1 Game Theory Search

(a) What are the differences between extensive form and normal form games?

(b) What is a strategy? What is the difference between a pure and a mixed strategy?

(c) We can practice computing utilities using this example that was given in lecture:

CRAM	DO HW	PLAY GAME	
98	100	85	P(EASY) = .2
97	90	65	P(HARD) = .8

- What is the utility of the pure strategy: cram?

- What is the utility of the pure strategy: do HW?

- What is the utility of the mixed strategy: $\frac{1}{2}$ cram, $\frac{1}{2}$ do HW?

(d) What is a Nash Equilibrium?

(e) Does a Nash Equilibrium always exist?

(f) Consider a two player game, where each player must simultaneously choose a number from $\{2, 3, ..., 99, 100\}$. Let x_1 represent the value chosen by player 1, and x_2 represent the value chosen by player 2. The rules of the game are such that the utility for a player 1 can be given as:

$$u(p_1) = \begin{cases} x_1 & x_1 = x_2 \\ x_2 - 2 & x_1 > x_2 \\ x_2 + 2 & x_1 < x_2 \end{cases}$$

Because the rules of the game for everyone are the same, the utility function for player 2 is symmetric to $u(p1)$. Does there exist a pure Nash Equilibrium for this game? It may help to try to play a few rounds of this game with someone next to you.

(g) What is a Correlated Equilibrium? What is its relationship to a Nash Equilibrium?

(h) Let us consider several voting strategies:

- Plurality
- Borda Count
- Single Transferable Vote
- Pairwise Elections
- Plurality with Runoff

- Condorcet Winner

(i) Which voting rule (Plurality, Borda Count, Both, Neither) is Condorcet consistent? Consider the following voting example:

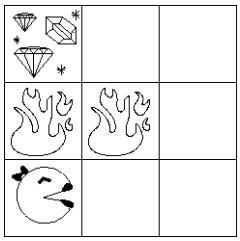
3 voters	2 voters
a	b
b	c
c	a

3 voters	2 voters	2 voters
a	b	c
b	c	b
c	a	a

(j) Which voting rule is the best?

2 RL: Treasure Hunting

While Pacman is busting ghosts, Ms. Pacman goes treasure hunting on GridWorld Island. She has a map showing where the hazards are, and where the treasure is. From any unmarked square, Ms. Pacman can take any of the deterministic actions (N, S, E, W) that doesn't lead off the island. If she lands in a hazard square or a treasure square, her only action is to call for an airlift (X), which takes her to the terminal *Done* state; this results in a reward of -64 if she's escaping a hazard, or +128 if she reached the treasure. There is no living reward.



(a) Let $\gamma = 0.5$. What are the optimal values V^* of each state in the grid above?

(b) How would we compute the Q-values for each state-action pair?

(c) What's the optimal policy?

Call this policy π_0.
Ms. Pacman realizes that her map might be out of date, so she uses Q-learning to see what the island is really like. She believes π_0 is close to correct, so she follows an ϵ-random policy, ie., with probability ϵ she picks a legal action uniformly at random (otherwise, she does what π_0 recommends). Call this policy π_ϵ.
π_ϵ is known as a *stochastic* policy, which assigns probabilities to actions rather than recommending a single one. A stochastic policy can be defined with $\pi(s, a)$, the probability of taking action a when the agent is in state s.

(d) Write a modified Bellman update equation for policy evaluation when using a stochastic policy $\pi(s, a)$ (this is similar to a problem seen on midterm 2).

(e) If the original map and our assumptions about the transitions and rewards are correct, what relationship will hold for all states s?

$$V^{\pi_0}(s) \leq V^{\pi_\epsilon}(s) \qquad V^{\pi_0}(s) = V^{\pi_\epsilon}(s) \qquad V^{\pi_0}(s) \geq V^{\pi_\epsilon}(s)$$

As it turns out, Ms. Pacman's map is mostly correct, but some of the fire pits seem to have fizzled out. She observes the following episodes during her Q-learning:
{(0, 0), N, 0}, {(0, 1), N, 0}, {(0, 2), X, 128}, Done
{(0, 0), N, 0}, {(0, 1), N, 0}, {(0, 2), X, 128}, Done
{(0, 0), N, 0}, {(0, 1), E, 0}, {(1, 1), X, -64}, Done

(f) What are her Q-values after observing these episodes? Assume Ms. Pacman initialized her Q-values all to 0 and used a learning rate of 0.1. You may omit any Q-states that were unaffected.

Now let's review a couple problems we've seen before.

(g) For each of the following functions, write which MDP/RL value the function computes, or none if none apply. We are given an MDP (S, A, T, γ, R), where R is only a function of the current state s. We are also given an arbitrary policy π.

Possible choices: $V^*, Q^*, \pi^*, V^\pi, Q^\pi$.

(i) $f(s) = R(s) + \sum_{s'} \gamma T(s, \pi(s), s') f(s')$

(ii) $g(s) = \max_a \sum_{s'} T(s, a, s')[R(s) + \gamma \max_{a'} Q^*(s', a')]$

(iii) $h(s, a) = \sum_{s'} T(s, \pi(s), s')[R(s) + \gamma h(s', a)]$

(h) We are given a pre-existing table of Q-values (and its corresponding policy), and asked to perform ϵ-greedy Q-learning. Individually, what effect does setting each of the following constants to 0 have on this process?

(i) α:

(ii) γ:

(iii) ϵ:

(i) Why can't we use the MDP policy extraction formula to extract a policy in TD-learning or Q-learning?

3 CSP Backtracking Search

In this problem, you are given a 3×3 grid with some numbers filled in. The squares can only be filled with the numbers $\{2, 3, \ldots, 10\}$, with each number being used once and only once. The grid must be filled such that adjacent squares (horizontally and vertically adjacent, but not diagonally) are relatively prime.

x_1	x_2	x_3
x_4	x_5	3
4	x_6	2

We will use backtracking search to solve the CSP with the following heuristics:

- Use the Minimal Remaining Values (MRV) heuristic when choosing which variable to assign next.
- Break ties with the Most Constraining Variable (MCV) heuristic.
- If there are still ties, break ties between variables x_i, x_j with $i < j$ by choosing x_i.
- Once a variable is chosen, assign the minimal value from the set of feasible values.
- For any variable x_i, a value v is infeasible if and only if: (i) v already appears elsewhere in the grid, or (ii) a variable in a neighboring square to x_i has been assigned a value u where $\gcd(v, u) > 1$, which is to say, they are not relatively prime.

Fill out the table below with the appropriate values.

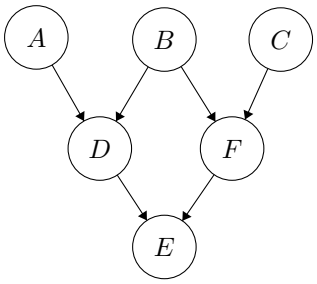
- Give initial feasible values in set form; x_1 has already been filled out for you.
- Assignment order refers to the order in which the final value assignments are given. If x_i is the j^{th} variable on the path to the goal state, then the assignment order for x_i is j.
- In the branching column, write "yes" if the algorithm branches (considers more than one value) at that node in the search tree, and write "B" if the algorithm backtracks at that node, meaning it is the highest node in its subtree that fails for a value, and has to be chosen again. Also write the values it tried then failed.

Variable	**Initial Feasible Values**	**Assignment Order**	**Final Value**	**Branch or Backtrack?**
x_1	$\{5, 6, 7, 8, 9, 10\}$			
x_2				
x_3				
x_4				
x_5				
x_6				

4 Bayes Nets

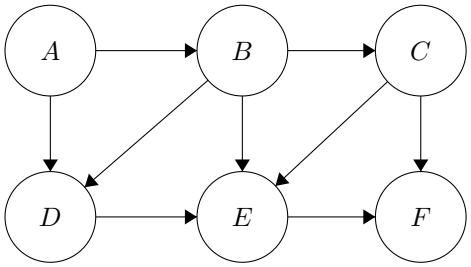
(a) For the following graphs, explicitly state the minimum size set of edges that must be removed such that the corresponding independence relations are guaranteed to be true.
Mark the removed edges with an 'X' on the graphs.

(i)



$A \perp\!\!\!\perp B \mid F$
$A \perp\!\!\!\perp F \mid D$
$B \perp\!\!\!\perp C$

(ii)

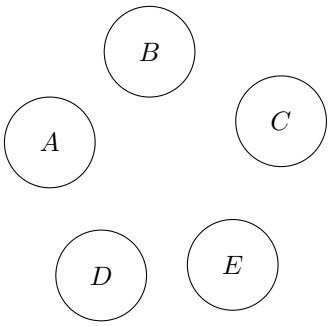


$A \perp\!\!\!\perp D \mid B$
$A \perp\!\!\!\perp F \mid C$
$C \perp\!\!\!\perp D \mid B$

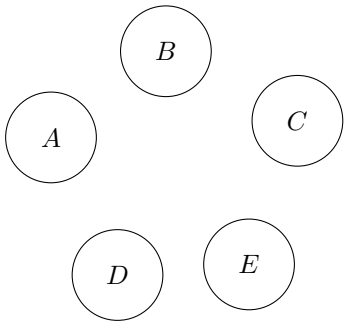
(b) You're performing variable elimination over a Bayes Net with variables A, B, C, D, E. So far, you've finished joining over (multiplying all factors containing C into one factor), but not summing out C, when you realize you've lost the original Bayes Net!

Your current factors are $f(A), f(B), f(B, D), f(A, B, C, D, E)$. Note: these are generic factors, NOT joint distributions. You don't know which variables are conditioned or unconditioned.

(i) What's the smallest number of edges that could have been in the original Bayes Net? Draw out one such Bayes Net below.



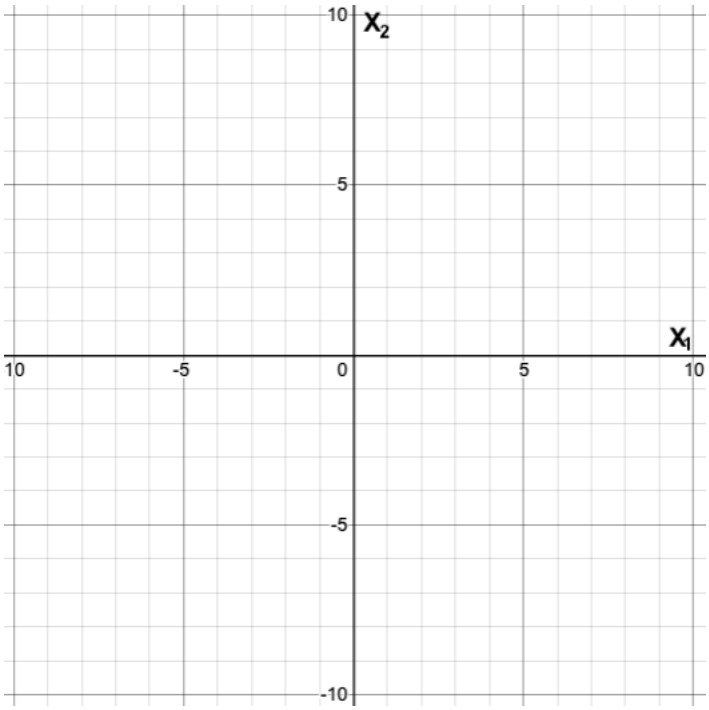
(c) What's the largest number of edges that could have been in the original Bayes Net? Draw out one such Bayes Net below.



5 Linear Programming & Integer Programming

(a) In the following optimization problem, plot the boundary lines for the three inequality constraints.

$$\min_x c^T x$$
$$\text{s.t. } Ax \leq b$$
$$A = \begin{bmatrix} 1 & 4 \\ 2 & -1 \\ -4 & -1 \end{bmatrix}, b = \begin{bmatrix} 5 \\ 8 \\ 8 \end{bmatrix}$$



(b) In your graph from part (a), mark the feasible region with an 'F'

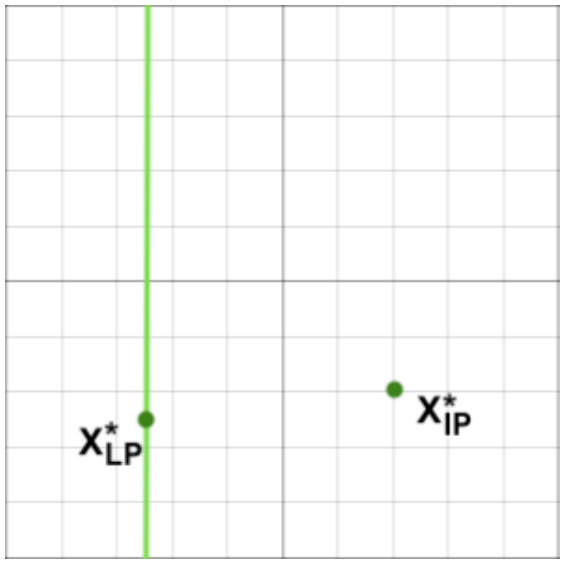
(c) In your graph from part (a), draw in a cost vector such that the optimal solution is the point (0, -8).

(d) List three cost vectors (that are not scaled versions of each other) that will lead to an infinite number of solutions.

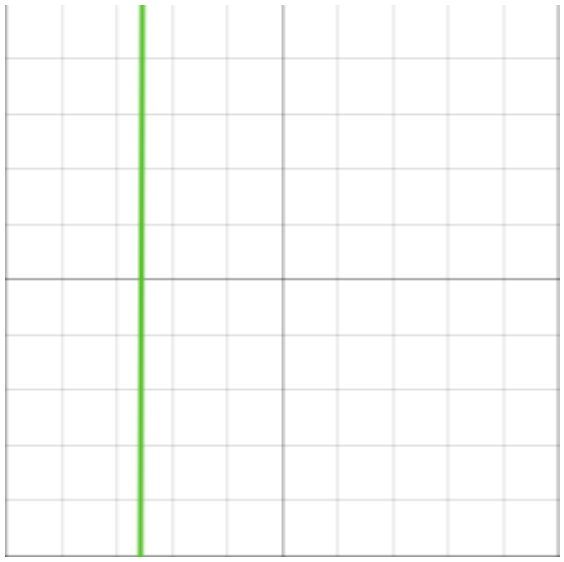
(e) Now, let us look at at the following constraint graphs. One inequality constraint has been drawn in.

For each:
-Choose a feasible region and mark it with an 'F'
-Draw two additional constraint boundaries such that the given conditions are met
-Draw a cost vector such that the given conditions are met

(i) The point x^*_{LP} is the linear programming solution and x^*_{IP} is the integer programming solution.



(ii) The minimum objective for the linear program is $-\infty$ and the integer program is infeasible.



(f) What is the outcome of running the branch and bound algorithm on each of the graphs from part (e)?

(i)

(ii)