CS 188 Spring 2020 Introduction to Artificial Intelligence — Written HW 2

Due: Wednesday 03/11/2020 at 11:59pm (submit via Gradescope).

Policy: Can be solved in groups (acknowledge collaborators) but must be written up individually

Submission: Your submission should be a PDF that matches this template. Each page of the PDF should align with the corresponding page of the template (page 1 has name/collaborators, question 1 begins on page 2, etc.). **Do not reorder, split, combine, or add extra pages.** The intention is that you print out the template, write on the page in pen/pencil, and then scan or take pictures of the pages to make your submission. You may also fill out this template digitally (e.g. using a tablet.)

First name	Shiv
Last name	Shankar
SID	3032662765
Collaborators	None

For staff use only:

Q1.	MDP: Eating Chocolate	/40
Q2.	MDPs and RL	/30
Q3.	Probability: Flowers	/30
	Total	/100

Q1. [40 pts] MDP: Eating Chocolate

We have a chocolate bar of dimensions 1×8, which contains 8 squares. Most of these squares are delicious chocolate squares, but some of them are poison! Although the chocolate and poison squares are visually indistinguishable, someone has told us which ones are which. The layout for our chocolate bar is shown below with P indicating poison squares.

P				P		P	

Eating a chocolate square immediately gives a reward of 1 and eating a poison square immediately gives a reward of -2. **Starting from the right end of the bar**, there are 3 possible actions that we can take (at each step), and all of these actions cause non-deterministic transitions as follows:

- Action b_1: Try to bite 1 square, which will result in you actually eating 0, 1, or 2 squares with equal probability.
- Action b_2: Try to bite 2 squares, which will result in you actually eating 1, 2, or 3 squares with equal probability.
- Action *Stop*: Stop biting, which will end the game definitively and result in no reward.

(a) [10 pts] Formulate this problem as an MDP.

States: Decide (and explain) how you want to represent a "state." Using that representation, list out all of your possible states in this MDP.

Let a state be the number of squares remaining (counting from the left)

$S \in \{8, 7, 6, 5, 4, 3, 2, 1, 0\}$

Actions:

$a \in \{b_1, b_2, \text{Stop}\}$

Transitions:

$T(s, b_1, s-2) = 1/3$
$T(s, b_1, s-1) = 1/3$
$T(s, b_1, s) = 1/3$
} for $s \geq 2$

$T(s, b_1, s-1) = 2/3$
$T(s, b_1, s) = 1/3$
} for $s = 1$

$T(s, b_2, s-3) = 1/3$
$T(s, b_2, s-2) = 1/3$
$T(s, b_2, s-1) = 1/3$
} for $s \geq 3$

$T(s, b_2, s-2) = 2/3$
$T(s, b_2, s-1) = 1/3$
} for $s = 2$

$T(s, b_2, s-1) = 1$ for $s = 1$

$T(s, \text{stop}, s) = 1$ for all s

Rewards: Skip the explicit formulation of the rewards for this problem.

(b) Value Iteration

(i) [2 pts] Perform value iteration for 3 iterations, using $\gamma = 1$. What are the values for each state, after each of these iterations?

State	1	2	3	4	5	6	7	8
$V_0(s)$	0	0	0	0	0	0	0	0
$V_1(s)$	0	0	1	2	0	0	0	0
$V_2(s)$	0	0	$1\frac{1}{3}$	$2\frac{1}{3}$	0	1	0	0
$V_3(s)$	0	0	$1\frac{4}{9}$	$2\frac{4}{9}$	$\frac{2}{9}$	$\frac{11}{9}$	$\frac{1}{9}$	$\frac{1}{3}$

(ii) [1 pt] We consider value iteration to have converged by iteration k if $\forall s, V_{k+1}(s) = V_k(s)$. Did the values converge by iteration 2?

○ Yes ● No

(iii) [1 pt] Given that we know a $V(s)$ for all states, how do we extract a policy $\pi(s)$ from that value function? Your answer should be a one-line math expression which uses some or all of the following: $s, a, s', V, R, T, \gamma, \sum$, argmax, max.

$\pi(s) =$ $\arg\max_a \sum_{s'} T(s,a,s')\,[R(s,a,s') + \gamma V(s')]$

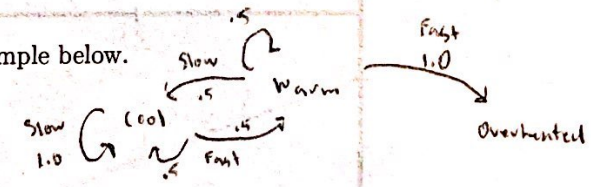
(iv) [1 pt] What is the resulting policy after extracting it from the values $V_2(s)$? If applicable, write all of the valid actions for a given state.

State	1	2	3	4	5	6	7	8
$\pi_3(s)$	Stay	Stay, b_1	b_1	b_2	b_1, b_2	b_2	Stop	b_2

(v) [5 pts] Conceptual: In general, we consider a policy to have converged if, for all states s, $pi(s)$ does not change with further calculation steps. Your friend Victor claims that **in any MDP setting, not limited to this game,** $\pi(s)$ can only be considered to be converged if the $V(s)$ that it came from was already converged. Is Victor right?

○ Victor is right, and I will provide a proof below.

● Victor is wrong, and I will provide a counter example below.



Proof/counter example:

The example in the textbook Ch.4 w/ cars:

	Cool	Warm
π_0	slow	slow
π_1	fast	slow
π_2	fast	slow

but

	Cool	Warm
V^{π_0}	2	2
V^{π_1}	1.5	0.5

so the policy converges before the values

For the rest of the problem, assume we have deterministic transitions. This means that taking the b_1 action bites exactly 1 square and taking the b_2 actions bites exactly 2 squares. Note that if you take b_1, then your reward for that step is the reward associated with that one square, and if you take b_2, then your reward for that step is the sum of rewards from the two squares.

(c) (i) [6 pts] Let's say that you took a sequence of 3 single bites, so you are now in state $s = 5$. At this point, you can either stop or continue to bite. For what range of discount values γ would you choose to stop versus continue to bite? Show your work and explain your answer.

The expected reward of biting must exceed the reward of stopping (0)
You should always take 2 bites when possible to reduce the discounting effect. We must bite twice to get a positive score and stop. So the first bite is worth $-2+1 = -1$ and the second bite is worth $\gamma(1+1) = \gamma 2$

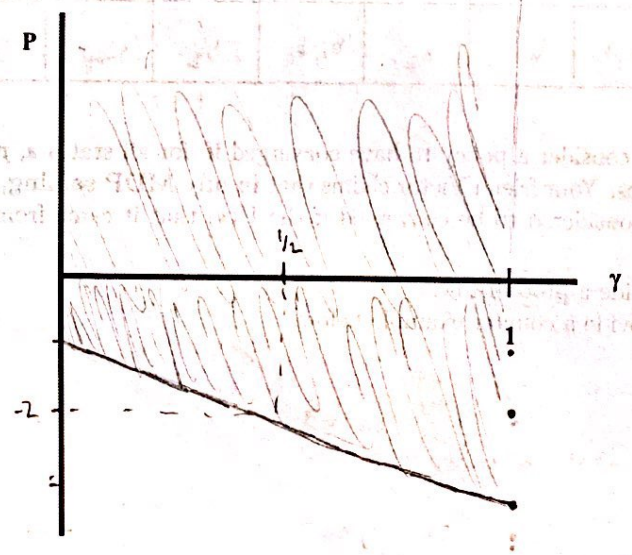
Therefore $-1 + 2\gamma \geq 0$

$\gamma \geq 1/2$

(ii) [4 pts] For this part, assume the same situation as the previous part, where you took 3 single bites already and you are now in state $s = 5$. Think about how the behavior of the optimal policy at this point changes, as we vary (a) the reward associated with eating a poison square and (b) the discount γ from 0 to 1. First, what's the mathematical condition that must hold for us to choose to continue to bite:

$(P+1) + \gamma(2) \geq 0$

Plot this condition in the plot below by shading in the areas where we will continue to bite. Label important points, including the point where $P = -2$ with the threshold γ value you found in part (c.i).



$P = -2\gamma - 1$

(d) **(i)** [2 pts] Define/create a policy below such that performing value iteration on that policy would give these values indicated here.

State	1	2	3	4	5	6	7	8
$V^{\pi}(s)$	0	-1	2	3	0	2	-1	0
$\pi(s)$	stop	b_2	b_2	b_1	stop	b_2	b_2	stop

(ii) [3 pts] Is the previous policy optimal? Perform one step of policy iteration to justify your answer. Please show your work.

State	1	2	3	4	5	6	7	8
$\pi_{i+1}(s)$	0	b_1	b_2, stop	b_1	b_1, b_2	stop, b_2	b_1	b_2

$$\pi_{i+1}(s) = \arg\max_a \sum_{s'} T(s,a,s')\left[R(s,a,s') + \gamma V^{\pi_i}(s')\right]$$

	stop	b_1	b_2
$s=1$	0	-2	-2
$s=2$	-1	1	-1
3	2	0	2
4	3	3	1
5	0	1	1
6	2	1	2
7	-1	0	-1
8	0	0	1

Was the policy in part (d.i) optimal: ○ Yes ● No

(iii) [5 pts]

We can build a graph from an MDP as follows. The vertices are the states, and there is an edge from state s to state s' labelled with an action a if $T(s, a, s')$ is nonzero. We can fully represent the MDP and assign a weight of $T(s, a, s')$ and a value of $R(s, a, s')$ to the corresponding edge from s to s' in the graph. Note that there may be more than one edge from s to s' if there is more than one action with a positive transition probability between s and s'. That is, there may be parallel edges in the graph.

Indeed, the MDP about the going fast and slow in a car in lecture is illustrated as a graph in the slides.

Notice that when viewed as a graph the state transitions of the above MDP forms a directed acyclic graph (DAG). It is useful here to explicitly add a terminal state named 0. The terminal state is reached from any state using the S action and every other action from any state leads to a lower numbered state or the terminal state. (To be sure, this would be true as well for the probabilistic version if action b_1 had zero probability of eating nothing.)

Sketch an algorithm to produce the optimal policy for $\gamma = 1$ in time linear in the number of states and edges in the associated graph in an MDP where the transitions form a DAG. You may assume there is a single terminal state with no outgoing transitions. (Hint: the value of the terminal state is 0.) Can you also produce the optimal policy for an arbitrary $\gamma < 1$?

We can linearize the DAG on the MDP graph of the chocolate problem. Then we can use a dynamic programming solution using the following recurrence relation:

$prev[0] = 0$

$V_0 = 0$

for $i = 1$ to 8

$$V_i = \max_{0 \le j < i} \{ V_j + e(j,i) \}$$

$$prev[i] = \arg\max_{j} \{ V_j + e(j,i) \}$$

Note that we track the optimal action by storing the predecessor of each node in the prev array. So to find the optimal actions we can just follow the path of states starting from $prev[8]$. We could also just store the edges.

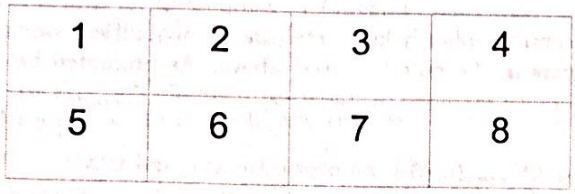
To produce the policy for $\gamma < 1$

change the relation to:

$$V_i = \max_{0 \le j < i} \{ \gamma \cdot V_i + e(j,i) \}$$

Q2. [30 pts] MDPs and RL

The agent is in a 2×4 gridworld as shown in the figure. We start from square 1 and finish in square 8. When square 8 is reached, we receive a reward of $+10$ at the game end. For anything else, we receive a constant reward of -1 (you can think of this as a time penalty).



The actions in this MDP include: up, down, left and right. The agent cannot take actions that take them off the board. In the table below, we provide initial non-zero estimates of Q values (Q values for invalid actions are left as blanks):

Table 1

	action=up	action=down	action=left	action=right
state=1		Q(1, down)=4		Q(1, right)=3
state=2		Q(2, down)=6	Q(2, left)=4	Q(2, right)=5
state=3		Q(3, down)=8	Q(3, left)=5	Q(3, right)=7
state=4		Q(4, down)=9	Q(3, left)=6	
state=5	Q(5, up)=5			Q(5, right)=6
state=6	Q(6, up)=4		Q(6, left)=5	Q(6, right)=7
state=7	Q(7, up)=6		Q(7, left)=6	Q(7, right)=8

(a) Your friend Adam guesses that the actions in this MDP are fully deterministic (e.g. taking down from 2 will land you in 6 with probability 1 and everywhere else with probability 0). Since we have full knowledge of T and R, we can thus use the Bellman equation to improve (i.e., further update) the initial Q estimates.

Adam tells you to use the following update rule for Q values, where he assumes that your policy is greedy and thus does $\max_a Q(s,a)$. The update rule he prescribes is as follows:

$$Q_{k+1}(s,a) = \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma \max_{a'} Q_k(s',a')]$$

(i) [1 pt] Perform one update of $Q(3, \text{left})$ using the equation above, where $\gamma = 0.9$. You may break ties in any way.

$Q_{k+1}(3,\text{left}) = \sum_{s'} T(3,\text{left},2)\,[R(3,\text{left},2) + \gamma \max_{a'} Q_k(2,a')]$

$= 1 \cdot [-1 + 0.9(6)] = 4.4$

(ii) [3 pts] For the Q update rule prescribed above, how does $Q_{k+1}(s,a)$ depend on $Q_k(s,a)$? How is this different from the normal Q learning update that we saw in lecture, which is $Q_{k+1}(s,a) = (1-\alpha)Q_k(s,a) + \alpha * \text{sample}$?

In the rule above, it uses model based learning since it knows the transition probability so $Q_{k+1}(s,a) \propto T(s,a,s')\,\gamma \max_{a'} Q_k(s',a')$ so it doesn't directly depend on $Q_k(s,a)$ but $Q(s',a')$

In normal Q learning Q_{k+1} is a running average of $Q_k(s,a)$ and the sample.

(b) After observing the agent for a while, Adam realized that his assumption of T being deterministic is wrong in one specific way: **when the agent tries to legally move down, it occasionally ends up moving left instead** (except from grid 1 where moving left results in out-of-bound). All other movements are still deterministic.

Suppose we have run the Q updates outlined in the equation above until convergence, to get $Q^*_{wrong}(s,a)$ under the original assumption of the wrong (deterministic) T. Suppose $Q^*_{correct}(s,a)$ denotes the Q values under the new correct T. Note that you don't explicitly know the exact probabilities associated with this new T, but you know that it qualitatively differs in the way described above. As prompted below, list the set of (s,a) pairs where $Q^*_{wrong}(s,a)$ is either an over-estimate or under-estimate of $Q^*_{correct}(s,a)$.

(i) [3 pts] List of (s,a) where $Q^*_{wrong}(s,a)$ is an over-estimate (and why):

(1,R), (2,D), (2,R), (3,L), (3,D), (3,R), (4,L), (4,D), (6,U), (7,U)

They are over estimates since they assume that taking the action down, will go down w/ probability 1 and bring them closer to state 8. Going left in all cases would increase the number of steps (which have -1 reward) so if they were included in Q_{wrong} then it would be lower hence they are overestimates. So anything that deals w/ the top row is an overestimate Note that for the bottom row the optimal policy is just to go left so it is fine.

(ii) [3 pts] List of (s,a) where $Q^*_{wrong}(s,a)$ is an under-estimate (and why):

None, In the best case the path from state s to state 8 requires no down actions, meaning that the Q^*_{wrong} value was a correct estimate and did not underestimate the q value. This means Q^*_{wrong} can only overestimate, never underestimate.

(c) [2 pts] Suppose that we have a mysterious oracle that can give us either all the correct Q-values $Q(s,a)$ or all the correct state values $V(s)$. Which one do you prefer to be given if you want to use it to find a greedy policy, and why?

For a greedy policy we want the agent to have the highest immediate expected reward. So we would want $V(s)$

(d) [2 pts] Suppose that you perform actions in this grid and observe the following episode: 3, right, 4, down, 8 (terminal).

With learning rate $\alpha = 0.2$, discount $\gamma = 0.9$, perform an update of $Q(3, right)$ and $Q(4, down)$. Note that here, we update Q values based on the sampled actions as in TD learning, rather than the greedy actions.

$\text{sample}_1 = R(3,R,4) + \gamma \max_{a'} Q(4,a') = -1 + 0.9(9) = 7.1$

$Q(3,\text{right}) = (.8)(7) + (.2)(7.1) = 5.6 + 1.42 = 7.02$

$\text{Sample}_2 = R(4,\text{down},9) + \gamma \max_{a'}(4,a') = 10 + 0.9(0) = 10$

$R(4,\text{down}) = 0.2(10) + 0.4(9) = 9.2$

(e) [2 pts] One way to encourage an agent to perform more exploration in the world is known as the "ϵ-greedy" algorithm. For any given policy $\pi(s)$, this algorithm says to take the original action $a = \pi(s)$ with probability $(1-\epsilon)$, and to take a random action (drawn from a uniform distribution over all legal actions) with probability ϵ. If ϵ can be tuned, would you assign it to be a high or low value at the beginning of training? What about at the end of the training? Please answer both questions and justify your choices.

High at the beggining so it explores more states in order to obtain accurate Q values.

Low at the end so it doesn't take random actions over a well developed policy (which has already visited all the states).

(f) Instead of using the "ϵ-greedy" algorithm, we will now do some interesting exploration with softmax. We first introduce a new type of policy: A stochastic policy $\pi(a|s)$ represents the probability of action a being prescribed, conditioned on the current state. In other words, the policy is a now a distribution over possible actions, rather than a function that outputs a deterministic action.

Let's define a new policy as follows:

$$\pi(a|s) = \frac{e^{Q(s,a)}}{\sum_{a'} e^{Q(s,a')}}$$

(i) [1 pt] Suppose we are at square 3 in the grid and we want to use the originally provided Q values from the table. What is the probability that this policy will tell us to go right? Note that the sum over actions prescribed above refers to a sum over legal actions.

$$\pi(R|3) = \frac{e^7}{e^8 + e^5 + e^7} = 0.259$$

(ii) [3 pts] What is the advantage of this exploration strategy, compared with "ϵ-greedy"? How are they qualitatively different?

ϵ greedy will randomly explore a state, this strategy explores w/ probability weighted by the respective Q value. So after many iterations, the algorithm will be more likely to take actions that it knows has higher Q values and not randomly explore bad states like ϵ greedy. 9

(g) [10 pts] Your friend Cody argues that we could still explicitly calculate Q updates (like Adam's approach in part (a)) even if we don't know the true underlying transition function $T(s, a, s')$, because he believes that our T can be roughly approximated from samples.

(i) [5 pts] Consider that you are given a moderate amount of (maybe 100,000) of transitions, in the form of (s_{start}, a, s_{end}). In a few sentences, describe **a sequence of steps** that you can take to compute $T_{approx}(s, a, s')$, which is an approximation of the true underlying (unknown) $T(s, a, s')$.

We can use model based learning to approximate $T(s,a,s')$. We simply count the number of times each tuple (s,a,s') occurs and divide it by the number of times the q state (s,a) occurs.

Since we have sufficiently many transition tuples our T_{approx} will converge toward the true T.

(ii) [5 pts] Now, consider a case where Cody does the following: He (1) uses samples to compute $T_{approx}(s, a, s')$ in the way that you described in the previous part, and then (2) uses that $T_{approx}(s, a, s')$ to perform Q updates with Adam's equations. What could be one potential problem with this approach?
Hint: If your $T_{approx}(s, a, s')$ is not exactly correct (i.e., you didn't get infinite data samples, so it's probably not exactly correct), what could go wrong?

Suppose the T_{approx} values are not exactly correct.

Then the Q updates: $Q_{hat}(s,a) = \sum_{s'} T_{approx}(s,a,s')[R(s,a,s') + \gamma \max_{a'} Q_k(s',a')]$

could be wrong. Re-updating the Q values would compound the errors of T_{approx} and the Q values could be suboptimal as the state transitions are not properly calculated

Q3. [30 pts] Probability: Flowers

Your friend Ethan wants to find the biggest flower in the field. In the field, there are n flowers where the value of n is unknown. The size of all the flowers can be ranked unambiguously if all are seen. Ethan will walk across this field, with each order of flowers being equally likely. Ethan decides not to walk one meter back because he is too tired. He also only gives him one chance to pick the flower. That means whenever he picks a flower, he is done with the whole sequence of flowers. He wants to have the highest probability of selecting the actual biggest flower.

(a) **(i)** [3 pts] Ethan thinks he can do the following: He will check out $r-1$ flowers without touching them so that he can have a good calibration. M is the biggest flower in the $r-1$ flowers. He then selects the first subsequent flowers that is better than M. For an arbitrary cutoff r, can you show the **probability that the biggest flower is actually selected.**

The probability

$$P(r) = \sum_{i=1}^{n} P(\text{flower } i \text{ is selected and it is the biggest}) \tag{1}$$

$$= \sum_{i=1}^{n} P(\text{flower } i \text{ is selected}|i \text{ is the best}) P(\text{flower } i \text{ is the best}) \tag{2}$$

$$= (\sum_{i=1}^{r-1} P(\text{flower } i \text{ is selected}|i \text{ is the best}) + \sum_{i=r}^{n} P(\text{flower } i \text{ is selected}|i \text{ is the best})) P(\text{flower } i \text{ is the best}) \tag{3}$$

Following this, can you derive the probability formulation for the probability that the biggest flower is actually selected.

Hint: Ethan will stop at any flower F larger than the calibration one. What's the probability of this one (F) to be the actual biggest one? Maybe all the flowers before F is smaller than M.

A_i: current biggest flower is chosen

B_i: event that A_i is the i^{th} flower

1 = biggest flower

This means flowers r to $i-1$ are worse than best flower in first $r-1$ flowers

So the 2nd biggest flower from the first i flowers is in the $r-1$ flowers

Note if F is in first $r-1$ flowers, we won't pick it

$$P[r|B_i] = \frac{r-1}{i-1}$$

$$P[r] = \sum_{i=r}^{n} P[B_i] \cdot P[A|B_i] = \frac{r-1}{n} \sum_{i=r}^{n} \frac{1}{i-1}$$

(ii) [4 pts] Now compute the probability when n goes to ∞.

Hint: you might need to do a bit of calculus here. For example, when $n \to \infty$, $\frac{1}{n}$ can be dt in an integral. Here t is a new variable we introduce.

$$\lim_{n\to\infty} P[r] = \frac{r-1}{n} \sum_{i=r}^{n} \frac{1}{i-1} = \frac{r-1}{n} \sum_{r}^{n} \frac{n}{i-1} \frac{1}{n} = \frac{r-1}{n} \left[\int_{1}^{n} \frac{1}{t}\,dt - \int_{1}^{r-1} \frac{1}{t}\,dt\right] = \frac{r-1}{n}\left[\ln(t)\Big|_{1}^{n} - \ln(t)\Big|_{1}^{r-1}\right]$$

let $\frac{1}{n} = dt$ and $\frac{1}{t} = \frac{n}{i-1}$

$$= \frac{r-1}{n}\left[\ln(n) - \ln(r-1)\right]$$

$$= \frac{r-1}{n} - \ln\left(\frac{n}{r-1}\right)$$

(iii) [3 pts] Take the derivative and tell Ethan when should he stop calibration. (what is r?) Please show all steps of your work

let $x = r-1/n$

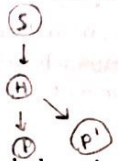
$$\frac{d}{dx} \lim_{n\to\infty} P[r] = \frac{-d}{dx}\frac{(r-1)}{n} \ln\left(\frac{n}{r-1}\right) = \frac{-d}{dx} x\ln(x) = -\left(\ln(x) + \frac{1}{x}\cdot x\right)$$

$$= -\ln(x) - 1 = 0$$

$$\Rightarrow \ln\left(\frac{r-1}{n}\right) = -1$$

$$r = ne^{-1} + 1$$

(b) **(i)** [1 pt] If it is a sunny day (+s), then Ethan will be happy (+h) with a high probability. If Ethan is happy (+h), he will execute his plan to pick a flower with a high probability (+p). Otherwise, he will execute his plan with a low probability (+p'). Sunny day (S), Ethan's happiness (H), and pick a flower (P) are three variables. Draw out the bayes net with those three variables.



(ii) [2 pts] If condition on Ethan's happiness, is the weather and flower picking independent? Explain your answer intuitively **without referring to the active/inactive triples.**

Yes; if we know Ethans happiness then we have all the information to know the flower picking probability, this is not affected by the weather.

(iii) [7 pts] Now we find more concrete information about this activity. For weather, it can be sunny and rainy. For happiness, it can be happy and unhappy. For flower pick, it can be pick or not pick. $Pr(sunny) = 0.4$, $Pr(happy|sunny) = 0.8$, $Pr(happy|rainy) = 0.5$, $Pr(pick|happy) = 0.8$, $Pr(pick|unhappy) = 0.2$. Can you compute $Pr(sunny|pick)$? Please show all steps of your work.

$$P[\text{sun}|\text{pick}] = \frac{P[\text{pick}|\text{sun}]P[\text{sun}]}{P[\text{pick}]} = \frac{(0.68)(0.4)}{.572} = 0.476$$

$$P[\text{pick}|\text{sun}] = P[\text{happy}|\text{sunny}] \cdot P[\text{pick}|\text{happy}] + P[\text{unhappy}|\text{sunny}] \cdot P[\text{pick}|\text{unhappy}]$$
$$= (0.8)(0.4) + (1-0.8)(0.2) = 0.64 + 0.04 = 0.68$$

$$P[\text{pick}] = P[\text{pick}|\text{happy}]P[\text{happy}] + P[\text{pick}|\text{unhappy}](1 - P[\text{happy}])$$
$$(0.8)(.62) + (0.2)(1-0.62) = .496 + .076 = .572$$

$$P[\text{happy}] = P[\text{happy}|\text{sunny}]P[\text{sunny}] + P[\text{happy}|\text{rainy}]P[\text{rainy}]$$
$$= (0.8)(0.4) + (0.5)(1-0.4) = .32 + .30 = .62$$

(c) Consider a modified setting where Ethan and his friends Fiona are **competing to pick a bigger flower.** Ethan and Fiona are walking in two separate fields with flowers of the same set of sizes $[s_1, s_2, s_3, s_4, ..., s_n]$, which in turn have ranks $[1, 2, 3, 4, ..., n]$ (where higher ranks means bigger sizes), but arranged in (possibly) different order. At each discrete time-step t, both Ethan and Fiona will see one more flower in their respective field.

Ethan and Fiona cannot observe the other person's field directly (and thus cannot see what the other person has seen so far).

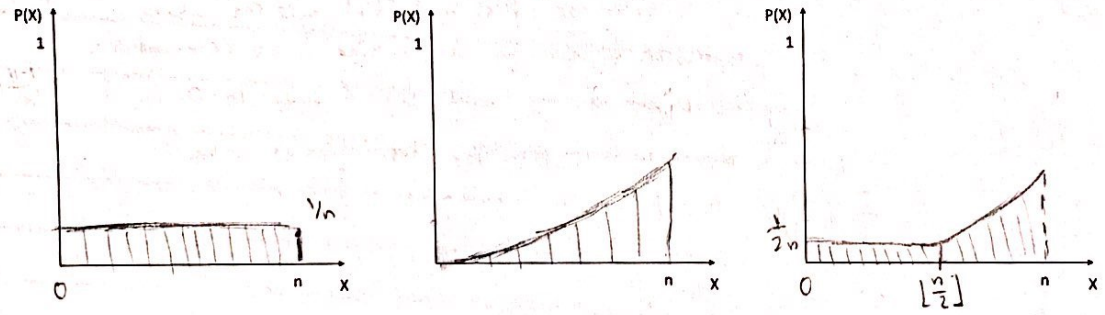
If Fiona knows that Ethan is committed to the strategy described in part (a.i),and the number of flowers Ethan is checking out is exactly r_e. What is Fiona's optimal r_f to get a bigger flower than Ethan?

(i) [1 pt] Consider the rank of the flower Ethan ended up picking as random variable X, and the rank of the flower Fiona ended up picking as random variable Y. What is the condition for Fiona strictly beating Ethan in the competition, in term of X and Y

$Y > X$

(ii) [3 pts] How does the probability distribution for $P(X = x)$ for $x = 1, 2, 3, ..., n$ depend on r_e? Please provide a sketch for $P(X)$ parametrized by $r_e = n - 1$, $r_e = 1$, $r_e = \lfloor \frac{n}{2} \rfloor$. Note that the function will be discrete.

You do not have to derive the expression for $P(X = x)$ in terms of r_e. Reasoning the shape should be adequate



$r_e = n - 1$ $\quad$ $r_e = 1$ $\quad$ $r_e = \lfloor \frac{n}{2} \rfloor$

* Ethan will always pick the last flower if he can't find anything

Justification for the shape of $r_e = n - 1$:

the n rank flower can either be in the first r_e, in which case he picks a flower w/ rank $< n$ or it can be the last flower. This occurs w/ probability $\frac{(n-1)!}{n!} = \frac{1}{n}$

Justification for the shape of $r_e = 1$:

Having only seen 1 flower gives a equal probability of the first flower being any flower. If the first flower is low rank, then a higher rank flowers can be chosen, but if the first flower is high rank, the lower rank flowers won't be chosen.

Justification for the shape of $r_e = \lfloor \frac{n}{2} \rfloor$:

At the very least, the max rank flower in r_e is rank $\lfloor n/2 \rfloor$, so any flower of rank lower will never be chosen. But if the greatest rank flower is in the first r_e, the last flower will be picked. This happens $\frac{1}{2}$ the time

(iii) [2 pts] Does r_f parametize $P(Y)$ the same way r_e paramatize $P(X)$?

○ Yes ● No

Explanation:

Since Fiona knows Ethans r_e strategy. r_f will parameterize $P(Y)$ differently

(iv) [2 pts] What is the probability of $P(Y > X)$? Your answer should include two summations, and you are not required to simplify $P(X)$, $P(Y)$.

$P(Y > X) =$

$$\sum_{x \in X} P[Y > x \mid X = x]\, P[X = x]$$

$$= \sum_{y=x+1}^{n} P[Y = y] \cdot \sum_{x \in X} P[X = x]$$

* Note $P(x) = P[X = x]$

$P(y) = P[Y = y]$

(v) [2 pts] Suppose you have the full expression for $P(X)$ and $P(Y)$ in terms of r_e, r_f respectively. How would you find the optimal r_f that would maximize the chance for Fiona to beat Ethan in this competition? Explanation:

- Since we are trying to maximize $P[Y > X]$, we can substitute $P(X)$ and $P(Y)$ into the above expression. Then we can take the derivative w/ respect to r_f and set it equal to 0. We could then solve for the optimal value of r_f.