به نام خداوند بخشنده مهربان

هوش مصنوعی

دانشکده مهندسی کامپیوتر

دکتر رهبان
بهار ۱۴۰۳

مهدی علی نژاد، ۴۰۱۱۰۶۲۶۶

تمرین تئوری ۵

سوال ۱

۱. (۲۰ نمره، درجه سختی ۵) درستی یا نادرستی جملات زیر را با ذکر دلیل مشخص کنید.

الف) در الگوریتم Value Iteration در صورتی که، سیاست بدست آمده برای یک استیت در T مرحله تغییر نکند، سیاست بدست آمده در این T مرحله همان سیاست بهینه خواهد بود.(به طور دقیق‌تر اگر داشته باشیم $\pi_k = \pi_{k+1} = ... = \pi_{k+T}$، به ازای یک T بزرگ، آنگاه π_{k+i}به ازای هر i همان سیاست بهینه است.)

ب) Q-learning تنها زمانی مقادیر بهینه Q-values را یاد می‌گیرد که کنش‌هایی (actions) که در نهایت انتخاب می‌شود، براساس سیاست(policy) بهینه باشد.

ج) در یک MDP قطعی (یعنی به ازای هر state و action به یک state مشخص بصورت قطعی می‌رویم.) الگوریتم Q-learning با استفاده از نرخ یادگیری(learning rate) $\alpha = 1$ در هر بروزرسانی، به طور صحیح مقدارهای بهینه Q-values را یاد می‌گیرد.

د) اگر بدانیم که $|S| >> |A|$ آنگاه پیچیدگی زمانی اجرای هر ایتریشن در policy iteration و value iteration با هم برابر میشود.

ه) در یک MDP با حالات محدود و پاداش با باند مشخص و $\gamma < ۱$ در صورتی همه پاداش‌ها با عدد ثابت c جمع شود، سیاست بهینه تغییر نمی‌کند.

(آ) درست، در الگوریتم Value Iteration در هر مرحله، مسیر بهینه را مقداری بهبود می دهیم، هنگامی که دیگر پس از گدشت زمان زیادی بهبودی رخ ندهد به این معنی می باشد که به الگوریتم بهینه رسیده ایم چون از قبل می دانیم که این الگوریتم در صورت وجود یک استراتژی بهینه به آن همگرا می شود.

(ب) نادرست، یک ویژگی خیلی خوب این الگوریتم دقیقا همین است که با اکشن های نه لزوما بهینه می تواند به سیاست بهینه همگرا شود.

(ج) درست، زیرا با قطعی بودن MDP از هر استیت، با یک اکشن خاص تنها به یک استیت خاص می رویم و دیگر نیاز نیست با تکرار این عملکرد سعی کنیم باقی احتمالات را نیز بدست آوریم، به عبارتی دیگر، $Q(s, a) = sample$ که $sample$ همان حاصل انجام اکشن a در استیت s است که می دانیم به طور قطعی و تنها برابر یک چیز است.

(د) درست، پیچیدگی زمانی policy iteration در هر اجرا برابر $S^۲$ و پیچیدگی زمانی value iteration در هر اجرا برابر $S^۲A$ است که در صورت کوچک بودن A نسبت به S می توان از آن چشم پوشی کرد.

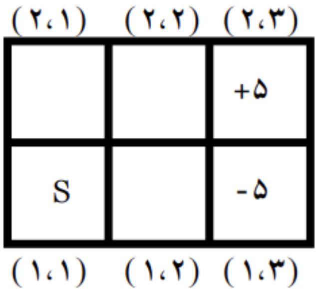
(ه) با فرض نبود terminal state درست، سیاست بهینه در MDP سیاستی است که ما را به بیشترین مجموع reward ها برساند، و با این کار به Value هر استیت مقدار $\frac{c}{۱-\gamma}$ را اضافه کرده ایم. در غیر این صورت، مثال نقض زیر ارائه می شود.

۹۰	۱۰	start	۲۰

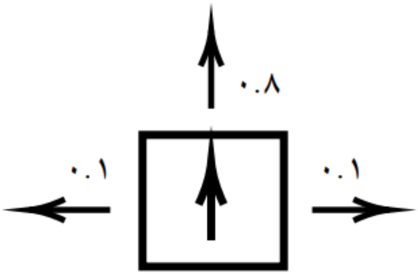
در این حالت سیاست بهینه با $\gamma = ۰/۱$ و ترمینال استیت های ۹۰ و ۲۰ حرکت به سمت چپ حرکت بهینه است ولی اگر ریوارد ها را با ۲۰ جمع کنیم سیاست بهینه حرکت به سمت چپ می شود.

سوال ۲

۲. (۱۵ نمره، درجه سختی ۶) در شکل زیر یک محرک قرار دارد که همواره از خانه (۱, ۱) که با S نشان داده شده است، شروع به حرکت می‌کند. دو تا از خانه‌های جدول خانه‌های پایانی هستند، خانه (۲, ۳) با جایزه به مقدار ۵+ و خانه (۱, ۳) به مقدار ۵−. در خانه‌هایی که پایانی نیستند، جایزه‌ای وجود ندارد.(جایزه برای یک خانه در صورتی دریافت می‌شود که محرک به آن خانه برود). تابع احتمال برای حرکت از هر خانه به خانه‌های مجاور به این صورت است که: برای هر حرکت به سمت بالا، پایین، چپ و راست با احتمال ۰/۸ این حرکت انجام می‌شود و با احتمال ۰/۱ حرکتی عمود بر حرکت در نظر گرفته شده انجام می‌شود. اگر محرک با دیوار برخورد کند، محرک در همان خانه‌ای که قرار داشته است می‌ماند.



شکل مربوط به سوال



تابع احتمال حرکت

الف) فرض کنید محرک احتمال‌هایی که برای حرکت‌ها وجود دارد را می‌داند. سه مرحله‌ی اولیه الگوریتم Value Iteration را برای هر وضعیت(خانه) انجام دهید. فرض $\gamma = ۰/۹$ و $V_۰ = ۰$ برای هر وضعیتی در نظر بگیرید.

ب) فرض کنید محرک احتمال‌های مربوط به حرکت را نمی‌داند. چه کاری باید انجام دهد تا سیاست بهینه را یاد بگیرد؟

ج) با استفاده از $\alpha = ۰/۱$ و مقدارهای اولیه صفر، آپدیت‌های Temporal Difference-Learning را بعد از تجربه‌ی (۱, ۳) − (۱, ۲) − (۱, ۱) و (۲, ۳) − (۲, ۲) − (۱, ۲) − (۱, ۱) برای $V(s)$ ها بنویسید($\gamma = ۰/۹$)
(توجه داشته باشید که در هر یک از مسیرهای گفته شده، محرک به ترتیب از چپ به راست خانه‌ها را طی کرده‌است.)

(آ) t = 0:

0	0	0
0	0	0

t = 1:

$$V_1(s) = \max_a \sum_{s'} T(s,a,s')\big(R(s,a,s') + \gamma V_0(s')\big) = \max_a \sum_{s'} R(s')T(s,a,s')$$

0	4	0
0	0	0

t = 2:

$$V_2(s) = \max_a \sum_{s'} T(s,a,s')\big(R(s,a,s') + \gamma V_1(s')\big)$$

$$V_2((2,2)) = \max \begin{bmatrix} 0.1(5+0.9\times 0)+0.8(0+0.9\times 4)+0.1(0+0) \\ 0.1(0+0.9\times 4)+0.8(5+0.9\times 0)+0.1(0+0) \\ 0.1(0+0)+0.8(0+0)+0.1(5+0.9\times 0) \\ 0.1(0+0)+0.8(0+0)+0.1(0+0.9\times 4) \end{bmatrix} = 4.36$$

$$V_2((2,1)) = \max \begin{bmatrix} 0.1(0+0.9\times 0)+0.8(0+0.9\times 0)+0.1(0+0.9\times 4) \\ 0.1(0+0.9\times 0)+0.8(0+0.9\times 4)+0.1(0+0) \\ 0.1(0+0)+0.8(0+0)+0.1(0+0.9\times 4) \\ 0.1(0+0)+0.8(0+0)+0.1(0+0.9\times 0) \end{bmatrix} = 2.88$$

$$V_2((1,2)) = \max \begin{bmatrix} 0.1(0+0.9\times 0)+0.8(0+0.9\times 4)+0.1(-5+0.9\times 0) \\ 0.1(0+0.9\times 4)+0.8(-5+0.9\times 0)+0.1(0+0) \\ 0.1(0+0)+0.8(0+0)+0.1(-5+0.9\times 0) \\ 0.1(0+0)+0.8(0+0)+0.1(0+0.9\times 4) \end{bmatrix} = 2.38$$

2.88	4.36	0
0	2.38	0

t = 3:

$$V_2(s) = \max_a \sum_{s'} T(s,a,s')\big(R(s,a,s') + \gamma V_1(s')\big)$$

$$V_2((2,2)) = \max \begin{bmatrix} 0.1(5+0.9\times 0)+0.8(0+0.9\times 4.36)+0.1(0+0.9\times 2.88) \\ 0.1(0+0.9\times 4.36)+0.8(5+0.9\times 0)+0.1(0+0.9\times 2.38) \\ 0.1(0+0.9\times 2.88)+0.8(0+0.9\times 2.38)+0.1(5+0.9\times 0) \\ 0.1(0+0.9\times 2.38)+0.8(0+0.9\times 2.88)+0.1(0+0.9\times 4.36) \end{bmatrix} = 4.61$$

$$V_2((2,1)) = \max \begin{bmatrix} 0.1(0+0.9\times 2.88)+0.8(0+0.9\times 2.88)+0.1(0+0.9\times 4.36) \\ 0.1(0+0.9\times 2.88)+0.8(0+0.9\times 4.36)+0.1(0+0) \\ 0.1(0+0.9\times 2.88)+0.8(0+0)+0.1(0+0.9\times 4.36) \\ 0.1(0+0.9\times 2.88)+0.8(0+0.9\times 2.88)+0.1(0+0.9\times 0) \end{bmatrix} = 3.40$$

$$V_2((1,2)) = \max \begin{bmatrix} 0.1(0+0.9\times 0)+0.8(0+0.9\times 4.36)+0.1(-5+0.9\times 0) \\ 0.1(0+0.9\times 4)+0.8(-5+0.9\times 0)+0.1(0+0.9\times 2.38) \\ 0.1(0+0)+0.8(0+0.9\times 4.36)+0.1(-5+0.9\times 0) \\ 0.1(0+0.9\times 2.38)+0.8(0+0)+0.1(0+0.9\times 4.36) \end{bmatrix} = 2.38$$

$$V_2((1,1)) = \max \begin{bmatrix} 0.1(0+0.9\times 0)+0.8(0+0.9\times 2.88)+0.1(0+0.9\times 2.38) \\ 0.1(0+0.9\times 2.88)+0.8(0+0.9\times 2.38)+0.1(0+0) \\ 0.1(0+0)+0.8(0+0)+0.1(0+0.9\times 2.38) \\ 0.1(0+0)+0.8(0+0)+0.1(0+0.9\times 2.88) \end{bmatrix} = 2.29$$

3.40	4.61	0
2.29	2.64	0

(ب) در این حالت نیاز است تا با انجام آزمون و خطا MDP خود را تخمین بزنیم، به این صورت که agent را در محیط قرار دهیم و هر حرکت از s به s' با اکشن a را یک نمونه در نظر بگیریم و در انتها، احتمالات را از روی این نمونه ها بدست آوریم.

(ج) .

(1, 1) -> (1, 2) -> (1, 3)

$$V(1,2) = (1-0.1)(0) + 0.1\Big[-5\Big] = -0.5$$

0	0	0
0	-0.5	0

(1, 1) -> (1, 2) -> (2, 2) -> (2,3)

$$V(1,1) = (1-0.1)(0) + 0.1\Big[0+0.9\times(-0.5)\Big] = -0.045$$

$$V(1,2) = (1-0.1)(-0.5) + 0.1\Big[0+0\Big] = -0.4$$

$$V(2,2) = (1-0.1)(0) + 0.1\Big[5+0.9\times(0)\Big] = 0.5$$

0	0.5	0
-0.045	-0.45	0

سوال ۳

۳. (۱۵ نمره، درجه سختی ۶)

(آ) توضیح دهید که الگوریتم Q-Learning چگونه در محیط grid world ۴ × ۴ کار می‌کند، جایی که عامل از موقعیت (۰, ۰) شروع می‌کند و هدف در (۳, ۳) است. نحوه استفاده از قاعده Bellman در update Q-Learning را شرح دهید.

(ب) با توجه به شرایط زیر در grid world ۴ × ۴، مقادیر Q را برای دو iteration به صورت دستی محاسبه کنید:

- عامل از (۰, ۰) شروع می‌کند و هدف در (۳, ۳) است.
- پاداش‌ها: ۱− برای حالات غیرنهایی، ۱۰+ برای رسیدن به هدف.
- اعمال: بالا، پایین، چپ، راست.
- ضریب تخفیف (gamma): ۰/۹.
- نرخ یادگیری (alpha): ۰/۱.
- عامل در این iterationها به طور تصادفی اقدامات را انتخاب می‌کند.

محاسبات به‌روزرسانی مقادیر Q را برای هر گامی که عامل برمی‌دارد تا به هدف برسد برای دو iteration نشان دهید.

(ج) در زمینه استراتژی epsilon-greedy در Q-Learning، نقش‌های اکتشاف و استفاده از اطلاعات را توضیح دهید. چگونه باید epsilon را بر اساس زمان تنظیم کرد و چرا؟

(آ) این الگوریتم با حرکت در محیط، مقادیر Q(s، a) را به دست می آورد. در یک جدول ۴ در ۴ نیز اتفاقی که می افتد این است که agent ما با حرکت در این محیط و مشاهده ی reward ها و احتمالات، این Q ها را محاسبه می کند. قاعده ی آپدیت نیز به این صورت است که با استفاده از sample مشاهده شده، با یه نسبتی که معمولا کوچک است، این sample را در Q آن خانه تاثیر می دهیم و در این آپدیت، مقدار sample خود برگرفته از معادلات بلمن حساب می شود.

(ب) همانطور که در حل تمرین گفته شده بود، کافیست برای این قسمت، اقدامات را حرکت به سمت راست و سپس حرکت به بالا در نظر بگیریم و نیازی نیست واقعا تصادفی باشد. مقدار Q جدید برای اکثر خانه های این مسیر یکسان است زیرا مقادیر اولیه و ریوارد های حرکت یکسان است و در کل دو مقدار متفاوت می بینیم.

Q-values:

0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0

iteration one

$$Q((3,2),up) = 0.9 \times 0 + 0.1 \times (R(s,a,s') + \gamma \max_{a'} Q(s',a'))$$
$$= 0.1 \times (10 + 0.9 \times 0) = 1$$
$$otherwise \quad Q(s,a) = 0.9 \times 0 + 0.1 \times (R(s,a,s') + \gamma \max_{a'} Q(s',a'))$$
$$= 0.1 \times (-1 + 0.9 \times 0) = -0.1$$

Q-values:

0	0	0	10
0	0	0	1
0	0	0	-0.1
-0.1	-0.1	-0.1	-0.1

iteration two

$$
\begin{aligned}
Q((3,2),up) &= 0.9\times 1 + 0.1\times(R(s,a,s') + \gamma\max_{a'} Q(s',a')) \\
&= 0.9 + 0.1\times(10 + 0.9\times 0) = 1.9 \\
Q((3,1),up) &= 0.9\times -0.1 + 0.1\times(R(s,a,s') + \gamma\max_{a'} Q(s',a')) \\
&= 0.1\times(-1 + 0.9\times 1) = -0.1 \\
otherwise \quad Q(s,a) &= 0.9\times -0.1 + 0.1\times(R(s,a,s') + \gamma\max_{a'} Q(s',a')) \\
&= 0.1\times(-1 + 0.9\times -1) = -0.199
\end{aligned}
$$

Q-values:

0	0	0	10
0	0	0	1.9
0	0	0	-0.1
-0.199	-0.199	-0.199	-0.199

حواسمان باشد که این مقادیر Q نوشته شده، فقط مقادیر Q با فرض بر انجام اکشن در جهت حرکت است. و باقی Q ها که برای باقی جهات است، صفر است.

(ج) برای یک محیطی که می خواهیم agent ما بتواند reward هایی نیز جمع آوری کند، باید یک توازنی بین امتحان کردن چیز های جدید و بهره وری از چیز های یاد گرفته شده برقرار کرد، این کار را می توان با الگوریتم epsilon-greedy انجام داد به این صورت که با احتمال epsilon حرکتی در راستای یادگیری چیز های جدید انجام می دهیم و در غیر این صورت از چیز های یاد گرفته شده برای بهره وری استفاده می کنیم. همچنین epsilon نیز با گذر زمان کاهش می یابد، یک حالت کاهش آن به این صورت است.

$$\epsilon = \frac{۱}{\log(t + ۱/۰۰۰۰۰۱)}$$

سوال ۴

۴. (۲۰ نمره، درجه سختی ۸) یک MDP متناهی مانند $M = (S, A, T, R, \gamma)$ در نظر بگیرید، به طوری که S فضای وضعیت، A فضای عمل، T احتمالات انتقال، R تابع پاداش و γ ضریب تخفیف (discount factor) است. Q^* را به صورت $Q^*(s,a) = Q_{\pi^*}(s,a)$ تعریف کنید بطوریکه π^* سیاست بهینه است. فرض کنید ما یک تخمین $\tilde{Q}$ از Q^* داشته باشیم، و $\tilde{Q}$ بوسیله نرم l_∞ با توجه به تعریف زیر محدود شده است:

$$||\tilde{Q} - Q^*||_\infty \leq \varepsilon$$

که در آن $||x||_\infty = max_{s,a}|x(s,a)|$ است.
فرض کنید ما به دنبال سیاست حریصانه برای $\tilde{Q}$ هستیم، $\pi(s) = argmax_{a\in A}\tilde{Q}(s,a)$. می‌خواهیم نشان دهیم که عبارت زیر برقرار است:

$$V_\pi(s) \geq V^*(s) - \frac{۲\varepsilon}{۱-\gamma}$$

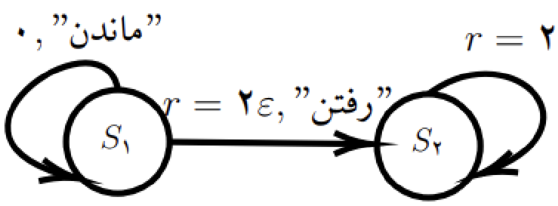
که در آن $V_\pi(s)$ تابع ارزش سیاست حریصانه π است و $V^*(s) = max_{a\in A}Q^*(s,a)$ تابع ارزش بهینه است. این نشان می‌دهد که اگر ما یک تابع ارزش state-action تقریباً بهینه را محاسبه کنیم و سپس سیاست حریصانه را برای آن تابع ارزش state-action تقریبی استخراج کنیم، سیاست نتیجه‌گرفته همچنان در MDP واقعی خوب عمل می‌کند. حال با کمک دو قسمت ابتدایی سوال زیر عبارت گفته شده را اثبات کنید.

الف) فرض کنید π^* سیاست بهینه باشد، V^* تابع ارزش بهینه و همانطور که در بالا تعریف شده است $\pi(s) = \arg\max_{a\in A} Q(s,a)$. نشان دهید نامساوی زیر برای تمام وضعیت‌های $s \in S$ برقرار است.

$$V^*(s) - Q^*(s, \pi(s)) \leq ۲\varepsilon$$

ب) با استفاده از نتیجه قسمت قبل اثبات کنید: $V_\pi(s) \geq V^*(s) - \frac{۲\varepsilon}{۱-\gamma}$

اکنون نشان می‌دهیم که حالت تساوی این نامساوی برقرار است. یک MDP دو وضعیتی را که در شکل زیر نشان داده شده است، در نظر بگیرید. وضعیت $s_۱$ دو عمل دارد، "ماندن" که با پاداش ۰ به خودش منتقل می‌شود و "رفتن" که به وضعیت $s_۲$ با پاداش ۲ε منتقل می‌شود. وضعیت $s_۲$ نیز به خودش با پاداش ۲ε منتقل می‌شود.



ج) مقدار ارزش بهینه $V^*(s)$ برای هر وضعیت و $Q^*(s,a)$ را برای وضعیت $s_۱$ و هر عمل محاسبه کنید.

د) نشان دهید که یک تابع ارزش state action تقریبی $\tilde{Q}$ با خطا ε (با استفاده از نرم l_∞) وجود دارد، به طوری که $V_\pi(s_۱) - V^*(s_۱) = -\frac{۲\epsilon}{۱-\gamma}$، که در آن $\pi(s) = \arg\max_{a\in A}\tilde{Q}(s,a)$ است.

(آ)

$$
\begin{aligned}
&V^* - Q^*(s,\pi(s)) \le 2\epsilon \to \max_{a\in A} Q^*(s,a) - Q^*(s, \arg\max_{a\in A} \tilde{Q}(s,a)) \le 2\epsilon \\
&||\tilde{Q} - Q^*||_\infty \le \epsilon \to |\tilde{Q}[i] - Q^*[i]| \le \epsilon \to \tilde{Q}[i] - \epsilon \le Q^*[i] \le \tilde{Q}[i] + \epsilon \quad \text{for } i \le \dim Q^* \\
&\max_{a\in A} \tilde{Q}(s,a) - \epsilon \le Q^*(s, \arg\max_{a\in A} \tilde{Q}(s,a)) \le \max_{a\in A} \tilde{Q}(s,a) + \epsilon \\
&\tilde{Q}(s, \max_{a\in A} Q^*(s,a)) - \epsilon \le \max_{a\in A} Q^*(s,a) \le \tilde{Q}(s, \max_{a\in A} Q^*(s,a)) + \epsilon \\
&\tilde{Q}(s, \max_{a\in A} Q^*(s,a)) \le \max_{a\in A} \tilde{Q}(s,a) \xrightarrow{\text{based on the two above lines}} \max_{a\in A} Q^*(s,a) - Q^*(s, \arg\max_{a\in A} \tilde{Q}(s,a)) \le 2\epsilon
\end{aligned}
$$

توضیحی در مورد خط آخر: با توجه به بازه های تعیین شده برای هرکدام از مقادیر، طبق علت خط آخر، حداکثر مقداری که ماکسیمم Q^* می تواند به خود بگیرد از مینیمم مقداری که $Q^*(s, \arg\max_{a\in A} \tilde{Q}(s,a))$ خواهد داشت، می توانند حداکثر ۲ϵ فاصله داشته باشند. فقط کافیست به سر و ته بازه ها نگاه کنیم.

(ب) V_i^π و Q_i^π را به صورت زیر تعریف می کنیم.

$$
\begin{aligned}
&V_0^\pi(s) = V^*(s), \quad Q_0^\pi(s,a) = Q^*(s,a) \\
&V_i^\pi(s) = Q_i^\pi(s,\pi(s)), \quad Q_i^\pi(s,a) = \sum_{s'} T(s,a,s')\Big(R(s,a,s') + \gamma V_{i-1}^\pi(s)\Big) \\
&\lim_{i\to\infty} V_i^\pi = V^\pi, \quad Q_i^\pi = Q^\pi \\
&\text{to prove } V^*(s) - V_i^\pi(s) \le \frac{1-\gamma^{i+1}}{1-\gamma} 2\epsilon \text{ we use induction} \\
&i = 0 : V^*(s) - V_0^\pi(s) = 0 \le \frac{2\epsilon}{1-\gamma} \text{since } \gamma < 1 \text{ and } \epsilon > 0 \\
&V^*(s) - V_{i+1}^\pi(s) = V^* - \sum_{s'} T(s,\pi(s),s')\Big(R(s,\pi(s),s') + \gamma V_i^\pi(s')\Big) \\
&\xrightarrow{\text{induction}} \le V^*(s) - \sum_{s'} T(s,\pi(s),s')\Big(R(s,\pi(s),s') + \gamma(V^*(s') - \frac{1-\gamma^{i+1}}{1-\gamma}2\epsilon\Big) \\
&\quad = V^*(s) - Q^*(s,\pi(s)) + 2\epsilon\gamma\frac{1-\gamma^{i+1}}{1-\gamma} \\
&\xrightarrow{\text{previous section}} \le 2\epsilon + 2\epsilon\gamma\frac{1-\gamma^{i+1}}{1-\gamma} = 2\epsilon(1 + \gamma\frac{1-\gamma^{i+1}}{1-\gamma}) = 2\epsilon\frac{1-\gamma^{i+2}}{1-\gamma} \to \text{induction proven} \\
&\lim_{i\to\infty} V^*(s) - V_i^\pi(s) \le \frac{1-\gamma^{i+1}}{1-\gamma}2\epsilon \xrightarrow{0\le\gamma<1} V^*(s) - V^\pi(s) \le \frac{2\epsilon}{1-\gamma}
\end{aligned}
$$

(ج)

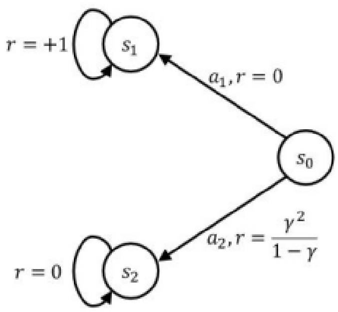
$$
\begin{aligned}
&V^*(s_2) = \max_a \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma V^*(s')] = 2\epsilon + \gamma V^*(s_2) \to V^*(s_2) = \frac{2\epsilon}{1-\gamma} = Q(s2) \\
&V^*(s_1) = \max_a \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma V^*(s')] = \max_a \begin{bmatrix} 0 + \gamma V^*(s_1) \\ 2\epsilon + \gamma V^*(s_2) \end{bmatrix} \\
&\quad = \max_a \begin{bmatrix} \gamma V^*(s_1) \\ 2\epsilon + \frac{2\gamma\epsilon}{1-\gamma} \end{bmatrix} \xrightarrow{\text{if first element becomes chosen, it would mean that } \gamma = 1 \text{ which is not acceptable}} V^* = 2\epsilon + \frac{2\gamma\epsilon}{1-\gamma} \to V^*(s_1) = \frac{2\epsilon}{1-\gamma} \\
&Q(s_1, \text{'stay'}) = \gamma V^*(s_1) = \frac{2\gamma\epsilon}{1-\gamma} \\
&Q(s_1, \text{'leave'}) = 2\epsilon + \frac{2\gamma\epsilon}{1-\gamma}
\end{aligned}
$$

(د)

$$
\begin{aligned}
&\tilde{Q}(s_1, \text{'leave'}) = 0, \quad \tilde{Q}(s_1, \text{'stay'}) = \frac{2\gamma\epsilon}{1-\gamma} \to \pi(s_1) = \text{'stay'} \\
&V^\pi(s_1) = 0 + V^\pi(s_1) \times \gamma = 0 \to V^*(s_1) - V^\pi(s_1) = \frac{2\epsilon}{1-\gamma}
\end{aligned}
$$

سوال ۵

۵. (۲۰ نمره، درجه سختی ۸) در این مسئله، یک مثال برای محدود کردن تعداد گام‌های لازم برای یافتن سیاست بهینه با استفاده از Value Iteration مشاهده می‌کنید. یک MDP با ضریب تخفیف (discount factor) $\gamma < 1$ که در شکل زیر نشان داده شده است را در نظر بگیرید. این MDP شامل ۳ وضعیت است، و پاداش‌ها به محض انجام یک عمل از وضعیت داده می‌شود. در وضعیت s_0، عمل a_1 دارای پاداش فوری صفر است و باعث یک انتقال قطعی به وضعیت s_1 می‌شود که پاداش $+1$ برای هر گام بعدی دارد (بدون توجه به عمل). از وضعیت s_0، عمل a_2 باعث یک انتقال قطعی به وضعیت s_2 با پاداش فوری $\frac{\gamma^2}{(1-\gamma)}$ می‌شود، اما وضعیت s_2 برای هر گام بعدی (بدون توجه به عمل) پاداش صفر دارد.



شکل ۱: MDP با سه وضعیت

الف) مجموع پاداش تخفیف داده شده $(\sum_{t=0}^{\infty} \gamma^t r_t)$ با انجام عمل a_1 از وضعیت s_0 در مرحله زمانی $t = 0$ چقدر است؟

ب) مجموع پاداش تخفیف داده شده $(\sum_{t=0}^{\infty} \gamma^t r_t)$ با انجام عمل a_2 از وضعیت s_0 در مرحله زمانی $t = 0$ چقدر است؟
عمل بهینه در این وضعیت کدام است؟

ج) فرض کنید هر وضعیت را صفر مقداردهی اولیه کنیم.
(یعنی در مرحله $n = 0$، $\forall s : V_{n=0}(s) = 0$). نشان دهید که الگوریتم Value Iteration تا زمانی ادامه می‌دهد که عمل sub-optimal را در n^* مرحله پیدا کند به صورتیکه:

$$n^* \geq \frac{\log(1-\gamma)}{\log \gamma} \geq \frac{1}{2}\log\left(\frac{1}{1-\gamma}\right)\frac{1}{1-\gamma}.$$

بنابراین، زمان الگوریتم Value Iteration سریع‌تر از $\frac{1}{(1-\gamma)}$ رشد می‌کند. (شما فقط باید درستی نامساوی اول را نشان دهید)

(آ)

$$\sum_{t=0}^{\infty} \gamma^t r_t \xrightarrow{r_0=0} \sum_{t=1}^{\infty} \gamma^t = \frac{\gamma}{1-\gamma}$$

(ب)

$$\sum_{t=0}^{\infty} \gamma^t r_t \xrightarrow{r_0=\frac{\gamma^2}{1-\gamma} \& r_{i!=0}=0} = \frac{\gamma^2}{1-\gamma}$$

چون گاما کمتر از ۱ است، پس داریم $\gamma^2 \leq \gamma$ پس انتخاب a_1 بهینه است.

(ج) در مرحله ی $n = ۰$ مقدار اولیه ی همه ی استیت ها را صفر در نظر بگیریم خواهیم داشت

$$
\begin{aligned}
&V_1(S_0) = \frac{\gamma^2}{1-\gamma} + \gamma \times 0 = \frac{\gamma^2}{1-\gamma} \\
&V_1(S_1) = 1 + \gamma \times 0 = 1 \\
&V_1(S_2) = 0 + \gamma \times 0 = 0 \\
&V_{k+1}(S_0) = \max(0 + \gamma * V_k(S_1), \frac{\gamma^2}{1-\gamma} + \gamma \times V_k(S_2)) \\
&V_{k+1}(S_1) = 1 + \gamma \times V_k(S_1) = 1 + \gamma + \gamma^2 + \cdots + \gamma^k \\
&V_{k+1}(S_2) = 0 + \gamma V_k(s_2) \times 0 = 0
\end{aligned}
$$

برای اینکه پس از n^* مرحله سیاست بهینه که رفتن به $a_۱$ است را انتخاب کند باید نامساوی زیر برقرار شود.

$$
\begin{aligned}
&\gamma \times V_k(s_1) \geq \frac{\gamma^2}{1-\gamma} \to \gamma \times (1 + \gamma + \gamma^2 + \cdots + \gamma^{n^*-1}) \geq \frac{\gamma^2}{1-\gamma} \\
&\to \frac{1-\gamma^{n^*}}{1-\gamma} \geq \frac{\gamma}{1-\gamma} \to 1 - \gamma^{n^*} \geq \gamma \to 1 - \gamma \geq \gamma^{n^*} \\
&\to \log(1-\gamma) \geq n^* \log\gamma \xrightarrow{\log\gamma < 0} n^* \geq \frac{\log(1-\gamma)}{\log\gamma}
\end{aligned}
$$