CS 495 & 540: Problem Set 2

Section: MW 12-1:50 pm

Total: 220pts Out: 10/03/2018 Due: 10/24/2018

Instructions:
1. I leave plenty of space on each page for your computation. If you need more sheet, please attach your work right behind the corresponding problem. If your answer is incorrect but you show the computation process, then partial credits will be given. Please staple your solution and use the space wisely.

2. This assignment contains two parts. Part I, due on 10/24 at the beginning of the class, is the written part while Part II, due on 10/31 at midnight on blackboard, is the programming part. Part I counts for 170/220 of the assignment. You are allowed to work on the homework in a group of size up to five. No late assignment is accepted. Identical solutions (same wording, paragraph, code), turned in by different groups (persons), will be considered cheating.

3. Full credit will be given only to the correct solution which is described clearly. Convoluted and obtuse descriptions might receive low marks, even when they are correct. Also, aim for concise solutions, as it will save you time spent on write-ups, and also help you conceptualize the key idea of the problem.

First Name:

Last Name:

Group ID:

Score: **/170** **Programming:** **/50** **Midterm Bonus:** **/10**

Problem 1 α, β Pruning: 10 pts

Please come up with your own binary tree (let say the height is 5, i.e. max-min-max-min-value) such that you have two alpha cuts and 1 beta cut. Please justify your designed tree does have those three cuts.

Problem 2 Concept: BFS, DFS, Iterative Deepening: 20pts

Given a balanced tree with branching factor = b and height = m. Please draw a simple graph (along with some brief description) for each sub question:
(a) BFS outperforms DFS in terms of time complexity

(b) DFS outperforms BFS int terms of time complexity

(c) Iterative Deepening gets defeated by DFS in terms of time complexity

(d) Iterative Deepening outperforms DFS in terms of time complexity

Problem 3 Heuristic Path Algorithm: 20pts

The **heuristic path algorithm**(Pohl. 1977) is a best-first search in which the evaluation function is $f(n) = (2 - \omega)g(n) + \omega h(n)$.
(a) For what values of ω is this **complete**?

(b) For what value is it **optimal**, assuming that h is admissible?

(c-e) What kind of search does this perform for
(c) $\omega = 0$

(d) $\omega = 1$

(e) $\omega = 2$?

Problem 4 Tracing: BFS, DFS, Iterative Deepening : 20 pts

Consider a state space where the start state is number 1 and each state k has two successors: numbers $2k$ (left child) and $2k + 1$ (right child).
(a) Draw the portion of the state space for states 1 to 15.

(b) Suppose the goal state is 11. List the order in which the nodes will be visited by
(I) BFS

(II) DFS

(III) iterative deepening search (6pts)

(c) Call the action going from k to $2k$ Left and the action going to $2k+1$ Right. Can you find an algorithm that outputs the solution to this problem without any search at all?

Problem 5 Adversarial Search: 10pts

Prove the following assertion: For every game tree, the utility obtained by MAX using minimax decisions against a suboptimal MIN will be never be lower than the utility obtained playing against an optimal MIN. Can you come up with a game tree in which MAX can do still better using a suboptimal strategy against a suboptimal MIN?

Problem 6 Adversarial Search: 20pts

Consider the two-player game described in Figure 5.17 in the textbook.
(a) Draw complete game tree, using the following conventions:

- Write each state as (S_A, S_B), where S_A and S_B denote the token locations.
- Put each terminal state in a square box and write its game value in a circle
- Put *loopstates*(states that already appear on the path to the root) in double square boxes. Since their value is unclear, annotate each with a "?"' in a circle.

(b) Mark each node with its backed-up minimax value (also in a circle). Explain how you handled the "?" values and why.

(c) Explain why the standard minimax algorithm would fail on this game tree and briefly sketch how you would fix it, drawing on your answer to (b). Does your modified algorithm give optimal decisions for all games with loops?

(d) This 4-square game can be generalized to n squares for any $n > 2$. Prove that A wins if n is even and loses if n is odd.

Problem 7 Adversarial Search: tic-tac-toe: 20pts

Consider the family of generalized tic-tac-toe games, defined as follows. Each particular game is specified by a set S of squares and a collection W of winning positions. Each winning position is a subset of S For example, in standard tic-tac-toe, S is a set of 9 squares and W is a collection of 8 subsets of W: the three rows, the three columns, and the two diagonals. In other respects, the game is identical to standard tic-tac-toe. Starting from an empty board, players alternate placing their marks on an empty square. A player who marks every square in a winning position wins the game. It is a tie if all squares are marked and neither player has won.
a. Let $N = |S|$, the number of squares. Give an upper bound on the number of nodes in the complete game tree for generalized tic-tac-toe as a function of N.

b. Give a lower bound on the size of the game tree for the worst case, where $W = \{\}$.

c. Propose a plausible evaluation function that can be used for any instance of generalized tic-tac-toe. The function may depend on S and W.

d. Assume that it is possible to generate a new board and check whether it is a winning position in $100N$ machine instructions and assume a 2 gigahertz processor. Ignore memory limitations. Using your estimate in (a), roughly how large a game tree can be completely solved by alphabeta in a second of CPU time? a minute? an hour?

Problem 8 ExpectedMax: 20 pts

This question considers pruning in games with chance nodes. Figure 5.19 shows the complete game tree for a trivial game. Assume that the leaf nodes are to be evaluated in left- to-right order, and that before a leaf node is evaluated, we know nothing about its valuethe range of possible values is $-\infty$ to ∞.
a. Copy the figure, mark the value of all the internal nodes, and indicate the best move at the root with an arrow.

b. Given the values of the first six leaves, do we need to evaluate the seventh and eighth leaves? Given the values of the first seven leaves, do we need to evaluate the eighth leaf? Explain your answers.

c. Suppose the leaf node values are known to lie between 2 and 2 inclusive. After the first two leaves are evaluated, what is the value range for the left-hand chance node?

d. Circle all the leaves that need not be evaluated under the assumption in (c).

Problem 9 CSP: Coloring: 5+3+2pts

Consider the Figure 6.1 in the textbook.
(a) How many solutions are there for the map-coloring problem, suppose 3 colors are allowed?

(b) How many solutions if four colors are allowed?

(c) How many solutions if two colors are allowed?

Problem 10 CSP: Most Constrained Variable 10pts

Explain why it is a good heuristic to choose the variable that is most constrained but the value that is least constraining in a CSP search.

Problem 11 CSP: K-Consistency 10pts

(a) Use the AC-3 algorithm to show that arc consistency can detect the inconsistency of the partial assignment $\{WA = green, V = red\}$ for the problem shown in Figure 6.1.

(b) Briefly describe the complexity cost for K-consistency

Problem 12 Programming Simulated Annealing: 50pts

Based on the given 3-SAT dimac file (100 variables ($v_1 v_2 \cdots v_{100}$ that is a binary string with100 bits), 425 clauses), please use 1010101...10 as the initial assignment to the 3-SAT. If an assignment completely satisfies this 3-SAT, its score should be 425.
(a) Run Hill-climbing and report local maximal (if not global) and the corresponding assignment value (turn that binary assignment into base 10 please).

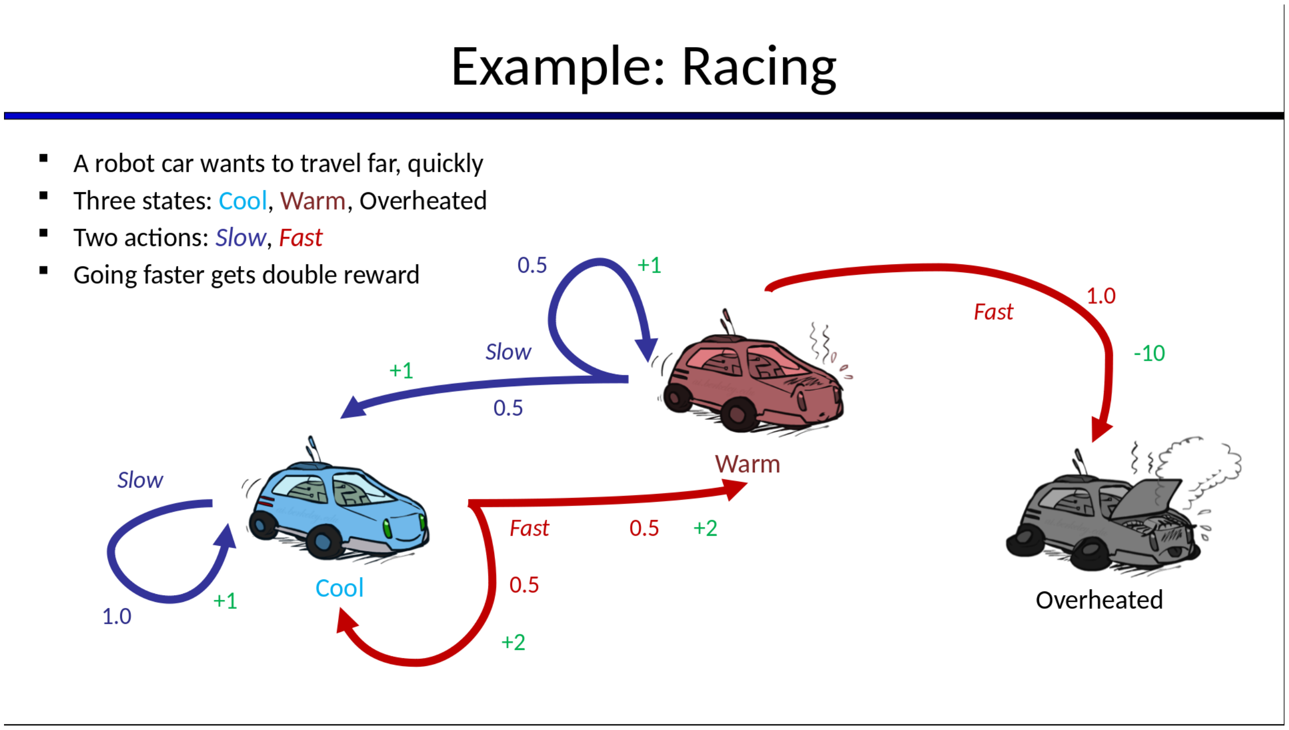
(b) Run Simulated Annealing on this to dodge the local optimal. Let us have two temperature cooling schemes, one goes down fast (let say 100 slots in the $[0, 1]$ range), one goes down much slower (let say 1500 slots in the $[0, 1]$ range). Is the slower cooling process producing a better solution that has a higher score? And what is your strategy of picking up the unsatisfied variable and why?

Problem 13 Programming MDP: 10 pts bonus for midterm

Please simulate the MDP using the value iteration for the scenario below and here we will have depreciation rate γ set to 0.8. Let k be the number of iterations. Please simulate
(a) (3pts) till $k = 50$ and report the corresponding value for
$V^*_{50}(Cold) =$
$V^*_{50}(Warm) =$
$V^*_{50}(Overheated) =$

(b) (3pts) $k = 100$ and report the corresponding value for
$V^*_{100}(Cold) =$
$V^*_{100}(Warm) =$
$V^*_{100}(Overheated) =$

(c) (4pts) Simulate policy iteration for this scenario.



Example: Racing

- A robot car wants to travel far, quickly
- Three states: Cool, Warm, Overheated
- Two actions: *Slow*, *Fast*
- Going faster gets double reward