NAME:_____ **SID#:**_____ **Login:**_____ **Sec:**_____

CS 188 Fall 2007 Introduction to Artificial Intelligence Midterm Solutions

You have 80 minutes. The exam is closed book, closed notes except a one-page crib sheet, basic calculators only. 80 points total. Don't panic!

Mark your answers ON THE EXAM ITSELF. Write your name, SID, login, and section number at the top of each page.

For true/false questions, CIRCLE *True* OR *False*.

If you are not sure of your answer you may wish to provide a *brief* explanation. All short answer sections can be successfully answered in a few sentences at most.

1. (18 points.) True/False

(a) *True/False*: If one search heuristic $h_1(s)$ is admissible and another one $h_2(s)$ is inadmissible, then $h_3(s) = \min(h_1(s), h_2(s))$ will be admissible.

True. $h_3(s) \leq h_1(s) \leq h^*(s)$.

(b) *True/False*: Greedy search has the same worst-case number of node expansions as DFS.

True. Both can expand the entire state space. With $h(s) = 0$, greedy might behave exactly like DFS.

(c) *True/False*: In A*, the first path to the goal which is added to the fringe will always be optimal.

False. The first path *removed* from the fringe is optimal with an admissible heuristic.

(d) *True/False*: If a CSP is arc consistent, it can be solved without backtracking.

False. Arc consistency may not determine the entire solution.

(e) *True/False*: A CSP with only binary constraints can be solved in time polynomial in n and d, the number of variables and size of the domains.

False. Binary CSPs are NP-hard. You may assume P is NP.

(f) *True/False*: The minimax value of a state is always less than or equal to the expectimax value of that state.

True. Expectimax allows for suboptimal behavior from minimizing agents, which can only raise the value of a node in the game tree.

(g) *True/False*: Alpha-beta pruning can alter the computed minimax value of the root of a game search tree.

False. Alpha-beta pruning only speeds up computation; it does not change the answer.

(h) *True/False*: When doing alpha-beta pruning on a game tree which is traversed from left to right, the leftmost branch will never be pruned.

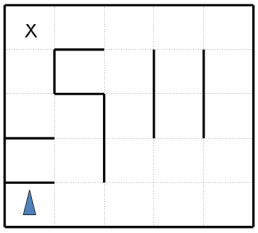
True. There are no alternatives to motivate pruning at the left-most branch of a game tree.

(i) *True/False*: Every search problem can be expressed as an MDP with at most as many states as the original search problem.

True. Search problems are MDPs with deterministic transitions, a terminal state, negative rewards, and no discount rate.

2. (18 points.) Search and Heuristics

Imagine a car-like agent wishes to exit a maze like the one shown below:



The agent is directional and at all times faces some direction $d \in (N, S, E, W)$. With a single action, the agent can *either* move forward at an adjustable velocity v *or* turn. The turning actions are *left* and *right*, which change the agent's direction by 90 degrees. Turning is only permitted when the velocity is zero (and leaves it at zero). The moving actions are *fast* and *slow*. *Fast* increments the velocity by 1 and *slow* decrements the velocity by 1; in both cases the agent then moves a number of squares equal to its NEW adjusted velocity. Any action which would collide with a wall crashes the agent and is illegal. Any action which would reduce v below 0 or above a maximum speed V_{max} is also illegal. The agent's goal is to find a plan which parks it (stationary) on the exit square using as few actions (time steps) as possible.

As an example: if the agent shown were initially stationary, it might first turn to the east using (*right*), then move one square east using *fast*, then two more squares east using *fast* again. The agent will of course have to *slow* to turn.

(a) (3 points) If the grid is M by N, what is the size of the state space? Justify your answer. You should assume that all configurations are reachable from the start state.

$M \times N \times 4 \times (V_{max} + 1)$

The state is specified by the position, direction and velocity.

(b) (2 points) What is the maximum branching factor of this problem? You may assume that illegal actions are simply not returned by the successor function. Briefly justify your answer.

3 is the maximum branching factor. When stopped, the agent can turn left, turn right, or go fast.

(c) (3 points) Is the Manhattan distance from the agent's location to the exit's location admissible? Why or why not?

Manhattan distance is *not admissible* because the agent can travel faster than 1 square per move.

(d) (4 points) State and justify a non-trivial admissible heuristic for this problem which is not the Manhattan distance to the exit.

(a) Manhattan distance divided by V_{max}

(b) Current velocity

(e) (2 points) If we used an inadmissible heuristic in A* tree search, could it change the completeness of the search?

An inadmissible heuristic will not change completeness.

(f) (2 points) If we used an inadmissible heuristic in A* tree search, could it change the optimality of the search?

An inadmissible heuristic does not imply optimality, so a suboptimal solution could be found.

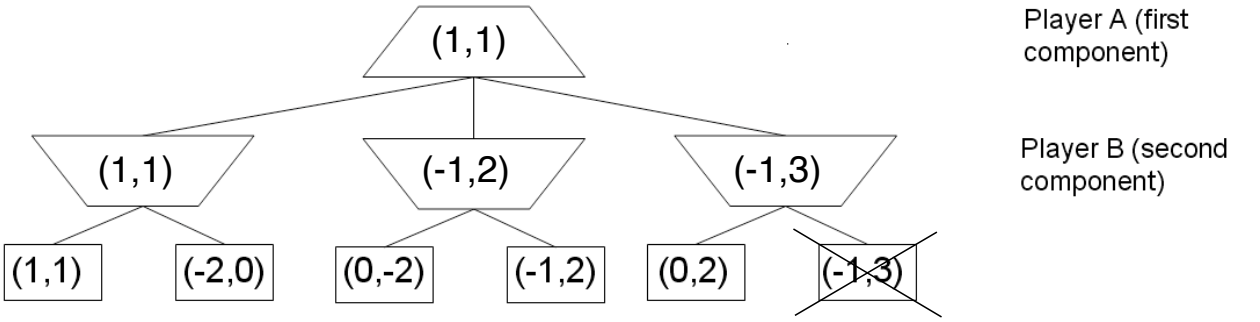
(g) (2 points) Give a general advantage that an inadmissible heuristic might have over admissible one.

Inadmissible heuristics often find solutions faster (expanding fewer nodes), although those solutions need not be optimal.

3. (16 points.) Game Search

The standard Minimax algorithm calculates worst-case values in a *zero-sum* two player game, i.e. a game in which for all terminal states s, the utilities for players A (MAX) and B (MIN) obey $U_A(s) + U_B(s) = 0$. In the zero sum case, we know that $U_A(s) = -U_B(s)$ and so we can think of player B as simply minimizing $U_A(s)$.

In this problem, you will consider the *non zero-sum* generalization in which the sum of the two players' utilities are not necessarily zero. Because player A's utility no longer determines player B's utility exactly, the leaf utilities are written as pairs (U_A, U_B), with the first and second component indicating the utility of that leaf to A and B respectively. In this generalized setting, A seeks to maximize U_A, the first component, while B seeks to **maximize** U_B, the second component.



(a) (4 points) Propagate the terminal utility pairs up the tree using the appropriate generalization of the minimax algorithm on this game tree. Fill in the values (as pairs) at each of the internal node. Assume that each player maximizes their own utility. *Hint:* just as in minimax, the utility pair for a node is the utility pair of one of its children.

(b) (2 points) Briefly explain why no alpha-beta style pruning is possible in the general non-zero sum case. *Hint:* think first about the case where $U_A(s) = U_B(s)$ for all nodes.

The values that the first and second player are trying to maximize are independent, so we no longer have situations where we know that one player will never let another player down the branch of the game tree. For instance, in the case where $U_A(s) = U_B(s)$, this problem reduces to searching for the max valued leaf, which could appear anywhere in the tree.

For minimax, we know that the value v computed at the root (say for player A = MAX) is a worse-case value, in the sense that, if the opponent MIN doesn't act optimally, the actual outcome v' for MAX can only be better, never worse, than v.

(c) (3 points) In the general non-zero sum setup, can we say that the value U_A computed at the root for player A is also a worst-case value in this sense, or can A's outcome be worse than the computed U_A if B plays suboptimally? Briefly justify.

A's outcome can be worse than the computed U_A. For instance, in the example game, if B chooses (-2,0) over (-1,2), then A's outcome will decrease from 1 to -1.

Now consider the *nearly zero sum* case, in which $|U_A(s) + U_B(s)| \leq \epsilon$ at all terminal nodes s for some ϵ which is known in advance. For example, the previous game tree is nearly zero sum for $\epsilon = 2$.

(d) (3 points) In the nearly zero sum case, pruning is possible. Draw an X in each node in this game tree which could be pruned with the appropriate generalization of alpha-beta pruning. Assume that the exploration is being done in the standard left to right depth-first order and the value of ϵ is known to be 2. Make sure you make use of ϵ in your reasoning.

(e) (2 points) Give a general condition under which a child n of a B node (MIN node) b can be pruned. Your condition should generalize α-pruning and should be stated in terms of quantities such as the utilities $U_A(s)$ and/or $U_B(s)$ of relevant nodes s in the game tree, the bound ϵ, and so on. Do not worry about ties.

$U_b > \epsilon - \alpha$

(f) (3 points) In the nearly zero sum case with bound ϵ, what guarantee, if any, can we make for the actual outcome u' for player A (in terms of the value U_A of the root) in the case where player B acts suboptimally?

$u' \geq U_A - 2\epsilon$

4. (15 points.) MDPs and Reinforcement Learning

In Flipper's Folly, a player tries to predict the total number of heads in two coin flips. The game proceeds as follows (also shown below):

(a) From the start state (XX), choose the special action *begin* (only possible action)

(b) Flip a coin and observe the result, arriving in the state HX or TX

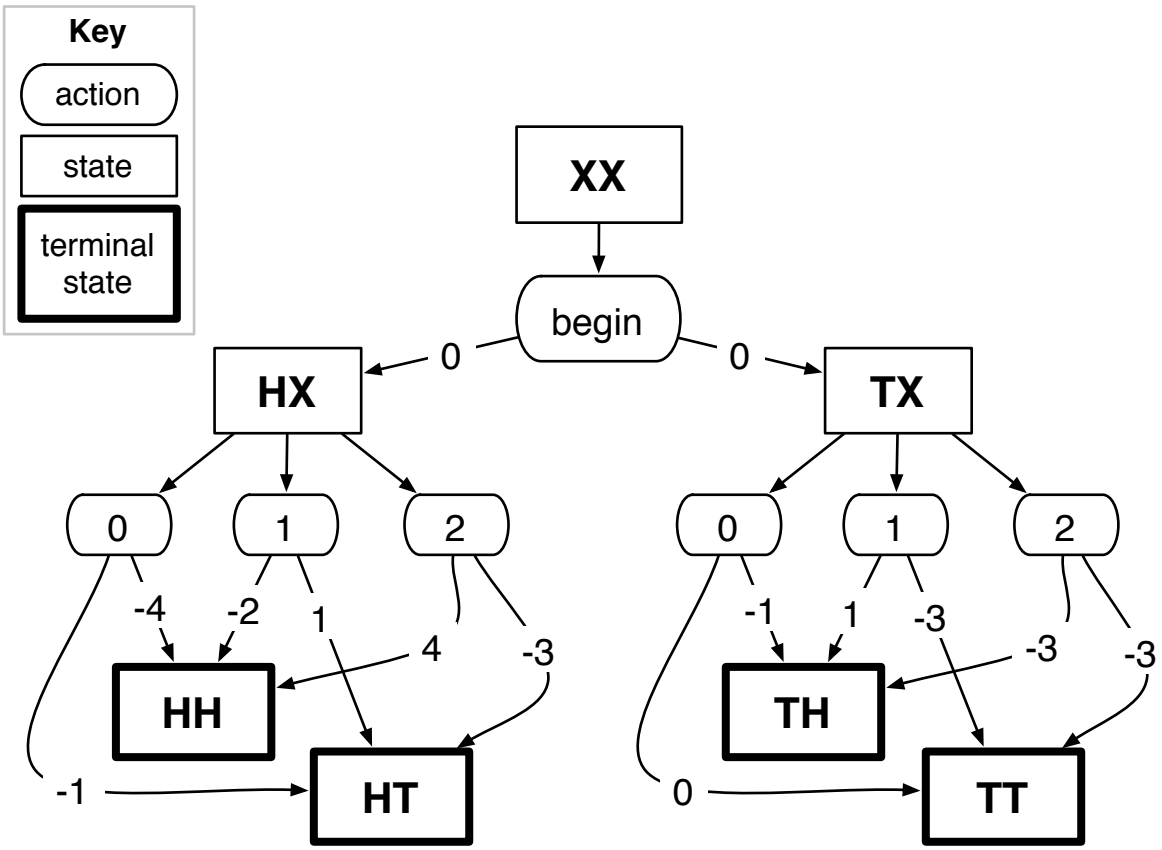
(c) Guess what the total number of heads will be: $a \in \{0, 1, 2\}$

(d) Flip a coin and observe the result, arriving in one of the states HH, HT, TH, TT.

(e) Count the total number of heads in the two flips; $c \in \{0, 1, 2\}$

(f) Receive reward $R(s, a, s') = \begin{cases} 2 \cdot a^2 - c^2 & \text{if } c \geq a \\ -3 & \text{if } c < a \end{cases}$ where c is the total number of heads in s'

Note that the rewards depend only on the action and the landing state, and that all rewards for leaving the start state are zero. The MDP for this game has the following structure, where all legal transitions have probability $\frac{1}{2}$. **Assume a discount rate of 1.**



(a) (3 points) What is the value of the start state under the policy of always guessing $a = 2$?

$\frac{1}{2} \cdot \left[\frac{1}{2}(4-3) + \frac{1}{2}(-3-3)\right] = -\frac{5}{4}$

(b) (5 points) Run value iteration on this MDP until convergence. *Hint*: values and q-values of terminal states are always 0.

	$V_k^*(s)$		
k	XX	HX	TX
0	0	0	0
1	0	$\frac{1}{2}$	$-\frac{1}{2}$
2			
3			
4			
5			

Value iteration converges after one iteration.

(c) (2 points) What is the optimal policy for this MDP?

$\pi^*(XX) = begin, \pi^*(HX) = 2, \pi^*(TX) = 0$

(d) (5 points) Run q-learning in this MDP with the following (s, a, s', r) observations. Use a learning rate of $\frac{1}{2}$. Leave zero entries blank.

				$Q(s,a)$						
s	a	s'	r	(XX,begin)	(HX,0)	(HX,1)	(HX,2)	(TX,0)	(TX,1)	(TX,2)
				0	0	0	0	0	0	0
XX	begin	HX	0							
HX	0	HT	-1		$-\frac{1}{2}$					
XX	begin	HX	0		$-\frac{1}{2}$					
HX	2	HH	4		$-\frac{1}{2}$		2			
XX	begin	HX	0	1	$-\frac{1}{2}$		2			

5. (13 points.) Short Answer

Each question can be answered in a single sentence!

(a) (2 pts) For A* search, why might we prefer a heuristic which expands more nodes over one which expands fewer nodes?

First, an admissible heuristic might expand more nodes than an inadmissible heuristic, but would be guaranteed to find the optimal answer. Second, even among two admissible heuristics, we might prefer an looser heuristic that is easier to compute per state.

(b) (2 pts) Why is minimax less reasonable than expectimax as a practical decision-making principle for a complex agent acting in the real world?

Expectimax handles very low probability events robustly, while minimax will be overly sensitive to low probability highly negative outcomes (getting hit be meteors, etc.).

(c) (4 pts) An agent prefers to be given an envelope containing \$4 rather than one containing either \$0 and \$10 (with equal probability). Give a justification for how the agent could be acting in accordance with the principle of maximum expected utility.

An agent might value \$4 almost as much as \$10. Utility of money often scales non-linearly.

(d) (4 pts) A *stochastic policy* π is one which does not recommend a single, deterministic action for each state s, but rather gives each possible action a a probability. Let $\pi(s, a)$ be the probability that the policy assigns to action a from state s. State a one-step lookahead Bellman equation for $V^{\pi}(s)$ for the case of stochastic policies π.

$V^{\pi}(s) = \sum_a \pi(s, a) \sum_{s'} T(s, a, s')[R(s, a, s') + \gamma V^{\pi}(s')]$

(e) (3 pts) Under what conditions can an MDP be solved using standard state space search techniques (DFS, BFS, etc.)?

When (i) all transitions are deterministic, (ii) all rewards are non-posititive (corresponding to non-negative step costs), (iii) there is no discount, and (iv) there is at least one terminal state (the goal state).