FOUNDATIONS OF ARTIFICIAL INTELLIGENCE

Sumit Kumar, Ashutosh Bhatt
Computer Science and Engineering Department,
Maharishi Dayanand University, Rohtak, Haryana, India

Abstract—**Artificial Intelligence was developed in 1956 and came into existence as a paradigm of cognition. It derived a powerful and lusty philosophical patrimony of functionalism and affirmatism. The history has shown a turn away from the functionalism of standard AI toward an alternative position that re-asserts the priority of development, interaction, and, more recently, emotion in cognitive systems, focusing now more than ever on enactive models of cognition. The method of looking for the solutions to problems, in Artificial Intelligence, can be brought about, in many ways, without cognition of the Domain, and in different situations, with knowledge of it. This procedure is usually called Heuristic Search. In such techniques matrix techniques reveal themselves as important. Their introduction can enable us to understand the precise way to the look for a solution. This paper explains the logical foundation of Artificial Intelligence with feasible applications.**

Index Terms—**Artificial Intelligence, Fuzzy Logic, Cognition.** *(key words)*

I. INTRODUCTION.

Artificial Intelligence (AI) has been studied for decades and is still one of the most elusive subjects in Computer Science. The Artificial Intelligence ranges from machines which are capable of thinking to search the procedures that necessarily require human senses to work upon. It founds its applications in nearly every way we use computers in society.

The modern AI was developed by classical philosophers who attempted to elaborate the process of human thinking as the mechanical manipulation of symbols. This work culminated in the invention of the programmable digital computer in the 1940s. The field of AI research was founded at a conference on the campus of Dartmouth College in the summer of 1956. The term artificial intelligence was first coined by John McCarthy in 1956 when he held the first academic conference on the subject. The main advancement in the AI has been made over the past fifty years particularly in search algorithms, machine learning algorithms, and integrating statistical analysis. However most of the breakthroughs in AI aren't noticeable to most people. A common theme in the field has been to overestimate the difficulty of foundational problems. In addition, there has always been a tendency to redefine what 'intelligent' means after machines have mastered an area or problem. The problems analysed by A.I. can be classified according to their level. The first level includes problems of decision, learning, and perception, planning and reasoning. The second level includes tasks of classification, representation and search.

II. SEARCH METHODS

In the searching process, all the uninformed search methods share three common requirements : 1) a collection of facts based on a choice of representation providing the current state, and the goal state. 2) a set of operators which defines possible transformations of states and 3) a strategy which describes how transformations amongst states will be carried out by applying operators. Reasoning from a current state in search of a state which is closer to a goal state is known as forward reasoning. Reasoning backwards to a current state from a goal state is known as backward reasoning.

The differentiation between: without information about the domain (Blind Search), and with information about of the domain (in this case, called Heuristic Search). can be exclusively made. A choice according to the kind of problem, between Extended Search and Deep Search together with other methods; some of these being derived. So, the ultimate or the final search is not the same as searching with the possibility of backward motion called as backtracking.

State Space Search

State space search is a process used in the computer science, including artificial intelligence (AI), in which consequent configurations or states of an instance are speculated, with the goal of finding a goal state with a desired property. Problems are often modelled as a state space, a set of states that a problem can be in. The set of states forms a graph where two states are connected if there is an operation that can be performed to transform the first state into the second. State space search often differs from traditional computer science search methods because the state space is implicit. The typical state space graph is much too large to generate and store in memory. Instead, nodes are generated as they are explored, and typically discarded thereafter. A solution to a combinatorial search instance may consist of the goal state itself, or of a path from some initial state to the goal state. Exhaustive search of a problem space (or search space) is often not feasible or practical due to the size of the problem space. In some instances it is however, necessary. More often, we are able to define a set of legal transformations of a state space (moves in the world of games) from which those that are more likely to bring us closer to a goal state are selected while others are never explored further. This technique in problem solving is known as split and prune. In AI the technique that emulates split and prune is called generate and test.

A. Depth First Search

The Depth First Search (DFS) is one of the most basic and fundamental Blind Search Algorithms. It is for those who want to probe deeply down a potential solution path in the hope that solutions do not lie too deeply down the tree. That is "DFS is a good idea when you are confident that all partial paths either reach dead ends or become complete paths after a reasonable number of steps.

To Start with DFS :

1. Put the Start Node on the list called OPEN.
2. If OPEN is Empty, Exit with failure, otherwise continue.
3. Remove the first node from OPEN. Call this node as n.
4. If the depth of n equals the depth bound, go to (2); otherwise continue.
5. Expand node n, generating all successors of n. Put these (in arbitrary order) at the beginning of OPEN and provide pointers back to n.
6. If any of the successors are goal nodes, exit with the solution obtained by tracing back through the pointers; otherwise go to 2.

B. Breadth First Search

Breadth First Search always explores nodes closest to the root node first, thereby visiting all nodes of a given length first before moving to any longer paths. It pushes uniformly into the search tree. Breadth first search is most effective when all paths to a goal node are of uniform depth. It is a bad idea when the branching factor (average number of offspring for each node) is large or infinite. Breadth First Search is also to be preferred over DFS if you are worried that there may be long paths (or even infinitely long paths) that neither reach dead ends nor become complete paths (Winston, 1992). For the tree in Figure 5 Breadth First Search would proceed alphabetically.

The algorithm for Breadth First Tree Search is:

1. Put the start node on a list called OPEN.
2. If OPEN is empty, exit with failure; Otherwise continue.
3. Remove the first node on OPEN and put on a list called CLOSED;
   Call this node n;
4. Expand node n, generating all of its successors. If there are no successors, go immediately to (2).
   Put the successors at the end of OPEN and provide pointers from these successors back to n.
5. If any of the successors are goal nodes, exit with the solution obtained by tracing back through the pointers; otherwise go to (2).

C. Extended Search

We advance in the tree (or the graph) by levels. So, we obtain the lowest-cost, if it exists.

Deep Search

We expand only one link each time, from the root-node. If we reach a blind alley (cul-de-sac) in the graph, we come back to the nearest node and from this we take a ramification (or alternative branch) in the tree.

In this type of search, it is normal to establish an exploration limit, or depth limit (dl), by fixing the maximal length of the path, from the root.

Also, in this case the direction assumed is from left to right. It can be interpreted as the ordered ($l \rightarrow r$) and exhaustive journey of an imaginary ship, visiting each fiord, on this imaginary coast. We are avoiding the algorithms associated with each one of these searching processes.

III. HEURISTIC SEARCH

A heuristic is a method that might not always find the best solution but is guaranteed to find a good solution in reasonable time.By sacrificing completeness it increases efficiency.

Useful in solving tough problems which could not be solved any other way solutions take an infinite time or very long time to compute.The classic example of heuristic search methods is the travelling salesman problem.

Heuristic Search methods Generate and Test Algorithm

1.Generate a possible solution which can either be a point in the problem space or a path from the initial state.

2.Test to see if this possible solution is a real solution by comparing the state reached with the set of goal states.

3. If it is a real solution, return. Otherwise repeat from 1.

This method is basically a depth first search as complete solutions must be created before testing. It is often called the British Museum method as it is like looking for an exhibit at random. A heuristic is needed to sharpen up the search. Consider the problem of four 6-sided cubes, and each side of the cube is painted in one of four colours. The four cubes are placed next to one another and the problem lies in arranging them so that the four available colours are displayed whichever way the 4 cubes are viewed. The problem can only be solved if there are at least four sides coloured in each colour and the number of options tested can be reduced using heuristics if the most popular colour is hidden by the adjacent cube.

Now, we introduce a new mathematical tool: the heuristic evaluation function, f. by such a function, we assign a value to each node, n. So, f(n) gives us the estimation of the real distance (unknown), from the actual node, n, until the final node, m.

One of these procedures is called the Gradient Method, or Climbing Search. According to this, in the expansion of each node, we must select the link which connects with the node of the subsequent level where the value of f is maximal, in the supposition off reaching the greatest value in the finish node. We can also proceed in the reversed sense: reaching the lesser value, in each step, until the minimal, in the last node. BRAIN-Broad Research in Artificial Intelligence and Neuroscience

There are critics on the Heuristic Search, because of its unpredictability. It can find good solutions, but not necessarily the best.

The A* algorithm. This is a convention for the introduction of this algorithm with their useful properties of completeness and admissibility. According to the last property, if there is any solution, it will find it.

A* is a particular case of the searching procedure of "first the best", into the strategies of "alternative explorations". It belongs to the Procedures of General Graph Search. In each step, we go revisiting the Open List. Initially, such a set is empty. Our successive elections would be based on the previous assignment to each node of the value of f in it. The selection of each node is attained according to the lesser value of the heuristic function on the nodes of their level, as a general rule. The comparison is carried out into the Open List, independent of the original level of each node. Generally, we prefer the solution of lesser cost. All the explored nodes, then, pass to be stored in the Closed List. Such nodes remain inactive for the rest of the process.

The heuristic function, f, can be decomposed in two parts or components, g and h:

$f(k) = g(k) + h(k)$

for each node k. Where g(k) gives us the real cost (known) of the best path found from the "root", or initial node, until the actual node, k. And h(k) is the estimation of the length of the optimum path (unknown, until now), from k to the final node m.

Fuzzy and Modal Logic

Fuzzy Logic provides functions, apps, and a Simulink® block for analysing, designing, and simulating systems based on fuzzy logic. The product guides you through the steps of designing fuzzy inference systems. Functions are provided for many common methods, including fuzzy clustering and adaptive neurofuzzy learning.

Fuzzy logic is a form of many-valued logic or probabilistic logic; it deals with reasoning that is approximate rather than fixed and exact. Compared to traditional binary sets (where variables may take on true or false values) fuzzy logic variables may have a truth value that ranges in degree between 0 and 1. Fuzzy logic has been extended to handle the concept of partial truth, where the truth value may range between completely true and completely false. Furthermore, when linguistic variables are used, these degrees may be managed by specific functions. Irrationality can be described in terms of what is known as the fuzzjective.

The fuzzy logic toolbox lets you model complex system behaviors using simple logic rules, and then implement these rules in a fuzzy inference system. You can use it as a stand-alone fuzzy inference engine. Alternatively, you can use fuzzy inference blocks in Simulink and simulate the fuzzy systems within a comprehensive model of the entire dynamic system.

Desirable properties for such Theories: To formulate the knowledge of the domain, D, in a very effective and efficient way, as theory, three characteristics could be necessary: completeness, consistency and tractability. According to the first of them (completeness), each formula must be demonstrable into the theory. For the second one (consistency), the new contributions to the system must not generate inner contradictions with the previous assertions or axioms. The tractability must give us a moderate complexity. That is, manipulating laws and premises in the Derivate Calculus, through the Inference process, must not result in excessive temporal and spatial complexities.

New trends in Algebra and Non-Monotonic Logic

There exists the possibility of obtaining an extended theory, in Linear Algebra, by the introduction of concepts and methods of Fuzzy Logic ([5], [8]), where the idea of sets, relations and so on, must be modified in the sense of adequately covering the indeterminacy or imprecision of the real world. In the problems concerning this "real world", only one of the "possible worlds", the Monotonic Logic seldom works. Such a type of Logic is Classical Logic in formal worlds, such as in the Mathematical fields of the past, because for the future we need to introduce uncertainty and their related problems. Instead of Monotonic Logic, the Non-Monotonic Logic ([1], [6]) must be developed and applied where the extension of the set of sentences can now modify consequences. This happens frequently in the real world: for instance, in the medical sciences. With these and similar techniques, we hope to obtain each time more interesting developments which may contribute to expand some classical and new fields in Game Theory.

Extended Search

We advance in the tree (or the graph) by levels. So, we obtain the lowest-cost, if it exists.

Deep Search

We expand only one link each time, from the root-node. If we reach a blind alley (cul-de-sac) in the graph, we come back to the nearest node and from this we take a ramification (or alternative branch) in the tree. In this type of search, it is normal to establish an exploration limit, or depth limit (dl), by fixing the maximal length of the path, from the root. Also, in this case the direction assumed is from left to right. It can be interpreted as the ordered (l $\rightarrow$ r) and exhaustive journey of an imaginary ship, visiting each fiord, on this imaginary coast. We are avoiding the algorithms associated with each one of these searching processes.

IV. Heuristic Search

Relative to searching with knowledge of the domain, in an initial phase, it was generally thought that all paths can be explored by the computer. But this is too optimistic: such an exploration can often be very difficult, because of the phenomenon of "combinatory explosion" of the ramifications, when we expand the search. Their spatial and temporal complexity can advise us against their realization. For this reason, we need to select, firstly, the most promising trajectories.

In this way, we cannot obtain the best solution (optima), but an efficient approach to it. Now, we introduce a new mathematical tool: the heuristic evaluation function, f. By such a function, we assign a value to each node, n. So, f(n) gives us the estimation of the real distance (unknown), from the actual node, n, until the final node, m.

One of these procedures is called the Gradient Method, or Climbing Search. According to this, in the expansion of each node, we must select the link which connects with the node of the subsequent level where the value of f is maximal, in the supposition of f reaching the greatest value in the finish node. We can also proceed in the reversed sense: reaching the lesser value, in each step, until the minimal, in the last node.

BRAIN. Broad Research in Artificial Intelligence and Neuroscience.

There are critics on the Heuristic Search, because of its unpredictability. It can find good solutions, but not necessarily the best.

The A* algorithm. This is a convention for the introduction of this algorithm with their useful properties of completeness and admissibility. According to the last property, if there is any solution, it will find it.

A* is a particular case of the searching procedure of "first the best", into the strategies of "alternative explorations". It belongs to the Procedures of General Graph Search. In each step, we go revisiting the Open List. Initially, such a set is empty. Our successive elections would be based on the previous assignment to each node of the value of f in it. The selection of each node is attained according to the lesser value of the heuristic function on the nodes of their level, as a general rule. The comparison is carried out into the Open List, independent of the original level of each node. Generally, we prefer the solution of lesser cost. All the explored nodes, then, pass to be stored in the Closed List. Such nodes remain inactive for the rest of the process.

The heuristic function, f, can be decomposed in two parts or components, g and h:

$f(k) = g(k) + h(k)$

for each node k. Where g(k) gives us the real cost (known) of the best path found from the "root", or initial node, until the actual node, k. And h(k) is the estimation of the length of the optimum path (unknown, until now), from k to the final node m.

V. Search problems with two adversaries

There are also strategies specially designed for the treatment of such problems. Their general purpose is to select the necessary steps to winning the game (chess, generally; in fact, this was their origin). We assume alternate moves. In each move, the ideal would be when the player knows his possibilities and even realizes the worst move for his adversary. But it is impossible to control it completely, generally because of the mentioned "combinatory explosion".

For this, we need to develop a tree of depth searching, with limited depth. Suppose each player always makes the best move in each turn. To estimate this goal, we need to introduce a more sophisticated function, which would measure, for each node, the possibilities of being a winner, a looser or a draw.

Let us go back to the algorithm A*, as in the "first the best" (which is a particular case of it), we mention the Open List, with the disposable nodes, in the successive steps, joining the value of the heuristic function in them. Each time we choose a node, according to the minimal distance, we store such a node into the Closed List. And so on, until reaching the final node in the tree search. More procedures of Heuristic Search: We can apply the exploration by graphs And/Or, where we represent a main problem and their sub-problems, as nodes pending from the root-node. In this case, we dispose of two types of links: Or, which indicates different options, and the link type And, which connects the father-nodes to the sub-problems. We can also consider the MINIMAX method, which makes an exhaustive exploration of the Search Tree. And the method of "pruning", where the purpose is to reach the reduction of the number of visited nodes, detecting through a double counter, the pair (α,β). So, I can cut, leaving the remaining nodes pending from such a node.

VI. Fuzzy Logic

Successive attempts were directed towards the summit of very complex problems. Not only in formal worlds, but for the real world as well. This requires new mathematical ideas, which are still in evolution today. Therefore, some essential formalisms appear, such as: Logic, Rules, Associative Networks, Frames, Scripts and so on. For each formalism here included, there are methods of handling the knowledge, which allows for our approach.

As you known, the step from disposable information to new information is called Inference, or Reasoning. The Classical Logic reveals promptly its inadequacy for A.I., where more sophisticated Logic is needed, extending the Classical. So, the Predicate Logic with Identity and then the Modal Logic (ML, in acronym). This extension (ML) of the Predicate Logic made evaluating arguments possible which included the concepts of necessity and possibility. Their symbols (as you know) are □

and ◇, respectively. Both of them would be called "modes of truth". This gives the surname of "Modal" for such a Logic.

Desirable properties for such Theories: To formulate the knowledge of the domain, D, in a very effective and efficient way, as theory, three characteristics could be necessary: completeness, consistency and tractability. According to the first of them (completeness), each formula must be demonstrable into the theory. For the second one (consistency), the new contributions to the system must not generate inner contradictions with the previous assertions or axioms. The tractability must give us a moderate complexity. That is, manipulating laws and premises in the Derivate Calculus, through the Inference process, must not result in excessive temporal and spatial complexities.

VII. Cognition :

One key idea in this paradigm was that AI revolves around the study of high-level cognition. When we say that humans exhibit intelligence, we are not referring to their ability to recognize concepts, perceive objects, or execute complex motor skills, which they share with other animals like dogs and cats. Rather, we mean that they have the capacity to engage in multi-step reasoning, to understand the meaning of natural language, to design innovative artifacts, to generate novel plans that achieve goals, and even to reason about their own reasoning. During AI's first 35 years, much of the discipline's research dealt with these issues, and the progress during that period arguably increased our understanding of the mind. This idea is still active in some AI subfields, such as planning and automated reasoning, although each has developed its own specialized methods, but, unfortunately, other subareas have effectively abandoned their initial concern with high-level cognition. For instance, machine learning, despite its early interest in complex tasks, now focuses almost exclusively on classification and reactive control, whereas natural language processing has replaced its original emphasis on understanding with text classification and information retrieval. These shifts have produced short-term gain with many applications and clear performance improvements on their narrowly defined tasks, but I question whether advances on these fronts tell us much about the nature of intelligence. A few researchers who take the cognitive systems perspective continue to address high-level behaviour (e.g., Friedman, Forbus, & Sherin, 2011; Scally, Cassimatis, & Uchida, 2011), but we need far more work in this important area.

References

[1] Brewka: Non-monotonic Reasoning. Logical Foundations of Commonsense. Cambridge
University Press, 1991.

[2] Garrido, Angel: "Some deep connections between Linear Algebra and Fuzzy Logic".
Contributed talk to the ILAS ´04, the "International Conference on Linear Algebra and Its
Applications", held in the University of Coimbra (Portugal), 2004.

[3] Garrido, Angel: "Matrix Theory and Searching Methods in Artificial Intelligence" International
Congress on Matrix Theory, held in the Center of Math. Studies of the Technion. Haifa, 2005.

[4] Garrido, Angel: "Logics in Artificial Intelligence" Congress "Logic in Hungary 2005".
Organized by the Janos Bolyái Mathematical Society. Budapest, 2005.

[5] Klir & Yuan: Fuzzy Sets and Fuzzy Logic. Theory and Applications. Prentice-Hall Inc., 1995.

[6] Mc Dermott & Doyle: "Non-monotonic logic I". Artificial Intelligence, 13, pp. 41-72, 1980.

[7] Mira: Aspectos Básicos de la Inteligencia Artificial. Sanz y Torres. Madrid, 1995.

[8] Zadeh: "Fuzzy Sets". Information and Control, 8, pp. 338-55, 1965.