: One person can be at any one place at a time.
Statement 2	: If I put my hand in a fire, then it will burn.

The above two statements are the examples of common sense reasoning which a human mind can easily understand and assume.

Monotonic Reasoning

In monotonic reasoning, once the conclusion is taken, then it will remain the same even if we add some other information to existing information in our knowledge base. In monotonic reasoning, adding knowledge does not decrease the set of prepositions that can be derived.To solve monotonic problems, we can derive the valid conclusion from the available facts only, and it will not be affected by new facts.

Monotonic reasoning is not useful for the real-time systems, as in real time; facts get changed, so we cannot use monotonic reasoning.Monotonic reasoning is used in conventional reasoning systems, and a logic-based system is monotonic."Theorem proving" is an example of monotonic reasoning.

Also, consider the statements "Earth revolves around the Sun", "The moon revolves around the earth", "Earth is not round," and etc.They are true facts, and they cannot be changed even if we add more sentences in the knowledge base.

Non-Monotonic Reasoning

Non-monotonic reasoning deals with incomplete and uncertain models.In Non-monotonic reasoning, some conclusions may be invalidated if we add some more information to our knowledge base.Human perceptions for various things in daily life - is a general example of non-monotonic reasoning. Let suppose the knowledge base contains the following knowledge.

Example:

Statement 1 : Birds can fly.

Statement 2 : Penguins cannot fly.

Statement 3 : Pitty is a bird.

So, from the above sentences, we can conclude that Pitty can fly. However, if we add one another sentence into knowledge base statement 4: "Pitty is a penguin", which concludes "Pitty cannot fly", so it invalidates the above conclusion.

6.4 Forward Chaining

Forward chaining is the logical process of inferring unknown truths from known data and moving forward using determined conditions and rules to find a solution.

Generally, complex tasks can be reduced to multiple simpler tasks that are performed either simultaneously or in a sequence, like a chain. Chaining is an effective method for teaching complex skills or processes with multiple steps.As a data-driven and bottom-up form of logic, forward chaining starts from known conditions and rules, then progresses towards a logical conclusion using *if-then* statements. It applies these conditions and rules to the problem until there are no furtherapplicable situations or until a set limit is reached.

Forward chaining searches for any available conclusions and can create an infinite number of possible conclusions.In artificial intelligence (AI), forward chaining is used to help an AI agent to solve logic problems by inspecting rules and previous learning to deduce ways to find solutions. Forward chaining is used to break down the logic sequence and work through it from beginning to end by attaching each step after the previous one is solved.

6.5 Backward Chaining

The opposite of forward chaining is backward chaining.Backward chaining methodology can be described as working back from a goal.Backward chaining moves backward from a conclusion to find the rules or conditions from which it resulted.Backward chaining is an inference method widely used in artificial intelligence, automated theorem provers and proof assistants. Many programming languages support backward chaining within their inference engines.Backward chaining is also referred to as backward reasoning.Forward chaining and its counterpart backward chaining represent deductive logic.

Summary

In this section, we have discussed what is inferencing and reasoning. The different techniques such as propositional logic and predicate logic are introduced with examples. The kind of problem solving was discussed with propositional logic and predicate logic separately. The differences between the propositional logic and predicate logic were clearly shown in this section. Logical reasoning is discussed along with six types of reasoning such asdeductive reasoning, inductive reasoning, abductive reasoning, common sense reasoning, monotonic reasoning and non-monotonic reasoning. Lastly, the forward chaining and backward chaining approaches were discussed.

Keywords

- Inference
- Propositional Logic
- Predicate Logic
- Forward Chaining
- Backward Chaining.

Self Assessment

Q1) What is the another name of Single inference rule?

A Reference

B Resolution

C Reform

D None of these

Q2) How many logical connectives are there in artificial intelligence?

A 2

B 3

C 4

D 5

Q3) Which is used to compute the truth of any sentence?

A Semantics of propositional logic

B Alpha-beta pruning

C First-order logic

D None of the above

Q4)Translate the following statement into FOL.

"For every x, if x is a philosopher, then x is a scholar".

A x philosopher(x) ➔ scholar(x)

B x philosopher(x) ➔ scholar(x)

C x philosopher(x) ⬅➔ scholar(x)

D None of the mentioned

Q5)Translate the given statements into logical expressions of predicates, quantifiers and logical connectives.

Consider the universe of discourse for the variable x is as all the students.

A(x) : x likes spicy food.

Translate the statement "Some students don't like spicy food".

A ¬ A(x)

B x ¬ A(x)

C x¬ A(x)

D None ofthe above

Q6) What kind of clauses are available in Conjunctive Normal Form?

A Disjunction of literals

B Disjunction of variables

C Conjunction of literals

D Conjunction of variables

Q7) What is the equivalent statement for the statement given below.

$\neg (A \vee B)$

A $\neg A \vee \neg B$

B $\neg A \wedge \neg B$

C A ➔ B

D None of the above

Q8)Consider the given propositions ; P : It is hot, Q : It is humid, R : It is raining.

What $(P \wedge Q) \rightarrow R$represents ?

A It is raining when it is hot and humid.

B It is raining when it is hot or humid.

C It is hot and humid, so it is raining.

D If it is hot and humid, then it is raining.

Q9)Which is the correct statement for the following question.

Consider the universe of discourse for the variable x is as all the students.

A(x) : x is taking Artificial-Intelligence-Course.

B(x) : x is a CSE Student.

What does " x(A(x)∧B(x))"means ?

A All the students are CSE student but few are taking Artificial-Intelligent-Course.

B Whoever taking Artificial-Intelligence-Course, they are not CSE student.

C All the students are CSE student and taking Artificial-Intelligent-Course.

D None of the above

Q10) Which is a refutation complete inference procedure for propositional logic?

A Clauses

B Variables

C Proposition

D Propositional resolution

Q11) Which of the following are a deductive type of inference rule ?

A Forward chaining

B Backward chaining

C Both options A and B

D None of the above

Q12) From which rule does the modus ponens are derived?

A Inference rule

B Module rule

C Both Inference & Module rule

D None of the mentioned

Q13) Forward chaining is called as ____________ inference technique.

A Path-driven
B Goal-driven
C Data-driven
D None of the above

Q14) Which algorithm are in more similar to forward chaining algorithm?

A Breadth-first search algorithm
B Depth-first search algorithm
C Hill-climbing search algorithm
D All of the mentioned

Q15) Which algorithm are in more similar to backward chaining algorithm?

E Breadth-first search algorithm
F Depth-first search algorithm
G Hill-climbing search algorithm
H All of the mentioned

Answers for Self Assessment

1.	B	2.	D	3.	A	4.	B	5.	C
6.	A	7.	B	8.	D	9.	C	10.	D
11.	C	12.	A	13.	C	14.	A	15.	B

Review Questions

1. Explain the basic concept of inference with an example.
2. Write down the rules of propositional logic.
3. Give the procedure to solve a problem using Forward Chaining.
4. Differentiate propositional logic and predicate logic.
5. Solve a problem using backward chaining. Assume your own example.

Further Readings

- Elaine Rich, Kevin Knight and Shivashankar B Nair, "Artificial Intelligence", Third Edition, Tata McGraw Hill Publishing Company Ltd., 2009.
- Stuart Russell and Peter Norvig, "Artificial Intelligence: A Modern Approach", Global Edition, Pearson Prentice Hall, 2016.

.

Web Links

- https://tutorialforbeginner.com/rules-of-inference-in-ai
- https://www.bartleby.com/subject/engineering/computer-

science/concepts/inference-in-ai

- https://www.educba.com/propositional-logic-in-ai/
- https://www.javatpoint.com/propositional-logic-in-artificial-intelligence\
- https://www.geeksforgeeks.org/difference-between-propositional-logic-and-predicate-logic/
- https://www.geeksforgeeks.org/types-of-reasoning-in-artificial-intelligence/
- https://www.geeksforgeeks.org/difference-between-backward-and-forward-chaining/

Unit 07: AI languages and Tools

CONTENTS

Objectives

- Understanding the basics of lisp programming language.
- Understanding the prolog programming language.
- Understanding the resolution with simple examples.
- Understanding the Conjunctive Normal Form and Clausal Form.
- Understanding the steps to convert into clausal form.

Introduction

In this unit, the basic and popular programming languages of Artificial Intelligence are given a focus. They are Lisp and Prolog. Lisp is an acronym for list processing. Lisp is a functional programming language that was designed for easy manipulation of data strings. Lisp offers several different dialects and has influenced the development of other languages. Also it one of the oldest programming languages still in use.Similarly, Prolog is a logic programming language. It has an important role in artificial intelligence. Unlike many other programming languages, Prolog is intended primarily as a declarative programming language. In prolog, logic is expressed as relations (called as Facts and Rules). Core heart of prolog lies at the logic being applied. Also the concepts of Clausal Form i.e., Conjunctive Normal Form is discussed here along with the conversion steps from other forms. The fundamentals of resolution arehighlighted with few examples.

7.1 Lisp

Lisp is the second-oldest high-level programming language in the world, which is invented by John McCarthy in 1958 at the Massachusetts Institute of Technology. This programming language has an overall style that is organized around expressions and functions. Every procedure used in Lisp Programming is a function. It returns a data object as its value when it is called. These programs are made up of three basic building blocks i.e., atom, list and string.

Atom

An atom is a number or string of contiguous characters. It includes numbers and special characters. Following are examples of some valid atoms used in Lisp program.

hello-from-online-program

name

123008907

hello

Block#221

abc123

List

A list is a sequence of atoms and/or other lists enclosed in parentheses. Following are examples of some valid lists as given below.

(i am a list)

(a (a b c) d e fgh)

(father tom (susan bill joe))

(sun montue wed thurfri sat)

( )

String

A string is a group of characters enclosed in double quotation marks. The semicolon symbol (;) is used for indicating a comment line. Following are examples of some valid strings.

" *I am a string*"

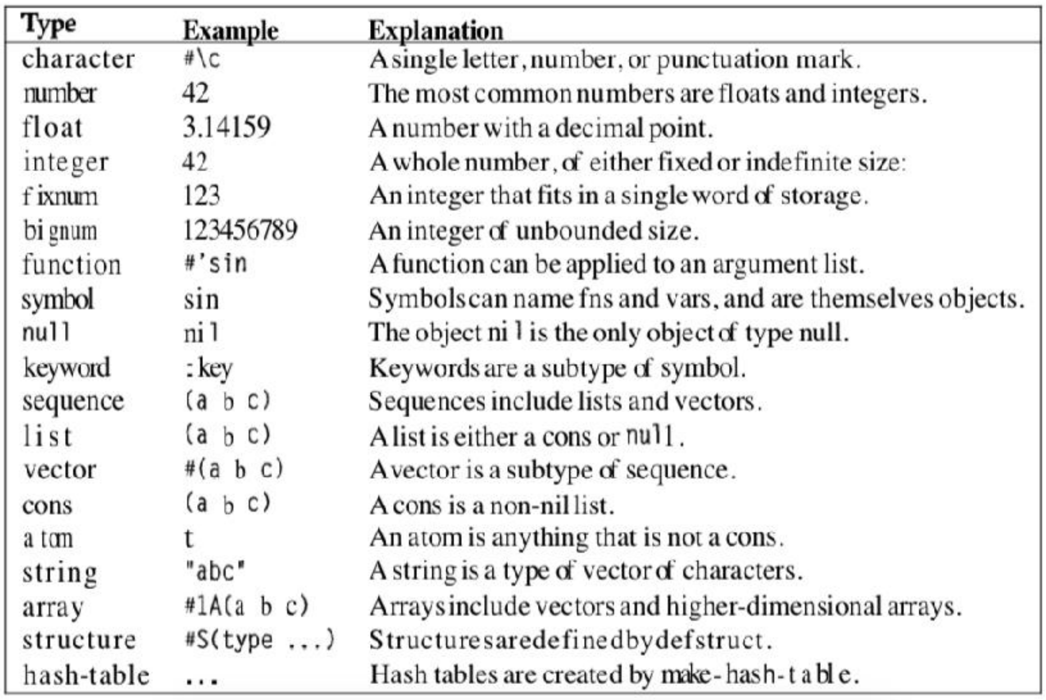
"*a ba c d efg #$%^&!*"

"*Please enter the following details :*"

"*Hello from* '*Online Class*'*!* "

Data type

The following,as in Figure 1,are the data types exist or used in Lisp programming.



Type	Example	Explanation
character	#\c	A single letter, number, or punctuation mark.
number	42	The most common numbers are floats and integers.
float	3.14159	A number with a decimal point.
integer	42	A whole number, of either fixed or indefinite size:
fixnum	123	An integer that fits in a single word of storage.
bignum	123456789	An integer of unbounded size.
function	#'sin	A function can be applied to an argument list.
symbol	sin	Symbols can name fns and vars, and are themselves objects.
null	nil	The object nil is the only object of type null.
keyword	:key	Keywords are a subtype of symbol.
sequence	(a b c)	Sequences include lists and vectors.
list	(a b c)	A list is either a cons or null.
vector	#(a b c)	A vector is a subtype of sequence.
cons	(a b c)	A cons is a non-nil list.
atom	t	An atom is anything that is not a cons.
string	"abc"	A string is a type of vector of characters.
array	#1A(a b c)	Arrays include vectors and higher-dimensional arrays.
structure	#S(type ...)	Structures are defined by defstruct.
hash-table	...	Hash tables are created by make-hash-table.

Figure 1 Data types in Lisp

The basic skeleton which allows you to define a function, a DEFUN, looks like this:

(defun name (parameter*)

"Optional documentation string." body-form*)

Any symbol can be used as a function name. Usually function names contain only alphabetic characters and hyphens, but other characters are allowed and are used in certain naming conventions.A function's parameter list defines the variables that will be used to hold the arguments passed to the function when it's called. If the function takes no arguments, the list is empty, written as ().

The three most basic components of all Lisp programs are functions, variables and macros.If a string literal follows the parameter list, it's a documentation string that should describe the purpose of the function. When the function is defined, the documentation string will be associated with the name of the function and can later be obtained using the DOCUMENTATION function.Finally, the body of a DEFUN consists of any number of Lisp expressions. They will be evaluated in order when the function is called and the value of the last expression is returned as the value of the function.

```
(defun hello-world ()
   (print "hello, world!"))
(defunverbose-sum (x y)
"Sum any two numbers after printing a message."
   (format t "Summing ~d and ~d.~\%" x y)
   (+ x y))
```

Variables

Lisp variables are named places that can hold a value. However, in Lisp, variables aren't typed the way they are in languages such as Java or C++. That is, you don't need to declare the type of object that each variable can hold. Instead, a variable can hold values of any type and the values carry type information that can be used to check types at runtime. Thus, Lisp is dynamically typed–type errors are detected.

For instance, if you pass something other than a number to the "+" function, Lisp will signal a type error. On the other hand, Common Lisp is a strongly typed language in the sense that all type errors will be detected–there's no way to treat an object as an instance of a class that it's not.

One way to introduce new variables you've already used is to define function parameters. As you saw in the previous Section, when you define a function with DEFUN, the parameter list defines the variables that will hold the function's arguments when it's called. For example, this function defines three variables–x, y, and z–to hold its arguments.

(defunfoo (x y z) (+ x y z))

The key to understanding lists is to understand that they're largely an illusion built on top of objects that are instances of a more primitive data type.Those simpler objects are pairs of values called cons cells, after the function CONS used to create them.CONS takes two arguments and returns a new cons cell containing the two values.2 These values can be references to any kind of object. Unless the second value is NIL or another cons cell, a cons is printed as the two values in parentheses separated by a dot, a so-called dotted pair.

(cons 1 2) ==> (1 . 2)

The two values in a cons cell are called the CAR and the CDR after the names of the functions used to access them.

7.2 Prolog

Prolog is a logic programming language. It has important role in artificial intelligence. Unlike many other programming languages, Prolog is intended primarily as a declarative programming language. In prolog, logic is expressed as relations (called as Facts and Rules). Core heart of prolog lies at the logic being applied. Running a query over relations carries out the formulation or computation.Some of the basic elements of Prolog are:Term, Fact statement and Rule statements.

Prolog language basically has three different elements:

- Facts − The fact is predicate that is true, for example, if we say, "Tom is the son of Jack", then this is a fact.
- Rules − Rules are extinctions of facts that contain conditional clauses. To satisfy a rule these conditions should be met. For example, if we define a rule as −
- grandfather(X, Y) :- father(X, Z), parent(Z, Y)
- This implies that for X to be the grandfather of Y, Z should be a parent of Y and X should be father of Z.
- Questions − And to run a prolog program, we need some questions, and those questions can be answered by the given facts and rules.

A Prolog term is a constant, a variable, or a structure.A constant is either an atom or an integer.Atoms are the symbolic values of Prolog and are similar to their counterparts in Lisp Programming Language. In particular, an atom is either a string of letters, digits, and underscores that begins with a lowercase letter or a string of any printable ASCII characters delimited by apostrophes.Fact statements: A fact statement is simply a proposition that is assumed to be true.

Example:

female(khushi). , it states Khushi is a female.

mother(varsha, khushi). , it states Varsha is Mother of Khushi.

Rule statements state rules of implication between propositions. This can be understood by an example as given below. It states that if Varsh is the mother of Khushi, then Varsha is an ancestor of Khushi.

Example:

ancestor(varsha, khushi) :- mother(varsha, khushi).

7.3 Resolution

Resolution yields a complete inference algorithm when coupled with any complete search algorithm.Resolution basically works by using the principle of proof by contradiction i.e., to find the conclusion we should negate the conclusion. Resolution makes use of the inference rules and performs the deductive inference.

One can perform Resolution from a Knowledge Base, that is a collection of facts or one can even call it a database with all facts.Then the resolution rule is applied to the resulting clauses. Each clause that contains complementary literals is resolved to produce a two new clause,which can be added to the set of facts (if it is not already present).This process continues until one of the two things happen.There are no new clauses that can be added.

An application of the resolution rule derives the empty clause. An empty clause shows that the negation of the conclusion is a complete contradiction,hence the negation of the conclusion is invalid or false or the assertion is completely valid or true.

Steps for Resolution Refutation:

- Convert all the propositions of KB to clause form (S).
- Negate and convert it to clause form. Add it to S.
- Repeat until either a contradiction is found or no progress canbe made:

In propositional logic it is easy to determine that two literals can not both be true at the same time. Simply look for L and ~L . In predicate logic, this matching process is more complicated, since bindings of variables must be considered.

Inference is the process of deriving new sentences from old or given sentences.A **valid sentence** is true in all worlds under all interpretations. **Sound** inference derives true conclusions given true premises. **Complete** inference derives all true conclusions from a set of premises. It is also worth noted that the **Propositional logic** commits only to the existence of facts that may or may not be the case in the world being represented.

Let us try to solve a problem. Here, there are four statements as given below. We have to prove R is true. The given statements are also called premises.

$$P$$
$$(P \wedge Q) \rightarrow R$$
$$(S \vee T) \rightarrow Q$$
$$T$$

The above four statements are expanded as given below. These statements are individually true which are similar to the above statements.

$$P \qquad (1)$$
$$\neg P \vee \neg Q \vee R \qquad (2)$$
$$\neg S \vee Q \qquad (3)$$
$$\neg T \vee Q \qquad (4)$$
$$T \qquad (5)$$

In order to prove R is true, we are going to prove that ¬ R is true. This is called Negated Assumption. If we can't prove, then we can say our assumption is wrong and hence R is true. Now, We have to prove that R is true as shown in the given figure 2.

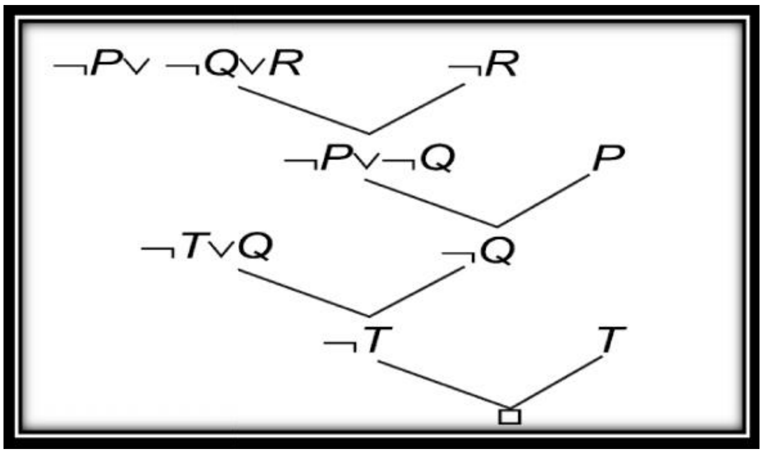


Figure 2 The proof using propositional logic

In the first matching of sentences, R and negated R cant be true at the same time. So, the remaining will be true for sure. That will be the implication of the first matching. Subsequent matching follows it. Every time, try to eliminate which are not supposed to be true. At the last, we have T and the negated T, which are not supposed to be true at the same time. This is known as contradiction. Hence, the final conclusion is the contradiction and hence it is false. This concludes that our assumption is proved as wrong / false. Hence, R is true.

Consider the following knowledge base.Premises are given. We should prove that "It will rain" is true.

1. The humidity is high or the sky is cloudy.
2. If the sky is cloudy then it will rain.
3. If the humidity is high then it is hot.
4. It is not hot.

Let us first denote the above clauses by the following symbols.

p = the humidity is high

q = the sky is cloudy

r = it will rain.

s = it is hot.

The propositions are formed for the given statements using the above mentioned symbols.

1. $p \vee q$
2. $\neg q \vee r$
3. $\neg p \vee s$
4. $\neg s$

Goal statement :¬ r

If we conclude with contradiction then ¬ r is false otherwise it is true. Proof is given below in Figure 3.

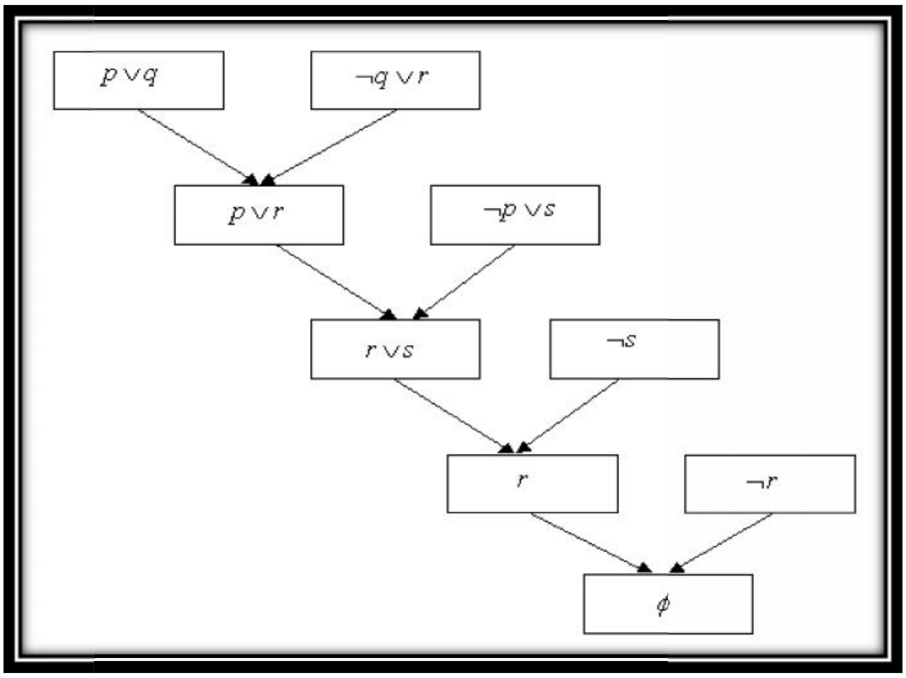


Figure 3 The proof using propositional logic

Now, we got the conclusion with a contradiction. Hence, ¬ r is false.

Unification

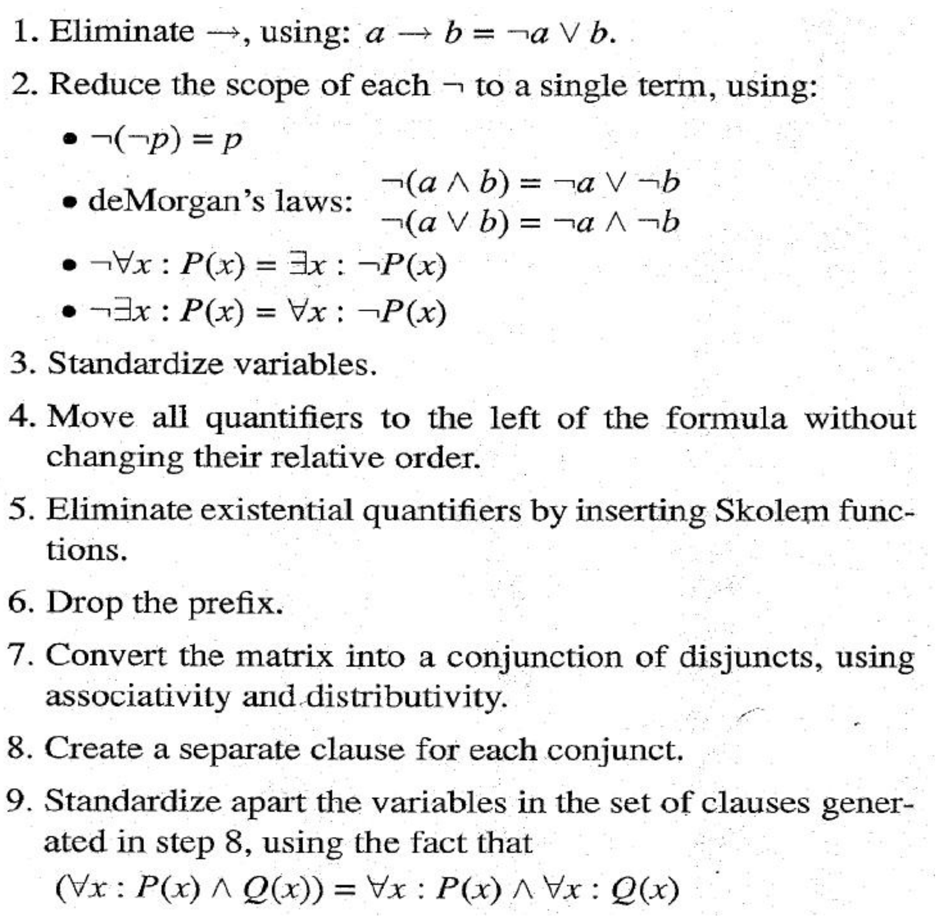
In order to determine contradictions, we need a matching procedure that compares two literals and discovers whether there exists a set of substitutions that makes them identical. For example man (john) and man(john) is a contradiction while man (john) and man(Himalayas) is not. There is a recursive procedure that does this matching. It is called Unification algorithm.

The Unification algorithm is listed below as a procedure UNIFY (L1, L2). It returns a list representing the composition of the substitutions that were performed during the match. An empty list NIL indicates that a match was found without any substitutions. If the list contains a single value F, it indicates that the unification procedure failed.

```
UNIFY (L1, L2)
        1. if L1 or L2 is an atom part of same thing do
                (a) if L1 or L2 are identical then return NIL
        (b) else if L1 is a variable then do
                (i) if L1 occurs in L2 then return F else return (L2/L1)
                        else if L2 is a variable then do
                (i) if L2 occurs in L1 then return F else return (L1/L2)
                        else return F.
    2. If length (L!) is not equal to length (L2) then return F.
    3. Set SUBST to NIL
        ( at the end of this procedure, SUBST will contain all the substitutions used to
        unify L1 and L2).
    4. For I = 1 to number of elements in L1 do
        i) call UNIFY with the ith element of L1 and I'th element of L2, putting the result
        in S
        ii) if S = F then return F
        iii) if S is not equal to NIL then do
                (A) apply S to the remainder of both L1 and L2
                (B) SUBST := APPEND (S, SUBST) return SUBST.
```

7.4 Clausal Form

Conjunctive Normal Form (CNF) is otherwise known as clausal form.CNF is conjunction of one or more clauses, where a clause is a disjunction of literals; otherwise this can be defined as CNF is a product of sums or an AND of ORs. The following as in Figure 4 are the steps used to convert the given form into clausal form.



1. Eliminate $\rightarrow$, using: $a \rightarrow b = \neg a \vee b$.
2. Reduce the scope of each $\neg$ to a single term, using:
   - $\neg(\neg p) = p$
   - deMorgan's laws: $\neg(a \wedge b) = \neg a \vee \neg b$; $\neg(a \vee b) = \neg a \wedge \neg b$
   - $\neg \forall x : P(x) = \exists x : \neg P(x)$
   - $\neg \exists x : P(x) = \forall x : \neg P(x)$
3. Standardize variables.
4. Move all quantifiers to the left of the formula without changing their relative order.
5. Eliminate existential quantifiers by inserting Skolem functions.
6. Drop the prefix.
7. Convert the matrix into a conjunction of disjuncts, using associativity and distributivity.
8. Create a separate clause for each conjunct.
9. Standardize apart the variables in the set of clauses generated in step 8, using the fact that
   $(\forall x : P(x) \wedge Q(x)) = \forall x : P(x) \wedge \forall x : Q(x)$

Figure 4 Steps to convert into clausal form

Let us have an example statement.

> "Suppose we know that all Romans who know Marcus either hate Caesar or thinkthat anyone who hates anyone is crazy."

This can be written in predicate logic as follows.

$$\begin{array}{l}\forall x : [Roman(x) \wedge know(x, Marcus)] \rightarrow \\ \quad [hate(x, Caesar) \vee (\forall y : \exists z : hate(y, z) \rightarrow \\ \quad thinkcrazy(x, y))]\end{array}$$

Now, let us proceed with the conversion steps and the resultant statements. After the very first step (**Eliminate $\rightarrow$)** the statement becomes,

$$\begin{array}{l}\vee x : \neg[Roman(x) \wedge know< x, Marcus)] \vee \\ \quad [hate(x, Caesar) \vee (\forall y : \neg(\exists z : hate(y, z)) \vee thinkcrazy(x,y))]\end{array}$$

Next, removing of negations, made the above statement as,

$$\begin{array}{l}\forall x : [\neg Roman(x) \vee \neg know(x, Marcus)] \vee \\ \quad [hate(x, Caesar) \vee (\forall y : \forall z : \neg hate(y, z) \vee thinkcrazy(x, y))]\end{array}$$

After the standardization and moving of quantifiers, made the above statement as,

$$\begin{array}{l}\forall x : \forall y : \forall z : [\neg Roman(x) \vee \neg know(x\ Marcus)] \vee \\ \quad [hate(x, Caesar) \vee (\neg hate(y, z) \vee thinkcrazy(x,y))]\end{array}$$

Eliminating existential quantifiers and moving the prefixes, made the above statement as,

$$\begin{array}{l}[\neg Roman(x) \vee \neg know(x, Marcus)] \vee \\ \quad [hate(x, Caesar) \vee (\neg hate(y, z) \vee thinkcrazy\{x, y))]\end{array}$$

Converting to conjunction of disjuncts, looks like as follows.

$$\begin{array}{l}\neg Roman(x) \vee \neg know(x, Marcus) \vee \\ \quad hate(x, Caesar) \vee \neg hate(y, z) \vee thinkcrazy(x, y)\end{array}$$

Summary

It is understood that, in this section, the popular programming languages of Artificial Intelligence were given a focus i.e., Lisp and Prolog. All the basic concepts and many statements or syntaxes have been discussed with respect to Lisp and Prolog. cAs you know now, Lisp offered several different dialects and had influenced the development of other languages.We know that Lisp was one of the oldest programming languages that still in use. Similarly, we studied other programming language called Prolog. It was a logic programming language. It was havingan important role in artificial intelligence. Unlike many other programming languages, Prolog was intended primarily as a declarative programming language. In prolog, logic was expressed as relations (called as Facts and Rules). Core heart of prolog lied at the logic being applied. Also the concepts of Clausal Form i.e., Conjunctive Normal Form was discussed here along with the conversion steps from other forms. The fundamentals of resolution werediscussed clearly with few examples.

Keywords

- Lisp
- Prolog
- Resolution

- Conjunctive Normal Form
- Clausal Form

Self Assessment

Q1) Which is used to represent the prompt in lisp?

A. #
B. $
C. &
D. *

Q2) Which notation facilitates uniformity in lisp?

A. Infix
B. Postfix
C. Prefix
D. None of the above

Q3) what is the use of '=' in prolog programming?

A. Unification
B. Arithmeticevaluation
C. Reduction
D. None of above

Q4) Convert the following into the clausal form.

$A(x) \rightarrow B(x)$

A. $A(x) \vee B(x)$
B. $\neg A(x) \vee B(x)$
C. $A(x) \vee \neg B(x)$
D. None of the above

Q5) What kind of clauses are available in conjunctive normal form ?

A. Conjunction of variables
B. Conjunction of literals
C. Disjunction of variables
D. Disjunction of literals

Q6) What is a Meta-program?

A. Meta-program is a program that uses other program as its data.
B. Meta-program is a program with metadata information.
C. Meta-program is program information.
D. None of above

Q7) What is the use of "halt" inbuilt predicated?

A. Usedto suspend the prolog system.
B. Usedto terminate the Prolog system.
C. Usedto resume the prolog system.
D. None of above

Q8) Which one of the following is not a variable?

A. X_yz
B. g_23A
C. '_Xyz'
D. B & C both

Q9)

Q10) A variable in Prolog must start with either an upper-case letter or an underscore (_).

A. True
B. False

Q11) Convert the following into clausal form.

For all x, Roman(x) →(loyal(x, Caesar) ∨hate（ x， Caesar)）

A. Roman (x) ∧¬loyal (x, Caesar) ∨hate（ x， Caesar)
B. Roman (x) ∨ loyal (x, Caesar) ∧¬hate（ x， Caesar)
C. ¬Roman (x) ∨ loyal (x, Caesar) ∨hate（ x， Caesar)
D. None of the above

Q12) What is the output of the following LISP program?

```
(setf chess '(king queen))
(write (cons 'rook chess))
```

A rook king queen
B king queen rook
C king queen
D rook

Q13) Let L = ['yellow', 'red', 'blue', 'green', 'black']. What does L[1:4] return?

A. ['red', 'blue', 'green']
B. ['blue', 'green', 'black']
C. ['yellow', 'red']
D. None of above

Q14) What are 2 and 3 referred as in the following LISP statement?

(+ 2 3)

A. List
B. Arguments

C. Both List and arguments
D. None of the above

Q15) What is the output of the given statement?

span class="sy0"> * (+ 3.14 2.71)

A. 3.14
B. 2.71
C. 5.84
D. 5.85

Answers for Self Assessment

1.	D	2.	C	3	A	4.	B	5.	D
6.	A	7.	B	8.	D	9.		10.	A
11.	C	12.	A	13.	A	14.	B	15.	D

Review Questions

1. Givethe basic programming fundamentals of lisp and prolog.
2. How the unification is performed?
3. Explain the concepts of resolution with an example.
4. Give the components of prolog programming language.
5. Discuss the steps used for converting into clausal form.
6. Compare the resolution with respect to predicate logic and propositional logic.

Further Readings

- Elaine Rich, Kevin Knight and Shivashankar B Nair, "Artificial Intelligence", Third Edition, Tata McGraw Hill Publishing Company Ltd., 2009.
- Stuart Russell and Peter Norvig, "Artificial Intelligence: A Modern Approach", Global Edition, Pearson Prentice Hall, 2016.

Web Links

- https://www2.cs.sfu.ca/CourseCentral/310/pwfong/Lisp/1/tutorial1.html
- https://common-lisp.net/downloads
- https://www.tutorialspoint.com/execute_lisp_online.php
- https://ai.vub.ac.be/sites/default/files/tutorial_1.pdf

Unit 08: Planning

CONTENTS

Objectives

1. Understanding the basic concepts of planning with examples.
2. Illustrating the various methodologies of planning.
3. Understanding the concepts of hierarchical planning.
4. Understanding the fundamentals of conditional planning.
5. Understanding blocks world planning problem.

Introduction

This unit discusses the planning and problem-solving process required in an advanced artificial intelligence. The term "planning" refers to determining a sequence of actions that are known to achieve a particular objective or task. The term "Problem solving" is finding a plan for anobjective / task. Artificial intelligence is an important technology in the future. Whether it is intelligent robots, self-driving cars, or smart cities, they will all use different aspects of artificial intelligence.Planning is necessary to find the best solutionat a given environment, but the tasks to be done at a particular time.A problem is difficult if an appropriate sequence of steps to solveit, is unknown. A plan is considered a sequence of actions, and each action has its preconditions that must be satisfied before it can act and some effects that can be positive or negative.We require domain description, task specification, and goal description for any planning system.Planning in artificial intelligence is about decision-making actions performed by robots or computer programs to achieve a specific goal. Execution of the plan is about choosing a sequence of tasks with a high probability of accomplishing a specific task. This unit also discusses few types of planning such as hierarchical planning and conditional planning along with blocks world problem, for example.

8.1 Basic Representation of Planning

Everything we humans do is with a definite goal in mind, and all our actions are oriented towards achieving our goal. Similarly, Planning is also done for Artificial Intelligence.As researchers grapple with various ways of extending planning systems to deal with specific aspects of real-world problems, the current work on planning is inevitably diverse. In conclusion, this unit discusses these works and summarizes various aspects of planning that are particularly relevant to cognitive science.That is why planning is considered as the logical side of acting. In other words, Planning is about deciding the tasks to be performed by the artificial intelligence system and the system's functioning under domain-independent conditions.Planning can be Classical planning or Non-classical planning. In the case of Classical Planning, the environment is fully observable, deterministic, static and discrete.Incase of Non-classical Planning, the environment is partially observable or non-deterministic or stochastic, i.e. the current state and chosen action cannot completely determine the next state of the environment.

Three kinds of Environments

Fully Observable	:	The agent always knows the current state.
Partially Observable	:	The agent knows only a certain amount about the actual state. (much more common in real world)
Unknown	:	The agent knows nothing about the current state

Planning researchers have settled on a factored representation, in which a state of the world is represented by a collection of variables. This use a language called PDDL, the Planning Domain Definition Language that allows the planner to express all 4T n^2 actions with one action schema. T represents time steps and n^2 represents current locations.We now show how PDDL describes the four things we need to define a search problem: the initial state, the actions that are available in a state, the result of applying an action, and the goal test.

For example, here is an action schema for flying a plane from one location to another.The schema consists of the action name, a list of all the variables used in the schema, a precondition and an effect.

Action (Fly (p, from , to),

PRECOND :$At(p, from) \wedge Plane(p) \wedge Airport(from) \wedge Airport(to)$

EFFECT : $\neg At(p, from) \wedge At(p, to)$)

As we discussed already, for any planning system, we need the domain description, action specification, and goal description. A plan is assumed to be a sequence of actions and each action has its own set of preconditions to be satisfied before performing the action and also some effects which can be positive or negative.

Algorithms for planning as state-space search

We discussed how the description of a planning problem defines a search problem. Let us now focus on the types of planning algorithms. They are forward (progression) state space search and backward (regression) relevant-state search.

Forward state space search

Given an initial state S in any domain, we perform some necessary actions and obtain a new state S' (which also contains some new terms), called a progression. It continues until we reach the target position. This goes in the forward direction hence it is called so.Consider the noble task of buying a copy of a book. Let us say X. Suppose there is an action schema Buy (isbn) with effect Own (isbn). ISBNs are 10 digits, so this action schema represents 10 billion ground actions. An uninformed forward-search algorithm would have to start enumerating these 10 billion actions to find one that leads to the goal. This shows that planning problems often here have large state spaces.

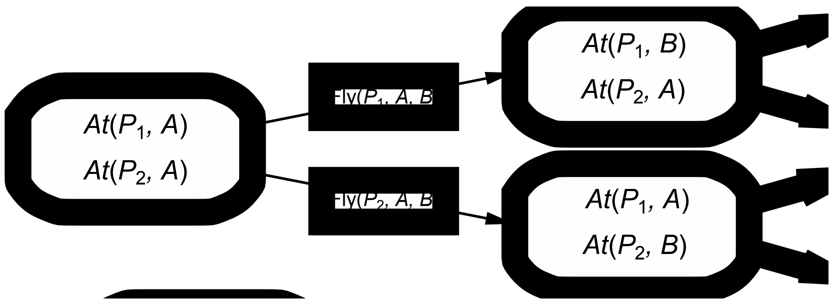


Figure Diagram for Forward Search

Backward relevant state search

In this search, we start at the goal state and apply the actions backward until we find a sequence of steps that reaches the initial state. It is called relevant-states search because we only consider actions that are relevant to the goal (or current state). In general, backward search works only when we know how to regress from a state description to the predecessor state description. The PDDL representation was designed to make it easy to regress actions.If a domain can be expressed in PDDL, then we can do regression search on it.Given a ground goal description g and a ground action a, the regression from g over a gives us a state description g′ defined by,

$$g' = (g - ADD(a)) \cup Precond(a).$$

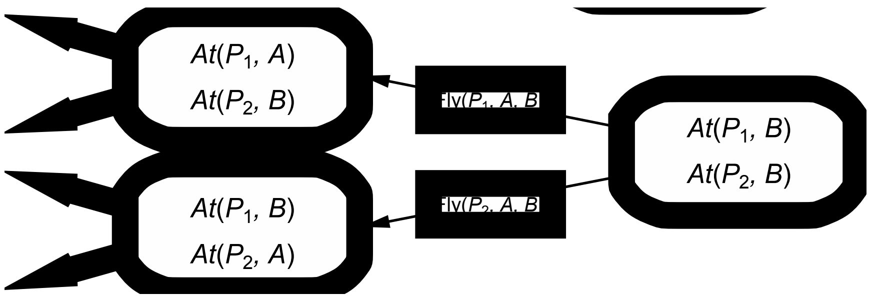
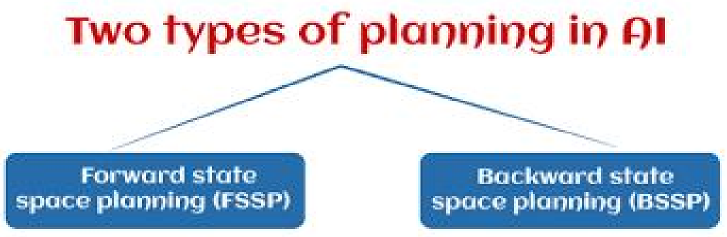


Figure Diagram for Backward Search

Backward search keeps the branching factor lower than forward search, for most problem domains. However, the fact that backward search uses state sets rather than individual states makes it harder to come up with good heuristics. That is the main reason why the majority of current systems favor forward search.


Two types of planning in AI
Forward state space planning (FSSP)
Backward state space planning (BSSP)

FigureTypes of planning

Forward State Space Planning (FSSP)

FSSP behaves in the same way as forwarding state-space search. It says that given an initial state S in any domain, we perform some necessary actions and obtain a new state S' (which also contains some new terms), called a progression. It continues until we reach the target position. Action should be taken in this matter.

Backward State Space Planning (BSSP)

BSSP behaves similarly to backward state-space search. In this, we move from the target state g to the sub-goal g, tracing the previous action to achieve that goal. This process is called regression (going back to the previous goal or sub-goal). These sub-goals should also be checked for consistency. The action should be relevant in this case.

8.2 Partial Order Planning

The idea of a partial-order planner is to have a partial ordering between actions and only commit to an ordering between actions when forced. This is sometimes also called a non-linear planner, which is a misnomer because such planners often produce a linear plan.

A partial ordering is a less-than relation that is transitive and asymmetric. A partial-order plan is a set of actions together with a partial ordering, representing a "before" relation on actions, such that any total ordering of the actions, consistent with the partial ordering, will solve the goal from the initial state. Write act0 < act1 if action act0 is before action act1 in the partial order. This means that action act0 must occur before action act1.

For uniformity, treat start as an action that achieves the relations that are true in the initial state, and treat finish as an action whose precondition is the goal to be solved. The pseudoaction start is before every other action, and finish is after every other action. The use of these as actions means that the algorithm does not require special cases for the initial situation and for the goals. When the preconditions of finish hold, the goal is solved.

An action, other than start or finish, will be in a partial-order plan to achieve a precondition of an action in the plan. Each precondition of an action in the plan is either true in the initial state, and so achieved by start, or there will be an action in the plan that achieves it.

We must ensure that the actions achieve the conditions they were assigned to achieve. Each precondition P of an action act1 in a plan will have an action act0 associated with it such that act0 achieves precondition P for act1. The triple act0,P,act1 is a causal link. The partial order specifies that action act0 occurs before action act1, which is written as act0 < act1. Any other action A that makes P false must either be before act0 or after act1.

Informally, a partial-order planner works as follows: Begin with the actions start and finish and the partial order start < finish. The planner maintains an agenda that is a set of P,A pairs, where A is an action in the plan and P is an atom that is a precondition of A that must be achieved. Initially the agenda contains pairs G,finish , where G is an atom that must be true in the goal state.

At each stage in the planning process, a pair G,act1 is selected from the agenda, where P is a precondition for action act1. Then an action, act0, is chosen to achieve P. That action is either already in the plan - it could be the start action, for example - or it is a new action that is added to the plan. Action act0 must happen before act1 in the partial order. It adds a causal link that records that act0 achieves P for action act1. Any action in the plan that deletes P must happen either before act0 or after act1. If act0 is a new action, its preconditions are added to the agenda, and the process continues until the agenda is empty.

This is a non-deterministic procedure. The "choose" and the "either ...or ..." form choices that must be searched over. There are two choices that require search:

- which action is selected to achieve G and
- whether an action that deletes G happens before act0 or after act1.

```
non-deterministic procedure PartialOrderPlanner(Gs)
2:      Inputs
3:          Gs: set of atomic propositions to achieve
4:      Output
5:          linear plan to achieve Gs
6:      Local
7:          Agenda: set of  P,A  pairs where P is atom and A an action
```

```
8:           Actions: set of actions in the current plan
9:           Constraints: set of temporal constraints on actions
10:          CausalLinks: set of   act0,P,act1   triples
11:      Agenda ←{   G,finish   :G   Gs}
12:      Actions ←{start,finish}
13:      Constraints ←{start<finish}
14:      CausalLinks←{}
15:      repeat
16:          select and remove   G,act1   from Agenda
17:          either
18:              choose act0   Actions such that act0 achieves G
19:          or
20:              choose act0   Actions such that act0 achieves G
21:              Actions ←Actions ∪{act0}
22:              Constraints ←add_const(start<act0,Constraints)
23:              for each CL   CausalLinks do
24:                  Constraints ←protect(CL,act0,Constraints)
25:
26:              Agenda ←Agenda ∪{   P,act0   : P is a precondition of act0 }
27:
28 :          Constraints ←add_const(act0<act1,Constraints)
29:          CausalLinks∪ {   acto,G,act1   }
30:          for each A   Actions do
31:              Constraints ←protect(   acto,G,act1   ,A,Constraints)
32:
33:      until Agenda={}
34:      return total ordering of Actions consistent with Constraints
```

Figure 8.5 Partial-order planner

The function add_const(act0<act1,Constraints) returns the constraints formed by adding the constraint act0<act1 to Constraints, and it fails if act0<act1 is incompatible with Constraints. There are many ways this function can be implemented. See Exercise 8.13.

The function protect(acto,G,act1 ,A,Constraints) checks whether A≠act0 and A≠act1 and A deletes G. If so, it returns either { A<act0 } ∪ Constraints or { act1<A } ∪ Constraints. This is a non-deterministic choice that is searched over. Otherwise it returns Constraints.

8.3 Planning the Blocks World

One of the most famous planning domains is known as the blocks world. This domain consists of a set of cube shaped blocks sitting on table. The blocks can be stacked, but only one block can fit directly on top of another. A robot arm can pick up a block and move it to another position, either on the table or on top of another block. The arm can pick up only one block at a time, so it cannot pick up a block that has another one on it. The goal will always be to build one or more stacks of blocks, specified in terms of what blocks are on top of what other blocks.

The block-world problem is known as the Sussmann anomaly.In the block-world problem, three blocks labeled 'A', 'B', and 'C' are allowed to rest on a flat surface. The given condition is that only one block can be moved at a time to achieve the target. The start position and target position are shown in the following diagram as in Figure.

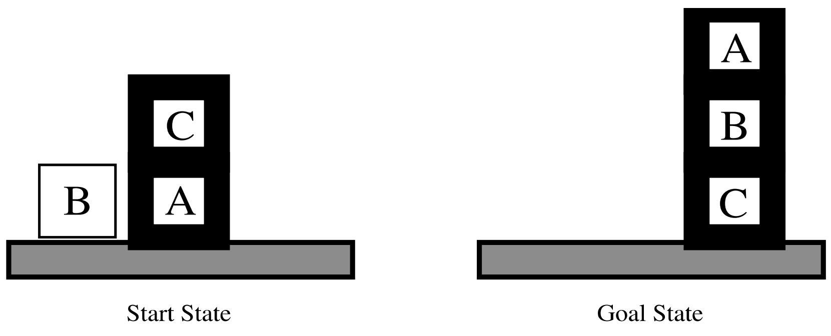


Figure Diagram of the blocks-world problem

We use On(b, x) to indicate that block b is on x, where x is either another block or the table. The action for moving block b from the top of x to the top of y will be Move(b, x, y). Now, one of the preconditions on moving b is that no other block be on it. In first-order logic, this would be ¬ x On(x, b) or, alternatively, x ¬On(x, b). But, basic PDDL does not allow quantifiers, so instead we introduce a predicate Clear(x) that is true when nothing is on x.The action Move moves a block b from x to y if both b and y are clear. After the move is made, b is still clear but y is not.

A first attempt at the Move schema is as given below.

Action(Move(b, x, y),

PRECOND :On(b,x) $\wedge$ Clear(b) $\wedge$ Clear(y),

EFFECT :On(b,y) $\wedge$ Clear(x) $\wedge$ ¬On(b,x) $\wedge$ ¬Clear(y)) .

Unfortunately, this does not maintain Clear properly when x or y is the table. When x is the Table, this action has the effect Clear(Table), but the table should not become clear; and when y = Table , it has the precondition Clear (Table), but the table does not have to be clearfor us to move a block onto it.

To fix this, we do two things. First, we introduce another action to move a block b from x to the table. Second, we take the interpretation of Clear(x) to be "there is a clear space on x to hold a block." Under this interpretation, Clear(Table) will always be true. The only problem is that nothing prevents the planner from using Move(b,x,Table) instead of MoveToTable(b,x).

Action (MoveToTable (b, x),

PRECOND :On(b,x) $\wedge$ Clear(b),

EFFECT :On(b,Table) $\wedge$ Clear(x) $\wedge$ ¬On(b,x)) .

8.4 Hierarchical Planning

The problem solving and planning methods of the preceding discussions operate with a fixed set of atomic actions. Actions can be strung together into sequences or branching. For plans executed by human brain, atomic actions are muscle activations. Even if, we restrict ourselves to planning over much shorter time horizons that is planning for a two week vacation to Hawaii. This needs 10^{10} actions.

Hierarchical Planning is an Artificial Intelligence (AI) problem solving approach for a certain kind of planning problems -- the kind focusing on problem decomposition, where problems are step-wise refined into smaller and smaller ones until the problem is finally solved. A solution hereby is a sequence of actions that's executable in a given initial state (and a refinement of the initial compound tasks that needed to be refined).

Hierarchical planning is also called as plan decomposition. Generally plans are organized in Hierarchical format.

Pop one level planner:

If you are planning to take a trip, then first you have to decide the location. To decide location we can search for various good locations from Internet based on, whether conditions, travelling expenses, etc. This is level one planning.

Hierarchy of actions:

In terms of major and minor or actions, hierarchy of actions can be decided. Minor activities would cover more precise activities to accomplish the major activities.

Major steps are given more importance. Once major steps are decided we attempt to solve the minor detailed actions.

Planner:

i. First identify a hierarchy of major conditions.

ii. Construct a plan in levels.

iii. Patch major level details.

iv. Finally demonstrate

- Planning for "Going to Goa this Cristmas"
    - Switch on computer
    - Start web browser
    - Open Indian Railways website
    - Select date
    - Select class
    - Select train
    - ... so on
  - Practical problems are too complex to be solved at one level

- Planning for "Going to Goa this Cristmas"
  - Major Steps :
    - Hotel Booking
    - Ticket Booking
    - Reaching Goa
    - Staying and enjoying there
    - Coming Back
  - Minor Steps :
    - Take a taxi to reach station / airport
    - Have candle light dinner on beach
    - Take photos

- Actions required for "Travelling to Goa":
    - Opening makemytrip.com (1)
    - Finding flight (2)

- Buy Ticket (3)
- Get taxi(2)
- Reach airport(3)
- Pay-driver(1)
- Check in(1)
- Boarding plane(2)
- Reach Goa(3)

- 1st level Plan :
    - Buy Ticket (3), Reach airport(3), Reach Goa(3)
- 2nd level Plan :
    - Finding flight (2), Buy Ticket (3), Get taxi(2), Reach airport(3), Boarding plane(2), Reach Goa(3)
- 3rd level Plan (final) :
    - Opening makemytrip.com (1), Finding flight (2), Buy Ticket (3), Get taxi(2), Reach airport(3), Pay-driver(1), Check in(1), Boarding plane(2), Reach Goa(3)

8.5 Conditional Planning

It works regardless of the outcome of an action. It deals with uncertainty by inspecting what is happening in the environment at predetermined points in the plan. It can take place in fully observable and non-deterministic environments. It will take actions and must be able to handle every outcome for the action taken. For instance, If <test-cond> then plan A else plan B. In case of a vacuum cleaner problem, If At Left ^ Clean then Right else Suck.

It's a planning method for handling bounded indeterminacy. The Bounded Indeterminacy is the actions that can have unpredictable effects, but the possible effects can be determined. Ex: flip a coin (outcome will be head or tail). It constructs a conditional plan with different branches for the different contingencies that could arise.It's a way to deal with uncertainty by checking what is actually happening in the environment at predetermined points in the plan. (Conditional Steps)

Example:Check whether SFO airport is operational. If so, fly there; otherwise, fly to Oakland.

Conditional Planning in Fully Observable Environments

Agent used conditional steps to check the state of the environment to decide what to do next.

Plan information stores in a library Ex: Action(Left)   Clean v Right

Syntax: If <test> then plan_A else plan_B

Partially Observable Environments

It used the same AND-OR-Graph-Search algorithm, but the belief states will defy differently. Three choices for belief states are given here.

- Sets of full state descriptions. Ex: {(AtR and CleanR and CleanL), (AtR and CleanR and not CleanL)}(not good, the size will become O(2^n))
- Logical sentences that capture exactly the set of possibleworlds (Open-world Assumption)Ex: AtR and CleanR(not that good, it can't represent all domains)
- Knowledge Propositions – describe the agent's knowledge (Closed-world Assumption)Ex: K(P)   means the agent knows that P is true, if it doesn't appear, it's assumed false.

- ❖ Any scheme capable of representing every possible belief state will require O(log2(2^2n)) = O(2^n) bit to represent each one in the worst case.
- ❖ Two kind of Sensing. They are ;

1. Automatic – auto. Check the state
2. Active – agents must use sensory action to check the state of environment. ex: CheckDirt

Agents are not capable of making tradeoffs between the probability of success and the cost of plan construction

Conditional Planning is harder than NP

• NP means that a candidate solution can be checked to see whether it really is a solution in polynomial time

Use a lot of space O(2^n)

Summary

This unit discussed the planning and problem-solving process required in an advanced artificial intelligence. Planning is necessary to find the best solution at a given environment, but the tasks to be done at a particular time.A problem is difficult if an appropriate sequence of steps to solveit, is unknown. Different approaches were analyzed in this unit. As you know, a plan is considered a sequence of actions, and each action has its preconditions that must be satisfied before it can act and some effects that can be positive or negative. Also, it is understood that we require domain description, task specification, and goal description for any planning system. Effective execution of the plan was about choosing a sequence of tasks with a high probability of accomplishing a specific task. This unit also discussed few types of planning such as hierarchical planning and conditional planning along with blocks world problem, for example.

Keywords

- Crisp set
- Planning
- Partial Ordering
- Hierarchical Ordering
- Total Ordering
- Conditional Ordering

Self Assessment

Q1) What is partial order planning in artificial intelligence?

A. Planning where actions must be executed in a fixed order.
B. Planning where actions can be executed in any order.
C. Planning that only involves one action.
D. Planning that doesn't consider preconditions.

Q2) What is a key characteristic of a partial order plan?

A It specifies a fixed execution sequence for actions.
B It defines only the start state and end state.
C It allows for flexible ordering of actions.
D It disregards goal states.

Q3) How are causal links used in partial order planning?

A To establish a strict order of actions.

B To represent the relationship between actions and their effects.

C To ignore the dependencies between actions.

D To represent only preconditions of actions.

Q4) What is the main advantage of partial order planning over total order planning?

A Simplicity of representation.

B Guarantee of finding an optimal plan.

C Flexibility in handling concurrency and parallelism.

D Ability to ignore preconditions.

Q5) What is a threat in partial order planning?

A. A situation where an action's precondition is satisfied by an earlier action.

B. A situation where there is a conflict between action preconditions.

C. A situation where actions are not relevant to the planning process.

D. A situation where there is a single optimal plan.

Q6) How does the "delete-relaxation" technique contribute to partial order planning?

A It eliminates all preconditions from actions.

B It adds extra preconditions to actions to ensure correctness.

C It temporarily removes delete effects for planning purposes.

D It enforces a strict order of actions in the plan.

Q7) What is hierarchical planning in artificial intelligence?

A Planning that involves only a single level of abstraction.

B Planning that breaks down complex tasks into subtasks and levels of abstraction.

C Planning that focuses on random actions without structure.

D Planning that is based on historical data.

Q8) In hierarchical planning, what is a decomposition?

A A method to make plans more complex.

B A process of combining multiple tasks into a single action.

C A technique to simplify plans by breaking them into smaller tasks.

D An approach used for parallel planning.

Q9) What is a benefit of using hierarchical planning?

A It avoids the need for decomposition.

B It always results in the shortest plan.

C It handles uncertainty better than other planning methods.

D It simplifies complex planning problems.

Q10) Which type of knowledge representation is commonly used in hierarchical planning?

A Neural networks

B Decision trees

C Graphs and subgoal networks

D Natural language sentences

Q11) What is a subgoal in hierarchical planning?

A The final goal of the planning process.

B An intermediate goal that needs to be achieved to reach the final goal.

C A goal that is not achievable.

D A goal that can be ignored during planning.

Q12) How does abstraction hierarchy help in hierarchical planning?

A It increases the complexity of the planning process.

B It breaks down goals into more specific subgoals.

C It eliminates the need for subgoals.

D It focuses only on low-level details.

Q13) What is conditional planning in artificial intelligence?

A Planning based on specific conditions

B Creating plans without any constraints

C Planning that involves random actions

D None of the above

Q14) Which term refers to a condition that must be met before an action can be executed in conditional planning?

A Postcondition

B Effect

C Prerequisite

D Constraint

Q15) Which type of planning involves handling uncertainty and incomplete information?

A. Deterministic planning

B. Stochastic planning

C. Hierarchical planning

D. Reactive planning

Q16) What is a conditional effect in AI planning?

A. An effect that always occurs

B. An effect that occurs with a certain probability

C. An effect that depends on external factors

D. An effect that is not relevant to planning□

Q17) Which technique is commonly used to represent conditional effects in AI planning?□

A. And-or graph
B. Decision tree
C. State transition diagram
D. Propositional logic□

Q18) What is the main goal of conditional planning?

A. Finding the shortest plan
B. Adapting plans to changing conditions
C. Maximizing the number of actions in a plan
D. Ignoring external factors during planning

Answers for Self Assessment

1.	B	2.	C	3.	B	4.	C	5.	A
6.	C	7.	B	8.	C	9.	D	10.	C
11.	A	12.	B	13.	A	14.	C	15.	B
16.	B	17.	D	18.	B				

Review Questions

1. Explain the concepts of planning with examples.
2. List the various types of planning.
3. Give an example for hierarchical planning.
4. Identify a real time situation that issuitable forconditional planning.
5. Understanding blocks world planning problem.

Further Readings

- Stuart Russell, Peter Norvig, Artificial Intelligence : A Modern Approach, Third Edition, Pearson Education Ltd., 2016.
- Elaine Rich, Kevin Knight, Shivashankar B Nair, Artificial Intelligence, Third Edition, Tata McGraw Hill, 2009.

Web Links

- https://towardsdatascience.com/fuzzy-inference-system-implementation-in-python-8af88d1f0a6e
- https://pythonhosted.org/scikit-fuzzy/auto_examples/plot_tipping_problem_newapi.html

Unit 09: Constraints

CONTENTS

Objectives

- Illustrating the representation of resource constraints.
- Understanding the concepts of partial order planning.
- Understanding the concepts of total order planning.

Introduction

Studying constraints in artificial intelligence (AI) is essential to harness the true potential of intelligent systems and bridge the gap between theoretical advancements and practical applications. Understanding constraints equips aspiring AI professionals with the tools to create solutions that are not only innovative but also feasible and ethical.By delving into constraints, you gain insight into the real-world complexities that AI systems encounter. This awareness empowers you to develop strategies that optimize limited resources, adapt to time-sensitive scenarios, and adhere to societal norms. Mastery of constraints enhances decision-making, enabling AI to navigate intricate environments while ensuring responsible and accountable outcomes.Moreover, constraints drive innovation. Overcoming challenges posed by constraints often leads to breakthroughs, propelling AI to new heights. From healthcare to finance, AI applications are contingent upon acknowledging and managing constraints effectively. Embracing constraints in AI education fosters well-rounded professionals capable of creating sustainable, impactful solutions that drive progress in an increasingly AI-driven world.

Constraints are the guiding boundaries that steer the course of planning in artificial intelligence (AI), imparting structure and realism to the decision-making process. They play a pivotal role in shaping AI systems' actions, ensuring that solutions align with real-world limitations and practical considerations. One crucial aspect of constraints is their ability to enhance efficiency and effectiveness. Time constraints, for instance, compel AI planners to devise solutions that optimize resources within specified time frames, leading to swift and pragmatic decision-making. Resource constraints, on the other hand, force agents to allocate finite resources judiciously, promoting resourcefulness and adaptability. Furthermore, domain-specific constraints tailor planning to the intricacies of particular environments. Spatial constraints guide navigation and movement, while logical constraints uphold the consistency of actions within a given context. Social constraints address ethical considerations, preventing AI systems from producing solutions that violate societal norms. In essence, constraints serve as the scaffolding that bolsters the foundation of AI planning, fostering intelligent agents that operate harmoniously within the confines of reality. Their recognition and incorporation underscore the maturity and practicality of AI solutions, bridging the gap between theoretical prowess and real-world applicability.

9.1 Representation of Resource Constraints

Resource constraints in artificial intelligence (AI) planning refer to limitations imposed by the availability and allocation of resources required for executing actions and achieving goals. These resources can encompass a wide range of factors, such as time, money, manpower, equipment, and materials. Proper representation of resource constraints is crucial for creating efficient and effective AI plans that account for real-world limitations.

Example 1: Project Scheduling
Consider a construction project where different tasks need to be scheduled over a specific time period. Each task requires a certain amount of labor and equipment. Resource constraints involve ensuring that the available manpower and equipment are allocated optimally to meet the project deadlines. The AI planner needs to represent these constraints, balancing the availability of resources with the demands of various tasks.

Example 2: Travel Planning
In travel planning, an AI system might help users create itineraries. Resource constraints could involve budget limitations and time constraints. The AI needs to represent the costs associated with transportation, accommodation, and activities, ensuring that the planned itinerary stays within the user's budget. Additionally, the AI must consider time constraints to ensure that the planned activities are feasible within the travel duration.

Representation Techniques:
Numerical Representation: Resources can be represented as numerical values, such as quantities of manpower, hours, or monetary amounts. This allows the AI planner to perform mathematical optimizations to allocate resources efficiently.

Symbolic Representation: Resources can be represented symbolically, indicating their availability and requirements in a qualitative manner. This approach is useful for handling imprecise or qualitative resource information.

Constraints Satisfaction Problems (CSP): Resource constraints can be formulated as CSPs, where variables represent tasks, and constraints represent resource limitations. Solving the CSP yields a feasible resource allocation.

Temporal Networks: Temporal networks extend resource representation to include time-based constraints. They allow modeling of resource usage over time, ensuring that resource availability aligns with task scheduling.

Challenges:

- Balancing conflicting resource demands while ensuring feasible solutions.
- Handling uncertainties and variations in resource availability.
- Addressing scenarios where resource constraints change dynamically during planning.
- Efficiently searching through the space of possible plans considering resource limitations.

Properly representing resource constraints in AI planning is pivotal for generating plans that are not only theoretically optimal but also realistic and achievable in practical scenarios. It ensures that AI systems can navigate and operate effectively within the limitations of the real world.

If you consider the project management, then followingdiscussion will give you overall idea and concept understanding what we discussed so far. A resource constraint is any limitation or risk related to resources allocated to projects. Identifying these resource management restrictions is part of the project planning process. Resource constraints can disrupt your project and impede effective delivery. Here are some of the most frequent project management constraints.

Scope

The term "scope constraint" refers not only to what the project team includes in the project but also to what they exclude. Prior to initiating a project, it is critical to thoroughly evaluate its scope to create realistic expectations for the result. To do this, competent project managers begin by developing a project statement that clearly defines the scope of the project, taking into account both internal and external variables.The scope of a project, in most circumstances, can refer to the particular deliverables that key stakeholders have agreed that upon and complete in time. Project managers then exclude deliverables that they did not include in the initial plan.

Cost

Time and cost constraints are the most self-explanatory and easily quantifiable project constraints that you may encounter on almost every project you manage. Cost (or budget) is simply the amount of money invested in a particular activity to achieve the desired outcome. Prior to initiating the project, it is essential to make the most precise cost estimate possible. This enables you to establish a baseline against which you can compare your spending and progress throughout the project.

Time

Another significant factor in determining the success of a project is time management. This is because a determinant of the success of a product is it being completed within the stipulated time frame. Early in a project's life-cycle, it's possible that important stakeholders may define "start no earlier than" and "end no later than" timelines that you can stick to. The timeframe for the project is critical to its success. Having a well-defined project schedule is a critical component of time management effectiveness.Dedicate time to planning upfront to reduce the number of adjustments required later on and to ensure that the team spends time on only productive activities.

Quality

The process of continuously monitoring the quality of all operations and making corrections until the project attains the quality desired in project quality management. Quality management procedures aid in cost control, the establishment of standards, and the determination of the actions necessary to achieve and validate those standards. Additionally, effective quality management reduces the chance of product failure or dissatisfied customers.The quality constraint focuses on the product's qualities. In general, the project's quality can determine how closely the output meets the expectations established during the planning stages.

Satisfaction of the Customer

Customer happiness is another resource constraint to consider. When considering customer happiness as a constraint, it is important for project managers to remember that merely completing a project on time, within budget, and within scope does not guarantee customer satisfaction. In some instances, you may meet the demands of important stakeholders, even if there are no formal clients. It is also important to consider if the initiative contributes to the achievement of the overall company or consumer goal.The more you try to answer the question of how to satisfy your clients, the more effectively you can produce a product, service, or deliverable that fulfills their demands.

Resources

Finally, project managers can evaluate the availability of resources, both material and human. This constraint is generally connected with cost since the amount of cash allocated to a project dictates the degree of experience and availability of resources. constraints on resource availability and accessibility. In today's dynamic business environment, project managers frequently lead teams that are geographically scattered or comprising diverse groups within a company.

Here, we give example problems where we understand the constraint satisfaction problems.

Question 1:

Consider a scheduling problem where you need to assign time slots to a set of lectures in a way that no two lectures with the same topic are scheduled in the same time slot. Each lecture has a list of available time slots. Formulate this problem as a Constraint Satisfaction Problem, defining variables, domains, and constraints.

Question 2:

You are given a Sudoku puzzle as a 9x9 grid with some cells filled in. Formulate solving this Sudoku puzzle as a CSP. Define the variables, domains, and constraints required to find a valid solution.

Question 3:

Imagine a map-coloring problem with a map of countries that need to be colored using three colors: red, green, and blue. Adjacent countries must have different colors. Provide a formulation of this problem as a CSP, including the variables, domains, and constraints.

Question 4:

A cryptarithmetic puzzle is a mathematical game where letters are assigned digits in order to make a valid arithmetic equation. For instance, "SEND + MORE = MONEY" is a cryptarithmetic puzzle. Define the variables, domains, and constraints necessary to solve a generic cryptarithmetic problem.

Question 5:

You are given a set of tasks that need to be assigned to a group of workers. Each worker has a specific skill set, and each task requires a certain combination of skills. Design a CSP to assign tasks to workers optimally, ensuring that each task is assigned to a worker with the required skills.

9.2 <u>Temporal Constraints</u>

Temporal constraints in artificial intelligence (AI) planning refer to limitations and dependencies related to the timing and sequencing of actions within a plan. These constraints ensure that actions are executed in a logical and coordinated manner, reflecting real-world scenarios where the order of events is critical. Proper handling of temporal constraints is essential for generating feasible and coherent plans.

Example 1: Cooking a Meal

Consider the task of preparing a meal with multiple cooking steps. Some ingredients need to be prepared before others can be used. Temporal constraints dictate the order in which these steps must occur. For instance, chopping vegetables must be completed before they can be added to a pan for cooking. AI planning must account for these dependencies to create a successful cooking plan.

Example 2: Project Management

In project management, tasks often have interdependencies and must be completed in a specific sequence. For instance, in software development, coding must precede testing, and testing must precede deployment. Temporal constraints ensure that each phase is executed at the appropriate time to maintain project coherence.

Representation Techniques:

Time Points and Intervals: Represent actions and events as time points or intervals on a timeline. Temporal constraints define the relative positions of these points or intervals. For example, "Action A must happen before Action B" is represented by the constraint $A < B$.

Partial Order Planning: Actions are partially ordered based on temporal constraints, allowing flexibility in the execution sequence while adhering to the required dependencies.

Temporal Logic: Express temporal constraints using temporal logic formalisms, such as Allen's Interval Algebra, which defines relations like "before," "meets," "overlaps," etc.

Temporal Networks: Represent temporal constraints using networks, where nodes represent events, and edges indicate temporal relationships between events.

Challenges:

Dealing with complex temporal relationships involving multiple actions and events.

Handling temporal uncertainty, such as actions with variable durations or uncertain start times.

Ensuring that temporal constraints are satisfied while also considering other types of constraints, like resource constraints.

9.3 Total Order Planning

Total order planning is a type of planning approach in artificial intelligence (AI) where actions are linearly ordered without any partial order or concurrency. In total order planning, the entire sequence of actions is predetermined, and each action is executed one after the other in the specified order. This approach simplifies the planning process by eliminating the need to consider multiple possible orders of actions.

Example 1: Assembly Line Production

Imagine an automobile assembly line where various tasks need to be performed in a specific order to construct a car. Total order planning in this scenario would involve a fixed sequence of actions: attach chassis, install engine, add wheels, mount seats, and so on. Each step is executed in a strict, predetermined order, without any parallelism or choice in the sequence.

Example 2: Baking a Cake

Consider the process of baking a cake, which involves several steps such as mixing ingredients, pouring into a pan, baking, and frosting. Total order planning for baking a cake would involve a predefined sequence of actions, ensuring that each step is executed in the specified order to achieve the desired outcome.

Advantages:

Simplicity: Total order planning is conceptually straightforward, as it eliminates the complexity of considering different combinations of action orders.

Predictability: The plan's execution is deterministic and predictable since there is a fixed order of actions.

Challenges:

Limited Flexibility: Total order planning may not capture the most efficient or optimal plan due to the rigid predetermined order.

Scalability: In complex scenarios with a large number of actions, total order planning can become cumbersome and less manageable.

Total Order Planning Process:

Action Selection: Choose actions that need to be executed, ensuring that they are linearly ordered.

Sequence Generation: Arrange the selected actions in a predetermined order, considering their dependencies and requirements.

Plan Execution: Execute the actions in the specified order to achieve the desired goal.

Use Cases:

Total order planning is suitable for scenarios where the sequence of actions is well-defined, and there is little room for variation or concurrency. It is often employed in domains with highly structured processes, such as manufacturing, assembly lines, and procedural tasks.

While total order planning offers simplicity and predictability, it may not be suitable for all planning scenarios, especially those involving complex dependencies, resource constraints, and parallelism. Partial order planning approaches, which allow for more flexible action sequencing, are often used in such cases

Summary

In this section, we have studied about the constraints, temporal constrains and total order planning very elaborately along with case studies.Temporal constraints ensure that plans are not only feasible in terms of resources but also logically consistent in terms of the order of actions. They enable AI systems to generate plans that accurately reflect real-world processes, resulting in more effective and reliable plan execution.In conclusion, temporal constraints are a fundamental aspect of AI planning, shaping the sequencing and coordination of actions within plans to ensure their successful execution in dynamic and time-sensitive environments

Keywords

- Constraints
- Temporal constraints
- Total order planning

Self Assessment

Q1: In the context of AI planning, what do temporal constraints specify?

A. The order of actions in a plan.
B. The duration of each action.
C. The resource allocation for actions.
D. The rewards for completing actions.

Q2: Which type of temporal constraint enforces that one action must start immediately after another action finishes?

A. Start-to-Start (SS)
B. Finish-to-Start (FS)
C. Start-to-Finish (SF)
D. Finish-to-Finish (FF)

Q3: In AI planning, what does the term "temporal flexibility" refer to?

A. The ability to delay actions indefinitely.
B. The ability to change the order of actions.
C. The ability to execute actions simultaneously.
D. The ability to skip certain actions.

Q4: Which AI planning approach allows for the representation of both temporal and resource constraints?

A. STRIPS planning
B. Hierarchical Task Network (HTN) planning
C. Partial Order Planning (POP)
D. Constraint Logic Programming (CLP)

Q5: In AI planning, what does the term "chronicles" refer to?

A. Historical records of past plans.
B. Temporal reasoning tools.
C. Formal representations of temporal constraints.
D. Sequential execution of actions.

Q6: Which algorithmic approach is commonly used to solve planning problems with temporal constraints?

A. A* algorithm
B. Monte Carlo Tree Search (MCTS)
C. Dijkstra's algorithm
D. Temporal Fast Downward (TFD)

Q7: What does the "distance-based heuristic" in AI planning with temporal constraints measure?

A. The duration of an action.
B. The spatial distance between locations.
C. The temporal distance between actions.
D. The cost of resource allocation.

Q8: In AI planning, what is the "temporal horizon"?

A. The maximum time limit for solving a planning problem.
B. The time at which a specific action must be completed.
C. The difference between the earliest and latest start times of an action.
D. The point in time when all actions are completed.

Q9: Which type of temporal constraint enforces that two tasks cannot overlap in time?

A. Exclusion constraint
B. Concurrency constraint
C. Dependency constraint
D. Temporal flexibility constraint

Q10: Which AI planning language provides explicit support for representing temporal constraints?

A. Prolog
B. Lisp
C. PDDL (Planning Domain Definition Language)
D. SQL (Structured Query Language)

Q11: In total order planning, what is the main objective?

A. To find a linear sequence of actions that achieves the goal.
B. To find the shortest sequence of actions to achieve the goal.
C. To find the most resource-efficient plan for the goal.
D. To find a parallel sequence of actions to achieve the goal.

Q12: Which type of planning allows actions to be executed in any order as long as the goal conditions are satisfied?

A. Hierarchical planning
B. Linear planning
C. Partial order planning
D. Sequential planning

Q13: What is a potential drawback of total order planning?

A. It can be computationally expensive for complex problems.
B. It cannot handle non-deterministic actions.
C. It cannot handle resource constraints.
D. It cannot handle parallel execution of actions.

Q14: What is a "mutex" in the context of total order planning?

A. A type of action that cannot be modified.
B. A synchronization mechanism to ensure exclusive access to resources.
C. A type of goal condition in planning.
D. A type of heuristic used in planning algorithms.

Q15: Which algorithm is commonly used for total order planning in AI?

A. A* algorithm
B. Dijkstra's algorithm
C. STRIPS algorithm
D. BFS algorithm

Q16: What is the keyadvantage of partial order planning over total order planning?

A. Greater efficiency in finding plans.
B. Ability to handle parallelism and concurrency.
C. Simplicity in implementation.

D. Better handling of non-deterministic actions.

Q17: In partial order planning, what represents the temporal relationship between actions?

A. Precondition
B. Effect
C. Causal link
D. Operator

Q18: Which approach allows for the refinement of a partial order plan into a total order plan?

A. Forward planning
B. Backward planning
C. Linearization
D. Constraint relaxation

Q19: What is the role of a "threat" in the context of partial order planning?

A. A constraint that blocks an action's execution.
B. An action that leads to a dead-end state.
C. A resource that is unavailable for an action.
D. A type of heuristic used for plan optimization.

Q20: What is the purpose of the "open conditions" in a partial order planning representation?

A. To track actions that have been executed.
B. To keep track of unsatisfied preconditions.
C. To represent the final goal state.
D. To indicate the actions that can be executed in parallel.

Answers for Self Assessment

1.	A	2.	B	3	B	4.	C	5.	C
6.	D	7.	C	8.	A	9.	B	10.	C
11.	A	12.	C	13.	D	14.	B	15.	C
16.	B	17.	C	18.	C	19.	A	20.	B

Review Questions

1. Explain the concept of total order planning in the context of artificial intelligence. How does it differ from partial order planning?
2. Consider a scenario where you're planning the steps to make a sandwich. Provide an example of a planning problem using total order planning, including the initial state, goal state, and a possible sequence of actions.

3. Define what temporal constraints are in the context of planning and scheduling. How do they impact the execution of plans?
4. Describe the difference between controllable and uncontrollable temporal constraints. Provide examples of each and explain how they affect the planning process.
5. Consider a project where tasks have both temporal constraints and resource dependencies. How can you create a schedule that optimizes both temporal aspects and resource allocation?

Further Readings

Stuart Russell and Peter Norvig, Artificial Intelligence: A Modern Approach, Third Edition, Prentice Hall of India, 2010

.

Web Links

- https://pages.cs.wisc.edu/~dyer/cs540/notes/pop.html
- https://users.aalto.fi/~rintanj1/jussi/temporalplanning.html
- https://en.wikipedia.org/wiki/Partial-order_planning
- http://aima.cs.berkeley.edu/newchap11.pdf

Unit 10: Uncertainty

CONTENTS

Objectives

1. Understanding the basic probability concepts.
2. Illustrating the various types of probability.
3. Understanding Bayesian belief networks.
4. Exploring the applications of bayes theorem.
5. Understanding the posterior probability and prior probability.

Introduction

In this unit, we introducethe basic concepts of probability starting from dependent events, independent events, joint probability, total probability and etc. Actually, probability and the uncertainty are not same. Probability is used to calculate the numerical value. Various techniques areusedto handle the uncertainty using the available information such as certainty factors, dumpster-shafer theory, fuzzy logic and so on. In this unit, we are going to discuss more on bayes theorem and Bayesian belief networks along with the related fundamentals. Probability theory is the base for all these techniques and hence we start with basic probability notions. This unit really gives you a different way of approach in solving the problems.

10.1 Basic Probability

The best example to understand the probability is flipping the coin. Consider we are flipping a coin multiple times. How do we calculate the likelihood of obtaining few numbers of heads during flipping is performed? This is a one of the simple classical example for probability theory. This can be extended into quite more complicated considering flipping multiple coins at a time.

Now, we use the term “Event” to refer coin flipping. Probability is used to compute a numerical value for the event how likely it is to happen. We will have probability value as 1 whenever the event will happen for sure and the probability value is assumed as 0 when the event will never happen for sure. Now, we can understand that the probability value will be ranging from 0 to 1 only.

Let us create our sample space for coin flipping. We flip it for 10 times, for example. The head and tails are as given below on these 10 times. They are, A = { H, T, T, H, T, H, T, H, T, T }.

$$\text{The probability of having Tail} = \frac{\text{Number of Tails Obtained}}{\text{Total number of flipping performed}} = \frac{6}{10} = 0.6$$

Hence, the probability of having tail is 0.6 and the probability of having head will be 0.4. This means that, the likelihood of getting the tail than the head is higher in the next flipping.

Let us do one more example. Abox containing 10 balls, where 3 are green, 2 are red and 5 balls are blue. From this box, we randomly pick a ball. Assume the G for picking green ball, R for picking red ball and B for picking blue balls.

The probability P(G) = Number of green balls / Total number of balls

= 3 / 10

= 0.3

The probability P(R) = Number of red balls / Total number of balls

= 2 / 10

= 0.2

The probability P(B) = Number of blue balls / Total number of balls

= 5 / 10

= 0.5

Now, let us calculate the probability of picking a green ball, if you know that the ball you picked is either green or red. This can be written like this, means that the probability P(G) is needed given that P(GUR) is true.

$$P(G|G\cup R) = \frac{P(G)}{P(G\cup R)} = \frac{3/10}{5/10} = \frac{3}{5} = 0.6$$

This is known as conditional Probability. One event with the presence of another and can be written as P(A / B) or P(A and B). These events can be of two types such as dependent events and independent events.

Dependent Events

Those events whose occurrence of one affect the probability of occurrence of the other is known as dependent events. Consider this situation, For example, suppose a bag has 3 red and 6 green balls. Two balls are to be drawn from the bag one after the other. In this case, one’s probability affect the propability of another.

Independent Events

Those events whose occurrence of one does not affect the probability of occurrence of the other is known as independent events. Consider this situation of same coin flipping example.Because one flip of the coin has no effect on the outcome of any other flips. Hence, it is known asan independent event.

Joint probability

This is calculated on the likelihood of events when those two events are occuring at the same time.Those events may be dependent events or independent events. Different formulas are used on different events. Let us assume A and B are events.

Formula in case of dependent events	: P(A , B) = P(A / B) * P(B)
Formula in case of independent events	:P(A , B) = P(A) * P(B)

Total Probability

Let us have a problem to start with."A person has undertaken a mining job. The probabilities of completion of the job on time with and without rain are 0.42 and 0.90 respectively. If the probability that it will rain is 0.45, then determine the probability that the mining job will be completed on time."Here, to be specific, we have lot of choices on the events that may happen at a given time such rain may come / rain may not come.

Hence, Total probability is used to calculate the probability for all the possible events.

10.2 Bayes Rule

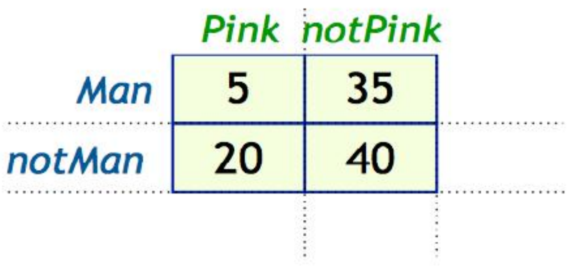
This is named after 18th-century British mathematician Thomas Bayes. This is dealing with conditional probability. Conditional probability is the likelihood of an outcome occurring, based on a previousoutcome having occurred in similar circumstances.In short, bayes theorem helps to find out the revised probability of an event occurring after receiving the new information. The formula of bayes rule or bayes theorem is given below in figure 1.

$$P(H_i \backslash E) = \frac{P(E \mid H_i) \cdot P(H_i)}{\sum_{n=1}^{k} P(E \mid H_n) \cdot P(H_n)}$$

Figure 1 Bayes Theorem

The notation E refers to the event and H refers to the hypothesis. The probability of P(H) which is old probability, is being revised by considering the new evidence, E.

This can be easily explained from an example. Considerthat 100 people are at a party, and you count how many wear pink color dress or not pink color dress, and do the same for men. Now, the numbers counted are shown in the figure 2.



	Pink	notPink
Man	5	35
notMan	20	40

Figure 2 Example for Bayes Theorem

We now calculate some probabilities from the above figure. Understand that total pink colored person is 25, not pink colored person is 75, total count of Man is 40 and Not Man is 60. Remember that the total number of person attending party is 100.

- Probability of person being a man is P(Man) is 40 / 100 = 0.4
- Probabilityof person wearing pink color is P(Pink) = 25 / 100 = 0.25
- Probabilitythat a man wears pink color is P(Pink / Man) = 5 / 40 = 0.125

Can you answer the question like what is the probabilitythat a person wearing pink colored dressis a man i.e., P(Man / Pink) = ?

Using the bayes theorem,

$$P(Man / Pink) = \frac{P(Man)\ P(Pink / Man)}{P(Pink)} = \frac{0.4 * 0.125}{0.25} = 0.2$$

Similarly, let us have another example to detect spam e-mails in my inbox. Let total emails is 100. The word 'offer' occurs in 80% of the spam mails and occurs in 10% of my desired e-mails. The percentage of spam in the whole e-mail is 30% and the percentage of desired mails is 70%.

A new mail received with a word 'offer', what is the probability that it is spam?

The details given in the question can be obtained clearly from the below diagram Figure 3.

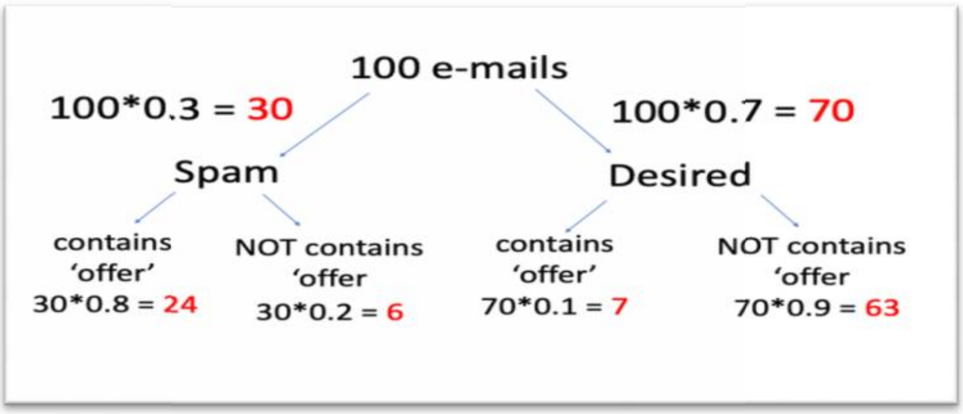

100 e-mails
100*0.3 = 30
100*0.7 = 70
Spam
Desired
contains 'offer' 30*0.8 = 24
NOT contains 'offer 30*0.2 = 6
contains 'offer' 70*0.1 = 7
NOT contains 'offer 70*0.9 = 63

Figure 3 Description of the Given Question

Now, apply the numbers in the Figure 3 and answer the question. Remember that P (contains offer / spam) = 0.8, P(spam) = 0.3, P(contains offer) = 0.3 * 0.8 + 0.7 * 0.1 = 0.31.

$$P(spam|contains\ offer) = \frac{0.8 * 0.3}{0.31} = 0.774$$

Hence, the probability that it is spam given that a new mail received with a word 'offer' is 0.774.

10.3 Prior Probability

Bayes' theorem relies on incorporating prior probabilitydistributions in order to generate posterior probabilities. The terms in the bayes theorem is explained in the figure 4. In short, Bayes theorem is used to compute the posterior probability using the prior probability.

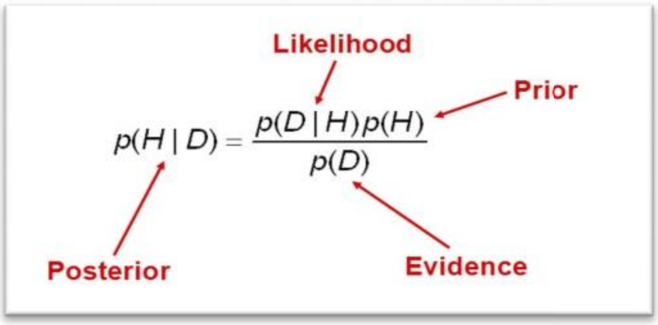


Figure 4 Posterior and prior probability

The prior probability is the probability assigned to an event before the arrival of some information that makes it necessary to revise the assigned probability.

10.4 Posterior Probability

Similar to the explanation ofprior probability in Section 10.3, the posterior probability is understood that it is the outcome of the revised probability after the event occurred.

10.5 Belief Networks

This network can be known in different names such as Bayesian Networks or Bayes Network or Belief Network or Bayesian Belief Networks. We use the term "Bayesian Network" in this section.Bayesian network is a type of probabilistic graphical model that uses Bayesian inference for probability computations.This model is represented in the form ofDirected Acyclic Graph as shown in Figure 5. As you look at this figure, you can understand that it represents and uses Joint Probability for the computations.Here, Figure 5 depicts a problem. Consider Burglary, Earthquake, JohnCalls and MarryCalls are events. Arrow marks gives you indication that one event enables another event in the given direction. This Links are representing Causal dependence. For example, if burglary happens then alarm will be ON. If the alarm is ON, then it enables calling of John and calling of Marry. Similarly, if the earthquake happens, then also alarm will be ON, John and Marry will be getting the calls. But, the probability of those events are given in the tables known as Conditional Probability Table (CPT).

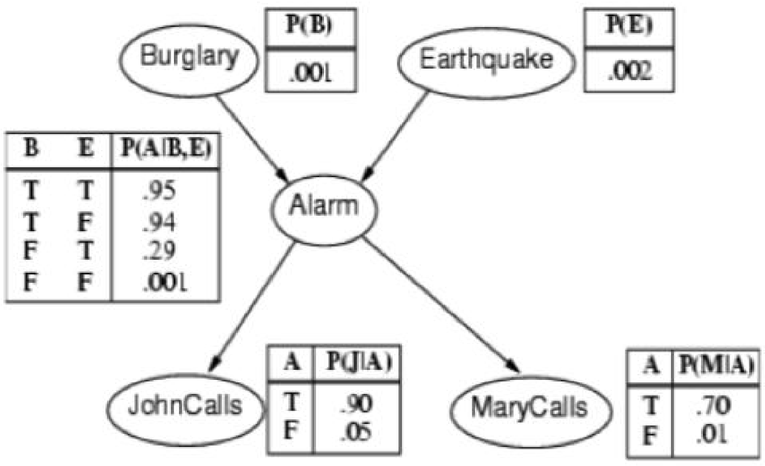


Figure 5 Directed Acyclic Graph Model

We discussed about the events in the previous sections too. The events are two types i.e., dependent and independent. The Dependent Events are those the outcome of one event is having impact on another event. Independent Events are those outcome of the one event is not having any impact on another event. The notations can be given to all these events like in the figure 6.

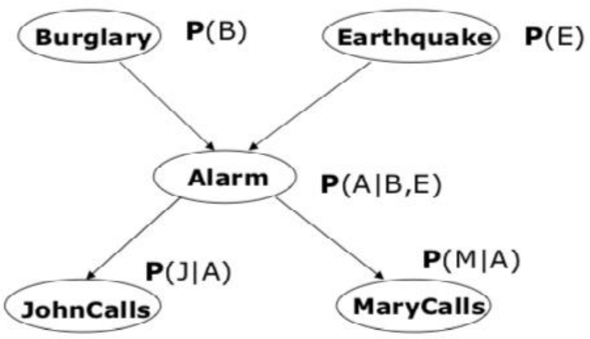


Figure 6 The notations of the events and the conditional probability

Now, we go for the problem like what is the probability of eventthat the alarm has sounded and no burglary but an earthquake has occurred and both Marry and John getting calls?

$$P(J \wedge M \wedge A \wedge \sim B \wedge E) = P(J|A) \times P(M|A) \times P(A|\sim B \wedge E) \times P(\sim B) \times P(E)$$
$$= 0.90 \times 0.70 \times 0.29 \times 0.999 \times 0.002 = 0.00036$$

Look at the next question from the same figure 5 and 6 like, what isthe probability of the event that alarm has sounded but neither a burglary nor an earthquake has occurred and john getting a call and Marry did not get a call ?

$$P(J \wedge \sim M \wedge A \wedge \sim B \wedge \sim E) = P(J|A) \times P(\sim M|A) \times P(A|\sim B \wedge \sim E) \times P(\sim B) \times P(\sim E)$$
$$= 0.90 \times 0.30 \times 0.001 \times 0.999 \times 0.998 = 0.00027$$

Let us see one more example as in given in Figure 7.Consider the following like A, B, C and D are Boolean random variables. If we know that A is true, then what is the probability of D being true?

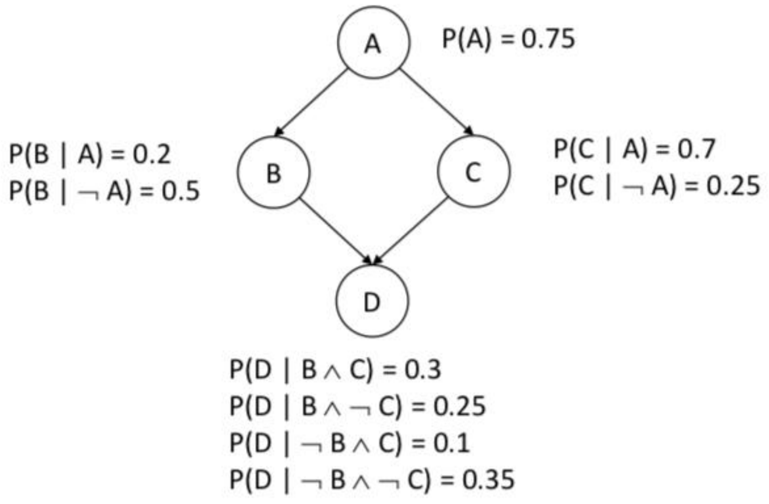


Figure 7 Bayesian Networks - Example

P (D / A) = P (A, D) / P(A)

= (P (A, B, C, D) + P (A, B, ~C, D) + P (A, ~B, C, D) + P(A, ~B, ~C, D)) / P(A)

= P(B / A) * P(C / A) * P(D/ B, C) + P(B / A) * P(~C/A) * P(D/B, ~C) + P(~B/A) * P(C/A) * P (D / ~B, C) + P(~B/A) * P(~C/A) * P(D/~B, ~C)

= (0.2 * 0.7 * 0.3) + (0.2 * 0.3 * 0.25) + (0.8 * 0.7 * 0.1) + 0.8 * 0.3 * 0.35)

= 0.042 + 0.015 + 0.056 + 0.084

= 0.197

Summary

In this unit, we have studied about the basic concepts of probability theory such as dependent events, independent events, joint probability, total probability, sure event and other related terminologies. We understood the difference between the probability and the uncertainty. The existing events were considered for the computations of probability. The bayes theorem was used to compute the posterior probability based on the current probability (prior probability). We understood how the probability changes if the new events occurred.Discussed about the

conditional probability in the Bayesian Networks. Examples were given for Bayesian networks and also bayes theorem for better understanding. We solved few problems too.

Keywords

- Uncertainty
- Probability
- Conditional Probability
- Bayes theorem
- Bayes belief networks
- Prior probability
- Posterior probability

Self Assessment

Q1) An event in the probability that will never be happened is called as ________ .

A Unsure event

B Sure event

C Possible event

D Impossible event

Q2) In a box, there are 8 orange, 7 white, and 6 blue balls. If a ball is picked up randomly, what is the probability that it is neither orange nor blue?

A 1/3

B 1/21

C 2/21

D 5/21

Q3) If a number is selected at random from the first 50 natural numbers, what will be the probability that the selected number is a multiple of 3 and 4?

A 7 / 50

B 4 / 25

C 2 / 25

D None of these

Q4) The probability of getting two tails when two coins are tossed is __________.

A 1/6

B 1/2

C 1/3

D ¼

Q5) A jar containing 8 marbles of which 4 red and 4 blue marbles are there. Find the probability of getting a red given the first one was red too.

A 2 / 11

B 3 / 7

C 4 / 13

D 8 / 15

Q6) What will be the value of P(not E) if P(E) = 0.07?

A 0.90

B 0.07

C 0.93

D 0.72

Q7) Previous probabilities in Bayes theorem that are changed with the new available information is called _______________.

A Independent probabilities

B Dependent probabilities

C Interior probabilities

D Posterior probabilities

Q8) Three companies A, B and C supply 25%, 35% and 40% of the notebooks to a school. Past experience shows that 5%, 4% and 2% of the notebooks produced by these companies are defective. If a notebook was found to be defective, what is the probability that the notebook was supplied by A?

A 44⁄69

B 25⁄69

C 13⁄24

D 11⁄24

Q9) How the bayesian network can be used to answer any query?

A Joint distribution

B Partial distribution

C Full distribution

D All the above

Q10) A bag contains 6 red, 5 blue, and 4 yellow balls. 2 balls are drawn as in the given order. The probability for P (red, then blue) is _____________.

A 5 / 14

B 6 / 15

C 1 / 7

D None of these

Q11) The Bayesian network graph does not contain any cyclic graph. Hence, it is known as _______.

A DCG

B DAG

C CAG

D SAG

Q12) A meeting has 12 employees. Given that 8 of the employees is a woman, find the probability that all the employees are women?

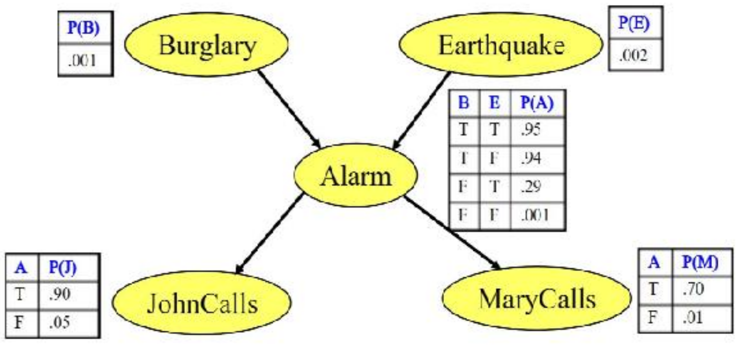
A 11 / 23

B 12 / 35

C 2 / 9

D 1 / 8

Q13) Consider the given diagram and calculate the value of P (B, E, A, J, M) ?



A P(B).P(E).P(A).(J).P(M)

B P(B).P(E).P(A/B,E).P(J/A).P(M/A)

C P(B).P(E).P(A).P(J/A,B).P(M/A,E)

D P(B).P(E/A).P(A/B).P(J/B).P(M/E)

Q14) Bayesian Belief Network is also known as ___________.

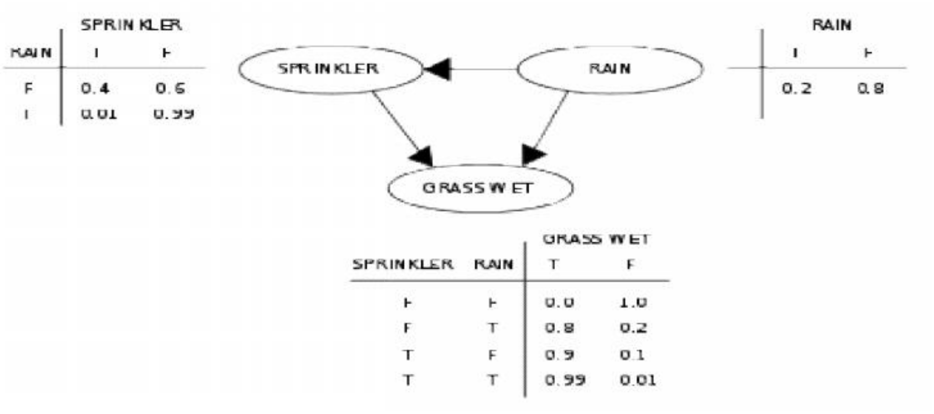
A Belief network

B Decision network

C Bayesian model

D All of the above

Q15) Consider the given diagram and calculate the value of **P (S, R, G) = ?**



A 0.00198

B 0.0098

C 0.99

D 0.015

Answers for Self Assessment

1.	D	2.	A	3	C	4.	D	5.	B
6.	C	7.	D	8.	B	9.	A	10.	C
11.	B	12.	C	13.	B	14.	D	15.	A

Review Questions

1. What do you mean by total probability.
2. Discuss the term "Likelihood" with an example.
3. Compare joint probability and conditional probability.
4. List out the advantages of Bayesian belief networks.
5. Discuss the dependent and independent events in probability theory.

Further readings

- Elaine Rich, Kevin Knight and Shivashankar B Nair, "Artificial Intelligence", Third Edition, Tata McGraw Hill Publishing Company Ltd., 2009.
- Stuart Russell and Peter Norvig, "Artificial Intelligence: A Modern Approach", Global Edition, Pearson Prentice Hall, 2016.

Web Links

- https://en.wikipedia.org/wiki/Uncertainty_theory
- https://www.cuemath.com/data/bayes-theorem/
- https://www.probabilitycourse.com/chapter9/9_1_1_prior_and_posterior.php
- https://faculty.math.illinois.edu/~r-ash/BPT/BPT.pdf
- https://www.javatpoint.com/bayesian-belief-network-in-artificial-intelligence
- https://people.cs.pitt.edu/~milos/courses/cs2001/cs2001-2.pdf
- https://www.tutorialspoint.com/what-is-bayesian-belief-networks

Unit 11: Fuzzy Set andFuzzy Logic

CONTENTS

Objectives

- Understanding the basic concepts of fuzzy logic.
- Illustrating the various methodologies of Fuzzification and Defuzzification.
- Understanding the difference between the crisp set and fuzzy set.
- Exploring Fuzzy Set Operations with examples.
- Understanding the basics of expert systems.

Introduction

In this unit, we introduce what is Fuzzy logic. Fuzzy logic is an approach for computing based on "degrees of truth" rather than the usual "true or false". Here, you need to recall our conventional way of thinking about logics. Usually we think in terms of Boolean such as yes / no or true / false. Generally speaking, on any job that you are planning to do, you will be preparing yourself on both the sides like what to do, ifthere is some unexpected problem happens and what next if all are going as per our plan.This is the style of thinking via Boolean known as "true or false".But, fuzzy logic is not using this Boolean, instead it is using the concepts of degrees of truth. Let me explain this little later.This is the right to time to mention, Prof.Lofti A. Zadeh, who is known as the father of fuzzy theory and fuzzy logic. Now the question comes what is called fuzzy. We are trying to explain from figure 1 given below. There, you can understand that one person is in dangerous position as a heavy object is coming very fast above his head. Also it has two sections. In the first section, a person is trying to help the second person with complete specification about the objects and its directions. Sometimes you may need more details for computation, which is supposed to be a time consumingprocess. The second person will be dead once we finish our process. This is the point, where fuzzy logic is coming to play an important role in the communications. How to communicate effectively without detailed information. Now, you can understand the second part of the figure 1. The first person just shouted at the second person. The second person understands it and ran away. This is what we represent by the terminology “fuzzy” or “fuzziness”. It does not take any concrete values or meaning. The peoplewill assume the meaning according to their own intelligence. Always, it seems to have confusion in the statement such as “look out”, which is actually called as fuzzy.By definition, Fuzzy means VAGUENESS andFuzzy Logic works well with uncertain data or imprecise data. So, it suits better for human language understanding becausehuman usually don't speak with complete details all the time.Observe your regular conversations with your friends and colleagues. Definitely you can understand, and it will be really a fun understanding the “fuzziness / vagueness” in their conversation.

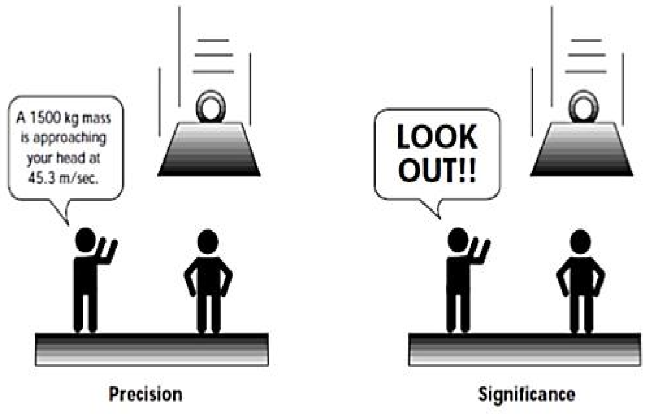


Figure 1 precision and significance

11.1 <u>Fundamentals of Fuzzy Logic</u>

Let us learn the fundamentals of fuzzy logic in this section. Let us start with a question. Assume that you are going in bike as in figure 2. Suddenly you have seen one speed breakon the road. Can you quantify how much break you should apply?We can do it casually, but it is very difficult to make rule for it. Atleast let us discuss on what are all the possible parameters, which can help to decide this. Now, fuzzy logic comes in framing the rules having no quantities or values. Otherwise, we can call these rules as fuzzy rules as it is having vagueness in them.

Figure 2 riding the bike

Fuzzy rules

The parameters involved in framing the fuzzy rules are speed of the bike, distance between bike and speed break, height of the speed break, width of the speed break and etc. We try to create few fuzzy rules. We need the measurement for these parameters not by the values, but by the fuzzy measurements such as low, very low, medium, big, small, average, big, very big, huge and etc.

- If the speed is medium then break is medium.
- If the speed is high then break is very high.
- If the speedbreak is medium size then break is tight.
- If the height of speed break is very less then break is low.

- It the distance is high then break is very smooth.
- If the distance is short then break is medium.
- And etc.…

IF <antecedent> THEN<consequent>

Linguistic Variables

- The idea of linguistic variables is essential to development of the fuzzy set theory.
- A variable whose values are words or sentences in a natural or artificial language.
- Example:Break, Speed, Distance, Speed-break Size are linguistic variables.

Boolean Logic

- Two values are used for a Linguistic variable as in Figure 3.
- Either YES or NO.

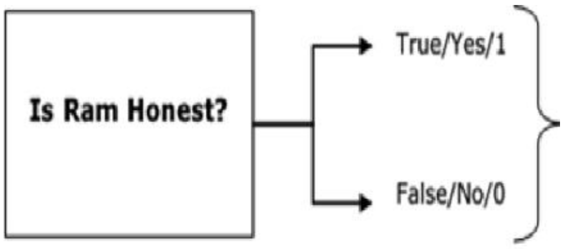


Figure 3Boolean logic

Degree of Truth

- Values are either 0 or 1 for the Crisp Logic and the values are ranging from 0 to 1 for non-crisp logic as in Figure 4.

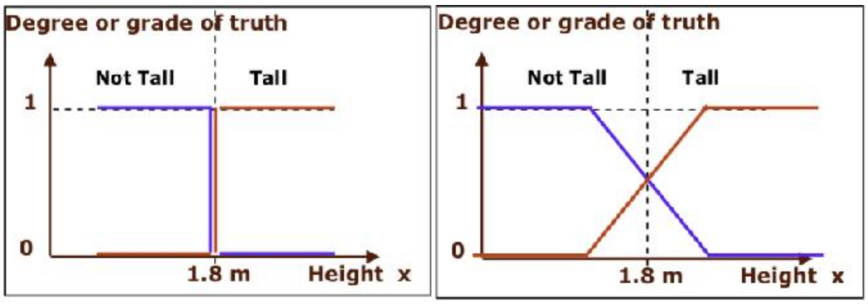


Figure 4Degree of truth for crisp and fuzzy logic

Multi-Valued Logic

- Multiple Values are used for a Linguistic Variable as in Figure 5.
- Also called as Fuzzy Logic.

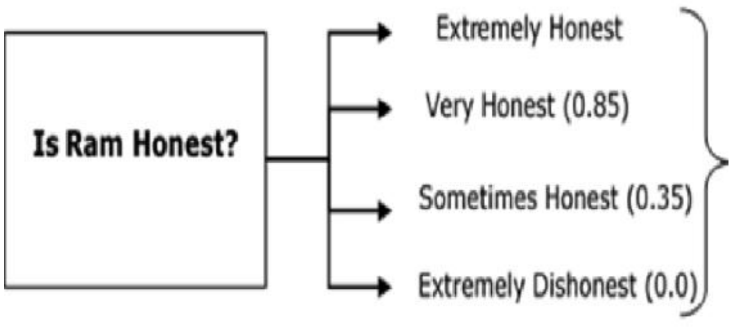


Figure 5Multi-valued logic

- Linguistic variables like Young, Middle-Aged, Old are also fuzzy.
- Difficult to predict the range of values to define them.

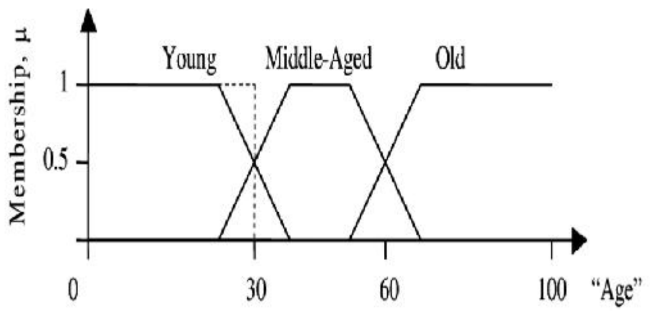


Figure 6Representation of Membership Functions

Crisp Set and Fuzzy Set

Fuzzy logic operates on the fuzzy sets. We can understand how to create fuzzy sets in this section and we can see how it is different from traditional or classical sets, which is known as crisp set.

Crisp set has a unique membership function (mu), represented as μA(x) ∈ {0, 1}. Meaning that the values are either 0 or 1. The values are defined asgiven below where A is the set and x is the element.

$$\mu_A(x) = 1 \text{ if } x \in A$$

$$\mu_A(x) = 0 \text{ if} x \notin A$$

Fuzzy set is fully defined by it's membership functions as in the given Figure 7.

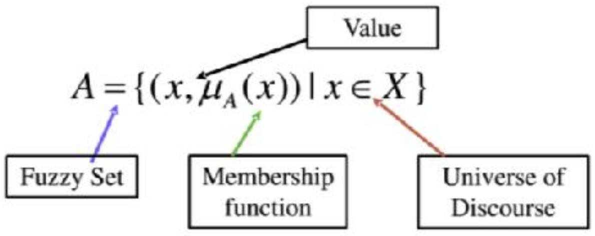


Figure 7Defining a Fuzzy Set

Fuzzy Set can have an infinite number of membership functions, represented as μA∈ [0,1]. Meaning that the values are ranging from 0 to 1. It will take any values between 0 and 1 depending upon the degrees of truth. Maximum value will be given if the degree of truth is maximum and others.

11.2 Operations on Fuzzy Sets

There are few operations like union, intersection, complement are discussed in this section. We are working directly on examples with respect to fuzzy sets, which makes this section simple.

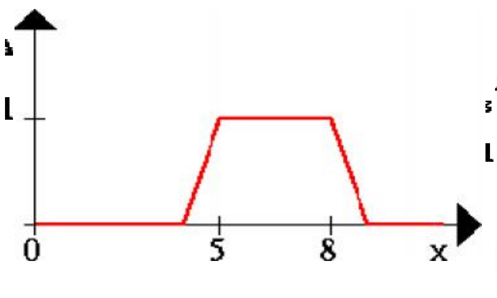
UNION (∪)

Let us take two fuzzy sets A and B. A ∪ B=? The union of two fuzzy sets is the maximum (MAX) of each element from two sets say A and B.

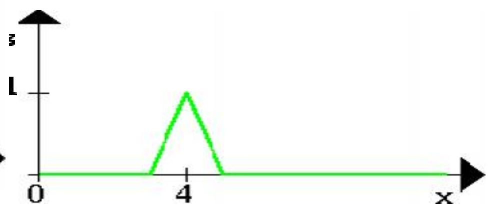
Let A = {(1, 0.4), (2, 0.4), (3, 0.6), (4, 0.7), (8, 0.9)} and B = {(1, 0.8), (3, 0.2), (4, 0.4), (5, 0.6), (6,0.7) }.

= {(1, max(0.4, 0.8)), (2, max(0.4, 0), (3, max(0.6, 0.2), (4, max(0.7, 0.4), (5, max(0, 0.6), (6, max(0, 0.7), (8, max(0.9, 0) }

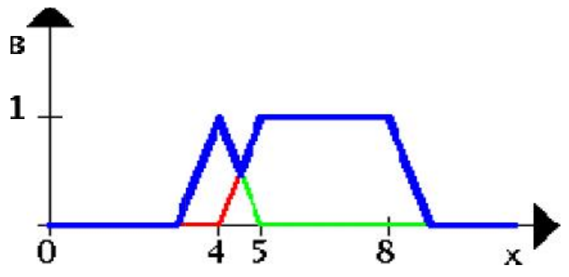
= { (1, 0.8), (2, 0.4), (3, 0.6), (4, 0.7), (5, 0.6), (6, 0.7),(8, 0.9) }



(a)



(b)



(c)

Figure 8(a) Set A, (b) Set B (c) Fuzzy Union

Fuzzy Intersection (Ç)

The intersection of two fuzzy sets is just the MIN of each element from the two sets.

Let A = {(1, 0.4), (2, 0.4), (3, 0.6), (4, 0.7), (8, 0.9) }

Let B = {(1, 0.8), (3, 0.2), (4, 0.4), (5, 0.6), (6,0.7) }

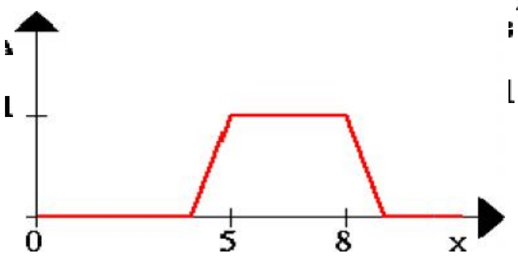
A ∩ B

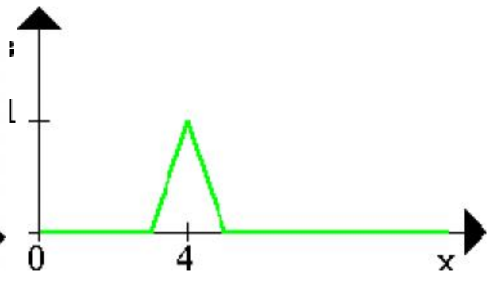
= {(1, min(0.4, 0.8)), (2, min(0.4, 0), (3, min(0.6, 0.2), (4, min(0.7, 0.4), (5, min(0, 0.6), (6, min(0, 0.7), (8, min(0.9, 0) }

= { (1, 0.4), (2, 0), (3, 0.2), (4, 0.4), (5, 0), (6, 0), (8, 0) }

= { (1, 0.4), (3, 0.2), (4, 0.4) }

INTERSECTION





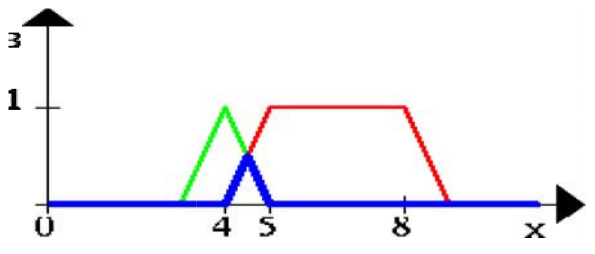


Figure 9(a) Set A, (b) Set B (c) Fuzzy Intersection

COMPLEMENT (-)

The intersection of two fuzzy sets is just the MIN of each element from the two sets.

Let A = {(1, 0.2), (2, 0.4), (3, 0.6), (4, 0.7), (8, 0.9) }

Complement of A

= { (1, (1 – 0.2)), (2, (1 - 0.4), (3, (1 - 0.6)), (4, (1 - 0.7), (8, (1 - 0.9) }

= { (1, 0.8), (2, 0.6), (3, 0.4), (4, 0.3), (8, 0.1) }

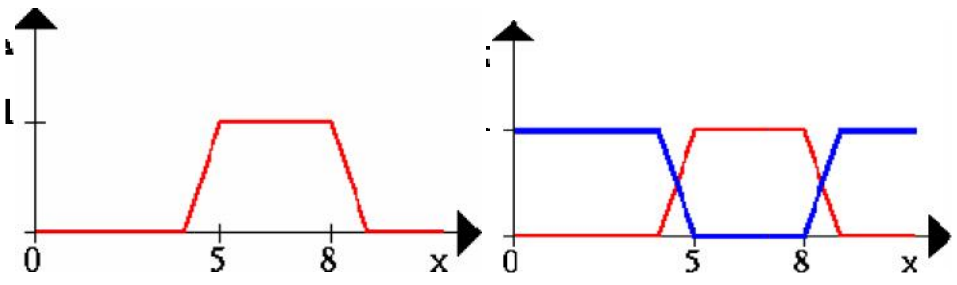


Figure 10(a) Set A (b) Fuzzy Complement of set A

11.3 Fuzzification and Defuzzification Methods

It is the process of converting the crisp values into fuzzy values.Fuzzification is done for each linguistic variable separately.

Membership Functions and Shapes

- There are several types of membership functions such as triangular, trapezoidal, sigmoidal, Gaussian, z-shape and s-shape functions.
- May be selected and used as per our suitability.

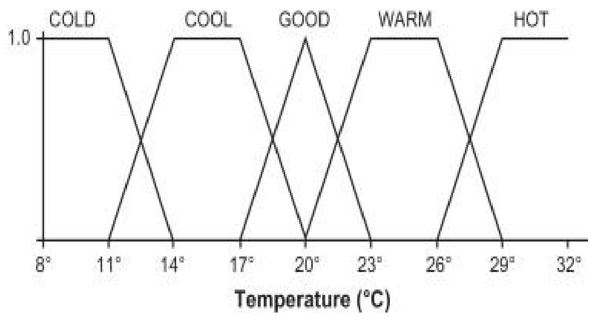


Figure 11Fuzzy Membership Representations

Membership Function

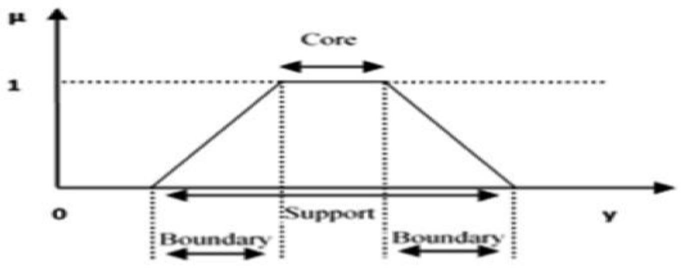


Figure 12Features of Membership Function

There are four methods to perform Fuzzification. They are;

- Intuition
- Inference
- Rank Ordering
- Angular Fuzzy Sets

Intuition Method

Human being's own intelligence and understanding, is used to develop membership functions for Fuzzification. Other expert knowledge is not used in this method of fuzzification.

Inference Method

- The knowledge can be obtained from other sources and the expert interaction.
- This inferred knowledge is used to develop the membership functions for fuzzification.

Rank Ordering

Preferences by a single individual, committee, a poll and other opinion methods, is used to assign membership values for fuzzification.

Angular Fuzzy Sets

- Angular fuzzy sets are defined on a universe of angles,
- They are of repeating shapes for every 2π cycles.
- Angular fuzzy sets are used in the quantitative description of the linguistic variables, which are known as "truth values".

Similarly there few methods used for defuzzification too. They are as follows.

- Lambda-Cut Method
- Weighted Average Method
- Maxima Methods
- Centroid Methods

11.4 Lamda Cut Method

- Lambda-cut method converts a fuzzy set (or a fuzzy relation) into crisp set (or relation).
- Also known as Alpha-cut Method.

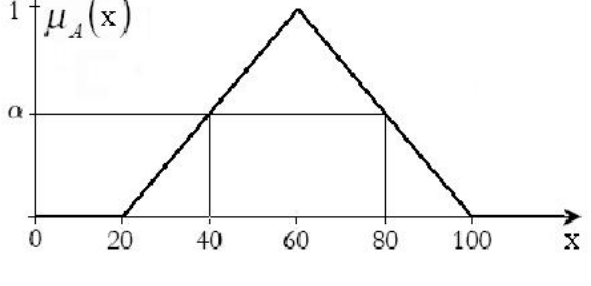


Figure 13 Lambda cut or alpha cut method

Lambda Cut on Fuzzy Sets

- Perform the lambda cut on the given set A1.
- Assume the lambda value as 0.6.

$$A_1 = \{(x_1, 0.9), (x_2, 0.5), (x_3, 0.2), (x_4, 0.3)\}$$

Output is 1 if the respective value is greater than or equal to 0.6 otherwise the output is taken as 0.

$$A_{0.6} = \{(x_1, 1), (x_2, 0), (x_3, 0), (x_4, 0)\}$$

Lambda Cut on Fuzzy Relations

- Perform the lambda cut on the given relation R.
- Assume the lambda value as 0.9.

$$R = \begin{bmatrix} 1 & 0.2 & 0.3 \\ 0.5 & 0.9 & 0.6 \\ 0.4 & 0.8 & 0.7 \end{bmatrix} R_{0.9} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Figure 14 Relation and lambda cut for 0.9

Output is 1 if the respective value is greater than or equal to 0.9 otherwise the output is taken as 0.

- Perform the lambda cut on the given relation R.
- Assume the lambda value as 0.5.

$$R = \begin{bmatrix} 1 & 0.2 & 0.3 \\ 0.5 & 0.9 & 0.6 \\ 0.4 & 0.8 & 0.7 \end{bmatrix} R_{0.5} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix}$$

Figure 15 Relation and lambda cut for 0.5

Output is 1 if the respective value is greater than or equal to 0.5 otherwise the output is taken as 0.

11.5 Weighted Average Method

This method considers averaging the maximum membership value by weighting each respective membership values.Also known as Sugeno Defuzzification.

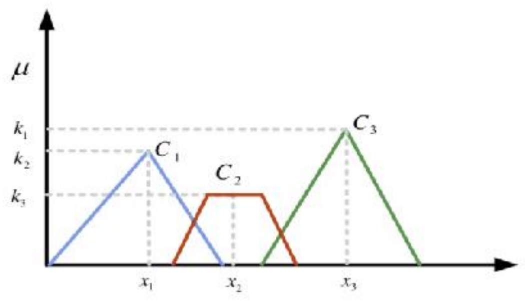
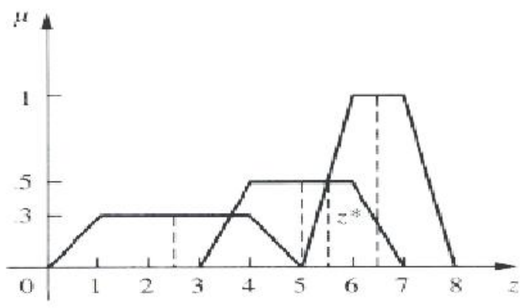


Figure 16 Weighted Average defuzzification

$$X^* = \frac{\sum_{i=1}^{n} \mu_{C_i}(x_i).(x_i)}{\sum_{i=1}^{n} \mu_{C_i}(x_i)}$$



$$z^* = \frac{(.3 \times 2.5) + (.5 \times 5) + (1 \times 6.5)}{.3 + .5 + 1} = 5.41 \text{ meters}$$

Figure 17 Weighted Average defuzzification - calculation

11.6 <u>Maxima Method</u>

This method is having few variants as given below with respect to occurrence / place of maxima.

- Height Method
- First of Maxima Method
- Last of Maxima Method
- Mean of Maxima Method

Height Method (Maxima Method)

- This method acknowledges the peak value for defuzzification.
- This is also referred as Max-Membership Principle.

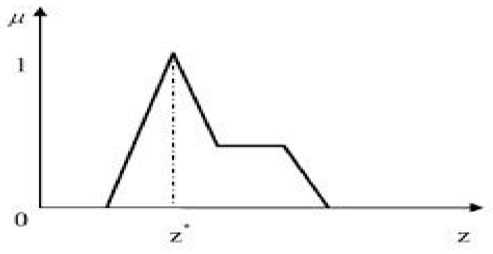


Figure 18 Height Method

First of Maxima Method

- This method considers the smallest value of the domain with maximum membership value.

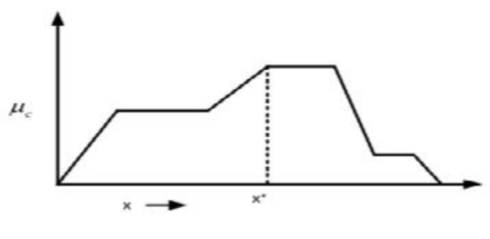


Figure 19 FoM Method

Last of Maxima Method

- This method considers the largest value of the domain with maximum membership value.

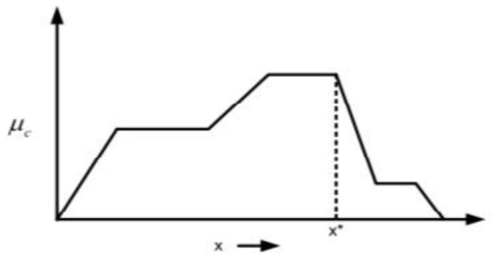


Figure 20LoM Method

Mean of Maxima Method

- This method finds the mean value for defuzzification when there are two peak values.
- This is also referred as middle of maxima.

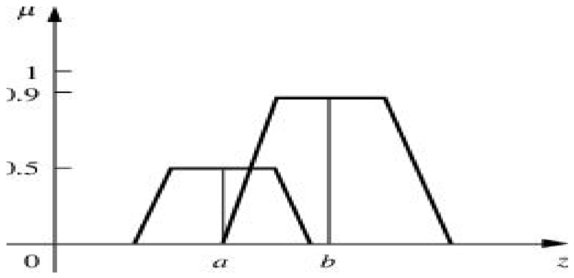


Figure 21 MoM Method

The mean value is calculated as per the given formula.

$$z^* = \frac{a(0.5) + b(0.9)}{0.5 + 0.9}$$

In case, if the multiple peaks exist, thenthe middle point is considered between two (extreme) peaks as shown in the given diagram below.

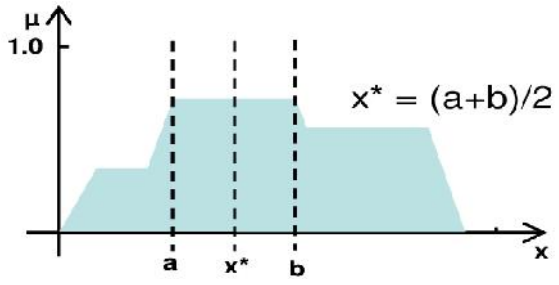


Figure 22Mean Value

11.7 Center of Area Method (COA)

This method is also known as the centre of mass, centre of area or centre of gravity. It is the most commonly used defuzzification method.

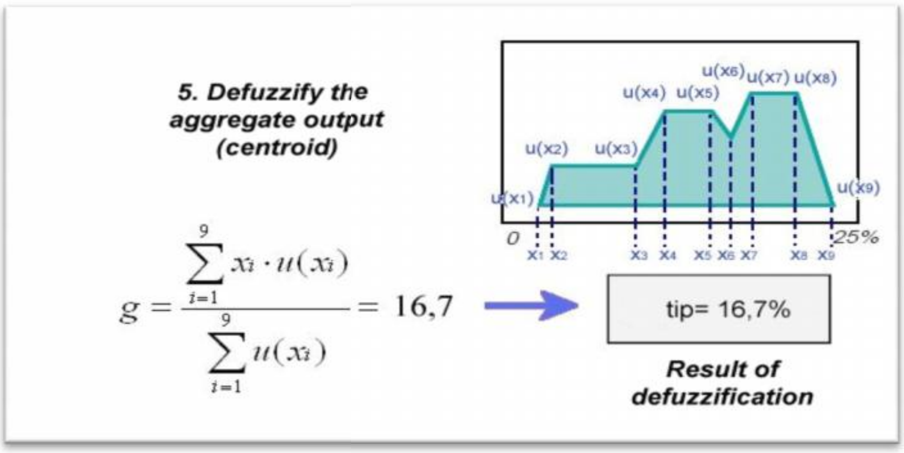

5. Defuzzify the aggregate output (centroid)
$g = \frac{\sum_{i=1}^{9} x_i \cdot u(x_i)}{\sum_{i=1}^{9} u(x_i)} = 16,7$
u(x1) u(x2) u(x3) u(x4) u(x5) u(x6) u(x7) u(x8) u(x9)
0
x1 x2 x3 x4 x5 x6 x7 x8 x9
25%
tip= 16,7%
Result of defuzzification

Figure 23CoA Defuzzification

11.8 Expert System

The concept of expert systems was first developed in the 1970s by Edward Feigenbaum, professor and founder of the Knowledge Systems Laboratory at Stanford University. The system is acting like a domain expert. The knowledge was given through rules. Fuzzy logic is used here for decision-making and for inference in particular.

The finalized fuzzy system is known as expert system, which receives the input and gives the output, which will be looking like expert's decision. The term "Fuzzy inference" is the process of formulating the mapping from a given input to an output using fuzzy logic. It uses the fuzzy sets, fuzzy rules as well as knowledge base. All the output of rules is considered to make a final fuzzy output as discussed already. The complete information is understood from the given diagram.

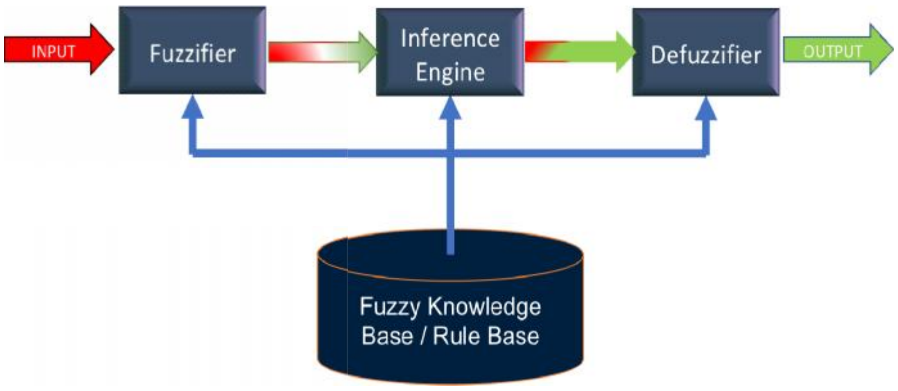


Figure 24Fuzzy Block Diagram

11.9 Decision Trees

Decision Trees (DTs) are a non-parametric supervised learning method used for classification and regression problems. The structure of the decision tree is understood from the given figure 25.

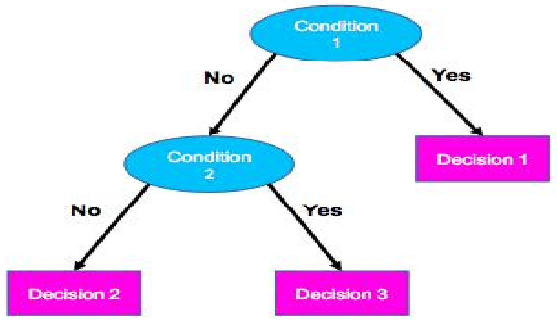


Figure 25 Decision Tree

Hypothesis of Decision Tree

- Tree size should be minimum.
- One who gets the required answer with less number of questions is assumed as efficient.

Components of Decision Tree

- Nodes
- Edges / branches
- Leaves

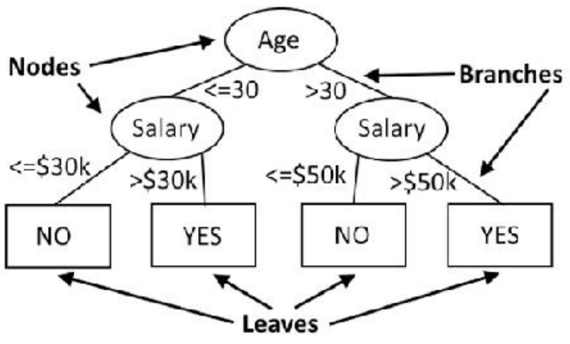


Figure 26 Decision Tree components

Training Dataset

The dataset used for training the decision tree and the working model can be seen in the link. The final trained decision tree is shown in Figure 27.

https://sites.ualberta.ca/~hadavand/DataAnalysis/notebooks/DecisionTree.html

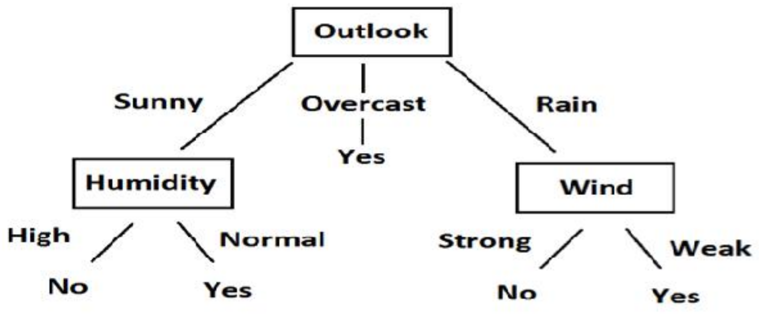


Figure 27 Decision Tree Model

Selection of Good Attribute

Different measures are used to select good attribute. They are listed here.

- Entropy

- Information Gain
- Gini Index

The algorithm ID3 (Iterative Dichotomiser 3) is using the entropy and information gain as metric to select the attributes. At the same time, the algorithm CART (Classification and Regression Trees) is using the Gini Impurity as metric to select the attributes.

Summary

In this section, we have studied about the basic concepts of fuzzy logic. We understood the difference between the conventional sets and fuzzy sets, as this is the basic element to undergo all the operations in the fuzzy logic. Different membership functions were discussed with appropriate examples. Simple problem statements were discussed for the fuzzy set operations like union, intersection and complement. We tried to make you clear how the membership function is helping to create the fuzzy set, which is the process of fuzzification. Also, we have discussed the process of defuzzification in this section. We discussed the various methodologies used for defuzzification is also highlighted with simple diagrams.Altogether, you understood all the components of fuzzy logic and fuzzy system including the fundamental concepts.

Keywords

- Crisp set
- Fuzzy set
- Union
- Intersection
- Complement
- Fuzzification
- Defuzzification
- Membership function
- Expert system
- Decision Trees

Self Assessment

1. What is the form of Fuzzy logic?

A. Two-valued logic

B. Crisp set logic

C. Many-valued logic

D. Binary set logic

2. Traditional set theory is also known as Crisp Set theory.

A. True

B. False

3. What is Fuzzy Logic?

A. A method of reasoning that resembles human reasoning.

B. A method of question that resembles human answer.

C. A method of giving answer that resembles human answer.

D. None of the Above

4. Fuzzy Logic can be implemented in?

A. Hardware

B. Software

C. Both A and B

D. None of the Above

5. Fuzzy Set theory defines fuzzy operators. Choose the fuzzy operators from the following.

A. AND

B. OR

C. NOT

D. All of the above

6. Fuzzy logic is usually represented as ___________

A. IF-THEN-ELSE rules

B. IF-THEN rules

C. Both IF-THEN-ELSE rules & IF-THEN rules

D. None of the mentioned

7. Which of the following is a type of Membership function?

A. Triangular

B. Trapezoidal

C. Sigma

D. All of the mentioned

8. If A and B are two fuzzy sets with membership functions: μ a(χ) ={0.2, 0.5., 0.6, 0.1, 0.9}and μ b (χ)= {0.1,0.5,0.2,0.7,0.8} then the value of μ a∩ μ b will be ________.

A. {0.2,0.5,0.6,0.7,0.9}

B. {0.2, 0.5,0.2, 0.1,0.8}

C. {0.1, 0.5, 0.6, 0.1,0.8}

D. {0.1, 0.5, 0.2, 0.1,0.8}

9. A __________ point of a fuzzy set A is a point x ∈ X at which μA(x) = 0.5.

A. Core

B. Support

C. Cross-over

D. α – cut

10. Given U = {1,2,3,4,5,6,7} and A = {(3, 0.7), (5, 1), (6, 0.8)} then complement of A will be ____.

A. {(4, 0.7), (2,1), (1,0.8)}

B. {(l, 1), (2, 1), (3, 0.3), (4, 1), (6,0.2), (7, 1)}

C. {(4, 0.3.): (5, 0), (6. 0.2) }

D. {(3, 0.3), (6.0.2)}

11. Consider a fuzzy set old as defined below: Old = {(20, 0.1), (30, 0.2), (40, 0.4), (50, 0.6), (60, 0.8), (70, 1), (80, 1)} then the alpha-cut for alpha = 0.4 for the set old will be _________.
A. {(40,0.4)}
B. {50, 60, 70, 80}
C. {(20, 0.1), (30, 0.2)}
D. {(20, 0), (30, 0), (40, 1), (50,1), (60, 1), (70, 1), (80, 1)}

12. What are the following sequence of steps taken in designing a fuzzy logic machine ?
A. Fuzzification → Rule evaluation → Defuzzification
B. Fuzzification → Defuzzification → Rule evaluation
C. Rule evaluation → Fuzzification → Defuzzification
D. Rule evaluation → Defuzzification → Fuzzification

13. The height h(A) of a fuzzy set A is defined as h(A) = sup A(x) if _____.
A. h(A)=1
B. h(A) = 0
C. h(A) <0
D. h(A)<1

14. The major components of the FLC are _____.
A. Fuzzifier&Defuzzifier
B. Fuzzy Rule Base & Knowledge Base
C. Inference Engine
D. All of the above

15. The truth values of traditional set theory is ____________ and that of fuzzy set is __________.
A. Either 0 or 1, between 0 & 1
B. Between 0 & 1, either 0 or 1
C. Between 0 & 1, between 0 & 1
D. Either 0 or 1, either 0 or 1

Answers for Self Assessment

1.	C	2.	A	3	A	4.	C	5.	D
6.	B	7.	D	8.	D	9.	C	10.	B
11.	D	12.	A	13.	A	14.	D	15.	A

Review Questions

1. Explain the concepts of fuzzy logic.
2. List all the methods used in fuzzification.

3. Discuss the methodology used for defuzzification.
4. Write down the different set operations using fuzzy sets.
5. Discuss the lambda cut operations.

Further Readings

- S. N. Sivanandam, S.N. Deepa, Principles Of Soft Computing,WileyPublications, Second Edition, 2011.
- Rajasekaran, S., Pai, G. A. Vijayalakshmi, Neural Networks,Fuzzy Logic and Genetic Algorithm Synthesis And Applications, Prentice Hall of India, 2013.
- N.P. Padhy, S. P. Simon, Soft Computing WithMatlab Programming, Oxford University Press, 2015.

Web Links

- https://towardsdatascience.com/fuzzy-inference-system-implementation-in-python-8af88d1f0a6e
- https://pythonhosted.org/scikit-fuzzy/auto_examples/plot_tipping_problem_newapi.html

Unit 12: Learning and its Types

CONTENTS

Objectives

1. Understanding the concepts of machine learning.
2. Understanding the difference between supervised and unsupervised learning.
3. Understanding different types of leaning.
4. Understanding the concepts of rule based and inductive learning.
5. Understanding the importance of reinforcement learning.

Introduction

In this unit, the concepts of machine learning are discussed in detail. The different types of machine learning approaches are discussed such as supervised learning, unsupervised learning, inductive learning, rule based learning and reinforcement learning using examples. The preparation of the data sets is discussed along with basic data types.Particularly, the basics were given focus along with importance of each approach.

12.1 Introduction to Machine Learning

Machine learning is a method of data analysis that automates analytical model building. It is a branch of artificial intelligence based on the idea that systems can learn from data, identify patterns and make decisions with minimal human intervention. This is now widely used in across all the domains, starting from classification problems to humanoid robot.

Data set is the collection of data used for machine learning. Basically, the dataset is divdided into three categories. They are training data, testing Data and validation Data. Here, the training data is considered for initial training purpose. Testing data is used for checking the trained machine. Validation data is used for tuning the trained machine with the help of important parameters.Training data is the data used to train an algorithm or machine learning model, is depicted in the given figure 1.

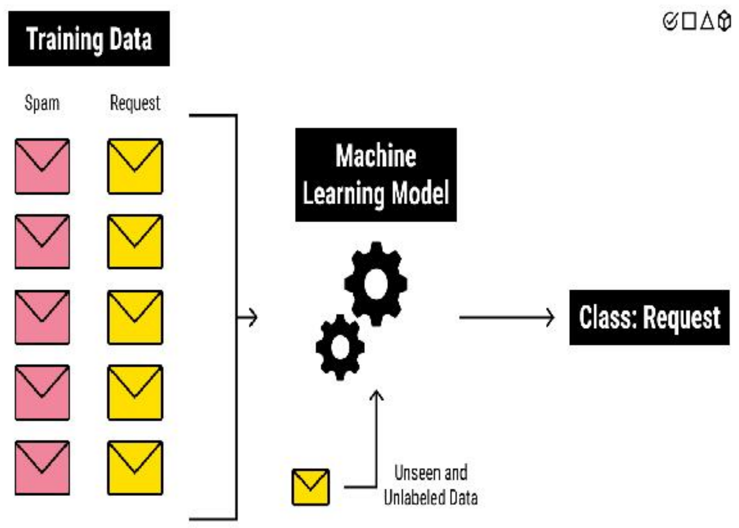


Figure 1 Machine Learning Model

Data preprocessing is the process of transforming raw data into an understandableformat. So that it is used for machine learning effectively. The important steps involved in the process of preprocessing, are Data cleaning, Data integration, Data transformation and Data reduction. Data cleaning. These steps are not in our focus of this unit. But, still, let me give you in short. Data integration is the process of combining data from different sources into a single, unified view.Data transformation is the process of changing the format, structure, or values of data.Data reduction is the process of reducing the amount of capacity required to store data.

12.2 Need for Machine Learning

Today, machine learning is used in a wide range of applications.Machine learning has seen use cases ranging from predicting customer behavior to forming the operating system for self-driving cars.Machine learning as technology helps analyze large chunks of data, easing the tasks of data scientists in an automated process and is gaining a lot of prominence and recognition. Machine learning has changed the way data extraction and interpretation works by involving automatic computing algorithms that have replaced traditional statistical techniques. There are few applications to mention the importance.

- Real-time chat-bot agents
- Decision support
- Customer recommendation engines
- Customer churn modeling
- Dynamic pricing tactics
- Market research and customer segmentation
- Fraud detection
- Image classification and image recognition
- Operational efficiencies
- Information extraction

12.3 Supervised Learning

Machines are trained using well-labelled training data. The data provided for training should be very much correct without any false data as the machine solely depending on the given data for its training as in Figure 2.

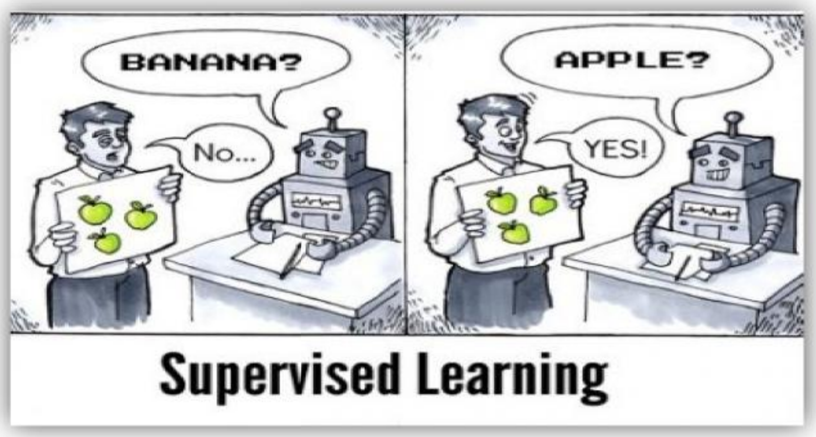


Figure 2Supervised Machine Learning

Four types of data are explained here, as it is often be handled in the process of dataset preparation or preprocessing. The data types are as given below.

- Numerical Data
- Categorical Data
- Time Series Data
- Text Data

Numerical Data

Numerical data is a datatype expressed in numbers as in Figure 3. This further classified as continuous and discontinuous data.

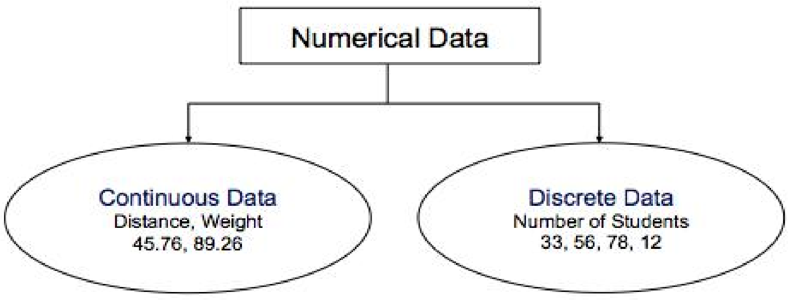
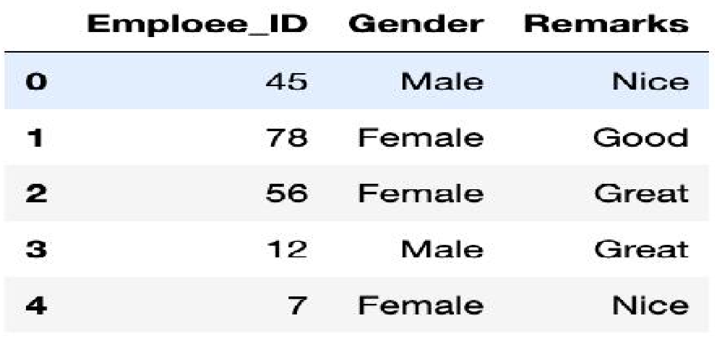


Figure 3Numerical Data

Categorical Data

Categorical data is a collection of information that is divided into groups. We can see the "Gender" and "Remarks" as the categorical data as in the given Figure 4. They are further divided into two types such as ordinal and nominal.



	Emploee_ID	Gender	Remarks
0	45	Male	Nice
1	78	Female	Good
2	56	Female	Great
3	12	Male	Great
4	7	Female	Nice

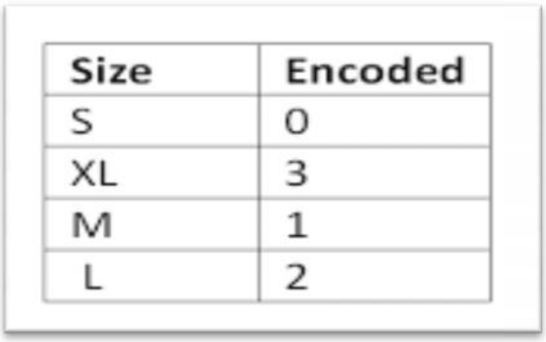
Figure 4Categorical Data

Ordinal Data

Ordinal data has ranking / ordering. Ordinal features are sorted or ordered as in the figure 5.

Size of T-Shirt - S, M, L, XL.

Convert string values into integer as per order like XL > L > M > S.

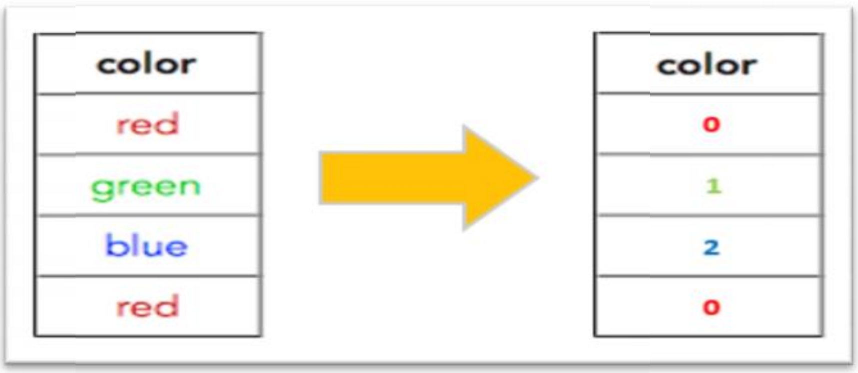


Size	Encoded
S	0
XL	3
M	1
L	2

Figure 5Ordinal Data

Nominal Data

- ❖ Nominal features are not ordered as in figure 6. Nominal data has No ranking / order.
- ❖ Color of T-Shirt : Red, Green, Blue.
- ❖ Assign numeric value to each feature.
- ❖ 0 -> Red, 1 -> Green, 2 -> Blue



color
red
green
blue
red

color
0
1
2
0

Figure 6Nominal Data

Time Series Data

A Time Series is a sequence of data points (as in Figure 7) that occur in successive order over some period of time. The similar graphs can be seen in the stock markets, is also an example for this.

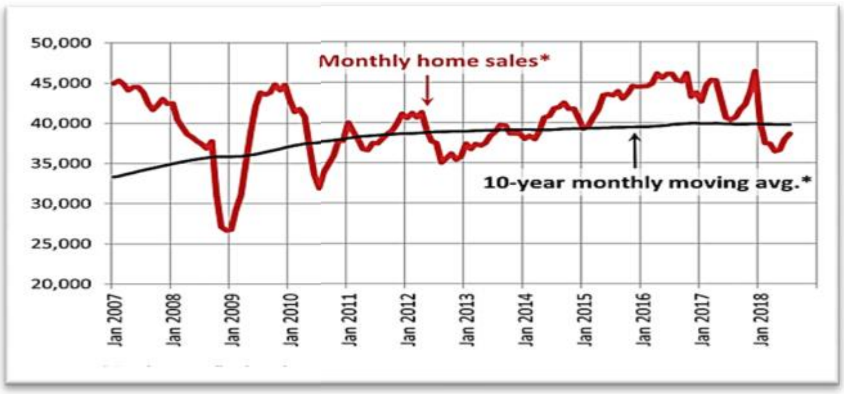


Figure 7Time Series Data

Text Data

Text data usually consists of documents, which can represent words, sentences or even paragraphs. Figure 8 describes the text data as it can be books, material, anything that is written or typed.

Figure 8 Text Data

12.4 Unsupervised Learning

Unsupervised learning is a kind of machine learning where a model must look for patterns in a given dataset with no labels marked in them. Figure 9 depicts the how human does the unsupervised learning. This avoids supervision / teacher.

- No external guidance in performing the task.
- It will identify patterns on its own from the given data sets, which are neither classified nor labelled.
- Pattern / Distance metric is used by Machine to group or classify the given data.

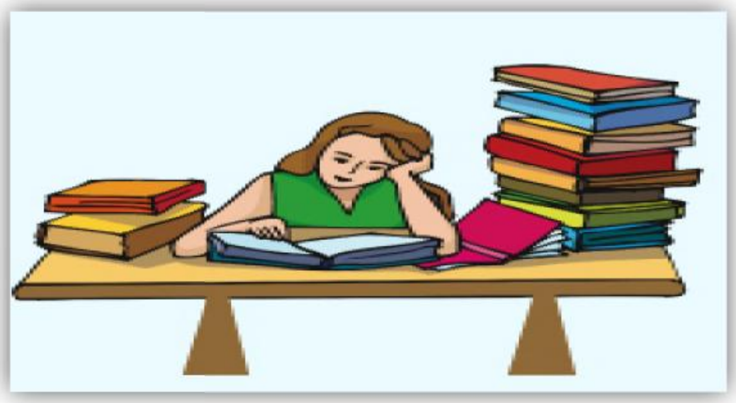


Figure 9Unsupervised Machine Learning

12.5 Inductive Learning

Inductive learning is a process where the learner discovers rules by observing examples. The inductive learning is also known as discovery learning. The Inductive learning is different from deductive learning, where students are given rules that they then need to apply.

We often work out rules for ourselves by observing examples to see if there is a pattern or not, to see if things regularly happen in the same way. We then try applying the rule in different situations to see if it works or not.

12.6 Rule based Learning

Rule-based learningis just another type of learning which makes the class decision depending by using various "if..else" rules. These rules are easily interpretable and thus these classifiers are generally used to generate descriptive models. The condition used with "if" is called the antecedent and the predicted class of each rule is called the consequent.

Either rules can be ordered, i.e. the class corresponding to the highest priority rule triggered is taken as the final class.

Otherwise, we can assign votes for each class depending on some of their weights, i.e. the rules remain unordered.

Actually, Rules-based systems are best suited to situations in which there are lower volumes of data and the rules are relatively simple

12.7 Reinforcement Learning

This is concerned with how software agents should take actions in an environment. Reinforcement learning approach helps you to maximize some portion of the cumulative reward. There are few terminologies to be remembered for this study as shown in Figure 10.

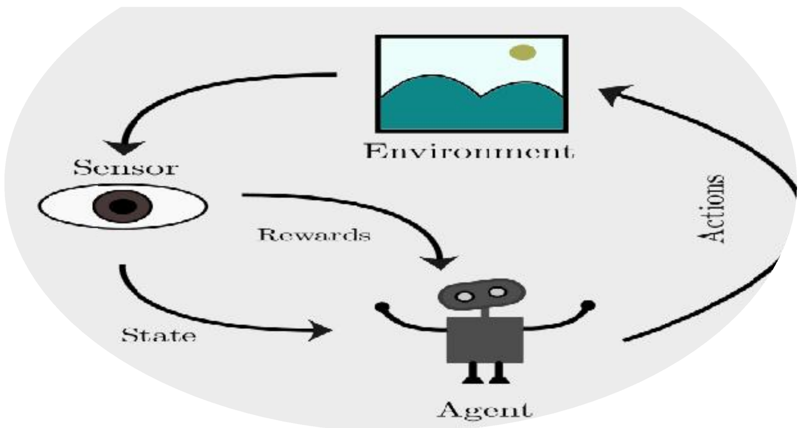


Figure 10Reinforcement Machine Learning

Agent

It is an assumed entity which performs actions in an environment to gain some reward.

Environment (e)

A scenario that an agent has to face.

Reward (R)

An immediate return given to an agent when he or she performs specific action or task.

State (s)

State refers to the current situation returned by the environment.

Policy (π)

It is a strategy which applies by the agent to decide the next action based on the current state.

Value (V)

It is expected long-term return with discount, as compared to the short-term reward.

Value Function

It specifies the value of a state that is the total amount of reward. It is an agent which should be expected beginning from that state.

Model of the environment

This mimics the behavior of the environment. It helps you to make inferences to be made and also determine how the environment will behave.

Model based methods

It is a method for solving reinforcement learning problems which use model-based methods.

Q value or action value (Q)

Q value is quite similar to value. The only difference between the two is that it takes an additional parameter as a current action.

To make this concept simple, let us take a situation in our house. Have a look at the given figure 11. The cat is an agent that is exposed to the environment. In this case, an example of a state could be the cat sitting, and you use a specific word for the cat to walk.

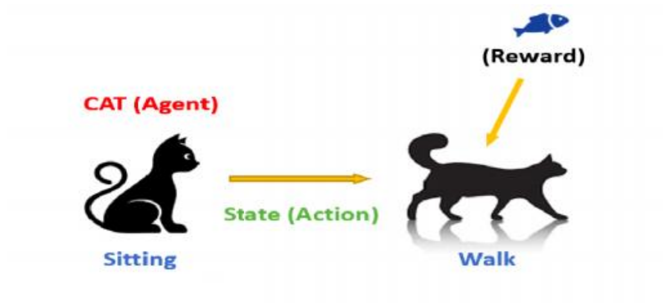


Figure 11 Example for Reinforcement learning

The cat agent reacts by performing an action transition from one “state” to another “state.”For example, the cat goes from sitting to walking.The reaction of an agent is an action, and the policy is a method of selecting an action given a state in expectation of better outcomes.After the transition, they may get a reward or penalty in return.

There are three approaches to implement a Reinforcement Learning algorithm. They are Value based, in which we should try to maximize a value function V(s). In this method, the agent is expecting a long-term return of the current states under policy π Second method is policy based, in which we develop a policy that the action is performed in every state helps you to gain maximum reward in the future. The last is Model based, in which we need to create a model for each environment and the agent learns to perform in that specific environment only.

Characteristics

- No prior experience.
- No perfect decision.
- Always trying to perform better every time.
- It is bound to learn from its own experience.

Types of Reinforcement Learning

There are two types of reinforcement learning methods, which are called positive and negative. Positive method is defining an agent that increases the positivity of the action, focusing only on positive rewards. On the other hand, negative method is defining an agent that removing the rewards given to that action, allowing the same action is repeated again for well training. Such as, A person learns to wear a raincoat, during the rainy season, to avoid getting wet and People wear helmets, to avoid getting injured, in case of a road accident, or getting fined by cops.Repeated actions created for learning.

Summary

In this unit, the concepts of machine learningare discussed along with the different approaches of machine learning. Each approach is discussed in detail with examples. The differences in each of the approaches would be better understood. Data set is very important for machine learning. Hence, it is necessary to understand about the basic data types, which is also explored thoroughly. This will help to convert or process the obtained data. But, there was also lot of challenges in processing the data set. This also covered in the name of preprocessing and data cleaning. The major tasks of preprocessing and the possible ways of data cleaning were also discussed. The terminology – feature engineering was highlighted as it was related to data cleaning.

Keywords

- Dataset
- Supervised learning
- Unsupervised learning
- Inductive learning
- Rule based learning
- Reinforcement learning

Self Assessment

1. Machine learning approach, which build a model based on sample data, is known as _______.

A. Supervised
B. Unsupervised
C. Reinforcement
D. None of the above

2. _________ approach uses the rewarding method for machine learning.

A. Supervised
B. Unsupervised
C. Reinforcement
D. None of the above

3.Which of the following dataset is used for supervised machine learning?

A. Training dataset
B. Testing dataset
C. Validation dataset
D. All the above

4. _________ machine learning approach uses unlabelled data for learning.

A. Supervised

B. Unsupervised
C. Reinforcement
D. None of the above

5. The two class problems are otherwise called _________.

A. Clustering
B. Binary Classification
C. Multiclass Classification
D. None of the above

6.Justify the statement. "Preprocessing is the process of converting raw data into data which will be suitable for machine learning".

A. True
B. False

7.Preprocessing is actually the combination of data cleaning and __________.

A. Data integration
B. Data transformation
C. Data reduction
D. All of the above

8.Supervised machine learning approachuses the ______ dataset.

A. Labelled Dataset
B. Unlabelled Dataset
C. Both Labelled and Unlabelled
D. None of the above

9. Imputation method is used for _____________.

A. Removing rows
B. Removing columns
C. Filling the missing values
D. None of the above

10. _________ is the process of changing the format, structure or values of data.

A. Data integration
B. Data cleaning
C. Data transformation
D. Data Preprocessing

11. Justify the given statement. "A rule-based system consists of a bunch of IF-THEN rules".

A True
B False

12. Rule based system also known as _____________.

A. Mycin based system
B. Human based system
C. Knowledge based system
D. None of the above

13. Which chaining rule has inductive approach?

A Forward Chaining
B Backward chaining
C Spanning Chaining
D All of the above

14. Reinforcement learning is _______________ learning.

A Supervised
B Unsupervised
C Award based
D None of the above

15. _________Reinforcement is defined as when an event, occurs due to a particular behavior.

A) Negative
B) Positive
C) Neutral
D) None of these

Answers for Self Assessment

1.	A	2.	C	3	D	4.	B	5.	B
6.	A	7.	D	8.	A	9.	C	10.	C
11.	A	12.	C	13.	B	14.	C	15.	B

Review Questions

1. Explain the different types of data.
2. Differentiate nominal and ordinal data types.
3. Give examples for categorical data.
4. List out the methods used for filling the missing values.
5. Identify the machine learning algorithms for each machine learning approaches.

Further readings

• S. N. Sivanandam, S.N. Deepa, Principles Of Soft Computing, Wiley Publications, Second Edition, 2011.

- Rajasekaran, S., Pai, G. A. Vijayalakshmi, Neural Networks, Fuzzy Logic and Genetic Algorithm Synthesis And Applications, Prentice Hall of India, 2013.
- N. P. Padhy, S. P. Simon, Soft Computing With Matlab Programming, Oxford University Press, 2015.

Web Links

- https://www.javatpoint.com/types-of-machine-learning
- https://www.geeksforgeeks.org/ml-types-learning-supervised-learning/
- https://towardsdatascience.com/feature-engineering-for-machine-learning-3a5e293a5114

Unit 13: Artificial Neural Networks

CONTENTS

Objectives

1. Understanding the processing of a neuron.
2. Understanding the Structure of Artificial Neural Networks.
3. Understanding the difference between Biological Neuron and Artificial Neuron.
4. Understanding the importance of Activation Functions.
5. Understanding the different data involved in training and testing.

Introduction

Artificial Neural Networks (ANNs) are relatively crude electronic models based on the neural structure of the brain. The brain learns from experience and stores in the cells as in Figure 1. Artificial neural networks try to mimic the functioning of brain.

Figure 1 Human Brain and Cells

Even simple animal brains are capable of functions that are currently impossible for computers. Computers do the things well, but they have trouble recognizing even simple patterns. This unit explores the technical aspects and makes you to understand the processing of neuron too along with ANN Architecture.

13.1 Biological Neuron and Structural Components

Basically, a biological neuron receives inputs from other sources, combines them in some way, performs a generally nonlinear operation on the result, and then outputs the final result. Fig. 2 shows the relationship of these four parts.

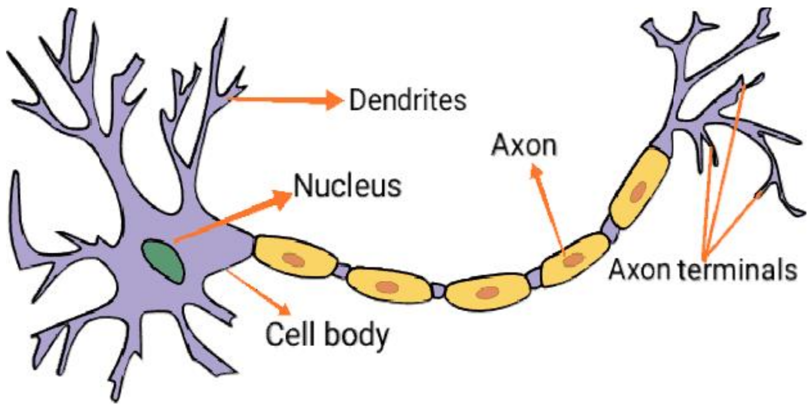


Figure 2Structure of Biological Neuron

Within humans there are many variations on basic type of neuron, yet, all biological neurons have the same four basic components. They are known by their biological names – cell body (soma), dendrites, axon, and synapses.

Cell body (Soma)

The body of neuron cell contains the nucleus and carries out biochemical transformation necessary to the life of neurons.

Dendrite

Each neuron has fine, hair like tubular structures (extensions) around it. They branch out into tree around the cell body. They accept incoming signals.

Axon

It is a long, thin, tubular structure which works like a transmission line. Synapse: Neurons are connected to one another in complex spatial arrangement. When axon reaches its final destination it branches again called as terminal arborization. At the end of axon are highly complex and specialized structures called synapses. Connection between two neurons takes place at these synapses.

Dendrites receive the input through the synapses of other neurons. The soma processes these incoming signals over time and converts that processed value into an output, which is sent out to other neurons through the axon and the synapses.

13.2 Artificial Neuron and its structure

An artificial neuron is a mathematical function conceived as a simple model of a real (biological) neuron. The artificial neuron simulates four basic functions of a biological neuron. Fig. 3 shows basic representation of an artificial neuron.

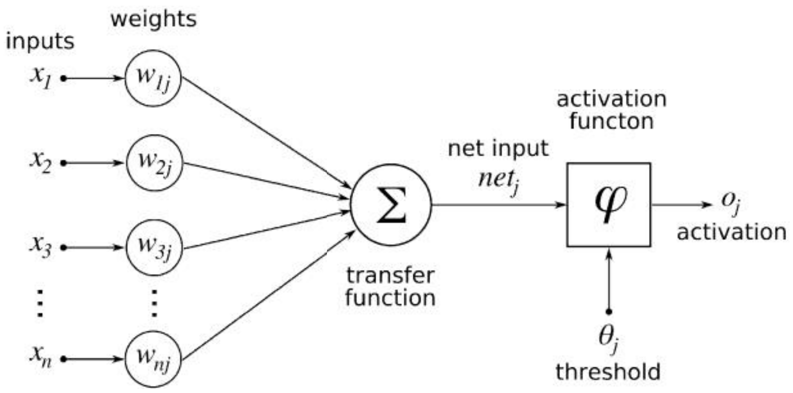


Fig 3. Structure of Artificial Neuron

In Fig. 3, various inputs to the network are represented by the mathematical symbol, x(n). Each of these inputs is multiplied by a connection weight. The weights are represented by w(n). In the simplest case, these products are summed, fed to a transfer function (activation function) to generate a result, and this result is sent as output. This is also possible with other network structures, which utilize different summing functions as well as different transfer functions.Some summation functions have an additional 'activation function' applied to the result before it is passed on to the transfer function for the purpose of allowing the summation output to vary with respect to time.

13.3 Historical Note

The historyis relevant because for nearly two decades the future of neural network remained uncertain.

- McCulloh and Pitts (1943) are generally recognized as the designers of the first neural network. They combined many simple processing units together that could lead to an overall increase in computational power. They suggested many ideas like, a neuron has a threshold level and once that level is reached, the neuron fires. It is still the fundamental way in which ANN operate. The McCulloh and Pitt's network had a fixed set of weights.
- Hebb (1949) developed the first learning rule, that is, if two neuron are active at the same time then the strength between them should be increased.
- In the 1950 and 1960's, many researchers (Block, Minsky, Papert and Rosenblatt) worked on perceptron. The neural network model could be proved to converge to the correct weights, that will solve the problem. The weight adjustment (learning algorithm) used in the perceptron was found more powerful than the learning rules used by Hebb. The perceptron caused great excitement. It was thought to produce programs that could think.
- Minsky and Papert (1969) showed that perceptron could not learn those functions, which are not linearly separable.

The neural networks research declined throughout the 1970 and unit mid 1980's because perceptron could not learn certain important functions.

Neural network regained importance in 1985-86. The reseachers, Parker and LeCun discovered a learning algorithm for multi-layer networks called back propagation that could solve problems that were not linearly separable.

13.4 Applications of ANN

Many of the networks being designed presently are statistically quite accurate (upto 85% to 90% accuracy). Currently, neural networks are not the user interface, which translates spoken words into instructions for a machine but some day it will be achieved. VCRs, home security systems, CD players, and word processors will simply be activated by voice. Touch screen and voice editing will replace the word processors of today while bringing spreadsheets and databases to a level of usability. Neural network design is progressing in other more promising application areas.

❖ Language Processing

  This enables machines to understand and respond to text or voice data.This helps to understand a document and also this helps to understand the speech.

❖ Character Recognition

  This helps to read character by character and understands which letter is written over there.

❖ Image (data) Compression

  This helps to reduce the size of the data or size of the image without reducing the meaning or content of the image.

❖ Pattern Recognition

  This helps to understand the given data and tries to look into their patterns. Patterns are the keys to classify the stored data into multiple categories according to different patterns.

13.5 Structure of Artificial Neural Networks

The structure of artificial neural networks is given below, for example, as in Fig. 4. Inputs enter into the processing element from the upper left.

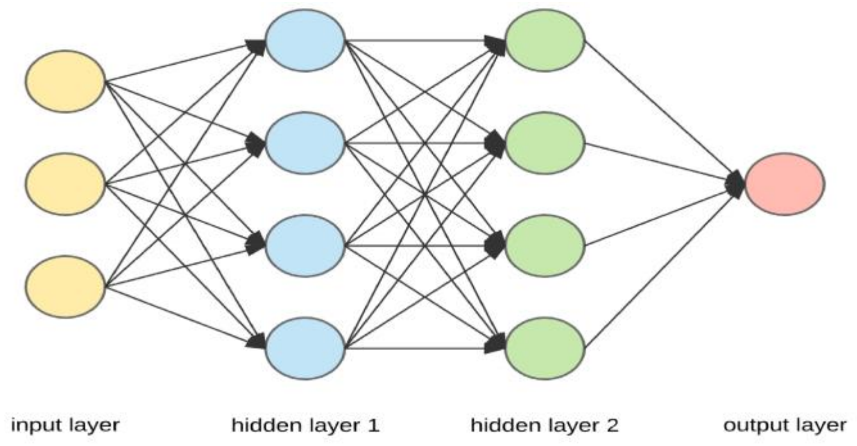


Figure 4 Artificial Neural Networks Architecture

The first step is to multiply each of these inputs by their respective weighting factor [w(n)]. These modified inputs are then fed into the summing function, which usually sums these products, however, many different types of operations can be selected. These operations can produce a number of different values, which are then propagated forward; values such as the average, the largest, the smallest, the ORed values, the ANDed values, etc. Other types of summing functions can also be created and sometimes they may be further complicated by the addition of an activation function which enables the summing function to operate in a time sensitive way.

The output of the summing function is then sent into a transfer function, which turns this number into a real output (a 0 or a 1, -1 or +1 or some other number) via some algorithm. The transfer function can also scale the output orcontrol its value via thresholds. This output is then sent to other processing elements or an outside connection, as dictated by the structure of the network.

Binary-Class Classification

The Figure 4 is describing the concept of binary class model. The output layer is having only one neuron illustrating the yes / no response. Output = 1 means Yes and the output = 0 means No. Hence, this model is known as binary-class classification.

Any data, which can be classified into two categories, is also known as binary-class classification as shown in Figure 5.

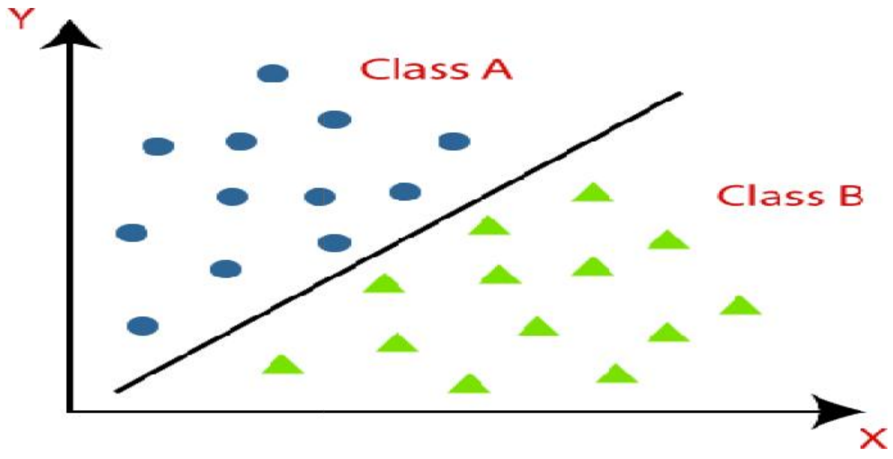


Figure 5 Binary-Class Classification

Multi-Class Classification Model

The Figure 6 is describing the concept of multi-class model. The output layer is having more than one neuron illustrating the yes / no response on each output neuron. Hence, this model is known as multi-class classification.

Any data, which can be classified into more than twocategories, is also known as multi-class classification as shown in Figure 7.

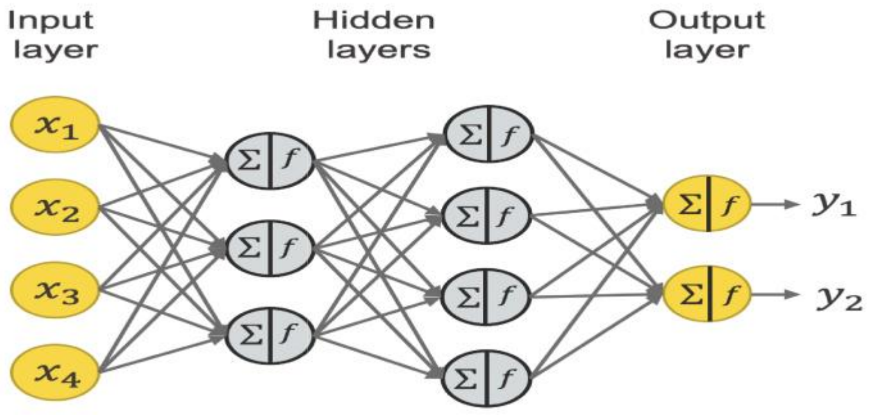


Fig. 6 Structure of Multi-Class ANN Classifier

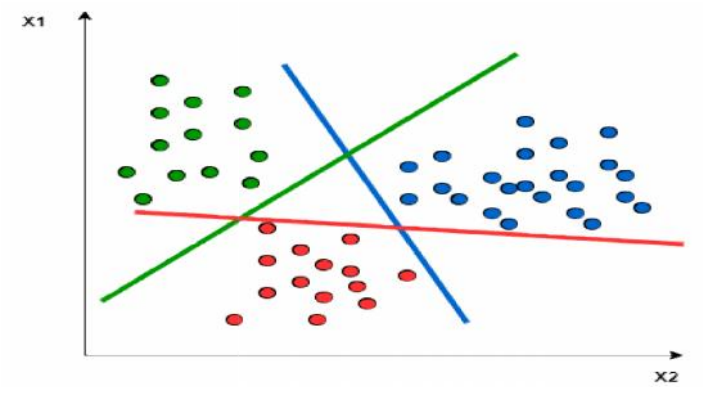
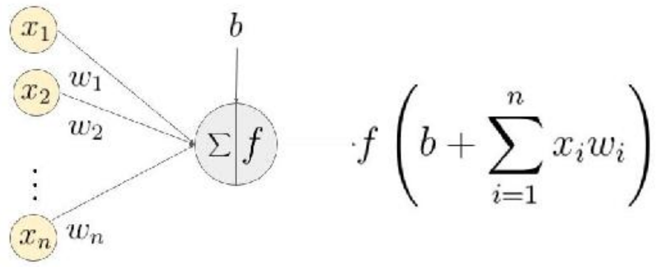


Fig. 7Multi-Class Classification

13.6 Processing of a Neuron

An artificial neuron is a connection point in an artificial neural network. Artificial neural networks, like the human body's biological neural network, have a layered architecture and each network node (connection point) has the capability to process input and forward output to other nodes in the network. This is shown in Fig. 8.



An example of a neuron showing the input (x_1 - x_n), their corresponding weights (w_1 - w_n), a bias (b) and the activation function f applied to the weighted sum of the inputs.

Fig. 8. Processing of a Neuron

These details are explained here with different components.

Component 1. Weighting Factors: A neuron usually receives many simultaneous inputs. Each input has its own relative weight, which gives the input the impact that it needs on the processing element's summation function. Some inputs are made more important than others to have a greater effect on the processing element as they combine to produce a neural response. Weights are adaptive coefficients that determine the intensity of the input signal as registered by the artificial neuron. They are a measure of an input's connection strength. These strengths can be modified in response to various training sets and according to a network's specific topology or its learning rules.

Component 2. Summation Function: The inputs and corresponding weights are vectors which can be represented as (i1, i2 . . . in) and (w1, w2 . . . wn). The total input signal is the dot product of these two vectors. The result; (i1 * w1) + (i2 * w2) +. ... + (in * wn) ; is a single number.

Component 3. Transfer Function: The result of the summation function is transformed to a working output through an algorithmic process known as the transfer function. In the transfer function the summation can be compared with some threshold to determine the neural output. If the sum is greater than the threshold value, the processing element generates a signal and if it is less than the threshold, no signal (or some inhibitory signal) is generated. Both types of response are significant. The threshold, or transfer function, is generally non-linear. Linear functions are limited because the output is simply proportional to the input.

Component 4. Scaling and Limiting: After the transfer function, the result can pass through additional processes, which scale and limit. This scaling simply multiplies a scale factor times the transfer value and then adds an offset. Limiting is the mechanism, which insures that the scaled result does not exceed an upper, or lower bound. This limiting is in addition to the hard limits that the original transfer function may have performed.

Component 5. Output Function (Competition): Each processing element is allowed one output signal, which it may give to hundreds of other neurons. Normally, the output is directly equivalent to the transfer function's result. Some network topologies modify the transfer result to incorporate competition among neighboring processing elements. Neurons are allowed to compete with each other inhibiting processing elements unless they have great strength. Competition can occur at one or both levels. First, competition determines which artificial neuron will be active or provides an output. Second, competitive inputs help determine which processing element will participate in the learning or adaptation process.

Component 6. Error Function and Back-Propagated Value: In most learning networks the difference between the current output and the desired output is calculated as an error which is then transformed by the error function to match a particular network architecture. Most basic architectures use this error directly but some square the error while retaining its sign, some cube the error, other paradigms modify the error to fit their specific purposes. The error is propagated backwards to a previous layer. This back-propagated value can be either the error, the error scaled in some manner (often by the derivative of the transfer function) or some other desired output depending on the network type. Normally, this back-propagated value, after being scaled by the learning function, is multiplied against each of the incoming connection weights to modify them before the next learning cycle.

Component 7. Learning Function: Its purpose is to modify the weights on the inputs of each processing element according to some neural based algorithm.

13.7 Activation Function

An Activation Function decides whether a neuron should be activated or not. This means that it will decide whether the neuron's input to the network is important or not in the process of prediction using simpler mathematical operations.The role of the Activation Function is to derive output from a set of input values fed to a node (or a layer).

There are three types of activation functions available. They are, binary step activation function, linear activation function and non-linear activation function.

Binary step activation function

Binary step activation function as in Fig. 9 depends on a threshold value that decides whether a neuron should be activated or not. The input fed to the activation function is compared to a certain threshold; if the input is greater than it, then the neuron is activated, else it is deactivated, meaning that its output is not passed on to the next hidden layer.

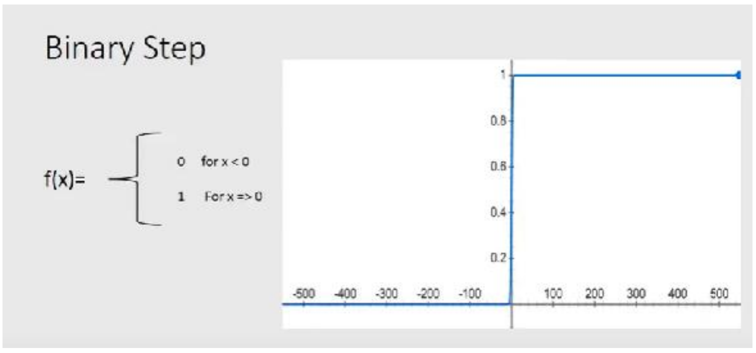


Fig. 9 Binary step activation function

Linear activation function

The linear activation function as in Fig. 10, also known as "no activation," or "identity function" (multiplied x1. 0), is where the activation is proportional to the input. The function doesn't do anything to the weighted sum of the input, it simply spits out the value it was given.

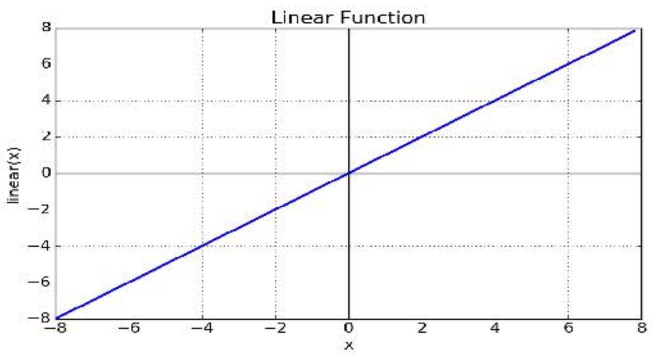


Fig. 10 Linear Activation Function

Non-linear activation functions

The non-linear functions are known to be the most used activation functions. It makes it easy for a neural network model to adapt with a variety of data and to differentiate between the outcomes. These functions are mainly divided basis on their range or curves:

- ❖ Sigmoid Activation Functions
- ❖ Tanh Activation Functions
- ❖ ReLU Activation Functions
- ❖ Leaky Relu
- ❖ Softmax Activation Function

13.8 Types of Data Involved

Training data and test data are two important types of data used in Artificial Neural Networks. In addition to this, one more type of data is needed for betterment of the ANN Architecture, is known as Validation Data.

- ❖ Training data (or a training dataset) is the initial data used to train the ANN.
- ❖ Testing Datais used to testthe trained ANN and evaluate its accuracy.

13.9 Python Tools

There are many libraries in python to help working with Artificial neural networks, which are listed below.

- ❖ TensorFlow
- ❖ PyTorch
- ❖ NeuroLab
- ❖ Ffnet
- ❖ pyrenn
- ❖ Scikit-Learn

Out of the above long list, we prefer to implement in Scikit-learn, as it is simple to use.

Summary

This unit explored the basic concepts of Artificial Neural Networks, stating from what is biological neuron. Understood that the imitation of biological neuron became artificial neuron. The processing of a artificial neuron was explained clearly using a diagram. The Structure of Artificial Neural Networks was discussed in detail.Understood the difference between Biological Neuron and

Artificial Neuron.Importance of Activation Functions and different types of activation function was explained in this unit.Training and testing data were highlighted.

Keywords

- Biological Neuron
- Artificial Neuron
- Artificial Neural Networks
- Activation Function
- Binary classification
- Multi-class classification

Self Assessment

Q1) How many hidden layers can be present in a multi layer neural network?

A. 0
B. 1
C. 2
D. 'N'

Q2) The fundamental unit of network is _______________.

A. Brain
B. Nucleus
C. Neuron
D. Axon

Q3) What are dendrites?

A. Fibers of nerves
B. Nuclear projections
C. Other name for nucleus
D. None of the above

Q4) What is an activation value?

A. Threshold value
B. Weighted sum of inputs
C. Main input to neuron
D. None of the above

Q5) The process of adjusting the weight is known as ___________.

A. Activation
B. Synchronization
C. Learning
D. None of the above

Q6) Which is true for artificial neural networks?

A. It has set of nodes and connections
B. Each node computes it's weighted input
C. Node could be in excited state or non-excited state
D. All of the above

Q7) Artificial Neural Networks can be used in __________ fields.

A. Classification
B. Data Processing
C. Compression
D. All of the above

Q8) What is called if the output layer is having only one neuron?

A. Zero Classification
B. Binary Classification
C. Multi-class Classification
D. All the above

Q9) ______ is the range of output from Sigmoid Activation Function.

A. 0 to 1
B. -1 to 0
C. -1 to +1
D. None of the abvoe

Q10) The first learning rule is developed by ___________.

A. McCulloh
B. Pitss
C. Hebb
D. Minsky and Papert

Q11) _________ layer receives the input from the user.

A. Input Layer
B. Hidden Layer
C. Output Layer
D. All of the above

Q12) Justify the given statement.

"All the neurons exists in Artificial Neural Networks is processing the input and transfers the output to next layer."

A. Yes
B. No

Q13) Linear activation function is also known as ______________.

A. Identity function
B. No activation function
C. Option (A) and (B)
D. None of These

Q14) Tangent Hyperbolic Activation function values ranging from ________ to ________.

A. 0 to 1
B. -1 to +1
C. -1 to 0
D. None of the above

Q15) Artificial Neural Network architecture is optimized using ____________ after completing the training of the model.

A. Training Dataset
B. Testing Dataset
C. Validation Dataset
D. Computer vision

Answers for SelfAssessment

1.	D	2.	C	3	A	4.	B	5.	C
6.	D	7.	D	8.	B	9.	A	10.	C
11.	A	12.	B	13.	C	14.	B	15.	C

Review Questions

1. Explain the architecture of Artificial Neural Networks.
2. List the various tools used to implement ANN.
3. What are all the activation functions used for training ANN?
4. Give an example how the weights are adjusted.
5. Differenciate biological neuron and artificial neuron.

Further readings

- S. N. Sivanandam, S.N. Deepa, Principles Of Soft Computing, Wiley Publications, Second Edition, 2011.
- Rajasekaran, S., Pai, G. A. Vijayalakshmi, Neural Networks, Fuzzy Logic and Genetic Algorithm Synthesis And Applications, Prentice Hall of India, 2013.
- N. P. Padhy, S. P. Simon, Soft Computing WithMatlab Programming, Oxford University Press, 2015.

Web Links

- https://www.javatpoint.com/artificial-neural-network
- https://www.analyticsvidhya.com/blog/2021/05/beginners-guide-to-artificial-neural-network/
- https://www.techopedia.com/definition/5967/artificial-neural-network-ann

Unit 14: Natural Language Processing

CONTENTS

Objectives

1. Understanding the different components of NLP.
2. Understanding the NLP Phases.
3. Understanding the difference among various ambiguities.
4. Understanding the importance of parsing and parse trees.
5. Understanding the applications of NLP.

Introduction

Natural language processing (NLP) is a branch of artificial intelligence that helps computers to understand human language. It helps to improve the communication between the human and the machine. The application of NLP is very familiar to us, for example, Email Spam filters, Google Translators, Chatbot Applications in various domains, Personal Assistant Applications in Mobiles and Sentiment Analysis on Products / Persons. In this unit, we are going to discuss the basic concepts of NLP and different phases of NLP in detail.

14.1 Problem Types

The NLP problem is attempted in two different ways, with respect to the inputs. They are Written Form and Audio Form. Meaning that, if our NLP handles the written documents as the input is known as written form and in case, if the NLP handles the audio as the input is known as audio form. The outcome of the NLP problem is coming at the same point, i.e., just to understand what is written or what is spoken. Hence, any problem in the field of NLP will be categorized into two as mentioned above.

The application is the speech recognition when we are handling the speech of a person and trying to understand what they are communicating as in Figure 1. Similarly, the process is called document recognition when the input becomes text.

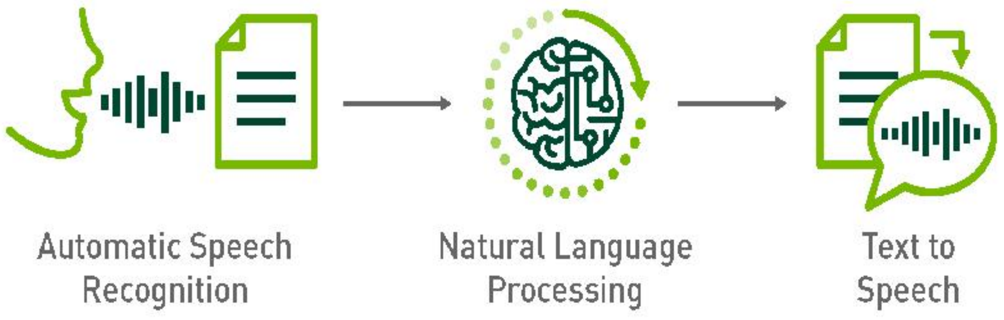


Figure 1 Speech Recognition using NLP

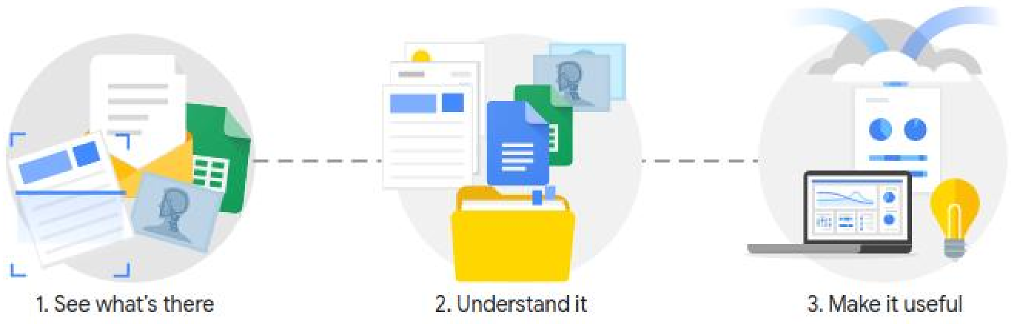


Figure 2Document Recognition using NLP

14.2 Components

NLP is having two components as shown in Figure 3. They are Natural Language Understanding and Natural Language Generation. They are represented in short as NLU and NLG respectively. The first component is responsible to read and understand what is written or what is spoken. The component will be doing all the operations or processes in order to achieve the understanding. Then the second component, Natural Language Generation, will get the input from the first component (NLU) and translates into some other human language as per the requirements. Hence, the NLG is will process all the necessary functionalities in order to translate the content with the exact meaning. Both the components can be visualized into one when you imagine the "Google translator" as an example.

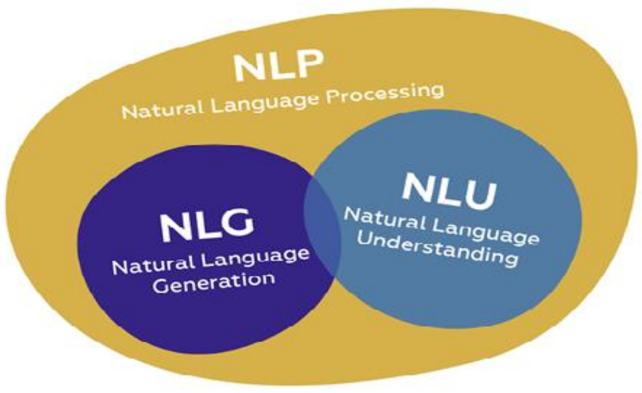

NLP
Natural Language Processing
NLG
Natural Language Generation
NLU
Natural Language Understanding

Figure 3 NLP Components

14.3 Phases of NLP

There are five phases in natural language processing as shown in Figure 4. They are lexical analysis, semantic analysis, semantic analysis, discourse integration and pragmatic analysis. We will discuss them one by one in detail.

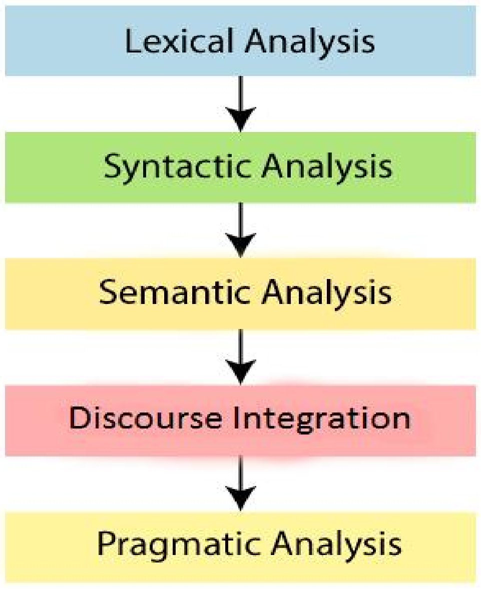


Figure 4 NLP Phases

Lexical analysis

This phase divide the content into word by word, known as lexical. Here is an example like "I am giving you a material on NLP". This phase will give the tokens as {I, am, giving, you, a, material, on, NLP}, having total number of Tokens = 8. The process of separating the words is known as Tokenization.

Syntactic analysis

This phase is primarily checking for the grammar, word arrangements and the relationship among the words.This process is called Parsing. All the errors are corrected and sent to the next phase. The errors may be understood from the given two sentences. First sentence "I ate an apple" is correct as per the language rules or grammars. But, the sentence "I an apple ate" is not correct as per the language rules or grammars. This phase will check these syntaxes before proceeding to the next phase.

Semantic Analysis

Statements may be correct grammatically. But, it may not be correct as per the meaning. Consider the statement "Horse is going to Factory". It is correct as per the grammar rules, meaning that it is as per the syntax. We can also understand the meaning of the statement. It is okay. But, if you consider an another statement "Factory is going to Horse", you may feel the change of meaning but it is correct as per the syntax. Such, unrealistic statements are detected and ignored from processing.Rest of the statements will be sent for the next phase.

Discourse integration

Usually, all the statements may not have complete information. Some content may be missing. We are integrating / joining the related statements together to understand them better. The process of joining the related statements is known as discourse integration. Hence, the missing information can be retrieved. Consider the given three statements as "Horse ran up the hill", "It was very steep" and "It soon got tired". Here, we unable to understand what does "It" refer in the second and third statements. You need to identify the related sentences and combine them in order to get the meaning of "It". This process is called discourse integration.

Pragmatic analysis

This phase tries to characterize the statements.Statements may be of any kind, for example it can be a request statement, may be an ordering statement, pleasing statement, statement showing love, angry statement, suspicious, with interest, hate and etc.This phase attempts to detect these emotions coming out of the statement. It helps in effective translation.

14.4 Ambiguity and Uncertainty

There are many challenges in the Natural Language Processing. Few challenges are discussed here, which helps us to understand the problem better. We start with a statement as "Few Horses are standing outside".This statement is not having complete information in it because we are unable to locate the place where the horses are standing. The word "outside" refers what? This will create incompleteness – the first challenge in NLP.

Consider one more statement as "Alan is poor". "Alan" is the name of the person. What the word "poor" refers?. We are unable to justify actual meaning of "poor" because this word may be giving different meaning with different context of readers. This is the second challenge in NLP.

List of ambiguities

- Lexical Ambiguity
- Syntactic Ambiguity
- Referential Ambiguity

Lexical Ambiguity

Ambiguity occurs because of the confusion from a word, hence it is called Lexical Ambiguity. You can understand this from a statement "Manya is looking for a match". Is manya is looking a cricket match ? or Ismanya is looking for a boy friend ?. We do not understand what is actually referred by match. This ambiguity is caused by a improper word.

Syntactic Ambiguity

Next, we see the ambiguity caused by a sentence level. Particularly the order of words is creating the confusion. This is known asSyntactic Ambiguity. You can understand from a statement "I saw the girl with the binocular". What is the actual meaning of the given statement? Once you understand, answer the question "who is having binocular?." Whether the binocular is with the girl or with the person who see the girl. This ambiguity is caused from a formation of words or order of words used to create this sentence. Hence this is called syntactic ambiguity.

Referential Ambiguity

The third type of ambiguity is called as Referential Ambiguity. Consider the statement "Kiran went to Sunita. She said, "I am hungry". There are two sentences mentioned. First sentence is clear that the first person has gone to meet the second person. Look at the second statement. Ambiguity arises to understand who is hungry, Kiran or Sunita. What "She" refers to?. This ambiguity is known as referential ambiguity.

14.5 Challenges in Translation

Usually the language "English" is used as the medium for all the NLP operations. Much research has been done on English to understand the written form or audio form. The ambiguity discussed above is considering English statements. There are presently more challenges have been faced by our researchers and student communities. Here are few for your consideration.

- The usage of multiple languages in a single sentence. Even documents may have few words from other languages. This Mixed languages becomes a challenge, as NLP needs to be aware of all the required languages.
- Processing the Research articles for translation, you have a challenge on conversion of equations or formula or mathematical derivations.
- Now people have started to use their native languages on social media and creating the content without following the grammars of the language. Writing the content in very short form. NLP need to handle this kind of sentences also.

14.6 Parsing

Parsing is a process used to check the given statement whether it is written correctly as per the syntax / grammar rules or not. This parsing is actually looking like a grammar expert. It knows all the grammar rules. It checks any kind of sentences. It knows all the verbs, adjectives, nouns, countable, uncountable, determiners, and etc. Parsing does not worry about the meaning of the statement.

Consider these two statements "I am riding a motor bike" and "Motor bike is riding me". Both the statements are correct as per the parser because they are correct as per the grammar rules.

14.7 Parse Tree

The tree structure, which is used for parsing, is called parse tree.This will help in syntactic analysis phase.Parsing will receive the input all the tokens from the first phase of NLP i.e., Lexical Analysis. The Figure 5 shown below is the parse tree created for a simple statement "I Studied". The figure 6 shown below is the parse tree created for a statement "I ate the apple". The original statement is always mentioned or symbolized as S.

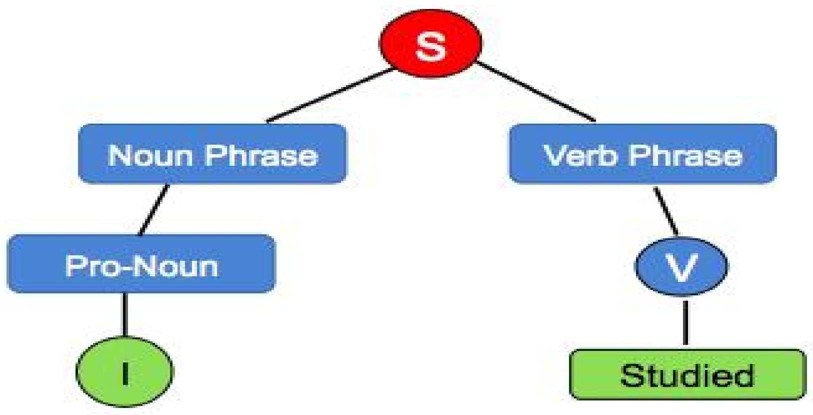


Figure 5 Parse Tree – Example 1

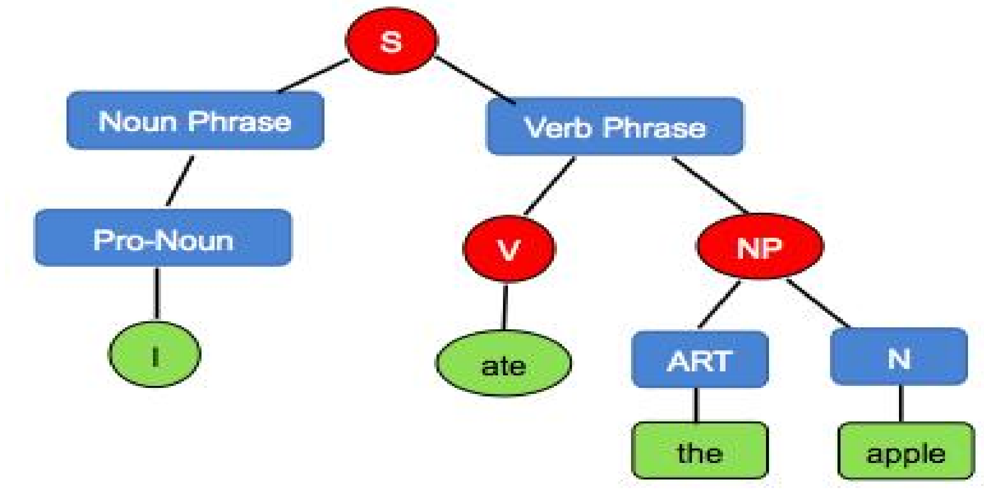


Figure 6 Parse Tree – Example 2

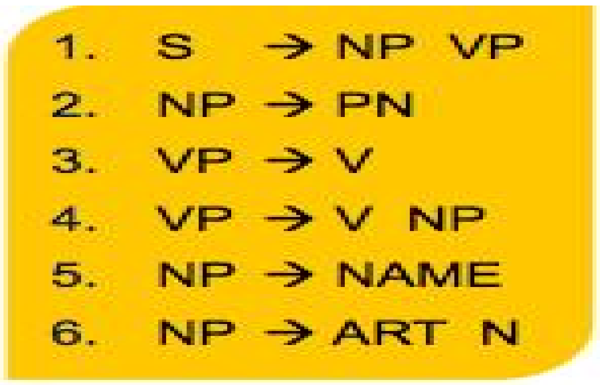
14.8 Parsing Components

The Parser should be having the thorough knowledge about the vocabulary used in the specified language, the perfect knowledge about the grammar rules and the list of possible words that comes in different category of words say verb, noun and etc. The above three are called as the components for the parser. The name of the components is lexicon, grammar rules and categorization.

List of components

- **Lexicon**
- **Grammar rules**
- **Categorization**

Lexicon is the vocabulary of a language. It should have all the words used in the language. Categorization will collect all the words under different categories i.e., Noun, Pronoun, Verb, Preposition, Adverb, Adjectives, Determiners, Conjunctions and etc. The rules used to check all the statements is called grammar rules. The format of grammar rules will be looking like as shown in the figure 7.



```
1.  S   → NP  VP
2.  NP  → PN
3.  VP  → V
4.  VP  → V  NP
5.  NP  → NAME
6.  NP  → ART  N
```

Figure 7 Grammar Rules

14.9 Types of Parsing

Parsing can be done in two different ways. They are known as Top-Down Parsing and Bottom-Up Parsing. The figure 8 is depicting the two different ways clearly. In particular, top down parsing is focusing on dividing the given sentence as per the grammar rules until no further divisions are possible. Here, all the words will be having their own places in the grammar rules. Hence, we can conclude that the given sentence is grammatically correct. But, if any words are missing or not finding any suitable places or unable to fit the missed out words into the grammar rules, then we can say that given sentence is not grammatically correct. In the context of Bottom up parsing, the parsing starts with the words and trying to frame possible sentences with the help of grammar rules. If any one of the sentences is as same as the given sentence, then we can say that given sentence is grammatically correct.

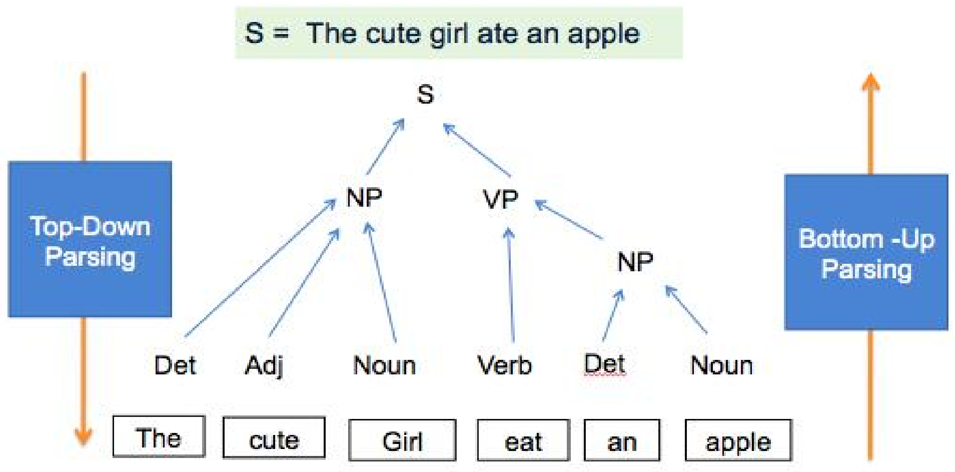


Figure 8 Top down and Bottom Up Parsing

14.10 Python Tools

NLTK and Spacy are the two popular tools used for Natural language processing. Here are few commands "pip install nltk" to install, "import nltk" to make use of all the libraries. Similary for Spacy, use the command "pip install spacy" to install and "import spacy" to make use of.

Summary

This unit attempted to give basics of natural language processing, starting from problem categories, say written and speech based. This also discussed the NLG and NLU, the components of NLP. Any problem should go through the five Phases of NLP, which are discussed with examples. Particularly, usage of parse tree is discussed in detail, which are helpful during the syntactic analysis. Also, the Top down and bottom up parsing are discussed which are the two different ways of performing parsing process.Components of parsing are very important, which consists of lexicon, grammar rules and categorization. This unit attempted to give all the fundamentals of NLP and the parsing.

Keywords

- Parsing
- Natural language processing
- Phases
- Ambiguity
- Parsetree
- Grammarrules

Self Assessment

Q1) What is the field of Natural Language Processing?

A. Computer Science
B. Artificial Intelligence
C. Linguistics
D. All of the mentioned

Q2) What is the main challenge/s of NLP?

A. Handling Ambiguity of Sentences
B. Handling Tokenization
C. Handling POS-Tagging
D. All of the mentioned

Q3) What is Machine Translation?

A. Converts one human language to another
B. Converts human language to machine language
C. Converts any human language to English
D. Converts Machine language to human language

Q4) How many steps of NLP is there?

A. 3
B. 4
C. 5
D. 6

Q5) ″I saw bats″ contains which type of ambiguity?

A. Syntactic
B. Lexical
C. Semantic
D. Anaphoric

Q6) ″Sita loves her mother and Gita does too″ contain which type of ambiguity?

A. Syntactic
B. Semantic
C. Lexical
D. Anaphoric

Q7) What is full form of NLG?

A. Natural Language Genes
B. Natural Language Growth
C. Natural Language Generator
D. Natural Language Generation

Q8) NLTK Stands for __________.

A. Numerical Level toolkit.
B. Neural Language Toolkit
C. Natural Language Toolkit
D. None of these

Q9) ____________ also converts a word to its root form

A. Rooting
B. Dreaming
C. Steaming
D. Lemmatization

Q10) NLP breaks down language into smaller, more basis pieces called ___________.

A. Parameters
B. Tokens
C. None
D. Arguments

Q11) _________ focuses about the proper ordering of words which can affect its meaning.

A. Syntax Analysis
B. Semantic Analysis
C. Lexical Analysis
D. Pragmatic Analysis

Q12) The words are transformed into the structure to show how the words are related to each other.

This process is called as ____________ .

A. Semantic Analysis
B. Lexical Analysis
C. Pragmatic Analysis
D. Syntax Analysis

Q13) when the meaning of the words themselves can be misinterpreted then ________ ambiguity occurs.

A. Semantic Ambiguity
B. Scope Ambiguity
C. Pragmatic Ambiguity
D. None of These

Q14) ______________ is a lexical database for the English language.

A. Corpus
B. WordNet
C. Lexicon
D. None of the above

Q15) NLP does not involve in ___________.

A. Speech recognition
B. Language understanding
C. Language generation
D. Computer vision

Answers for Self Assessment

1.	D	2.	A	3	A	4.	C	5.	B
6.	B	7.	D	8.	C	9.	D	10.	B
11.	A	12.	D	13.	A	14.	B	15.	D

Review Questions

1. Explain the different problem categories with examples.
2. List the phases of NLP.
3. What are all the possible challenges that come during translation?
4. Give an example for Discourse Integration.
5. Illustrate the importance of handling ambiguity in NLP.

Further readings

- S. N. Sivanandam, S.N. Deepa, Principles Of Soft Computing, Wiley Publications, Second Edition, 2011.
- Rajasekaran, S., Pai, G. A. Vijayalakshmi, Neural Networks, Fuzzy Logic and Genetic Algorithm Synthesis And Applications, Prentice Hall of India, 2013.
- N. P. Padhy, S. P. Simon, Soft Computing WithMatlab Programming, Oxford University Press, 2015.

Web Links

- https://towardsdatascience.com/your-guide-to-natural-language-processing-nlp-48ea2511f6e1
- https://www.tutorialspoint.com/natural_language_processing/index.htm
- https://www.javatpoint.com/nlp