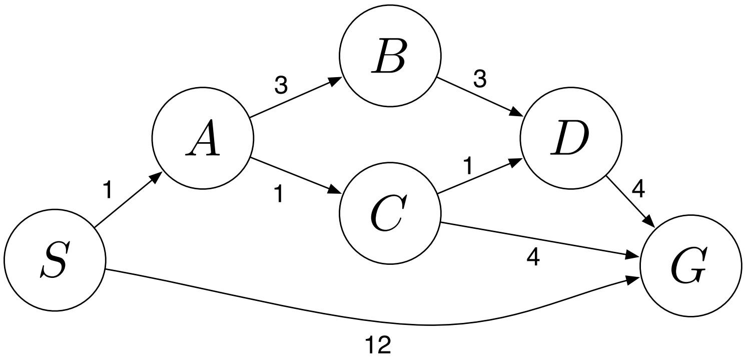
Table 1: For instructor's use

Question	Points Scored	Possible Points
1		12
2		8
3		12
4		12
5		12
6		8

This exam is closed book. You are allowed 2 sheets of notes (4 pages front and back). You may use any format for your notes that you like. Please explain all of your answers fully to receive full credit.

Here is some extra space. **Show all of your work on the questions!** If you need more paper just ask. Good luck!!

Question 1. Answer the following **graph search** questions using the graph below, where *S* is the start node and *G* is the goal node. Break any ties alphabetically.



(a) (2 pts) What *solution path* would breadth-first graph search return for this problem?

S - G

(b) (2 pts) What *explored list and solution path* would uniform cost graph search return for this problem?

Frontier:

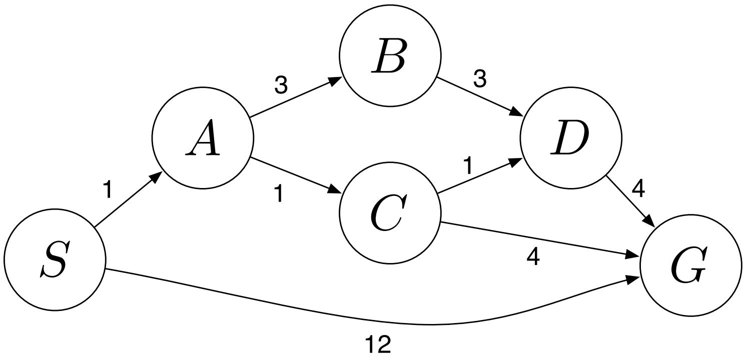
S(0)
A(1) G(12)
C(2) B(4) G(12)
D(3) B(4) G(6)
B(4) G(6)
G(6)

Explored: S A C D B G *Solution:* S-A-C-G

(c) (2 pts) What *explored list and solution path* would depth-first graph search return for this problem?

Explored: S A B D G *Solution:* S-A-B-D-G

Graph repeated for your convenience:



(d) (2 pts) What *solution path* would A* graph search, using a consistent heuristic, return for this problem?

Solution: S-A-C-G

(d) (4 pts) Design a consistent heuristic such that the explored list for A* graph search is:

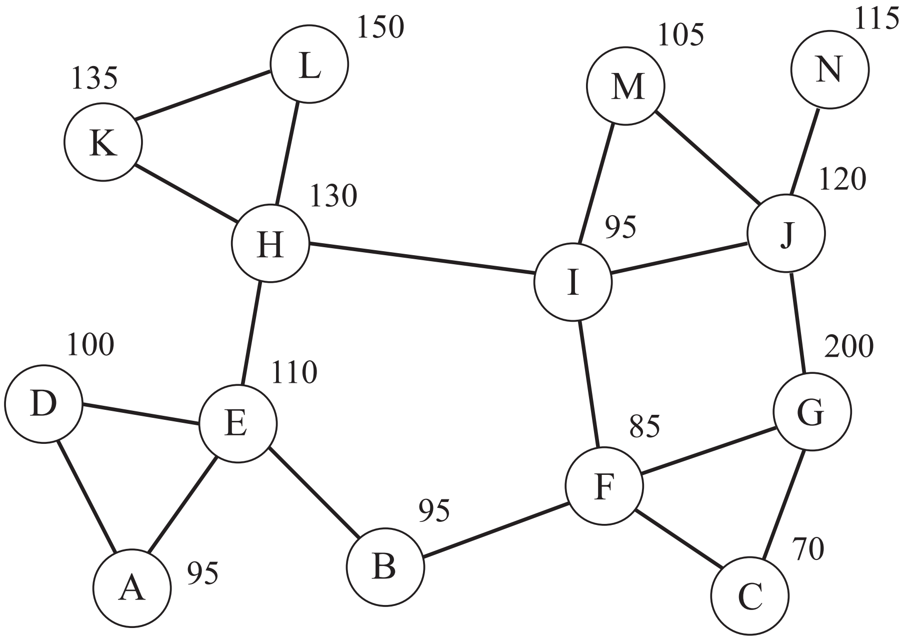
Explored: S, A, B, C, G

Note that you can provide your answer by writing the heuristic value next to each state in the graph above.

SOLUTION:

State	h_1
S	1
A	3
B	1
C	4
D	4
G	0

Question 2. In this question you will use the **Hill Climbing search algorithm** on the graph shown below to *maximize the objective function*, which is given by the number next to each state.



(a) (1 pts) If hill climbing starts in state D, which adjacent state would be visited next? E

(b) (3 pts) Give the order in which nodes would be visited by hill climbing, starting at D:

Visited: D E H L

(c) (1 pts) Which state is the global maximum of the objective function? G

(d) (3 pts) List all of the starting states with the property that hill climbing search will reach the global maximum.

Starting States: C F J N M G. Note that I is excluded because from I you will go to H.

Question 3. Probabilistic Duck

Your friend *Howard the Duck* has constructed a highly-biased probabilistic model of the animal kingdom, consisting of the joint probability distribution $P(W, Q, D)$, where W stands for "walks," Q stands for "quacks," and D stands for "is-a-duck."

	$W = 0$		$W = 1$	
	$Q = 0$	$Q = 1$	$Q = 0$	$Q = 1$
$D = 0$	0.15	0.08	0.05	0.02
$D = 1$	0.05	0.10	0.10	0.45

(a) (1 pts) Suppose that you are drawing random samples from this probabilistic model. What is the likelihood that a randomly-sampled animal will *walk*, *quack*, and *be a duck*?

$P(W = 1, Q = 1, D = 1) = 0.45$

(b) (4 pts) Calculate the following joint probability tables through marginalization (i.e. "summing out"): $P(W, Q)$, $P(D, Q)$, and $P(D)$. Note: we have provided the table $P(D, W)$ as an example.

	$W = 0$	$W = 1$
$D = 0$	0.23	0.07
$D = 1$	0.15	0.55

	$Q = 0$	$Q = 1$
$W = 0$	0.2	0.18
$W = 1$	0.15	0.47

	$Q = 0$	$Q = 1$
$D = 0$	0.2	0.1
$D = 1$	0.15	0.55

$D = 0$	0.3
$D = 1$	0.7

Sample Computations:

$P(W = 0, Q = 0) = \sum_D P(W = 0, D, Q = 0) = 0.15 + 0.05 = 0.2$

$P(D = 1, Q = 1) = \sum_W P(W, D = 1, Q = 1) = 0.1 + 0.45 = 0.55$

(c) (4 pts) If you are observing a duck, what is the probability that it can quack (i.e. $P(Q = 1|D = 1)$)? What is the probability that it can both quack and walk?

From the definition of conditional probability:

$$P(Q = 1|D = 1) = \frac{P(Q = 1, D = 1)}{P(D = 1)} = \frac{0.55}{0.7} = 0.786$$

$$P(Q = 1, W = 1|D = 1) = \frac{P(Q = 1, W = 1, D = 1)}{P(D = 1)} = \frac{0.45}{0.7} = 0.643$$

(d) (3 pts) You observe that a particular animal walks and quacks. What is the probability that it is a duck? Compare this answer to your answer in part (a). Are the probabilities the same or different? Explain why.

From the definition of conditional probability:

$$P(D = 1|Q = 1, W = 1) = \frac{P(D = 1, Q = 1, W = 1)}{P(Q = 1, W = 1)} = \frac{0.45}{0.47} = 0.957$$

The two numbers are different, because in the case of part (d) we have evidence and in the case of part (a) we do not. In other words, 0.45 is the marginal probability which indicates how often the event $D = 1, W = 1, Q = 1$ occurs as a fraction of all animals and behaviors. In (d), we consider the probability that the animal is a duck out of all animals that walk and quack. The observations $W = 1, Q = 1$ are evidence, and the resulting probability is a conditional probability.

Question 4. Mark each of the following statements as *TRUE or FALSE*.

If FALSE, **rewrite the sentence** changing just a few words to make it true. Two points each.

- A game of poker (such as Texas Hold'em or Five Card Draw) is an example of a stochastic, ~~fully-observable~~ partially-observable, multi-agent task environment. FALSE

- The primary difference between A* search and uniform cost search is the use of a ~~priority queue~~ consistent or admissable heuristic. FALSE

- Given two admissable heuristics a and b, the heuristic defined by $\max\{a, b\}$ dominates both of them. TRUE

- The primary benefit of a reinforcement learning method like Monte Carlo Control over Policy Iteration is that Monte Carlo methods can learn directly from data, and therefore don't require a model. TRUE

- Given two random variables a and b, if ~~$\sum_a p(a,b) = p(b)$~~ $p(a, b) = p(a)p(b)$, then we can conclude that b is independent of a. FALSE

- Given an MDP, suppose that we have run Value Iteration until convergence. If state c can be reached from state b by taking the optimal action a (e.g. $T(b, a, c) > 0$), then it follows that ~~$V(c) \geq V(b)$~~ $V(s_t = b) = R(b) + \gamma \sum_{s_{t+1}} P(s_{t+1}|s_t = b, a)V(s_{t+1})$. FALSE

Question 5. **Markov Decision Process**

Consider the following grid world for an MDP, where the states are $\{S_1, S_2, S_3\}$ and there are four goal terminals $\{G_1, G_2, G_3, G_4\}$. The reward for being in each state is -0.1. The reward for reaching G_1 is 10, while the reward for reaching each of the other goals is 1. The agent starts in S_1 and has *two* possible actions: *Right* and *Up*. The probabilistic transition model is as follows:

- When moving *Right*, the agent will go *Up* by accident with probability 0.2
- When moving *Up*, the agent will go *Right* by accident with probability 0.2.

1 G$_2$	1 G$_3$	1 G$_4$	▨
-0.1 S$_1$	-0.1 S$_2$	-0.1 S$_3$	10 G$_1$

(a) (6 pts) Assume that all of the *utilities are initialized to 0.1* and there is *no discounting* (i.e. $\gamma = 1$). Calculate the updated utilities for states S_1, S_2, and S_3 over *three rounds* of value iteration, by filling in the missing elements below. (Note that $U^i(S_j)$ denotes the utility for state j in iteration number i.)

SOLUTION:

$$U^1(S_3) = -0.1 + \max_{R,U} \begin{cases} R: 0.8 \times 10 + 0.2 \times 1 = \underline{8.2} \\ U: 0.8 \times 1 + 0.2 \times 10 = 2.8 \end{cases} = 8.1$$

$$U^2(S_2) = -0.1 + \max_{R,U} \begin{cases} R: 0.8 \times 8.1 + 0.2 \times 1 = \underline{6.68} \\ U: 0.8 \times 1 + 0.2 \times 8.1 = 2.42 \end{cases} = 6.58$$

$$U^3(S_1) = -0.1 + \max_{R,U} \begin{cases} R: 0.8 \times 6.58 + 0.2 \times 1 = \underline{5.46} \\ U: 0.8 \times 1 + 0.2 \times 6.58 = 2.12 \end{cases} = 5.36$$

Note that the optimal utility selected by max is underlined in each equation. The action associated with that utility will be the optimal action for the policy associated with the value function.

SOLUTION (continued)

Note that the ability to compute the value for only a single state in each step of value iteration is the result of the special design of this grid world. Once we have updated $V(S_3)$ it will never change, because it depends only on two fixed rewards (10 and 1). It follows that once $V(S_3)$ has been updated, then updating $V(S_2)$ will produce the final optimal value because S_2 only depends on S_3 and a fixed reward. Likewise, once $V(S_2)$ has been updated, then updating $V(S_1)$ produces the final value. The evaluation sequence given in the problem exploits this ordering property of the gridworld.

A naïve approach would compute updated values for all states at each iteartion of value iteration. Those updates will change the value function from its initial setting, because each state is directly connected to a reward. Those intermediate values will be overwritten, however, when the final updates given in the problem are computed. However, for your benefit we provide the updates for the intermediate values here:

$$U^1(S_1) = -0.1 + \max_{R,U} \begin{cases} R: 0.8 \times 0.1 + 0.2 \times 1 = 0.28 \\ U: 0.8 \times 1 + 0.2 \times 0.1 = \underline{0.82} \end{cases} = 0.72$$

$$U^1(S_2) = -0.1 + \max_{R,U} \begin{cases} R: 0.8 \times 0.1 + 0.2 \times 1 = 0.28 \\ U: 0.8 \times 1 + 0.2 \times 0.1 = \underline{0.82} \end{cases} = 0.72$$

$$U^2(S_1) = -0.1 + \max_{R,U} \begin{cases} R: 0.8 \times 0.72 + 0.2 \times 1 = 0.78 \\ U: 0.8 \times 1 + 0.2 \times 0.72 = \underline{0.94} \end{cases} = 0.84$$

$$U^2(S_3) = U^1(S_3) = 8.1$$

$$U^3(S_2) = U^2(S_2) = 6.58$$

$$U^3(S_3) = U^2(S_3) = 8.1$$

(b) (3 pts) When value iteration has converged, indicate on the grid world figure from the previous page what you believe the final (optimal) policy will be. How many iterations will be required for value iteration to converge (i.e. reach the point where the values no longer change)? Explain your answer.

SOLUTION:

Final Policy: Go Right from all states

Value iteration will converge in three iterations. This is because the states S_1 and S_2 are connected to their neighbor to the right and a fixed reward above, and state S_3 is connected only to fixed rewards. Since there are three states, it will take one iteration to update $V(S_3)$ and two more iterations to propagate its value to the other two states. In general, for a gridworld of this structure with N states, convergence will require N iterations.

It may be helpful to consider the convergence of Policy Iteration also. It consists of interleaved steps of Policy Evaluation (PE), where the utilities are updated, and Policy Improvement (PI), where the policy is optimized. The number of iterations required by policy iteration depends upon the initialization. In the standard approach, an initial policy is selected and the algorithm is run. Even if we initialize all states with the UP action, the algorithm still terminates in three iterations as shown below:

Iter 1, PE: No optimal values are produced

Iter 1, PI: $\pi^*(S_3) = R, \quad \pi^*(S_2) = R$

Iter 2, PE: $U^*(S_3) = 8.1, \quad U^*(S_2) = 6.58$

Iter 2, PI: $\pi^*(S_1) = R$

Iter 3, PE: $U^*(S_1) = 5.36$

Iter 3, PI: No change in policy, terminate

Note that we terminate in 3 iterations instead of 4 because $Q(S_3, U) = 2.8 > 1$. As a result, even with $\pi^1(S_3) = U$, the value $V^1(S_3) = 2.8$ computed in iteration 1 of PE is larger than the fixed reward of 1 available by going Up from S_2. Therefore we choose the optimal action $\pi^1(S_2) = R$, even though $V(S_3)$ has not converged yet. If the rewards along the top row were increased, we could force policy iteration to require four iterations to converge in the worst case.

(c) (3 pts) Demonstrate that your policy in part (b) is optimal using *policy iteration*. Policy iteration has converged when we compute the expected utility $U(\pi^*)$ for a policy π^*, and then demonstrate that π^* is optimal under $U(\pi^*)$.

SOLUTION:

We have a policy π from part (b) and want to show that it is optimal. Compute the utilities associated with that policy using policy evaluation:

$$U^\pi(S_3) = -0.1 + 0.8 \times 10 + 0.2 \times 1 = 8.1$$

$$U^\pi(S_2) = -0.1 + 0.8 \times 8.1 + 0.2 \times 1 = 6.58$$

$$U^\pi(S_1) = -0.1 + 0.8 \times 6.58 + 0.2 \times 1 = 5.36$$

We can see that these utilities are equal to the values produced at the convergence of value iteration in (a). Now we compute the optimal policy under these utilities:

$$\pi(S_1) = \arg\max_{R,U} \begin{cases} R : 6.58 \\ U : 1 \end{cases} = R$$

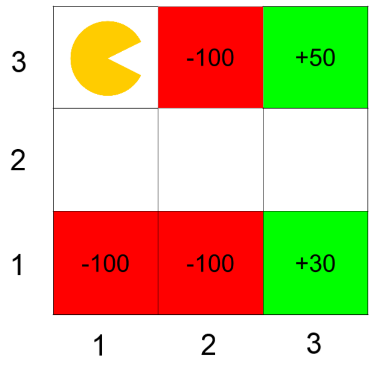
$$\pi(S_2) = \arg\max_{R,U} \begin{cases} R : 8.1 \\ U : 1 \end{cases} = R$$

$$\pi(S_3) = \arg\max_{R,U} \begin{cases} R : 10 \\ U : 1 \end{cases} = R$$

Since this policy equals your choice in (b), it is therefore optimal.

Question 6. Q Learning

Consider the grid-world MDP below. Rewards are only awarded for taking the *Exit* action from one of the red or green (shaded) states. Taking this action moves the agent to the Done state, and the MDP terminates. Assume $\gamma = 1$ and $\alpha = 0.5$ for all calculations. If γ and α are needed in any equation then they must be included explicitly.



(a) (3 pts) The agent starts from the top left corner and you are given the following episodes from runs of the agent through this grid-world. Each line in an Episode is a tuple containing (s, a, s', r).

Episode 1	Episode 2	Episode 3	Episode 4	Episode 5
(1,3), S, (1,2), 0	(1,3), S, (1,2), 0	(1,3), S, (1,2), 0	(1,3), S, (1,2), 0	(1,3), S, (1,2), 0
(1,2), E, (2,2), 0	(1,2), E, (2,2), 0	(1,2), E, (2,2), 0	(1,2), E, (2,2), 0	(1,2), E, (2,2), 0
(2,2), E, (3,2), 0	(2,2), S, (2,1), 0	(2,2), E, (3,2), 0	(2,2), E, (3,2), 0	(2,2), E, (3,2), 0
(3,2), N, (3,3), 0	(2,1), Exit, D, -100	(3,2), S, (3,1), 0	(3,2), N, (3,3), 0	(3,2), S, (3,1), 0
(3,3), Exit, D, +50		(3,1), Exit, D, +30	(3,3), Exit, D, +50	(3,1), Exit, D, +30

Fill in the following Q-values obtained from direct evaluation from the samples:

Q((3,2), N) = 50 Q((3,2), S) = 30 Q((2,2), E) = 40

Direct evaluation is just averaging the discounted reward after performing action a in state s. Since there is no discounting with $\gamma = 1$, we just record the average terminal reward across episodes in which the state-action pair occurs. In detail:

Q((3,2), N): [50 (Ep 1) + 50 (Ep 4)]/2 = 50

Q((3,2), S): [30 (Ep 3) + 30 (Ep 5)]/2 = 30

Q((2,2), E): [50 (Ep 1) + 30 (Ep 3) + 50 (Ep 4) + 30 (Ep 5)]/4 = 40

(b) (3 pts) Q-learning is an online algorithm to learn optimal Q-values in an MDP with unknown rewards and transition function. The update equation is:

$$Q(s_t, a_t) = (1 - \alpha)Q(s_t, a_t) + \alpha(R_{t+1} + \gamma \max_{a'} Q(s_{t+1}, a'))$$

where γ is the discount factor, α is the learning rate and the sequence of observations are $(\cdots, s_t, a_t, s_{t+1}, r_t, \cdots)$. Given the episodes in (a), fill in the time at which the following Q values first become non-zero. Your answer should be of the form (**episode#,iter#**) where **iter#** is the Q-learning update iteration in that episode. If the specified Q value never becomes non-zero, write *never*.

Q((1,2), E) = (4,2) Q((2,2), E) = (3,3) Q((3,2), S) = (3,4)

SOLUTION:

This question investigates the manner in which Q-values propagate through the state space. The first thing to understand is that Q-learning is run in the following order: Observations from ep 1, then observations from ep 2, and so on. The first two episodes will update the Q values for the pairs (3,2),N and (2,2),S that are adjacent to terminal states. At the start of episode 3, the only nonzero Q values will be Q((3,2),N) $= 25$ and Q((2,2),S) $= -50$. Episode 3 will produce the following updates:

Iter 2: $Q\text{((1,2),E)} = 0.5Q\text{((1,2),E)} + 0.5(0 + \max_a Q\text{((2,2)},a)) = 0$

Note that the updated value is zero even though row (2,2) of the Q table has a nonzero value, the -50 associated with S. The max will still select 0 in this case.

Iter 3: $Q\text{((2,2),E)} = 0.5Q\text{((2,2),E)} + 0.5(0 + \max_a Q\text{((3,2)},a)) = 12.5$

Here the max will select the value 25 associated with action N. Note that the fact that the next action was S doesn't matter, because Q-Learning will always choose the optimal action. The final iteration backs up from the terminal state (3,1):

Iter 4: $Q\text{((3,2),S)} = 0.5Q\text{((3,2),S)} + 0.5(30) = 15$

Note that there is no max over the terminal state in this update because the terminal state has no actions other than exit, and its value is always given by its reward. These updates give the answers for Q((2,2),E) and Q((3,2),S) above. Following Episode 3, the full Q table is as follows (it is not necessary to write this to complete the problem, this is just for your benefit):

	N	S	E	W
(1,2)	0	0	0	0
(1,3)	0	0	0	0
(2,2)	0	-50	12.5	0
(3,2)	25	15	0	0

Processing Episode 4 results in the following updates:

Iter 2: $Q(\text{(1,2),E}) = 0.5Q(\text{(1,2),E}) + 0.5(0 + \max_a Q(\text{(2,2)},a)) = 0.5 \times 12.5 = 6.25$

Iter 3: $Q(\text{(2,2),E}) = 0.5Q(\text{(2,2),E}) + 0.5(0 + \max_a Q(\text{(3,2)},a)) = 0.5 \times 12.5 + 0.5 \times 25 = 18.75$

Iter 4: $Q(\text{(3,2),N}) = 0.5Q(\text{(3,2),N}) + 0.5(50) = 0.5 \times 25 + 0.5 \times 50 = 37.5$

These updates give the answer for Q((1,2),E) above, completing the problem. Following Episode 4, the Q table is:

	N	S	E	W
(1,2)	0	0	6.25	0
(1,3)	0	0	0	0
(2,2)	0	-50	18.75	0
(3,2)	37.5	15	0	0

Processing Episode 5 gives the final Q table:

	N	S	E	W
(1,2)	0	0	12.5	0
(1,3)	0	3.13	0	0
(2,2)	0	-50	28.13	0
(3,2)	37.5	22.5	0	0

(c) (2 pts) In Q-learning, we look at a window of $(s_t, a_t, r_{t+1}, s_{t+1})$ to update our Q-values. One can think of using an update rule that uses a larger window to update these values. Give an update rule for $Q(s_t, a_t)$ given the window $(s_t, a_t, r_{t+1}, s_{t+1}, a_{t+1}, r_{t+2}, s_{t+2})$.

$Q(s_t, a_t) = (1-\alpha)Q(s_t, a_t) + \alpha(r_{t+1} + \gamma r_{t+2} + \gamma^2 \max_{a'} Q(s_{t+2}, a'))$

(Sample of the expected discounted reward looking ahead to $t+2$)

$Q(s_t, a_t) = (1-\alpha)Q(s_t, a_t) + \alpha(r_{t+1} + \gamma((1-\alpha)Q(s_{t+1}, a_{t+1}) + \alpha(r_{t+2} + \gamma \max_{a'} Q(s_{t+2}, a'))))$

(Nested Q-learning update)

$$\begin{aligned} Q(s_t, a_t) = (1-\alpha)Q(s_t, a_t) + \alpha(r_{t+1} + \gamma \max\{&\max_{a'} Q(s_{t+1}, a'), \\ &(1-\alpha)Q(s_{t+1}, a_{t+1}) + \alpha(r_{t+2} + \gamma \max_{a'} Q(s_{t+2}, a'))\}) \end{aligned}$$

(Max of normal Q-learning update and one step look-ahead update)

Note the purpose of this problem was to explore different possible ways to generalize the basic Q-Learning update equation. There are many possible variants that would be correct, here we are just giving three examples. You aren't expected to know these different variants or their names, and we would be unlikely to ask this part (c) on the actual midterm (but parts (a) and (b) are fair game).