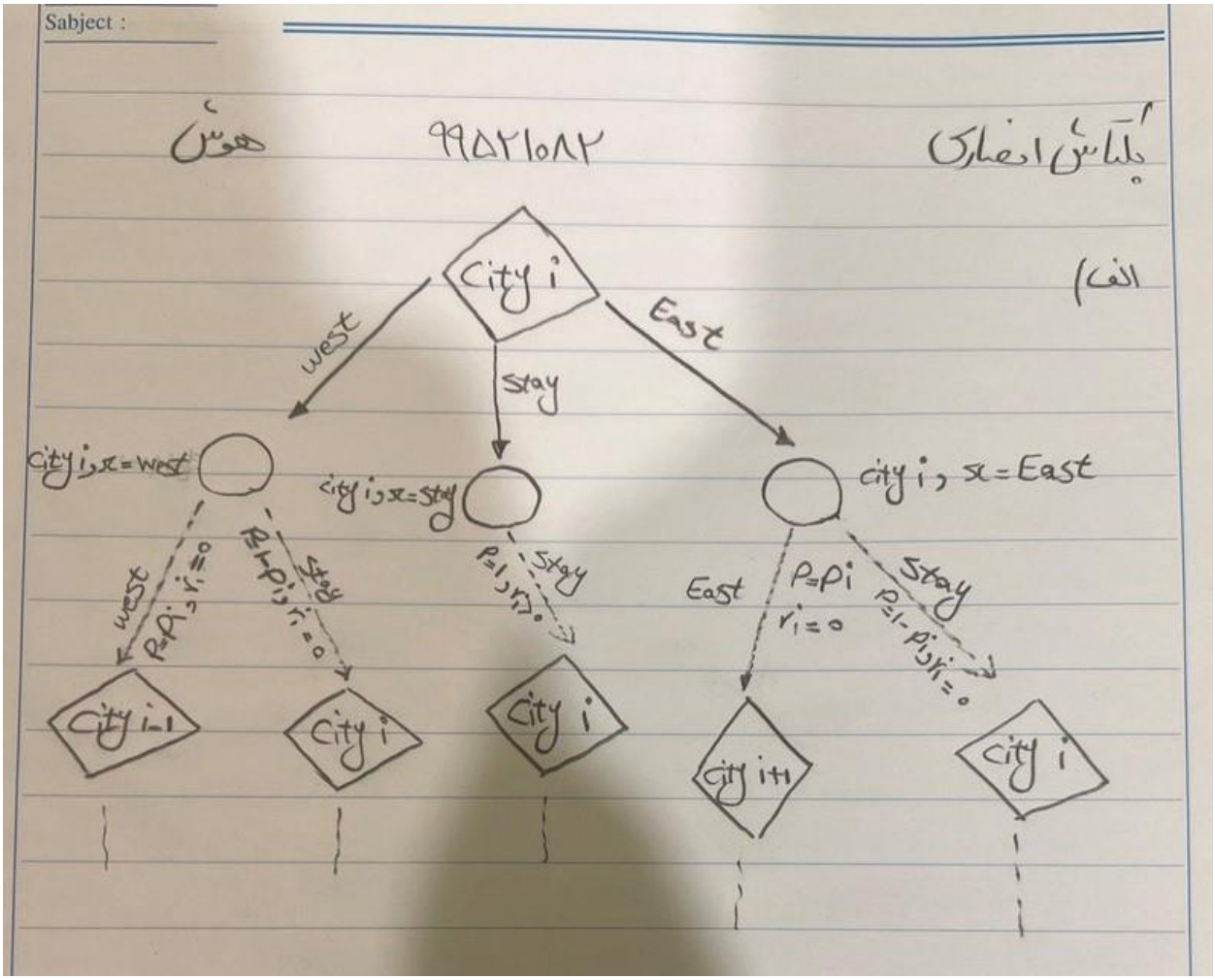
تمرین سری 12

بکتاش انصاری

99521082

بخش تئوری :



ب)

$\gamma = \frac{1}{2}$, $r_i = 1$, $P_i = 1$

در ابتدا تمام مقادیر V ها صفر هستند $\quad V_0^{stay}(1) = 0$

چون با $\pi = stay$ تنها $s' = s$ داریم: $\quad 1 = \leftarrow$ $\quad \leftarrow$ این مقدار

$$V_1^{stay}(1) = \sum_{s} T(s, stay, s)\left[\underbrace{R(s, stay, s)}_{=1} + \frac{1}{2} V_0^{stay}(s)\right]$$

$$\Rightarrow V_1^{stay}(1) = 1 \times \left[1 + \frac{1}{2} \times 0\right] = 1$$

$$V_2^{stay}(1) = 1 \times \left[1 + \frac{1}{2} \times 1\right] = 3/2$$

$$V_{k+1}^{stay}(1) = 1 \times \left(1 + \frac{1}{2} V_k^{stay}(1)\right)$$

$\Longleftarrow$ که هر عبارت بالا در $\infty \leftarrow k$ مقدارش ۲ می‌شود $\quad V^{stay}(1) = 2$

پ)

$$V_{k+1}^{*}(1) = \max_{a} \sum_{s'} T(s, a, s')\left[R(s, a, s') + \gamma V_k^{*}(s')\right]$$

$k=1$

$a = stay \rightsquigarrow V_1^{stay}(1) = 1 \times \overbrace{\left[1 + \tfrac{1}{2} \times 0\right]}^{stay} = 1$

$a = R \rightsquigarrow V_1^{R}(1) = 1 \times \underbrace{\left[1 + \tfrac{1}{2} \times 0\right]}_{Right} = 1$

$V_1^{*}(1) = 1$, $V_2^{*}(1) = \frac{3}{2}$, $V_3^{*}(1) = \frac{7}{4}$ ... $V_k^{*}(1) = 2 - \frac{1}{2^k}$

$k \to \infty \Longrightarrow V^{*}(1) = 2$

بخش ت)

چون مقدار گاما برابر 1 است پس ما مشکل کاهش ریوارد به ازای قد های بلند نداریم پس میتوانیم از این مورد عبور کنیم.

حال از انجایی که ممکن است در یک قدم ما دارای ریوارد 0 یا منفی باشیم اما آن ریوارد صفر باعث یک ریوارد بهتر در قدم های بعدی ما باشد. پس میتوانیم مانند روش hill climbing در هر مرحله بهترین reward را انتخاب کنیم. که این روش مشکلی که دارد این است که ممکن است در local maximum گیر کنیم که برای آنکه این مشکل برطرف شود میتوانیم یک بازه انتخاب کنیم (مثلا تعداد n قدم از محل شروع) و تمام این n قدم را طی کنیم و از بین آن ها ماکسیمم مقدار را انتخاب کنیم.

Date : / /

Subject :

(s, a, r, s')	Q_1, stay	Q_1, East	Q_2, west	Q_2, stay
initial	0	0	0	0
(1, stay, 4, 1)	$\frac{1}{2}\times 0 + \frac{1}{2}\times(4+\frac{1}{2}\times 0) = 2$	$\frac{1}{2}\times 0 + \frac{1}{2}\times(4+\frac{1}{2}\times 0) = 2$	0	0
(1, east, 0, 2)	$\frac{1}{2}\times 2 + \frac{1}{2}(0+\frac{1}{2}\times 0) = 1$	$\frac{1}{2}\times 2 + \frac{1}{2}(0+\frac{1}{2}\times 0) = 1$	0	0
(2, stay, 6, 2)	1	1	$\frac{1}{2}\times 0 + \frac{1}{2}(4+\frac{1}{2}\times 0) = 2$	$\frac{1}{2}\times 0 + \frac{1}{2}(4+\frac{1}{2}\times 0) = 2$
(2, west, 0, 1)	1	1	$\frac{1}{2}\times 2 + \frac{1}{2}(0+\frac{1}{2}\times 1) = \frac{5}{4}$	$\frac{1}{2}\times 2 + \frac{1}{2}(0+\frac{1}{2}\times 1) = \frac{5}{4}$
(1, stay, 4, 1)	$\frac{1}{2}\times 1 + \frac{1}{2}(4+\frac{1}{2}\times 1) = \frac{11}{4}$	$\frac{1}{2}\times 1 + \frac{1}{2}(4+\frac{1}{2}\times 1) = \frac{11}{4}$	$\frac{5}{4}$	$\frac{5}{4}$

$$\text{Sample} = R(s,a,s') + \gamma \max_{a'} Q(s',a')$$

$$Q(s,a) = (1-\alpha)\,Q(s,a) + \alpha\ \text{sample}$$

$$\alpha = \frac{1}{2},\ \gamma = \frac{1}{2}$$

PAPA

بخش عملی :

ابتدا توضیحی راجب پیاده سازی توابع میدهم :

تابع getQvalue :

از آنجایی که ما از Counter که یک دیکشنری خاص است استفاده میکنیم اگر key مورد نظر در دیکشنری وجود نداشته باشد به عنوان value به ما مقدار 0 را برمیگرداند و ValueError نخواهیم داشت. که اگر این مقدار برابر 0 باشد مقدار 0.0 ریترن میشود.

نکته پیاده سازی این تابع در این است که ما به عنوان کلید هر مقدار یک تاپل دوتایی میدهیم تا state و action هر دو مشخص شوند.

تابع computeValueFromQvalues :

در این تابع ما مقدار ماکسیمم Q ها را برای یک state مشخص و بین تمام اکشن های legal محاسبه میکنیم که اگر اکشن مجازی وجود نداشته باشد باید مقدار 0.0 ریترن شود.

تابع computeActionFromQvalues :

در این بخش مانند بخش قبل باید اکشن بهترین Q را خروجی بدهیم.

که در اینجا برای پیاده سازی من ابتدا مقدار تابع قبل را محاسبه کردم تا بهترین Q محاسبه شود و سپس کلید آن value را بدست آورده که تاپل دوتایی است و مقدار دوم تاپل را خروجی دادم که action مورد نظر خواهد بود.

تابع action :

در اینجا ما باید با احتمال epsilon محاسبه کنیم که چقدر agent ما از مقدار های محاسبه شده و از تجربه خودش استفاده کند و با چه احتمالی یک تجربه رندوم جدید را اعمال کند که برای این کار با استفاده از تابع flip_coin یک عدد رندوم بین صفر و یک ایجاد میکنیم که اگر آن عدد از epsilon کمتر بود یک مقدار رندوم در غیر اینصورت از مقادیر تجربه شده استفاده کند.

تابع update :

در این تابع نیز ما فرمول زیر را با استفاده از sample داده شده محاسبه میکنیم.

$$sample = R(s,a,s') + \gamma \max_{a'} Q(s',a')$$

no longer policy evaluation!

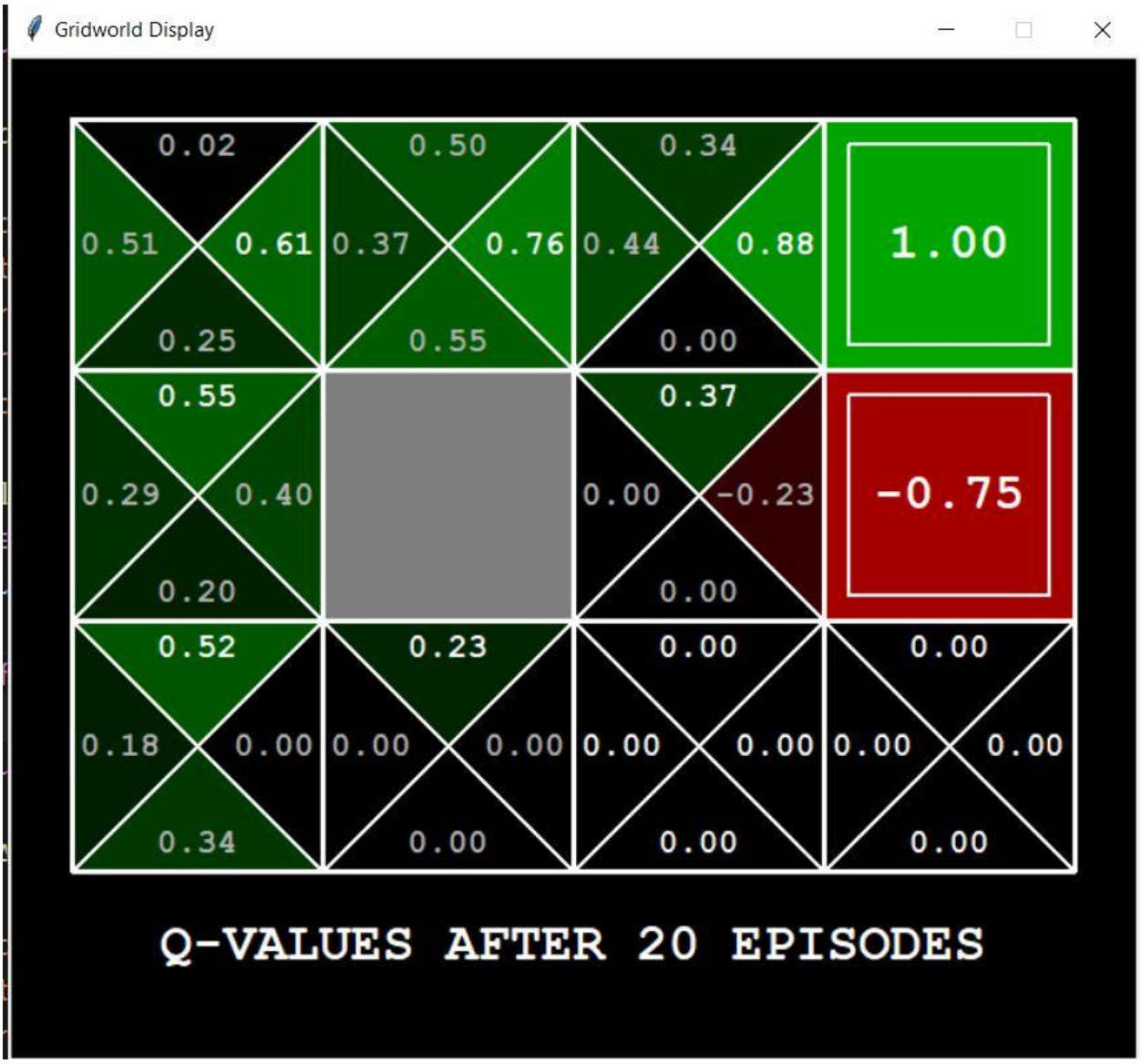
- Incorporate the new estimate into a running average:

$$Q(s,a) \leftarrow (1-\alpha)Q(s,a) + (\alpha)\,[sample]$$

حال به سوالات جواب میدهیم :

دلیل آنکه agent همواره مطابق جهتی که ما نشان میدهیم نمیرود این است که حرکت agent بصورت قطعی نیست و هر 4 جهت یک احتمالی دارند و قطعیتی برقرار نیست.

تصویر نهایی :



لاگ ترمینال در فایل زیپ موجود است.

برداشت من این است که هر چه تعداد iteration ها زیاد شود مسیر بهینه که به صورت بالا راست میباشد احتمال بیشتری پیدا کرده و ما از اینکه این مسیر بهینه است بیشتر مطمئن میشویم و احتمال جهت های این مسیر بیشتر میشود. و مسیر هایی که به ترمینال منفی ختم میشوند احتمال کمتری میگیرند.

: Crawler

مقادیر موجود :

: learning rate : Alpha

در اصل این مقیاس میزان یادگیری agent ما از روی سمپل جدید را مشخص میکند که در اصل باعث میشود که ما در Q جدید چ مقدار از مقدار قبلی Q و چه مقدار از sample جدید را استفاده کنیم.

این مقدار عددی بین صفر و یک است که اگر این مقدار برابر صفر باشد. Agent ما از sample یا action جدید چیزی یاد نمیگیرد و اگر مقدار آن برابر یک شود agent ما به طور کامل تمام دانش قبلی خود را ignore میکند و تنها از sample استفاده میکند.

هر چه این مقدار زیاد باشد باعث میشود Q بصورت سریع مقدار عوض کند.

: Epsilon

این مقدار مربوط به انتخاب به روش epsilon-greedy است که ما در آن برای انتخاب action مورد نظر استفاده میکنیم که در کد آن را در تابع action پیاده کردیم.

به این شکل که این مقیاس مشخص میکند که ما از یک action رندوم استفاده کنیم یا از اکشن های خوبی که تا الان داشتیم استفاده کنیم.

اگر این مقدار صفر باشد ما هیچوقت explore نداریم و فقط exploit داریم و از دانش کنونی خود استفاده میکنیم و تجربه جدیدی نمیکنیم.

اگر این مقدار برابر یک باشد ما هیچوقت از تجربه های گذشته خود استفاده نمیکنیم و فقط از مقادیر رندوم استفاده میکنیم.

معمولا این مقدار کوچک و نزدیک صفر است.

: Discount

مقیاسی است که در مقدار پیشبینی شده برای مقادیر پیش رو ضرب میشود تا مقدار Reward مقادیر دور با مقادیر نزدیک برابر نشود.

اگر مقدار آن را صفر در نظر بگیریم agent ما بصورت کامل reward های آینده را ignore میکند و تنها به reward کنونی اهمیت میدهد.