1 Integer and Linear Programming

1. What is the relationship between branch & bound, and pruning?

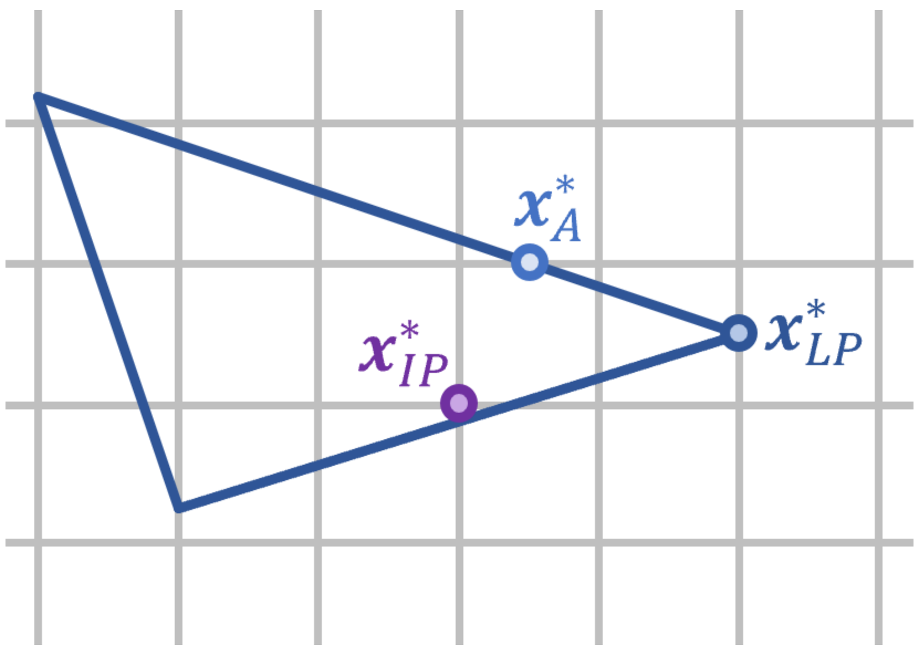
   In branch & bound, we stop recurring in a branch when the LP returns a worse objective than the best feasible IP objective we have seen before. This is analagous to pruning in minimax; we know that we can do better by taking another branch, so there's no point exploring this option further!

   Other reasons to stop incurring include finding an integer result from the LP, or discovering the LP to be infeasible.

2. T/F. As the magnitude of of **c** increases, the distance between the contour lines of the objective $c^T x$ increases as well. Explain your answer below.

   False. The distance between the contour lines actually decreases, because an increase in the magnitude of c implies that a lesser distance needs to be traversed in order to incur the same increase in cost.

3. Consider the integer programming problem illustrated by the figure below.



   The bold diagonal lines represent the boundaries of the feasible region and the gray vertical and horizontal lines represent the integer values for each axis.

   - $\boldsymbol{x}_{IP}^*$ is the point that minimizes the integer program.
   - $\boldsymbol{x}_{LP}^*$ is the point that minimizes the relaxed linear program. It happens to lie on a vertical gray line.
   - $\boldsymbol{x}_A^*$ is a specific point that lies on both the top constraint boundary as well as a horizontal gray line.

We will use the branch and bound algorithm to find the solution, $\boldsymbol{x}_{IP}^*$, to this IP. When executing branch and bound, we will always explore the x_i less than constraint subtree before the x_i greater than constraint subtree ("less" is to the left and down if that is not already clear).

(a) At what depth of the branch and bound tree will the IP solution be found? Note that we define depth of the root node to be zero.

The IP solution will be found at a depth of 3.

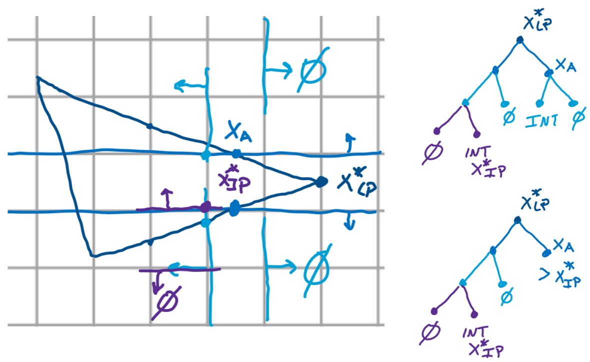
(b) If the minimum LP objective value at $\boldsymbol{x}_A$ is less than the minimum IP objective value at $\boldsymbol{x}^*_{IP}$, how many times must an LP solver be run to find the solution?

An LP solver must be run 9 times.

(c) If the minimum LP objective value at $\boldsymbol{x}_A$ is greater than the minimum IP objective value at $\boldsymbol{x}^*_{IP}$, how many times must an LP solver be run to find the solution?

An LP solver must be run 7 times.

The LP at the root has fractional value for x_2, so we first branch with horizontal lines (medium blue) below and above $\boldsymbol{x}^*_{LP}$. We first follow the less than branch, and that LP returns a location one grid point below $\boldsymbol{x}_A$, which has a fractional value for x_1. Continuing the depth first search, we branch with vertical lines (light blue) to the left and to the right of this point. Following the left most branch (we are at depth two at this point), the LP still doesn't return an integer solution. This time, it returns the light blue point just below $\boldsymbol{x}^*_{IP}$. One more pair of branches (purple horizontal lines) gives us an infeasible result and then finally $\boldsymbol{x}^*_{IP}$, which is at depth 3!



After we find $\boldsymbol{x}^*_{IP}$, we don't stop because we could potentially find a better integer solution. We continue depth first search, backing up to the root and eventually going down the right subtree to $\boldsymbol{x}_A$. If the objective at $\boldsymbol{x}_A$ is greater than the best integer point objective we have seen so far (the objective at $\boldsymbol{x}^*_{IP}$), then we can just stop (after 7 total LPs), because going deeper will only constrain the problem more and give worse objective values. Otherwise, we keep going. We only have to go one more level down, because the next set of branches (light blue vertical lines) results in a suboptimal integer point and an infeasible LP, finishing with 9 total LPs.

2 All About Logic

1. Propositional Logic

   (a) Vocab check: are you familiar with the following terms?

      i. Symbols

         Variables that can be T/F (capital letter)

      ii. Operators

         and, or, not, implies, equivalent

      iii. Sentences

         Symbols connected with operators, can be T/F

      iv. Equivalence

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

      v. Literals

         atomic sentence

      vi. Knowledge Base

         Sentences agents know to be true

      vii. Entailment

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

      viii. Query

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

      ix. Satisfiable

         At least one model makes the sentence true

      x. Valid

         True for all models

      xi. Clause - Definite, Horn clauses

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

      xii. Model Checking

         check if sentences are true in given model/checks entailment

      xiii. Theorem Proving

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

   (b) List the 5 operators in propositional logic. Are there any other operators other than these five?

      $\vee, \wedge, \Rightarrow, \Leftrightarrow, \neg$. These are the only 5 operators.

   (c) Which of the following have true/false values?

      ii,iii,iv,v

      i. Model
      ii. Sentence
      iii. Knowledge Base
      iv. Query

v. Literal

(d) Determine whether the sentences below are satisfiable, valid, or unsatisfiable.

i. $(\neg(y \vee \neg y) \vee x) \wedge (x \vee (z \iff \neg z))$

Satisfiable but not valid.

ii. $\neg(x \vee \neg(x \wedge (z \vee T))) \implies \neg(y \wedge (\neg y \vee (\top \implies \bot)))$

Valid

iii. $((\top \iff \neg(x \vee \neg x)) \vee z) \vee z) \wedge \neg(z \wedge ((z \wedge \neg z) \implies x))$

Unsatisfiable

2. First Order Logic

(a) Vocab check: are you familiar with the following terms?

i. Objects

Things that can be described with relations, and can be returned from functions

ii. Relations

Sentences that describes relationships between objects

iii. Functions

Describe an object using another object (no T/F values)

iv. Constants

Already-specified objects

v. Predicates

Sentences that need to be true before further inference (appears before $\Rightarrow$)

vi. Variables

Lower case letters that can hold different objects

vii. Connectives

Implies relationships between sentences (same as operators in Prop Logic)

viii. Equality

Given an interpretation, two terms are equal if they are mapped to the same object

ix. Quantifiers

$\exists, \forall$

x. Atomic Sentence

Sentences from only one relation

xi. Unification

Assignment of variables with known objects(constants)

(b) Consider the following two expressions in first order logic: $\text{Fatherof}(a, b)$, $\text{Fatherof}(a)$. Which one is a relation and which one is a function?

The former is a relation and the latter is a function

(c) Which of the following FOL sentences correctly expressed the idea of corresponding English sentences?

i. There was a student at CMU that never did 381 homework but passed the class.
$\exists x, \text{IsStudent}(x, \text{CMU}) \wedge \neg\text{DoesHW}(x, 381) \implies \text{Pass}(x, 381)$

F

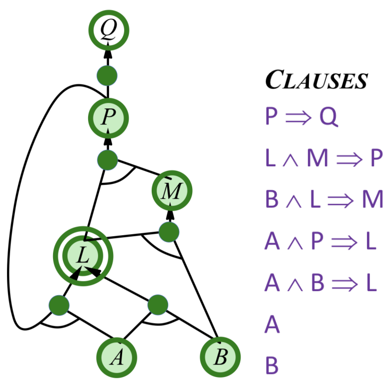
ii. If a student likes Pat, he'll pass the class.
$\forall x, \text{Student}(x) \land \text{Likes}(x, \text{Pat}) \implies \text{Pass}(x, 381)$
T

3. Forward Chaining

(a) What are the requirements for Knowledge Base in Forward Chaining?

KB can only contain definite clauses.

(b) A toy example from class for Forward Chaining:



(c) Is Forward chaining a sound algorithm? Is it a complete algorithm?

Yes, check the slides of Logical Agents for detailed proof.

(d) What does FOL-FC return?

A unification.

4. Planning

(a) Vocab check: are you familiar with the following terms?

i. Predicates
Conditions that the world must satify to perform an operation.

ii. Operators
Perform actions that change the state of the world

iii. Linear Planning
Solve one goal at a time.

iv. Non-linear Planning
Interleave goals to achieve plans

v. Inconsistency
The effects of two actions negate each other

vi. Interference
One action deletes/negates the precondition of the other

vii. Competing Needs

The precondition of two actions negates each other

viii. Complete

Can always find a solution whenever one exists

ix. Sound

All solutions found are legal.

x. Optimal

Shortest path to goal/Or the solution found is consistent with some other measure of path quality

(b) What are in the knowledge base of logic agents and classical planning problem, respectively?

Logical agents: Symbols and Implications
Planning: Predicates and operators

(c) What are the 3 components when defining an operator?

Predicates, add list, delete list

(d) Is linear planning complete? Optimal? What about non-linear planning?

Linear Planning: Not complete, one goal can immediately undo the other, and the plan stucks there. Not optimal: The plan naively attempts in order where there can be shortcuts among some goals.
Non-linear Planning: Complete and optimal. It performs search and would always reach the depth of the solution whenever there exists one. It returns the solution as soon as one is found so there can't be any shorter path (if all interleavings are searched).

5. **Facebook or Chipotle?**

Ethan gets easily bored in class, and often starts to get distracted. Two things usually pop up in his mind: whether or not he should go to Chipotle for lunch after class, and if he should check Facebook. Once he starts thinking about these options, he'll definitely choose to do at least one, because they're so tempting.

Ethan knows that going on Facebook is a mistake. He's so addicted that if he logs on, he'll lose track of time scrolling through his NewsFeed and probably fail the class. If he goes to Chipotle after class, he'll feel happy and satisfied. And, if he manages not to feel bored at all, he'll pass the class no matter what.

Let B be the symbol representing if Ethan gets bored, C representing going to Chipotle, F representing going on Facebook, and P representing passing the class. Below is the truth table for the four symbols.

B	C	F	P
$\bot$	$\top$	$\bot$	(1) $\top$
$\bot$	$\top$	$\top$	(2) $\top$
(3) $\bot$	$\bot$	$\top$	$\top$
$\top$	$\top$	(4) $\top$	$\bot$
$\top$	(5) $\top$	$\bot$	(6) $\top$
$\top$	$\bot$	$\top$	(7) $\bot$

(a) Fill in the holes in the chart based on the given information.

(b) Sean is another student in class. In his knowledge base, he admits that if he doesnt get bored, he will pass the class no matter what he does later on. However, if he gets bored, he will go directly to Chipotle and believes he can always pass the class after getting food. Represent Sean's knowledge base in propositional logic. (Select all that can represent his Knowledge Base)

i, iv

i. $\neg B \Rightarrow P; B \wedge C \Rightarrow P$
ii. $\neg B \Rightarrow P; \neg P \Rightarrow \neg C \vee F$
iii. $B \vee P; (\neg B \wedge \neg C \wedge F) \vee P$
iv. $B \vee P; (\neg B \vee \neg C) \vee P$

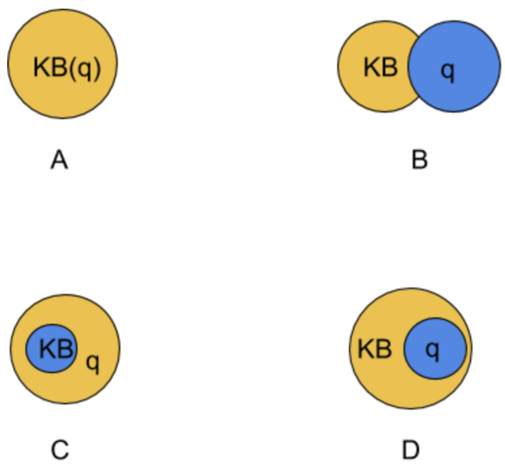
(c) For each row in the chart in (a), Fill in the corresponding value in the chart below for the "KB" column, where "KB" represents the propositional value where both sentences in Sean's KB are assigned to True.

B	C	F	P	KB	Query
$\bot$	$\top$	$\bot$		(1) $\top$	(7) $\top$
$\bot$	$\top$	$\top$		(2) $\top$	(8) $\top$
	$\bot$	$\top$	$\top$	(3) $\top$	(9) $\top$
$\top$	$\top$		$\bot$	(4) $\bot$	(10) $\top$
$\top$		$\bot$		(5) $\top$	(11) $\bot$
$\top$	$\bot$	$\top$		(6) $\top$	(12) $\top$

(d) A query q for this problem is the sentence $B \implies \neg P$. Fill in the truth assignment for the query in the chart above according to value assignments on each row for the "Query" column in chart (a).

(e) Which of the following graph would correctly represent relationship between truth values of the knowledge base KB and query q?

B



(f) In general, if statement A implying statement B is always True for all truth assignments, what can we infer about the entailment relationship between A and B? (multiple choice)

ii

i. B entails A
ii. A entails B
iii. Neither of A or B entails the other
iv. Need more information of values assigned to A and B

3 MDPs and RL

1. In a certain country there are N cities, all connected by roads in a circular fashion. A wandering poet is travelling around the country and staging shows in its different cities. He can choose to move from a city to each of the neighboring ones or he can stay in his current city i and perform, getting a reward r_i. If he chooses to travel, he will have a success probability of p_i. There is a $1 - p_i$ chance he will encounter a dragon along the way, which means he will have to turn back and wait the next day. If he is successful in travelling, he gains a reward of 0 for the day. And if he is unsuccessful at travelling, he can still perform a little bit when he gets back, giving him a reward of $r_i/2$.

   a) Let $r_i = 1$ and $p_i = 0.5$ for all i and let $\gamma = 0.5$. For $1 \leq i \leq N$, answer the following questions with real numbers:
   Hint: Recall that $\forall u \in (1, 0),\ \sum_{j=0}^{\infty} u^j = \frac{1}{1-u}$

      i) What is the value $V^{\text{stay}}(i)$ under the policy the wandering poet always chooses to stay?

$$V^{\text{stay}}(i) = r_i + \gamma V^{\text{stay}}(i)$$
$$V^{\text{stay}}(i) = 1 + 0.5V^{\text{stay}}(i)$$
$$V^{\text{stay}}(i) = 2$$

      ii) What is the value $V^{\text{next}}(i)$ under the policy the wandering poet always chooses to go to the next city?

$$V^{\text{next}}(1) = 0.5(\frac{1}{2} + 0.5V^{\text{next}}(1)) + 0.5(0.5V^{\text{next}}(2))$$
$$V^{\text{next}}(1) = V^{\text{next}}(2) = \cdots = \frac{1}{2}$$

   b) Let N be even, and let $p_i = 1$ for all i, and for all i, let the reward for cities be given as:

$$r_i = \begin{cases} a & i \text{ is even} \\ b & i \text{ is odd} \end{cases}$$

   where a and b are constants and $a > b > 0$.

      i) Suppose we start at an even-numbered city. What is the range of values of the discount factor γ for such that the optimal policy is to stay at the current city forever? Your answer may depend on a and b.

      For all possible values of γ, staying at an even city will be optimal.

      ii) Suppose we start at an odd-numbered city. What is the range of values of the discount factor γ such that the optimal policy is to stay at the current city forever? Your answer may depend on a and b.

      The poet should only move if losing that one extra day fo reward is worth it. So, either he can get the reward of staying for an infinite amount of time at an odd city, which is $b \cdot \frac{1}{1-\gamma}$ or he can move to city a and lose a whole day of as reward, which is $a \cdot \frac{1}{1-\gamma} - a$ He will only stay if the former is greater than the latter, which is only when $\gamma < \frac{a}{b}$

      iii) Suppose we start at an odd-numbered city and γ does not lie in the range you computed. Describe the optimal policy.

      The poet should move to an even city and stay there forever.

   c) Let N be even, $r_i \geq 0$, and the optimal value of being in the city 1 be positive, i.e., $V^*(1) > 0$. Define $V_k(i)$ to be the value of city i after the k^{th} time-step. Letting $V_0(i) = 0$ for all i, what is the largest k for which $V_k(1)$ could still be 0? Be wary of off-by-one errors.

Because $V^*(1) > 0$, there must be one $r_i > 0$ for some i. It then follows that $V_1(i) > 0, V_2(i-1), V_2(i+1) > 0$ and so on. The worst case is when the diametrically opposite to 1 is the only one having a nonzero ri. This implies that after $k = N/2$ steps, $V_{k+1}(1) > 0$ is guaranteed.

d) Let $N = 3$, and $[r_1, r_2, r_3] = [0, 2, 3]$ and $p_1 = p_2 = p_3 = 0.5$ and $\gamma = 0.5$. Compute:

i) $V^*(3)$

Notice that $Q^*(1, \text{stay}) = V^*(1)$. Clearly $\pi^*(1) = \text{go to 3}$.

$$V^*(1) = Q^*(1, \text{go to 3}) = 0.5\gamma V^*(1) + 0.5\gamma V^*(3)$$

$$V^*(3)Q^*(3, \text{stay}) = 3 + V^*(3)$$

Since $\gamma = 0.5$, we have that $V^*(3) = 6$.

ii) $V^*(1)$

$$V^*(1) = \frac{4}{3}\frac{1}{4}V^*(3) = 2$$

iii) $Q^*(1, \text{stay})$

$$Q^*(1, \text{stay}) = 1$$

2. Vocab check: are you familiar with the following terms?

(a) 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

(b) What does ϵ greedy mean, and in what context is it used?

ϵ greedy is an exploitation approach used in Q-learning. With probability ϵ, we choose a random next action. With a slightly larger probability $1 - \epsilon$ we choose our current policy's action.

(c) α, γ, ϵ, living rewards, ... how do these affect an agent's policy search?

conceptually: how do these change the behavior of things (and connecting this with greedy and other things)

i. ϵ is the exploration factor. This represents how likely it is for an agent to deviate from its current policy in active learning.

ii. 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.

iii. γ is the discount factor. It determines the how much the value of a state should take into account surrounding states. The higher the discount factor, the more one state would value distant states.

iv. Living rewards are rewards given each iteration an agent does not arrive at a terminal state. When living rewards are negative, a quick termination is preferred, and when it is positive, it is in the agents best interest to stay in the simulation as long as possible (not terminate).

(d) 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. Usually a combination of the two would work better for the agent.

(e) What is off-policy learning? Why, and how, is this used?

During learning, the action does not change until all the computations of a round are completed. Q-learning is an example for offline learning, where we don't know enough about the environment, so we can't give a good enough initial policy.