تمرین ۴ **ایمان محمدی ـ ۹۹۱۰۲۲۰۷**

نیم‌سال دوم ۱۴۰۱ـ۱۴۰۲ **زمان آپلود: ۳۱ اردیبهشت**

سوال ۱.

الف

روش Value Iteration یکی از الگوریتم‌های حل مسئله در یادگیری تقویتی است که برای حل مسئله‌های مارکوف تصمیم‌گیری (MDP) استفاده می‌شود. این الگوریتم به صورت تکراری و تمامیتی عمل می‌کند و به طور تدریجی مقادیر ارزش برای هر حالت را به‌روزرسانی می‌کند تا به یک تابع ارزش بهینه برسد.

برای استفاده از الگوریتم Value Iteration ، نیازمند تعریف MDP هستیم. MDP توسط چهار مؤلفه اصلی تعریف می‌شود:

مجموعه حالات (States): مجموعه‌ای از حالات قابل مشاهده در مسئله را نشان می‌دهد.

مجموعه عمل‌ها (Actions): مجموعه‌ای از عمل‌های قابل انجام در هر حالت را نشان می‌دهد.

تابع انتقال (Transition Function): تابعی است که نشان می‌دهد با انجام یک عمل در یک حالت، ما به چه حالتی خواهیم رفت و با چه احتمالی.

تابع پاداش (Reward Function): تابعی است که برای هر حالت و عمل، پاداش مشخص می‌کند.

هدف Value Iteration یافتن تابع ارزش بهینه است که برای هر حالت، ارزش بهینه انتظاری را نشان می‌دهد. الگوریتم Value Iteration به صورت تکراری عمل می‌کند و به تدریج مقادیر ارزش را به‌روزرسانی می‌کند تا به تابع ارزش بهینه برسد.

حالا که تمام حالات ۰ هستند، تا عمق ۲ را محاسبه می‌کنیم. (همچنین از نوشتن یک سری محاسبات عددی چشم‌پوشی می‌کنیم چون محاسبات بسیار زیادی دارد این سوال.)

$$V(s) = \max\left[\sum p(s'|s,a)\cdot(R(s,a,s')+\gamma\cdot V(s'))\right]$$

پس داریم:

$$V_۱(S_۱) = max[۱\times[-۱+V_۰(S_۲)], ۱\times[-۱+V_۰(S_۶)], ۰/۶\times(۳+V_{goal}) + ۰/۴\times(-۱۰+V_{fail})]$$

$$\rightarrow V_۱(S_۱) = max[-۱,-۱,-۲/۲] = -۱$$

به همین شکل برای باقی نیز خواهیم داشت:

$$\rightarrow V_۱(S_۲) = max[-۱,-۱,-۱,-۱] = -۱$$

$$\rightarrow V_۱(S_۳) = max[-۱,-۱,-۱,۰/۸] = ۰/۸$$

$$\rightarrow V_{۱}(S_{۴}) = max[-۱, -۱, -۱, -۱] = -۱$$

$$\rightarrow V_{۱}(S_{۵}) = max[-۱, -۱, -۲/۲] = -۱$$

$$\rightarrow V_{۱}(S_{۶}) = max[-۱, -۱, -۲/۲] = -۱$$

$$\rightarrow V_{۱}(S_{۷}) = max[-۱, -۱, -۱, -۱] = -۱$$

$$\rightarrow V_{۱}(S_{۸}) = max[-۱, -۱, -۱, -۱] = -۱$$

$$\rightarrow V_{۱}(S_{۹}) = max[-۱, -۱, -۱, -۱] = -۱$$

$$\rightarrow V_{۱}(S_{۱۰}) = max[-۱, -۱, -۲/۲] = -۱$$

حالا یک عمق دیگر پیش می‌رویم:

$$\rightarrow V_{۲}(S_{۱}) = max[-۲, -۲, -۲/۲] = -۲$$

$$\rightarrow V_{۲}(S_{۲}) = max[-۲, -۲, -۰/۲, -۱] = -۰/۲$$

$$\rightarrow V_{۲}(S_{۳}) = max[-۲, -۲, -۰/۸, -۲] = -۰/۸$$

$$\rightarrow V_{۲}(S_{۴}) = max[-۲, -۲, -۱, -۰/۲] = -۰/۲$$

$$\rightarrow V_{۲}(S_{۵}) = max[-۲, -۲, -۲/۲] = -۲$$

$$\rightarrow V_{۲}(S_{۶}) = max[-۲, -۲, -۲/۲] = -۲$$

$$\rightarrow V_{۲}(S_{۷}) = max[-۲, -۲, -۰/۲, -۱] = -۰/۲$$

$$\rightarrow V_{۲}(S_{۸}) = max[-۲, -۲, -۰/۲, -۱] = -۰/۲$$

$$\rightarrow V_{۲}(S_{۹}) = max[-۲, -۲, -۲/۲] = -۲$$

$$\rightarrow V_{۲}(S_{۱۰}) = max[-۲, -۲, -۰/۲, -۱] = -۰/۲$$

ب

الگوریتم Q-Learning یکی از الگوریتم‌های یادگیری تقویتی است که برای حل مسئله یادگیری تقویتی بدون مدل (Model-Free) استفاده می‌شود. این الگوریتم به صورت تکراری و تمامیتی عمل می‌کند و به تدریج تابع Q-Value را به‌روزرسانی می‌کند تا به تابع Q-Value بهینه برسد.

برای استفاده از الگوریتم Q-Learning ، نیازمند تعریف یک MDP هستیم. MDP توسط چهار مؤلفه اصلی تعریف می‌شود:

مجموعه حالات (States): مجموعه‌ای از حالات قابل مشاهده در مسئله را نشان می‌دهد.

مجموعه عمل‌ها (Actions): مجموعه‌ای از عمل‌های قابل انجام در هر حالت را نشان می‌دهد.

تابع انتقال (Transition Function): تابعی است که نشان می‌دهد با انجام یک عمل در یک حالت، ما به چه حالتی خواهیم رفت و با چه احتمالی.

تابع پاداش (Reward Function): تابعی است که برای هر حالت و عمل، پاداش مشخص می‌کند.

هدف Q-Learning یافتن تابع Q-Value بهینه است که برای هر حالت و عمل، ارزش بهینه انتظاری را نشان می‌دهد. الگوریتم Q-Learning به صورت تکراری و مبتنی بر خبره (Experience-Based) عمل می‌کند و به تدریج مقادیر Q-Value را به‌روزرسانی می‌کند تا به تابع Q-Value بهینه برسد.

حالا الگوریتم را ران می‌کنیم:

$$Episode۱ :$$

$$Q(s_{۸}, R) = ۰/۵ \times -۱ = -۰/۵$$

$$Q(s_{۹}, U) = ۰/۵ \times -۱ = -۰/۵$$

$$Q(s_{۴}, S) = ۰/۵ \times ۱ = ۰/۵$$

$$Episode۲ :$$

$$Q(s_{۸}, S) = ۰/۵ \times ۱ = ۰/۵$$

$$Q(s_{۷}, R) = ۰/۵ \times ۱ = ۰/۵$$

$$Q(s_{۸}, R) = ۰/۵ \times ۰/۵ + ۰/۵ = ۰/۷۵$$

$$Episode۳ :$$

$$Q(s_{۱۰}, S) = ۰/۵ \times ۲ = ۱$$

$$Q(s_{۹}, R) = ۰/۵ \times ۱ = ۰/۵$$

$$Q(s_{۸}, R) = ۰/۷۵ + ۰/۵ \times ۰/۲۵ = ۰/۸۷۵$$

$$Episode۴ :$$

$$Q(s_{۸}, S) = ۰/۵ \times ۴/۵ = ۲/۲۵$$

$$Q(s_{۷}, L) = ۰/۵ \times ۰/۵ = ۰/۲۵$$

$$Q(s_{۸}, R) = ۰/۸۷۵ + ۰/۵ \times ۰/۱۲۵ = ۰/۹۳۷۵$$

$$Episode۵ :$$

$$Q(s_{۱۰}, S) = ۱ + ۰/۵ \times ۲ = ۲$$

$$Q(s_{۹}, U) = ۰/۵ + ۰/۵ \times -۰/۵ = ۰/۲۵$$

$$Q(s_{۸}, R) = ۰/۹۳۷۵ + ۰/۵ \times ۰/۰۶۲۵ = ۰/۹۶۸۷۵$$

سوال ۲.

الف

صحیح

ب

نه، این جمله درست نیست. در عمل، سیاست‌های پیداشده توسط الگوریتم‌های Policy Iteration و Value Iteration ممکن است با هم تفاوت داشته باشند و بسته به مسئله و شرایط مختلف، نتایج ممکن است متفاوت باشند. اما در حالت کلی، نمی‌توان گفت که سیاست‌های پیداشده توسط Policy Iteration ضعیف‌تر از سیاست‌های پیداشده توسط Value Iteration هستند یا برعکس.

در Policy Iteration ، به‌روزرسانی‌ها در دو مرحله انجام می‌شود: در مرحله اول، تابع ارزش بهینه برای سیاست فعلی به‌روزرسانی می‌شود و در مرحله دوم، سیاست جدید بر اساس تابع ارزش بهینه جدید تعیین می‌شود. به این ترتیب، الگوریتم Policy Iteration بهبودی پیوسته در سیاست و تابع ارزش بهینه ایجاد می‌کند.

در Value Iteration ، تابع ارزش بهینه به صورت تکراری به‌روزرسانی می‌شود و در هر مرحله، تابع ارزش بهینه به تدریج به مقدار نهایی همگرا می‌شود. در نهایت، می‌توان از تابع ارزش بهینه برای تولید سیاست بهینه استفاده کرد.

به طور کلی، می‌توان گفت که Policy Iteration و Value Iteration به دست آوردن سیاست بهینه را هدف دارند و در نهایت به نتایج مشابهی می‌رسند. اما در مسائلی که سیاست‌ها بیش از یک سیاست بهینه دارند، ممکن است سیاست‌های پیداشده توسط Policy Iteration و Value Iteration با هم تفاوت داشته باشند. در این حالت، می‌توان گفت که هیچ‌کدام از این الگوریتم‌ها برتری مطلق را دارا نیستند و نتیجه نهایی بسته

پ

صحیح

ت

نه، این جمله درست نیست. تخفیف در الگوریتم‌های یادگیری تقویتی برای کنترل تأثیر پاداش‌های آینده استفاده می‌شود و به عنوان یک عامل تعدیل‌کننده (discounting factor) عمل می‌کند. ارزش تخفیف باید بین ۰ و ۱ قرار بگیرد و انتخاب مقدار مناسب آن بسته به مسئله و شرایط خاص است.

تخفیف باعث می‌شود پاداش‌های آینده کمتر ارزش داشته باشند و به ارزش پاداش‌های فعلی بیشتری توجه شود. با تخفیف بالا، عامل به دنبال دریافت پاداش‌های نزدیک در زمان حال است و با تخفیف کم، عامل تمایل دارد به پاداش‌های آینده بیشتر توجه کند. اما استنتاج کلی این نیست که تخفیف کوچکتر از ۱ همواره به عنوان یک پاداش منفی در نظر گرفته می‌شود.

ث

صحیح

سوال ۳.

الف

از راست شروع می‌کنیم:

$$p(۱۰ + \gamma \times ۰) + (۱ - p)(۰ + \gamma \times ۰) = ۱۰p$$
$$p(۰ + \gamma \times ۱۰p) + (۱ - p)(۰ + \gamma \times ۰) = ۱۰p^۲\gamma$$
$$۱(۰ + \gamma \times ۱۰p^۲\gamma) = ۱۰p^۲\gamma^۲$$
$$۱(۰ + \gamma \times ۱۰p^۲\gamma^۲) = ۱۰p^۲\gamma^۳$$

ب

از چپ شروع می‌کنیم.

$$۱(۵ + \gamma \times ۰) = ۵$$
$$۱(۰ + \gamma \times ۵) = ۵\gamma$$
$$p(۰ + \gamma \times ۵\gamma) + (۱ - p)(۰ + \gamma \times ۰) = ۵p\gamma^۲$$
$$p(۰ + \gamma \times ۵p\gamma^۲) + (۱ - p)(۰ + \gamma \times ۰) = ۵p^۲\gamma^۳$$

ج

باید معادله‌ی زیر را حل کنیم. به صورت کلی نیز می‌دانیم که $۰ < p, \gamma < ۱$ است.

$$۱۰p^۲\gamma^۲ > ۵\gamma \implies ۲p^۲\gamma > ۱ \implies p^۲ > \frac{۱}{۲\gamma} \implies$$
$$p > \frac{۱}{\sqrt{۲\gamma}}$$

د

در یادگیری تقویتی (Reinforcement Learning) ،

Policy Iteration و Value Iteration دو روش اصلی برای یافتن بهترین راه حل (optimal policy) در یک محیط تصادفی هستند.

Value Iteration: این الگوریتم از تخمین مقدار فعلی یک حالت شروع می کند و سپس برای هر حالت امتیاز تمام اقدامات ممکن را محاسبه می کند. این امتیازات به روز رسانی می شوند و این فرایند تا زمانی که تغییرات کمتر از یک آستانه خاص شود، ادامه می یابد. در نهایت، سیاست بهینه براساس تابع ارزش به روز شده تولید می شود.

Policy Iteration: در این روش، ما از یک سیاست تصادفی شروع می کنیم و سپس در یک حلقه دو قدم را تکرار می‌کنیم: (a) ثابت کردن سیاست و محاسبه تابع ارزش برای آن سیاست (Policy Evaluation) ، و

تفاوت اصلی این دو روش در نحوه ی به‌روزرسانی سیاست است. در Value Iteration ، ما فقط یک بار سیاست را به روز می‌کنیم (پس از اینکه تابع ارزش کافیا ثابت شده باشد)، در حالی که در Policy Iteration ، ما سیاست را در هر مرحله به روز می‌کنیم.

یکی از نکاتی که باید توجه داشته باشید این است که در برخی موارد، Policy Iteration ممکن است سریع‌تر از Value Iteration به سیاست بهینه برسد، زیرا هر بروزرسانی در Policy Iteration می‌تواند به سیاست بهینه

نزدیک‌تر شود، در حالی که در Value Iteration ، سیاست بهینه فقط در انتها تولید می‌شود. با این حال، Value Iteration می‌تواند در محیط هایی با فضای حالت یا عمل بزرگ یا در مواردی که محاسبات Policy Evaluation زمان‌بر هستند، کارآمدتر باشد.

سوال ۴.

الف

وقتی از عبارت «در صورتی که discount را نداشته باشیم» استفاده می‌کنیم، منظور از discount میزان تاثیر یا اهمیتی است که به مراحل آینده در محاسبه reward نسبت داده می‌شود. در الگوریتم‌ها و مدل‌های مختلف یادگیری تقویتی، از این عامل برای تعیین اهمیت مراحل آینده در محاسبه reward استفاده می‌شود. با داشتن discount ، مقدار reward برای مراحلی که در آینده قرار خواهند گرفت، کاهش می‌یابد.

بنابراین، در حالتی که discount را نداشته باشیم، ممکن است که به دلیل عدم توجه به اهمیت مراحل آینده، reward کلی که در ۱۰۰۰ مرحله بدست می‌آید، بیشتر به نظر برسد و ما بیشتر تمایل داشته باشیم آن را انتخاب کنیم. اما وقتی discount وجود دارد، reward ی که بعد از ۱۰ مرحله بدست می‌آید ممکن است اهمیت بیشتری داشته باشد و ما تمایل داشته باشیم آن را انتخاب کنیم.

همچنین، ممکن است که در مواقعی که دو حالت به تعداد مراحل یکسانی کمیت مشابهی از reward را ارائه می‌دهند، ما مترددِ شویم و تصمیم‌گیری را به تعویق بیندازیم. این به معنای این است که می‌توانیم در انتخاب بین این دو حالت کمی اضافه زمان صرف کنیم تا مطمئن شویم کدام گزینه بهتر است.

به طور کلی، در صورتی که عواملی مانند discount در نظر گرفته نشوند، اهمیت مراحل آینده و تفاوت بین reward ها کاهش می‌یابد و این می‌تواند در فرآیند تصمیم‌گیری تاثیر داشته باشد.

ب

بله، بالاخره این مراحل باید طی شوند و در نهایت به یک جواب همگرا می‌شوند.

ج

در رابطه‌ی ارائه شده برای جفت حالت‌ها، $R_{max|min}$ نشان‌دهنده‌ی حداکثر یا حداقل مقدار reward است که در آن جفت حالت بدست می‌آید. همچنین، γ مقداری بین ۰ و ۱ است که به آن عامل تخفیف گفته می‌شود و نشان‌دهنده‌ی اهمیت مراحل آینده در محاسبه‌ی reward است.

در صورتی که $\gamma < ۱$ باشد، جمع هندسی $\sum \gamma^i$ به مقدار $\frac{۱}{۱-\gamma}$ همگرا می‌شود. این به این معنی است که اهمیت مراحل آینده در محاسبه‌ی reward با افزایش تعداد مراحل کاهش می‌یابد، اما به طور متناسب با عامل تخفیف گفته می‌شود. به عبارت دیگر، هر چه γ کوچک‌تر باشد، اهمیت مراحل آینده بیشتر خواهد بود.

اگر $\gamma \geq ۱$ باشد، مجموعه $\sum \gamma^i$ به نام همگرایی دارای مقدار بی‌نهایت می‌شود. این به این معنی است که در این حالت، اهمیت مراحل آینده بی‌نهایت است و تاثیر مراحل از لحاظ زمانی نامحدود است.

بنابراین، برای اطمینان از همگرایی وجود عامل تخفیف، مقدار γ باید بین ۰ و ۱ قرار گیرد. در نهایت، مقدار $U_۲$ که نشان‌دهنده‌ی utility حالت است، به $\frac{R_{max|min}}{۱-\gamma}$ همگرا می‌شود. این به این معنی است که با در نظر گرفتن عامل

تخفیف، utility حالت برابر با تقسیم مقدار حداکثر یا حداقل reward بر تفاوت یک منهای γ خواهد بود. این مقدار نشان‌دهنده‌ی ارزش حالت در نظر گرفته شده با اهمیت مراحل آینده است.

موفق باشید.