

به نام خدا



دانشگاه تهران
پردیس دانشکده‌های فنی
دانشکده برق و کامپیوتر



درس سیستم‌های هوشمند

تمرین شماره ۵

نام و نام خانوادگی محمدمهدی رحیمی

شماره دانشجویی ۸۱۰۱۹۷۵۱۰

دی ۱۴۰۰

فهرست سوالات

- سوال ۱ : بیز ساده انگارانه ۳
- سوال ۲: یادگیری تقویتی مبتنی بر مدل ۴
- ب: تحلیلی ۴
- سوال ۳: یادگیری تقویتی غیر مبتنی بر مدل ۶
- بخش امتیازی: ۷
- پیوست: ۱۰

سوال ۱: بیز ساده انگارانه

در این بخش ابتدا داده ها را وارد محیط برنامه کرده سپس با استفاده از توابعی که ایجاد کردیم مسوله را حل می کنیم. در ابتدا تابعی را داریم که با توجه به یک ویژگی خاص و مقدار آن تعداد داده هایی که آن ویژگی مقدار برابر با داده را دارند در دو کلاس مختلف را پیدا می کند. سپس با استفاده از فرمول زیر احتمال را برای هر داده مشخص کرده و آن را به کلاسی که بیشتر احتمال دارد اختصاص می دهیم.

$$\hat{P}(w|c) = \frac{\text{count}(w, c) + 1}{\text{count}(c) + |V|}$$

شکل ۱-۱: فرمول محاسبه شده برای محاسبه احتمال پسین

حال چون ویژگی ها مستقل اند احتمال تمام ویژگی ها در هم ضرب می شود و در نهایت ماتریس آشفتگی رسم می شود و دقت نمایش داده می شود.

Accuracy = 0.9942748091603053			
Actual			
	p	e	
Predict	p	272	0
	e	3	249

شکل ۱-۲: دقت و ماتریس آشفتگی با استفاده از بیز ساده انگارانه

سوال ۲: یادگیری تقویتی مبتنی بر مدل

ب: تحلیلی

با استفاده از الگوریتم مبتنی بر تکرار دو فرمول زیر را داریم:

❖ Solve the linear equations

$$V_{\pi}(s) = R_s^{\pi(s)} + \gamma \sum_{s'} P_{ss'}^{\pi(s)} V_{\pi}(s')$$

❖ Improve the policy at each state

$$\pi'(s) := \arg \max_{s'} R_s^a + \gamma \sum_{s'} P_{ss'}^a V_{\pi}(s')$$

شکل ۱-۲: فرمول های استفاده شده در روش

در اینجا ضریب گاما یک در نظر گرفته شده و داریم:

$$V(1,3) = 0.6(0 + 1 * 3) = 1.8$$

برای بقیه صفر می باشد.

حال برای مرحله بعد داریم:

$$V(1,3) = 0.6(0 + 1 * 3) + 0.2(0 + 1 * 1.8) = 2.16$$

$$V(1,2) = 0.6(0 + 1 * 1.08) = 1.08$$

$$V(2,3) = 0.6(0 + 1 * 1.8) + 0.2(0 + 1 * -2) = 0.88$$

برای بقیه صفر می باشد.

حال برای مرحله بعد داریم:

$$V(1,3) = 0.6(0 + 1 * 3) + 0.2(0 + 1 * 2.16) = 2.4$$

$$V(1,2) = 0.6(0 + 1 * 2.16) + 0.2(0 + 1 * 1.08) = 1.51$$



$$V(2,3) = 0.6(0 + 1 * 2.16) + 0.2(0 + 1 * -2) = 0.9$$



$$V(1,1) = 0.6(0 + 1 * 1.08) = 0.65$$



$$V(2,2) = 0.6(0 + 1 * 1.08) + 0.2(0 + 1 * 0.88) = 0.82$$



$$V(1,2) = 0.6(0 + 1 * 0.88) = 0.54$$

که در نهایت شکل های زیر را داریم:

0	0	0	3	→	→	→	3
0	0	0	-2	→	→	←	-2
0		0	0	↑		→	↓

0	0	1.8	3	→	→	→	3
0	0	0	-2	→	→	↑	-2
0		0	0	↑		→	↓

0	1.08	2.16	3	→	→	→	3
0	0	0.88	-2	→	↑	↑	-2
0		0	0	↑		↑	↓

0.65	1.51	2.4	3	→	→	→	3
0	0.82	0.9	-2	→	↑	↑	-2
0		0.54	0	↑		↑	↓

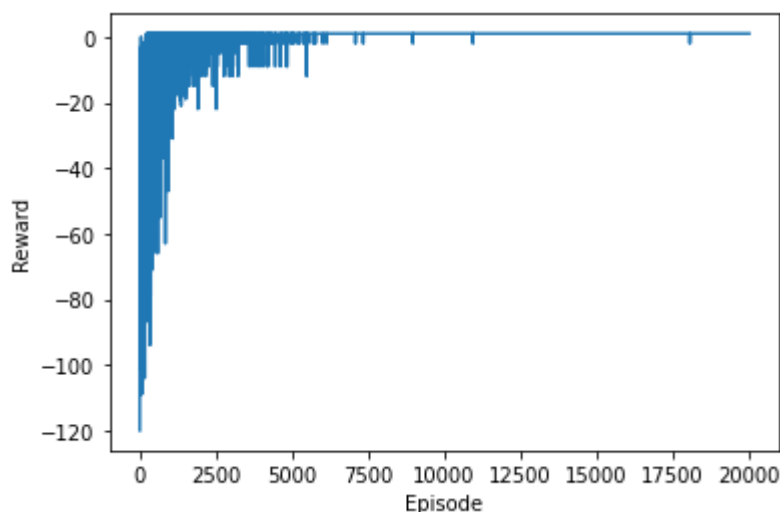
شکل ۲-۲: مراحل حل تحلیلی

سوال ۳: یادگیری تقویتی غیر مبتنی بر مدل

در این سوال ابتدا ماتریس Q را به اندازه دلخواه با توجه به تعداد حالت ها و تعداد حرکت های ممکن تشکیل می دهیم. سپس پارامتر هایی که برای آموزش مدل نیاز می باشد را انتخاب می کنیم. برای مثال تعداد اپیزود را بر روی ۲۰۰۰۰ قرار می دهیم زیرا در مقادیر کمتر ماتریس Q کامل نمی شود و برای حالت های خاصی این ماتریس جواب نمی دهد. برای تعداد اپیزود ۱۰۰۰۰ امتحان شد و تعدادی از سطر ها یا همان حالت ها خالی ماند. و اگر تعداد اپیزود را افزایش ندهیم در صورت قرار گیری در آن حالت ها چون تمام مقادیر ماتریس Q به طور پیش فرض صفر در نظر گرفته شده است حرکتی نخواهد کرد. و مقادیر دیگری همچون نرخ یادگیری و گاما نیز با آزمایش بدست آمده اند. گاما را اگر از حدی که قرار داده شده است کمتر کنیم آموزش ناقص می ماند زیرا این ضریب به پاداش های جلو تر مربوط می باشد و اهمیت آن ها را افزایش می دهد و چون در مراحل عادی پاداشی نداریم باید پاداش نهایی را در نظر بگیریم برای همین گاما بزرگ انتخاب شده است. و اما اپسیلون که در ابتدا یک در نظر میگیریم و با نرخ ۰,۹۹ آن را کاهش می دهیم تا بین حالت حریصانه و حرکت رندوم به مرور جا به جا شویم.

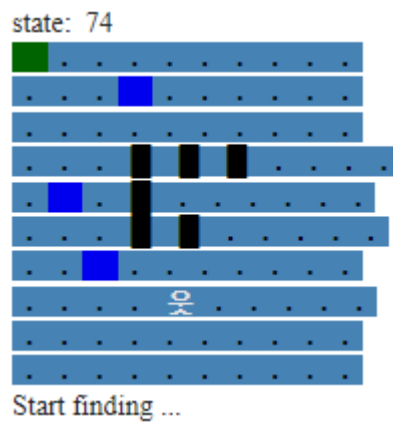
حال که مقادیر پارامتر ها را انتخاب کردیم الگوریتم را اجرا کرده و در طی تکرار ها با توجه به مرحله ای که طی می کنیم به مرور ماتریس را کامل می کنیم تا بعدا بتوان از آن استفاده کرد. با توجه به عدد رندومی که انتخاب می شود و با توجه به اپسیلون حرکت رندومی رخ می دهد یا با توجه به ماتریس Q حرکت می کنیم. و در نهایت با توجه به حرکت حالت جدید و پاداش را محاسبه می کنیم. که اگر پاداش -۱۰ گرفته باشیم از ابتدا دوباره شروع می کنیم .

حال اگر الگوریتم را اجرا کنیم و پاداش را بر حسب اپیزود رسم کنیم داریم:



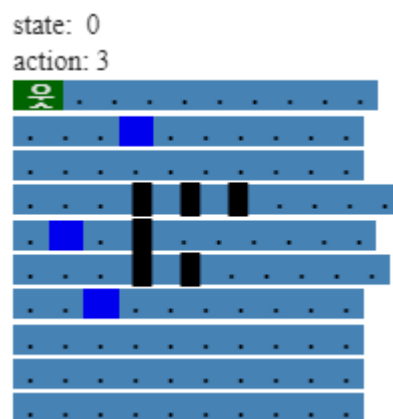
شکل ۳-۱: پاداش بر حسب اپیزود

حال برای تست کردن الگوریتم ابتدا به حالت ۷۴ می رویم سپس اجازه می دهیم تا الگوریتم اجرا شود تا نتیجه کار را مشاهده کنیم.



شکل ۲-۳ : حالت اولیه برای تست الگوریتم

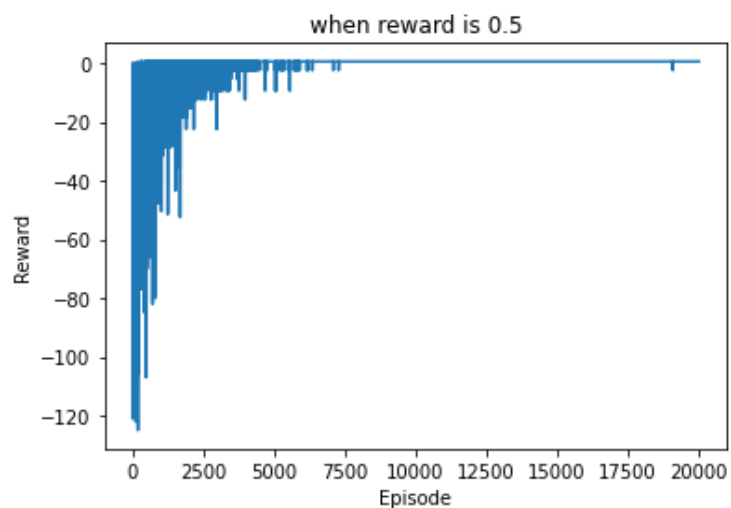
بعد از اجرای الگوریتم در ۱۵ مرحله به نقطه مورد نظر می رسیم.



شکل ۳-۳ : پس از اجرای الگوریتم و رسیدن به نقطه مطلوب

بخش امتیازی:

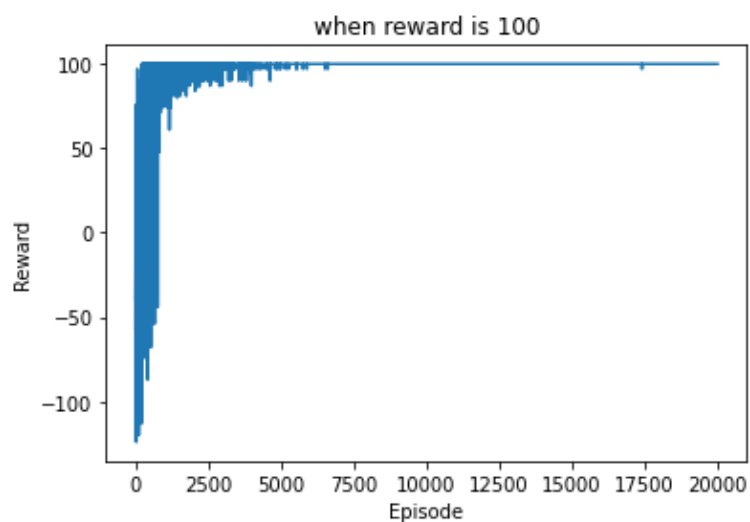
در این بخش پاداش را به ترتیب اعداد ۰,۵ و ۱۰ و ۱۰۰ و ۱۰۰۰ قرار می دهیم و پاداش بر حسب اپیزود را رسم می کنیم که داریم:



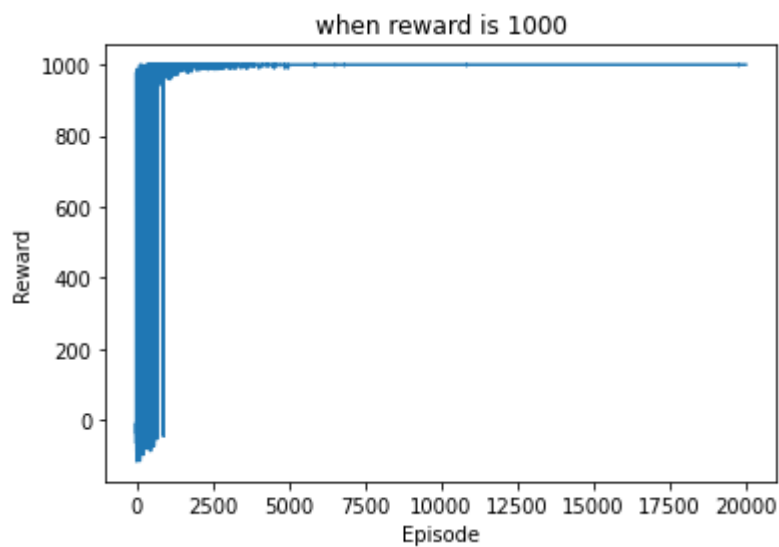
شکل ۳-۴ : نمودار پاداش بر حسب اپیزود برای پاداش نهایی ۰.۵.



شکل ۳-۵ : نمودار پاداش بر حسب اپیزود برای پاداش نهایی ۱۰.



شکل ۳-۶ : نمودار پاداش بر حسب اپیزود برای پاداش نهایی ۱۰۰.



شکل ۳-۷۶: نمودار پاداش بر حسب اپیزود برای پاداش نهایی ۱۰۰۰

همانطور که مشاهده کردید با افزایش پاداش سرعت همگرایی افزایش داشت و حتی در پاداش ۱۰۰۰ با حدود ۲۰۰۰ اپیزود همگرا می شود.

در اصل با تغییر پاداش تاثیر پاداش نهایی را افزایش می دهیم. زیرا در فرمول این مقدار در توان هایی از گاما ضرب می شود. حال اگر این مقدار افزایش یابد تاثیر آن بیشتر شده و آموزش سریع تر انجام می شود

پیوست:

تمام کد ها ضمیمه شده اند. کد های سوال یک به اسم IS_HW5_Q1.py می باشد و کد سوال سوم به اسم IS_HW5_Q3.ipynb می باشد که در Googlecolab نوشته شده و در این محیط تست و اجرا شده است.