CS 495 & 540 Fall 2018 Artificial Intelligence — Midterm

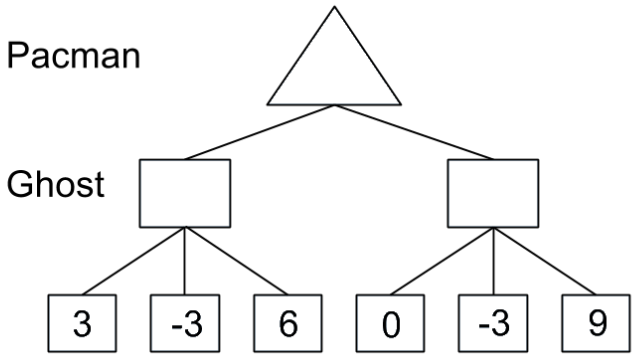
- You have approximately 110 minutes. Use your time wisely; approximately 1 minute per 1 point.
- The exam is closed book, closed notes. There should be sufficient space in the scratch paper area for your computation. Please do not use any additional personal sheets for computation purpose.
- Mark your answers ON THE EXAM ITSELF. If you are not sure of your answer you may wish to provide a *brief* explanation. In general, each sub-problem is worth 5 points unless otherwise specified.
- There are 10 problems worth 112 points for this exam. If your score is more than 100, the exceeding points will be used as bonus.

First name	
Last name	

Q1. [15 pts] Variants of Trees

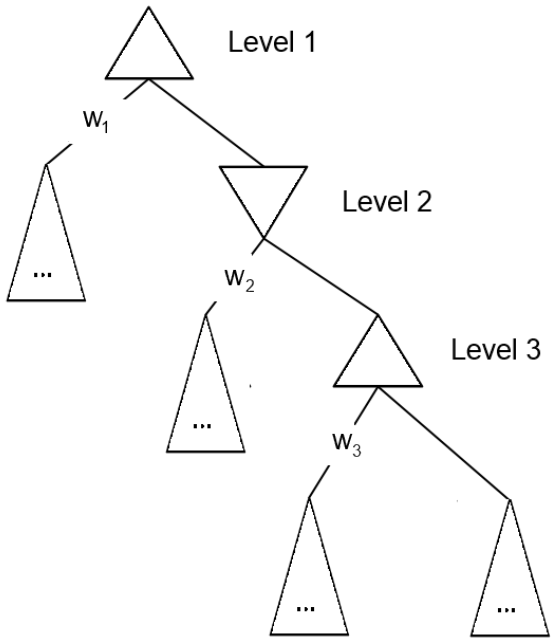
(a) Pacman is going to play against a careless ghost, which makes a move that is optimal for Pacman $\frac{1}{3}$ of the time, and makes a move that that minimizes Pacman's utility the other $\frac{2}{3}$ of the time.

(i) [2 pts] Fill in the correct utility values in the game tree below where Pacman is the maximizer: **Note: let 9 become -12, 6 become 12**



(ii) [2 pts] Draw a complete game tree for the game above that contains only max nodes, min nodes, and chance nodes.

(b) Consider a modification of alpha-beta pruning where, rather than keeping track of a single value for α and β, you instead keep a list containing the best value, w_i, for the minimizer/maximizer (depending on the level) at each level up to and including the current level. Assume that the root node is always a max node. For example, consider the following game tree in which the first 3 levels are shown. When the considering the right child of the node at level 3, you have access to w_1, w_2, and w_3.



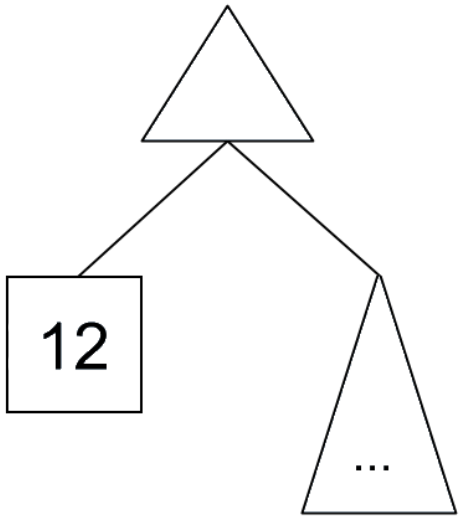
(i) [1 pt] Under this new scenario, what is the pruning condition for a max node at the n^{th} level of the tree (in terms of v and $w_1...w_n$)?

(ii) [1 pt] What is the pruning condition for a min node at the n^{th} level of the tree?

(iii) [2 pts] What is the relationship between α, β and the list of $w_1...w_n$ at a max node at the n^{th} level of the tree?

- ◯ $\sum_i w_i = \alpha + \beta$
- ◯ $\max_i w_i = \alpha, \quad \min_i w_i = \beta$
- ◯ $\min_i w_i = \alpha, \quad \max_i w_i = \beta$
- ◯ $w_n = \alpha, \quad w_{n-1} = \beta$
- ◯ $w_{n-1} = \alpha, \quad w_n = \beta$
- ◯ None of the above. The relationship is _____

(c) Pacman is in a dilemma. He is trying to maximize his overall utility in a game, which is modeled as the following game tree.



The left subtree contains a utility of 12. The right subtree contains an unknown utility value. An oracle has told you that the value of the right subtree is one of -3, -12, or 24. You know that each value is equally likely, but without exploring the subtree you do not know which one it is.

Now Pacman has 3 options:

1. Choose left;
2. Choose right;
3. Pay a cost of c = 1 to explore the right subtree, determine the exact utility it contains, and then make a decision.

(i) [3 pts] What is the expected utility for option 3?

(ii) [4 pts] For what values of c (for example, $c > 5$ or $-2 < c < 2$) should Pacman choose option 3? If option 3 is never optimal regardless of the value for c, write None.

Q2. [10 pts] Games and Utilities

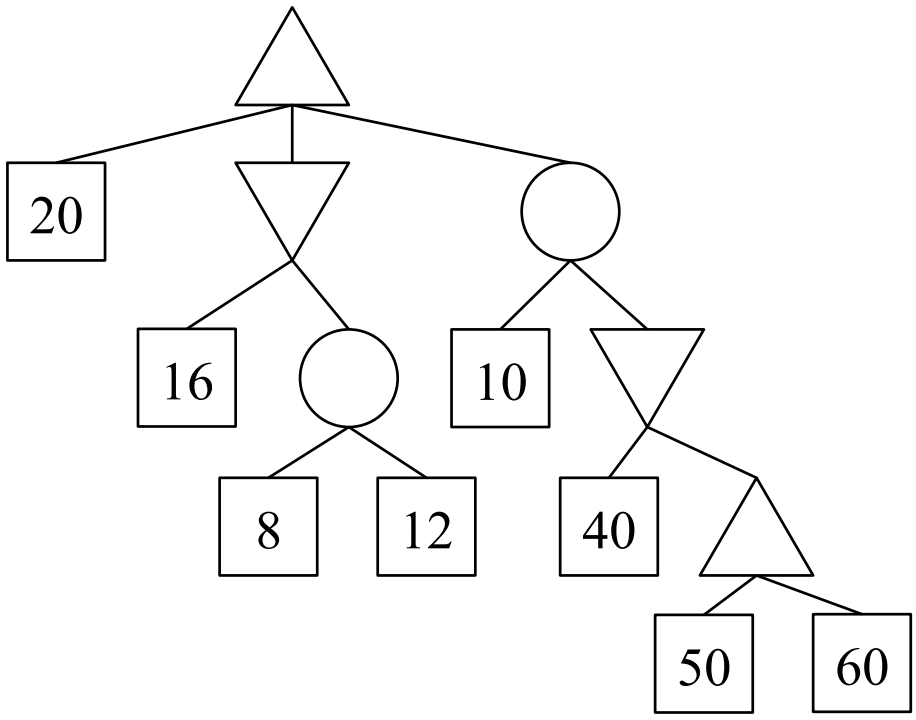
(a) Games. Consider the game tree below, which contains maximizer nodes, minimizer nodes, and chance nodes. For the chance nodes the probability of each outcome is left child with **probability 2/5** and right child with **probability 3/5**.

(i) [3 pts] Fill in the values of each of the nodes.

(ii) [4 pts] Is pruning possible?

◯ No. Brief justification: _____

◯ Yes. Cross out the branches that can be pruned.



(b) Utilities. Pacman's utility function is $U(\$x) = \sqrt{x}$. He is faced with the following lottery: $[0.5, \$36\ ;\ 0.5, \$64]$. Compute the following quantities:

(i) [1 pt] What is Pacman's expected utility?

$EU([0.5, \$36\ ;\ 0.5,\ \$64]) =$

(ii) [1 pt] What is Equivalent Monetary Value for this lottery?

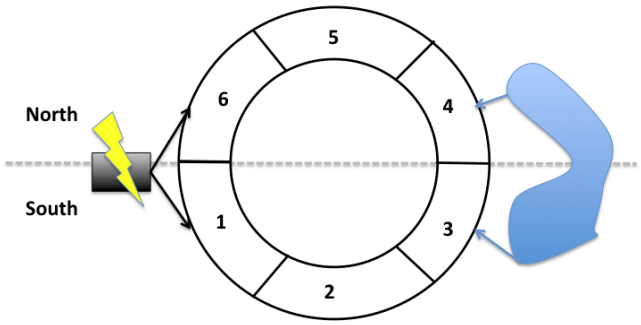
$EMV([0.5, \$36\ ;\ 0.5, \$64]) =$

(iii) [1 pt] What is the maximum amount Pacman would be willing to pay for an insurance that guarantees he gets \$64 in exchange for giving his lottery to the insurance company?

Q3. [9 pts] CSPs: Apple's New Campus

Apple's new circular campus is nearing completion. Unfortunately, the chief architect on the project was using Google Maps to store the location of each individual department, and after upgrading to iOS 6, all the plans for the new campus were lost!

The following is an approximate map of the campus:



The campus has six offices, labeled 1 through 6, and six departments:

[noitemsep,topsep=0in]Legal (L) Maps Team (M) Prototyping (P) Engineering (E) Tim Cook's office (T) Secret Storage (S)

Offices can be *next to* one another, if they share a wall (for an instance, Offices 1-6). Offices can also be *across* from one another (specifically, Offices 1-4, 2-5, 3-6).

The Electrical Grid is connected to offices 1 and 6. The Lake is visible from offices 3 and 4. There are two "halves" of the campus – South (Offices 1-3) and North (Offices 4-6).

The constraints are as follows:

i. (L)egal wants a view of the lake to look for prior art examples.

ii. (T)im Cook's office must not be across from (M)aps.

iii. (P)rototyping must have an electrical connection.

iv. (S)ecret Storage must be next to (E)ngineering.

v. (E)ngineering must be across from (T)im Cook's office.

vi. (P)rototyping and (L)egal cannot be next to one another.

vii. (P)rototyping and (E)ngineering must be on opposite sides of the campus (if one is on the North side, the other must be on the South side).

viii. No two departments may occupy the same office.

(a) [3 pts] **Constraints**. Note: There are multiple ways to model constraint *viii*. In your answers below, assume constraint *viii* is modeled as multiple pairwise constraints, not a large n-ary constraint.

(i) [1 pt] Circle your answers below. Which constraints are unary?

i *ii* *iii* *iv* *v* *vi* *vii* *viii*

(ii) [1 pt] In the constraint graph for this CSP, how many edges are there?

(iii) [1 pt] Write out the explicit form of constraint *iii*.

(b) [6 pts] **Domain Filtering.** *We strongly recommend that you use a pencil for the following problems.*

(i) [2 pts] The table below shows the variable domains after unary constraints have been enforced and the value 1 has been assigned to the variable P.

Cross out all values that are eliminated by running Forward Checking after this assignment.

L			3	4		
M	1	2	3	4	5	6
P	1					
E	1	2	3	4	5	6
T	1	2	3	4	5	6
S	1	2	3	4	5	6

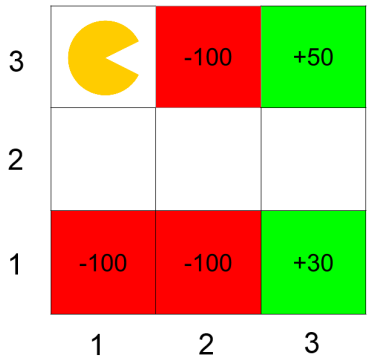
(ii) [4 pts] The table below shows the variable domains after unary constraints have been enforced, the value 1 has been assigned to the variable P, and now the value 3 has been assigned to variable T.

Cross out all values that are eliminated if arc consistency is enforced after this assignment. (Note that enforcing arc consistency will subsume all previous pruning.)

L			3	4		
M	1	2	3	4	5	6
P	1					
E	1	2	3	4	5	6
T			3			
S	1	2	3	4	5	6

Q3. [9 pts] Deep inside Q-learning

Consider the grid-world given below and an agent who is trying to learn the optimal policy. Rewards are only awarded for taking the *Exit* action from one of the shaded states. Taking this action moves the agent to the Done state, and the MDP terminates. Assume $\gamma = 1$ and $\alpha = 0.5$ for all calculations. All equations need to explicitly mention γ and α if necessary. **Note: -100 is now -75**



(a) [3 pts] The agent starts from the top left corner and you are given the following episodes from runs of the agent through this grid-world. Each line in an Episode is a tuple containing (s, a, s', r).

Episode 1	Episode 2	Episode 3	Episode 4	Episode 5
(1,3), S, (1,2), 0	(1,3), S, (1,2), 0	(1,3), S, (1,2), 0	(1,3), S, (1,2), 0	(1,3), S, (1,2), 0
(1,2), E, (2,2), 0	(1,2), E, (2,2), 0	(1,2), E, (2,2), 0	(1,2), E, (2,2), 0	(1,2), E, (2,2), 0
(2,2), E, (3,2), 0	(2,2), S, (2,1), 0	(2,2), E, (3,2), 0	(2,2), E, (3,2), 0	(2,2), E, (3,2), 0
(3,2), N, (3,3), 0	(2,1), Exit, D, -100	(3,2), S, (3,1), 0	(3,2), N, (3,3), 0	(3,2), S, (3,1), 0
(3,3), Exit, D, +50		(3,1), Exit, D, +30	(3,3), Exit, D, +50	(3,1), Exit, D, +30

Fill in the following Q-values obtained from direct evaluation from the samples:

$Q((3,2), \text{N}) =$ _____ $Q((3,2), \text{S}) =$ _____ $Q((2,2), \text{E}) =$ _____

(b) [3 pts] Q-learning is an online algorithm to learn optimal Q-values in an MDP with unknown rewards and transition function. The update equation is:

$$Q(s_t, a_t) = (1 - \alpha)Q(s_t, a_t) + \alpha(r_t + \gamma \max_{a'} Q(s_{t+1}, a'))$$

where γ is the discount factor, α is the learning rate and the sequence of observations are $(\cdots, s_t, a_t, s_{t+1}, r_t, \cdots)$. Given the episodes in (a), fill in the time at which the following Q values first become non-zero. Your answer should be of the form (**episode#,iter#**) where **iter#** is the Q-learning update iteration in that episode. If the specified Q value never becomes non-zero, write *never*.

$Q((1,2), \text{E}) =$ _____ $Q((2,2), \text{E}) =$ _____ $Q((3,2), \text{S}) =$ _____

(c) [3 pts] In Q-learning, we look at a window of (s_t, a_t, s_{t+1}, r_t) to update our Q-values. One can think of using an update rule that uses a larger window to update these values. Give an update rule for $Q(s_t, a_t)$ given the window $(s_t, a_t, r_t, s_{t+1}, a_{t+1}, r_{t+1}, s_{t+2})$.

$Q(s_t, a_t) =$

$Q(s_t, a_t) =$

$Q(s_t, a_t) =$

Q4. [10 pts] Simulated Annealing

As in PS1 and PS2, we study the simulated annealing and implement the algorithm. Given the following pseudo code, please answer the following questions:

```
function SIMULATED-ANNEALING(problem, schedule) returns a solution state
  inputs: problem, a problem
          schedule, a mapping from time to "temperature"
  local variables: current, a node
                   next, a node
                   T, a "temperature" controlling prob. of downward steps

  current ← MAKE-NODE(INITIAL-STATE[problem])
  for t ← 1 to ∞ do
      T ← schedule[t]
      if T = 0 then return current
      next ← a randomly selected successor of current
      ΔE ← VALUE[next] – VALUE[current]
      if ΔE > 0 then current ← next
      else current ← next only with probability e^(Δ E/T)
```

(a) [5 pts] In the pseudo code above, basically the objective function is to maximum. However, sometimes we would rather do a minimal search, instead of maximal, while the objective function VALUE remains the same. Please point out those 2 spots where you need to modify the pseudo code.

(b) [5 pts] In the 3-SAT problem in your PS2, we use the objective function as an indicator for finding the number of clauses being satisfied. This time, instead of following the pseudo code, you decided to do the jumping with probability $e^{(1/T)}$ by ignoring ΔE. What is the biggest problem with this (purely mathematically)?

Q5. [12 pts] Offline MDP Simulation

Please simulate the MDP using the value iteration for the scenario below and here we will have depreciation rate γ set to **0.8**. Let k be the number of iterations. We know offline MDP has the following formula:

$$V_{k+1}(s) \leftarrow max_a \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma V_k(s')]$$

Please simulate and compute that

(a) [6 pts] When $k = 1$

$V_1(Cold) =$

$V_1(Warm) =$

$V_1(Overheated) =$

(b) [6 pts] When $k = 2$

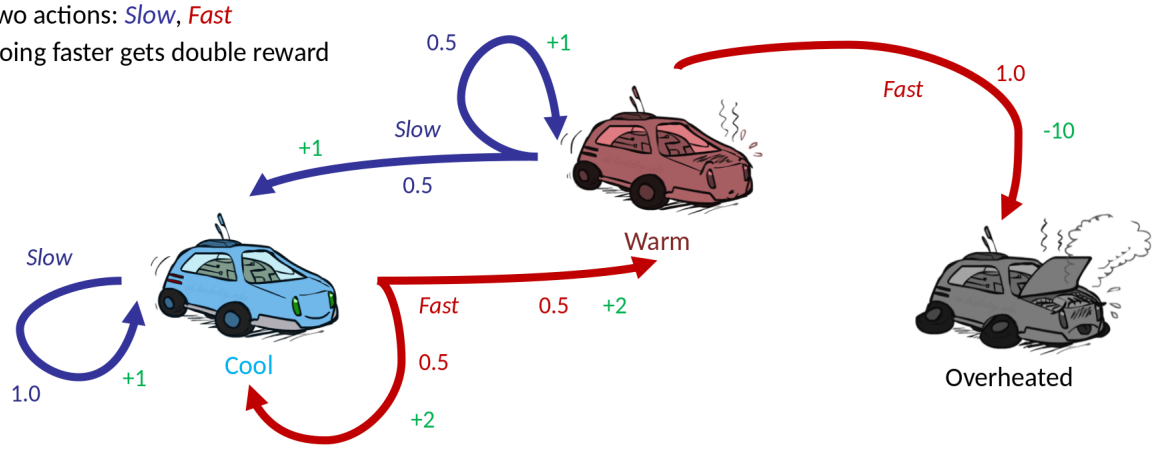
$V_2(Cold) =$

$V_2(Warm) =$

$V_2(Overheated) =$

Example: Racing

- A robot car wants to travel far, quickly
- Three states: Cool, Warm, Overheated
- Two actions: *Slow*, *Fast*
- Going faster gets double reward

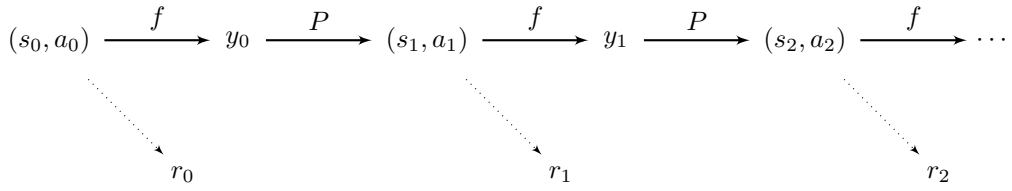


Q6. [14 pts] Bellman Equations for the Post-Decision State

Consider an infinite-horizon, discounted MDP (S, A, T, R, γ). Suppose that the transition probabilities and the reward function have the following form:

$$T(s, a, s') = P(s'|f(s, a))$$
$$R(s, a, s') = R(s, a)$$

Here, f is some deterministic function mapping $S \times A \to Y$, where Y is a set of states called *post-decision states*. We will use the letter y to denote an element of Y, i.e., a post-decision state. In words, the state transitions consist of two steps: a deterministic step that depends on the action, and a stochastic step that does not depend on the action. The sequence of states (s_t), actions (a_t), post-decision-states (y_t), and rewards (r_t) is illustrated below.



You have learned about $V^\pi(s)$, which is the expected discounted sum of rewards, starting from state s, when acting according to policy π.

$$V^\pi(s_0) = E\left[R(s_0, a_0) + \gamma R(s_1, a_1) + \gamma^2 R(s_2, a_2) + \dots\right]$$
$$\text{given } a_t = \pi(s_t) \text{ for } t = 0, 1, 2, \dots$$

$V^*(s)$ is the value function of the optimal policy, $V^*(s) = \max_\pi V^\pi(s)$.

This question will explore the concept of computing value functions on the post-decision-states y. [1]

$$W^\pi(y_0) = E\left[R(s_1, a_1) + \gamma R(s_2, a_2) + \gamma^2 R(s_3, a_3) + \dots\right]$$

We define $W^*(y) = \max_\pi W^\pi(y)$.

(a) [2 pts] Write W^* in terms of V^*.
$W^*(y) =$

- ◯ $\sum_{s'} P(s' \mid y) V^*(s')$
- ◯ $\sum_{s'} P(s' \mid y)[V^*(s') + \max_a R(s', a)]$
- ◯ $\sum_{s'} P(s' \mid y)[V^*(s') + \gamma \max_a R(s', a)]$
- ◯ $\sum_{s'} P(s' \mid y)[\gamma V^*(s') + \max_a R(s', a)]$
- ◯ None of the above

(b) [2 pts] Write V^* in terms of W^*.
$V^*(s) =$

- ◯ $\max_a[W^*(f(s, a))]$
- ◯ $\max_a[R(s, a) + W^*(f(s, a))]$
- ◯ $\max_a[R(s, a) + \gamma W^*(f(s, a))]$
- ◯ $\max_a[\gamma R(s, a) + W^*(f(s, a))]$
- ◯ None of the above

[1] In some applications, it is easier to learn an approximate W function than V or Q. For example, to use reinforcement learning to play Tetris, a natural approach is to learn the value of the block pile *after* you've placed your block, rather than the value of the pair (current block, block pile). TD-Gammon, a computer program developed in the early 90s, was trained by reinforcement learning to play backgammon as well as the top human experts. TD-Gammon learned an approximate W function.

(c) [4 pts] Recall that the optimal value function V^* satisfies the Bellman equation:

$$V^*(s) = \max_a \sum_{s'} T(s,a,s') \left(R(s,a) + \gamma V^*(s')\right),$$

which can also be used as an update equation to compute V^*.

Provide the equivalent of the Bellman equation for W^*.

$W^*(y) =$_____

(d) [3 pts] Fill in the blanks to give a policy iteration algorithm, which is guaranteed return the optimal policy π^*.

- Initialize policy $\pi^{(1)}$ arbitrarily.
- For $i = 1, 2, 3, \ldots$
  - Compute $W^{\pi^{(i)}}(y)$ for all $y \in Y$.
  - Compute a new policy $\pi^{(i+1)}$, where $\pi^{(i+1)}(s) = \arg\max_a$ ____(1)____ for all $s \in S$.
  - If ____(2)____ for all $s \in S$, **return** $\pi^{(i)}$.

Fill in your answers for blanks (1) and (2) below.

(1)
- ◯ $W^{\pi^{(i)}}(f(s,a))$
- ◯ $R(s,a) + W^{\pi^{(i)}}(f(s,a))$
- ◯ $R(s,a) + \gamma W^{\pi^{(i)}}(f(s,a))$
- ◯ $\gamma R(s,a) + W^{\pi^{(i)}}(f(s,a))$
- ◯ None of the above

(2) _____

(e) [3 pts] In problems where f is known but $P(s'|y)$ is not necessarily known, one can devise reinforcement learning algorithms based on the W^* and W^π functions. Suppose that an the agent goes through the following sequence of states, actions and post-decision states: $s_t, a_t, y_t, s_{t+1}, a_{t+1}, y_{t+1}$. Let $\alpha \in (0,1)$ be the learning rate parameter.

Write an update equation analogous to Q-learning that enables one to estimate W^*

$$W(y_t) \leftarrow (1-\alpha)W(y_t) + \alpha\,(\underline{\qquad\qquad\qquad\qquad\qquad\qquad})$$

Q7. [15 pts] BFS, DFS, Iterative Deepning Search

Given a balanced tree with branching factor = b and height = m. Let suppose the goal is hidden at level k (leave node is at level 0). We can label the node as the following : (level, how far from the left most node in the same level). So, root will be $(m, 0)$, then the children of the root will be $(1, 0), (1, 1), \cdots, (1, b-1)$. Please answer the following:

(a) [5 pts] Among the following, let t be the node where the solution is. So, which one guarantees that iterative Deepening gets defeated by DFS in terms of time complexity? Let assume $b = 5, m = 20$

(A) $t = (10, 5)$
(B) $t = (18, 5)$
(C) $t = (3, 320)$
(D) $t = (15, 0)$
(E) $t = (0, 10000)$

Your answer:

(b) [5 pts] Iterative Deepening outperforms DFS in terms of time complexity.

(A) $t = (19, 5)$
(B) $t = (0, 2)$
(C) $t = (3, 320)$
(D) $t = (15, 0)$
(E) $t = (0, 10000)$

Your answer:

(c) [5 pts] BFS outperforms DFS in terms of time complexity.

(A) $t = (18, 5)$
(B) $t = (3, 5)$
(C) $t = (10, 123)$
(D) $t = (1, 2)$
(E) $t = (0, 1)$

Your answer:

Q8. [10 pts] Offline MDP: Value Iteration, Policy Evaluation and Policy Iteration

We are given the following parameters: $S = \{s_1, ..., s_m\}$: the set of possible states. $A = \{a_1, \cdots a_n\}$: the set of actions. Let $p(s_j|s_i, a_t)$ be defined as the probability of moving from state s_i to state s_j when action a_t is taken. It is clear that $|S| = m$ and $|A| = n$. Suppose the height of the tree is k (i.e. we have level 0, ..., level k) and you are also given a policy $\pi_1 = \{a_i|a_i \in A\}$ and $|\pi_1| = k$. [If explanation is required, answers without justification will not be given partial credits]

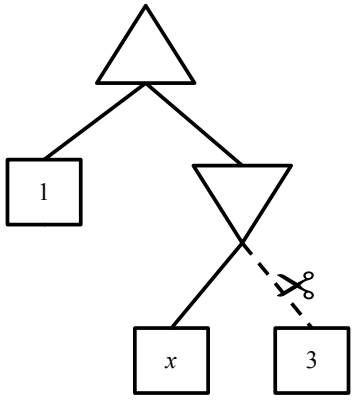
(a)[5] What is the complexity of **evaluating the whole tree** if the code is simply recursive for the value iteration:

(b) We are morphing (moving) by using **policy iteration**, from π_i then π_2, and so on, in order to find the best policy to achieve optimal, which value iteration does in one shot. Please describe the the complexity of this approach by describing
(1)[3] How many possible moves (π_i policies) are there?

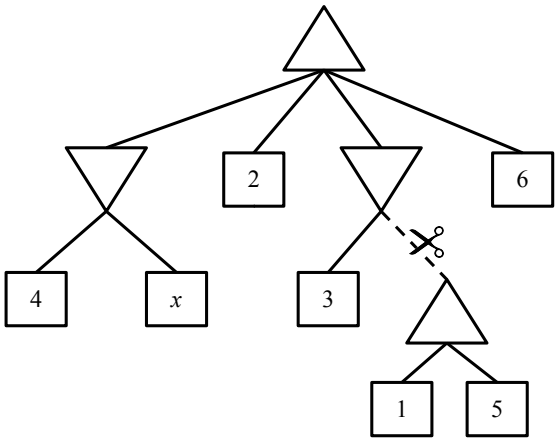
(2)[2] What is the lower bound and upper bound of this policy iteration approach?

Q9. [8 pts] Games: Alpha-Beta Pruning

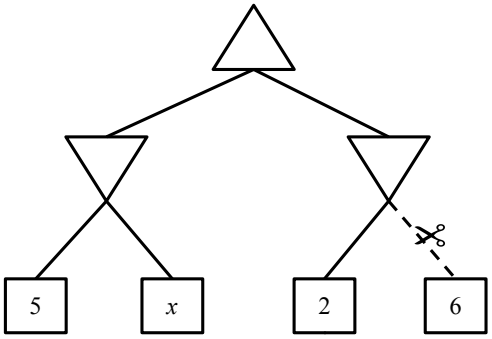
For each of the game-trees shown below, state for which values of x the dashed branch with the scissors will be pruned. If the pruning will not happen for any value of x write "none". If pruning will happen for all values of x write "all". (see next page, you can use this page as scratch paper)



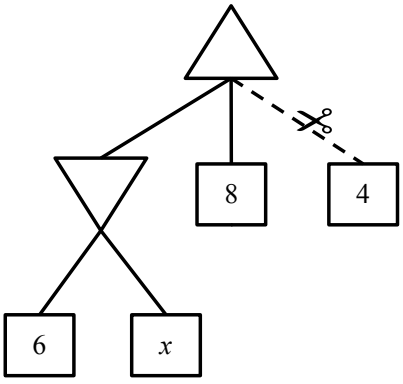
(a) Example Tree. Answer: $\underline{\mathbf{x \leq 1}}$.



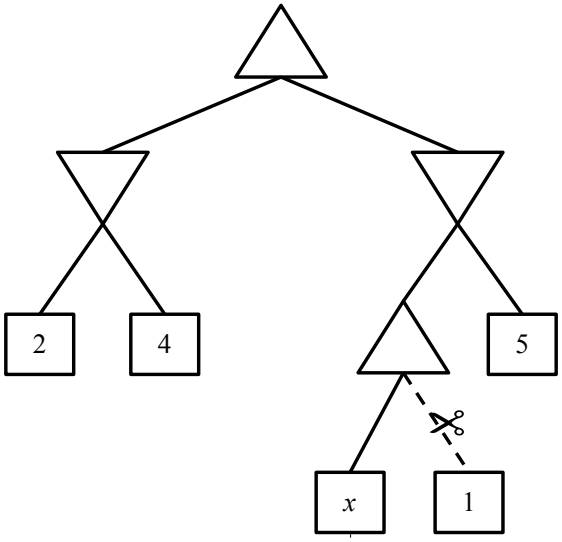
(b) Tree 1. Answer: _____



(c) Tree 2. Answer: _____



(d) Tree 3. Answer: _____



(e) Tree 4. Answer: _____

Q10. [0 pts] Scratch paper: Do Not Detach

(a) [0 pts]

Q11. [0 pts] Scratch paper: Do Not Detach

(a) [0 pts]

Q12. [0 pts] Scratch paper: Do Not Detach

(a) [0 pts]