Role of Heuristic on Informed and Uninformed Search

Thant Thant
UT-YCC(Pyin Oo Lwin),
thantthant.tt72@gmail.com

Khin Nyein Myint
UCS(Monywa)
khinnyeinmyint@ucsmonywa.edu.mm

Thandar Than
UCS(Taunggi), Myanmar
thandathan@ucstgi.edu.mm

Abstract

The aspiration of this paper is to presents the heuristic function is the important role in artificial intelligence (AI) algorithms. It describes the use of these search algorithms in two types, uninformed and informed search. Firstly, it finds out the solution of the shortest path or cost optimization in uninformed search. Then, some informed search strategies will be operated based on the additional information. Informed search is developed to reduce the amount of search by making the intelligence choice for expansion. To achieve effective outcomes, the decision making depends on the heuristic function. And then the heuristic search to reduce the memory requirement by keeping the generated nodes in memory and pruning some of them away when the memory runs out. This research emphasizes to attention the role of problem-specific information to get optimal solution more efficiently than uninformed strategies.

Keywords: Informed search, uninformed search, shortest path, Heuristic function, Cost optimization

1. Introduction

Artificial intelligence (AI) goes beyond human mental capabilities have exponentially enlarged over time. A full absorbing may not be approachable at once due to be magnitude and complexity, research now targets commercial is able aspects of AI towards providing intelligence services to the human [1]. A search algorithm form of the extract for artificial intelligence agent. Many of the AI problems can be based as a search problem where the province is to reach the goal from the initial state by using heuristic method. The search technology can be commonly grouping into two divisions:

Uninformed (Blind) search method found out solution without getting any additional information that is already granted in the problem situation. The examples are described including uniform-cost search and iterative deepening search (IDS).

Informed (Heuristic) search method is tried out the optimal solution by using additional specified-information in exploring the next step. This method is more efficient, low cost and high performance compared to the uninformed search method. Some best first search examples are described as such algorithms.

The informed search algorithm uses a series of knowledge such as how from the goal, path cost, how to reach to goal node, etc. Greedy best-first search (GBFS) expends the next node based assessment by a heuristic function. So, the behavior of GBFS heavily depends on the quality of the heuristic method. Inaccurate heuristics can affect GBFS drags to the way from a goal. The artificial intelligence can be considered into satisficing and optimal solution. In optimal design, the aim is to achieve the best possible solution with minimizing the sum of action costs of finding. GBFS selects the next action in the environment greedily, i.e. it gets the most favorable state currently known.

In practical, most optimization problems cannot be determined accurately because of the following reasons:

-The available information about the environment concern with the problem is usually defined incomplete. This is important to be able to predict the facts to achieve the optimal solution.

-The too complex information causes the intractable process that less computation resources.

-The changing over time real-world systems, building the hard to have approaches that can be accommodated to new information fast.

Most of the field such as Machine learning (ML) has been deal as an approach that provided with enough training data. A machine learning model can in theory find the near-best hypotheses that capture regularities in the training data with an almost unexplainable tendency to generalize to unobserved situations [2]. In particular, deep learning models can learn hierarchical representations of the data, which would enable them to learn more complex regularities in a more compact (fast to train and deploy) way [3].

The heuristic search is a fundamental and widely-used problem solving agent in artificial intelligence. In addition to finding the most optimal solutions, heuristic method supports how to adapt several of the algorithms to bounded suboptimal search. Although it might not always give the best solution, guaranteed to find a good solution in reasonable time. At the same time, the memory bounded heuristic strategies have development by pruning the unnecessary state to reduce memory requirement.

2. Approach on Searching Algorithms

2.1. Uniform- Cost Search

Uniform-Cost Search (UCS) expands the node on path with low cost. It is implemented using the priority queue. The following equation finds cost of every node with this equation,

$$c(m) = c(n) + c(n, m). \qquad (1)$$

In this equation c(m) is the cost of the current node, c(n) is the cost of the previous node, and C (n, m) is the weight of the edge. Time complexity is $O(b^{\lfloor 1+C^*/e \rfloor})$ and space complexity is $O(b^{\lfloor 1+C^*/e \rfloor})$, where C is the optimal solution cost and each activity costs at least ε [4].

2.2. A* Search

A* algorithm build up from aspects of uniform-cost search and pure heuristic search to productively calculate optimal solution [5]. For calculation in cost of every node with this equation,

$$f(m) = g(m) + h(m) \quad (2)$$

In above equation, f(m) f(m) compute the most reduced collective cost to get the optimal solution, g(m) is the path cost from start node to node m, h(m) is heuristic function that estimated cost of the optimal path from node m to a goal. The consideration of heuristic method in variety of fields such as shortest path finding problem, for example, in which straight line distance is a good reasonable heuristic method. Admissible heuristic function ($h(m) \leq h^*(m)$) brings visited node back from the closed list to open list to get an optimal solution. Time complexity is O(bd) and space complexity is O(bd), where b is branching factor and d is solution depth [6]. The A* algorithm required memory allocation which rapidly change to the environment in larger map size, thus it may cause excessive memory before produce the solution [7].

2.3 Greedy Best-First Search

Greedy best-first search (GBFS) provided to expand the node that is nearest to the goal based on the heuristic searching that evaluates nodes by using just the heuristic function; f(n)=h(n). This function provides an estimate of the length of the path from any given state to the closest goal state and it is used to order nodes in terms of how promising they are expected to lead to a goal.

In GBFS searches an expansion that is often applied meaningful is when the search expands a state with a lower heuristic value than the minimal heuristic value of all previously expanded states. As an example, the boosted dual-queue search algorithm by Richter and Helmert [13] uses such an occurrence to further higher order expansions based on preferred operators. Getting a new lowest h-value in GBFS does not necessarily interpret into meaningful, quantifiable progress in finding an optimal solution.

2.4 Best First Search Analysis

If we analyze searching as a form of traversal in a graph, an uninformed search algorithm would blindly traverse to the next node in a given manner without all thing considered the cost associated with that step. An informed search, like Best First Search (FBS), would use an evaluation function to decide which among the various available nodes is the most favorable before traversing to that node [8]. Best first search is the heuristic search technology and adopts the concept of a priority queue. In searching the graph space, the best first search method uses two lists for tracking the traversal. An 'Open' list which keeps track of the current generated nodes available for traversal and 'CLOSED' list that keeps track of the nodes already traversed.

There are various ways to identify the best node for traversal and accordingly there are various flavors of best first search algorithm with different heuristic evaluation functions f(n). We will describe two most popular versions of the algorithm in this article, namely greedy best first search and A* best first search [14].

2.5. Iterative Deepening Search

The Iterative Deepening Search (IDS) is one of the state space search strategy in which the nodes are expanded on depth by depth. Each depth bound is analyzed as iteration. That is, a depth-limited version of depth-first search, also known as Iterative Deepening Depth First Search (IDDFS), is run repeatedly with increasing depth limits until the goal is found. The time complexity is O (bd) and space complexity is O(bd), where 'b' is the branching factor and 'd' is the depth of the shallowest solution. The IDS also works based on the stack, Last In First Out (LIFO) data structure [9]. The iterative deepening search technique is basic idea of memory bounded heuristic search.

2.6. Memory bounded heuristic search

The foundational way to reduce memory requirements for A* search is to adapt the idea of iterative deepening to the heuristic search context, and the resulting is the iterative-deepening A* search (IDA*) algorithm. The central different between IDA* and standard iterative-deepening is that the cutoff used is the f-cost (g+h) rather than the depth limitation, at each iteration, the cutoff value is the smallest f-cost of any node that go above the cutoff on the previous iteration. There are two more memory bounded algorithms, called Recursive Best First Search (RBSF) and Simplied Memory-bounded A*(SMA*) [10].

Recursive best-first search is a simple recursive algorithm that attempts to similar the operation of standard BFS, but using only linear space. Its structure is similar to that of recursive depth-first search, but rather than continuing indefinitely down the current path, it keeps track of the f- value of the best alternative path available from any ancestor of the current node. If the current node exceeds this limit, the recursion unwinds back to the alternative path. As the recursion unwinds, RBFS replaces the f-value of each node along the path with the best f-value of its children. In this way, RBFS remembers the f-value of the best leaf in the forgotten sub-tree and can therefore decide whether it's worth re-expanding the sub-tree at some later time.

SMA* based on the A* algorithm. The main advantage of SMA* is that it uses a bounded memory, while the A* algorithm might need exponential memory. All other characteristics of SMA* are inherited from A* [11]. It cannot add a new node to the search tree without dropping an old one. SMA* always drops the worst leaf node that the one with the highest f-value. If all the leaf nodes have the same f-value, SMA* expands the newest best leaf and deleting the oldest worst leaf. Like A*, it expands the most promising branches according to the heuristic.

2.7 Related work

There have some published that presented the strength of informed search over the uninformed search in the performance of algorithm in time complexity, space complexity, optimality and completeness. Simone Guercini (2019) and Camilo Caraveo, Fevrier Valdez, Oscar Castillo (2018) presented the use of heuristic optimization heuristic algorithms [8] [12]. The main contradistinguish of in this proposal that prefers to attention the important of exploring the heuristic function for an optimization algorithm.

3. Experimental Results

Consider the operations of uniform-cost search as uninformed search, A* search and greedy best first search as informed search on example of graph depicted in Figure 1.

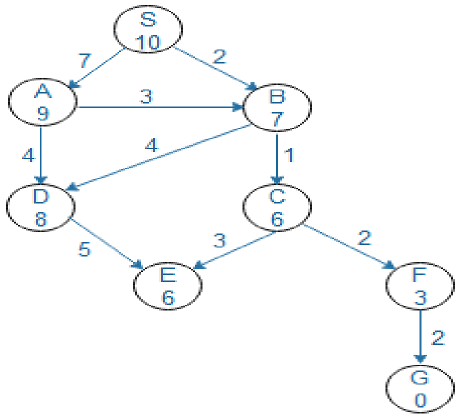

Figure 1. Graph for UCS, A* search, GBFS

The following tables show steps to reach goal state from the start for uniform cost search:

Uniform-cost search	
Generated	Path
S(0)	

Start by adding S to the Generated list

Uniform-cost search	
Generated	Path
B(2)	S(0)
A(7)	

After moving S to Generated list

Uniform-cost search	
Generated	Path
C(3)	S(0)
D(6)	B(2)
A(7)	

After moving B to Generated

Uniform-cost search	
Generated	Path
F(5)	S(0)
D(6)	B(2)
E(6)	C(3)
A(7)	

After moving C to Generated

Uniform-cost search	
Generated	Path
D(6)	S(0)
E(6)	B(2)
A(7)	C(3)
G(7)	F(5)

After moving F to Generated

Uniform-cost search	
Generated	Path
E(6)	S(0)
A(7)	B(2)
G(7)	C(3)
E(11)	F(5)
	D(6)

After moving D to Generated

Uniform-cost search	
Generated	Path
A(7)	S(0)
G(7)	B(2)
	C(3)
	F(5)
	D(6)
	E(6)

After moving E to Generated

Uniform-cost search	
Generated	Path
G(7)	S(0)
	B(2)
	C(3)
	F(5)
	D(6)
	E(6)
	A(7)

After moving A to Generated

Uniform-cost search	
Generated	Path
	S(0)
	B(2)
	C(3)
	F(5)
	D(6)
	E(6)
	A(7)
	G(7)

Goal G is reached

The steps to calculate the cost for all the nodes based on the associated evaluation function f(n)=h(n) for greedy best-first search. The A* search find out the optimal path with the evaluation function f(m) = g(m) + h(m) as following steps in tables:

Greedy FBS			A* Search				
Generated		Path	Generated				Path
Node	h(n)		Node	g(n)	h(n)	f(n)	
S	10		S	0	10	10	

Start by adding S to the Path

Greedy FBS			A* Search				
Generated		Path	Generated				Path
Node	h(n)		Node	g(n)	h(n)	f(n)	
B	7	S	B	2	7	9	S
A	9		A	7	9	16	

After moving S to Path

Greedy FBS			A* Search				
Generated		Path	Generated				Path
Node	h(n)		Node	g(n)	h(n)	f(n)	
C	6	S	C	3	6	9	S
D	8	B	D	6	8	14	B
A	9		A	7	9	16	

After moving B to Path

Greedy FBS			A* Search				
Generated		Path	Generated				Path
Node	h(n)		Node	g(n)	h(n)	f(n)	
F	3	S	F	5	3	8	S
E	6	B	E	6	6	12	B
D	8	C	D	6	8	14	C
A	9		A	7	9	16	

After moving C to Path

Greedy FBS			A* Search				
Generated		Path	Generated				Path
Node	h(n)		Node	g(n)	h(n)	f(n)	
G	0	S	G	7	0	7	S
E	6	B	E	6	6	12	B
D	8	C	D	6	8	14	C
A	9	F	A	7	9	16	F

After moving F to Path

Greedy FBS			A* Search				
Generated		Path	Generated				Path
Node	h(n)		Node	g(n)	h(n)	f(n)	
E	6	S	E	6	6		S
D	8	B	D	6	8	12	B
A	9	C	A	7	9	14	C
		F				16	F
		G					G

After moving G to Path and G is goal.

In the previous operations, the individual working explained with step by step performances for uniform cost search, greedy best-first search and A* search. It is clear that there are 8 nodes in the path of uniform cost search and informed with 5 nodes to reach the goal state. The informed search finds the solution with minimum cost more quickly. The main deviation between GBFS and A* BFS depends on the definition of the evaluation function.

In the informed search, although both greedy best-first search and A* search algorithm achieve same path to get the goal, it is noticeable that the A* algorithm is able to find optimal solution out more exactly with an additional information. The evaluation function of Greedy BFS is f (m) = h (m) while the evaluation function for A* is f (m) = g (m) + h (m).

In some problem, greedy BFS returns sub-optimal solution due to the incomplete information of the heuristic method. Actually, both greedy best first and A* search are best-first search but greedy best first is neither complete, nor optimal. The heuristic function h (m) is an admissible that supports the A* search do not exceed estimation the cost to reach the goal.

And then, the memory bounded heuristic strategies have development based on uninformed search technologies using cutoff value. This is accomplished by storing the generated nodes in memory and pruning some of them away when the memory is full.

4. Conclusion and future work

This article had presented the informed search is more efficient and acceptable than uninformed search (blind search). The uninformed search technique has lack of knowledge about the problem. The strength of informed search is developing the solution in optimality as well as completeness by using well-formed heuristic function. Among heuristic search, the memory bounded-algorithms are developed to reduce memory requirements.

This article describes gradually development in search technology. It is the evident in the role of heuristic. Actually our approach is general, and then this article concludes the ability of the heuristic method is more imperative features than data structure searching technology in AI. Consideration of future work will be how to innovative the heuristic methods based on the problem characteristic including cognitive neuroscience that demonstrate the relevance of the field to general AI such as machine learning, deep learning, robotics, and data science, domain-independent planning, and computer games, etc.

Acknowledgment

We would like to acknowledge my senior and associates who gave us a lot of valuable advice and encouragement. Finally, we also thankful to all admirable colleagues for supporting contributed towards the success of this research.

References

[1] Deepika Garg, COMPARATIVE STUDY OF VARIOUS SEARCHING ALGORITHMS, National Conference on Innovative Trends in Computer Science Engineering (ITCSE-2015) held at BRCMCET, Bahal on 4th April 2015.

[2] C. Zhang, S. Bengio, M. Hardt, B. Recht, Vinyals and O, “Understanding Deep Learning Requires Re-thinking Generalization,” in ICLR, 2017.

[3] Y. e. a. Bengio, “Learning deep architectures for AI,” Foundations and trends® in Machine Learning, 2(1), p. 1–127, 2009.

[4] Maharshi J. Pathak, Ronit L. Patel, Sonal P. Rami, “Comparative Analysis of Search Algorithms”, International Journal of Computer Applications (0975 – 8887) Volume 179 – No.50, June 2018.

[5] Deepika Garg, COMPARATIVE STUDY OF VARIOUS SEARCHING ALGORITHMS, National Conference on Innovative Trends in Computer Science Engineering (ITCSE-2015) held at BRCMCET, Bahal on 4th April 2015.

[6] Chandel, and Manu Sood, Searching and Optimization Techniques in Artificial Intelligence: A Comparative Study & Complexity Analysis, International Journal of Advanced Research in Computer Engineering & Technology (IJARCET) Volume 3 Issue 3, March 2014.

[7] Algfoor, Zeyad Abd, Mohd Shahrizal Sunar, and Hoshang Kolivand. "A comprehensive study on pathfinding techniques for robotics and video games." International Journal of Computer Games Technology 2015 (2015):7.

[8] Camilo Caraveo,Fevrier Valdez,•Oscar Castillo,”A new optimization meta-heuristic algorithm based on self-defense mechanism of the plants with three reproduction operators,“Springer-VerlagGmbH Germany, part of Springer Nature 2018

[9] Mr. Girish P Potdar, and Dr.R C Thool, COMPARISON OF VARIOUS HEURISTIC SEARCH TECHNIQUES FOR FINDING SHORTEST PATH, International Journal of Artificial Intelligence & Applications (IJAIA), Vol. 5, No. 4, July 2014.

[10] Stuart J. Russell and Peter Norvig, “Artificial Intelligence, A Modern Approach” , Second Edition

[11] Russell, S. (1992). "Efficient memory-bounded search methods". In Neumann, B. (ed.). Proceedings of the 10th European Conference on Artificial intelligence. Vienna, Austria: John Wiley & Sons, New York, NY. pp. 1–5. CiteSeerX 10.1.1.105.7839

[12] Simone Guercini,“Heuristics as tales from the field: the problem of scope,” Springer-Verlag GmbH Germany, part of Springer Nature 2019

[13] Richter, S., and Helmert, M. 2009. Preferred operators and deferred evaluation in satisficing planning. In Gerevini, A.; Howe, A.; Cesta, A.; and Refanidis, I., eds., Proceedings of the Nineteenth International Conference on Automated Planning and Scheduling (ICAPS 2009), 273–280. AAAI Press.

[14] https://www.mygreatlearning.com/blog/best-first-search-bfs/