Important thing for search

Zhongliang Guo

December 2, 2021

1 Introduction

Just list the thing I think it is important but hard to remember. If want to use it, you need to fully understand all things in lectures 16–19.

2 pseudo-code

This section is the pseudo code for the template search algorithms.

Listing 1: **pseudo code for search algorithm template**

```
0.: initialize
    agenda = [ [start] ]          # a list of paths or partial solutions
    [extended_list = {};]         # optional: closed list of nodes
    while agenda is not empty:
       1. path = agenda.pop(0)    # remove first element from agenda
       2. if is-path-to-goal(path, goal)
              return path
       3. otherwise extend path [if the ending node is not in extended_list]
              [add the ending node to extended_list]
              for each connected node (child node of the ending node of path)
                      make a new path that extends the connected node
       4. reject  all new paths from step 3 that contain cycles
       5. add new paths from step 4 to agenda and reorganise agenda
                                  # algorithms differ here!
6. return failure
```

Change part 5 can create new algorithms.

3 Comparison of different search algorithms

	Complete	Optimal	Time complexity	Space complexcity	agenda reorganisation
DFS	No	No	$O(b^d)$	$O(bd)$	paths added to the front agenda; or stack agenda
BFS	Yes	No in general	$O(b^d)$	$O(b^d)$	paths added to the back of agenda; or queue agenda
NDS	No	No	$O(b^d)$	$O(b^d)$	paths randomly inserted to the agenda
UCS	Yes	Yes	$O(b^{C^*/\epsilon})$	$O(b^{C^*/\epsilon})$	paths added agenda then sort the agenda based on path costs (optional: trim down agenda by only retaining the most promising ones, i.e. if there are multiple paths ending with the same node, keep the one with the optimal cost); or PriorityQueue agenda

For UCS, C^* is the optimal cost, ϵ is the lower bound of action costs,
So the optimal depth $d = C^*/\epsilon$