راه حل تکلیف شماره ۳

فصل ۱۶ و ۱۷

تصمیم گیری

۱) الف) احتمال اینکه اولین شیر در n امین پرتاب ظاهر شود، 2^{-n} است، پس :

$$EMV(L) = \sum_{n=1}^{\infty} 2^{-n} \cdot 2^{n} = \sum_{n=1}^{\infty} 1 = \infty$$

ب) پاسخ های نوعی در بازه‌ی $4\$$ تا $100\$$ است.

ج) فرض می‌کنیم سرمایه اولیه (پس از پرداخت c برای ورود به بازی) برابر با $(k-c)\$$ باشد.

در این صورت :

$$U(L) = \sum_{n=1}^{\infty} 2^{-n} \cdot (a \log_2 (k - c + 2^n) + b)$$

برای سادگی فرض می‌کنیم $k - c = 0\$$ در این صورت :

$$\begin{aligned} U(L) &= \sum_{n=1}^{\infty} 2^{-n} \cdot (a \log_2 (2^n) + b) \\ &= \sum_{n=1}^{\infty} 2^{-n} \cdot an + b \\ &= 2a + b \end{aligned}$$

د) حداکثر مبلغ c از حل معادله زیر بدست می‌آید :

$$a \log_2 k + b = \sum_{n=1}^{\infty} 2^{-n} \cdot (a \log_2 (k - c + 2^n) + b)$$

برای حالت ساده‌ی فرض شده : $(k = c)$

$$a \log_2 k + b = \sum_{n=1}^{\infty} 2^{-n} \cdot (a \log_2 2^n + b) \Longrightarrow$$

$$a \log_2 c + b = 2a + b \Longrightarrow$$

$$c = 4$$

۲) ایستان بودن لازم می دارد که عامل ترجیح های یکسانی بین جفت دنباله های

$[s_0, s_1, s_2, \ldots]$ و $[s_0, s_1', s_2', \ldots]$

و بین جفت دنباله های

$[s_1, s_2, \ldots]$ و $[s_1', s_2', \ldots]$

داشته باشد. اگر سودمندی یک دنباله ماکزیمم پاداش باشد، ایستان بودن به سادگی نقض می شود.

برای مثال:

$$[4,3,0,0,0,\ldots] \sim [4,0,0,0,0,\ldots]$$

اما

$$[3,0,0,0,\ldots] \succ [0,0,0,0,\ldots]$$

هنوز می توانیم $U^{\pi}(s)$ را به عنوان ماکزیمم پاداش مورد انتظار با اجرای π با شروع از s تعریف کنیم.
با این حال ترجیح های عامل عجیب و غریب به نظر می رسد.
برای مثال اگر حالت جاری s پاداش R_{max} داشته باشد، عامل بین همه ی کنش ها بی تفاوت خواهد بود اما همین که کنش اجرا شود و عامل دیگر در s نباشد، ناگهان شروع به مراقبت در مورد آنچه در بعد اتفاق می افتد، خواهد کرد.

۳) یک مسئله ی جستجوی شناخی با موارد زیر تعریف می شود:

s_0 حالت آغازین

$S(s)$ تابع مابعد (که مجموعه ای از زوج های کنش ـ حالت (a, s') را برمی گرداند.)

$c(s,a,s')$ یک تابع هزینه ی گام

GOAL-TEST(s) تابع آزمون هدف.

یک راه حل بهینه یک مسیر کمترین هزینه از s_0 به هر هدفی است.
برای ساخت MDP شناور، تعریف می کنیم: $R(s,a,s') = -c(s,a,s')$ مگر اینکه s یک حالت هدف باشد که در این صورت $R(s,a,s') = 0$.
همچنین تعریف می کنیم $T(s,a,s') = 1$ اگر $(a,s') \in S(s)$ و نیز $\gamma = 1$.
راه حل بهینه برای این MDP یک سیاست (policy) است که مسیر کمترین هزینه از هر حالت تا نزدیک ترین حالت هدف به آن را می یابد.

۴)

الف) عامل به صورت شهودی می‌خواهد تا جای ممکن سریع‌تر به حالت 3 برسد، زیرا برای هرگام زمانی که حالت‌های 1 و 2 صرف خواهد کرد، هزینه می‌پردازد.

اما تنها کنش که به حالت 3 می‌رسد، (کنش b) دارای احتمال پایین است، بنابراین عامل باید هزینه‌ای که متحمل می‌شود (در هنگام تلاش برای رسیدن به حالت پایانی) را در تصمیم کند.

این پیشنهاد می‌کند که عامل باید به‌طور معین در حالت 1 کنش b را آزمایش کند. در حالت 2 می‌تواند بهتر باشد که کنش a را برای ورود به حالت 1 آزمایش کند (که امکان بهتری برای انتظار پذیرش به حالت 3 است) به جای اینکه مستقیماً بخواهد وارد حالت 3 شود. این تصمیم در حالت 2 شامل یک بده‌بستان عددی است.

ب) به‌کارگیری الگوریتم تکرار سیاست در گام‌های متناوب : تعیین ارزش و به‌هنگام‌سازی سیاست

Initialization : $U \leftarrow \langle -1, -2, 0 \rangle$, $\pi \leftarrow \langle b, b \rangle$.

Value determination :

$$\begin{cases} u_1 = -1 + 0.1\, u_3 + 0.9\, u_1 \\ u_2 = -2 + 0.1\, u_3 + 0.9\, u_2 \\ u_3 = 0 \end{cases} \Rightarrow$$

$$u_1 = -10 \ , \ u_2 = -20$$

Policy update :

In state 1, $\sum_j T(1,a,j)\, u_j = 0.8 \times (-20) + 0.2 \times (-10) = -18$

$\sum_j T(1,b,j)\, u_j = 0.1 \times 0 + 0.9 \times (-10) = -9$

پس در حالت 1 هنوز کنش b ترجیح داده می‌شود.

In state 2, $\sum_j T(2,a,j)\, u_j = 0.8 \times (-10) + 0.2 \times (-20) = -12$

$\sum_j T(2,b,j)\, u_j = 0.1 \times (0) + 0.9 \times (-20) = -18$

پس در حالت 2 کنش a ترجیح داده می‌شود.

قرار می‌دهیم : unchanged ? $\leftarrow$ false و ادامه می‌دهیم :

Value determination :

$$\begin{cases} u_1 = -1 + 0.1\, u_3 + 0.9\, u_1 \\ u_2 = -2 + 0.8\, u_1 + 0.2\, u_2 \\ u_3 = 0 \end{cases} \Rightarrow$$

$$u_1 = -10 \ , \ u_2 = -15$$

Policy update :

In state 1 , $\sum_j T(1,a,j)\, u_j = 0.8 \times (-15) + 0.2 \times (-10) = -14$

$\sum_j T(1,b,j)\, u_j = 0.1 \times 0 + 0.9 \times (-10) = -9$

پس کنش b در حالت 1 هنوز ترجیح داده می‌شود .

In state 2 , $\sum_j T(2,a,j)\, u_j = 0.8 \times (-10) + 0.2(-15) = -11$

$\sum_j T(2,b,j)\, u_j = 0.1 \times 0 + 0.9 \times (-15) = -13.5$

پس کنش a هنوز در حالت 1 ترجیح داده می‌شود .

unchanged ? بدون تغییر و true می‌ماند و خاتمه می‌یابد .

* توجه کنید که سیاست حاصل با شهود ما تطابق دارد : وقتی در حالت 2 هستیم سعی می‌کنیم به حالت 1 برویم و وقتی در حالت 1 هستیم سعی می‌کنیم وارد حالت 3 شویم .

ج) سیاست آغازین با کنش a در هر دو حالت ، $\pi \leftarrow \langle a, a \rangle$ ، به یک مسئله‌ی غیرقابل حل منجر می‌شود .

مسئله‌ی تعیین ارزش آغازین فرم

$$\begin{cases} u_1 = -1 + 0.2\, u_1 + 0.8\, u_2 \\ u_2 = -2 + 0.8\, u_1 + 0.2\, u_2 \\ u_3 = 0 \end{cases}$$

را دارد که دو معادله‌ی اول آن ناسازگار هستند .

اگر سعی کنیم آنها را به طور عددی حل کنیم ، مقادیری می‌یابیم که به $-\infty$ میل می‌کند .

استفاده از ضریب تخفیف γ با محدود کردن جمله‌هائی که عامل در هریک از حالات متحمل می‌شود ، (هزینه‌های تخفیف یافته‌ی مورد انتظار) منجر به راه حل‌های خوش تعریف می‌شود . با این وجود انتخاب ضریب تخفیف بر سیاست حاصله اثر می‌گذارد . برای γ کوچک ، هزینه‌ی تحمیل شده در آینده‌ی دور ، نقش قابل چشم پوشی در محاسبه‌ی ارزش بازی می‌کند ، زیرا "همه نزدیک صفراست" .

به عنوان یک نتیجه عامل می‌تواند کنش b را در حالت 2 انتخاب کند ، زیرا هزینه‌ی تخفیف یافته‌ی کوتاه مدت باقیمانده در حالت‌های غیرپایانی (حالت‌های 1 و 2) از هزینه‌ی تخفیف یافته‌ی بلندمدت کنش b که به طور مکرر شکست می‌خورد و عامل را در حالت 2 باقی می‌گذارد ، وزن بیشتری پیدا می‌کند .

۵)

a) کلید کار محاسبه‌ی max و $\sum$ در جای مناسب خود است.

برای $R(s,a)$ داریم:

$$U(s) = \max_a \left[R(s,a) + \gamma \sum_{s'} T(s,a,s') U(s') \right]$$

برای $R(s,a,s')$ داریم:

$$U(s) = \max_a \sum_{s'} T(s,a,s') \left[R(s,a,s') + \gamma U(s') \right]$$

b) برای این کار راه‌حل‌های مختلفی وجود دارد. یک راه ایجاد یک "pre_state" برای $pre(s,a,s')$ هر s,a,s' است به این صورت که اجرای a در s به s' منجر می‌شود بلکه منجر به $pre(s,a,s')$ می‌شود. در این حالت این واقعیت کد شده است که عامل از حالت s آمده است و انجام a آن را به اینجا رسانده است. از این pre_state تنها یک کنش b است که همیشه به s' منجر می‌شود.

فرض می‌کنیم MDP جدید، گذار T'، پاداش R' و نرخ تخفیف جدید γ' را داشته باشد، در این صورت

$$T'(s,a,pre(s,a,s')) = T(s,a,s')$$

$$T'(pre(s,a,s'),b,s') = 1$$

$$R'(s,a) = 0$$

$$R'(pre(s,a,s'),b) = \gamma^{-\frac{1}{2}} R(s,a,s')$$

$$\gamma' = \gamma^{\frac{1}{2}}$$

c) با ایده‌ی کنش b، می‌توانیم حالت‌های $post(s,a)$ را برای هر s و a ایجاد کنیم به گونه‌ای که:

$$T'(s,a,post(s,a,s')) = 1$$

$$T'(post(s,a,s'),b,s') = T(s,a,s')$$

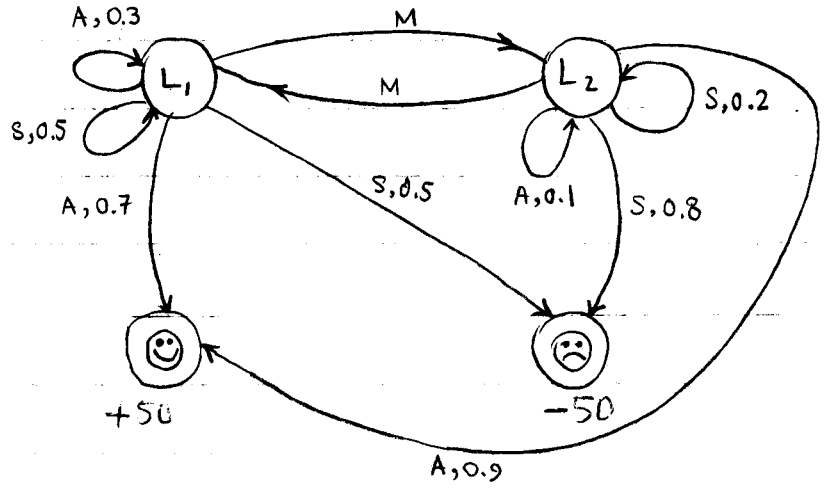
$$R'(s) = 0$$

$$R'(post(s,a,s')) = \gamma^{-\frac{1}{2}} R(s,a)$$

$$\gamma' = \gamma^{\frac{1}{2}}$$

Markov Decision Process (۶

نمودار زیر یک مدل MDP از یک fierce battle را به تصویر کشیده است. (دعوای خیابانی)



در این دعوا

می‌توان بین مکان‌های L_1 و L_2 جابجا شد. یکی از این مکان‌ها به حریف نزدیک‌تر است.
اگر از نزدیک‌ترین حالت حمله کنید،

- شانس بیشتری برای موفقیت دارید (۹۵٪) (در مقابل ۷۰٪ برای مکان دورتر از حریف)
- به هر حال ممکن است دیده شوید (با شانس ۸۵٪) و کشته شوید (در مقابل شانس ۵۰٪ برای مکان دورتر).

شما می‌توانید دیده شوید اگر در همان مکان خود باقی بمانید.
شما لازم دارید که یک نقشه‌ی action برای این موقعیت داشته باشید.
یکان‌ها، عمل‌های ممکن را نشان می‌دهند:

M	:	Move	قطعی
A	:	Attack	تصادفی
S	:	Stay	تصادفی

بر روی کمان‌ها، (a,p) نوشته شده است (عمل و احتمال آن).
همه پاداش‌ها صفر هستند به جز برای حالات پایانی که در آن موفقیت شما با پاداش 50+ نشان داده می‌شود، در حالی که موفقیت حریف، با پاداش 50- برای شما نشان داده می‌شود.
با بکارگیری فاکتور تخفیف $\gamma = 0.9$، سیاست بهینه (optimal policy) یا همان نقشه‌ی کنش‌ها را محاسبه کنید.

راه حل :

محاسبهٔ action_value ها برای همهٔ حالات و کنش ها لازم است.

$$V^*(\text{Far}) = \max\{Q^*(\text{Far}, M), Q^*(\text{Far}, S), Q^*(\text{Far}, A)\}$$

$$V^*(\text{Close}) = \max\{Q^*(\text{Close}, M), Q^*(\text{Close}, S), Q^*(\text{Close}, A)\}$$

$$V^*(☺) = +50$$

$$V^*(☹) = -50$$

در value_iteration با پیش‌فرض‌های آغازین زیر شروع می‌کنیم :

$$\forall s \forall a \quad Q_0(s, a) = 0$$

سپس همهٔ action_value ها را با توجه به قاعدهٔ زیر همگام می‌کنیم :

$$Q_{n+1}(s, a) = r(s, a) + \gamma \sum_{s'} p(s' | s, a) V_n(s')$$

که در آن $V_n(s') = \max_{a'} Q_n(s', a')$

بنابراین در تکرار اول الگوریتم به دست می‌آوریم :

$$Q_1(\text{Far}, A) = 0 + 0.9[0.7(+50) + 0.3(0)] = 31.5$$

$$Q_1(\text{Far}, S) = 0 + 0.9[0.5(-50) + 0.5(0)] = -22.5$$

$$Q_1(\text{Close}, A) = 0 + 0.9[0.9(+50) + 0.1(0)] = 40.5$$

$$Q_1(\text{Close}, S) = 0 + 0.9[0.8(-50) + 0.2(0)] = -36$$

ارزش عمل‌های M همان صفر باقی می‌ماند :

$$Q_1(\text{Far}, M) = 0 + 0.9(1 \times 0) = 0$$

$$Q_1(\text{Close}, M) = 0 + 0.9(1 \times 0) = 0$$

پس از این تکرار مقادیر این دو حالت خواهد بود :

$$V_1(\text{Far}) = \max_a Q(\text{Far}, a) = 31.5$$

$$V_1(\text{Close}) = \max_a Q(\text{Close}, a) = 40.5$$

که متناظر با کنش عجله (A) در هر دو حالت می‌باشد.

در تکرار بعدی خواهیم داشت :

$$Q_2(Far, A) = 0 + 0.9[0.7(+50) + 0.3(31.5)] = 40.5$$

$$Q_2(Far, S) = 0 + 0.9[0.5(-50) + 0.5(31.5)] = -8.325$$

$$Q_2(Far, M) = 0 + 0.9\, V_1(Close) = 0.9(40.5) = 36.45$$

$$Q_2(Close, A) = 0 + 0.9[0.9(+50) + 0.1(40.5)] = 44.145$$

$$Q_2(Close, S) = 0 + 0.9[0.8(-50) + 0.2(40.5)] = -28.71$$

$$Q_2(Close, M) = 0 + 0.9\, V_1(Far) = 0.9(31.5) = 28.35$$

و

$$V_2(Far) = 40$$

$$V_2(Close) = 44.145$$

که متناظر با عمل حمله در هر دو حالت است.

این فرایند می‌تواند ادامه یابد تا مقادیر بین تکرارهای متوالی تغییر چندانی نداشته باشند.
براساس آنچه تا این نقطه دیده‌ایم، بهترین نقشهٔ کنش حمله کردن در همهٔ زمان‌ها به نظر می‌رسد.
آیا می‌توان بدون شبیه‌سازی کامپیوتری چیز بیشتری گفت؟
بدیهی است که عمل "Stay" در هر دو حالت زیان‌بار است.
در حالت Close، بدیهی است که بهترین کار برای انجام به طور مداوم حمله کردن "Attack" است.
(که برای آن هزینه‌ای نمی‌پردازیم)
در واقع می‌توانیم ارزش‌ها را به طور تحلیلی در حد محاسبه کنیم (تغییرات در همنگام سازی از یک تکرار به تکرار دیگر)

$$\begin{aligned} Q^*(Close, A) &= \gamma \times 0.9 \times 50 + \gamma^2(1-0.9)(0.9)\times 50 \\ &\quad + \gamma^3(1-0.9)^2(0.9)\times 50 \\ &\quad + \cdots \\ &= \gamma \times 0.9 \times 50\,(1 + \gamma(1-0.9) + (\gamma(1-0.9))^2 + \cdots) \\ &= \frac{0.9\gamma}{1-\gamma(1-0.9)} \times 50 \approx 39.1304 \end{aligned}$$

حال برای حالت Far ، سؤال بین Attack یا Move به کدام نزدیک تر Close است.
ارزش‌ها را برای هر دوی این کنش‌ها محاسبه کنید (به همان روش قبلی) :

$$Q^*(\text{Far}, \text{M}) = 0.9\ V^*(\text{Close}) \approx 44.5055$$

$$Q^*(\text{Far}, \text{A}) = \frac{0.9 \times 0.7}{1 - 0.9 \times 0.3} \times 50 = 43.1507$$

بنابراین رفتن به Close بهتر است.

سیاست بهینه برای این صورت مساله (!) رفتن به Close (نزدیک تر شدن) و حمله کردن از آنجاست.
می‌توان تصور کرد که یک سیاست متفاوت حس بهتری برای این مساله ایجاد می‌کند ؟

۷)

الف) برای U_A :

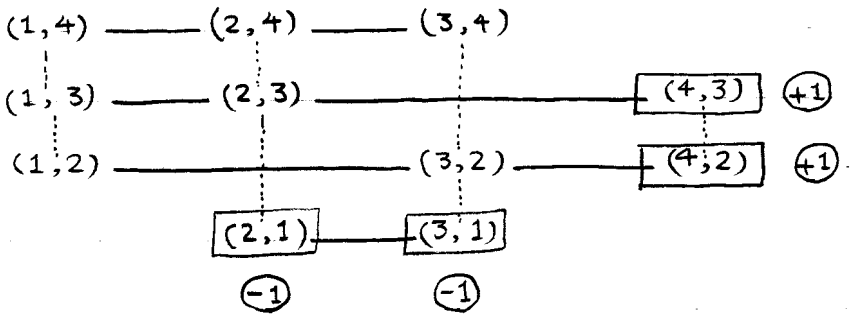
$$U_A(s) = R(s) + \max_a \sum_{s'} P(s'|a,s)\, U_B(s')$$

برای U_B :

$$U_B(s) = R(s) + \min_a \sum_{s'} P(s'|a,s)\, U_A(s')$$

ب) برای انجام تکرار ارزش، به طور ساده هریک از معادلات فوق را به معادله‌ی جهانگام سازی بلمن تبدیل می‌کنم و آنها را به طور متناوب به کار می‌بریم و هریک را برای همه‌ی حالت‌ها به طور همزمان به کار می‌بندیم. این فرآیند زمانی خاتمه می‌یابد که بردار سودمندی برای یک بازیکن، مثلاً بردار سودمندی متقی برای همان بازیکن شود (یعنی دوگام قبل‌تر). [توجه کنید که معمولاً U_A و U_B در موقعیت تعادل یکسان نمی‌باشند.]

ج)



د) اجرای الگوریتم تکرار ارزش در زیر نشان داده شده است. ارزش حالت پایانی برای هر بازیکن با دایره دور آن نشان داده شده است. سایر ارزش‌ها با ۰ مقداردهی اولیه شده‌اند. این الگوریتم به صورت زیر ادامه می‌یابد:

	(1,4)	(2,4)	(3,4)	(1,3)	(2,3)	(4,3)	(1,2)	(3,2)	(4,2)	(2,1)	(3,1)
U_A	0	0	0	0	0	(+1)	0	0	(+1)	(-1)	(-1)
U_B	0	0	0	0	-1	(+1)	0	-1	(+1)	(-1)	(-1)
U_A	0	0	0	-1	+1	(+1)	-1	+1	(+1)	(-1)	(-1)
U_B	-1	+1	+1	-1	-1	(+1)	-1	-1	(+1)	(-1)	(-1)
U_A	+1	+1	+1	-1	+1	(+1)	-1	+1	(+1)	(-1)	(-1)
U_B	-1	+1	+1	-1	-1	(+1)	-1	-1	(+1)	(-1)	(-1)

سیاست بهینه برای هر بازیکن به صورت زیر است:

	(1,4)	(2,4)	(3,4)	(1,3)	(2,3)	(4,3)	(1,2)	(3,2)	(4,2)	(2,1)	(3,1)
π_A^*	(2,4)	(3,4)	(2,4)	(2,3)	(4,3)			(3,2)	(4,2)		
π_B^*	(1,3)	(2,3)	(3,2)	(1,2)	(2,1)			(1,3)	(3,1)		