Review 10/18/2016

COSC 6368: Artificial Intelligence

COSC 6368

Midterm Exam

Tuesday, October 26, 2004



*Name:*

*SSN:*

1. A\* & Best-first Search (15 points)
2. More Search and EC (15 points)
3. Decision Trees (13 points):
4. FOPL as a Language (7 points):
5. Resolution (10 points)

Point Total (out of 60):

Number Grade:

The exam is “open books and notes” and you have 75 minutes to complete the exam. Write all your answers on this document.

**1) Best first search and A\* [15]**

Consider the search space below, where *S* is the start node and *G1* and *G2* satisfy the goal test. Arcs are labeled with the cost of traversing them and the estimated cost to a goal (h function) is reported inside nodes.

For each of the following search strategies, indicate which goal state is reached (if any) and list, *in order*, all the states *popped off of the OPEN list*. When all else is equal, nodes should be removed from OPEN in alphabetical order.

##### Best-First-Search (using function h only) [3]

 Goal state reached: \_\_\_\_\_G2\_\_ States popped off OPEN: S, A, E, G2\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

##### A\* (using f=g+h)[4]

 Goal state reached: S2\_\_\_\_\_\_\_ States popped off OPEN: S, A, E, D, G2\_\_\_\_\_\_\_\_\_\_\_\_\_

2

1

7

5

2

5

4

1

2

2

1

1

8

9

3

d) Assume you have 2 admissible heuristics h1(s) and h2(s) are given for a given seach problem. Is h3(s)=min(h1(s),h2(s)) also admissible? Would you prefer using h2 or using h3 in conjuction with A\*? Give reasons for your anwers[4].

Yes, h3 is admissible. If h1 and h2 always underestimate the “true” cost then the lesser of the two will certainly underestimate the true cost as well; therefore, h3 is admissible.

I will prefer h2, because h2 is always greater equal than h3 and therefore it provides a closer approximation of the true cost. As a matter of fact, h2 dominates h3, which translates into equal or better efficiency of the search, as discussed on the bottom of page 106 of our textbook.

1. Assume you apply randomized hill climbing to a minimization involving a continuous, differentiable function that has 3 minima. Will it always find the optimal solution? Give reasons for your answer! [3]

No, HC might climb down the wrong minimum depending on the chosen starting point

b) What is the “main” difference between simulated annealing and randomized hill climbing? [2]

… SA does allow downward steps…

d) Assume you apply a version of backtracking that checks for repeated states on the current path[[1]](#footnote-1), but which does not use a depth bound to the 8-puzzle. Will it always find a solution if a solution exists (assuming that there are enough computational resources)? [6]

Yes!

Because the search-tree for the 8-puzzle is finite and because the algorithm checks for loops in the current path the algorithm will sooner or later stop moving forward, backtrack, and find the solution eventually.

### H1 Problem 3: Termination of A\*

### H1 Problem 4: Comparision of Seach Algorithms

Compare Backtracking, Traditional Hill Climbing/Randomized Hill Climbing and Best-first Search! What are the main differences between the three approaches?

n be the size of the search space, the number of nodes in the search tree

b the branching factor, the number of successors

m is the length of the longest path in the search tree

|  |  |  |  |
| --- | --- | --- | --- |
|  | Backtracking | Randomized HC | Best First Search |
| The way they search | Explore a single path at a time | Explore a single path | Can explore multiple paths in general |
|  | Moves Forward along a single path and backtracks  | Only moves forward, cannot move backward | Jumps between states |
| Storage | O(b\*logb(n)) | O(1) / O(m)[[2]](#footnote-2) | O(n) |
| Runtime  | O(n) in the worst case, quite fast  | O(m) but might stop prematurely, fast | O(n) in the worst case, not fast |
| Finding solutions in finite search spaces | Yes, if run with a depthbound d≥m | might terminate prematurely; might go into the wrong direction and get stuck | Yes |
| Find global optimum | no, terminates prematurely | no | no, terminates permaturely |
| Parameter Selection | Straight forward | Choosing neighbor hood size and sampling rate for RHC is challenging | Straight Forward |
| Incorperating Heuristics  | Needs good operatior selection funciton | Needs good state evaluation function | Need good state evaluation function  |
| Other  | deterministic | RHC is a probabilistic algorithm (usually) returns different solutions in different runs | deterministic |

Problem1 2001 AI Final Exam

1) Heuristic Search [9]

Assume A\* is applied to the 8-puzzle with a state evaluation
function f'(s)=g'(s)+h'(s) with:

* g'(s):= "The number of moves applied to reach s from
the initial state."
* h'(s):= "The number of incorrect positions
of s with respect to the goal state g (not counting ‘\*’)"

For example, for the initial state ini given below f'(ini)=0+6=6.

Moreover, we assume that the state with the lowest value is considered
to be the best state and that in cases of ties random selection is used.

6 2 3 4 6 2
8 5 \* 1 5 \*
4 1 7 3 4 7
 ini g

a) Assume A\* is applied to the above search problem assuming ini is the initial state and g is the goal state. How does A\*'s search graph look like after 3 nodes have been expanded? Also give for each state s in the search tree its f’(s) value! [5]

b) Does A\* ---  if used with f' --- always find the shortest solution, if a solution exists? Give reasons for your answer! [4]

### H1 Problem 5: Games; *look a the TA’s solution; you can ask questions on Thursday*

### H1 Problem 1: Define a Search Space; *will not receive a problem like this in the midterm exam, but might receive such a problem for the final exam!*

### H1, Problem 7: Planning *Dr. Eick will give a short review on planning on Oct. 20*

###

H1 Problem 6: Determine Nash Equilibrium: See Nguyen Solution



1. This version backtracks, if a loop in the current path is encountered. [↑](#footnote-ref-1)
2. Only if it is necessary to return the solution path, as in the WRKBK problem! [↑](#footnote-ref-2)