Artificial Intelligence

Satvik Gupta

February 6, 2023

Intelligence: Ability to understand and react.

Allows:

1. React dynamically and quickly.
2. Recognize the relative importance of different elements of the situation.
3. Handling ambiguity, contradictions, incomplete and uncertain information.

AI

Artificial Intelligence (AI) refers to the development of abilities of natural intelligence to an extent ia machine.

Definitions

John McCarthy

The term "AI" was coined by John McCarthy (MIT, 1956). AI being a branch of computer science, is concerned with making computers behave like humans.

Rich and Knight

"AI is the study of how to make computers do things which at the moment people are better at doing" – Rich and Knight

Akerkar and Sajia

"AI is the branch of computer science that attempts to solve problems by mimicking the human thought process using heuristics and a symbolic, non-algorithmic approach" - Akerkar and Sajia

John Durkin

"AI is the field of study in computer science that pursues the goal of making a computer reason in a manner similar to humans" - John Durkin

Approaches

Thinking Humanly:

- Cognitive modeling approach
- This approach tries to solve a problem by seeing how human brains work and tries to incorporate the human problem-solving process into a machine by one of three ways:
  - Introspection (Observing one's own thoughts)
  - Psychological experiments (Observing someone else in action)
  - Brain imaging (Observing the brain in action)

Thinking Rationally:

- Laws of thought approach
- This approach is based on logic and the process of reasoning.
- It uses syllogisms which when provided patterns for argument structures always yield conclusions given correct premises.

Acting Humanly:

- Turing test approach
- In this the interrogator asks questions to both a human and machine. If he is unable to discriminate who is who, the machine passes the Turing test.
- To pass the Turing test, a machine must be equipped with techniques for natural language processing (NLP), knowledge representation, automated reasoning and machine learning

Acting Rationally:

- Rational agent approach
- A rational agent is one, so as to achieve the best outcome or the best expected outcome when there is uncertainty
- It is more generalized than the Laws of thought approach

Task Domains

1. Mundane tasks:
   - Perception
   - Communication (NLP)
   - Common sense reasoning
2. Formal Tasks:
   - Game Playing
   - Mathematical reasoning
3. Expert tasks:
   - Engineering tasks
   - Medical tasks

AI Techniques

1. Search based Techniques
   1.1. Brute Force Search (Uninformed)
   1.2. Informed Search (Heuristic)
2. Knowledge
3. Abstraction

Search techniques:

1. Search provides a way of solving problems for which no more direct approach is available. It also provides a framework into which any direct techniques that are available, can be embedded.
2. A search program finds the solution of a problem by trying various sequences of actions until a solution is found.

Advantages:

- Search techniques might be the best way when all other options are exhausted.
- The search process itself finds the sequence of actions to achieve the goal.

Disadvantages:

- In real world problems, the search space is so large that it is impossible or impractical to explore the entire search space.

Knowledge based Techniques:

The use of knowledge provides a way of solving complicated problems by manipulating the structures of concerned objects. Knowledge is indispensable but also voluminous and constantly changing (dynamic), hard to characterize accurately. Also, the organization of knowledge greatly impacts its usage and efficiency of the technique using it.

An AI technique, is a method that exploits knowledge that should be represented in such a way that:

- The knowledge captures generalization so that a separate representation of individual situations is not required.
- Important properties of situations are grouped together else the knowledge reduces to simply a large amount of data.
- It is understandable by the people involved in the project.
- It is easily modifiable to reflect changing situations.
- It is usable, even when information is not accurate or complete.
- It should be able to narrow down the range of search possibilities.
- It finds a way of separating important features and notifications from the unimportant ones, that would confuse any process.

AI Solutions

Two categories:

1. Weak Solutions/Weak AI
    - General search methods.
    - Not motivated by achieving human level performance.
    - Primary aim for these solutions is ***NOT*** to model how a human thinks.
    - They require more computations and less knowledge.
    - These methods aim to solve any problem.
    - Not very effective in specific tasks.
    - For e.g. A* search.
2. Strong solutions/Strong AI

    AKA Knowledge-based, or intensive solutions

    - Used for expert systems.
    - More knowledge, hence less computations.
    - Achieve better performance in specific tasks.
    - Can be guided by weak AI.

Phases of AI Solutions:

1. Search and Logic
2. Probability Based
3. Neural Networks

Problem Formulation

State - Information about environment. It is a general representation of a problem. Technically, the state should contain *all* information about its environment. While describing a problem, we generally only include all the information necessary to make a decision for the task at hand.

State Space - The set of all possible states. Each state represents a possible configuration of the problem. The problem can make a transition from one state to another by using one of the actions or operators available. An operator refers to some representation of an action. An operator usually includes information about what must be true in the world before the operator is applied, and how the world is changed after the operator is applied.

State Space Search

The state space search problem solving method is a way of defining a given problem as a problem of moving around in a state space, where each state corresponds to a legal position in the problem.

The problem solving progresses by starting at an initial state, using a set of rules to move from one state to another, and attempting to end up in a final state.

Types of State Views

1. Atomic View States are numbered to be differentiated, but we cannot view the details of each state, or the values of the variables in the state.
2. Propositional/Factored View We can view inside the state.
3. Relational View State objects are represented in how they relate to each other.
4. First Order View (Relational + Functions)

Question - **Consider a problem where we have 2 rooms and 1 vacuum cleaner. Each room may or may not have dirt in it. The vacuum cleaner can only be in one room at a time.How will this problem be represented in all the 4 views?**

R1 - Room1

R2 - Room2

D - Dirt

ND - No Dirt

V - has vacuum cleaner

Atomic View

1. (R1DV, R2D)
2. (R1D, R2DV)
3. (R1DV, R2ND)
4. (R1D, R2NDV)
5. (R2NDV,R1D)
6. (R2ND,R1DV)
7. (R1NDV,R2ND)
8. (R1ND,R2NDV)

Do rest later

Problem Solving

1. Define/Formulate the Problem.
2. Analyze
3. Task Knowledge
   - Isolation
   - Knowledge Representation
4. Choose an appropriate technique and apply it.

Steps to Define/Formulate a Problem

1. States
2. Initial State
3. Actions
4. Transition Model
5. Goal Test
6. Path Cost

Question Travel from City A to City B (formulate the problem using above 6 steps)

1. **States**

   Each state is a location between A and B.

2. **Initial State** - A

3. **Actions**

   - Choosing a mode of transport from A to some intermediate location, say C.

4. **Transition Model**

   - The state resulting from travelling from a previous location to the current location will have the current location as source, for the next segment of the journey.

5. **Goal Test**

   - Is current location same as specified destination location.

6. **Path Cost**

   - Path cost can be either the amount of fare expended, time taken, or distance traveled.

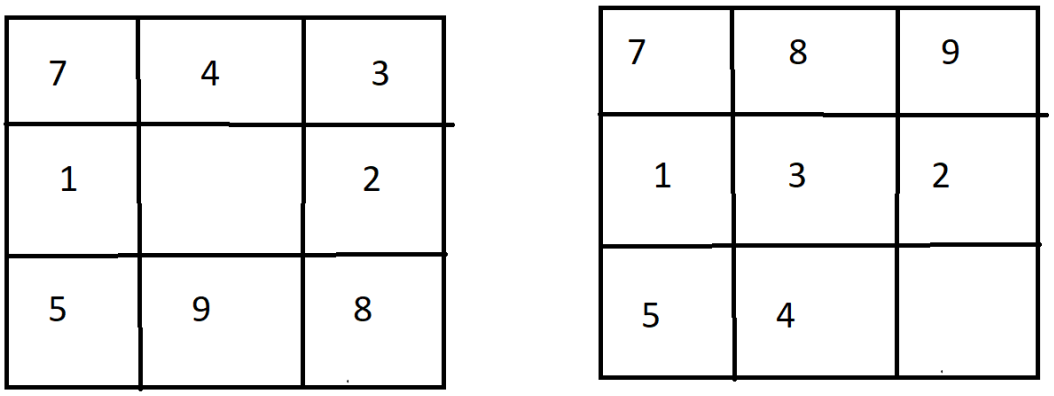
Chess Board

1. States 32 pieces, each of which can have 64 position. Each state will be defined by a 32-tuple containing the X and Y co-ordinate of each piece, and which player's turn it is (B/W). If a piece has been moved off the board, it's coordinates will be (-1,-1).
2. Initial State Each piece will be at the starting position defined in chess. Starting player is white.
3. Actions Each piece has certain rules for movement. Each player takes turns in moving, and can only move one piece at a time, following the legal rules for that piece. Pieces off the board cannot be moved.
4. Transition Model Each movement will result in a new state defined by the new positions of the pieces, and the turn will change. Movement for each piece will be according to the rules for that piece.
5. Goal Test
   - Checkmate from either side. (Win/Lose)
   - No possible moves from either side. (Draw)
6. Path cost can be either the time taken, the number of pieces lost, or the number of moves taken until the game ends.

***Question* 8 Puzzle Problem**

Consists of a 3x3 board with 8 numbered tiles and a blank space. A tile adjacent to the blank space can slide into the space. The objective is to reach a specified goal state.

For example:



(Initial State on the left, Desired Final State on the Right)

1. **States**

   A state description specifies the location of each of the 8 tiles and the blank space in one of the nine squares.

2. **Initial State**

   Any possible configuration can be designated as initial state.

3. **Actions**

   Actions can be defined as *movements of the* ***blank*** space left,right,up or down. Different subsets of these are possible depending on where the blank space is.

4. **Transition Model**

   Given a state and action, this returns the resulting state. For example, if we apply the action *left* to the start state, the resulting state will have 1 and the blank space switched.

5. **Goal Test**

   In this we compare whether the current configuration matches the goal configuration.

6. **Path Cost**

   Let us assume each move costs 1. Path cost will be number of steps.

Production System

A production system provides a structure for the search process in AI system It is a program or system that facilitates in describing the search process and performing the search process

Production syntax

$$if \rightarrow then$$

$$antecedent \rightarrow consequence$$

Control Strategies

Control Strategies decide which rule to apply next during the search process It needs to deal with certain situations like:

- absence of any rules exactly matching the given fact
- more than one rule matching the given fact

  In this case a conflict resolution strategy must be used to decide on a single matching rule

A good control strategy saves time.

Characteristics of a Good Control Strategy

1. It should cause movement which takes us nearer to our goal. Each application of an if-else rule should take us closer to our goal.
2. It should be systematic.

Water Jug Problem

You are given 2 jugs 4L and 3L without any marking and a tap to fill the 2 containers with water.

The Goal is to get exactly 2L of water in the 4L water jug.

1. Formulate it as a space search Problem
2. Enlist all the possible rules
3. Find a possible solution
4. Apply BFS and DFS

X - water in 4L jug

Y - water in 3L jug

Initial State $(x, y) \rightarrow (0, 0)$

A*NS**WER***

(0,0)

(4,0)

(1,3)

(1,0)

(0,1)

(4,1)

(2,3)

Rules

$$(x, y) \rightarrow (0, 3)$$

$$(x, y) \rightarrow (4, 0)$$

$$(x, y) \rightarrow (x, 0)$$

$$(x, y) \rightarrow (0, y)$$

if (x+y>=4 and y>0) :

$$(x, y) \rightarrow (4, y - (4 - x))$$

if (x+y>=3 and x>0):

$$(x, y) \rightarrow (x - (3 - y), 3)$$

if(x+y<=4 and y>0):

$$(x, y) \rightarrow (x, y, 0)$$

if(x+y<=3 and x>0):

$$(x, y) \rightarrow (0, x + y)$$

Formulating with 6 steps

States

Both jug can be empty or may have certain amount of water in them. Represented as:

$(0...4, 0...3)$

Initial States

Both jugs are empty

$(0, 0)$

Actions

- Fill up either container.
- Empty either container.
- Pour contents of one container into another.

Transitions (same as rules mentioned above.)

Goal Test

We will have reached our goal if 4L jug has 2L water.

I.e, if $state == (2, x)$, where $x \in [0, 3]$

Path Cost

Path cost can be:

- Number of moves.
- Amount of Water taken from tap.

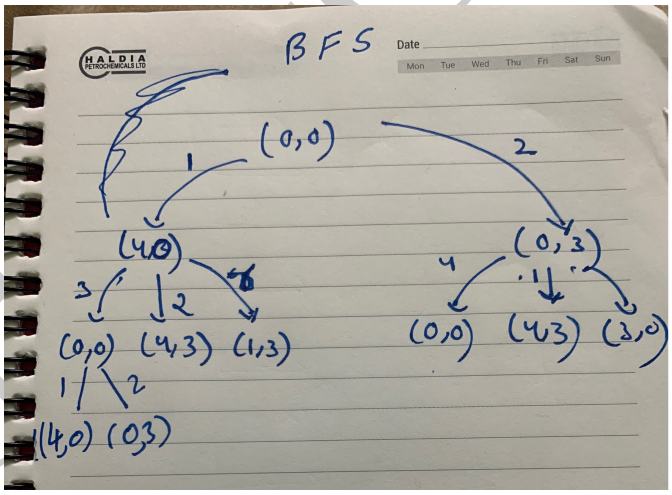


Figure 1: State Transition Tree for Water Jug Problem

Characteristics of Production System

Monotonic Production System

In this, the application of a rule never prevents the later application of another rule, that could also have been applied at the time the first rule was selected.

Non-monotonic production systems are those in which the above is not true.

Partially Commutative Prod System

If the application of a particular sequence of rules transforms a state $X \to Y$, then any allowable permutation of these rules also performs the transformation $X \to Y$

Commutative Prod System

Both Monotonic and Partially Commutative

	Monotonic	Non-Monotonic
Partially Comm.	Theorem Planning	Robot Navigation
Not Partially Comm.	Chemical Synthesis	Chess

Characteristics of a Problem

1. **Decomposability**
2. **Ignoring/Undoing steps**:
   - Ignorable (Theorem Proving)
   - Recoverable (8 Puzzle)
   - Irrecoverable (Chess)
3. **Predictability**:

   If we're able to predict whether we will reach our goal state, and how. In 8 Puzzle we can predict. In chess we cannot, since goal depends on the opponent's actions also.
4. **Nature of Solution**:

   Is there a single goal state or multiple goal states are acceptable?
5. **Solution Type**:
   - State
   - Path
6. **Role of Knowledge**:

   Is knowledge necessary (as in NLP), or is it just used to speed things up (as in chess engine, where we can brute force everything but knowledge helps optimization by a lot.)
7. **Interactive**
8. **Problem Classification**

Problems in Search Methodology

1. Search direction:
    - Forward
    - Backward
2. Search Strategy
3. Node Representation
4. Search Process Representation:
    - Tree
    - Graph

Heuristic Search Techniques

They're also called **informed search techniques**. These are useful when direct techniques are impractical or impossible. These methods are guided by direct methods. These are general purpose techniques independent of problem domain.

When applied to particular problems, they apply domain-specific knowledge. These are also called weak methods. They provide a framework into which domain specific knowledge can be placed. They form the core of most AI systems.

Generate and Test Technique

1. Generate a possible solution/path.
2. See if it is a goal state/ solution.
    - If yes, return and quit.
    - If not, backtrack to step 1.

Hill Climbing

Simple HC

1. Generate an initial state.
    - If it is a goal state, return and quit.
    - If not, call it as current state.
2. Apply an operator (that hasn't yet been applied) to current state and generate a new state.
3. If new state is a goal state, return and quit.

   Otherwise,compare new state with current state.
    - If new state is better than current state, make new state as current state.
    - If new state is not better than current state, go back to step 2.

Here, *better* is evaluated by a heuristic function. It can be cost, or number of steps, distance, etc. depending on the problem.

Steepest Ascent HC

Chooses the best successor as current state. It generates all successors from the current state.

Simple HC chooses the first better new state as current state.

1. Check if initial state is goal state.
    - If yes, return and quit.
    - If no, make initial state as current state.
2. Loop until a solution is found, or until a complete iteration produces no change in the current state (CS):

1. Let $SUCC$ be the worst possible successor to CS (i.e, no successor to CS will be worse than $SUCC$)
2. For each operator applicable to CS
    1. Apply the operator to CS and generate the new state (NS).
    2. Evaluate the NS. If it is a goal state, return and quit. If not, compare it with $SUCC$. If NS is better than $SUCC$, then set $SUCC = NS$. Otherwise, let $SUCC$ remain as it is.
    3. If $SUCC$ is better than CS, set $CS = SUCC$.

Issues with HC

Both types of HC may get stuck. Either algorithm may terminate, not by finding a goal state, but by getting to a state from which no better states can be generated. This can happen in 3 cases:

- **Local Maxima**

  It is a state which is better than all its neighbors, but there me a far off state that is better than it. All immediate moves from the local maxima seem to make things worse.

  Local maxima often occur almost within sight of a solution. In this case, they are called foothills.

  Solution:

  Backtrack and try another path.

- **Plateau**

  It is a flat area of search space, with all neighbors having the same value. All the immediate next moves seem same. No best move exists.

  Solution:

  Make a big jump in some random direction to try to get a new section of the search space. We can apply a simple one-move rule several times in the same direction to accomplish this.

- **Ridge**

  It is a special kind of local maxima. It is an area in the search space that is higher than surrounding areas, but it itself has a slope (that we wish to climb).

  Solution:

  Apply two or more rules before doing the test.

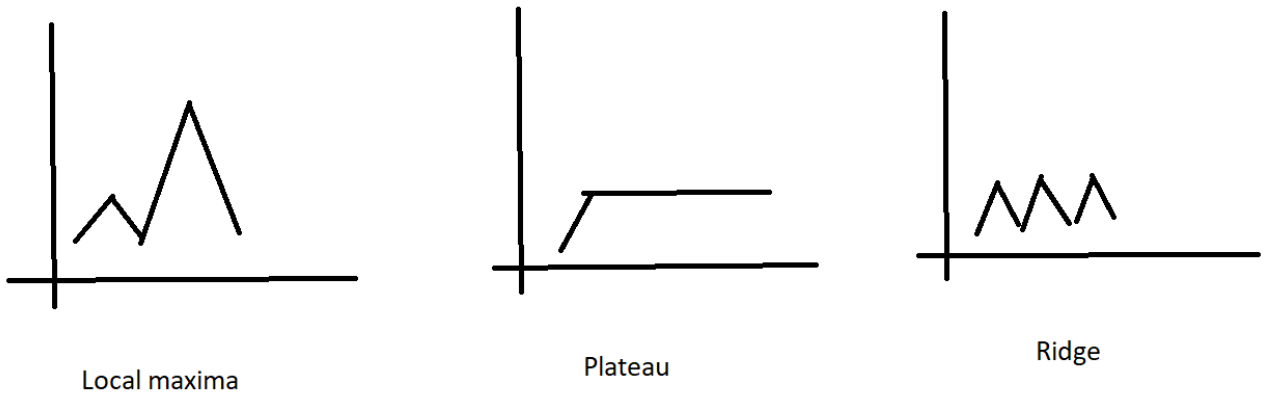


Figure 2: Issues in Hill Climbing Techniques

Simulated Annealing

Heuristic Function is referred to as the objective function. We try to minimize the value of the objective function .

Annealing is the process of melting metals, and gradually cooling them until a solid state is reached. Goal is to find a minimal energy final state.

Objective function is the energy level. There is some probability that a transition to a higher energy state will occur (even though, generally physical substances move towards lower energy configurations). The probability of such a transition to a higher energy state is

$$p = e^{\Delta E/kT}$$

Where ΔE is change in Energy level T is temperature and k is the Boltzmann constant.

Mapping this to AI and HC, we use the formula

$$p' = e^{\Delta E/T}$$

where T is the objective function.

A schedule for T must be maintained, called as annealing schedule. It should have - an initial value of T - a criteria to reduce T - amount by which T must be reduced - When to quit

Simulated annealing is generally used with problems having a large search space.

The annealing schedule must specify:

1. Initial Value of T.
2. By what amount to reduce T
3. Criteria for reducing T
4. When to Quit

Algorithm for Simulated Annealing

1. Evaluate the initial state. If it is also a goal state, then return it and quit. Otherwise, continue with the initial state as the current state.
2. Initialize **BEST-SO-FAR** to the current state.
3. Initialize T according to the annealing schedule.
4. Loop until a solution is found or until there are no new operators left to be applied in the current state.
    1. Select an operator that has not yet been applied to the current state and apply it to produce a new state.
    2. Evaluate the new state. Compute

    $$\Delta E = (value\ of\ current) - (value\ of\ new\ state)$$

    - If the new state is a goal state, then return it and quit.
    - If it is not a goal state but is better than the current state, then make it the current state. Also set **BEST-SO-FAR** to this new state.
    - If it is not better than the current state, then make it the current state with probability p' as defined above. This step is usually implemented by invoking a random number generator to produce a number in the range $[0, 1]$. If the number is less than p', the move is accepted, otherwise do nothing

    3. Revise T as per annealing schedule.
5. Return **BEST-SO-FAR** as the answer.

Best First Search && A*