ARTIFICIAL INTELLIGENCE

UNIT 1

What is AI?

Artificial Intelligence is a way of **making a computer, a computer-controlled robot, or a software think intelligently**, in the similar manner the intelligent humans think.

Thus, the development of AI started with the intention of creating similar intelligence in machines that we find and regard high in humans.

Importance of AI

1. In the field of Medical Sciences:
    - Artificial intelligence has been creating a virtual care private assistant which is specifically built for people's needs and is widely used for monitoring researching different types of cases and for analysis over the past cases and their outcomes.
    - Use of healthcare bots to provide 24/7 assistance and take up the less important work of managing appointments.
2. In the Field of Air Transport
    - Machine is involved in planning the routes, along with flight landing and take-off charts.
    - Quick check of the entire cockpit panel to ensure the correct working of every component has been performed with the use of artificial intelligence in many aircraft.

3. In the field of Banking and Financial Institutions

- Anti-money laundering: where the suspicious financial transactions are being monitored and are reported to the regulators.
- Credit systems analysis which are popular among credit card companies where the suspicious credit card transactions are tracked on the geographic level and based on various parameters is worked upon and resolved.

4. In the Field of Gaming and Entertainment

- Virtual reality games to the modern games which are there today, this is one industry where artificial intelligence has taken the biggest leap. The bots are always present to play with you and therefore you are not required to have a second person to play.
- The level of personal details and the graphics are also possible due to the advent of

  artificial intelligence and is taking this industry on a different level.

Fields of AI

1. Machine learning : Machine Learning is the science that enables machines to translate, execute and investigate data for solving real-world problems.

2. Neural Network : Incorporating cognitive science and machines to perform tasks, the neural network is a branch of artificial intelligence that makes use of neurology (a part of biology that concerns the nerve and nervous system of the human brain).

3. Robotics: Robotics is an interdisciplinary field of science and engineering incorporated with mechanical engineering, electrical engineering, computer science, and many others.

4. Expert Systems: An expert system refers to a computer system that mimics the decision-making intelligence of a human expert.

5. Fuzzy Logic: Fuzzy logic is a technique that represents and modifies uncertain information by measuring the degree to which the hypothesis is correct.

6. Natural Language Processing: Natural language processing depicts the developing methods that assist in communicating with machines using human languages such as English.

AI Techniques

1. Heuristics: Involves or serves as an aid to learning, discovery, or problem-solving by experimental and especially trial and error methods. In practice, this means that whenever problems get too complex to find the guaranteed best possible solution using exact methods, Heuristics serves to employ a practical method for finding a solution that is not guaranteed to be optimal, but one that is sufficient for the immediate goals.

2. Support Vector Machines

3. Artificial Neural Networks: Artificial Neural Networks (ANN) can be described as processing devices that are loosely modeled after the neural structure of a brain. The biggest difference between the two is that the ANN might have hundreds or thousands of neurons, whereas the neural structure of an animal or human brain has billions.

4. Markov Decision Process: A Markov Decision Process (MDP) is a framework for decision-making modeling where in some situations the outcome is partly random and partly based on the input of the decision maker.

5. Natural Language Processing: Natural Language Processing (NLP) is used to refer to everything from speech recognition to language generation, each requiring different techniques.

Problems

In AI, we will formally define a problem as

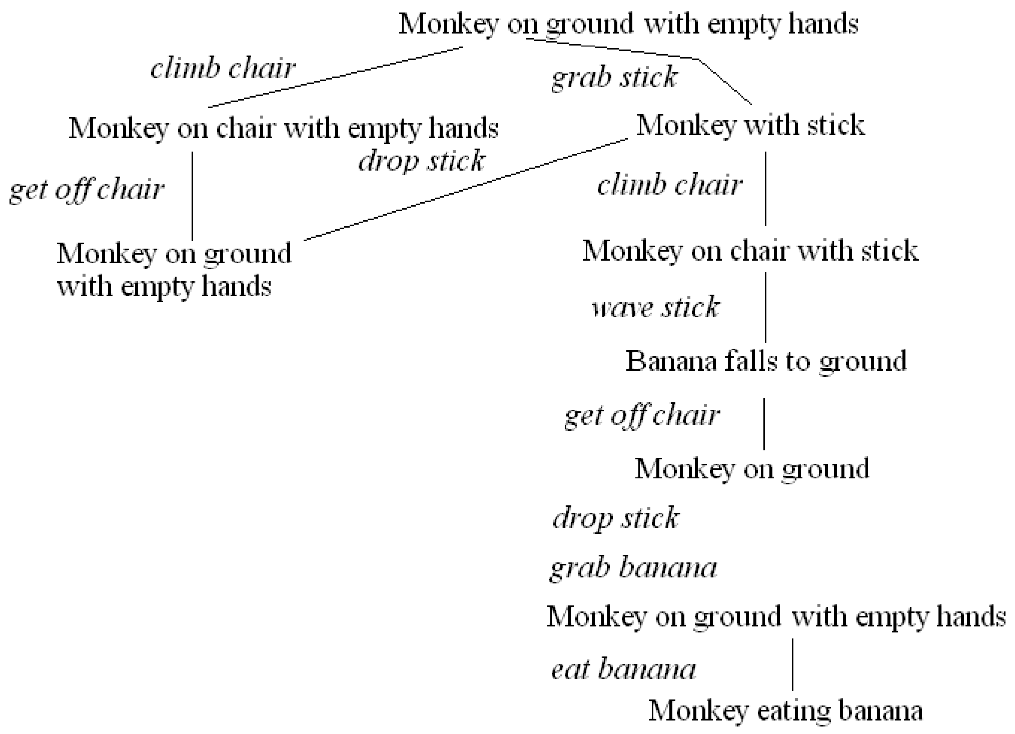
- – a space of all possible configurations where each configuration is called a state
    - thus, we use the term state space
- – an initial state
- – one or more goal states
- – a set of rules/operators which move the problem from one state to the next

Example of Problem solving:

The Monkey & Bananas Problem

- A monkey is in a cage and bananas are suspended from the ceiling, the monkey wants to eat a banana but cannot reach them
    - – in the room are a chair and a stick
    - – if the monkey stands on the chair and waves the stick, he can knock a banana down to eat it

What are the actions the monkey should take?



Initial state:

monkey on ground

with empty hand

bananas suspended

Goal state:

monkey eating

Actions:

climb chair/get off

grab

wave

eat

Production System and Its characterstics

Production system or production rule system is a computer program typically used to provide some form of artificial intelligence, which consists primarily of a set of rules about behavior but it also includes the mechanism necessary to follow those rules as the system responds to states of the world.

Characterstics:

1. Simplicity: The structure of each sentence in a production system is unique and uniform as they use the "IF-THEN" structure. This structure provides simplicity in knowledge representation. This feature of the production system improves the readability of production rules.

2. Modularity: This means the production rule code the knowledge available in discrete pieces. Information can be treated as a collection of independent facts which may be added or deleted from the system with essentially no deleterious side effects.

3. Modifiability: This means the facility for modifying rules. It allows the development of production rules in a skeletal form first and then it is accurate to suit a specific application.

4. Knowledge-intensive: The knowledge base of the production system stores pure knowledge. This part does not contain any type of control or programming information. Each production rule is normally written as an English sentence; the problem of semantics is solved by the very structure of the representation.

Issues in the design of the search problem

- The direction in which to conduct the search (forward or backward reasoning).
- How to select applicable(matching) rules?
- How to represent each node of the search process(knowledge representation problem)

Heuristic Search Technique

A Heuristic is a technique to solve a problem faster than classic methods, or to find an approximate solution when classic methods cannot. This is a kind of a shortcut as we often trade one of optimality, completeness, accuracy, or precision for speed. A Heuristic (or a heuristic function) takes a look at search algorithms. At each branching step, it evaluates the available information and makes a decision on which branch to follow. It does so by ranking alternatives.

1. Direct Heuristic Search Techniques in AI

Other names for these are Blind Search, Uninformed Search, and Blind Control Strategy. These aren't always possible since they demand much time or memory. They search the entire state space for a solution and use an arbitrary ordering of operations. Examples of these are Breadth First Search (BFS) and Depth First Search (DFS).

2. Weak Heuristic Search Techniques in AI

Other names for these are Informed Search, Heuristic Search, and Heuristic Control Strategy. These are effective if applied correctly to the right types of tasks and usually demand domain-specific information. We need this extra information to compute preference among child nodes to explore and expand. Each node has a heuristic function associated with it. Examples are Best First Search (BFS) and A*.

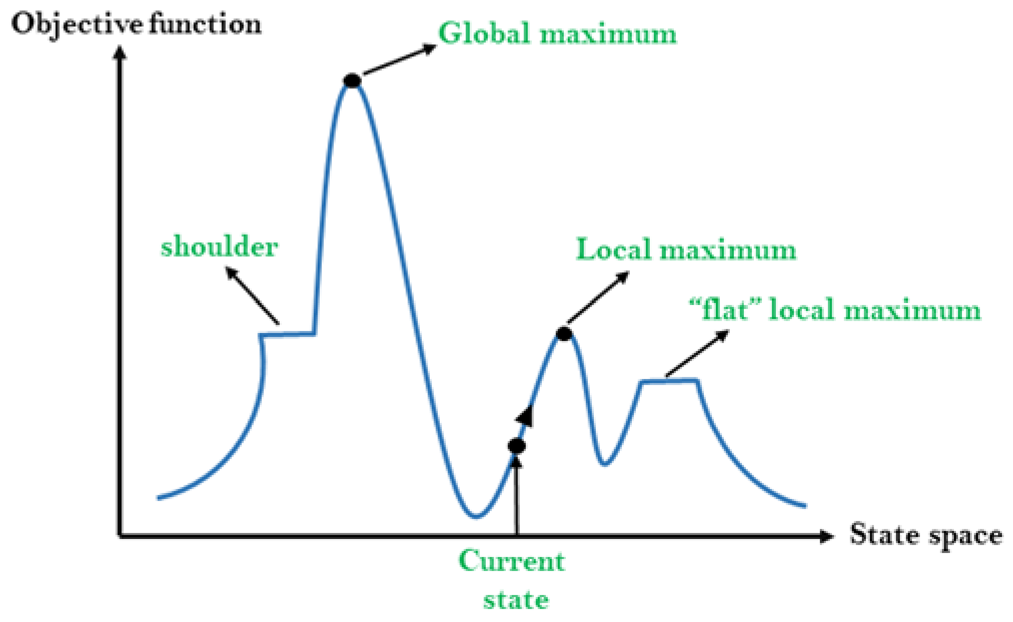
Hill Climbing Technique

Hill climbing algorithm is a technique which is used for optimizing the mathematical problems. Hill climbing algorithm is a local search algorithm which continuously moves in the direction of increasing elevation/value to find the peak of the mountain or best solution to the problem. It terminates when it reaches a peak value where no neighbor has a higher value.

It is also called greedy local search as it only looks to its good immediate neighbor state and not beyond that.

The state-space landscape is a graphical representation of the hill-climbing algorithm which is showing a graph between various states of algorithm and Objective function/Cost.

On Y-axis we have taken the function which can be an objective function or cost function, and state-space on the x-axis. If the function on Y-axis is cost then, the goal of search is to find the global minimum and local minimum. If the function of Y-axis is Objective function, then the goal of the search is to find the global maximum and local maximum.



Different regions in the state space landscape:

Local Maximum: Local maximum is a state which is better than its neighbor states, but there is also another state which is higher than it.

Global Maximum: Global maximum is the best possible state of state space landscape. It has the highest value of objective function.

Current state: It is a state in a landscape diagram where an agent is currently present.

Flat local maximum: It is a flat space in the landscape where all the neighbor states of current states have the same value.

Shoulder: It is a plateau region which has an uphill edge.

Best-first Search Algorithm

Greedy best-first search algorithm always selects the path which appears best at that moment. It is the combination of depth-first search and breadth-first search algorithms. It uses the heuristic function and search. Best-first search allows us to take the advantages of both algorithms. With the help of best-first search, at each step, we can choose the most promising node. In the best first search algorithm, we expand the node which is closest to the goal node and the closest cost is estimated by heuristic function, i.e.

1. f(n)= g(n).

Were, h(n)= estimated cost from node n to the goal.

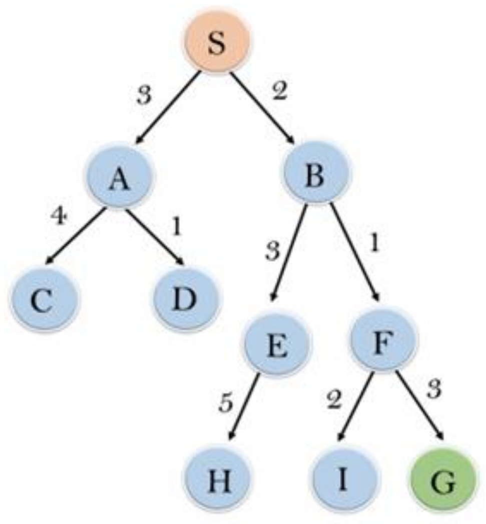
The greedy best first algorithm is implemented by the priority queue.

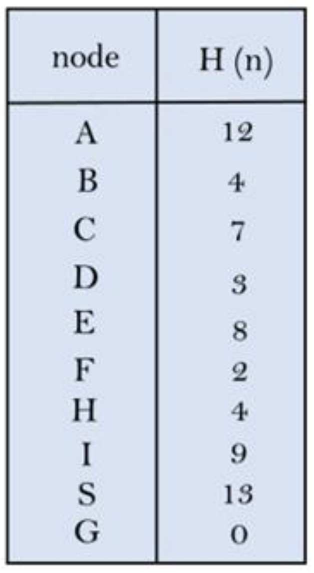
Best first search algorithm:

- **Step 1:** Place the starting node into the OPEN list.
- **Step 2:** If the OPEN list is empty, Stop and return failure.
- **Step 3:** Remove the node n, from the OPEN list which has the lowest value of h(n), and places it in the CLOSED list.
- **Step 4:** Expand the node n, and generate the successors of node n.
- **Step 5:** Check each successor of node n, and find whether any node is a goal node or not. If any successor node is goal node, then return success and terminate the search, else proceed to Step 6.
- **Step 6:** For each successor node, algorithm checks for evaluation function f(n), and then check if the node has been in either OPEN or CLOSED list. If the node has not been in both list, then add it to the OPEN list.
- **Step 7:** Return to Step 2.

Example of BFS

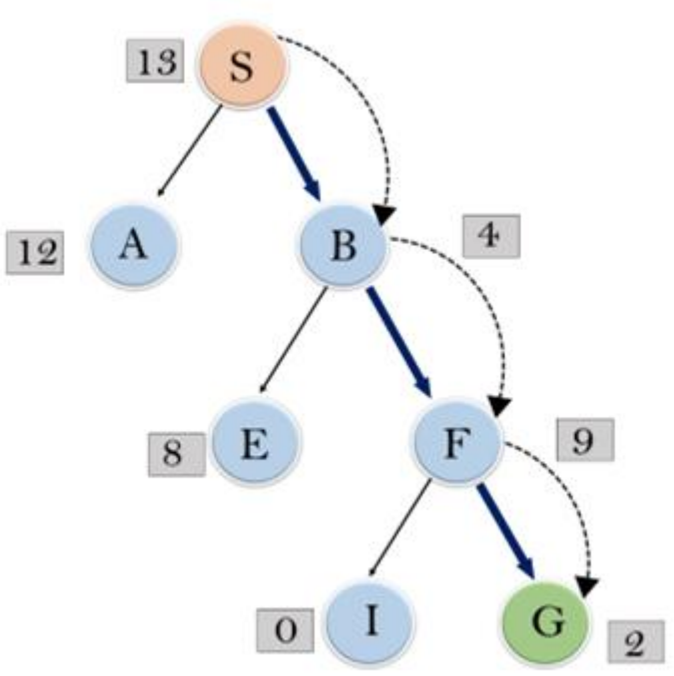
Consider the below search problem, and we will traverse it using greedy best-first search. At each iteration, each node is expanded using evaluation function f(n)=h(n) , which is given in the below table.





node	H (n)
A	12
B	4
C	7
D	3
E	8
F	2
H	4
I	9
S	13
G	0

In this search example, we are using two lists which are **OPEN** and **CLOSED** Lists. Following are the iteration for traversing the above example.



Expand the nodes of S and put in the CLOSED list

Initialization: Open [A, B], Closed [S]

Iteration 1: Open [A], Closed [S, B]

Iteration 2: Open [E, F, A], Closed [S, B]
: Open [E, A], Closed [S, B, F]

Iteration 3: Open [I, G, E, A], Closed [S, B, F]
: Open [I, E, A], Closed [S, B, F, G]

Hence the final solution path will be: **S----> B----->F----> G**

Problem Reduction

A solution to a problem can be obtained by decomposing it into smaller sub-problems. Each of this sub-problem can then be solved to get its sub solution. These sub solutions can then recombined to get a solution as a whole. That is called is **Problem Reduction.**

This method generates arc which is called as **AND** arcs. One AND arc may point to any number of successor nodes, all of which must be solved in order for an arc to point to a solution.

Problem Reduction Algorithm

1. Initialize the graph to the starting node.
2. Loop until the starting node is labelled **SOLVED** or until its cost goes above **FUTILITY**:
   (i) Traverse the graph, starting at the initial node and following the current best path and accumulate the set of nodes that are on that path and have not yet been expanded.
   (ii) Pick one of these unexpanded nodes and expand it. If there are no successors, assign FUTILITY as the value of this node. Otherwise, add its successors to the graph and for each of them compute f'(n). If f'(n) of any node is O, mark that node as SOLVED.
   (iii) Change the f'(n) estimate of the newly expanded node to reflect the new information provided by its successors. Propagate this change backwards through the graph. If any node contains a successor arc whose descendants are all solved, label the node itself as SOLVED.

Constraint Satisfaction

Constraint satisfaction is the process of finding a solution to a set of constraints that impose conditions that the variables must satisfy. A solution is therefore a set of values for the variables that satisfies all constraints—that is, a point in the feasible region.

The techniques used in constraint satisfaction depend on the kind of constraints being considered. Often used are constraints on a finite domain, to the point that constraint satisfaction problems are typically identified with problems based on constraints on a finite domain. Such problems are usually solved via search, in particular a form of backtracking or local search.