Dr. Eick

Artificial Intelligence

COSC 6368

Solution Sketches

Midterm Exam

Tuesday, October 25, 2016



*Name:*

*Student id:*

1. A\* & Best-first Search (10 points)
2. Backtracking, Hill Climbing and SA (13 points)
3. Game Theory (5 points)
4. Planning (9 points)
5. Designing a State Evaluation Function (7 points)
6. Games and Adversarial Search (6 points)

Point Total (out of 50):

Number Grade:

The exam is “open books and notes”, but no computers and cell phones allowed; you have 75 minutes to complete the exam. The exam is slightly too long: you are expected to solve 90% of the exam problems! Write all your answers on this document (you can use back sides!).

**1) Best first Search and A\* [10]**

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 (the h function itself) 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.

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

Goal state reached: G2 [1]

States popped off OPEN: S, E, G2 [2]

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

Goal state reached: S2 [1]

States popped off OPEN: S, A, B, G1 [3]

7

2

1

2

1

5

2

9

2

3

8

4

1

4

5

c) Assume 2 admissible heuristics h1(s) and h2(s) are given for a given seach problem. Is h3(s)=max(h1(s),h2(s)) also admissible? Would you prefer using h2 or using h3 in conjuction with A\*? Give reasons for your anwers! [3]

h3(s) is bounded by the higher value of h1(s) and h2(s); as both functions are admissible neither h2(s) not h3(s) overestimate the true cost to reach the goal-state⇒ Thus h3 is admissible. [1.5]

Prefer h3: provides a closer estimation of the true cost as h3(s) ≥h1(s) and h3(s) ≥h2(s) ⇒can result in the lower number of nodes expansions as the goal state is popped of earlier due to the closer approximation to the “true” cost.

**2) Backtracking, SA, and Hill Climbing [13]**

a) Assume you have to apply hill-climbing to a minimization problem, in which we like to find a minimum (a low value) for the following function f in

[0,10]x[0,10]x[0,10]x[0,10]:

f(a,b,c,d)= |a+b-0.5|+|a\*a+d\*b-0.7|+ |1.5 –a\*(1-b\*c-d\*(a+0.2))|

Design a hill climbing[[1]](#footnote-1) algorithm that solves the above problem! Describe how your algorithm will solve the above problem. [7]

Any Hill Climbing variation will be accepted, given that:

* Clearly define how to generate the candidates, for example: coordinate-wise fixed (or random) step size, using acceleration, or random offset.
* Clearly define the stop condition: no significant improvement, or reaching the limit of number of iterations.
* **Common mistake**: Generate some candidates (i.e. 100), select the best f value, repeat and replace the f value if there is improvement. This is not H.C: the candidate must be the chosen as the base for the next iteration, the step must be small enough to only look at the “surrounding” of the current base.

b) What role does temperature play when using Simulated Annealing (SA)? Why is the temperature quite high early when running SA but quite low near the end of the run of SA? [4]

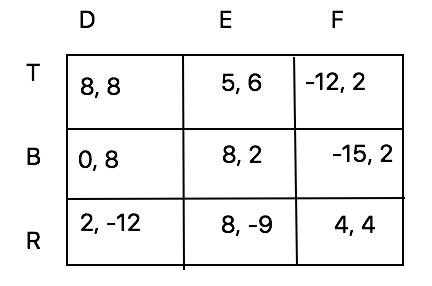
Temperature controls the degree of **exploration** and **exploitation** that occurs during the search. At the beginning, high temperature allows for traversing non-promising states to avoid to get stuck in a “bad” local optimal hill. Towards the end, temperature gradually becomes lower, to focus on reaching the actual maximum of the currently explored hill rather than on switching from one hill to a more promising other hill.

c) What advantages does Backtracking have—if compared to Best-first Search? [2]

* Require lower storage [1]
* Other answers might receive credit!

**3) Game Theory [5]**

What is the Nash Equilibrium for the following game, whose payoff matrix is depicted below [3]?



(4,4) and (8,8) [3]. If making one mistake [1]

What is the main property of a Nash Equilibrium? [2]

Citing the standard text is the best solution.

Some mistakes:

* Nash Equilibrium is the optimum (No, it is not).
* Did not mention: “exactly/any one player changes...” or “while all other...” This is important, Nash Equilibrium cannot say anything about the rewards when more than one players change their strategies.



**4) Planning [9]**

a) What is the frame problem in planning? How do STRIPS-like systems “solve” this problem? [3]

It provides add and delete lists to describe operators and makes the ‘closed world assumption; consequently, the truth values of facts that are not mentioned in the add and delete-list remain unchanged.

b) What, do you believe, are the main contributions of the IJCAI best paper award paper by Javier Segovia, Sergio Jimenez and Anders Jonsson? Limit your answer to 4-6 sentences! [6]

Things you could mention include:

* Proposed finite state controllers that effectively represent sequential plans compactly
* The hierarchical FSC represents plans more compactly than stand-alone, non-hierarchical FSCs
* Can implement recursion and define iteration less complicated
* Reformulation of the transition function of FSCs to allow binary branching
* Hierarchical FSCs are created incrementally
* They support compilation for classical planning , making it possible to use off the shelf classical planners to generate hierarchical FSCs
* The paper demonstrated convincingly that they compilation approach to generate hierarchical planners that can branch on any fluent since the proposed approach considers all fluents, in contrast to previous approaches.
* Use prior knowledge in form of existing FSCs to automatically complete the definition of the ,remaining FSCs

1. **Finding a Good State Evaluation Function for Heuristic Search [7]**

Assume a **non-finite maze** is given in which a robot has the task to move from an initial position (0,0) to a goal position. The available operators are north, south, east, and west that move the robot one field in the indicated direction. However, sometimes the robot cannot move in a particular direction e.g. if the robot faces a wall in the north it cannot move north.

(0,0) (0,1) (0,2) (0,3) (0,4)…

(1,0) (1,1) (1,2) (1,3) (1,4)…

(2,0) (2,1) (2,2) (2,3) (2,4)…

(3,0) (3,1) (3,2) (3,3) (3,4)…

(4,0) (4,1) (4,2) (4,3) (4,4)…

……………………………..

Assume best-first search is used for this particular problem, and it is known that the goal position is either in position (5000,1000) or (999, 999). Design a state evaluation function that searches the space efficiently and always finds a path to the goal state (be aware of the fact the search space is non-finite and that that there might be obstacles that might make it more difficult to reach a goal position), if such a path exists. You can use if-then-else and any function of your liking when specifying h!

h(x,y):= min(|x – 5000| + |y – 1000|, |x – 999| + |y – 999|)

As we have 2 accepted goals, the heuristic only needs to estimate the distance to the nearest goal. Given the grid world, it is simple enough to use the Manhattan distance (Euclidean distance is also accepted).

h(x,y) defined above is admissible: in the best case, it is the optimal traveling distance to nearest goal. If there are obstacles obstructing the path, the traveling cost can only be longer.

When the above h is used, they agent switches between paths leading to both goal states eventually reaching one of the goal states.

Briefly explain how your evaluation function will search the space, and why it will always reach the goal state.

The search algorithm is straight forward from the standard A\* tree search, with a slight modification. To find the next states from the current state, check all four directions for obstacle. If a direction is obstacle-free, put that state into the Open List. Because f is an admissible heuristic, A\* tree-search is guarantee to find the optimal solution.

**6) Games and Adversarial Search [6]**

a) Assume you use the minmax algorithm for a game instead of alpha-beta algorithm. Will the moves generated by the minmax algorithm differ from those that were selected by the alpha-beta algorithm? Give a reason for your answer. [2]

**No** difference [1]

Alpha-beta just prunes the unnecessary evaluation branches [1]

b) What are the main challenges in using Adversarial Search for card games, if compared with using Adverserial Search for games with complete information, such as Go or Chess? [4]

* The problem is **not fully observable**. [1]
* Build the probabilistic search tree, where nodes are possible combinations for all hands, and calculate estimated evaluation[1]
  + Difficult to calculate the probability of any combination [1], as this not any relies on probability theory, but only the history of cards played and information obtained from bidding. [1]
  + A lot of domain specific knowledge that is challenging to computerize [1]
  + Good bridge player use concealment by playing the card that gives the least information to the opponents, but automating this capability is quite challenging.

1. Any hill climbing approach is fine! [↑](#footnote-ref-1)