ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 4, Special Issue 21, August 2017

A COMPARISON STUDY ON VARIOUS HEURISTIC SEARCH TECHNIQUES

M.Ganesh Babu[1] , Homer Benny Bandela[2], DonavalliVenkataVidya Deepthi[3]

1. Assistant Professor, Dept. of C.S.E, Sir C.R.Reddy College of Engineering, Eluru, Andhra Pradesh, India.

2. Assistant Professor, Dept. of C.S.E, Sir C.R. Reddy College of Engineering, Eluru, Andhra Pradesh, India.

3. Assistant Professor, Dept. of C.S.E, Sir C.R. Reddy College of Engineering, Eluru, Andhra Pradesh, India.

---*---

ABSTRACT

There are various algorithm are used for different purpose and were observed in their optimality and simplicity with speed. Heuristic search techniques make use of problem specific knowledge to find efficient solutions. Most of these techniques determine the next best possible state leading towards the goal state by using evaluation function. This paper shows the practical performance of the different heuristic algorithm. While implementing these algorithms, this analysis helps in choosing the algorithm which effects the performance of algorithms significantly.

Keywords: Informed search techniques, Heuristic function, Heuristic algorithm.

1.INTRODUCTION

Heuristic search algorithms have exponential time and space complexities as they store complete information of the path including the explored intermediate nodes. Hence many applications involving heuristic search techniquesto find optimal solutions tend to be expensive. Despite of these, the researchers have strived to find optimal solution in best possible time. In this paper we have considered major algorithms which are applied to find the shortest path: hill – climbing, steepest –ascent, best first and A* [1,2,4].

Hill climbing algorithms expand the most promising descendant of the most recently expanded node until they encounter the solution. Steepest – ascent hill climbing differs from hill climbing algorithm only the way in which the next node is selected. In this method it selects best successor node for expansion, unlike the first successor node for expansion, as done in hill climbing. Though this method tries to choose best possible path , but this method , like hill climbing method may fail to find a solution by reaching to a node from were no improvements can be done [5,8]. Best first search method selects the "best" node for further expansion by applying a

heuristic function. It then generates the successor node in similar fashion till the goal node is reached. This technique tries to explore the advantages of breadth first and depth first search technique and provides better time bound solution. Best first algorithm involves OR graph, it avoids the node duplication and also works on the assumption that each node has parent link to give the best node from the node where it is derived and link to successors. A* algorithm is a slight modified version of best search algorithm. The difference is that in A* the estimate to the goal state is given by heuristic function and also it makes use of the cost of the path developed [2,3,6].

We will now discuss each of these methods for finding the shortest path.

2.HILL CLIMBING METHOD FOR SHORTEST PATH FINDING

Hill climbing algorithm expands one node at a time beginning with the initial node. Each time it expands only the best node reachable from current node. Thus this method does not involve complex computation and due to this reason cannot ensure the completeness of the solution. Hill climbing method does not give a solution as may terminate without

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 4, Special Issue 21, August 2017

reaching the goal state [12].Now let us look at algorithm of hill climbing for finding shortest path:

Procedure for hill climbing algorithm to find the shortest path:

```
hill_climb (I, F, Q)
{
// I& F are start and goal nodes respectively.
// Q is queue which stores the successor
// nodes.
// let curr_node indicate current working
//   node.
// path _cost gives the cost of the path.
initialiseQ; curr_node = I; path_cost=0; while (1)
{
if (curr_node is goal node) then terminate the process with SUCCESS;
else
{
find successor node of curr_node;
addthis node in Q ;
}
if(Q is empty)then
terminate the process with FAILURE;
else
{
temp_node = first node of Q ;
path_cost = path_cost  +
edge_cost [curr_node][temp_node];
curr_node = temp_node;
delete first node from Q ;
}
}
```

One may notice that there can be failure state when algorithm may fail to reach the goal node. This will happen especially when the processing has reached to a node from where no new best nodes are available for further expansion. This will happen especially when the processing has reached to a node from where no new best nodes are available for further expansion.

3. STEEPEST ASCENT HILL CLIMBING METHOD FOR SHORTEST PATH FINDING

This method is a result of variation in hill climbing. Here, instead of moving the immediate best node, all the reachable nodes from current node are considered and among these the best one is chosen. In case of simple hill climbing, the first successor node which is better, is selected, due to this we may omit the best one. On the contrary steepest ascent hill climbing method not only reaches to the better state but also climbs up the steepest slope.

The variation in algorithm will be only in finding the best successors node from all the possible successor nodes from all possible successor, and not just the first best node [2,12,15]. [7] proposed a system in which the cross-diamond search algorithm employs two diamond search patterns (a large and small) and a halfway-stop technique. It finds small motion vectors with fewer search points than the DS algorithm while maintaining similar or even better search quality. The efficient Three Step Search (E3SS) algorithm requires less computation and performs better in terms of PSNR. Modified objected block-base vector search algorithm (MOBS) fully utilizes the correlations existing in motion vectors to reduce the computations. Fast Objected - Base Efficient (FOBE) Three Step Search algorithm combines E3SS and MOBS. By combining these two existing algorithms CDS and MOBS, a new algorithm is proposed with reduced computational complexity without degradation in quality.

One can notice that hill climbing and steepest – hill climbing may fail to find a solution. Either algorithm may not reach goal node as it may reach to a node where we may not find better nodes. In such cases we may need to back-track as use more rules before choosing the next node. However this process will be time consuming.

Both the methods discussed, may terminate not by finding a goal node but may reach node from where no better nodes can be generated.

This will happen if the processing has reached to one of the following situations:

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 4, Special Issue 21, August 2017

i) A node might have been selected which may be better that its neighbors, however there may be few better nodesavailable which are step away. This situation is termed as local maxima.

ii) A node might have been selected, whose neighbors may have the same value and hence choosing next best node is difficult. This is known as plateau.

iii) A ridge is a special kind of local maximum, though the path selected so far may be the best, yet making further moves difficult.

The next algorithms described here try to overcome these problems.

4. BEST FIRST METHOD FOR SHORTEST PATH FINDING

Best first search is a type of graph search algorithm. Here the nodes are expanded one at time by choosing lowest evaluation value. This evaluation value is a result of heuristic function giving a measure of distance to the goal node. For typical applications such as shortest path problems, the evaluation function will be accurate as it accounts for distance or an absolute value [14,19].

Best first search is a combination of breadth and depth first search. Depth first search has an advantage of arriving at solution without computing all nodes, whereas breadth first arriving at solution without search ensured that the process does not get trapped. Best-first search, being combination of these two, permits switching between paths. At every stage the nodes among the generated ones, the best suitable node is selected for further expansion, may be this node belong to the same level or different, thus can toggle between depth-first and breadth-first. This method involves OR graph, avoids node duplication, and also requires two separate lists for processing. OPEN list keeps the nodes whose heuristic values are determined, but yet to be expanded. CLOSE list have the nodes which have been already checked, further these nodes are kept in this list to ensure no duplications. It implies that the OPEN list has the nodes which need to be considered for further processing and the entries in CLOSE list indicate the nodes which may not be re-required in further steps [6,7].

5. A* ALGORITHM FOR SHORTEST PATH FINDING

We know that the various search techniques are designed, tested and are being used for various purposes whatever it is for system software or application software. But the base for this is however mainly because of the problems in planning domain. Classical approaches to heuristic search algorithm work on assumption of the existence of deterministic model of sequential decision making leading to the solution. The research work focused on solving planning problems under uncertainty [1]. Heuristic algorithms have given a new looked into the problems belonging to this domain [6,10].

The shortest path problem can be solved by A* algorithm. The heuristic function needs to evaluate two costs, g and h. Let g(n), in shortest path problem, represent cost of choosing the path from starting node to node n; and h(n) represents optimal cost of node n to the goal node. Now the cost of node n is given by: $f^*(n) = g(n) + h^*(n)$. However the value of $h^*(n)$ will be unknown in most of the situations, which results in unknown value of $f^*(n)$. A* algorithm, however makes a best approximation for $h^*(n)$[16,17].

The A* algorithm to solve the shortest path problem can be written as: [10]

Step 1: Start from the start node; place it in OPEN list. This will be current working node. Step 2: Explore all the nodes adjacent to the one in OPEN list.

Step 3: Determine the cost function for all the nodes obtained in step 2; and place them in OPEN

list in increasing order of cost function values.

Step 4: Move current working node, from OPEN list to CLOSE list.

Step 5: Now the first node in OPEN List will be the current working node (which is having least cost function due to insertion criteria in step 3).

Step 6: If this current working node is not the goal state (final node), then repeat step 2 to step 5. Step 7: The CLOSE list gives the shortest path and the value of last cost function obtained gives the optimal cost.

6. EXPERIMENTAL RESULTS

All the algorithms discussed in previous sections were implemented in C++ and run on 2.4 GHz Intel C2D system with 2GB RAM. The random data sets were created for varying number of input nodes and saved in separate files. While testing these algorithms stored data was given as input data and processed. The algorithms were tested for the number nodes and edges explored/visited were compared. The Number of nodes and edges considered during the process for various algorithms are given in Table 1 and Table 2 respectively.

ST_HC --Steepest Ascent, BFS—Best First Search and A*.

Table 1: Number of nodes considered.

Nodes	Hill Climbing	ST-HC	BFS	A*
1200	2400	1211	1900	1100
1400	2800	1326	2400	1188
1600	3050	1500	250	200
1800	3400	1800	2908	1500
2000	3800	1965	2800	1567
2200	4300	2200	3562	2100
2400	4890	2391	1800	1000

The resulting graphs of the two algorithms are given in Fig 1

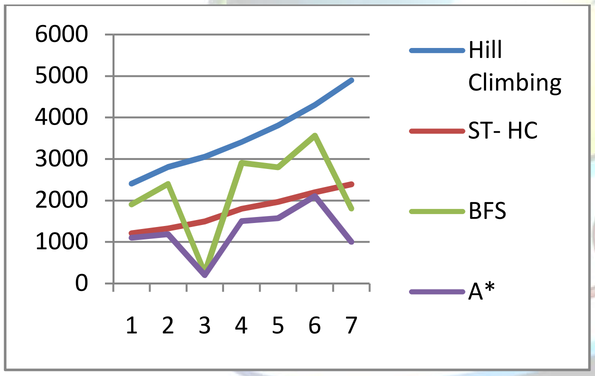

Figure 1: Comparison of number nodes considered against total nodes in graph.

Table-1 shows that there is significant amount of improvement on number of nodes being considered in Hill climbing algorithm compared to the rest of the methods.

One may also observe here that certain unexpected variations in the values. This is mainly due to the fact that these algorithms were executed till they find the solution and were not run for fixed number of iterations.

10 CONCLUSION

We have presented major class of heuristic algorithms. The comparison shows that though all these algorithms can be applied to find the shortest path, but should not be used unless there is a real-time, event driven actions are anticipated. The comparison gives us clear idea that best-first search and A* algorithms are very well suitable when goal node cannot be reached from all nodes. However there may be interesting scenarios that may come out when these algorithms are applied with different data structures.

REFERENCES

[1] BlaiBonet and Eric A. Hansen, (2010)"Heuristic Search for Planning under Uncertainty", Chapter in Heuristics, Probability and Causality: A Tribute to Judea Pearl College Publications. pp 3-22

[2] Eric A Hansen, Rong Zhou, (2007) "Anytime Heuristic Search", Journal of Artificial Intelligence Research 28, pp 267-297

[3] G.Cornuejols and G L Nemauser, (1978) "Tight bounds for christofides" travelling salesman heuristic* Short Communication Mathematical Programming, Vol. 14, Issue 1, pp 116-121

[4] Anne L. Gardner, (Sept 1980) "Search: An Overview", AI magazine, Vol. 2, Number 1

[5] R. Korf, (1990) "Real time heuristic search", Artificial Intelligence ACM Digital Library, Vol. 42, pp189-211

[6] RinaDechter and Judia Pearl, (July 1985) "Generalized Best-First Search Strategies and the Optimality of A*.", Journal of the Association for Computing Machinery, Vol. 32, No. 3, pp 505-536

[7] Christo Ananth, A.Sujitha Nandhini, A.Subha

Shree, S.V.Ramyaa, J.Princess, “Fobe Algorithm for Video Processing”, International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering (IJAREEIE), Vol. 3, Issue 3,March 2014 , pp 7569-7574

[8] Girish P. PotdarandDr.R.C.Thool, (2013) “An Alternate way of implementing Heuristic Searching Technique” International Journal of Research in Computer andCommunication Technology, Vol. 2, No 9, pp-793-795

[9] Hen-Yong Pang, Alicia Tang Y.C., (2006) “A Route Advisory System (RAS) For Travelling

Salesman Problem”, Journal of Applied Sciences Research 2(1), pp 34-38

[10] C.H. Peng, J.S. Wangand R.C.T. Lee, (1994)“Recognizing Shortest Path Trees in Linear Time”, Information Processing Letters, Vol. 57, pp 77-85

[11] Potdar, Girish P., and R. C. Thool. "Comparison of Various Heuristic Search Techniques for Finding Shortest Path." International Journal of Artificial Intelligence & Applications 5.4 (2014): 63.