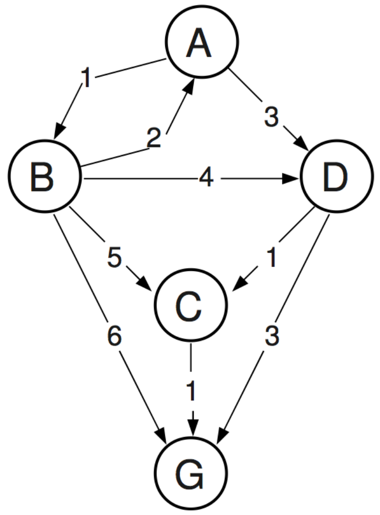
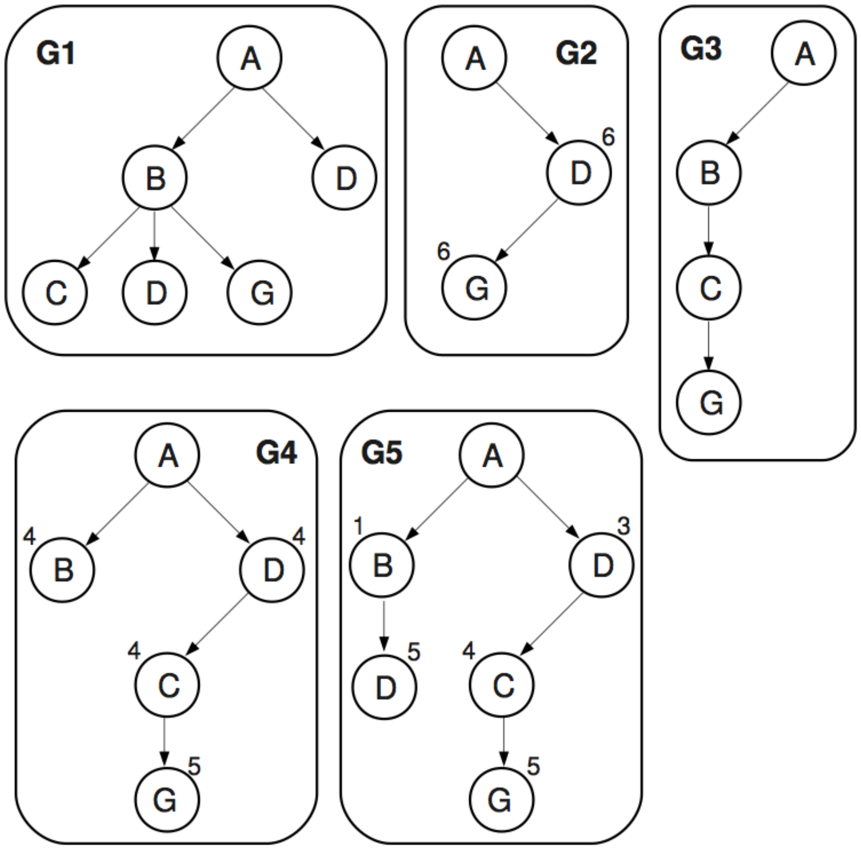
CS 188 Spring 2010 — Introduction to Artificial Intelligence — Midterm Exam Solutions

Q1. [15 pts] Search Traces

Each of the trees (G1 through G5) was generated by searching the graph (below, left) with a *graph search* algorithm. Assume children of a node are visited in alphabetical order. Each tree shows *only the nodes that have been expanded.* Numbers next to nodes indicate the relevant "score" used by the algorithm's priority queue. The start state is A, and the goal state is G.





For each tree, indicate:

1. Whether it was generated with depth first search, breadth first search, uniform cost search, or A^* search. Algorithms may appear more than once.
2. *If* the algorithm uses a heuristic function, say whether we used

   H1 = $\{h(A) = 3,\ h(B) = 6,\ h(C) = 4,\ h(D) = 3\}$

   H2 = $\{h(A) = 3,\ h(B) = 3,\ h(C) = 0,\ h(D) = 1\}$
3. For all algorithms, say whether the result was an optimal path (assuming we want to minimize sum of link costs). If the result was *not* optimal, state why the algorithm found a suboptimal path.

Please fill in your answers on the next page.

(a) [3 pts] **G1**:

1. Algorithm: Breadth-First search
2. Heuristic (if any): None
3. Did it find least-cost path? If not, why? No. Breadth-first search will only find a path with the minimum number of edges. It does not consider edge cost at all.

(b) [3 pts] **G2**:

1. Algorithm: A^* search
2. Heuristic (if any): H1
3. Did it find least-cost path? If not, why? No. A^* search is only guaranteed to find an optimal solution if the heuristic is admissible. $H1$ is not admissible.

(c) [3 pts] **G3**:

1. Algorithm: Depth-First Search
2. Heuristic (if any): None
3. Did it find least-cost path? If not, why? No. Depth first search simply finds any solution - there are no guarantees of optimality.

(d) [3 pts] **G4**:

1. Algorithm: A^* search
2. Heuristic (if any): H2
3. Did it find least-cost path? If not, why? Yes. H2 is an admissible heuristic; therefore, A^* finds the optimal solution.

(e) [3 pts] **G5**:

1. Algorithm: Uniform Cost Search
2. Heuristic (if any): None
3. Did it find least-cost path? If not, why? Yes. Uniform cost search is guaranteed to find a shortest-cost path.

Q2. [16 pts] Multiple-choice and short-answer questions

In the following problems please choose **all** the answers that apply, if any. You may circle more than one answer. You may also circle no answers (none of the above)

(a) [2 pts] Consider two consistent heuristics, H_1 and H_2, in an A^* search seeking to minimize path costs in a graph. Assume ties don't occur in the priority queue. If $H_1(s) \leq H_2(s)$ for all s, then

(i) A^* search using H_1 will find a lower cost path than A^* search using H_2.

(ii) A^* search using H_2 will find a lower cost path than A^* search using H_1.

(iii) A^* search using H_1 will not expand more nodes than A^* search using H_2.

(iv) A^* search using H_2 will not expand more nodes than A^* search using H_1.

(iv). Since H_2 is less optimistic, it returns values closer to the real cost to go, and thereby better guides the search. Heuristics do not affect the length of the path found – A^* will eventually find the optimal path for an admissible heuristic.

(b) [2 pts] Alpha-beta pruning:

(i) May not find the minimax optimal strategy.

(ii) Prunes the same number of subtrees independent of the order in which successor states are expanded.

(iii) Generally requires more run-time than minimax on the same game tree.

None of these are true. Alpha-beta will always find the optimal strategy for players playing optimally. If a heuristic is available, we can expand nodes in an order that maximizes pruning. Alpha-beta will require less run-time than minimax except in contrived cases.

(c) [2 pts] Value iteration:

(i) Is a model-free method for finding optimal policies.

(ii) Is sensitive to local optima.

(iii) Is tedious to do by hand.

(iv) Is guaranteed to converge when the discount factor satisfies $0 < \gamma < 1$.

(iii) and (iv). Value iteration requires a model (an specified MDP), and is not sensitive to getting stuck in local optima.

(d) [2 pts] Bayes nets:

(i) Have an associated directed, acyclic graph.

(ii) Encode conditional independence assertions among random variables.

(iii) Generally require less storage than the full joint distribution.

(iv) Make the assumption that all parents of a single child are independent given the child.

(i), (ii), and (iii) – all three are true statements. (iv) is false – given the child, the parents are not independent.

(e) [2 pts] True or false? If a heuristic is admissible, it is also consistent.
False, this is not necessarily true. Admissible heuristics are a subset of consistent heuristics.

(f) [2 pts] If we use an ϵ-greedy exploration policy with Q-learning, the estimates Q_t are guaranteed to converge to Q^* only if:

(i) ϵ goes to zero as t goes to infinity, or

(ii) the learning rate α goes to zero as t goes to infinity, or

(iii) both α and ϵ go to zero.

(ii). The learning rate must approach 0 as $t \to \infty$ in order for convergence to be guaranteed. Note that Q-learning learns off policy (in other words, it learns about the optimal policy, even if the policy being executed is sub-opitimal). This means that ϵ need not approach zero for convergence.

(g) [2 pts] True or false? Suppose X and Y are correlated random variables. Then

$$P(X = x, Y = y) = P(X = x)P(Y = y|X = x)$$

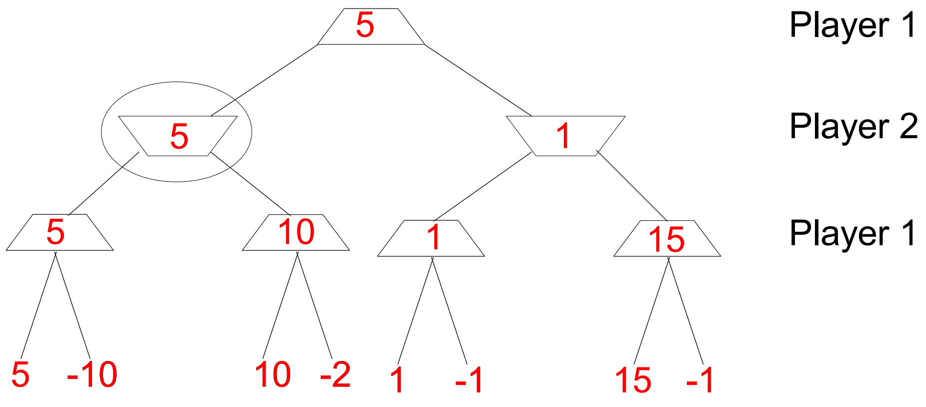
True. This is the product rule.

(h) [2 pts] When searching a zero-sum game tree, what are the advantages and drawbacks (if any) of using an evaluation function? How would you utilize it?

We can use an evaluation function by treating non-terminal nodes of a certain depth as terminal nodes with values given by the evaluation function. This allows us to search in games with arbitrarily large state spaces, but at the cost of sub-optimality.

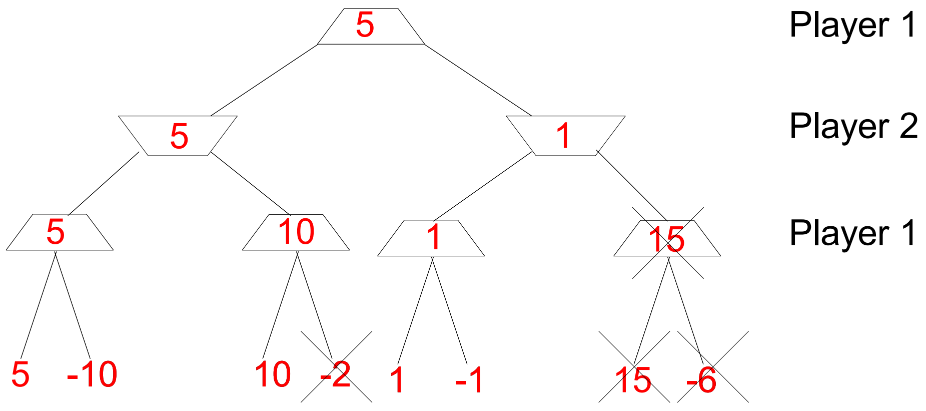
Q3. [12 pts] Minimax and Expectimax

(a) [2 pts] Consider the following zero-sum game with 2 players. At each leaf we have labeled the payoffs Player 1 receives. It is Player 1's turn to move. Assume both players play optimally at every time step (i.e. Player 1 seeks to maximize the payoff, while Player 2 seeks to minimize the payoff). Circle Player 1's optimal next move on the graph, and state the minimax value of the game. Show your work.

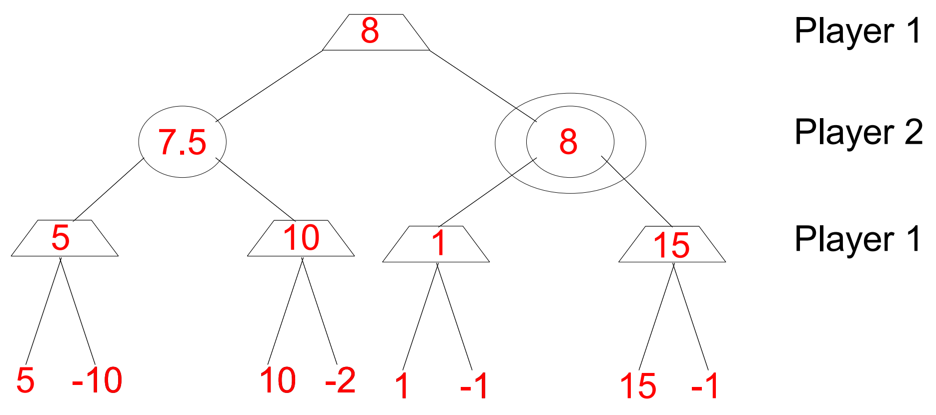


Player 1 should play Left for a payoff of 5.

(b) [2 pts] Consider the following game tree. Player 1 moves first, and attempts to maximize the expected payoff. Player 2 moves second, and attempts to minimize the expected payoff. Expand nodes left to right. Cross out nodes pruned by alpha-beta pruning.

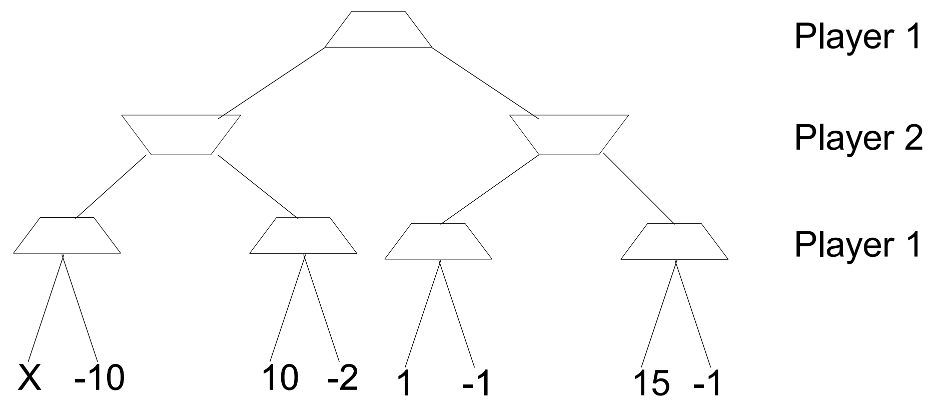


(c) [2 pts] Now assume that Player 2 chooses an action uniformly at random every turn (and Player 1 knows this). Player 1 still seeks to maximize her payoff. Circle Player 1's optimal next move, and give her expected payoff. Show your work.

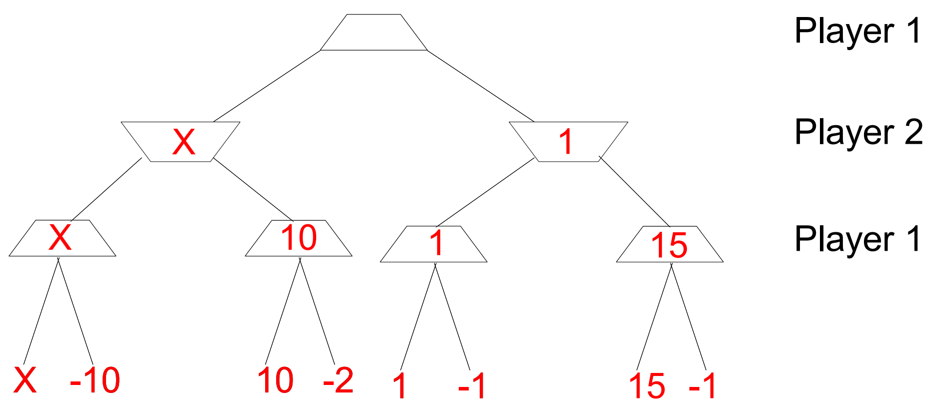


Player 1 should play Right for a payoff of 8.

Consider the following modified game tree, where one of the leaves has an unknown payoff x. Player 1 moves first, and attempts to maximize the value of the game.



(d) [2 pts] Assume Player 2 is a minimizing agent (and Player 1 knows this). For what values of x does Player 1 choose the left action?

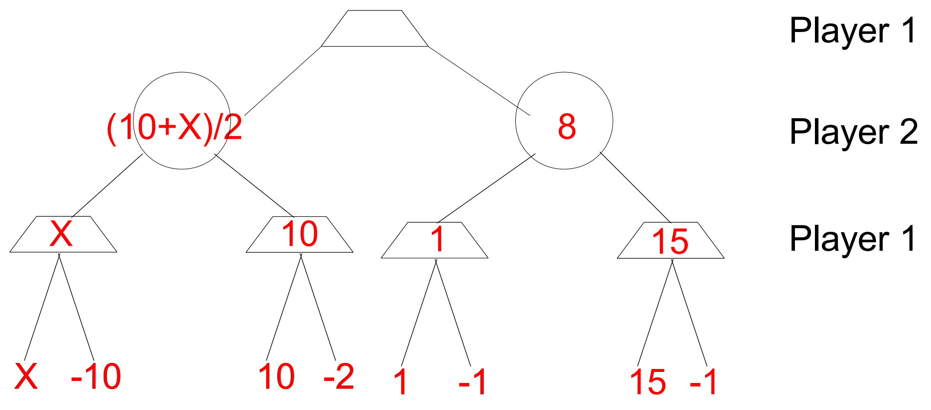


As long as $-10 < x < 10$, the min agent will choose the action leading to x as a payoff. Since the right branch has a value of 1, for any $x > 1$ the optimal minimax agent chooses Left. ($x \geq 1$ also acceptable)

Common mistakes:

- if you said $x \leq 10$. Even if $x > 10$, the optimal action is still to move left initially.
- for $x = 10$. The minimax value is equal to the expectimax value here, but we want the minimax value to be worth more than the expectimax value.

(e) [2 pts] Assume Player 2 chooses actions at random (and Player 1 knows this). For what values of x does Player 1 choose the left action?



Running the expectimax calculation we find that the left branch is worth $(10 + x)/2$ while the right branch is worth 8. Calculation shows that the left branch has higher payoff if $x > 6$. ($x \geq 6$ also acceptable)

(f) [2 pts] For what values of x is the minimax value of the tree worth more than the expectimax value of the tree?

The minimax value of the tree can never exceed the expectimax value of the tree, because the only chance nodes are min nodes. The value of the min node is (weakly) less than the value of the corresponding chance node, so the value the max player receives at the root is (weakly) less under minimax than expectimax.

Common mistakes:

- If you assume the minimax value of the tree is x for $x > 1$ and the expectimax value of the tree is $(x + 10)/2$ for $x > 6$ and solve the inequality, you get $x > 10$ as the critical value for x, corresponding to a minimax payoff of x. However, the minimax value can not exceed 10, or the min player will choose the branch with value 10, so you cannot get payoffs $x > 10$.

- If you calculate the value of the left branch under minimax rules and the value of the right branch under expectimax, you may get $x > 8$.

Q4. [10 pts] n-pacmen search

Consider the problem of controlling n pacmen simultaneously. Several pacmen can be in the same square at the same time, and at each time step, each pacman moves by at most one unit vertically or horizontally (in other words, a pacman can stop, and also several pacmen can move simultaneously). The goal of the game is to have all the pacmen be at the same square in the minimum number of time steps. In this question, use the following notation: let M denote the number of squares in the maze that are not walls (i.e. the number of squares where pacmen can go); n the number of pacmen; and $p_i = (x_i, y_i) : i = 1 \dots n$, the position of pacman i. Assume that the maze is connected.

(a) [1 pt] What is the state space of this problem?

n-Tuples, where each entry is in $\{1, \dots, M\}$. (Code 4.1: Deficient notation, e.g. using {} instead of (), no points marked off)

(b) [1 pt] What is the size of the state space (not a bound, the exact size).

M^n

(c) [2 pts] Give the tightest upper bound on the branching factor of this problem.

5^n (Stop and 4 directions for each pacman). (Code 4.2: Forgot the STOP action, no points marked off)

(d) [2 pts] Bound the number of nodes expanded by uniform cost *tree* search on this problem, as a function of n and M. Justify your answer.

$5^{\frac{nM}{2}}$, because the max depth of a solution is $M/2$ while the branching factor is 5^n. (Code 4.5: No justifications, Code 4.7: Wrong answer but consistent with c)

(e) [4 pts] Which of the following heuristics are admissible? Which one(s), if any, are consistent? Circle the corresponding Roman numerals and briefly justify all your answers.

1. The number of (ordered) pairs (i, j) of pacmen with different coordinates: $h_1 : \sum_{i=1}^{n} \sum_{j=i+1}^{n} (p_i \neq p_j)$
   (i) Consistent? (ii) Admissible? Consider $n = 3$, no wall, and state s such that pacmen are at positions $(i+1, j), (i-1, j), (i, j+1)$. Then all pacmen can meet in one step while $h(s) > 1$.
2. $h_2 : \frac{1}{2} \max(\max_{i,j} |x_i - x_j|, \max_{i,j} |y_i - y_j|)$
   (i) Consistent? (ii) Admissible? Admissible because floor $\frac{1}{2} \max(\max_{i,j} |x_i - x_j|, \max_{i,j} |y_i - y_j|)$ corresponds to the relaxed problem where there are no walls and pacmen can move diagonally. It is also consistent because each absolute value will change by at most 2 per step.

(Code 4.3: Evaluation of h or c in the proof is off)

Q5. [12 pts] CSPs: Course scheduling

An incoming freshman starting in the Fall at Berkeley is trying to plan the classes she will take in order to graduate after 4 years (8 semesters). There is a subset R of required courses out of the complete set of courses C that must all be taken to graduate with a degree in her desired major. Additionally, for each course $c \in C$, there is a set of prerequisites $\text{Prereq}(c) \subset C$ and a set of semesters $\text{Semesters}(c) \subseteq S$ that it will be offered, where $S = \{1, \ldots, 8\}$ is the complete set of 8 semesters. A maximum load of 4 courses can be taken each semester.

(a) [5 pts] Formulate this course scheduling problem as a constraint satisfaction problem. Specify the set of variables, the domain of each variable, and the set of constraints. **Your constraints *need not* be limited to unary and binary constraints.** You may use any precise and unambiguous mathematical notation.

Variables:
For each course $c \in C$, there is a variable $S_c \in S \cup \{\text{NotTaken}\}$ specifying either when the course is scheduled, or alternatively that the course is not to be taken at all.

Constraints:
[Prerequisite] For each pair of courses c, c' such that $c' \in \text{Prereq}(c)$, if $S_c \neq \text{NotTaken}$, then $s_{c'} < S_c$.

[Requirements] For each course $c \in R$, $S_c \neq \text{NotTaken}$.

[Course load] For all $C_5 \subseteq \{c \in C : S_c \neq \text{NotTaken}\}$ such that $|C_5| = 5$, $|\{S_c : c \in C_5\}| > 1$.

(b) [4 pts] The student managed to find a schedule of classes that will allow her to graduate in 8 semesters using the CSP formulation, but now she wants to find a schedule that will allow her to graduate in as few semesters as possible. With this additional objective, formulate this problem as an uninformed search problem, using the specified state space, start state, and goal test.

State space: The set of all (possibly partial) assignments x to the CSP.

Start state: The empty assignment.

Goal test: The assignment is a complete, consistent assignment to the CSP.

Successor function: $\text{Successors}(x) =$

The set of all partial assignments x' to the CSP that extend x with a single additional variable assignment and are consistent with the constraints of the CSP. Since the goal test already includes a test for consistency, it is correct, albeit less efficient, to skip the check for consistency in the successor function.

Cost function: $\text{Cost}(x, x') = \text{graduationSemester}(x') - \text{graduationSemester}(x)$,

where for any partial assignment x, $\text{graduationSemester}(x) = \max_{c \in C : c \neq \text{NotTaken}, c \text{ assigned in } x} S_c(x)$ and $S_c(x)$ denotes the value of S_c in x.

(c) [3 pts] Instead of using uninformed search on the formulation as above, how could you modify backtracking search to efficiently find the least-semester solution?

Backtracking search can first be run with the number of semesters limited to 1; if a solution is found, that solution is returned. Otherwise, the number of semesters is repeatedly increased by 1 and backtracking search re-run until a solution is found.

Q6. [19 pts] Cheating at cs188-Blackjack

Cheating dealers have become a serious problem at the cs188-Blackjack tables. A cs188-Blackjack deck has 3 card types (5,10,11) and an honest dealer is equally likely to deal each of the 3 cards. When a player holds 11, cheating dealers deal a 5 with probability $\frac{1}{4}$, 10 with probability $\frac{1}{2}$, and 11 with probability $\frac{1}{4}$. You estimate that $\frac{4}{5}$ of the dealers in your casino are honest (H) while $\frac{1}{5}$ are cheating ($\neg H$).

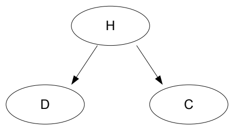
Note: You may write answers in the form of arithmetic expressions involving numbers or references to probabilities that have been directly specified, are specified in your conditional probability tables below, or are specified as answers to previous questions.

(a) [3 pts] You see a dealer deal an 11 to a player holding 11. What is the probability that the dealer is cheating?

$$P(\neg H|D=11) = \frac{P(\neg H, D=11)}{P(D=11)} = \frac{P(D=11|\neg H)P(\neg H)}{P(D=11|\neg H)P(\neg H) + P(D=11|H)P(H)} = \frac{(1/4)(2/10)}{(1/4)(2/10)+(1/3)(8/10)} = 3/19$$

The casino has decided to install a camera to observe its dealers. Cheating dealers are observed doing suspicious things on camera (C) $\frac{4}{5}$ of the time, while honest dealers are observed doing suspicious things $\frac{1}{4}$ of the time.

(b) [4 pts] Draw a Bayes net with the variables H (honest dealer), D (card dealt to a player holding 11), and C (suspicious behavior on camera). Write the conditional probability tables.



H	$P(H)$
1	0.8
0	0.2

H	D	$P(D\|H)$
1	5	1/3
1	10	1/3
1	11	1/3
0	5	1/4
0	10	1/2
0	11	1/4

H	C	$P(C\|H)$
1	1	1/4
1	0	3/4
0	1	4/5
0	0	1/5

(c) [2 pts] List all conditional independence assertions made by your Bayes net.

$$D \perp\!\!\!\perp C|H$$

Common mistakes:

- Stating that two variables are NOT independent; Bayes nets do not guarantee that variables are dependent. This can only be verified by examining the exact probability distributions.

(d) [4 pts] What is the probability that a dealer is honest given that he deals a 10 to a player holding 11 and is observed doing something suspicious?

$$\begin{aligned} P(H|D=10,C) &= \frac{P(H,D=10,C)}{P(D=10,C)} \\ &= \frac{P(H)P(D=10|H)P(C|H)}{P(H)P(D=10|H)P(C|H)+P(\neg H)P(D=10|\neg H)P(C|\neg H)} \\ &= \frac{(4/5)(1/3)(1/4)}{(4/5)(1/3)(1/4)+(1/5)(1/2)(4/5)} \\ &= \frac{5}{11} \end{aligned}$$

Common mistakes:

- -1 for not giving the proper form of Bayes rule, $P(H|D=10,C) = P(H,D=10,C)/P(D=10,C)$
- -1 For a correctly drawn Bayes net C and D are not independent, which means that $P(D=10,C) \neq P(D=10)P(C)$

You can either arrest dealers or let them continue working. If you arrest a dealer and he turns out to be cheating, you will earn a \$4 bonus. However, if you arrest the dealer and he turns out to be innocent, he will sue you for -\$10. Allowing the cheater to continue working will cost you -\$2, while allowing an honest dealer to continue working will get you \$1. Assume a linear utility function $U(x) = x$.

(e) [3 pts] You observe a dealer doing something suspicious (C) and also observe that he deals a 10 to a player holding 11. Should you arrest the dealer?

Arresting the dealer yields an expected payoff of

$$4 * P(\neg H|D=10,C) + (-10) * P(H|D=10,C) = 4(6/11) + (-10)(5/11) = -26/11$$

Letting him continue working yields a payoff of

$$(-2) * P(\neg H|D=10,C) + 1 * P(H|D=10,C) = (-2)(6/11) + (1)(5/11) = -7/11$$

Therefore, you should let the dealer continue working.

(f) [3 pts] A private investigator approaches you and offers to investigate the dealer from the previous part. If you hire him, he will tell you with 100% certainty whether the dealer is cheating or honest, and you can then make a decision about whether to arrest him or not. How much would you be willing to pay for this information?

If you hire the private investigator, if the dealer is a cheater you can arrest him for a payoff of \$4. If he is an honest dealer you can let him continue working for a payoff of \$1. The benefit from hiring the investigator is therefore

$$(4) * P(\neg H|D = 10, C) + 1 * P(H|D = 10, C) = 4(6/11) + (1)(5/11) = 29/11$$

If you do not hire him, your best course of action is to let the dealer continue working for an expected payoff of $-7/11$. Therefore, you are willing to pay up to $29/11 - (-7/11) = 36/11$ to hire the investigator.

Q7. [16 pts] Markov Decision Processes

Consider a simple MDP with two states, S_1 and S_2, two actions, A and B, a discount factor γ of 1/2, reward function R given by

$$R(s,a,s') = \begin{cases} 1 & \text{if } s' = S_1; \\ -1 & \text{if } s' = S_2; \end{cases}$$

and a transition function specified by the following table.

s	a	s'	$T(s,a,s')$
S_1	A	S_1	1/2
S_1	A	S_2	1/2
S_1	B	S_1	2/3
S_1	B	S_2	1/3
S_2	A	S_1	1/2
S_2	A	S_2	1/2
S_2	B	S_1	1/3
S_2	B	S_2	2/3

(a) [2 pts] Perform a single iteration of value iteration, filling in the resultant Q-values and state values in the following tables. Use the specified initial value function V_0, rather than starting from all zero state values. Only compute the entries not labeled "skip".

s	a	$Q_1(s,a)$
S_1	A	1.25
S_1	B	1.50
S_2	A	skip
S_2	B	skip

s	$V_0(s)$	$V_1(s)$
S_1	2	1.50
S_2	3	skip

(b) [2 pts] Suppose that Q-learning with a learning rate α of 1/2 is being run, and the following episode is observed.

s_1	a_1	r_1	s_2	a_2	r_2	s_3
S_1	A	1	S_1	A	-1	S_2

Using the initial Q-values Q_0, fill in the following table to indicate the resultant progression of Q-values.

s	a	$Q_0(s,a)$	$Q_1(s,a)$	$Q_2(s,a)$
S_1	A	$-1/2$	$1/4$	$-1/8$
S_1	B	0	(0)	(0)
S_2	A	-1	(-1)	(-1)
S_2	B	1	(1)	(1)

(c) [4 pts] Assuming that an ϵ-greedy policy (with respect to the Q-values as of when the action is taken) is used, where $\epsilon = 1/2$, and given that the episode starts from S_1 and consists of 2 transitions, what is the probability of observing the episode from part b? State precisely your definition of the ϵ-greedy policy with respect to a Q-value function $Q(s,a)$.

The ϵ-greedy policy chooses $\arg\max_a Q(s,a)$ with probability ϵ, and chooses uniformly among all possible actions (including the optimal action) with probability $1-\epsilon$. Since action a_1 is sub-optimal with respect to $Q_0(s_1)$, it had a probability of $\epsilon/2 = 1/4$ of being selected by the ϵ-greedy policy. Action a_2 is optimal with respect to $Q_1(s_2)$, and therefore had a probability of $(1-\epsilon)+\epsilon/2 = 3/4$ of being selected. Thus, the probability, p, of observing the sequence is the product

$$\begin{aligned} p &= \prod_{i=1}^{2} \Pr(a_i|s_i, \pi_\epsilon(Q_{i-1})) \cdot \Pr(s_{i+1}|s_i,a_i) \\ &= (1/4 \cdot 1/2) \cdot (3/4 \cdot 1/2). \end{aligned}$$

(d) [4 pts] Given an arbitrary MDP with state set S, transition function $T(s,a,s')$, discount factor γ, and reward function $R(s,a,s')$, and given a constant $\beta > 0$, consider a modified MDP (S,T,γ,R') with reward function $R'(s,a,s') = \beta \cdot R(s,a,s')$. Prove that the modified MDP (S,T,γ,R') has the same set of optimal policies as the original MDP (S,T,γ,R).

$V^{\pi}_{\text{modified}} = \beta \cdot V^{\pi}_{\text{original}}$ satisfies the Bellman equation

$$\begin{aligned}
\beta \cdot V^{\pi}_{\text{original}}(s) &= V^{\pi}_{\text{modified}}(s) \\
&= \sum_{s'} T(s,\pi(s),s')[R'(s,\pi(s),s') + \gamma \cdot V^{\pi}_{\text{modified}}(s')] \\
&= \sum_{s'} T(s,\pi(s),s')[\beta \cdot R(s,\pi(s),s') + \gamma \cdot \beta \cdot V^{\pi}_{\text{original}}(s')] \\
&= \beta \cdot \sum_{s'} T(s,\pi(s),s')[R(s,\pi(s),s') + \gamma \cdot V^{\pi}_{\text{original}}(s')] \\
&= \beta \cdot V^{\pi}_{\text{original}}(s').
\end{aligned}$$

It follows that for any state s, the set of policies π that maximize $V^{\pi}_{\text{original}}$ is precisely the same set of policies that maximize $V^{\pi}_{\text{modified}}$.

(e) [4 pts] Although in this class we have defined MDPs as having a reward function $R(s,a,s')$ that can depend on the initial state s and the action a in addition to the destination state s', MDPs are sometimes defined as having a reward function $R(s')$ that depends only on the destination state s'. Given an arbitrary MDP with state set S, transition function $T(s,a,s')$, discount factor γ, and reward function $R(s,a,s')$ that *does depend* on the initial state s and the action a, define an *equivalent* MDP with state set S', transition function $T'(s,a,s')$, discount factor γ', and reward function $R'(s')$ that depends only on the destination state s'.

By *equivalent*, it is meant that there should be a one-to-one mapping between state-action sequences in the original MDP and state-action sequences in the modified MDP (with the same value). **You do not need to give a proof of the equivalence.**

States: $S' = S \times S \times A$, where A is the set of actions.

Transition function:

$$T'(s,a,s') = \begin{cases} T(s''', a, s'''') & \text{if } s = (s'', a'', s''') \text{ and } s' = (s''', a, s''''); \\ 0 & \text{otherwise.} \end{cases}$$

Discount factor: $\gamma' = \gamma$

Reward function: $R'(s') = R(s,a,s'')$, where $s' = (s,a,s'')$.