CMPUT397 Winter 2021 Final3

导师： Baihong Qi

2. Consider a neural network, where the input *column* vector $\mathbf{s}$ is mapped by the input-weight matrix $\mathbf{A}$ and activation function g to feature vector $\mathbf{x} \doteq g(\boldsymbol{\psi})$, where $\boldsymbol{\psi} = \mathbf{A}\mathbf{s}$. Then the feature vector is mapped by output-weight matrix $\mathbf{B}$ linearly to the output vector $\hat{\mathbf{y}} \doteq \mathbf{B}\mathbf{x}$.

   Here, for a vector $\mathbf{c}$, the ith element is denoted by c_i, and for a matrix $\mathbf{C}$, the element at the ith row and the jth column is denoted by $C_{i,j}$.

   Recall the following gradients

$$\frac{\partial \hat{y}_k}{\partial B_{k,j}} = x_j,$$
$$\frac{\partial \hat{y}_k}{\partial A_{i,j}} = B_{k,i}\frac{\partial x_i}{\partial A_{i,j}} = B_{k,i}\frac{\partial g(\psi_i)}{\partial \psi_i} s_j.$$

   (a) What are the derivatives specifically for the relu activation g?

   (b) We talked about carefully initializing the weights for the NN. For example, each weight can be sampled from a Gaussian distribution. Imagine instead you decided to initialize all the weights to zero. Why would this be a problem? Hint: Consider the derivatives in (a).

6 (20).

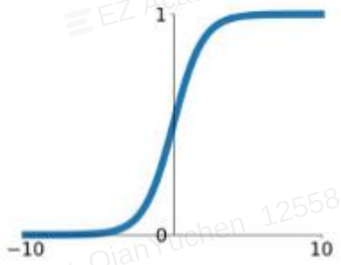
The same as worksheet 11 question 2 regarding gradient computations for neural networks except that the activation g is tanh instead of relu.

Easy Edu

Activation Function

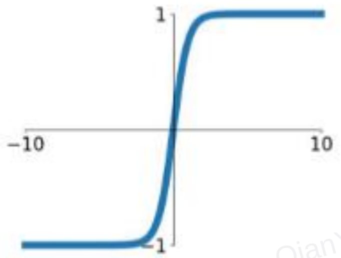
Sigmoid

$\sigma(x) = \frac{1}{1+e^{-x}}$



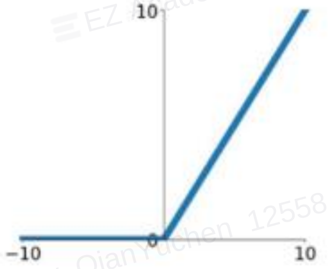
tanh

$\tanh(x)$



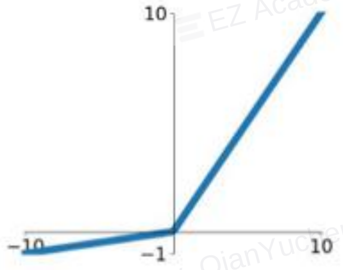
ReLU

$\max(0, x)$



Leaky ReLU

$\max(0.1x, x)$

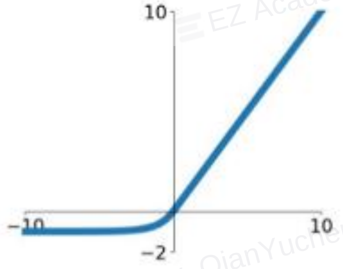


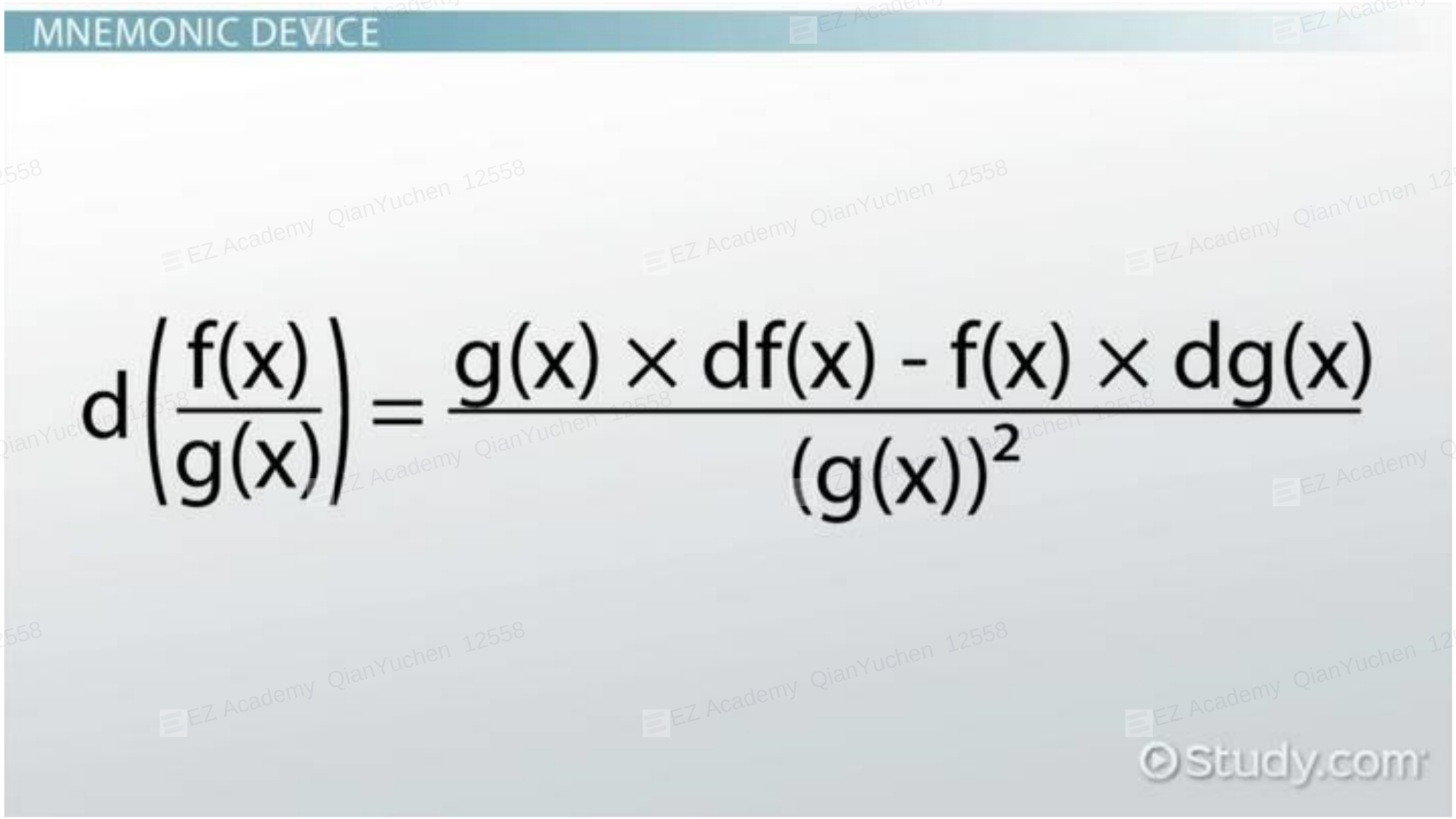
Maxout

$\max(w_1^T x + b_1, w_2^T x + b_2)$

ELU

$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$





MNEMONIC DEVICE

$$d\left(\frac{f(x)}{g(x)}\right) = \frac{g(x) \times df(x) - f(x) \times dg(x)}{(g(x))^2}$$

You can write:

$$\tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$$

It is now possible to derive using the rule of the quotient and the fact that:
derivative of e^{x} is e^{x} and
derivative of e^{-x} is $-e^{-x}$

So you have:

$$\frac{d}{dx}\tanh(x) = \frac{(e^{x} + e^{-x})(e^{x} + e^{-x}) - (e^{x} - e^{-x})(e^{x} - e^{-x})}{(e^{x} + e^{-x})^{2}}$$

$$= 1 - \frac{(e^{x} - e^{-x})^{2}}{(e^{x} + e^{-x})^{2}} = 1 - \tanh^{2}(x)$$

Ans.

$$g(x) = \tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}},$$

$$g'(x) = \frac{\partial g(x)}{\partial x} = 1 - g(x)^2,$$

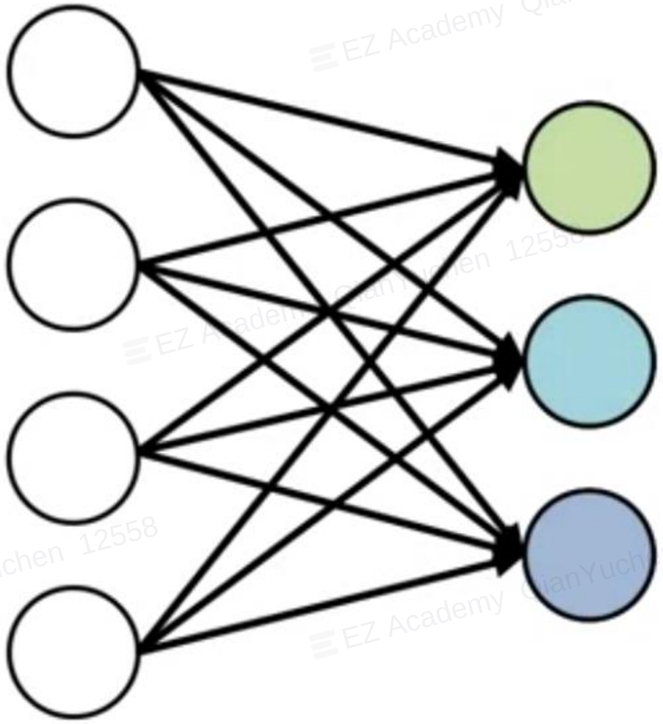
$$g(0) = 0.$$

(a) $$\frac{\partial \hat{y}_k}{\partial B_{k,j}} = x_j = g(\psi_i) = \frac{e^{\psi_i} - e^{\psi_i}}{e^{\psi_i} + e^{\psi_i}},$$

And $$\frac{\partial \hat{y}_k}{\partial A_{i,j}} = B_{k,i}\, g'(\psi_i) s_j = B_{k,i}\left(1 - \left(\frac{e^{\psi_i} - e^{\psi_i}}{e^{\psi_i} + e^{\psi_i}}\right)^2\right) s_j.$$

Optimization strategies for NN

- Weight initialization

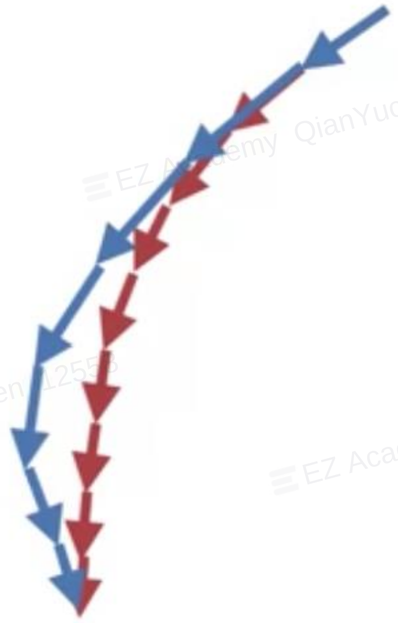


$$\mathbf{w}_{init} \sim \frac{\mathcal{N}(0,\ 1)}{\sqrt{n_{inputs}}}$$

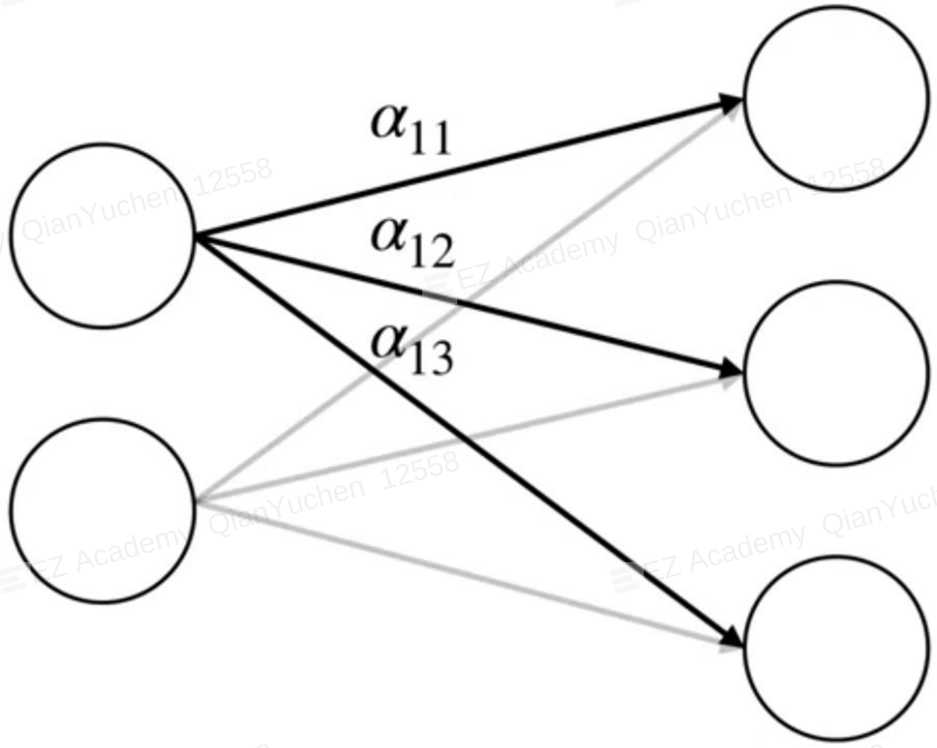
Update momentum

$$\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t - \alpha \nabla_{\mathbf{w}} L(\mathbf{w}_t) + \lambda \mathrm{M}_t$$

$$M_{t+1} \leftarrow \lambda M_t - \alpha \nabla_{\mathbf{w}} L$$



Vector step sizes



(b). $\psi_i = \sum_{\ell} A_{i,\ell} s_\ell = 0$ (as $A_{i,j} = 0 \;\; \forall i, j$)

$$\therefore \frac{\partial \hat{y}}{\partial B_{k,j}} = g(\psi_i) = g(0) = 0.$$

And $B_{k,i} = 0$

$$\therefore \frac{\partial y_k}{\partial A_{i,j}} = B_{k,i}\, g'(\psi_i)\, s_j = 0.$$

$1 \times 0.25 +$

$+10 \times 0.75 +$

0 $+1$

$+10$

V_π

3. Off-policy Monte Carlo prediction allows us to use sample trajectories to estimate the value function for a policy that may be different than the one used to generate the data. Consider the following MDP, with two states B and C, with 1 action in state B and two actions in state C, with $\gamma = 1.0$. Assume the target policy π has $\pi(A = 1|C) = 0.9$ and $\pi(A = 2|C) = 0.1$, and that the behaviour policy b has $b(A = 1|C) = 0.25$ and $b(A = 2|C) = 0.75$.

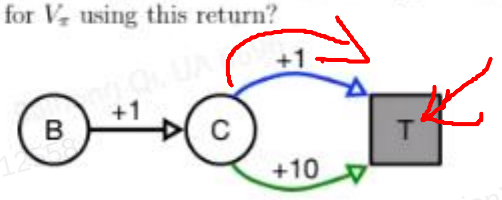
(a) What are the true values v_π?

(b) Imagine you got to execute π in the environment for one episode, and observed the episode trajectory $S_0 = B, A_0 = 1, R_1 = 1, S_1 = C, A_1 = 1, R_2 = 1$. What is the return for B for this episode? Additionally, what are the value estimates V_π, using this one episode with Monte Carlo updates?

(c) But, you do not actually get to execute π; the agent follows the behaviour policy b. Instead, you get one episode when following b, and observed the episode trajectory $S_0 = B, A_0 = 1, R_1 = 1, S_1 = C, A_1 = 2, R_2 = 10$. What is the return for B for this episode? Notice that this is a return for the behaviour policy, and using it with Monte Carlo updates (without importance sampling ratios) would give you value estimates for b.

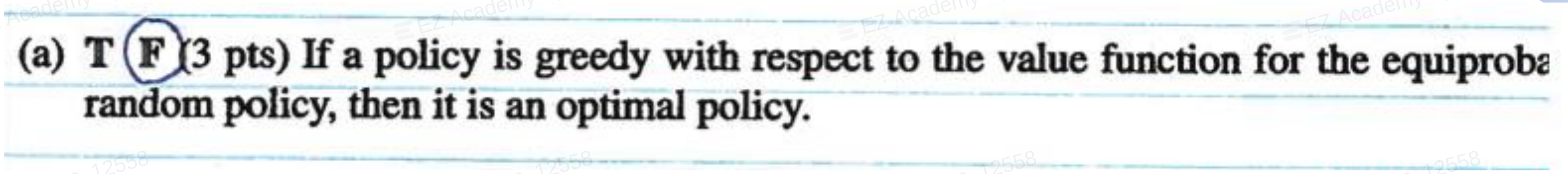
(d) But, we do not actually want to estimate the values for behaviour b, we want to estimate the values for π. So, we need to use importance sampling ratios for this return. What is the return for B using this episode, but now with importance sampling ratios? Additionally, what is the resulting value estimate for V_π using this return?

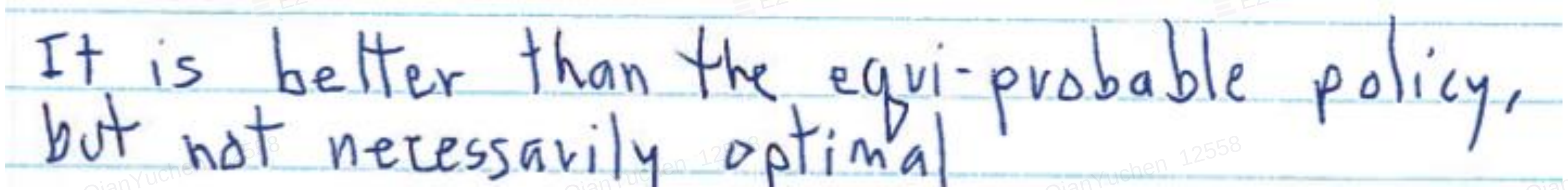
$$P = \frac{\pi}{b}$$



$$\underbrace{1 \times 0.25 + \frac{0.9}{0.25} + 10 \times 0.75 \times \frac{0.1}{0.75}}_{V_\pi}$$

判断正误


(a) T [✓ F] (3 pts) If a policy is greedy with respect to the value function for the equiproba
random policy, then it is an optimal policy.


It is better than the equi-probable policy,
but not necessarily optimal

About Q value

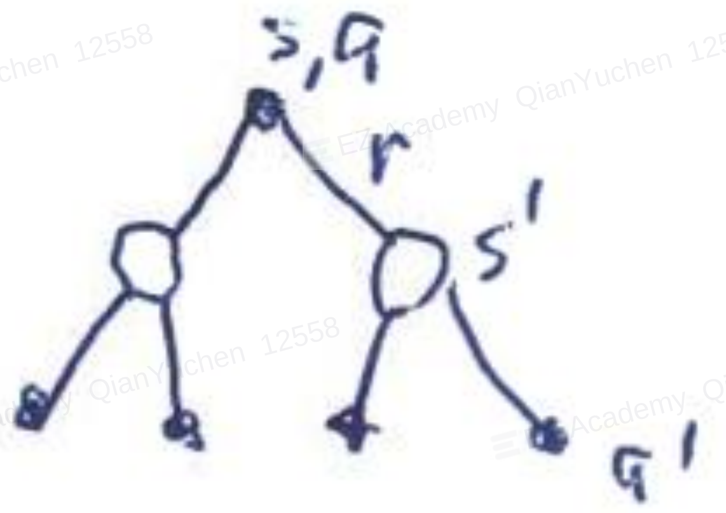
Consider the value function q_π for a continuing problem with discounting.

(a) [3 pts] Give an equation *defining* $q_\pi(s, a)$ in terms of the subsequent rewards $R_{t+1}, R_{t+2}, \ldots$ that would follow if you were in s, a at time t. (If you choose to write it in terms of the return, G_t, define your return notation in terms of the underlying rewards.)

$$q_\pi(s,a) = E_\pi[G_t \mid S_t = s, A_t = a]$$

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots$$

(b) [3 pts] Sketch the backup diagram for q_π. Find a place to attach the labels s, a, r, s' and a'.



(c) [4 pts] What is the Bellman equation for q_π? Write it in an explicit form in terms of $p(s'|s,a)$ and $r(s,a,s')$ so that no expected value notation appears.

$$q_\pi(s,a) = \sum_{s',r} p(s',r|s,a) \Big(r + \gamma \sum_{a'} \pi(a'|s')\, q_\pi(s',a'$$

(d) [4 pts] Consider the simplest, asynchronous dynamic-programming algorithm for computing q_π. An array $q(s, a)$ is initialized to zero. Then there are repeated sweeps through the state–action space, with an update done for each state–action pair. What is that DP update?

- Use $\leftarrow$ instead of =

(e) [4 pts] Consider the simplest temporal-difference learning method for estimating q_π from experience. An array $Q(s, a)$ is initialized to zero. Then an infinite sequence of experience, $S_0, A_0, R_1, S_1, A_1, R_2, S_2, \ldots$, is processed, with one update to $Q(S_t, A_t)$ done for each transition. What is the equation for that TD update?

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha\left[R_{t+1} + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t)\right]$$

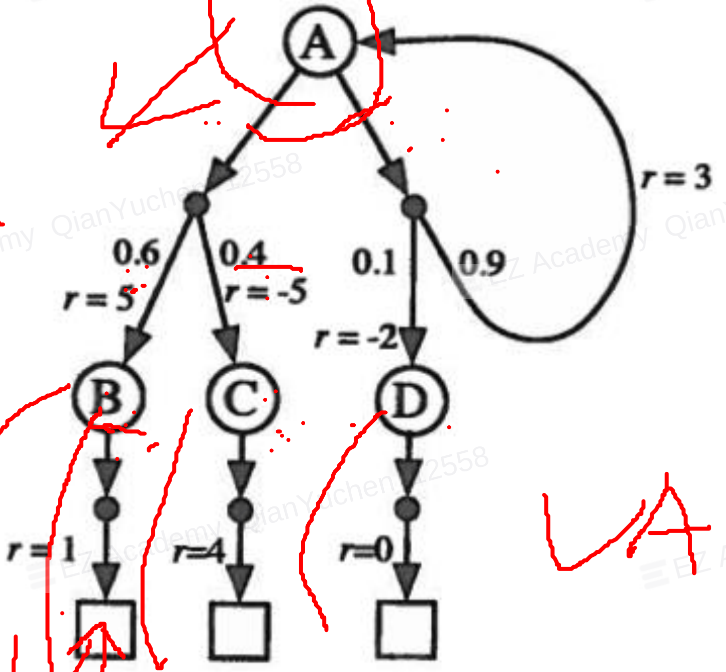
(f) [4 pts] Now consider the simplest Monte Carlo learning method for estimating q_π from episodic experience. An array $Q(s, a)$ is initialized to zero. Then an infinite sequence of episodes is experienced, where an individual episode of experience is denoted $S_0, A_0, R_1, S_1, A_1, R_2, S_2, \ldots, R_T, S_T$, where T is the final time step of the episode, S_T is the terminal state, and the value of all actions from the terminal state are taken to be zero. When an episode is processed, one update to $Q(S_t, A_t)$ is made for each time step $t < T$. What is the equation for that Monte-Carlo update? (If you choose to write it in terms of the return, define your return notation in terms of the underlying rewards.)

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha\left[G_t - Q(S_t, A_t)\right]$$

G_t as in (a)

4. **MDP values (7 points total)**

Consider the following fragment of an MDP graph, where open circles represent states, solid circles represent actions, and boxes represent the terminal state. The fractional numbers indicate the world's transition probabilities and the whole numbers indicate expected rewards. The discount rate is 0.9.



$VB = 100\% \times (r + V_T)$ 0

$VB = 1$

$VC = 4$

$VD = 0$

$VA = 0.6 \times (5 + 0.9 \times 1) +$

$0.4 \times (-5 + 0.9 \times 4) +$

(a) (2 pts) Under the policy that always takes the left-side action out of state A, what are the values of the four states?

$$v_\pi(B) = 1$$

$$v_\pi(C) = 4$$

$$v_\pi(D) = 0$$

$$v_\pi(A) = 0.6*5.9+0.4*(-5+.9*4)$$

(b) (2 pts) Under the policy that always takes the right-side action out of state A, what are the values of the four states?

$v_\pi(B) = 1$

$v_\pi(C) = 4$

$v_\pi(D) = 0$

$v_\pi(A) = 0.1*(-2)+0.9*(3+0.9*v(A))$, or $v(A)=250/19$

(c) (3 pts) Under the policy that takes both actions out of state A with equal probability, what is the value of state A?

v(A)=0.5*[0.6*(5+0.9*v(B)) + 0.4*(-5+0.9*v(C))] + 0.5*[0.1*(-2+0.9*v(D)) + 0.9*(3+0.9*v(A))]

Backup diagrams (15 points total)

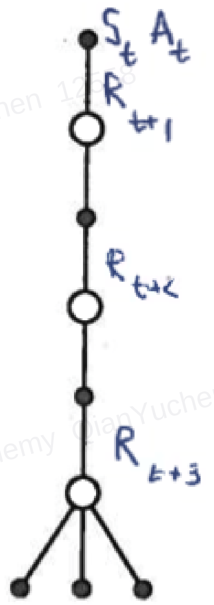
A backup diagram is a pattern of nodes and edges representing the update done by a particular algorithm for updating a value function. The node for the state or state-action pair being updated goes at the top and the things it is updated from go below. Open circles are used to represent state nodes, solid circles for action nodes, and squares for terminal states.

Sketch the backup diagrams for the following tabular algorithms. It is not necessary to attach label to the nodes or links.

(a) (3 pts) TD(0)

(b) (3 pts) single-step full backup of v_π

8. (4 points) Write the tabular learning rule that corresponds to the backup diagram below. This update rule should be of the same form as our standard learning rule: on the left-hand side should be a new estimated value for events at time t and on the right-hand side should be the old value, a step-size, and an error between the old value and a target for the value. Use the diagram to figure out what the target should be. Assume a discounted continuing problem so that you don't have to worry about episode termination. Hint: first carefully label the diagram with S_t, A_t at the top and then continuing with $R_{t+1}, S_{t+1}\ldots$



$$Q(S_t,A_t) \leftarrow Q(S_t,A_t) + \alpha\left[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 \sum_a \pi(a|S_{t+3})\, Q(S_{t+3},a) - Q(S_t,A_t)\right]$$

11. (7 pts) What are the key differences between on-policy and off-policy learning? (point form ok)

In on-policy learning, there is one policy that you learn about and use to generate the data.

Whereas, in off-policy learning, there are two policies, one you learn about and the other to generate the data that you learn from.

In off-policy learning, exploration is easier, but using what you have learned is

With linear function approximation, on-policy learning is sound while off-policy learning may diverge.

15. Problem formulation (18 points total)

The *Genius* is a gas-electric hybrid car that automatically switches between two modes, one using an electric motor and one using a gas engine, depending on the battery level and the desired acceleration as indicated by the position of the accelerator foot pedal. When the gas engine is used, the battery is charged at a constant rate up to its maximum capacity. When the electric motor is used, the battery is depleted at a rate proportional to the desired acceleration. Whichever power mode is used, the engine or motor is automatically run so as to achieve the desired acceleration. A decision about which mode to use is made once each second, and we would like to use reinforcement learning to find a decision-making policy that minimizes gas consumption, which is zero when using the electric motor and proportional to the desired acceleration when using the gas engine, in a discounted sense.

This system constitutes a continuous-state MDP. Specify the

(i) (3 pts) states

The battery level

(ii) (3 pts) actions

The two modes (gas engine or electric motor)

(iii) (3 pts) rewards

gas consumption

(iv) (3 pts) time steps

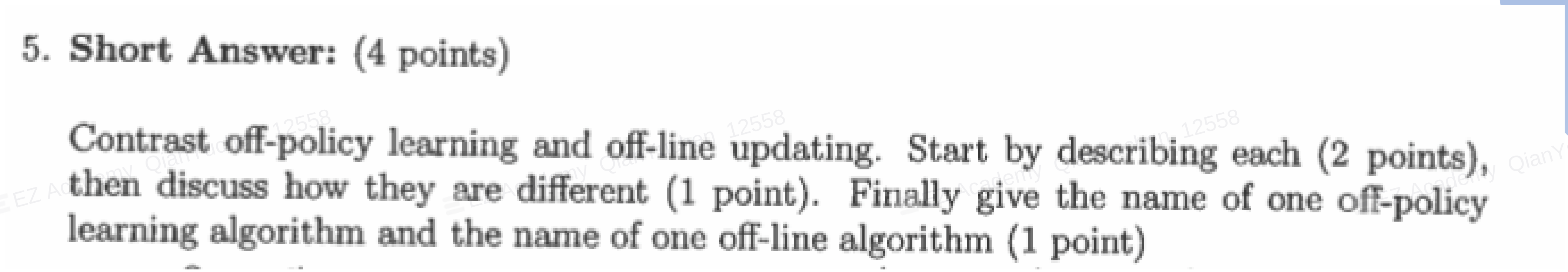
once per second

(v) (3 pts) returns

the total discounted gas consumption from each second forward

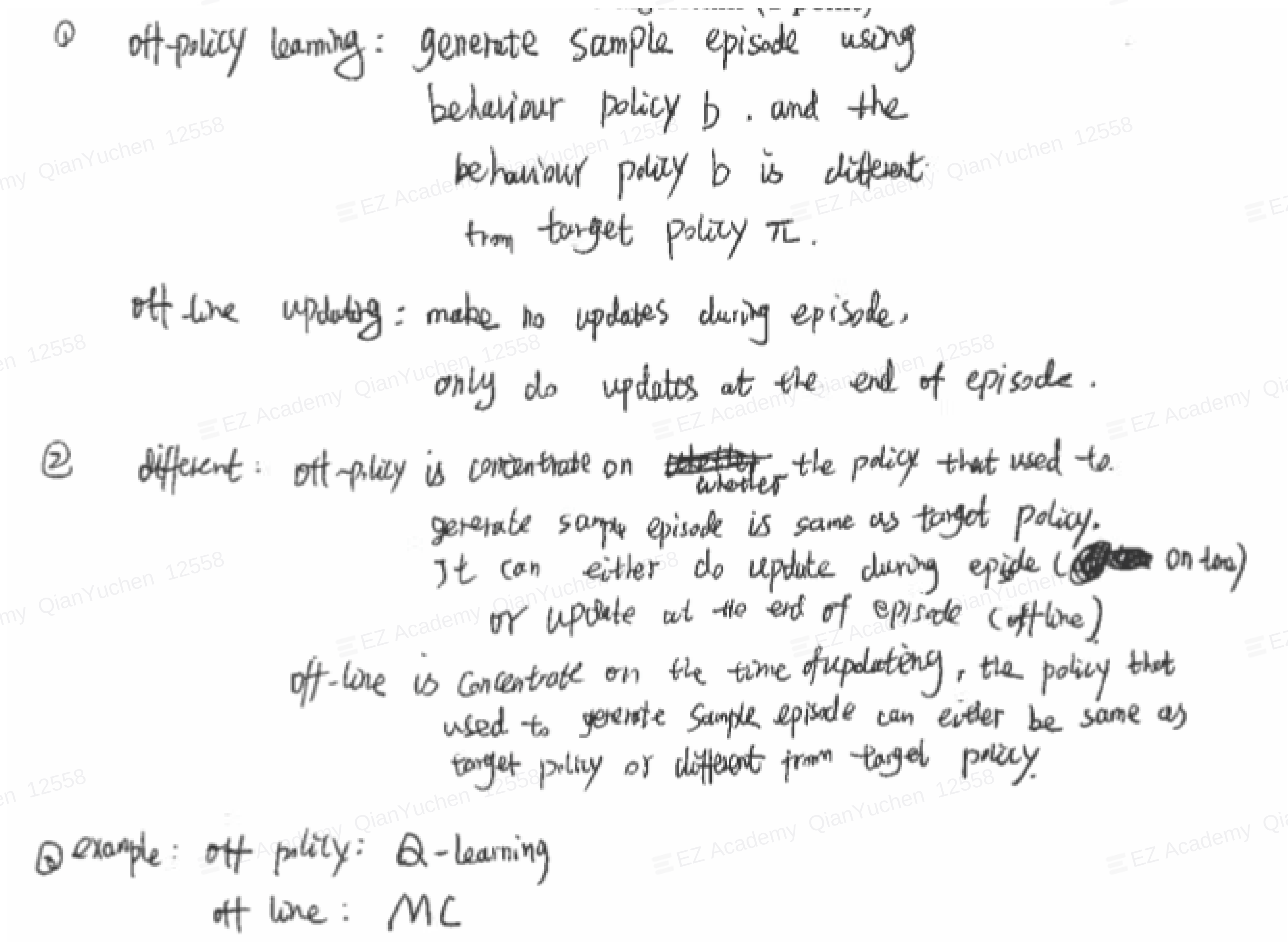
(vi) (3 pts) values (in general terms)

the expected discounted gas consumption from a state, given average acceleration requests



5. **Short Answer:** (4 points)

Contrast off-policy learning and off-line updating. Start by describing each (2 points), then discuss how they are different (1 point). Finally give the name of one off-policy learning algorithm and the name of one off-line algorithm (1 point)



① off-policy learning: generate sample episode using behaviour policy b. and the behaviour policy b is different from target policy π.

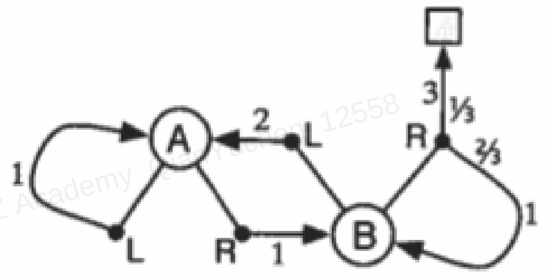
off-line updating: make no updates during episode. only do updates at the end of episode.

② different: off-policy is concentrate on ~~whether~~ whether the policy that used to generate sample episode is same as target policy.
It can either do update during epside (on-line) or update at the end of episode (off-line)

off-line is concentrate on the time of updating, the policy that used to generate sample episode can either be same as target policy or different from target policy

③ example: off policy: Q-learning
off line: MC

7. Trajectories, returns, and values (12 points total)



Consider the MDP above, in which there are two nonterminal states, A and B, one terminal state, shown by the box, and two actions, R and L, leaving each nonterminal state to the left or right. The deterministic rewards on each transition are as indicated by the whole numbers and the transition probabilities are either one or as indicated by the fractions on the outgoing edges from the R action from B. That is, if the R action is taken in B, then the transition may be either to the terminal state with a reward of 3 or back to B with a reward of 1. These two possibilities occur with probabilities 1/3 (for the transition to the terminal state) and 2/3 (for the transition back to state B). Assume discounting with $\gamma = 0.9$. The answers to several subparts of this question will result in numbers with many decimal places. In these subparts feel free to express your answer as a fraction or an expression rather than reducing it to a number.

Consider three deterministic policies, π_1, π_2, and π_3:

$\pi_1(\mathsf{A}) = \mathsf{L}$	$\pi_2(\mathsf{A}) = \mathsf{R}$	$\pi_3(\mathsf{A}) = \mathsf{R}$
$\pi_1(\mathsf{B}) = \mathsf{L}$	$\pi_2(\mathsf{B}) = \mathsf{L}$	$\pi_3(\mathsf{B}) = \mathsf{R}$

(a) Show a trajectory (sequence of states, actions and rewards) starting from A for policy π_1:

A L 1 A L 1 A L 1 ···

(b) For this trajectory, considering its first state to be S_0, what is the return G_0?

$$G_0 = R_1 + \gamma R_2 + \gamma^2 R_3 + \cdots = \sum_{k=0}^{\infty} \gamma^k R_{k+1} = 1 \times \frac{1}{1-\gamma} = 1 \times \frac{1}{1-0.9} = 10$$

(c) What is $v_{\pi_1}(\text{A})$?

$$v_{\pi_1}(A) = \mathbb{E}_{\pi}[10 \mid S_t = A] = 10$$

(d) What is $q_{\pi_1}(\text{A}, \text{R})$?

$$q_{\pi_1}(A,R) = 1 + 2 \times 0.9 + 0.9^2 \times 10 = 1 + 1.8 + 8.1 = 10.9$$

(e) Show a trajectory (sequence of states, actions and rewards) from A for policy π_2:

A R 1 B L 2 A R 1 B L 2 ···

(f) For this trajectory, considering its first state to be S_0, what is G_0? (It would be ok to write it as an expression rather than reducing it to a number.)

$$G_0 = R_1 + \gamma R_2 + \gamma^2 R_3 + \gamma^3 R_4 + \cdots$$

$$= 1 \times \gamma^0 + 2 \times \gamma^1 + 1 \times \gamma^2 + 2 \times \gamma^3 + \cdots$$

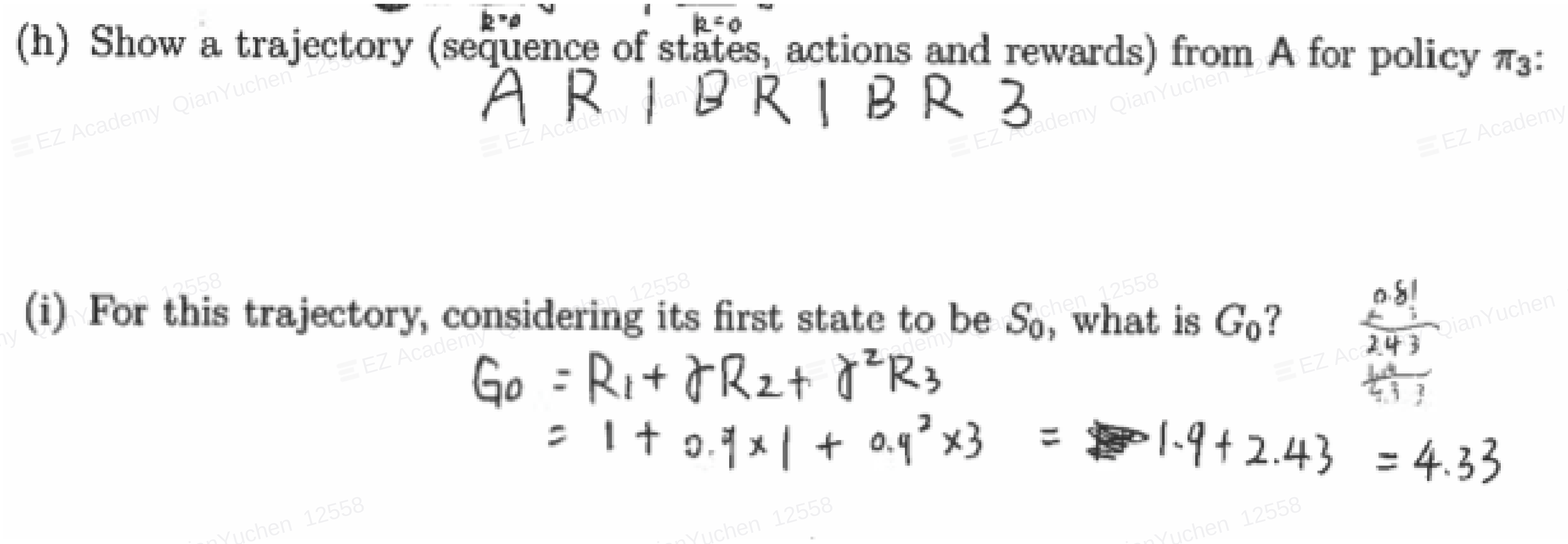
$$= \sum_{k=0}^{\infty} \gamma^{2k} + 2 \sum_{k=0}^{\infty} \gamma^{2k+1}$$

(g) For this trajectory, considering its first state to be S_0, what is G_1? (It would be ok to write it as an expression rather than reducing it to a number.)

$$G_1 = R_2 + \gamma R_3 + \gamma^2 R_4 + \cdots$$

$$= 2\times\gamma^0 + 1\times\gamma^1 + 2\times\gamma^2 + \cdots$$

$$= 2\sum_{k=0}^{\infty}\gamma^{2k} + \sum_{k=0}^{\infty}\gamma^{2k+1}$$



(h) Show a trajectory (sequence of states, actions and rewards) from A for policy π_3:

A R 1 B R 1 B R 3

(i) For this trajectory, considering its first state to be S_0, what is G_0?

$$G_0 = R_1 + \gamma R_2 + \gamma^2 R_3$$

$$= 1 + 0.9 \times 1 + 0.9^2 \times 3 = 1.9 + 2.43 = 4.33$$

$$\begin{array}{r} 0.81 \\ \times\ 3 \\ \hline 2.43 \\ +\ 1.9 \\ \hline 4.33 \end{array}$$

(j) What is $v_{\pi_3}(B)$?

(k) What is $v_{\pi_3}(A)$?

(l) What is $q_{\pi_3}(B, L)$?

(j) $V_{\pi_3}(B) = \frac{1}{3}(3+0) + \frac{2}{3}(1+\gamma V_{\pi_3}(B))$

$= 1 + \frac{2}{3} + \frac{9}{10}\times\frac{2}{3} V_{\pi_3}(B) = \frac{5}{3} + \frac{3}{5} V_{\pi_3}(B)$

~~$\frac{1}{3} V_{\pi_3}(B) = \frac{5}{3}$~~ ~~$V_{\pi_3}(B) = 5$~~

$\frac{2}{5} V_{\pi_3}(B) = \frac{5}{3}$ $V_{\pi_3}(B) = \frac{5}{3}\times\frac{5}{2} = \frac{25}{6}$

(k) $V_{\pi_3}(A) = 1+\gamma V_{\pi_3}(B)$

$= 1+\frac{9}{10}\times\frac{25}{6}$

$= 1+\frac{15}{4} = \frac{19}{4}$

(l) $q_{\pi_3}(B,L) = 2+\gamma q_{\pi_3}(A,R)$

$= 2+\frac{9}{10}\times\frac{19}{4}$

$= 2+\frac{171}{40} = \frac{251}{40}$

9. All about q_* (20 points total)

Consider the value function q_* for a continuing Markov decision process with discounting.

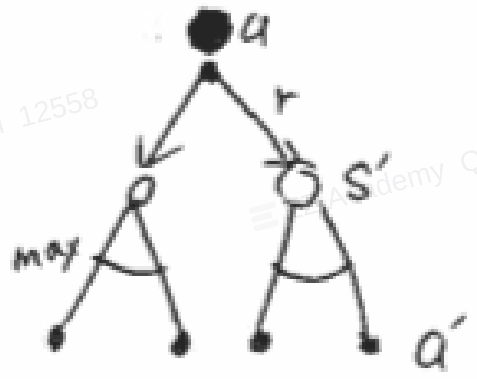
(a) [4 points] Give an equation *defining* $q_*(s,a)$ in terms of the subsequent rewards $R_{t+1}, R_{t+2}, \ldots$ given that you start, at time t, in state s taking action a. If you define q_* in terms of other things (e.g., q_π, G_t, or π_*) then be sure to define them in terms of the underlying rewards as well. Start with the definition of $q_*(s,a)$, and see if you can define it in terms of other things that eventually involve R_{t+1}, R_{t+2}, ... etc. This will require multiple equations.

$$q_*(s,a) \doteq \max_{\pi} q_\pi(s,a)$$

where $q_\pi(s,a) \doteq \mathbb{E}_\pi[G_t \mid S_t=s, A_t=a] = \mathbb{E}_\pi[\sum_{k=0}^{\infty} k^k R_{t+k+1} \mid S_t=s\}, A_t=a]$

where $G_t \doteq R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$

(b) [3 points] Sketch the update diagram (also sometimes known as a backup diagram) for the dynamic programming algorithm for q_*. Find a place to attach the labels a, r, s', and a'.

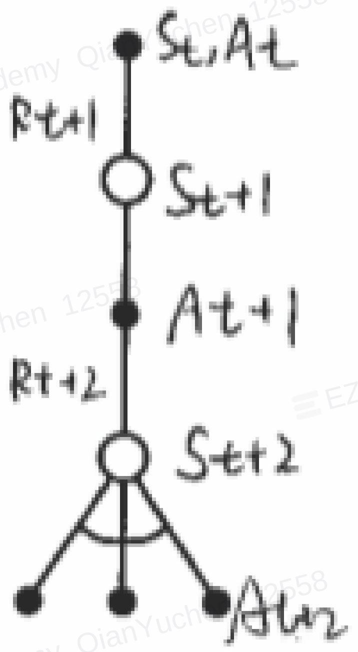


(c) [5 points] What is the Bellman equation for q_*? Write it in an explicit form in terms of $p(s', r|s, a)$ so that no expected-value notation appears.

[redacted].

$$q_*(s,a) = \sum_{s',r} p(s',r|s,a)\left[r + \gamma \max_{a'} q_*(s',a')\right]$$

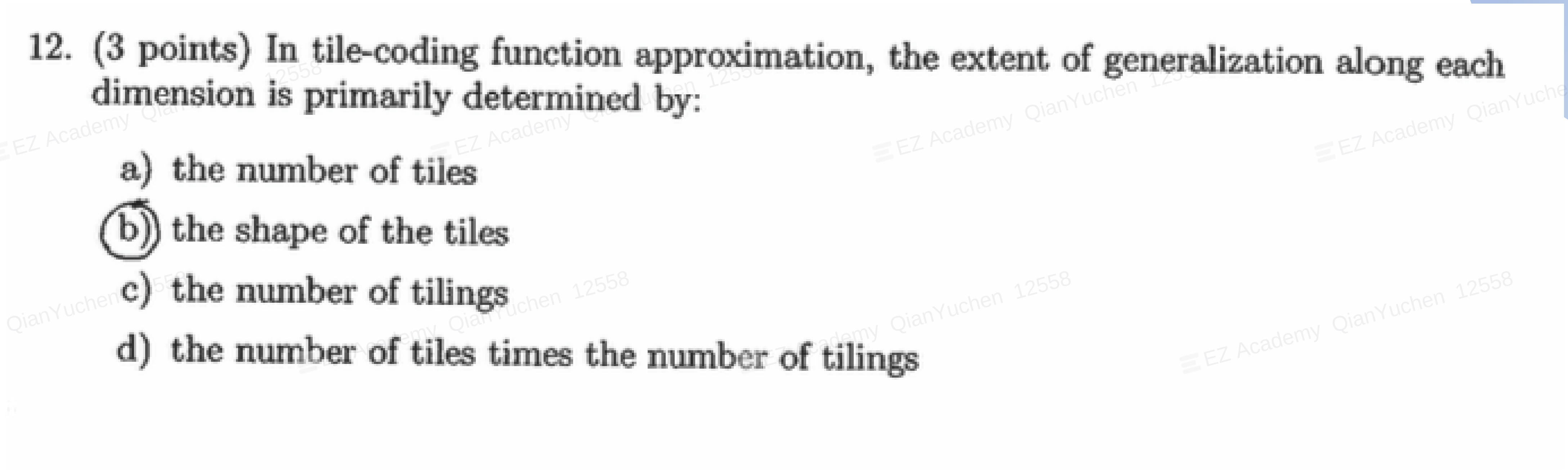
11. (4 points) Write the tabular learning rule that corresponds to the update diagram below. This update rule should be of the same form as our standard learning rule: on the left-hand side should be a new estimated value for events at time t and on the right-hand side should be the old value, a step-size, and an error between the old value and a target for the value. Use the diagram to figure out what the target should be. Assume a discounted continuing problem so that you don't have to worry about episode termination. Hint: first carefully label the diagram with S_t, A_t at the top and then continuing with R_{t+1}, $S_{t+1} \ldots$



$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha \left[G_t - Q(S_t, A_t) \right]$$

where

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 \max_a Q(S_{t+2}, a)$$



12. (3 points) In tile-coding function approximation, the extent of generalization along each dimension is primarily determined by:

a) the number of tiles

[✓ b)] the shape of the tiles

c) the number of tilings

d) the number of tiles times the number of tilings