Volume 6, Issue 9, September 2016 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Games and their Search Algorithms

Vinay Chandragiri
Department of Computer Science and Engineering, IIT Guwahati,
Assam, India

Abstract—This idea of this paper is to explore the real time search algorithms for path planning in static terrain which is required in video games.[2] A new time division model is implemented, where uniformly partitioned time intervals are present, one movement during each time interval can be executed by the agent, and search and movements happens parallelly. The task is to minimize the number of time intervals to move the agent from start state to goal state. The experimental results suggest that Time bounded A* (TBA*), a real time search algorithm for undirected terrain matches with the performance of A* algorithm for known terrain (same number of time intervals for solving), but it cannot be used when the terrain is unknown initially. This paper also presents experimental results which suggests that Time bounded adaptive A*(TBAA*), a combined version of TBA* and adaptive A* algorithms, moves the agent from start state to goal state on cost minimal path in minimum possible intervals if a path between them exists else detects the non-existent property. In the domain of path-finding on video game maps TBA* expands an order of magnitude fewer states than traditional real-time search algorithms, while finding paths of comparable quality. It reaches the same level of performance as recent state-of-the-art real-time search algorithms but, unlike these, requires neither state-space abstractions nor precomputed pattern databases.

Keywords— A*, Game time model, Path Planning with the Freespace Assumption, Time-Bounded A*, Restarting Time Bounded A*, Time-Bounded Adaptive A*, Video Games

I. INTRODUCTION

Game developers generally restrict the planning time to 1-3ms for all simultaneous path-finding units which is very less for real time search algorithms to give high quality solutions.[1] This resulted in the development of more advanced real-time heuristic search algorithms which boosts their performance with the help of abstractions and precomputation mechanisms. Game characters executes one movement per game cycle in video games so game time model is introduced where movement per time interval takes place along with parallel execution of search and movement. Complete search algorithms, such as A* search first and only then move the agent along the resulting path, needs several time intervals for determining the complete path which creates a long delay before the start of agent movement. On the other hand real time search algorithms overcome this by performing search and movement parallelly.

Real time search algorithms operates on known terrains as well as initially partially or unknown terrains. For unknown terrain they initially assume that the terrain is free from obstacles. TBA* algorithm picks the smallest f-value state from the OPEN list at the end of each time interval to execute the movement towards that best path location to reach goal state(if path exists). But it cannot be used when terrain is unknown. For this conversion of it to on line path planning will be helpful. In this we extend the TBA* to RTBA* which is identical to TBA* except that when the agent encounters an obstacle on it's current path to a state in OPEN list with minimum f-value, it starts a new A* search from the current location towards goal location. It then continues the execution of movement in a similar way towards goal state. This RTBA* is further extended to time bounded adaptive A* (TBAA*) in which the updating of h-values takes place in order to help the future A* searches perform better. This papers shows experimentally that TBAA* actually moves the agent from it's start state to goal state in unknown terrain in minimum time intervals than the real time search algorithms and equally with D* lite.

II. NOTATION OF SEARCH PROBLEM

A search problem is tuple consisting of finite weighed digraph with start and goal states along with cost values associated to each edge. In this current paper we introduce an assumption that the graph is undirected giving rise to equal edge cost from s to t and t to s. As in A* algorithm g represents the cost of the cost minimal path from start state to current state, h represents the cost of cost minimal path from current state to goal state, and f is the sum of above two

III. TIME BOUNDED A*

The search problem we use in this is grid consisting of both blocked and unblocked cells along with provided start and goal states. Movement from unblocked cells to blocked cells is restricted in this. The agent can move to any of the four possible directional unblocked cells namely (North, South, East and West). The edge is defined as a neighbouring connection from one unblocked cell to other unblocked cardial cell having cost value equal to 1. The objective is to move the agent from start location to goal location. TBA* can be applied on search problems with known cost functions. This algorithm executes A* search from start state towards goal state. After each time interval it chooses the smallest f-value state from the open list it keeps track of(using priority queue). For simplification we assume the time per state expansion which depends on the length of open list is constant also the time per movement determination which depends on the length of the path is zero which results in a constant number of state expansions in an interval.

IV. ALGORITHM DESCRIPTION

- The h and g values are initialized to user given h value and infinity when required for the first time. (h is the manhattan distance)
- Search number is current A* search number and Search of a state is the search number of A* search in which g value is initialized last.
- Expand open list states until it becomes empty.
- Pick the minimum f-value state and add the neighbours of it to open list which are not visited yet.
- If current state lies on the path obtained by the search function then movement to the next state on the path from current state takes place.
- If current state doesn't lie on the above mentioned path then movement takes place towards parent state.
- Search and movement occurs parallelly in the same time interval.

V. RESTARTING TIME BOUNDED A*

As TBA* cannot be applied to search problems with unknown cost functions, it is extended to a on-line path planning algorithm called RTBA*. In this we use a restarting A* search implementation. Generally TBA* requires the edge cost before the traversal so for unknown terrains if lower bounds of edge costs are provided then the agent executes the movement per time interval based on the lower bound cost. By this the edge cost may change once from lower bound cost to actual cost. In the process always the edge cost increases as the assumed costs are lower bound values.

Initially it doesn't get to know about the blocked cells in the grid so it assumes all the cells as unblocked cells. Thus the edge cost of all the edges assumed to be 1 initially. During the movement of agent if it finds any cell as blocked one it increases the edge cost of all the incoming and outgoing edges of that cell to infinity. At this point the agent can execute the TBA* search again but it might end up unnecessarily exiting the current A* search again and again. So a better way to handle this is by restarting the TBA* search only when the edge costs on path from current state to best possible state have increased while running the current A* search. This restart of TBA* gave this algorithm the name Restarting time bounded A* search.

VI. TIME BOUNDED ADAPTIVE A*

The information RTBA* stores while performing search is lost whenever it restarts a new A* search. As some of the real time search algorithms solves the issue by updating the cost of cost minimal path from current state to goal state (h) value to make use of it. Thus we consider TBAA* a combined version of RTBA* and a lazy implementation of Adaptive A*. So whenever RTBA* starts a new A* search the updating of h-values of all the generated states of previous search happens. The new h-value would be the difference between f-value and g-value of that state.

TBA* Algorithm

```
  Function Search() expansions ← 0
  while openlist not equal to 0ANDexpansions < kANDg(goal) + h(goal) > minOPEN(g(t) + h(t)) do
    s = argminOPEN(h(t) + g(t))
    remove_s_from_OPEN t 2 Succ(s)
    Initialize State(t)
    if g(t) > g(s) + c(s, t)
         then g(t) g(s) + c(s; t)
      parent(t) ← s
      insert_t_into OPEN
    end if
    expansions ← expansions + 1
    end while
  if OPEN = then
  return false
  end if
  S_best = argminOPEN(g(t) + h(t))
  if S_best = S_goal
  then flag 1
 end if
path_from_root_to_best
return true
```

RTBA* Algorithm

Start_NewSearch() :

```
if edge_cost_increased then
    Initialize_Search()
    path ← empty
end if
Move To Goal() :
currentstate ← startstate
Initialize_Search()
while current not equal to goal do
if flag = 0 then
     if search = false then
      return false
  end if
  Start_New_Search()
end if
 if path not equal to empty then
    if current_is_on_path then
   current = next_state
  else
  current = parent(s)
 end if
   if edge_cost_increases then
   start_new_search
  end if
 end if
end while
```

TBAA* Algorithm

```
Initialize_State() :
if search(s) = 0 then
  $h(s) \leftarrow H(s)$
  $g(s) \leftarrow \infty$
else if search(s) not equal to searchnumber then
  if $h(s) \leq pathcost(search(s)g(s))$ then
    $h(s) \leftarrow pathcost(search(s) - g(s))$
  end if
 $g(s) \leftarrow \infty$
end if
search(s) = searchnumber
procedure StartNewSearch() :
if edge_costs_on_path_increased then
   pathcost(searchnumber) = min(g(s) + h(s))
   Initialize_search()
    path = empty
end if
```

VII. EXPERIMENTAL RESULTS

I use three spacecraft maps namely Enigma, Inferno, Wheel Of War and three Dragon age maps namely orz103d, orz702d, orz703d. The paper compares the TBA*, RTBA* and TBAA* search algorithms by running them on different search problems of game map

A. Results with TBA* algorithm

Game map	Time intervals	Movements
Orz103d	1583	866
Enigma	382	381
Inferno	796	795
orz702d	1287	1286
orz703d	111	110
Wheel	3396	3395

VIII. CONCLUSION

Analyzed the TBA*, RTBA*, and TBAA* real time search algorithms performance with the introduction of game time models for 6 different game maps consisting of 300 search problems each. A* search gives better results in known terrain game map where the objective is to move the agent from start state to goal state. However TBA* achieves the same objective in minimum time intervals than that of RTAA* and daRTAA* search algorithms.

REFERENCES

[1] Y. Bjornsson, V. Bulitko, and N. R. Sturtevant. Tba*: Time-bounded a*.

[2] C. Hernandez, J. Baier, T. Uras, and S. Koenig. Time-bounded adaptive ´ a*. In Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 2, pages 997–1006. International Foundation for Autonomous Agents and Multiagent Systems, 2012.