

Question 1

s	a	s'	a'	$p(s', a s, a)$
high	search	high	search	α
high	search	low	search	$(1-\alpha)$
low	search	high	-3	$(1-\beta)$
low	search	low	search	β
high	wait	high	wait	1
low	wait	low	wait	1
low	recharge	high	0	1

which is a state of 3 states. It plots as follows
initial state goes to action update with buffer

Question 3Ex 3.15

We know that :

$$V_{\pi} = E_{\pi} [G_t | S_t = s]$$

$$V_{\pi} = E_{\pi} \left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} | S_t = s \right]$$

Now if we add a constant c to each reward R_t , then

$$R_{\text{new}} = R_{\text{old}} + c$$

$$\text{and } R'_{t+k+1} = R_{t+k+1} + c$$

$$\text{Thus: } R'_{t+k+1} = R_{t+k+1} + c$$

$$V'_{\pi} = E_{\pi} \left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} + c \sum_{k=0}^{\infty} \gamma^k | S_t = s \right]$$

$$V_{\pi'} = E_{\pi} \left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} + c \left(\frac{1}{1-\gamma} \right) | S_t = s \right]$$

Trees,

$$V_{\pi'} = V_{\pi} + c \left(\frac{1}{1-\gamma} \right)$$

$$V_{\pi'} = V_{\pi} + c$$

Thus adding a constant 'c' to each reward does not affect the relative value of any state under any policy. A term of V_c gets added to the value of each state whose value is:

$$V_c = c \left(\frac{1}{1-\gamma} \right)$$

Thus, the signs of these rewards are more important than the intervals between them.

Now

Similarly, for episodic task:

$$V_{\pi'} = E_{\pi} \left[\sum_{k=0}^{T-t-1} \gamma^k R_{t+k+1} + \sum_{k=0}^{T-t-1} \gamma^k c | S_t = s \right]$$

$$V_{\pi'} = E_{\pi} \left[\sum_{k=0}^{T-t-1} \gamma^k R_{t+k} + c \left(\frac{1 - \gamma^{(T-t-1)}}{1-\gamma} \right) | S_t = s \right]$$

$$V_t' = V_{t^*} + c \mathbb{E}_{t^*} \left[\frac{1 - \gamma}{1 - \gamma} (T-t-1) | S_t = s \right]$$

$$V_t^1 = V_{t-1} + C \left(\frac{1 - \gamma^{(T-t-1)}}{1 - \gamma} \right)$$

$$V_{t\bar{A}} = V_{t\bar{A}} + v_c(t)$$

Here C remains constant for all states but $V_c(t)$ is dependent on $(T-t-1)$ as well and as we reach

④ towards the terminal state,

is the smallest air

The value of γ is the smallest as compared to initial states and thus state values

The value of γ is the smallest as compared to initial states and thus a relative change in the state values can be seen in episodic tasks.

a relative change in the state
can be seen in episodic, ~~cases~~ tasks.

Question 5

For expressing v^* in terms of q^* , we need to ensure that q^* is calculated for the optimal action that gives the max reward for the first step. Therefore,

$$v^*(s) = \max_{a \in A(s)} q^*(s, a)$$

Question 8

Given:

$$1) p(s'|s, a) = P[S_{t+1} = s' | S_t = s, A_t = a]$$

$$2) p(s'|s, a) = \sum_{a' \in A} p(s', a' | s, a)$$

$$= P[S_{t+1} = s' | S_t = s, A_t = a]$$

$$3) p(a|s, a) = \sum_{s'} p(s', a | s, a)$$

$$= P[R_{t+1} = a | S_t = s, A_t = a]$$

$$4) \pi(s) = P[A_t = a | S_t = s]$$

5) Other notations:

a) $A_t = a$ b) $A_{t+1} = a'$

c) $R_{t+1} = r'$

d) $R_{t+2} = r''$

e) $S_t = s$

f) $S_{t+1} = s'$

g) $S_{t+2} = s''$

To calculate:

$$P[R_{t+2} = r' | S_t = s, A_t = a]$$

Answer:

- 1) we already know $p(r' | s', a')$ from eq 3, for $t+2$ that gives probability of $R_{t+2} = r'$ for all s' and a' pairs.

- 2) Now we need to calculate probability of transition to $S_{t+1} = s'$ given $S_t = s$ and

$$A_t = a \quad (\text{using eq2})$$

$$p(s' | s, a)$$

- 3) We also need probability of all action $A_{t+1} = a'$ for all s' (using eq 4) for s' .

$$\pi(a' | s')$$

- 4) Step 2 and 3 should be done for all s and $a' \in A(s')$ thus we need to sum over all s' and a' .

Hence

$$P[R_{t+2} = a' | S_t = s, A_t = a] = \left[\sum_{s', a'} p(s' | s, a) \pi(a' | s') \right].$$
$$\Rightarrow \left[\sum_{s', a'} p[S_{t+1} = s' | S_t = s, A_t = a] P[A_{t+1} = a' | S_{t+1} = s'] \right]$$
$$\times P[R_{t+2} = a' | S_{t+1} = s', A_{t+1} = a']$$

$$\Rightarrow \left[\sum_{s', a'} p(s', a | s, a) \pi(a' | s') \right] \sum_{s''} p(s'', a'' | s', a')$$

~~for different actions, we can choose any one~~

Question 9

$$E[X] = \sum_{x \in S_X} x p_x(x)$$

Thus using our result from previous question

$$E[R_{t+2} | S_t = s, A_t = a] = \sum_{a' \in R} a' \times \left[\left[\sum_{s', a', a} p(s', a' | s, a) \pi(a' | s') \right] \times \sum_{s''} p(s'' | s', a') \right]$$

Final answer

Question 10

11th April 2023

$$V_{\pi}(s) = E_{\pi} [G_t | S_t = s]$$

$$\Rightarrow E_{\pi} [R_{t+1} + \gamma G_{t+1} | S_t = s]$$

$$(as G_t = R_{t+1} + \gamma G_{t+1})$$

$s \in A$

$$= \sum_a \pi(a|s) \underbrace{\sum_{s'} \sum_a p(s'|s, a)}_{\text{or } \gamma E_{\pi}} [r_t + \gamma V_{\pi}(s')]$$

$$\Rightarrow \sum_a \pi(a|s) \sum_{s'} \sum_a p(s'|s, a) [r_t + \gamma E_{\pi} [G_{t+1} | S_{t+1} = s']]$$

$$V_{\pi}(s) = \sum_a \pi(a|s) \sum_{s', a} p(s'|s, a) [r_t + \gamma V_{\pi}(s')], \forall s \in S,$$

Bellman equation ↑

$$0.3(0.8) + 0.4(0.2)$$

$$0.24 + 0.08 = 0.32$$

$$0.32 \times 0.8 = 0.256$$

$$0.256 + 0.4(0.2)$$

$$0.256 + 0.08 = 0.336$$

Question 11

Q1 (without D)

$$R_1 = 2$$

$$R_2 = -1$$

$$R_3 = 10$$

$$R_4 = -3$$

$$\gamma = 0.5$$

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

$$G_t = R_{t+1} + \gamma R_{t+2} \quad \text{--- (1)}$$

$$G_t = R_{t+1} + \gamma G_{t+1} \quad \text{--- (2)}$$

$$G_t = R_{t+1} + \gamma G_{t+1}$$

So,

$$G_3 = R_4 + \gamma G_4$$

$$G_3 = -3$$

Now

$$G_2 = R_3 + \gamma G_3$$

$$G_2 = 10 + (0.5)(-3)$$

$$G_2 = 8.5$$

Now

$$G_1 = R_2 + \gamma G_2$$

$$G_1 = -1 + (0.5)(8.5)$$

$$G_1 = 3.25$$

Now

$$G_0 = R_1 + \gamma G_1$$

$$G_0 = 2 + (0.5)(3.25)$$

$$G_0 = 3.625$$

Now if constant reward 'c' is received at every time step; for $\gamma < 1$, then infinite horizon discounted return is:

$$G_{t+} = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

$$R_{t+k+1} = c$$

$$G_{t+} = \sum_{k=0}^{\infty} \gamma^k c$$

$$c = c \left[\sum_{k=0}^{\infty} \gamma^k \right]$$

Now this form an infinite GP, with $a = c$ and $r = \gamma$. Hence the ~~total~~ return is

$$G_{t+} = c \left(\frac{1}{1-\gamma} \right) = \left(\frac{c}{1-\gamma} \right)$$

Fence proved.

Question 12

Given :

$$v^*(s), \text{ for all } s \in S \quad \boxed{\mathbb{E}[R_t + \gamma v^*(s_{t+1})] = 0}$$

For each state, according to Bellman equation, we would need to find the best / greedy one-step reward and the action that gives us this greedy one-step reward. This would be part of the optimum policy.

$$\pi^*(s) = \operatorname{argmax}_{a \in A(s)} \mathbb{E}[R_{t+1} + \gamma v^*(s_{t+1}) \mid s_t = s, a_t = a]$$

$\boxed{\mathbb{E}[R_{t+1} + \gamma v^*(s_{t+1})] = 0 \text{ for all states}}$

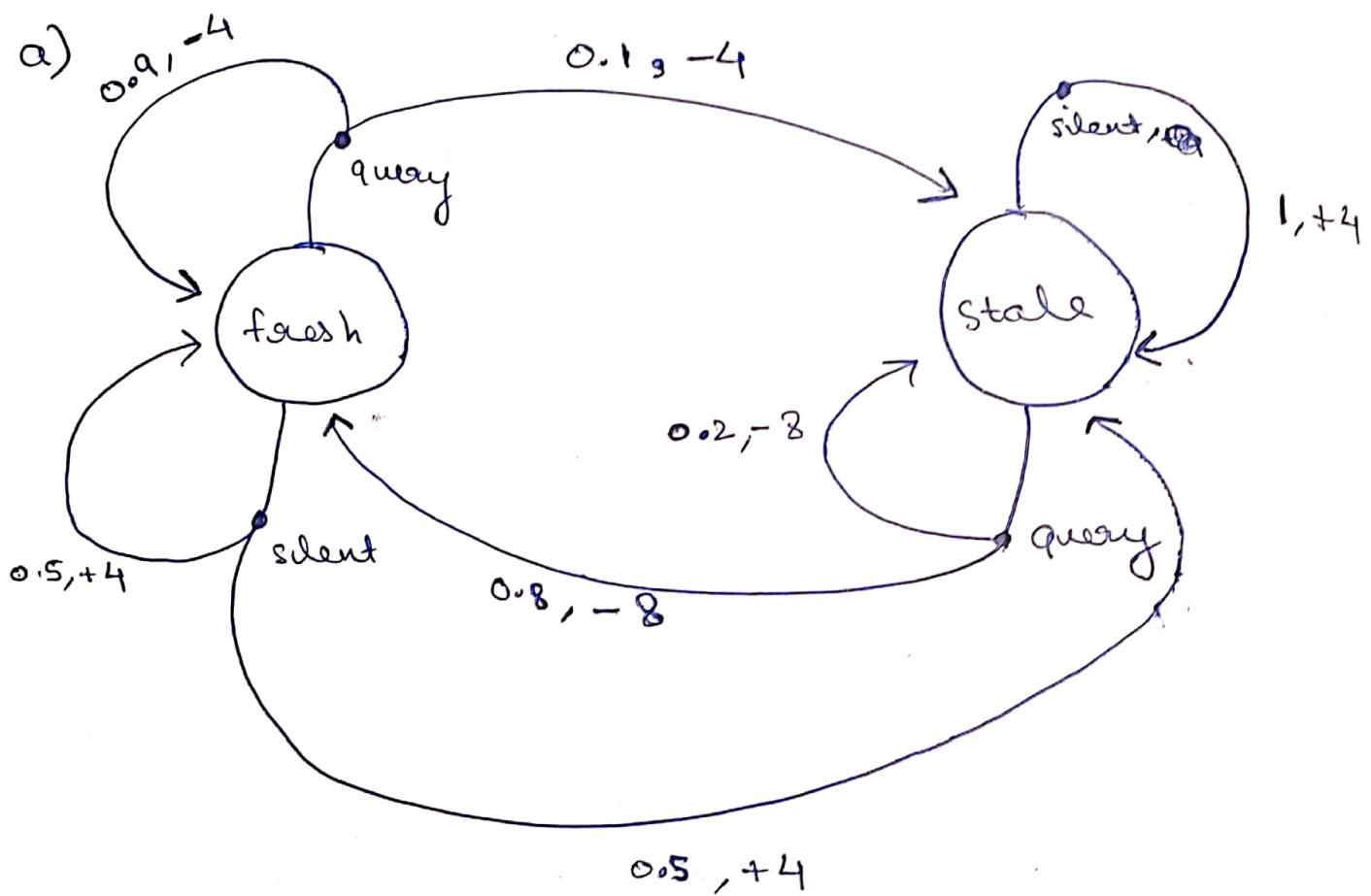
$$\pi^*(s) = \operatorname{argmax}_{a \in A(s)} \sum_{s'} \sum_a (\mathbb{E}[R_{t+1} + \gamma v^*(s')] \mid s_t = s, a_t = a)$$

$\boxed{\mathbb{E}[R_{t+1} + \gamma v^*(s')] = 0 \forall s' \in S}$

In other words, the step function is zero at each state.

$$\left[\left(\frac{-2}{\gamma - 1} \right) + \left(\frac{1}{\gamma - 1} \right) \right] \gamma = 0$$

Question 13



s	a_i	s'	$p(s' s, a)$	$r_i(s, a, s')$	$p(s', g_i s, a)$
fresh	query	fresh	0.9	-4	0.9
fresh	query	stale	0.1	-4	0.1
fresh	silent	fresh	0.5	+4	0.5
fresh	silent	stale	0.5	+4	0.5
stale	query	fresh	0.8	-8	0.8
stale	query	stale	0.2	-8	0.2
stale	silent	stale	1	+4	1

$$\gamma = 0.5$$

b) $v_0(\text{state}) = 0$ 

$$v_0(\text{fresh}) = 0$$

1st iteration

$$v_0(\text{fresh}) = \max \left[\begin{array}{l} (-4 + 0.5(0))(0.9) + (-4 + 0.5(0))0.1 \\ \oplus \\ (+4 + 0.5(0))(0.5) + (+4 + 0.5(0))0.5 \end{array} \right]$$

$$\cancel{-3.6 - 0.4} + 2 + 2$$

$$\Rightarrow \max (-4, +4)$$

$$\Rightarrow +4$$

$$v_0(\text{stale}) = \max \left[\begin{array}{l} (-3 + 0.5(0))0.8 + (-3 + 0.5(0))0.2 \\ \oplus \\ (+4 + 0.5(0))0.1 \end{array} \right]$$

$$\Rightarrow \max \left[(-0.64 - 0.16) \cancel{+ 4} \right]$$

$$\Rightarrow +4$$

2nd Iteration

$$V_{0,1}(\text{presh}) = \max \left[(-4 + 0.5(4))^{(0.9)} + (-4 + 0.5(-4))^{(0.1)}, (4 + 0.5(4))^{(0.5)} + (4 + 0.5(-4))^{(0.5)} \right]$$

$$V_{0,1}(\text{stale}) = \max \left[(-3 + 0.5(4))0.3 + (-3 + 0.5(4))0.7 + [4 + 0.5(4)]1 \right]$$

$$f'(x) = \max(-6, 6) = 6$$

~~#1000~~

3rd Iteration

$$V_2(\text{push}) = \max \left[\begin{array}{l} ((6 + 0.5(6)) 6.9) + (-14 + 0.5(6)) 0.1 \\ ((14 + 0.5(6)) 0.5 + (-6 + 0.5 \times 6) 0.5) \end{array} \right]$$

$$\Rightarrow \max [(8.1 \cancel{4} - 1.1) \text{ } \cancel{g} \text{ } (3.5 \cancel{2} - 1.5)]$$

$$\Rightarrow \max (7, 7)$$

$V_2(\text{push}) \Rightarrow +7$

$$V_2(\text{stale}) = \max \left[\begin{array}{l} ((2 + 0.5(6)) 0.3 + (-18 + 0.5(6)) 0.2) \\ (-6 + 0.5(6)) \end{array} \right]$$

$$\Rightarrow \max [4 \cancel{4} - 3 \cancel{g} - 3]$$

$V_2(\text{stale}) \Rightarrow +1$

To calculate the optimal costs and the optimal policy, we use the approach of value iteration (sort of).

We calculate V_t (stale) and V_t (fresh)

for 3 iterations and on the last iteration we include the ± 10 reward as mentioned in the question.

Since we are calculating $V(s)$ at all the 3 iterations

thus we get an idea of all

8 (2^3) possible paths to follow to reach

the state at $t = 3$.

Consistently, we see that silent action gives the maximum expected return

for both states 'stale' and 'fresh'

thus we can say that optimal policy =

$$\pi^*(s) = \text{silent} \quad \forall s \in S.$$

Also, since we used the property of base evaluation, thus the optimal cost found at the end are approximate and can be improved if more iterations are done.

$$c) \gamma = 0.5$$

$$v_0(\text{state}) = 0$$

$$v_0(\text{fresh}) = 0$$

1st iteration

$$v_0(\text{fresh}) = \max \left[(-4 + 0.5(0))(0.9) + (+4 + 0.5(0))(0.5) \right]$$

$$\cancel{-3.6 - 0.4} + 2 + 2$$

$$\Rightarrow \max (-4, +4)$$

$$\Rightarrow +4$$

$$v_0(\text{state}) = \max \left[(-3 + 0.5(0))0.8 + (-3 + 0.5(0))0.2 \right]$$

$$\Rightarrow \max \left[(-6.4) - 1.6 \right] + 4$$

$$\Rightarrow +4$$

2nd Iteration

$$V_{0.1}(\text{presh}) = \max \left[(-4 + 0.5(4))(0.9) + (-4 + 0.5(-4))(0.1), (4 + 0.5(4))(0.5) + (4 + 0.5(-4))(0.5) \right]$$

$$\begin{aligned} & \max [(-1.8 - 0.2)_{\geq 3+3}] \\ & \Rightarrow \max [(-2, 6)] \end{aligned}$$

$$V_0 \text{ (stale)} = \max \left[(-8 + 0.5(4))0.3 + (-3 + 0.5(4))0.2 + [4 + 0.5(4)] \right]$$

$$\begin{aligned} f'(x_0) &= \max(-6, 6) = 6 \\ &= 6 \cdot (x_0 - 3) + 6 \end{aligned}$$

~~W. J. G. S.~~

3rd Iteration

$$v_2(\text{fresh}) = \max \left[(-4 + 0.5(6)) 0.9 + \frac{(-4 + 0.5(6))}{0.1} \right]$$

$$= \max \left[(4 + 0.5(6))(0.5) + \frac{(4 + 0.5(6))}{0.5} \right]$$

$$= \max \left[(-0.9 + -0.1) \cdot 3.5 + 3.5 \right]$$

$$= \max \left[-7 \right]$$

$$= 7$$

$$v_2(\text{stale}) = \max \left[(-8 + 0.5(6)) 0.2 + \frac{(-8 + 0.5(6))}{0.2} \right]$$

$$= \max \left[(-4 + -1) \cdot 7 \right]$$

$$= \max \left[-4 \cdot 7 \right]$$

$$= -7$$

4th Iteration

$$V_3(\text{fresh}) = \max \left[(-4 + 0.5(7)) 0.9 + (-4 + 0.5(7)) \right]$$

$$(4 + 0.5(7)) 0.5 + (4 + 0.5(7))$$

$$\begin{aligned} &= \max \left[-0.45 - 0.05 ; 7.05 \right] \\ &= \boxed{7.05} \end{aligned}$$

stal

$V_{\frac{4}{3}}(\text{stal})$

$$V_{\frac{4}{3}}(\text{stal}) = \max \left[(-8 + 0.5(7)) 0.8 + (-8 + 0.5(7)) \right]$$

$$(4 + 0.5(7)) 1 \right]$$

$$\begin{aligned} &= \max \left[-4.5 - 0.05 ; 7.05 \right] \\ &= \boxed{7.05} \end{aligned}$$

Therefore,

$$V_0(s) = 0$$

$$V_1(s) = 4$$

$$V_2(s) = 7$$

$$V_3(s) = 7.5$$

$$V_0(f) = 0$$

$$V_1(f) = 4$$

$$V_2(f) = 7$$

$$V_3(f) = 7.5$$

Policy Iteration

$$V_0(\text{fresh}) = 0$$

$$\pi(\text{fresh}) = [0.5, 0.5]$$

$$V_0(\text{stale}) = 0$$

$$\pi(\text{stale}) = [0.5, 0.5]$$

1st Iteration

Evaluation

$$V_1(\text{fresh}) = 0.5 \left[(-4 + 0.5(0))^{0.9} + (-4 + 0.5(0))^{0.1} + (4 + 0.5(0))^{0.5} + (4 + 0.5(0))^{0.5} \right] \\ \Rightarrow 0$$

$$V_1(\text{stale}) = 0.5 \left[(-8 + 0.5(0))^{0.3} + (-8 + 0.5(0))^{0.2} + (4 + 0.5(0))^{1.0} \right] \\ \Rightarrow 0.5(-8 - 8 + 4) / 1 + 0.5(4 + 1) \\ \Rightarrow -2$$

Improvement

$$\text{old } \pi(\text{fresh}) = [0.5, 0.5]$$

$$\pi(\text{fresh}) = \text{argmax} \left[\frac{(-4 + 0.5(0))^{0.9}}{+ (-4 + 0.5(-2))^{0.1}} \right] \\ \quad \left[\frac{(4 + 0.5(0))^{0.5}}{+ (4 + 0.5(-2))^{0.5}} \right] \\ = \text{argmax} [-3.6 - 0.5, 2 + 10^5]$$

$= 1$ (action = silent)

$$\text{new } \pi(\text{fresh}) = [0, 1] \rightarrow \text{prob. of silent}$$

Now

$$\text{old } \pi(\text{stale}) = [0.5, 0.5]$$

$$\pi(\text{stale}) = \text{argmax} \left[(-3 + 0.5(0))^{0.5} + (-3 + 0.5(-2))^{0.5} \right]$$

$$= \text{argmax} \left[-6.4^{0.5} - 1.8^{0.5} + 3 \right]$$

$$\Rightarrow 1 \quad (\text{action = silent})$$

$$\text{new } \pi(\text{stale}) = [0, 1] \rightarrow \text{prob. of silent}$$

Iteration 2

Evaluation

$$V_2(\text{push}) = 0 + 1 \left[((+4 + 0.5(0))^{0.5} + (4 + 0.5(-2))^{0.5}) \right]$$

$$= 2 + 1.5$$

$$= 3.5$$

$$V_2(\text{stale}) = 0 + 1 \left[(+4 + 0.5(-2))^{0.5} \right]$$

$$\Rightarrow 1.5$$

Improvement

$$\text{old } \pi(\text{fresh}) = [0, 1] \quad (2) \pi^{\text{new}}$$

$$\begin{aligned} \pi(\text{fresh}) &= \text{argmax} \left[(-4 + 0.5(3.5))^{0.9} + (-4 + 0.5(3))^{0.1} \right. \\ &\quad \left. (+4 + 0.5(3.5))^{0.5} \right. \\ &\quad \left. + (4 + 0.5(3))^{0.5} \right] \end{aligned}$$

$$= \text{argmax} \left[-2.025 \quad -0.25 \right]$$

$$= (2) \pi^{\text{new}} \left[\frac{2.875}{2.875 + 2.75} \right]$$

$$= 1 \text{ [action = silent]}.$$

$$\boxed{\text{new } \pi(\text{fresh}) = [0, 1]}$$

$$\text{old } \pi(\text{stale}) = [0, 1]$$

$$\begin{aligned} \pi(\text{stale}) &= \text{argmax} \left[(-8 + 0.5(3.5))^{0.8} \right. \\ &\quad \left. + (-8 + 0.5(3))^{0.2} \right] \\ &= \text{argmax} \left[4 + 0.5(3) \right] \end{aligned}$$

$$= \text{argmax} \left[-5 - 1.3 \quad 5.5 \right]$$

$$= 1 \text{ (action = silent)}$$

$$\boxed{\text{new } \pi(\text{stale}) = [0, 1]}$$

Since our old $\pi(s)$ and new $\pi(s)$ are equal hence we can say that we have converged to the optimal policy and there is no need to continue the same iteration for two more steps.

Optimal policy $\pi(s) = \text{silent} + s \in S$

Note: We only do evaluation using the previous time step values in accordance with generalised policy iterations as it says that even though we choose $M=1$ (the lazy evaluation) for $v_{\pi_k}(s)$ $= T_{\pi_k}^M(v_{\pi_{k-1}}(s))$, eventually this also converges to v^* . ■

Question 14

To show: 1) Policy improvement either improves policy or leaves it unchanged.

2) If it leaves policy unchanged, then the policy is the optimal policy.

Proof: For 1), we can prove that π_{k+1} is better than π_k if $V_{\pi_{k+1}} \geq V_{\pi_k}$ for all s .

From policy improvement step, we get:

$$T_{\pi_{k+1}} V_{\pi_k} = T V_{\pi_k}$$

Now In $T V_{\pi_k}$, we are picking the greedy action that gives the max reward a with second-to-go as V_{π_k} . This would clearly be greater than or equal to picking any other action or the greedy action respectively. Thus

$$T_{\pi_{k+1}} V_{\pi_k} = T V_{\pi_k} \geq T_{\pi_k} V_{\pi_k} = V_{\pi_k}$$

Since V_{π_k} is the fixed point of T_{π_k} ,

thus

$$T_{\pi_{k+1}} V_{\pi_k} \geq V_{\pi_k}$$

Now we know that

$$T_{\pi_{k+1}} (T_{\pi_{k+1}} V_{\pi_k}) \geq T_{\pi_{k+1}} V_{\pi_k} \quad (2)$$

as now we use $T_{\pi_{k+1}} v_{\pi_k}$ as the reward to go function and the return can be either greater or equal to one reward to go given.

So

$$T_{\pi_{k+1}}(T_{\pi_{k+1}} v_{\pi_k}(s)) \geq v_{\pi_k}(s)$$

selected in

last task over has

from (1) & (2)

This can be done N times thus.

$$T_{\pi_{k+1}}^N(v_{\pi_k}(s)) \geq v_{\pi_k}(s) \quad \text{--- (3)}$$

Now since $v_{\pi_k}(s)$ is the fixed point for T_{π_k} with respect to π_k , we know that L.H.S of (3) will converge to $v_{\pi_{k+1}}(s)$.

$$\text{If } T_{\pi_{k+1}}^N(v_{\pi_k}(s)) = v_{\pi_{k+1}}(s) \quad \text{--- (4)}$$

hence, using eqn (3) & (4)

$$v_{\pi_{k+1}}(s) \geq v_{\pi_k}(s)$$

Hence proved.

for proving 2) .

Suppose $V_{\pi_{k+1}}(s)$

From policy improvement if we get .

$$T_{\pi_{k+1}} V_{\pi_k}(s) = V_{\pi_k}(s) \quad \text{--- (5)}$$

It is guaranteed that $T_{\pi_{k+1}}$ has a unique fixed point $V_{\pi_{k+1}}$

thus ,

$$V_{\pi_k}(s) = V_{\pi_{k+1}}(s) \quad \forall s$$

Also $T_{\pi_{k+1}} V_{\pi_k}(s) = T V_{\pi_k}(s) \quad \text{--- (6)}$

(policy improvement step)

Thus from (5) & (6), ~~we have~~

$$V_{\pi_k}(s) = T V_{\pi_k}(s)$$

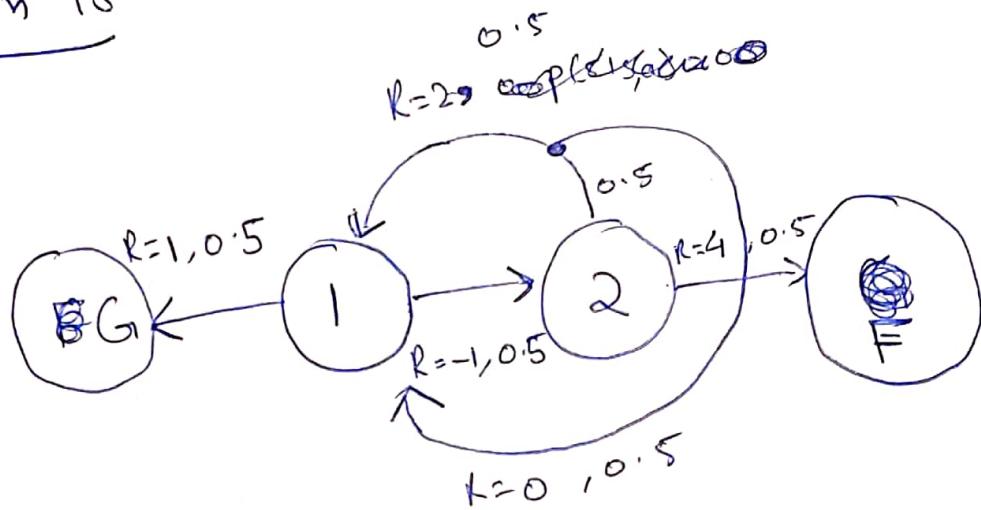
This means that ,

$$V_{\pi_k}(s) = \max A_t \mathbb{E}[r_{t+1} + \gamma V_{\pi_k}(s_{t+1}) \mid s_t = s, A_t = a]$$

This implies that V_{π_k} is the solution of the Bellman optimality equations .

Thus $\boxed{V_* = V_{\pi_k} = V_{\pi_{k+1}}}$

Question 15



s	a	s'	$p(s' s,a)$	$\pi(s,a,s')$	$\rho(s',r s,a)$
1	des	BG	0.5	1	1
1	asc	2	0.5	-1	1
2	des	1	0.5	2	0.5
2	des	1	0.5	0	0.5
2	asc	F	0.5	+4	1

Note: We only do Evaluation

using ~~values~~ at the previous time step values in accordance with generalized policy iteration as it says that even though we choose $M=1$ (the lazy $v_{\pi_K}^{(s)}$) for $v_{\pi_K}^{(s)} = T_{\pi_K}^M(v_{\pi_K}^{(s)})$ eventually this also converges to v^* .

Policy Iteration

$$V_0(1) = -0^2 e^{-2}(0+1) + 2e^{-2}(0+0)$$

$$V_0(2) = 0$$

$$V_F(0) = 0, V_F(F) = 0$$

1st ~~Iteration~~ Iteration

Evaluation

$$V_1(1) = 0.5 \left[(1 + V_0(0)) \cancel{0} \right] + (-1 + V_0(0)) \cancel{0}$$

$$\Rightarrow 0.5 \left[\cancel{0} + 1 - 1 \right]$$

$$= 0$$

$$V_1(2) = 0.5 \left[(2 + V_0(0)) \cancel{0.5} + (0 + V_0(0)) \cancel{0.5} + (4 + V_0(0)) \cancel{1} \right]$$

$$\Rightarrow 0.5 \left[1 + 4 \right]$$

$$= 2.5$$

Improvement:

~~$\pi_0(1) = [0.5, 0.5]$~~

$$\pi_1(1) = \arg\max \left[(1 + V_0(0)), -1 + 2.5 \right]$$

$$\pi_1(1) = \arg\max \left[1, 1.5 \right] \text{ (Action = acc)}$$

$$\Rightarrow 1 \quad (\text{action = acc})$$

$$\pi_1(1) = [1, 0]$$

$$\pi_0(2) = [0.5, 0.5]$$

$$\begin{aligned}\pi_1(2) &= \text{argmax} [(2+0)^{0.5} + (0+0)^{0.5}, (4+0)] \\ &= \text{argmax} [1, 4] \\ &= 1 \quad (\text{action} = \text{asc})\end{aligned}$$

$$\boxed{\pi_1(2) = [1, 0]}$$

2nd iteration

Evaluation

$$\begin{aligned}v_2(1) &= 0 + 1[-1 + 2.5] \\ &= +1.5\end{aligned}$$

$$\begin{aligned}v_2(2) &= 0 + 1[4 + 0] \\ &= 4\end{aligned}$$

Improvement

$$\pi_1(1) = [1, 0]$$

$$\begin{aligned}\pi_2(1) &= \text{argmax} [(1+0)^{0.5}, (-1+4)] \\ &= \text{argmax} [1, 3]\end{aligned}$$

$$= 1 \quad (\text{action} = \text{asc})$$

$$\boxed{\pi_2(1) = [1, 0]}$$

$$\boxed{\pi_2(2) = [1, 0]}$$

$$\begin{aligned}\pi_2(2) &= \text{argmax} [(2+1.5)^{0.5} + (0+1.5)^{0.5}, 4] \\ &= \text{argmax} [1.75 + 0.75, 4]\end{aligned}$$

= 1 (action = asc)

$$\pi_2(2) = [1, 0]$$

Now since $\pi_1 = \pi_2$, thus we have
converged to the optimal policy. Thus
policy iteration gets terminated.

Optimal policy = $\pi^*(1) = \text{ascend}$
 $\pi^*(2) = \text{ascend}$.

* ALSO as n increases, number of iterations
increases, increases 