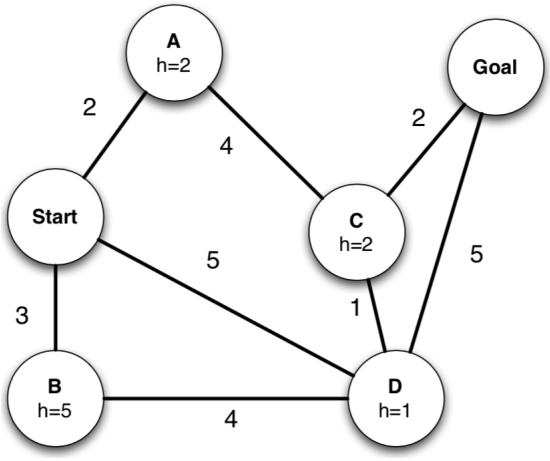
1 Search



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, Goal
Path Returned: Start-A-C-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 values 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?

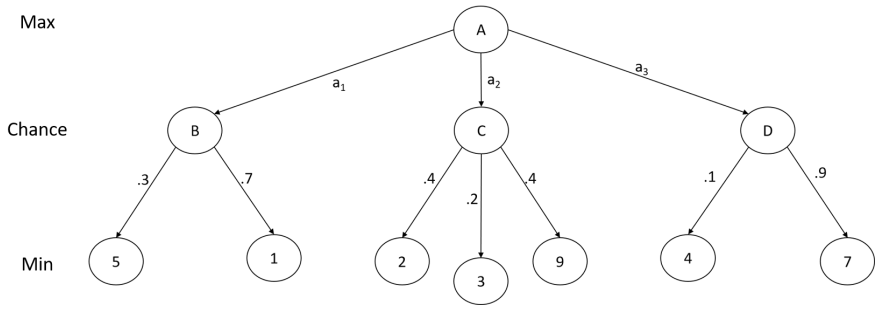
   It on average speeds up minimax algorithms by reducing the number of nodes that need to be examined.This is achieved by "pruning" nodes which have been found not to change the result produced by thealgorithm.

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 following 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.

   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? Why?

   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 (for example, x_4 considers more than 1 value), 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

To get started, we can assign 5 to x_4. This forces us to cross 5 off the remaining values for each of the other variables. We additionally have to cross off 10 from x_1 and x_5 since 5 and 10 are not relatively prime. Next, we assign 7 to x_5. Then, we assign 9 to x_6 since this is the only value left in its domain. Next, we can consider the value 6 for x_1. However, that leaves no value for x_2 to take on so instead, we backtrack and consider changing the values of x_5 and x_6 until we settle on 8 for x_5 and 9 for x_6. This leaves the values of 7, 6, and 10 for x_2, x_1, and x_3 respectively (the process is more clearly laid out in the following diagram).

① Assign $x_4 = 5$ (break tie with x_6 using MCV)

x_1	x_2	x_3
x_4 5	x_5	3
4	x_6	2

x_1 : ~~5~~ ~~6~~ 7 8 9 ~~10~~

x_2 : ~~5~~ 6 7 8 9 10

x_3 : ~~5~~ 7 8 10

x_4 : (5) 7 9

x_5 : ~~5~~ 7 8 10

x_6 : ~~5~~ 7 9

② Assign $x_5 = 7$ (break tie with x_6 using MCV)

x_1	x_2	x_3
x_4 5	x_5 7	3
4	x_6	2

x_1 : ~~5~~ ~~6~~ ~~7~~ 8 9 ~~10~~

x_2 : ~~5~~ 6 ~~7~~ 8 9 10

x_3 : ~~5~~ ~~7~~ 8 10

x_4 : (5) ~~7~~ 9

x_5 : ~~5~~ (7) 8 ~~10~~

x_6 : ~~5~~ ~~7~~ 9

③ Assign $x_6 = 9$

x_1	x_2	x_3
x_4 5	x_5 7	3
4	x_6 9	2

x_1 : ~~5~~ ~~6~~ ~~7~~ 8 ~~9~~ ~~10~~

x_2 : ~~5~~ 6 ~~7~~ 8 ~~9~~ 10

x_3 : ~~5~~ ~~7~~ 8 10

x_4 : (5) ~~7~~ ~~9~~

x_5 : ~~5~~ (7) 8 ~~10~~

x_6 : ~~5~~ ~~7~~ (9)

④ Assign $x_1 = 6$. this leaves no possible values in the domain of x_2 so we must backtrack!

x_1	x_2	x_3
x_1 6		
x_4 5	x_5 7	3
4	x_6 9	2

x_1 : ~~5~~ (6) ~~7~~ 8 ~~9~~ ~~10~~

x_2 : ~~5~~ ~~6~~ ~~7~~ ~~8~~ ~~9~~ ~~10~~

x_3 : ~~5~~ ~~7~~ 8 10

x_4 : (5) ~~7~~ ~~9~~

x_5 : ~~5~~ (7) 8 ~~10~~

x_6 : ~~5~~ ~~7~~ (9)

⑤ After trying out combos of values for x_5 and x_6 we settle on $x_5 = 8$, $x_6 = 9$

x_1	x_2	x_3
x_4 5	x_5 8	3
4	x_6 9	2

x_1 : ~~5~~ ~~6~~ 7 ~~8~~ ~~9~~ ~~10~~

x_2 : ~~5~~ ~~6~~ 7 ~~8~~ ~~9~~ ~~10~~

x_3 : ~~5~~ 7 ~~8~~ 10

x_4 : (5) 7 ~~9~~

x_5 : ~~5~~ 7 (8) ~~10~~

x_6 : ~~5~~ 7 (9)

⑥ Assign $x_2 = 7$

x_1	x_2	x_3
	x_2 7	
x_4 5	x_5 8	3
4	x_6 9	2

x_1 : ~~5~~ ~~6~~ ~~7~~ ~~8~~ ~~9~~ ~~10~~

x_2 : ~~5~~ 6 (7) ~~8~~ ~~9~~ ~~10~~

x_3 : ~~5~~ 7 ~~8~~ 10

x_4 : (5) 7 ~~9~~

x_5 : ~~5~~ 7 (8) ~~10~~

x_6 : ~~5~~ 7 (9)

⑦ Assign $x_1 = 6$

x_1	x_2	x_3
x_1 6	x_2 7	
x_4 5	x_5 8	3
4	x_6 9	2

x_1 : ~~5~~ (6) ~~7~~ ~~8~~ ~~9~~ ~~10~~

x_2 : ~~5~~ ~~6~~ (7) ~~8~~ ~~9~~ ~~10~~

x_3 : ~~5~~ ~~7~~ ~~8~~ 10

x_4 : (5) 7 ~~9~~

x_5 : ~~5~~ 7 (8) ~~10~~

x_6 : ~~5~~ 7 (9)

⑧ Assign $x_3 = 10$

x_1	x_2	x_3
x_1 6	x_2 7	10
x_4 5	x_5 8	3
4	x_6 9	2

x_1 : ~~5~~ (6) ~~7~~ ~~8~~ ~~9~~ ~~10~~

x_2 : ~~5~~ ~~6~~ (7) ~~8~~ ~~9~~ ~~10~~

x_3 : ~~5~~ ~~7~~ ~~8~~ (10)

x_4 : (5) 7 ~~9~~

x_5 : ~~5~~ 7 (8) ~~10~~

x_6 : ~~5~~ 7 (9)

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

     https://www.overleaf.com/project/5e9122f51d9c8a00013057c8 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

3. 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*.

   $(\neg(x \vee \neg x) \wedge y) \vee ((x \vee (z \Rightarrow \neg z)) \wedge ((x \Rightarrow z) \vee \neg(F \Rightarrow T)))$

   Satisfiable. {x: T, z: F}

$(\neg(x \vee \neg x) \wedge y) \vee ((x \vee (z \Rightarrow \neg z)) \wedge ((z \Rightarrow x) \vee \neg(F \Rightarrow T)))$	
$(\neg(T) \wedge y) \vee ((x \vee (z \Rightarrow \neg z)) \wedge ((z \Rightarrow x) \vee \neg(F \Rightarrow T)))$	$x \vee \neg x$ is True
$(F \wedge y) \vee ((x \vee (z \Rightarrow \neg z)) \wedge ((z \Rightarrow x) \vee \neg(F \Rightarrow T)))$	Negation
$F \vee ((x \vee (z \Rightarrow \neg z)) \wedge ((z \Rightarrow x) \vee \neg(F \Rightarrow T)))$	$F \wedge T$ is False
$(x \vee (z \Rightarrow \neg z)) \wedge ((z \Rightarrow x) \vee \neg(F \Rightarrow T))$	$F \vee a$ is a
$(x \vee (z \Rightarrow \neg z)) \wedge ((z \Rightarrow x) \vee \neg(T))$	$F \Rightarrow T$ is True
$(x \vee (z \Rightarrow \neg z)) \wedge ((z \Rightarrow x) \vee F)$	Negation
$(x \vee (z \Rightarrow \neg z)) \wedge (z \Rightarrow x)$	$a \vee F$ is a
$(x \vee (\neg z \vee \neg z)) \wedge (z \Rightarrow x)$	Implication Elimination
$(x \vee \neg z) \wedge (z \Rightarrow x)$	$a \vee a$ is a
$(x \vee \neg z) \wedge (\neg z \vee x)$	Implication Elimination
$x \vee \neg z$	$a \wedge a$ is a

6 First Order Logic

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

i. All the Kardashians love Kim: $\forall x$ Kardashian(x) $\wedge$ Love(x, Kim)

This is incorrect. This translates to everyone is a Kardashian and everyone loves Kim. Thus, if someone isn't a Kardashian, or if someone in the world doesn't love Kim, this expression becomes false, which is clearly not equivalent to the English sentence.

Typically, $\Rightarrow$ is the main connective with $\forall$.

ii. Some Kardashian dislikes sugar: $\exists x$ Kardashian(x) $\Rightarrow$ Dislikes(x, sugar).

This is incorrect. In the logical expression, if there exists a person that's not a Kardashian, this expression becomes $True$ (by implication rule). That's unrelated to our English expression, which btw is also clearly $False$ because everyone loves sugar except stupid Kourtney!!!)

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

(b) Forward Chaining: Using the following statements and generalized modus ponens, derive: RidesPrivateJet(North) (In English, this means derive that North West rides a private jet).

1. Kardashian(Kim)
2. Rich(Kim)
3. Parent(North, Kim)
4. $\forall x$ Kardashian(x) $\Rightarrow$ Celebrity(x)
5. $\forall x, y$ Kardashian(x) $\wedge$ Parent(y,x) $\Rightarrow$ Kardashian(y)
6. $\forall x, y$ Rich(x) $\wedge$ Parent(y,x) $\Rightarrow$ Rich(y)
7. $\forall x$ Rich(x) $\wedge$ Celebrity(x) $\Rightarrow$ RidesPrivateJet(x)

1, 3, 5 : Kardashian(North)
2, 3, 6 : Rich(North)
Kardashian(North), Rich(North), 7 : RidesPrivateJet(North)

(c) 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 Bs.

ii. ***R(w, w, z, f(z)), R(f(x), f(m(5)), x, y)***

{w/f(x), x/m(5), z/x, y/f(z)}

7 Satisfiability and Planning

In the recent pandemic, suppose we are tasked with making a plan to deliver N-95 masks around the U.S. We first try taking a SATplan (logical planning) approach, and formulate the following propositions:

- $at(loc, t)$: our cargo plane is at location loc at time t
- $fuel(x, t)$: the fuel level is at x at time t, $x \in [0, 5]$.
- $hasFuel(loc, t)$: location loc has fuel (to re-fuel the plane with) at time t
- $hasMasks(loc, t)$: location loc has masks at time t

Our starting state is $at(Pittsburgh, 0) \wedge fuel(5, 0)$.

1. Using the above predicates, formulate successor-state axioms for the actions $refuel(t)$, $deliver(t)$, and $fly(origin, destination, t)$. We can only refuel at a location that has fuel, and the fuel level jumps to 5 as a result of refueling. We can fly between any distinct locations, as long as the fuel level is less than 3; after flying, fuel level decreases by 1. We can deliver masks anywhere, and the result is that the location of the plane now has masks.

   refuel: $at(loc, t) \wedge fuel(x, t) \wedge refuel(t) \wedge hasFuel(loc) \iff fuel(5, t + 1)$
   deliver: $at(loc, t) \wedge deliver(t) \iff hasMasks(loc, t + 1)$
   fly: $at(origin, t) \wedge fuel(x, t) \wedge x > 0 \wedge fly(origin, destination, t) \wedge origin \neq destination \iff at(destination, t + 1) \wedge fuel(x - 1, t + 1)$

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 $a = at(loc, t)$, $b = deliver(t)$, $c = hasMasks(loc, t + 1)$. 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 P3!)

3. Suppose our goal is to deliver presents to NYC. Describe an algorithm which uses a SAT solver to find a plan for this goal.

   Again, think back to P3 - 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 $\wedge$ $hasMasks(NYC, i)$). 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 $hasMasks(Pittsburgh, 1)$ is feasible with our knowledge base $at(Pittsburgh, 0) \wedge fuel(0, 0) \wedge D$, where D is your *deliver* axiom instantiated with $loc = Pittsburgh$, $t = 0$ (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

$$at(NP,0) \wedge fuel(0,0) \wedge (at(NP,0) \wedge deliver(t) \Leftrightarrow hasMasks(NP,1)) \wedge hasMasks(NP,1)$$

is satisfiable. Using our CNF version of D from the question above, the full sentence we would pass into DPLL (using abbreviations) is as follows:

$$a(P,0) \quad f(0,0) \quad (\neg a(P,0) \vee \neg d(P,0) \vee hM(P,1)) \quad (\neg hM(P,1) \vee a(P,0)) \quad (\neg hM(P,1) \vee d(P,0)) \quad hM(P,1)$$

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

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

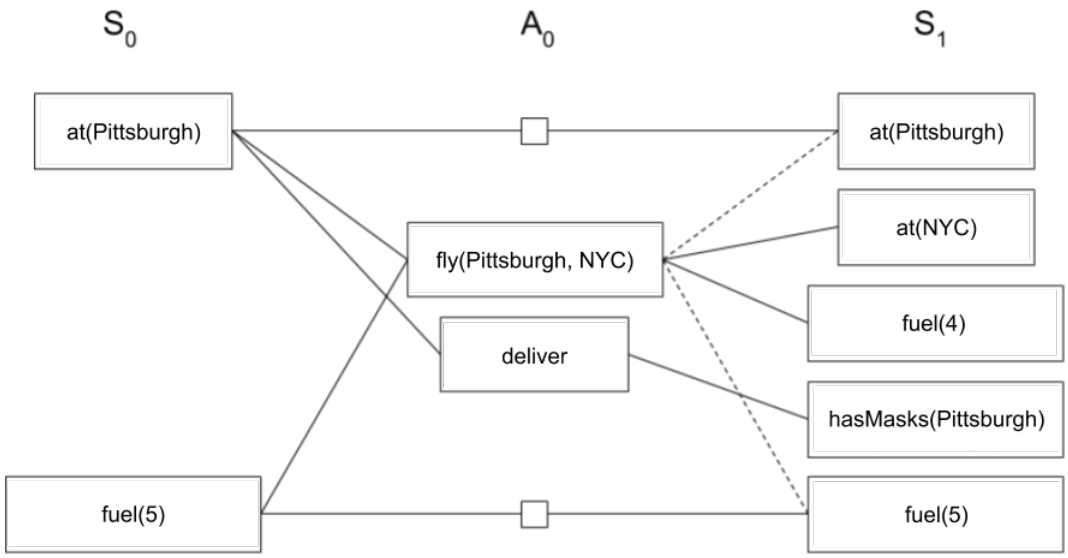
Rearranging the sentence, we get
$\neg(\neg A \vee \neg B)$
$\neg(A \Rightarrow \neg B)$. Based on the result from DPLL, we know this sentence is unsatisfiable, i.e. it is false in all models. That means its negation, $A \Rightarrow \neg B$, must be true in all models. By definition, this means $A \models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

6. Now we take 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):

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

	$refuel$	$fly(o,d)$	$deliver$
Precondition	$at(loc)$, $hasFuel(loc)$, $fuel(x)$	$at(o)$, $fuel(x)$, $x > 0$, $o \neq d$	$at(loc)$
Add	$fuel(5)$	$fuel(x-1)$, $at(d)$	$hasMasks(loc)$
Delete	$fuel(x)$	$fuel(x)$, $at(o)$	

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



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

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

9. In general, when does GraphPlan stop extending the 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.

10. 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.

8 Probability

1. *Independence*

   For each of the following four subparts, you are given three joint probability distribution tables. For each distribution, please identify if the given independence / conditional independence assumption is true or false.
   For your convenience, we have also provided some marginal and conditional probability distribution tables that could assist you in solving this problem.

   (a) X is independent from Y.

X	Y	$P(X,Y)$
0	0	0.240
1	0	0.160
0	1	0.360
1	1	0.240

X	$P(X)$
0	0.600
1	0.400

Y	$P(Y)$
0	0.400
1	0.600

   True; Two variables X, Y are independent if $P(X,Y) = P(X)P(Y)$ or equivalently, $P(X \mid Y) = \frac{P(X,Y)}{P(Y)} = P(X)$. The way to solve this problem is to see if $P(X,Y) = P(X)P(Y)$ or $P(X) = P(X \mid Y)$ for all combinations of X, Y.

   (b) X is independent from Y.

X	Y	$P(X,Y)$
0	0	0.540
1	0	0.360
0	1	0.060
1	1	0.040

X	$P(X)$
0	0.600
1	0.400

X	Y	$P(X \mid Y)$
0	0	0.600
1	0	0.400
0	1	0.600
1	1	0.400

   True; Two variables X, Y are independent if $P(X,Y) = P(X)P(Y)$ or equivalently, $P(X \mid Y) = \frac{P(X,Y)}{P(Y)} = P(X)$. The way to solve this problem is to see if $P(X,Y) = P(X)P(Y)$ or $P(X) = P(X \mid Y)$ for all combinations of X, Y.

(c) X is independent from Y given Z.

X	Y	Z	$P(X,Y,Z)$
0	0	0	0.280
1	0	0	0.070
0	1	0	0.210
1	1	0	0.140
0	0	1	0.060
1	0	1	0.060
0	1	1	0.030
1	1	1	0.150

X	Z	$P(X,Z)$
0	0	0.700
1	0	0.300
0	1	0.300
1	1	0.700

Y	Z	$P(Y,Z)$
0	0	0.500
1	0	0.500
0	1	0.400
1	1	0.600

X	Y	Z	$P(X,Y \mid Z)$
0	0	0	0.400
1	0	0	0.100
0	1	0	0.300
1	1	0	0.200
0	0	1	0.200
1	0	1	0.200
0	1	1	0.100
1	1	1	0.500

False; Two variables X, Y are conditionally independent given Z if $P(X, Y \mid Z) = P(X \mid Z)P(Y \mid Z)$ or equivalently, $P(X \mid Y, Z) = \frac{P(X,Y|Z)}{P(Y|Z)} = P(X \mid Z)$. You can solve these problems similarly to how you solved the last two problems.

(d) X is independent from Y given Z.

X	Y	Z	$P(X,Y,Z)$
0	0	0	0.140
1	0	0	0.140
0	1	0	0.060
1	1	0	0.060
0	0	1	0.048
1	0	1	0.192
0	1	1	0.072
1	1	1	0.288

X	Z	$P(X,Z)$
0	0	0.500
1	0	0.500
0	1	0.200
1	1	0.800

Y	Z	$P(Y,Z)$
0	0	0.700
1	0	0.300
0	1	0.400
1	1	0.600

X	Y	Z	$P(X,Y \mid Z)$
0	0	0	0.350
1	0	0	0.350
0	1	0	0.150
1	1	0	0.150
0	0	1	0.080
1	0	1	0.320
0	1	1	0.120
1	1	1	0.480

True; Two variables X, Y are conditionally independent given Z if $P(X, Y \mid Z) = P(X \mid Z)P(Y \mid Z)$ or equivalently, $P(X \mid Y, Z) = \frac{P(X,Y|Z)}{P(Y|Z)} = P(X \mid Z)$. You can solve these problems similarly to how you solved the last two problems.

2. *Chain Rule*

(a) When is $P(A, B \mid C)$ equivalent to the following?

(i) $\frac{P(C|A)P(A|B)P(B)}{P(C)}$

C is independent of B given A

(ii) $\frac{P(B,C|A)P(A)}{P(B,C)}$

Never, because this is $P(A \mid B, C)$

(iii) $P(A \mid B, C)P(B \mid C)$

Always

(iv) $\frac{P(A|C)P(B,C)}{P(C)}$

A is independent of B given C

(b) When is $P(A \mid B, C)$ equivalent to the following?

(i) $\frac{P(C|A)P(A|B)P(B)}{P(C)}$

Never

(ii) $\frac{P(B,C|A)P(A)}{P(B,C)}$

Always

(iii) $\frac{P(A|C)P(C|B)P(B)}{P(B,C)}$

A is independent of B given C

(iv) $\frac{P(C|A,B)P(B|A)P(A)}{P(B|C)P(C)}$

Always

3. *Probability Tables*

Let A be a random variable representing the choice of protein in the sandwich with three possible values, $\{mutton, bacon, egg\}$, let B be a random variable representing the choice of bread with two possible values, $\{toast, naan\}$, and let K be a random variable representing the presence of ketchup or not, $\{+k, k\}$.

How many values are in each of the probability tables and what do the entries sum to?
Write **'?'** if there is not enough information given.

Table	**num**	**sum**
$P(A, B)$		
$P(A, B, +k)$		
$P(A, B \mid +k)$		
$P(B \mid +k, A)$		

Table	**num**	**sum**
$P(A, B)$	6	1
$P(A, B, +k)$	6	?
$P(A, B \mid +k)$	6	1
$P(B \mid +k, A)$	6	3

9 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.

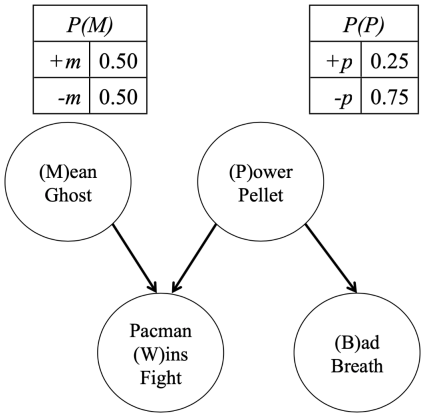
(a) What is $P(+w)$, the marginal probability that Pacman will win?

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

(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?

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

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

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

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

(e) 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?

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

(f) 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.

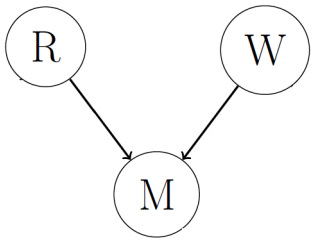
$$\begin{aligned} E(U_f| + m, +b) &= 10 \times P(+w| + m, +b) + (-5) \times P(-w| + m, +b) \\ &\approx 10 \times 0.352 - 5 \times 0.648 \\ &= 0.28 > -1 = U_r \end{aligned}$$

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

10 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 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.

11 HMMs and Particle Filtering

Consider the following hidden Markov model with a binary hidden state X. The transition probabilities and initial distribution are:

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 (i.e., after a dynamics update), 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 $\hat{P}(X_t)$?

X_t	$\hat{P}(X_t)$
0	
1	

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

(e) At some 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 updating our belief and resampling? When resampling, use this list of numbers as a source of randomness: [0.182, 0.703, 0.471, 0.859, 0.382] and fix the order of states to be $X_t = 0$, $X_t = 1$.

We can first find the joint probability $\hat{P}(X_t, e_t)$ (e_t being 1):

X_t	$\hat{P}(X_t, e_t)$
0	$\hat{P}(X_t = 0) * P(E_t = 1\|X_t = 0) = 2/5 * (1 - 0.6) = 0.16$
1	$\hat{P}(X_t = 1) * P(E_t = 1\|X_t = 1) = 3/5 * 0.6 = .36$

We then normalize to get the posterior $\hat{P}(X_t|e_t)$:

X_t	$\hat{P}(X_t \mid e_t)$
0	$.16/(.16 + .36) = .307$
1	$.36/(.16 + .36) = .693$

Finally, using our fixed order and the given random number list, we resample:

$$0.182 < 0.307 \rightarrow X_t = 0$$

$$0.703 >= 0.307 \rightarrow X_t = 1$$

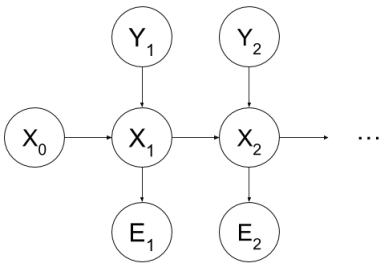
$$0.471 >= 0.307 \rightarrow X_t = 1$$

$$0.859 >= 0.307 \rightarrow X_t = 1$$

$$0.382 >= 0.307 \rightarrow X_t = 1,$$

getting the samples [0, 1, 1, 1, 1].
There are 4 particles with state value 1, and 1 particle with state value 0.

(f) Suppose we now have the following modified HMM structure, in which the hidden variables now have a parent variable Y_t, starting at $t = 1$:



Write expressions for answering the following queries. Make sure your expressions are solely in terms of the probability tables from the HMM, and that they are in the simplest possible form (hint: conditional independence!). You must explicitly write out any normalization constants.

(i) $P(X_1 \mid E_1)$

$$P(X_1 \mid E_1) = \frac{P(X_1, E_1)}{P(E_1)} = \frac{\sum_{y_1, x_0} P(X_1, E_1, x_0, y_1)}{\sum_{y_1, x_0, e_1} P(X_1, e_1, x_0, y_1)}$$
$$= \frac{\sum_{y_1} P(y_1) P(X_1 \mid x_0, y_1) P(E_1 \mid X_1) P(x_0)}{\sum_{y_1, x_1} P(y_1) P(x_1 \mid x_0, y_1) P(E_1 \mid x_1) P(x_0)}$$

(ii) $P(Y_1 \mid X_1, X_0)$

$P(Y_1 \mid X_1, X_0) = P(Y_1 \mid X_1)$ since $Y_1 \perp\!\!\!\perp X_0 \mid X_1$ (Markov property).

$$P(Y_1 \mid X_1) = \frac{P(X_1, Y_1)}{P(X_1)} = \frac{\sum_{x_0} P(X_1, Y_1, x_0)}{\sum_{x_0, y_1} P(X_1, y_1, x_0)} = \frac{\sum_{x_0} P(Y_1) P(X_1 \mid x_0, Y_1) P(x_0)}{\sum_{x_0, y_1} P(y_1) P(X_1 \mid x_0, y_1) P(x_0)}$$

(iii) $P(Y_1 \mid E_1)$

$$P(Y_1 \mid E_1) = \frac{P(E_1, Y_1)}{P(E_1)} = \frac{\sum_{x_0, x_1} P(Y_1) P(E_1 \mid x_1) P(x_1 \mid x_0, Y_1) P(x_0)}{\sum_{x_0, x_1, y_1} P(y_1) P(E_1 \mid x_1) P(x_1 \mid x_0, y_1) P(x_0)}$$

(iv) $P(Y_2 \mid E_1, E_2)$

$$P(Y_2 \mid E_1, E_2) = \frac{P(Y_2, E_1, E_2)}{P(E_1, E_2)}$$
$$= \frac{\sum_{x_0, x_1, x_2, y_1} P(Y_2) P(y_1) P(x_2 \mid x_1, Y_2) P(x_1 \mid x_0, y_1) P(E_2 \mid x_2) P(E_1 \mid x_1) P(x_0)}{\sum_{x_0, x_1, x_2, y_1, y_2} P(y_2) P(y_1) P(x_2 \mid x_1, y_2) P(x_1 \mid x_0, y_1) P(E_2 \mid x_2) P(E_1 \mid x_1) P(x_0)}$$

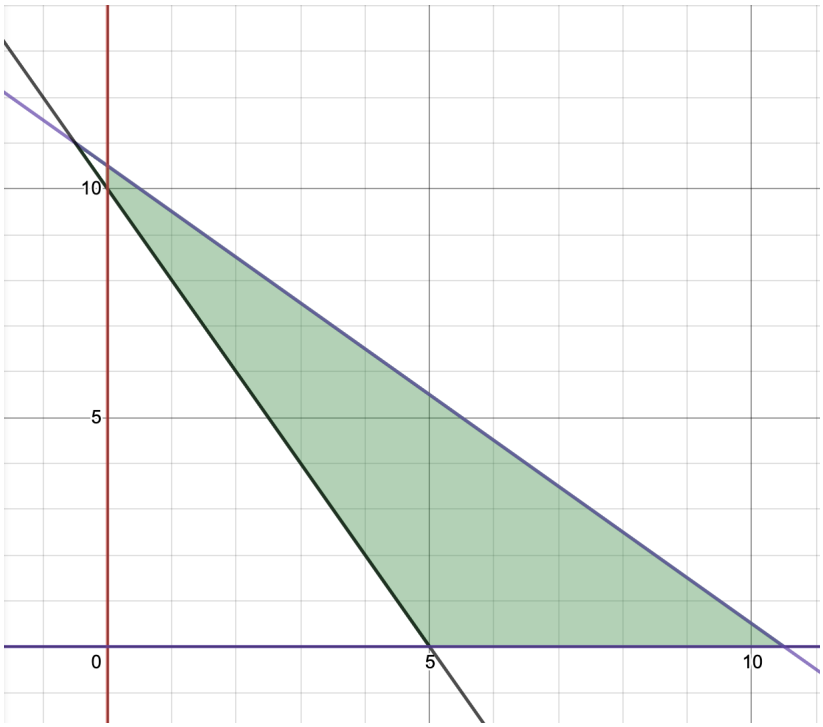
12 LP

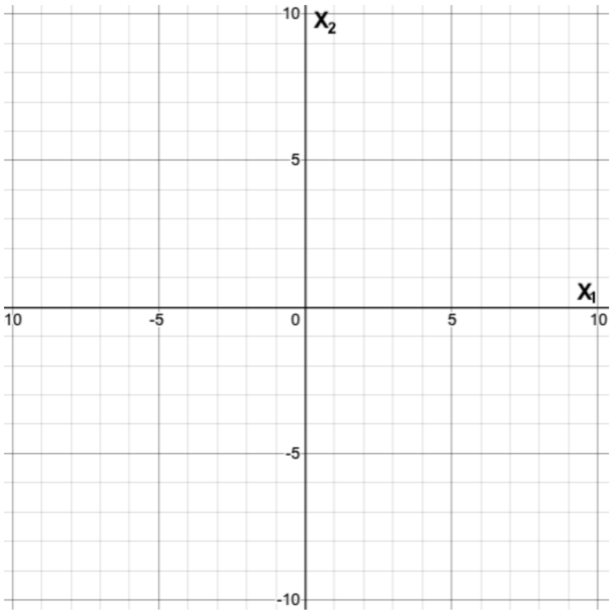
The Easter bunny is running late and is struggling to pack all the candy that he wants to hand out. This year, he is giving out massive amounts of chocolate and Peeps.

Let x_1 represent pounds of chocolate and let x_2 represent pounds of Peeps. We assume we can deliver a fraction of a pound of chocolate or Peeps. Unfortunately, the Easter bunny's basket can only fit 10.5 pounds of sweets. Furthermore, the Easter bunny wants to provide enough candy for 10 kids. Each pound of chocolate is enough for 2 kids while a pound of Peeps is only enough for 1 kid. However, the Easter bunny also wants to maximize the children's happiness. Chocolate brings 4 units of happiness while Peeps only bring 1.

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

$$\min_x c^T x \text{ st } Ax \leq b \text{ where A} = \begin{bmatrix} -1 & 0 \\ 0 & -1 \\ 1 & 1 \\ -2 & -1 \end{bmatrix}, \text{ b} = \begin{bmatrix} 0 \\ 0 \\ 10.5 \\ -10 \end{bmatrix}, c = \begin{bmatrix} -4 \\ -1 \end{bmatrix}$$





2. What would the optimal solution be?

   The optimal point would be (10.5, 0).

3. If the Easter bunny didn't know how much happiness chocolate and peppermints bring, what would be a cost vector that makes the optimal solution (0, 10.5)?

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

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

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

5. If the Easter bunny can only give out chocolate in the form of 1 pound bars that cannot be divided up, which constraints will you have to add in the first iteration of Branch and Bound?

   Left branch: $x \leq 10$
   Right branch: $x \geq 11$
   (order of branches does not matter)

13 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. MDPs - Micro-Blackjack: In micro-blackjack, you repeatedly draw a card (with replacement) that is equally likely to be a 2, 3, or 4. You can either Draw or Stop if the total score of the cards you have drawn is less than 6. Otherwise, you must Stop. When you Stop, your utility is equal to your total score (up to 5), or zero if you get a total of 6 or higher. When you Draw, you receive no utility. There is no discount ($\gamma = 1$).

   (a) What are the states and the actions for this MDP?

   The state is the current sum of your cards, plus a terminal state:

   $$0, 2, 3, 4, 5, Done$$

   The actions are $\{Draw, Stop\}$.

   (b) What is the transition function and the reward function for this MDP?

   The transition function is

   $$T(s, Stop, Done) = 1$$

   $$T(s, Draw, s') = \begin{cases} 1/3 \text{ if } s' - s \in \{2, 3, 4\} \\ 1/3 \text{ if } s = 2 \text{ and } s' = Done \\ 2/3 \text{ if } s = 3 \text{ and } s' = Done \\ 1 \text{ if } s \in \{4, 5\} \text{ and } s' = Done \\ 0 \text{ otherwise} \end{cases}$$

   The reward function is

   $$R(s, Stop, s') = s, s \leq 5$$

   $$R(s, a, s') = 0 \text{ otherwise}$$

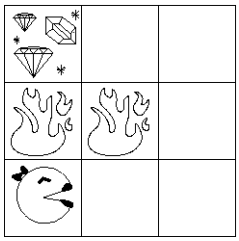
(c) Fill out the value iteration table below. We have filled out the first row for you. (Recall that we always initialize $V_0(s)$ to 0 for all states s.) Then, perform policy extraction and give the optimal policy for this MDP.

V	0	2	3	4	5	Done
V_0	0	0	0	0	0	0
V_1						
V_2						
V_3						
Policy Extraction						

V	0	2	3	4	5	Done
V_0	0	0	0	0	0	0
V_1	0	2	3	4	5	0
V_2	3	3	3	4	5	0
V_3	10/3	3	3	4	5	0
Policy Extraction	$10/3_{Draw}$	3_{Draw}	3_{Stop}	4_{Stop}	5_{Stop}	0_{Stop}

The optimal policy is **Draw if $s \leq 2$, Stop otherwise.** The optimal policy is given by taking the *argmax* instead of *max*, after the final iteration of value iteration.

3. 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? You may use the grid below to fill in the optimal action for each state.

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 $\pi(s, a)$.

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:

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

14 Game Theory

1. Warm Up

   (a) To formulate a game, what needs to be defined?

   We need to define players, action sets for each player, and utilities on each action outcome.

   (b) How can we define a strategy?

   A strategy for player i is a probability distribution over player i's actions.

   (c) What is the type of a solution concept of a game?

   A solution concept is a strategy profile. We define one by defining a strategy for each player. Hence, for a game with n players, it is an n-tuple of probability distributions over each player i's action.

   (d) What is a Nash Equilibrium?

   A Nash Equilibrium is a strategy profile where none of the participants benefit from unilaterally changing their decision.

   (e) Does a Nash Equilibrium always exist? Does a pure Nash always exist?

   If there is a finite number of players and a finite number of actions, a Nash Equilibrium always exists. But it is not necessarily a pure Nash (example RPS).

   (f) Give an example of a game with infinite actions such that no Nash Equilibrium exists.

   Consider a game with 1 player, and the player can choose any number x between 0 and 1 inclusive. If the player chooses x=0 or x=1, they get a payoff of zero. Otherwise, the player gets utility x. Clearly, there is no Nash Equilibrium. If player plays x, there is always move $\frac{x+1}{2}$ (the average of x and 1) that is higher than x and gives a higher payoff.

2. Equilibria

   With the Grinch now reformed, he has started helping Santa deliver presents. They both can either take a northern path or a southern path to deliver presents. Santa prefers to deliver on the northern path and the Grinch prefers the southern path. However, both are happier when they deliver together. If they deliver together, the one who prefers that location gets a payoff of 9, and the other gets a payoff of 5. If they both deliver to their preferred place, Santa gets a payoff of 3 and the Grinch gets a payoff of 4. If they both deliver to their unpreferred place, they both get a payoff of 1.

   (a) Finish formulating the game by filling in the values and actions for the two players.

		The Grinch	
SANTA			

		The Grinch	
		North	South
SANTA	North	(9,5)	(3,4)
	South	(1,1)	(5,9)

(b) Identify the pure strategy Nash Equilibria in this game.

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

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

Santa's strategy is $(\frac{8}{9}, \frac{1}{9})$ and the Grinch's strategy is $(\frac{1}{5}, \frac{4}{5})$.

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: $9(q) + 3(1-q) = 6q + 3$
South: $1q + 5(1-q) = -4q + 5$

If Santa decides to randomize, that means that he is indifferent between the two actions so the utilities must be equal. $6q + 3 = -4q + 5$ gives us $q = \frac{1}{5}$.
Utilities for the Grinch:

North: $5p + 1(1-p) = 4p + 1$
South: $4p + 9(1-p) = -5p + 9$

The Grinch is indifferent between the two options so the utilities of the two must be equal. Thus, $4p + 1 = -5p + 9$ so we get $p = \frac{8}{9}$.

(d) With the mixed strategy Nash Equilibrium, what is the probability each action outcome actually gets played.

(North, North) is played with probability $p * q = 8/45$.
(North, South) is played with probability $(1-p) * q = 1/45$.
(South, North) is played with probability $p * (1-q) = 32/45$.
(South, South) is played with probability $(1-p) * (1-q) = 4/45$.

(e) Use the probabilities found in the last part to find the expected utility for each Santa and the Grinch in the mixed Nash. Who gets a higher utility?

$U_S = \frac{8*9+1*1+32*3+4*5}{45} = \frac{189}{45}$. $U_G = \frac{8*5+1*1+32*4+4*9}{45} = \frac{205}{45}$

(f) Now let's consider the important fact that Santa is still the boss. And so the game here does not reflect a one-shot game, as before, but a Stackelberg game with commitment and a leader, Santa. With Santa as a leader, find the Stackelberg Equilibrium. What are the utilities for each player with this dynamic, and how does it compare to the mixed nash utility for each player?

Given Santa chooses north, the Grinch will choose north (because the payoff of 5 for the Grinch is bigger than the payoff of 4 for the Grinch), and this yields a payoff for Santa of 9 because the Grinch chose north. Given Santa chooses south, the Grinch will choose south (because the payoff of 9 for the Grinch is bigger than the payoff of 1 for the Grinch), and this yields a payoff for Santa of 5 because the Grinch chose south.
Because Santa has a higher payoff choosing north, the Stackelberg equilibrium is (North, North). This causes Santa to have a higher payoff of 9, and the Grinch to have a lower payoff of 5.
As an exercise, try formulating the game as a tree, like in adversarial search. Does the analysis of this tree come to be the same?

(g) The Grinch is a little annoyed about the fact that Santa is seeming dictatorial in the previous part. To combat this, he tells Santa that he will always choose south no matter what Santa does, so Santa should choose south. Still, Santa has the first official choice. Why does the Grinch's statement not matter?

Santa still has the first official choice, and given that Santa chooses North, it would be irrational for the Grinch to choose south instead. In this sense, the Grinch's statement has no weight due to the nature of the game.