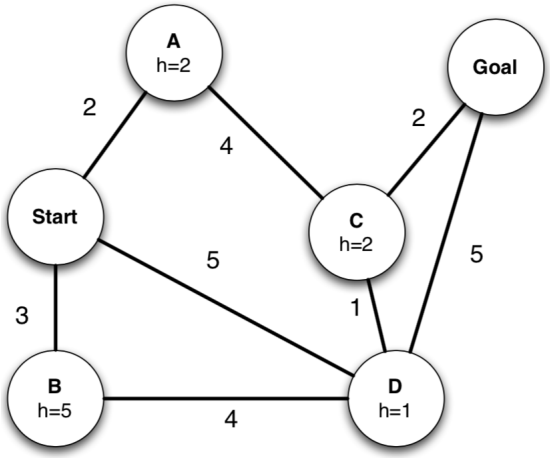
1 Search



For each of the following graph search strategies, work out the order in which states are expanded, as well as the path returned by graph search. In all cases, break ties in alphabetical order. The start and goal state use letter S and G, respectively. Remember that in graph search, a state is expanded only once.

(a) Depth-first search.

States Expanded: Start, A, C, D, B, Goal
Path Returned: Start-A-C-D-Goal

(b) Breadth-first search.

States Expanded: Start, A, B, D, C, Goal
Path Returned: Start-D-Goal

(c) Uniform cost search.

States Expanded: Start, A, B, D, C, Goal
Path Returned: Start-A-C-Goal

(d) Greedy search with the heuristic h shown on the graph.

States Expanded: Start, D, Goal
Path Returned: Start-D-Goal

(e) $A^\star$ search with the same heuristic.

States Expanded: Start, A, D, B, C, Goal
Path Returned: Start-A-C-Goal

2 Adversarial Search

Warm up

1. What is the advantage of adding alpha-beta pruning to a minimax algorithm

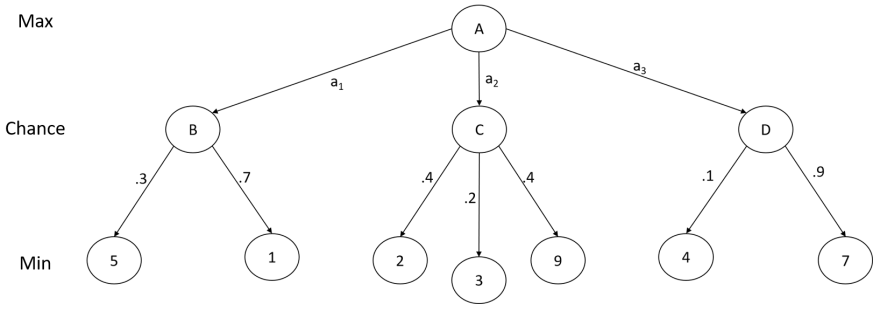
   It on average speeds up minimax algorithms by reducing the number of nodes that need to be examined.This is achieved by "pruning" nodes which have been found not to change the result produced by thealgorithm.

2. Give two advantages of Iterative Deepening minimax algorithms over Depth Limited minimax algorithms.

   I) Solution availability: You always have the solution of the previous iteration available during the execution of the current iteration (this is particularly useful when under a time constraint).

   II) Information gleaned during the current iteration can be employed to increase pruning in successive iterations (Recall HW2 Question 5). Because successive iterations require exponentially more time, and searching at lower depths is typically insignificant while increased pruning at higher depths can be very significant.

The three questions are about the following adversarial "chance" tree.



Expectiminimax

1. Calculate the EXPECTIMINIMAX values for nodes B, C and D in the above adversarial "chance"tree. Show your calculations!

   EXPECTIMINIMAX(B) = .3 * 5 + .7 * 1 = 1.5 + .7 = 2.2
   EXPECTIMINIMAX(C) = .4 * 2 + .2 * 3 + .4 * 9 = .8 + .6 + 3.6 = 5.0
   EXPECTIMINIMAX(D) = .1 * 4 + .9 * 7 = .4 + 6.3 = 6.7

2. Which action will MAX choose, a_1, a_2, or a_3? Explain your answer!

   MAX will choose action a_3 because it has the highest EXPECTIMINIMAX value.

3. If the utility values given for MIN were multiplied with a positive constant c, which action would MAX then choose?

   MAX would still choose action a_3 because multiplying with a positive constant is a positive linear transformation and such transformations do not change decisions made on the basis of EXPECTIMINIMAX values.

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	_____	_____	_____	_____

Variable	Initial Feasible Values	Assignment Order	Final Value	Branch or Backtrack?
x_1	$\{5, 6, 7, 8, 9, 10\}$	5	6	No
x_2	{5,6,7,8,9,10}	4	7	No
x_3	{5,7,8,10}	6	10	No
x_4	{5,7,9}	1	5	Yes
x_5	{5,7,8,10}	2	8	B: 7
x_6	{5,7,9}	3	9	B: 7

4 Local Search

(a) Which of the following local search algorithm are complete and/or optimal? If necessary, specify the conditions that must be true for completeness or optimality.

- First-choice Hill Climbing
  (i) Complete?
  No, the algorithm only takes uphill moves, so it could get "stuck" on a shoulder or local optima.
  (ii) Optimal?
  No
- Random-restart Hill Climbing
  (i) Complete?
  Yes, it is trivially complete with probability approaching 1 because it will eventually generate a goal state as its initial state.
  (ii) Optimal?
  No
- Simulated Annealing
  (i) Complete?
  Yes, the algorithm combines hill climbing with random walk and is able to take downhill moves.
  (ii) Optimal?
  No
- Genetic Algorithm
  (i) Complete?
  No, the algorithm combines exploration (crossover and mutation) with an uphill tendency that keeps the best states to further evolve (fitness function). This is similar to local beam search.
  (ii) Optimal?
  No
- Local Beam Search
  (i) Complete?
  No, the algorithm chooses the k best states at each iteration, which can result in a lack of diversity.
  (ii) Optimal?
  No

(b) Of the local search algorithms above, which one(s) would perform best in a continuous state space and why?

First-choice hill climbing and simulated annealing: both of these search algorithms do not have infinite branching factors, so they would be able to handle continuous state spaces. AIMA Page 129 includes more discussion on local search in continuous spaces.

(c) What are the disadvantages and advantages of allowing sideways moves? How can we modify our search algorithm to address the disadvantages?

Advantage: the algorithm can find a better state if we make a sideways move along a shoulder.
Disadvantage: allowing sideways moves could result in an infinite loop if we are at a local maximum. One potential modification to address this disadvantages would be to limit the number of consecutive sideways moves taken.

5 Propositional Logic

1. Warm Up: Are you familiar with these terms?

   - Symbols

     Variables that can be T/F (capital letter)

   - Operators

     and, or, not, implies, equivalent

   - Sentences

     Symbols connected with operators, can be T/F

   - Equivalence

     True in all models that a and b implies each other (a equivalent to b)

   - Literals

     atomic sentence

   - Knowledge Base

     Sentences agents know to be true

   - Entailment

     a entails b iff $\forall$ models, a true implies b true

   - Query

     A sentence we want to know whether it's true (usually we want to know whether KB entails q)

   - Satisfiable

     At least one model makes the sentence true

   - Valid

     True for all models

   - Clause - Definite, Horn clauses

     Clause - disjunction of literals; definite - clause with exactly one positive literal, horn - clause with at most one positive literal

   - Model Checking

     check if sentences are true in given model/checks entailment

   - Theorem Proving

     Search for a sequence of proof steps. (e.g. Forward Chaining)

   - Modus Ponens

     From $P, (P \longrightarrow Q)$, infer Q

2. Indicate whether the following sentence is *valid*, *satisfiable*, or *unsatisfiable*. If satisfiable, give a model such that the sentence is satisfied. Prove your answer by reducing the sentence to its simplest form. Remember to **show all the steps and write down an explanation of each step.** Let T stand for the atomic sentence *True* and F for the atomic sentence *False*.

   $((T \Leftrightarrow \neg(x \vee \neg x)) \vee z) \wedge \neg(z \wedge ((z \wedge \neg z) \Rightarrow x))$

   Unsatisfiable

$((T \Leftrightarrow \neg(x \vee \neg x)) \vee z) \wedge \neg(z \wedge ((z \wedge \neg z) \Rightarrow x))$	
$((T \Leftrightarrow (\neg x \wedge x)) \vee z) \wedge \neg(z \wedge ((z \wedge \neg z) \Rightarrow x))$	De Morgan's Law
$((T \Leftrightarrow F) \vee z) \wedge \neg(z \wedge (F \Rightarrow x))$	$a \wedge \neg a$ is equivalent to False
$(((T \Rightarrow F) \wedge (F \Rightarrow T)) \vee z) \wedge \neg(z \wedge (F \Rightarrow x))$	Biconditional Elimination
$(((\neg T \vee F) \wedge (\neg F \vee T)) \vee z) \wedge \neg(z \wedge (\neg F \vee x))$	Implication Elimination
$(((F \vee F) \wedge (T \vee T)) \vee z) \wedge \neg(z \wedge (T \vee x))$	Negation
$((F \wedge T) \vee z) \wedge \neg(z \wedge T)$	$F \vee F$ is $F, T \vee T$ is T, $T \vee a$ is T
$(F \vee z) \wedge \neg(z \wedge T)$	$F \wedge T$ is False
$(F \vee z) \wedge \neg z \vee \neg T)$	De Morgan's Law
$(F \vee z) \wedge (\neg z \vee F))$	Negation
$z \wedge \neg z$	$F \vee a$ is a
F	$a \wedge \neg a$ is F

6 Satisfiability and Planning

With the holidays coming up, Santa needs to start making a plan (and checking it twice) to deliver presents. First he tries taking a SATplan (logical planning) approach, and formulates the following propositions:

- $at(loc, t)$: Santa's sled is at location loc at time t
- $reindeerHunger(x, t)$: the reindeers' hunger level is at x at time t, $x \in [0, 5]$.
- $hasCarrots(loc, t)$: location loc has carrots to feed reindeer with at time t
- $hasPresents(loc, t)$: location loc has presents at time t

His starting state is $at(NorthPole, 0) \wedge reindeerHunger(0, 0)$.

1. Using the above predicates, formulate successor-state axioms for the actions $feedReindeer(t)$, $deliver(t)$, and $fly(origin, destination, t)$. Santa can only feed the reindeer at a location that has carrots, and their hunger level drops to 0 as a result of feeding. Santa can fly between any distinct locations, as long as the reindeers' hunger level is less than 3; after flying, their hunger level increases by 1. Santa can deliver presents anywhere, and the result is that the location of his sled now has presents.

   feedReindeer: $at(loc, t) \wedge reindeerHunger(x, t) \wedge feedReindeer(t) \wedge hasCarrots(loc) \iff reindeerHunger(0, t+1)$
   deliver: $at(loc, t) \wedge deliver(t) \iff hasPresents(loc, t+1)$
   fly: $at(origin, t) \wedge reindeerHunger(x, t) \wedge x < 3 \wedge fly(origin, destination, t) \wedge origin \neq destination \iff at(destination, t+1) \wedge reindeerHunger(x+1, t+1)$

2. Convert your *deliver* axiom into conjunctive normal form. You may want to abbreviate each proposition. What's the purpose of converting logical sentences into CNF (besides solving recitation problems)?

   Let $a = at(loc, t)$, $b = deliver(t)$, $c = hasPresents(loc, t+1)$. We start with $a \wedge b \Leftrightarrow c$.
   $((a \wedge b) \Rightarrow c) \wedge (c \Rightarrow (a \wedge b))$
   $(\neg(a \wedge b) \vee c) \wedge (\neg c \vee (a \wedge b))$
   $(\neg a \vee \neg b \vee c) \wedge (\neg c \vee a) \wedge (\neg c \vee b)$
   SAT solvers require input sentences to be in CNF, so we have to convert sentences we'd like to deduce the satisfiability of into CNF before using the solver. (Think back to P2!)

3. Suppose Santa's goal is to deliver presents to NorthPole. Describe an algorithm which uses a SAT solver to find a plan for this goal.

   Again, think back to P2 - we can iterate from $i = 0$ to some reasonable max timestep, on each iteration incrementally adding all the axioms with timestep i to our current knowledge base (which is initially the starting state) and running DPLL or some other SAT solver on (knowledge base $\wedge$ $hasPresents(NorthPole, i)$). Once we hit a timestep at which DPLL returns a model satisfying the sentence, we can extract the plan from that model.
   (We should also include axioms stating exactly only one action can be taken at every timestep, and that Santa must be in exactly one place at a given time).)

4. Run DPLL to determine whether the goal $hasPresents(NorthPole, 1)$ is feasible with our knowledge base $at(NorthPole, 0) \wedge reindeerHunger(0, 0) \wedge D$, where D is your *deliver* axiom instantiated with $loc = NorthPole$, $t = 0$ (we can leave the other axioms out because they won't be relevant). What's a possible model found by DPLL?

We want to determine whether the sentence $at(NP,0) \wedge reindeerHunger(0,0) \wedge (at(NP,0) \wedge deliver(t) \Leftrightarrow hasPresents(NP,1)) \wedge hasPresents(NP,1)$ is satisfiable. Using our CNF version of D from the question above, the full sentence we would pass into DPLL (with abbreviations) is as follows:

$$a(NP,0) \quad rH(0,0) \quad (\neg a(NP,0) \vee \neg d(NP,0) \vee hP(NP,1)) \quad (\neg hP(NP,1) \vee a(NP,0))$$
$$(\neg hP(NP,1) \vee d(NP,0)) \quad hP(NP,1)$$

Using the unit clause heuristic, DPLL assigns $a(NP,0)$, $rH(0,0)$, and $hP(NP,1)$ to be true. After these assignments, the only remaining unsatisfied clause is $(\neg hP(NP,1) \vee d(NP,0))$. Using either the pure symbol or unit clause heuristic, DPLL assigns $d(NP,0)$ to be true. Thus DPLL would return true, and the final (partial) model DPLL finds is $\{a(NP,0) : True, rH(0,0) : True, hP(NP,1) : True, d(NP,0) : True\}$.

5. Suppose DPLL returned $False$ on some sentence $A \wedge B$. What entailment conclusions can we draw involving A and B?

Rearranging the sentence, we get
$\neg(\neg A \vee \neg B)$
$\neg(A \Rightarrow \neg B)$. Based on the result from DPLL, we know this sentence is unsatisfiable, i.e. it is false in all models. That means its negation, $A \Rightarrow \neg B$, must be true in all models. By definition, this mean $A \models \neg B$.
A symmetric argument can be made to show that $B \models \neg A$.

6. Now Santa tries taking a GraphPlan approach. Define each action as an operator in the following table (note that we can drop the t parameter from each predicate and action):

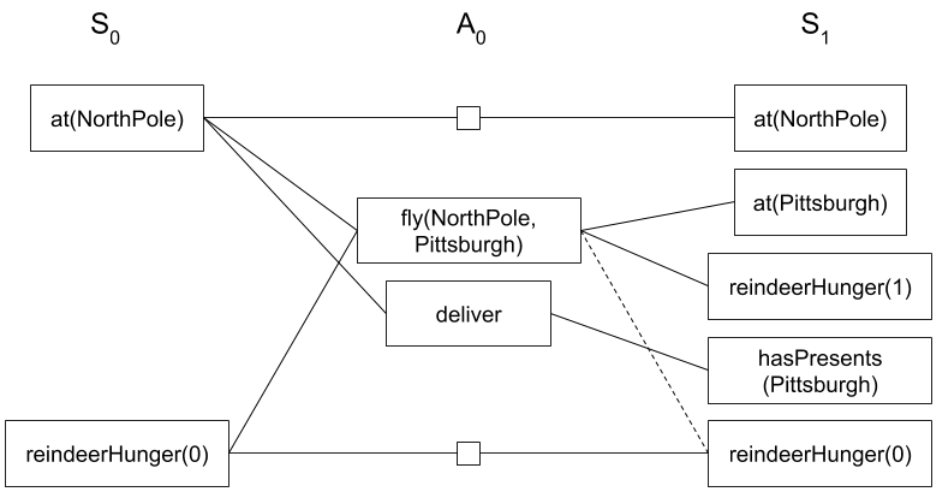
	$feedReindeer$	$fly(o,d)$	$deliver$
Precondition			
Add			
Delete			

	$feedReindeer$	$fly(o,d)$	$deliver$
Precondition	$at(loc)$, $hasCarrots(loc)$, $reindeerHunger(x)$	$at(o)$, $reindeerHunger(x)$, $x < 3$, $o \neq d$	$at(loc)$
Add	$reindeerHunger(0)$	$reindeerHunger(x+1)$, $at(d)$	$hasPresents(loc)$
Delete	$reindeerHunger(x)$	$reindeerHunger(x)$	

7. Which of the operators are mutually exclusive? What type of mutex relation does each pair have?

$feedReindeer$ and $fly(o,d)$ with any o,d, because the former sets the reindeer hunger to 0, while the latter results in a nonzero reindeer hunger. This is an **inconsistency** mutex relation.

8. Now draw the GraphPlan graph up to proposition level S_1. Suppose Pittsburgh is the only other location besides NorthPole.



9. Which operators are mutually exclusive in A_0? Which propositions are mutually exclusive in S_1?

   noop and fly(NorthPole, Pittsburgh) have an inconsistent effects mutex relation.
   at(Pittsburgh) and at(NorthPole) are mutex; reindeerHunger(0) and reindeerHunger(1) are mutex.

10. In general, when does GraphPlan stop extending the planning graph?

    GraphPlan stops extending the graph once it reaches a proposition level where all goal propositions are present, or when the graph levels off, i.e., two consecutive proposition levels are identical.

11. Is GraphPlan sound? complete? optimal? What about the SATPlan algorithm you described above?

    GraphPlan is sound, complete, but not optimal (with respect to the number of actions in the plan returned, which is generally the metric we use).
    SATPlan as we've described above is sound, complete, and optimal.

7 First Order Logic

(a) For each of the logical expressions, state whether it correctly expresses the English sentence and explain.

i. Everyone in 281 is awesome: $\forall x$ In(x, 281) $\wedge$ Awesome(x)

No it is incorrect. This translates to everyone is in 281 and everyone is awesome. Thus, if someone isn't in 281, or if someone in the world isn't awesome, this expression becomes false, which is clearly not equivalent to the English sentence.

Typically, => is the main connective with $\forall$

ii. Someone taking 281 actually dislikes the class: $\exists x$ In(x, 281) => Dislikes(x, 281).

No it is incorrect. In the logical expression, if there exists a student that's not in 281, this expression becomes $True$ (by implication rule). That's unrelated to our English expression, which btw is also clearly $False$;)

Typically, $\wedge$ is the main connective with $\exists$

(b) For each pair of atomic sentences, give the most general unifier if it exists:

i. ***Q(f(a), f(b)), Q(f(x),f(x))***

No unifier exists. x cannot bind to both a and b.

ii. ***R(a, a, b, f(b)), R(f(x), f(m(5)), x, y)***

{a/f(x), x/m(5), b/x, y/f(b)}

8 Bayes' Nets: Representation, Independence

For this problem, any answers that require division can be left written as a fraction.
PacLabs has just created a new type of mini power pellet that is small enough for Pacman to carry around with him when he's running around mazes. Unfortunately, these mini-pellets don't guarantee that Pacman will win all his fights with ghosts, and they look just like the regular dots Pacman carried around to snack on.
Pacman (P) just ate a snack , which was either a mini-pellet ($+p$), or a regular dot ($-p$), and is about to get into a fight (W), which he can win ($+w$) or lose ($-w$). Both these variables are unknown, but fortunately, Pacman is a master of probability. He knows that his bag of snacks has 5 mini-pellets and 15 regular dots. He also knows that if he ate a mini-pellet, he has a 70% chance of winning, but if he ate a regular dot, he only has a 20% chance.

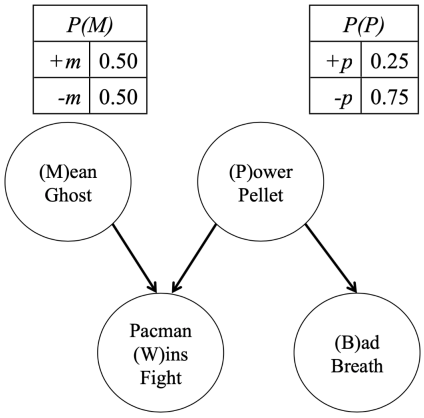
(a) What is $P(+w)$, the marginal probability that Pacman will win?

$$\begin{aligned} P(+w) &= \sum_p P(+w|p)P(p) \\ &= \frac{7}{10} \times \frac{1}{4} + \frac{2}{10} \times \frac{3}{4} = \frac{13}{40} = 0.325 \end{aligned}$$

(b) Pacman won! Hooray! What is the conditional probability $P(+p \mid +w)$ that the food he ate was a mini-pellet, given that he won?

$$\begin{aligned} P(+p|+w) &= \frac{P(+w,+p)}{P(+w)} = \frac{P(+w|+p)P(+p)}{P(+w)} \\ &= \frac{\frac{7}{10} \times \frac{1}{4}}{\frac{13}{40}} = \frac{7}{13} \approx 0.538 \end{aligned}$$

Pacman can make better probability estimates if he takes more information into account. First, Pacman's breath, B, can be bad ($+b$) or fresh ($-b$). Second, there are two types of ghost (M): mean ($+m$) and nice ($-m$). Pacman has encoded his knowledge about the situation in the following Bayes' Net:



P(M)	
+m	0.50
-m	0.50

P(P)	
+p	0.25
-p	0.75

P(W|M, P)			
+m	+p	+w	0.60
+m	+p	-w	0.40
+m	-p	+w	0.10
+m	-p	-w	0.90
-m	+p	+w	0.80
-m	+p	-w	0.20
-m	-p	+w	0.30
-m	-p	-w	0.70

P(B|P)		
+p	+b	0.80
+p	-b	0.20
-p	+b	0.40
-p	-b	0.60

(c) What is the probability of the event $(-m, +p, +w, -b)$, where Pacman eats a mini-pellet and has fresh breath before winning a fight against a nice ghost?

$P(-m, +p, +w, -b) = P(-m)P(+p)P(+w| - m, +p)P(-b| + p) = \frac{1}{2} \times \frac{1}{4} \times \frac{4}{5} \times \frac{1}{5} = \frac{1}{50} = 0.02$

(d) Which of the following conditional independence statements are guaranteed to be true by the Bayes' Net graph structure?

i. $W \perp\!\!\!\perp B$
ii. $W \perp\!\!\!\perp B|P$
iii. $M \perp\!\!\!\perp P$
iv. $M \perp\!\!\!\perp P|W$
v. $M \perp\!\!\!\perp B$
vi. $M \perp\!\!\!\perp B|P$
vii. $M \perp\!\!\!\perp B|W$

ii, iii, v, vi

For the remaining of this question, use the half of the joint probability table that has been computed for you below:

$P(M, P, W, B)$				
$+m$	$+p$	$+w$	$+b$	0.0800
$+m$	$+p$	$+w$	$-b$	0.0150
$+m$	$+p$	$-w$	$+b$	0.0400
$+m$	$+p$	$-w$	$-b$	0.0100
$+m$	$-p$	$+w$	$+b$	0.0150
$+m$	$-p$	$+w$	$-b$	0.0225
$+m$	$-p$	$-w$	$+b$	0.1350
$+m$	$-p$	$-w$	$-b$	0.2025

(e) What is the marginal probability, $P(+m, +b)$ that Pacman encounters a mean ghost and has bad breath?

$P(+m, +b) = 0.08 + 0.04 + 0.015 + 0.135 = 0.27$

(f) Pacman observes that he has bad breath and that the ghost he's facing is mean. What is the conditional probability, $P(+w \mid +m, +b)$, that he will win the fight, given his observations?

$P(+w| + m, +b) = \frac{P(+w,+m,+b)}{P(+m,+b)} = \frac{0.08+0.015}{0.27} = \frac{19}{54} \approx 0.352$

(g) Pacman's utility is +10 for winning a fight, -5 for losing a fight, and -1 for running away from a fight. Pacman wants to maximize his expected utility. Given that he has bad breath and is facing a mean ghost, should he stay and fight, or run away? Justify your answer.

Let U_f be the utility of fighting and U_r be the utility of running.

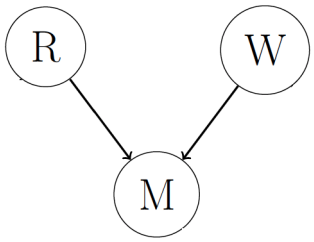
$$\begin{aligned} E(U_f| + m, +b) &= 10 \times P(+w| + m, +b) + (-5) \times P(-w| + m, +b) \\ &\approx 10 \times 0.352 - 5 \times 0.648 \\ &= 0.28 > -1 = U_r \end{aligned}$$

Since $E(U_f| + m, +b) > E(U_r| + m, +b)$, Pacman should stay and fight.

9 Bayes' Nets: Sampling

Consider the following Bayes Net and corresponding probability tables.

R	P(R)
+r	0.4
-r	0.6



W	P(W)
+w	0.2
-w	0.8

M	R	W	P(M | R,W)
+m	+r	+w	0.1
-m	+r	+w	0.9
+m	+r	-w	0.45
-m	+r	-w	0.55
+m	-r	+w	0.35
-m	-r	+w	0.65
+m	-r	-w	0.9
-m	-r	-w	0.1

Consider the case where we are sampling to approximate the query $P(R \mid +m)$.

Fill in the following table with the probabilities of drawing each respective sample given that we are using each of the following sampling techniques. Let $P(+m) = a$.

Method	$< +r, -w, +m >$	$< +r, +w, -m >$
Prior sampling	0.4*0.8*0.45 = 0.144	0.4*0.2*0.9 = 0.072
Rejection sampling	$\frac{P(+r,-w,+m)}{P(+m)} = \frac{0.144}{a}$	0
Likelihood weighting	$P(+r)P(-w) = 0.4 * 0.8 = 0.32$	0

We are going to use Gibbs sampling to estimate the probability of getting the sample $< +r, -w, +m >$. We will start from the sample $< -r, +w, +m >$ and resample W first then R. What is the probability of drawing sample $< +r, -w, +m >$?

$$P(-w \mid -r, +m) = \frac{P(-w, -r, +m)}{\sum_w P(W = w, -r, +m)} = \frac{0.8 * 0.6 * 0.9}{0.8 * 0.6 * 0.9 + 0.2 * 0.6 * 0.35} = \frac{0.432}{0.474} = 0.9114$$

$$P(+r \mid -w, +m) = \frac{P(+r, -w, +m)}{\sum_r P(R = r, -w, +m} = \frac{0.4 * 0.8 * 0.45}{0.4 * 0.8 * 0.45 + 0.6 * 0.8 * 0.9} = \frac{0.144}{0.576} = 0.25$$

The probability of sampling $(+r, -w, +m,)$ is the product of the two sampling probabilities. So 0.9114 * 0.25 = 0.228.

10 HMMs and Particle Filtering

Consider the following Markov Model with a binary state X (i.e. X_t is either 0 or 1). The transition probabilities and initial distribution are as follows:

X_0	$P(X_0)$
0	0.5
1	0.5

X_t	X_{t+1}	$P(X_{t+1}\|X_t)$
0	0	0.9
0	1	0.1
1	0	0.5
1	1	0.5

(a) After one timestep, what is the new belief distribution $P(X_1)$?

X_1	$P(X_1)$
0	
1	

X_1	$P(X_1)$
0	$.5 * .9 + .5 * .5 = .7$
1	$.5 * .1 + .5 * .5 = .3$

Now, we incorporate sensor readings as our observations. The sensor model is parameterized by some value $\beta \in [0, 1]$:

X_t	E_t	$P(E_t\|X_t)$
0	0	β
0	1	$1 - \beta$
1	0	$1 - \beta$
1	1	β

(b) At $t = 1$, we get the first sensor reading, $E_1 = 0$. Find $P(X_1 = 0|E_1 = 0)$ in terms of β.

$$P(X_1 = 0|E_1 = 0) = \frac{P(E_1 = 0|X_1 = 0)P(X_1 = 0)}{\sum_x P(E_1 = 0|X_1 = x)P(X_1 = x)}$$
$$= \frac{\beta * .7}{\beta * .7 + (1 - \beta) * .3}$$

(c) For what range of values of β will a sensor reading $E_1 = 0$ increase our belief that $X_1 = 0$? In other words, what is the range of β for which $P(X_1 = 0|E_1 = 0) > P(X_1 = 0)$?

$\beta \in (0.5, 1]$. Intuitively, observing $E_1 = 0$ will only increase the belief that $X_1 = 0$ if $E_1 = 0$ is more likely under $X_1 = 0$ than not. We specify $\beta > 0.5$ because $\beta = 0.5$ is uninformative since the initial distribution is uniform.

(d) Now, we want to use particle filtering to predict what state value our model currently assumes. At time t, there are 2 particles in state value 0, and 3 particles in state value 1. What is the prior belief distribution $\hat{P}(X_t)$?

$\hat{P}(X_t = 0) = 2/5$, $P(X_t = 1) = 3/5$

(e) At time t, we receive our first sensor reading $E_t = 1$. Given $\beta = 0.6$ and the previous table for $P(E_t|X_t)$, how many particles will be in each state value after resampling? For a source of randomness, use this list of numbers: [0.182, 0.703, 0.471, 0.859, 0.382]

We can first find the joint probability $\hat{P}(X_t, E_t)$:
$\hat{P}(X_t = 0, E_t = 1) = \hat{P}(X_t = 0) * P(E_t = 1|X_t = 0) = 2/5 * (1 - 0.6) = 2/5 * 0.4 = 0.16$
$\hat{P}(X_t = 1, E_t = 1) = \hat{P}(X_t = 1) * P(E_t = 1|X_t = 1) = 3/5 * 0.6 = .36$

We normalize to get the posterior $\hat{P}(X_t|E_t)$:
$\hat{P}(X_t = 0|E_t = 1) = .16/(.16 + .36) = .307$
$\hat{P}(X_t = 1|E_t = 1) = .36/(.16 + .36) = .693$

Fixing an order ($\hat{P}(X_t = 0|E_t = 1)$, $\hat{P}(X_t = 1|E_t = 1)$) and resampling using the given random number list would give us: [0, 1, 1, 1, 1]. There will be 4 particles with state value 1, and 1 particle with state value 0.

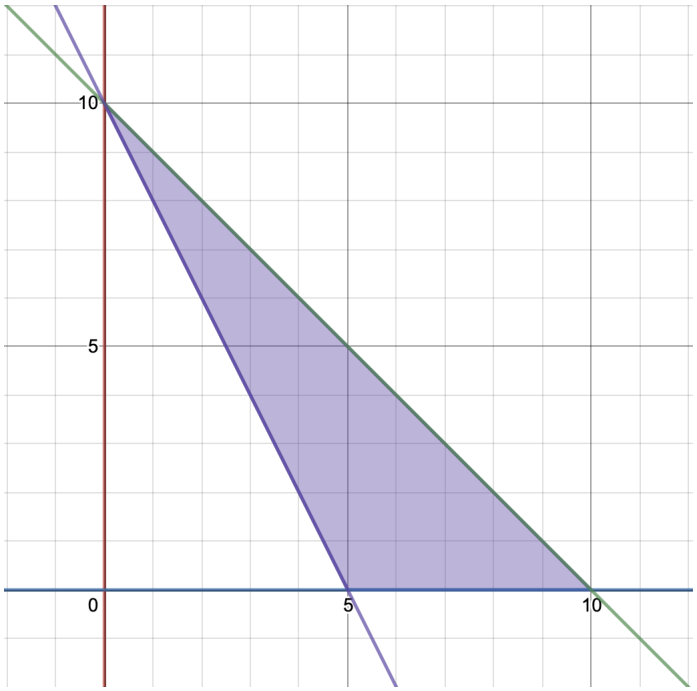
11 LP

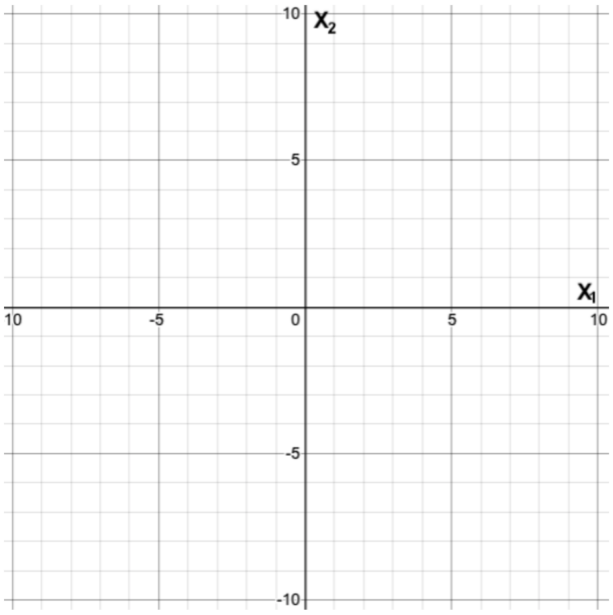
Santa is struggling to pack all the presents that he wants to deliver onto his sled. This year, Santa is giving out massive amounts of chocolate and peppermints.

Let x_1 represent pounds of chocolate and let x_2 represent pounds of peppermints. We assume we can deliver a fraction of a pound of chocolate or peppermints. Unfortunately, Santa's sled can only fit 10 pounds of sweets. Furthermore, Santa wants to provide enough presents for 10 kids. Each pound of chocolate is enough for 2 kids while a pound of peppermint is only enough for 1 kid. However, Santa also wants to maximize the children's happiness. Chocolate brings 4 units of happiness while peppermints only bring 1.

1. Represent the following problem as an LP and graph the constraints in the provided graph.

$$\min_x c^T x \text{ st } Ax \leq b \text{ where A} = \begin{bmatrix} -1 & 0 \\ 0 & -1 \\ 1 & 1 \\ -2 & -1 \end{bmatrix}, \text{ b} = \begin{bmatrix} 0 \\ 0 \\ 10 \\ -10 \end{bmatrix}, c = \begin{bmatrix} -4 \\ -1 \end{bmatrix}$$





2. What would the optimal solution be?

   The optimal point would be (10, 0).

3. If Santa didn't know how much happiness chocolate and peppermints bring, what would be a cost vector that makes the optimal solution (0, 10)?

   A possible cost vector would be [-1, -4].

4. List three cost vectors that will lead to an infinite number of solutions.

   $\begin{bmatrix}2\\1\end{bmatrix}, \begin{bmatrix}-1\\-1\end{bmatrix}, \begin{bmatrix}0\\1\end{bmatrix}$

12 MDPs/RL

1. Warm Up

   - What does the Markov Property state?

     Markov Property states that action outcomes depend on the current state only. It states that action outcomes do not depend on the past.

   - What are the Bellman Equations, and when are they used?

     The Bellman Equations give a definition of "optimal utility" via expectimax recurrence. They give a simple one-step lookahead relationship amongst optimal utility values.

   - What is a policy? What is an optimal policy?

     A policy is a function that maps states to actions. $\pi(s)$ gives an action for state s. An optimal policy is a policy that maximizes the expected utility if an agent follows it.

   - How does the discount factor γ affect the agent's policy search? Why is it important?

     γ determines how much the value of a state should take into account future states. The higher the discount factor, the more one state would value distant states. Having $0 < \gamma < 1$ also helps our algorithms converge.

   - What are the two steps to Policy Iteration?

     Policy evaluation and policy improvement.

   - What is the relationship between $V^*(s)$ and $Q(s, a)$?

     $V^*(s) = \max_a Q(s, a)$

   - Exploration, exploitation, and the difference between them? Why are they both useful?

     Exploration: trying out unknown actions; Exploitation: Following the known policy.
     Exploration allows the agent to see if there are any other actions that lead to a better reward by taking random actions. Exploitation guarantees that the agents get some reward at least.

   - What is the difference between on-policy and off-policy learning?

     For on-policy learning, it attempts to evaluate and improve the policy that is being used to make decisions. For off-policy learning, it attempts to evaluate or improve a policy different from the one that is being used to generate the data.

   - What is the difference between model-based and model-free learning?

     For model-based learning, the agent learns an approximate model based on experiences and solves for values as if the learned model were correct. A model-free learning is an algorithm which does not use the transition probability distribution.

   - 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:

     α is the the learning rate. It determines by how much the q-values should change each iteration given the new information found. The smaller α, the slower the policy will approach a solution, but the more accurate the solution would be; thus,
     $Q(s, a) = Q(s, a) + \alpha[r + \gamma max_{a'} Q(s', a') - Q(s, a)]$ becomes $Q(s, a)$.
     We put 0 weight on newly observed samples, never updating the Q-values we already have.

(iii) ϵ:

By definition of an ϵ-greedy policy, we randomly select actions with probability 0 and select our policy's recommended action with probability 1; we exclusively exploit the policy we already have.

- 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: $V^*, Q^*, \pi^*, V^\pi, Q^\pi$.

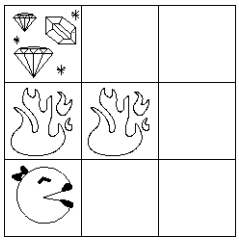
  (i) $f(s) = R(s) + \sum_{s'} \gamma T(s, \pi(s), s') f(s')$

  $f = V^\pi$. This is only different from the given formula for $V^\pi(s)$ on the formula sheet in that the reward function only depends on s here. Thus, we consider $R(s)$ outside the summation over s' - and do not discount it (because the reward is wrt. our current state).

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

  $g = V^*$. What this function does is essentially extract optimal values from optimal Q-values. Of our possible actions, we take the actions that yields the max sum of reward + future discounted rewards given by Q^*, summed over all possible successor states (each weighted by the successor's probability).

2. Ms.Pacman: 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?

128	64	32
-64	-64	16
2	4	8

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

Run Q-value iteration (Q-iteration on the MDP/RL notation sheet) until convergence.

(c) What's the optimal policy?

X	W	W
X	X	N
E	E	N

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).

We'll keep most of the original evaluation formula, but additionally sum over all possible actions recommended by the policy, each weighted by the probability of taking that action via the policy:

$$V^{\pi}_{k+1}(s) = \sum_{a} \pi(s, a) \sum_{s'} P(s' \mid s, a)[R(s, a, s') + \gamma V^{\pi}_{k}(s')]$$

3. Consider the fictitious play matching pennies example from lecture. A (rows) wants to match and B (columns) does not.

A/B	Heads	Tails
Heads	1,-1	-1,1
Tails	-1,1	1,-1

We will play a few more rounds to understand how fictitious play can arrive at a Nash equilibrium. Remember that w(a) refers to the number of times the opponent plays action a.

Round	A action	B action	A's belief for w_B	B's belief for w_A
0			(1,1)	(1,1)
1	H	H	(2,1)	(2,1)
2				
3				
4				

(a) Play round 2. What action will A choose? What action will B choose? Update the beliefs accordingly.

A wants to match, and believes that B is more likely to choose heads, so A will choose heads. B does not want to match, and believes that A is more likely to choose heads, so B will choose tails.

(b) Fill out the table completely by simulating all the rounds. Assume that both players break ties by choosing heads. What are the final beliefs? Do you notice interesting changes in the players' strategies? What do you think the equilibrium state is?

Round	A action	B action	A's belief for w_B	B's belief for w_A
0			(1,1)	(1,1)
1	H	H	(2,1)	(2,1)
2	H	T	(2,2)	(3,1)
3	H	T	(2,3)	(4,1)
4	T	T	(2,4)	(4,2)

(c) Will this game converge? What do you think is the Nash equilibrium?

Yes, because it is zero-sum. The Nash equilibrium is where each player chooses heads or tails with equal probability.

13 Game Theory

1. Warm Up

   (a) What is a strategy?

   A strategy is a probability distribution over actions.

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

   A pure strategy is deterministic (chosen with probability 1) while a mixed strategy uses randomized selection.

   (c) What is a Nash Equilibrium?

   A strategy profile where none of the participants benefit from unilaterally changing their decision

   (d) Does a Nash Equilibrium always exist?

   Yes if there is a finite number of players and a finite number of actions

2. Nash Equilibria

   With the Grinch now reformed, he has started helping Santa deliver presents. They have different preferred towns to deliver to but they are both happier if they are delivering presents together. The following table denotes the expected payoffs for Santa and the Grinch for the two different directions they can travel in.

		The Grinch	
		North	South
SANTA	North	(8, 8)	(1, 1)
	South	(3, 4)	(2, 9)

   (a) Identify the pure strategy Nash Equilibrium in this game.

   The pure strategy Nash Equilibria are (North, North) and (South, South).

   (b) Determine the mixed strategy Nash Equilibrium in this game.

   Santa's strategy is $(\frac{5}{6}, \frac{1}{6})$ and the Grinch's strategy is $(\frac{1}{6}, \frac{5}{6})$.

   Let p and (1-p) be the probability that Santa chooses to go North and South, respectively.
   Let q and (1-q) be the probability that the Grinch goes North and South, respectively.

   Utilities for Santa:
   North: $8(q) + 1(1-q) = 7q + 1$
   South: $3q + 2(1-q) = q + 2$
   If Santa decides to randomize, that means that he is indifferent between the two actions so the utilities must be equal. $7q + 1 = q + 2$ gives us $q = \frac{1}{6}$.

   Utilities for the Grinch:
   North: $8p + 4(1-p) = 4p + 4$
   South: $7p + 9(1-p) = 9 - 2p$
   The Grinch is indifferent between the two options so the utilities of the two must be equal. Thus, $4p + 4 = 9 - 2p$ so we get $p = \frac{5}{6}$.

3. Voting Rules

(a) Match each voting rule, axiom, or property to its description.

(i) _____ For $m-1$ rounds, each voter gets 1 vote per round and alternative with least plurality votes per round is eliminated. Alternative left is the winner.

(ii) _____ Each voter give one vote to top ranked preference, alternative with most votes wins.

(iii) _____ Voter can never benefit from lying about preferences.

(iv) _____ Alternative that beats every other alternative in pairwise election.

(v) _____ Alternative x beats y if majority of voters prefer x to y.

(vi) _____ First round: top two plurality winners advance to second round. Second round: pairwise election between the two.

(vii) _____ Each voter awards $m-k$ votes to their rank k, alternative with most votes wins.

i. Plurality
ii. Borda Count
iii. Plurality with runoff
iv. Single Transferable Vote
v. Pairwise election
vi. Condorcet winner
vii. Strategyproof

(i) iv. Single Transferable Vote

(ii) i. Plurality

(iii) vii. Strategyproof

(iv) vi. Condorcet winner

(v) v. Pairwise Election

(vi) iii. Plurality with runoff

(vii) ii. Borda Count

(b) Which voting rule(s) (Plurality, Borda Count, Plurality with Runoff, Single Transferable Vote) are **Condorcet consistent**? It may help to consider the following two examples.

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

None of the voting rules are Condorcet Consistent

(c) Which voting rule(s) (Plurality, Borda Count, Plurality with Runoff, Single Transferable Vote) are **strategyproof**?

None of these are strategyproof