CS 188
Fall 2018

Introduction to
Artificial Intelligence

Practice Midterm 1

To earn the extra credit, one of the following has to hold true. Please circle and sign.

A I spent 2 or more hours on the practice midterm.

B I spent fewer than 2 hours on the practice midterm, but I believe I have solved all the questions.

Signature: ____

To simulate midterm setting, print out this practice midterm, complete it in writing, and then scan and upload into Gradescope. It is due on Saturday 10/6, 11:59pm.

Exam Instructions:

- You have approximately 2 hours.
- The exam is closed book, closed notes except your one-page crib sheet.
- Please use non-programmable calculators only.
- Mark your answers ON THE EXAM ITSELF. 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.

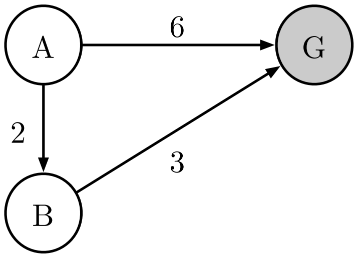
First name	
Last name	
SID	
First and last name of student to your left	
First and last name of student to your right	

For staff use only:

Q1.	Search: Heuristic Function Properties	/6
Q2.	Search: Slugs	/8
Q3.	CSPs: Apple's New Campus	/9
Q4.	Bounded Expectimax	/18
Q5.	Games: Alpha-Beta Pruning	/8
Q6.	MDPs and RL: Mini-Grids	/16
Q7.	Utilities: Low/High	/8
	Total	/73

Q1. [6 pts] Search: Heuristic Function Properties

For the following questions, consider the search problem shown on the left. It has only three states, and three directed edges. A is the start node and G is the goal node. To the right, four different heuristic functions are defined, numbered I through IV.



	h(A)	h(B)	h(G)
I	4	1	0
II	5	4	0
III	4	3	0
IV	5	2	0

(a) [4 pts] **Admissibility and Consistency**

For each heuristic function, circle whether it is admissible and whether it is consistent with respect to the search problem given above.

	Admissible?	Consistent?
I	[✓ Yes] No	Yes [✓ No]
II	Yes [✓ No]	Yes [✓ No]
III	[✓ Yes] No	[✓ Yes] No
IV	[✓ Yes] No	Yes [✓ No]

II is the only inadmissible heuristic, as it overestimates the cost from B: $h(B) = 4$, when the actual cost to G is 3.

To check whether a heuristic is consistent, ensure that for all paths, $h(N) - h(L) \leq \text{path}(N \to L)$, where N and L stand in for the actual nodes. In this problem, $h(G)$ is always 0, so making sure that the direct paths to the goal ($A \to G$ and $B \to G$) are consistent is the same as making sure that the heuristic is admissible. The path from A to B is a different story.

Heuristic I is not consistent: $h(A) - h(B) = 4 - 1 = 3 \leq \text{path}(A \to B) = 2$.
Heuristic III is consistent: $h(A) - h(B) = 4 - 3 = 1 \leq 2$
Heuristic IV is not consistent: $h(A) - h(B) = 5 - 2 = 3 \leq 2$

(b) [2 pts] **Function Domination**

Recall that *domination* has a specific meaning when talking about heuristic functions.

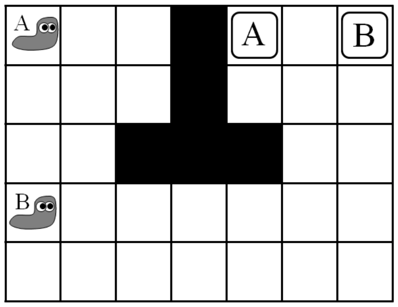
Circle all true statements among the following.

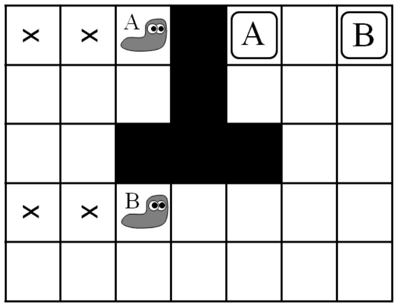
1. Heuristic function III dominates IV.
2. Heuristic function IV dominates III.
3. [✓ Heuristic functions III and IV have no dominance relationship.]
4. Heuristic function I dominates IV.
5. [✓ Heuristic function IV dominates I.]
6. Heuristic functions I and IV have no dominance relationship.

For one heuristic to dominate another, *all* of its values must be greater than or equal to the corresponding values of the other heuristic. Simply make sure that this is the case. If it is not, the two heuristics have no dominance relationship.

Q2. [8 pts] Search: Slugs

You are once again tasked with planning ways to get various insects out of a maze. This time, it's slugs! As shown in the diagram below to the left, two slugs A and B want to exit a maze via their own personal exits. In each time step, both slugs move, though each can choose to either stay in place or move into an adjacent free square. The slugs cannot move into a square that the other slug is moving into. In addition, the slugs leave behind a sticky, poisonous substance and so they cannot move into any square that *either* slug has ever been in. For example, if both slugs move right twice, the maze is as shown in the diagram below to right, with the x squares unpassable to either slug.





You must pose a search problem that will get them to their exits in as few time steps as possible. You may assume that the board is of size N by M; all answers should hold for a general instance, not simply the instance shown above. (You do not need to generalize beyond two slugs.)

(a) [3 pts] How many states are there in a minimal representation of the space? Justify with a brief description of the components of your state space.

$2^{MN}(MN)^2$

The state includes a bit for each of the MN squares, indicating whether the square has been visited (2^{MN} possibilities). It also includes the locations of each slug (MN possibilities for each of the two slugs).

(b) [2 pts] What is the branching factor? Justify with a brief description of the successor function.

$5 \times 5 = 25$ for the first time step, $4 \times 4 = 16$ afterwards.

At the start state each slug has at most five possible next locations (North, South, East, West, Stay). At all future time steps one of those options will certainly be blocked off by the snail's own trail left at the previous time step. Only 4 possible next locations remain.

We accepted both 25 and 16 as correct answers.

(c) [3 pts] Give a non-trivial admissible heuristic for this problem.

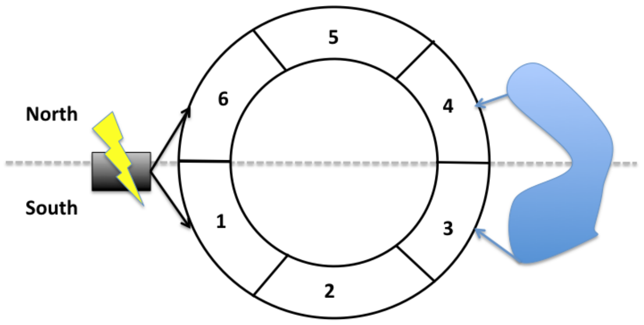
max(maze distance of bug A to its exit, maze distance of bug B to its exit)

Many other correct answers are possible.

Q3. [9 pts] CSPs: Apple's New Campus

Apple's new circular campus is nearing completion. Unfortunately, the chief architect on the project was using Google Maps to store the location of each individual department, and after upgrading to iOS 6, all the plans for the new campus were lost!

The following is an approximate map of the campus:



The campus has six offices, labeled 1 through 6, and six departments:

[noitemsep,topsep=0in]Legal (L) Maps Team (M) Prototyping (P) Engineering (E) Tim Cook's office (T) Secret Storage (S)

Offices can be *next to* one another, if they share a wall (for an instance, Offices 1-6). Offices can also be *across* from one another (specifically, Offices 1-4, 2-5, 3-6).

The Electrical Grid is connected to offices 1 and 6. The Lake is visible from offices 3 and 4. There are two "halves" of the campus – South (Offices 1-3) and North (Offices 4-6).

The constraints are as follows:

i. (L)egal wants a view of the lake to look for prior art examples.

ii. (T)im Cook's office must not be across from (M)aps.

iii. (P)rototyping must have an electrical connection.

iv. (S)ecret Storage must be next to (E)ngineering.

v. (E)ngineering must be across from (T)im Cook's office.

vi. (P)rototyping and (L)egal cannot be next to one another.

vii. (P)rototyping and (E)ngineering must be on opposite sides of the campus (if one is on the North side, the other must be on the South side).

viii. No two departments may occupy the same office.

(a) [3 pts] **Constraints**. Note: There are multiple ways to model constraint *viii*. In your answers below, assume constraint *viii* is modeled as multiple pairwise constraints, not a large n-ary constraint.

(i) [1 pt] Circle your answers below. Which constraints are unary?

[✓ i] *ii* [✓ iii] *iv* *v* *vi* *vii* *viii*

(ii) [1 pt] In the constraint graph for this CSP, how many edges are there?

Constraint *vii* connects each pair of variables; there are $\binom{6}{2} = 15$ such pairs.

(iii) [1 pt] Write out the explicit form of constraint *iii*.

$P \in \{1, 6\}$

(b) [6 pts] **Domain Filtering.** *We strongly recommend that you use a pencil for the following problems.*

(i) [2 pts] The table below shows the variable domains after unary constraints have been enforced and the value 1 has been assigned to the variable P.

Cross out all values that are eliminated by running Forward Checking after this assignment.

L			3	4		
M	$\boxed{1}$	2	3	4	5	6
P	1					
E	$\boxed{1}$	$\boxed{2}$	$\boxed{3}$	4	5	6
T	$\boxed{1}$	2	3	4	5	6
S	$\boxed{1}$	2	3	4	5	6

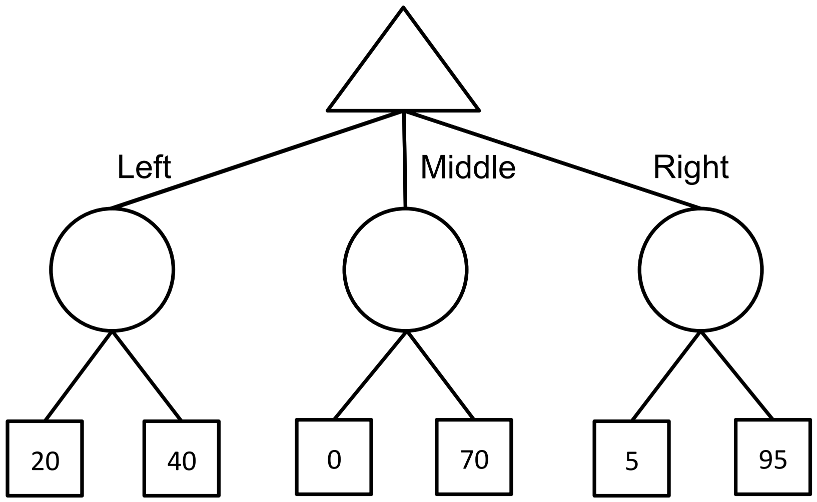
(ii) [4 pts] The table below shows the variable domains after unary constraints have been enforced, the value 1 has been assigned to the variable P, and now the value 3 has been assigned to variable T.

Cross out all values that are eliminated if arc consistency is enforced after this assignment. (Note that enforcing arc consistency will subsume all previous pruning.)

L			$\boxed{3}$	4		
M	$\boxed{1}$	2	$\boxed{3}$	$\boxed{4}$	$\boxed{5}$	$\boxed{6}$
P	1					
E	$\boxed{1}$	$\boxed{2}$	$\boxed{3}$	$\boxed{4}$	$\boxed{5}$	6
T			3			
S	$\boxed{1}$	$\boxed{2}$	$\boxed{3}$	$\boxed{4}$	5	$\boxed{6}$

Q4. [18 pts] Bounded Expectimax

(a) [4 pts] **Expectimax.** Consider the game tree below, where the terminal values are the *payoffs* of the game. Fill in the expectimax values, assuming that player 1 is maximizing expected payoff and player 2 plays uniformly at random (i.e., each action available has equal probability).



(b) [2 pts] Again, assume that Player 1 follows an expectimax strategy (i.e., maximizes expected payoff) and Player 2 plays uniformly at random (i.e., each action available has equal probability).

(i) [2 pts] What is Player 1's expected payoff if she takes the expectimax optimal action?

50

(ii) [1 pt] Multiple outcomes are possible from Player 1's expectimax play. What is the worst possible payoff she could see from that action?

5

(c) [3 pts] Even if the average outcome is good, Player 1 doesn't like that very bad outcomes are possible. Therefore, rather than purely maximizing expected payoff using expectimax, Player 1 chooses to perform a modified search. In particular, she only considers actions whose worst-case outcome is 10 or better.

(i) [1 pt] Which action does Player 1 choose for this tree?

Left

(ii) [1 pt] What is the expected payoff for that action?

30

(iii) [1 pt] What is the worst payoff possible for that action?

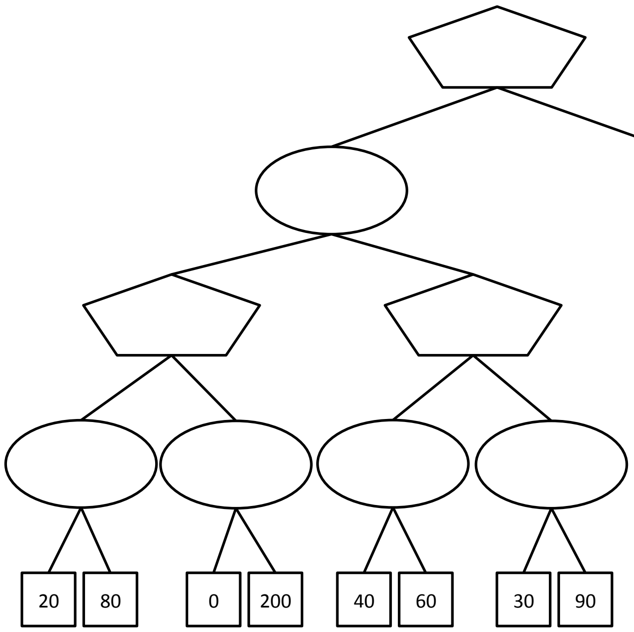
20

(d) [4 pts] Now let's consider a more general case. Player 1 has the following preferences:

- Player 1 prefers any lottery with worst-case outcome of 10 or higher over any lottery with worst-case outcome lower than 10.
- Among two lotteries with worst-case outcome of 10 or higher, Player 1 chooses the one with the highest expected payoff.
- Among two lotteries with worst-case outcome lower than 10, Player 1 chooses the one with the highest worst-case outcome (breaking ties by highest expected payoff).

Player 2 still always plays uniformly at random.

To compute the appropriate values of tree nodes, Player 1 must consider both expectations and worst-case values at each node. For each node in the game tree below, fill in a pair of numbers (e, w). Here e is the expected value under Player 1's preferences and w is the value of the worst-case outcome under those preferences, assuming that Player 1 and Player 2 play according to the criteria described above.



Last expect-layer, $(50, 20)$, $(100, 0)$, $(50, 40)$, $(60, 30)$
Funny max layer on top of lowest expect layer, $(50, 20)$, $(60, 30)$
Expect layer, $(55, 20)$
Funny max at top, $(70, 70)$

(e) [4 pts] Now let's consider the general case, where the lower bound used by Player 1 is a number L not necessarily equal to 10, and not referring to the particular tree above. Player 2 still plays uniformly at random.

(i) [2 pts] Suppose a Player 1 node has two children: the first child passes up values (e_1, w_1), and the second child passes up values (e_2, w_2). What values (e, w) will be passed up by a Player 1 node if

1. $w_1 < w_2 < L$ (e_2, w_2)

2. $w_1 < L < w_2$ (e_2, w_2)

3. $L < w_1 < w_2$ $(max(e_1, e_2), w_{argmax(e_1,e_2)})$

(ii) [2 pts] Now consider a Player 2 node with two children: the first child passes up values (e_1, w_1) and the second child passes up values (e_2, w_2). What values (e, w) will be passed up by a Player 2 node if

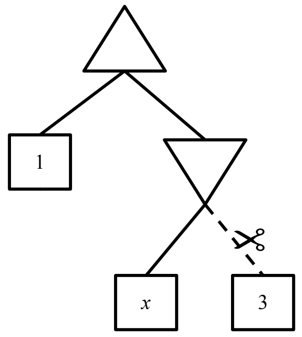
1. $w_1 < w_2 < L$ $(mean(e_1, e_2), min(w1, w2))$

2. $w_1 < L < w_2$ $(mean(e_1, e_2), min(w_1, w_2))$

3. $L < w_1 < w_2$ $(mean(e_1, e_2), min(w_1, w_2))$

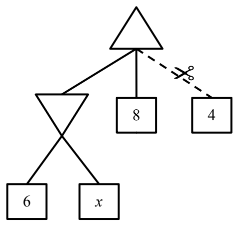
Q5. [8 pts] Games: Alpha-Beta Pruning

For each of the game-trees shown below, state for which values of x the dashed branch with the scissors will be pruned. If the pruning will not happen for any value of x write “none”. If pruning will happen for all values of x write “all”.

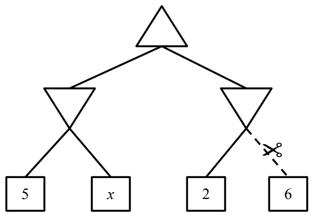


[Example Tree. Answer: $\mathbf{x \leq 1}$.]

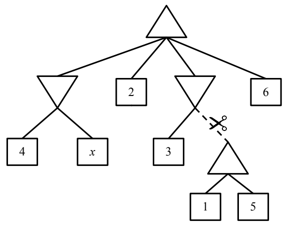
We are assuming that nodes are evaluated left to right and ties are broken in favor of the latter nodes. A different evaluation order would lead to different interval bounds, while a different tie breaking strategies could lead to strict inequalities ($>$ instead of $\geq$).
Successor enumeration order and tie breaking rules typically impact the efficiency of alpha-beta pruning.



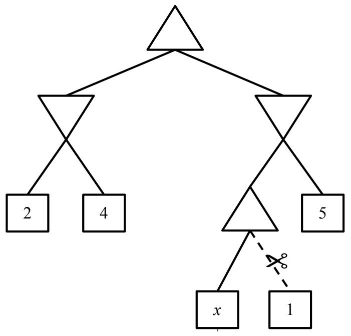
[Tree 1. Answer: None]



[Tree 2. Answer: $x \geq 2$]



[Tree 3. Answer: $x \geq 3$,]



[Tree 4. Answer: None]

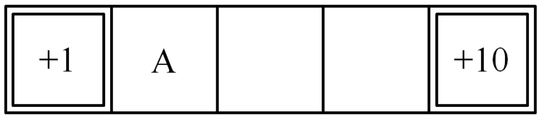
Q6. [16 pts] MDPs and RL: Mini-Grids

The following problems take place in various scenarios of the gridworld MDP (as in Project 3). In all cases, A is the start state and double-rectangle states are exit states. From an exit state, the only action available is *Exit*, which results in the listed reward and ends the game (by moving into a terminal state X, not shown).

From non-exit states, the agent can choose either *Left* or *Right* actions, which move the agent in the corresponding direction. There are no living rewards; the only non-zero rewards come from exiting the grid.

Throughout this problem, assume that value iteration begins with initial values $V_0(s) = 0$ for all states s.

First, consider the following mini-grid. For now, the discount is $\gamma = 1$ and legal movement actions will always succeed (and so the state transition function is deterministic).



(a) [1 pt] What is the optimal value $V^*(A)$?

10

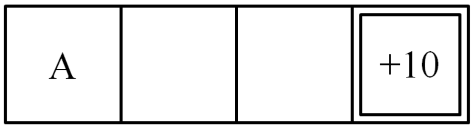
Since the discount $\gamma = 1$ and there are no rewards for any action other than exiting, a policy that simply heads to the right exit state and exits will accrue reward 10. This is the optimal policy, since the only alternative reward if 1, and so the optimal value function has value 10.

(b) [1 pt] When running value iteration, remember that we start with $V_0(s) = 0$ for all s. What is the first iteration k for which $V_k(A)$ will be non-zero?

2
The first reward is accrued when the agent does the following actions (state transitons) in sequence: Left, Exit. Since two state transitions are necessary before any possible reward, two iterations are necessary for the value function to become non-zero.

Let's kick it up a notch! The *Left* and *Right* movement actions are now stochastic and fail with probability f. When an action fails, the agent moves *up* or *down* with probability $f/2$ each. When there is no square to move *up* or *down* into (as in the one-dimensional case), the agent stays in place. The *Exit* action does not fail.

For the following mini-grid, the failure probability is $f = 0.5$. The discount is back to $\gamma = 1$.



(c) [1 pt] What is the optimal value $V^*(A)$?

10. Same reasoning as for the previous problem.

(d) [1 pt] When running value iteration, what is the smallest value of k for which $V_k(A)$ will be non-zero?

4. Same reasoning as for the previous problem, but now the only reward-accruing sequence of actions is Left, Left, Left, Exit.

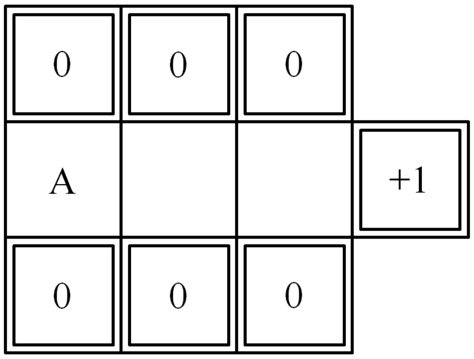
(e) [1 pt] What will $V_k(A)$ be when it is first non-zero?

10/8. Although $\gamma = 1$, the probability that the agent succesfully completes the sequence of actions that leads to a reward at $k = 4$ (Left, Left, Left, Exit) is only $\frac{1}{2} \times \frac{1}{2} \times \frac{1}{2} = \frac{1}{8}$, as at each non-Exit step it has only a $\frac{1}{2}$ probability of success.

(f) [1 pt] After how many iterations k will we have $V_k(A) = V^*(A)$? If they will never become equal, write *never*.

Never. There is always only a $\frac{1}{2}$ probability of success on any movement action, so while V_k will asymptotically approach V^*, it won't ever equal it. Consider the square right next to the exit, which we'll call C: $V_{k+1}(C) = \frac{1}{2}10 + \frac{1}{2}V_k(C)$.

Now consider the following mini-grid. Again, the failure probability is $f = 0.5$ and $\gamma = 1$. Remember that failure results in a shift *up* or *down*, and that the only action available from the double-walled exit states is *Exit*.



(g) [1 pt] What is the optimal value $V^*(A)$?

1/8. Same reasoning as for the previous problem. Note that the exit node value is now only 1, not 10.

(h) [1 pt] When running value iteration, what is the smallest value of k for which $V_k(A)$ will be non-zero?

4

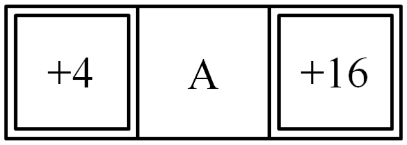
(i) [1 pt] What will $V_k(A)$ be when it is first non-zero?

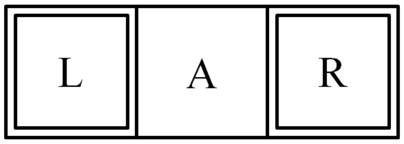
1/8

(j) [1 pt] After how many iterations k will we have $V_k(A) = V^*(A)$? If they will never become equal, write *never*.

4. This problem is different from the previous one, in that a state transition never fails by looping to the same state. Here, a movement action may fail, but that always moves the agent into an absorbing state.

Finally, consider the following mini-grid (rewards shown on left, state names shown on right).





In this scenario, the discount is $\gamma = 1$. The failure probability is actually $f = 0$, but, now we do not actually know the details of the MDP, so we use reinforcement learning to compute various values. We observe the following transition sequence (recall that state X is the end-of-game absorbing state):

s	a	s'	r
A	*Right*	R	0
R	*Exit*	X	16
A	*Left*	L	0
L	*Exit*	X	4
A	*Right*	R	0
R	*Exit*	X	16
A	*Left*	L	0
L	*Exit*	X	4

(k) [2 pts] After this sequence of transitions, if we use a learning rate of $\alpha = 0.5$, what would temporal difference learning learn for the value of A? Remember that $V(s)$ is intialized with 0 for all s.

3. Remember how temporal difference learning works: upon seeing a s, a, r, s' tuple, we update the value function as $V_{i+1}(s) = (1-\alpha)V_i(s) + \alpha(r + V_i(s'))$. To get the answer, simply write out a table of states, all initially with value 0, and then update it with information in each row of the table above. When all rows have been processed, see what value you ended up with for A.

(l) [2 pts] If these transitions repeated many times and learning rates were appropriately small for convergence, what would temporal difference learning converge to for the value of A?

10. We are simply updating the value function with the results of following this policy, and that's what we will converge to. For state A, the given tuples show the agent going right as often as it goes left Clearly, if the agent goes left as often as it goes right from A, the value of being in A is only $16/2 + 4/2 = 10$.

(m) [2 pts] After this sequence of transitions, if we use a learning rate of $\alpha = 0.5$, what would Q-learning learn for the Q-value of (A, *Right*)? Remember that $Q(s, a)$ is initialized with 0 for all (s, a).

4. The technique is the same as in problem (o), but use the Q-learning update (which includes a max). How do you get the max? Here's an example:

The sample sequence: $(A, Right, R, 0)$.

$Q(s, a) \leftarrow (1 - \alpha)Q(s, a) + \alpha(r + \gamma \max_{a'}(s', a'))$.

$Q(A, right) \leftarrow (1 - \alpha)Q(A, right) + \alpha(r + \gamma \max_{a'}(R, a'))$.

But since there is only one exit action from R, then:

$Q(A, right) \leftarrow (1 - \alpha)Q(A, right) + \alpha(r + \gamma Q(R, Exit))$.

Note that this MDP is very small – you will finish the game in two moves (assuming you have to move from A).

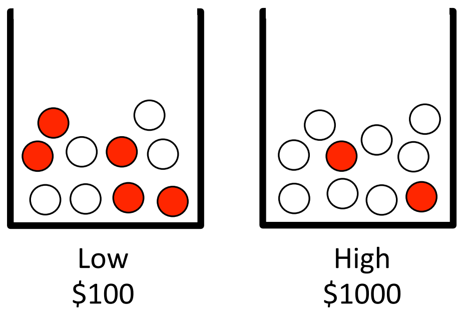
(n) [2 pts] If these transitions repeated many times and learning rates were appropriately small for convergence, what would Q-learning converge to for the Q-value of (A, *Right*)?

16. Q-learning converges to the optimal Q-value function, if the states are fully explored and the convergence rate is set correctly.

Q7. [8 pts] Utilities: Low/High

After a tiring day of eating food and escaping from ghosts, Pacman heads to the casino for some well-deserved rest and relaxation! This particular casino has two games, Low and High, which are both free to play.

The two games are set up very similarly. In each game, there is a bin of marbles. The Low bin contains 5 white and 5 dark marbles, and the High bin contains 8 white and 2 dark marbles:



Play for each game proceeds as follows: the dealer draws a single marble at random from the bin. If a dark marble is drawn, the game pays out. The Low payout is \$100, and the High payout is \$1000. The payout is divided evenly among everyone playing that game. For example, if two people are playing Low and a dark marble is drawn, they each receive \$50. If a white marble is drawn, they receive nothing. The drawings for both games are done simultaneously, and only once per night (there is no repeated play).

(a) [2 pts] **Expectations.** Suppose Pacman is at the casino by himself (there are no other players). Give his expected winnings, in dollars:

(i) [1 pt] From playing a single round of Low: $\frac{5}{10} \cdot \$100 + \frac{5}{10} \cdot \$0 = \$50$

(ii) [1 pt] From playing a single round of High: $\frac{2}{10} \cdot \$1000 + \frac{8}{10} \cdot \$0 = \$200$

(b) [6 pts] **Preferences.** Pacman is still at the casino by himself. Let p denote the amount of money Pacman wins, and let his utility be given by some function $U(p)$. Assume that Pacman is a rational agent who acts to maximize expected utility.

(i) [3 pts] If you observe that Pacman chooses to play Low, which of the following must be true about $U(p)$? Assume $U(0) = 0$. (circle any that apply)

$U(50) \geq U(1000)$ $\qquad$ $U(100) \geq U(1000)$

$\frac{1}{2}U(100) \geq \frac{2}{10}U(1000)$ $\qquad$ $U(50) \geq U(100)$

Review Axioms of Rationality.

(ii) [3 pts] Given that Pacman plays Low, which of the following are possibilities for $U(p)$? You may use $\sqrt[3]{100} \approx 4.6$, although this question should not require extensive calculation. (circle any that apply)

p $\qquad$ $-p$ $\qquad$ $2^p - 1$ $\qquad$ p^2 $\qquad$ $\sqrt[3]{p}$

Check whether the response you gave for the previous question applies to these functions.