دانشگاه تهران

پردیس دانشکده های فنی

دانشکده مهندسی برق و کامپیوتر

تکلیف شماره ۳ درس یادگیری ماشین

استاد درس :

جناب آقای دکتر نیلی

نام دانشجو :

کوروش محمودی

۸۱۰۱۹۸۰۵۰

سؤال (۱)

قسمت الف) در این قسمت سعی می کنیم مسئله Frozen Lake را به کمک روش یادگیری n-step SARSA حل کنیم. pseudo code مربوط به این الگوریتم به شکل زیر می باشد.

n-step Sarsa for estimating $Q \approx q_*$ or q_π

Initialize $Q(s,a)$ arbitrarily, for all $s \in \mathcal{S}, a \in \mathcal{A}$
Initialize π to be ε-greedy with respect to Q, or to a fixed given policy
Algorithm parameters: step size $\alpha \in (0,1]$, small $\varepsilon > 0$, a positive integer n
All store and access operations (for S_t, A_t, and R_t) can take their index mod $n+1$

Loop for each episode:
  Initialize and store $S_0 \neq$ terminal
  Select and store an action $A_0 \sim \pi(\cdot|S_0)$
  $T \leftarrow \infty$
  Loop for $t = 0, 1, 2, \ldots$:
  | If $t < T$, then:
  | Take action A_t
  | Observe and store the next reward as R_{t+1} and the next state as S_{t+1}
  | If S_{t+1} is terminal, then:
  | $T \leftarrow t+1$
  | else:
  | Select and store an action $A_{t+1} \sim \pi(\cdot|S_{t+1})$
  | $\tau \leftarrow t-n+1$ (τ is the time whose estimate is being updated)
  | If $\tau \geq 0$:
  | $G \leftarrow \sum_{i=\tau+1}^{\min(\tau+n,T)} \gamma^{i-\tau-1} R_i$
  | If $\tau + n < T$, then $G \leftarrow G + \gamma^n Q(S_{\tau+n}, A_{\tau+n})$ $(G_{\tau:\tau+n})$
  | $Q(S_\tau, A_\tau) \leftarrow Q(S_\tau, A_\tau) + \alpha\left[G - Q(S_\tau, A_\tau)\right]$
  | If π is being learned, then ensure that $\pi(\cdot|S_\tau)$ is ε-greedy wrt Q
  Until $\tau = T-1$

سپس با تغییر مقدار تعداد قدم ها (n) میزان regret را در طول اپیزودهای یادگیری بررسی می کنیم.

ابتدا پارامترهای شبیه سازی را مرور می کنیم.

- سیاست نرم: سیاست $\varepsilon - greedy$ نسبت به مقادیر ارزش action ها در state ها تنظیم می شود. میزان ε در اولین اپیزود برابر ۱ است و در هر طول اپیزود ثابت می ماند و با شروع اپیزود بعدی به صورت $\varepsilon \leftarrow \varepsilon * \exp(-episode\ number)$ به روز می شود.

- نرخ یادگیری α: مقدار نرخ یادگیری در طول اپیزود ثابت و از رابطه $\alpha = c(episode\ number)^{-\eta}$ و $0.5 < \eta < 1$ تعیین می شود که مقدار $c = 0.55$ و $\eta = 1$ منظور شده اند. میزان α برای هر جفت (state,action) در ابتدا یک است پس از اولین بار دیده شدن در طول یک اپیزود مقدار آن بروز شده و پس از آن هر چند بار دیگر که با آن در طول اپیزود روبرو شویم تغییر نمی کند و تا اپیزود بعد ثابت می ماند.
- Discount factor γ : مقدار DF برابر $\gamma = 0.9$ در نظر گرفته شده است.
- تعداد اپیزودها: تعداد اپیزودها ۱۰۰۰ در نظر گرفته شده است.
- تعداد step ها: تعداد step ها در روش n-step SARSA از ۱ تا ۱۰ تغییر داده شده اند

در این شبیه سازی به ازای هر n، ۱۰۰ بار یادگیری از ابتدا انجام شده و متوسط نتایج نمایش داده می شوند.

محاسبه **Regret**:

روش اول : میزان regret در هر اپیزود را از رابطه زیر بدست می آوریم:

$$regret(e) = BestReward - r_e$$

$$r_e = \sum_{t=1}^{T_e} r_t \quad sum\ of\ all\ rewards\ recieved\ during\ episode\ e, \quad T_e: length\ of\ episode\ e$$

در این فرمولبندی از مقدار هزینه ای که در طول تغییر state برای رسیدن از نقطه شروع و طی بهترین مسیر و رسیدن به مقصد صرف می شود صرف نظر شده و فقط پاداش رسیدن به آن یعنی ۱۰۰ به عنوان بهترین حالت ممکن در نظر گرفته شده است. بهترین حالت یعنی با خوشبینی هرگاه از مبدأ شروع می کردیم با صرف کمترین هزینه (تعدادی ۰.۱- به خاطر پیمودن کوتاه ترین راه ممکن) حتما به مقصد رسیده و جایزه ۱۰۰ را می گرفتیم.

میزان regret در انتهای هر اپیزود (به صورت تجمعی) را از رابطه زیر بدست می آوریم:

$$Regret(e) = BestReward * E - \sum_{e=1}^{E} r_e \quad E: number\ of\ episodes\ we\ had\ so\ far$$

$$r_e: sum\ of\ all\ rewards\ recieved\ during\ episode\ e$$

$$r_e = \sum_{t=1}^{T_e} r_t \quad , T_e: length\ of\ episode\ e$$

روش دوم : همچنین برای محاسبه regret می توان کار دیگری نیز انجام داد. در این رویکرد از به صورت قدم به قدم در طول یک اپیزود regret محاسبه می شود. به طور که میزان regret در یک قرم اپیزود برابر است با تفاضل میزان متوسط پاداش لحظه ای که عامل از انجام عمل a در حالت s می گیرد و متوسط پاداش لحظه ای که از انجام عمل بهینه در این حالت می گرفت. سپس با تجمیع این regret های لحظای به نوعی regret کل اپیزود را محاسبه می کنیم. و برای نمایش تغییرات regret در طول کل اپیزودها به صورت تجمعی regret های هر اپیزود را با اپیزود قبلی جمع و نمودار آن را رسم می نماییم.

میزان regret در هر اپیزود:

$$regret(e) = \sum_{t=1}^{T_e} \left[\sum_{s'} R_{ss'}^{a_t} P_{ss'}^{a_t} - \sum_{s'} R_{ss'}^{a_t^*} P_{ss'}^{a_t^*} \right]$$

T_e: length of episode e

a_t: action done at step t in episode e

a_t^*: best action could have been done at at step t in episode e

$R_{ss'}^{a}$: average instan reward getting from s to s' doing a

$P_{ss'}^{a}$: possibility of getting from s to s' doing a

$\sum_{s'} R_{ss'}^{a} P_{ss'}^{a}$: average instan reward of doing a at state s

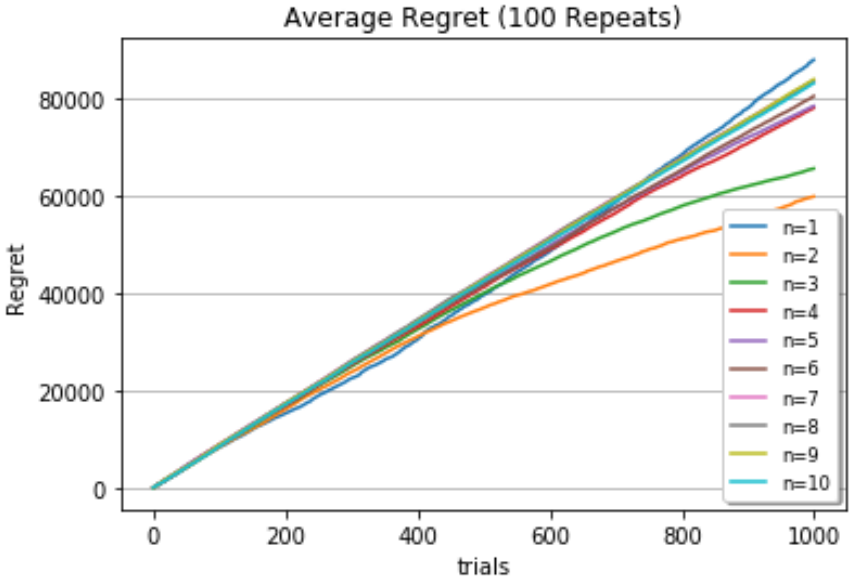
میزان regret در انتهای هر اپیزود (به صورت تجمعی):

$$Regret(e) = \sum_{e=1}^{E} regret(e) \quad , E: number\ of\ episodes\ we\ had\ so\ far$$

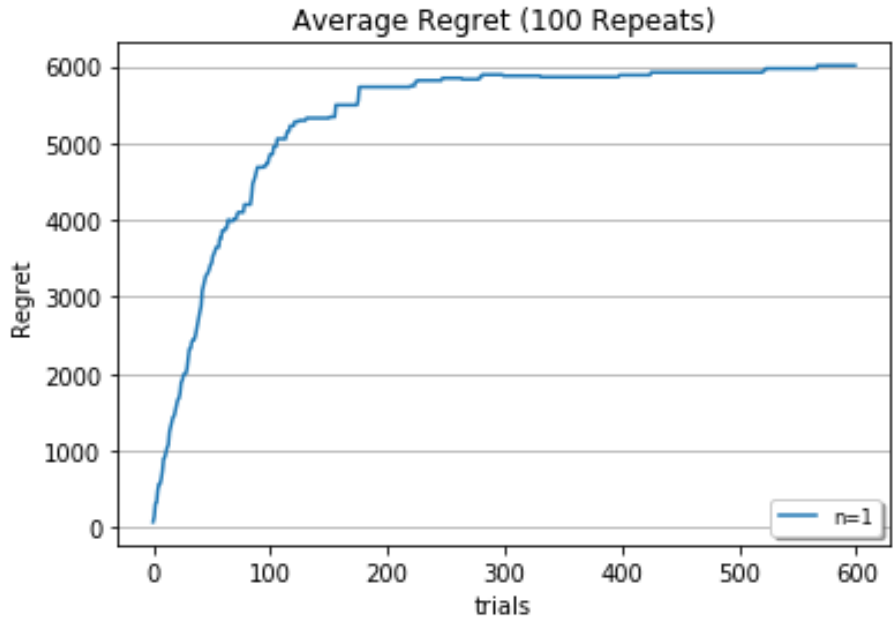
ما از این نوع رویکرد استفاده می کنیم. همچنین عمل بهینه در هر state را از یک روش value iteration بدست می آوریم.

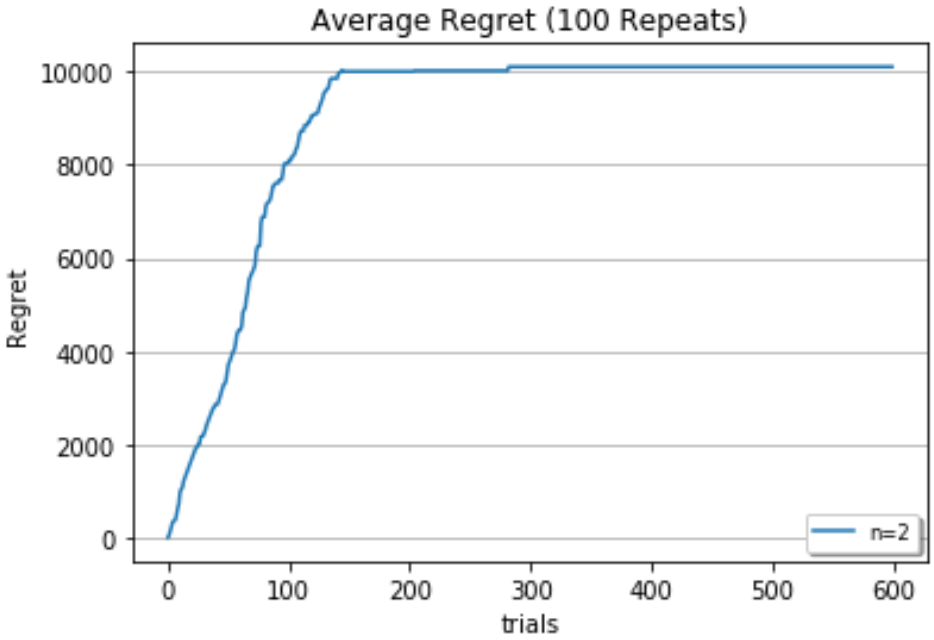
نتیجه:

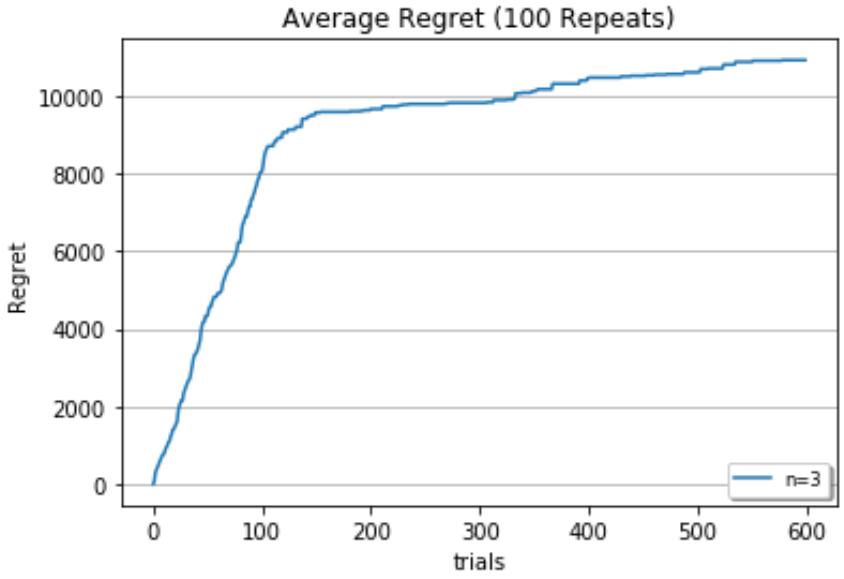
این الگوریتم برای n های بیشتر از ۱ به طور متوسط سرعت کمتری دارد. میزان سرعت همگرایی خیلی به نحوه تغییر میزان ε و α در طول یادگیری دارد. در زیر میزان regret را به ازای n=1 تا n=10 می بینیم. در این قسمت میزان نرخ یادگیری $\alpha = 0.01$ ثابت فرض شده است ولی اپسیلون به همان شکل اشاره شده در بالا در طول اپیزود کاهش داشته است.

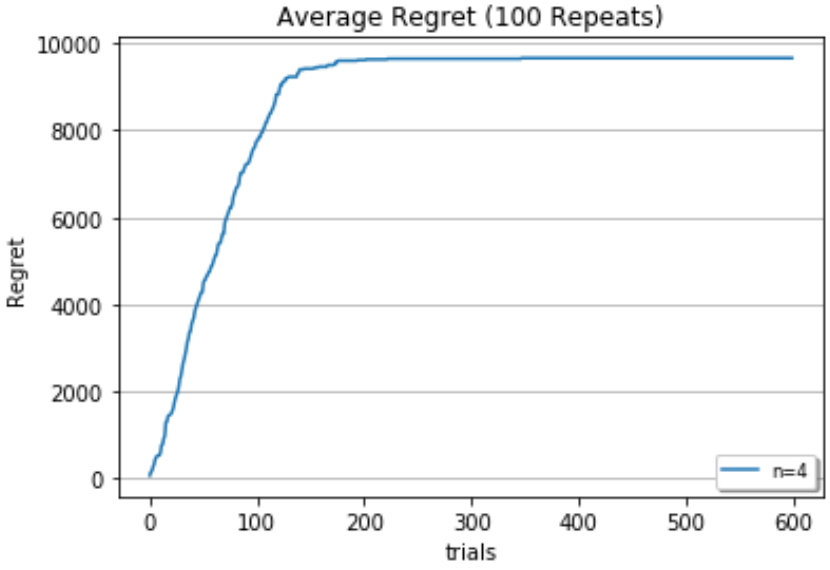


همین طور که از نتایج پیداست، میزان regret در n=2 و سپس در n=3 شرایط بهتری داشته است. اما نتایج نشان می دهند که به طور مثال برای n=2 به صورت متوسط در۶۰٪ مواقع در طول این هزار اپیزود به مقصد نرسیدیم. شیب خطی نمودار regret نشان می دهد که در طول یادگیری حرکت از exploration به سمت exploitation مناسب نبوده است. بسیار سعی شد که با تغییر نحوه کاهش میزان ε و α در طول یادگیری، این مسئله بهتر شود. همچنین نحوه کاهش میزان ε و α بسیار بر روی سرعت الگوریتم نیز تاثیر به سزایی دارد. بنابراین نوع کاهش میزان ε و α را به پارامتر n وابسته کردیم $\varepsilon \leftarrow \varepsilon * \exp(-episode\ number/(10 \times n))$ و نتایج بهتری به دست آمد:









برای n=1 تا n=5 تنظیم $\alpha = c(episode\ number)^{-\eta}$ به ترتیب به صورت $c = 1, 0.4, 0.4, 0.4, 0.1$ و $\eta = 0.55, 0.7, 0.85, 0.85, 0.9$ منظور شده است. مشاهده می شود که عملکرد n=1 بهتر از سایر مقادیر n است. همچنین سرعت همگرایی در n=2 بهتر بوده است.

قسمت ب) افزایش n باعث می شود که عامل از هر اپیزود یادگیری بیشتری داشته باشد. وقتی از روش های n-step استفاده می کنیم، مقدار ارزش action های انجام شده در طول مسیر اپیزود از پاداش گرفته شده در n قدم جلوتر در طول اپیزود، تأثیر می پذیرند. بنابراین میزان یادگیری با افزایش n بیشتر خواهد شد. اما اینکه میزان n چقدر باید باشد جای سؤال دارد. دو نهایت در روش های n-step، روش مونته کارلو (MC) و روش TD(0) است. معمولا انتخاب یک n میانی مناسب خواهد بود. اما میزان این n به مسئله متفاوت است.

قسمت ج) در این قسمت سعی می کنیم مسئله Frozen Lake را به کمک روش یادگیری SARSA(λ) حل کنیم. pseudo code مربوط به این الگوریتم به شکل زیر می باشد.

Initialize $Q(s,a)$ arbitrarily and $e(s,a)=0$, for all s,a
Repeat (for each episode):
. Initialize s, a
. Repeat (for each step of episode):
. . Take action a, observe r, s'
. . Choose a' from s' using policy derived from Q
. . . (e.g., ϵ -greedy)
. . $\delta \leftarrow r+\gamma Q(s',a')-Q(s,a)$
. . $e(s,a) \leftarrow e(s,a)+1$
. . For all s,a:
. . . $Q(s,a) \leftarrow Q(s,a)+\alpha\delta e(s,a)$
. . . $e(s,a) \leftarrow \gamma\lambda e(s,a)$
. . $s \leftarrow s'$; $a \leftarrow a'$
. until s is terminal

Figure 7.11: Tabular Sarsa(λ) .

به ازای سه مقدار $\lambda = 0\,,0.3\,,1$ میزان regret را در طول اپیزودهای یادگیری بررسی می کنیم.

ابتدا پارامترهای شبیه سازی را مرور می کنیم.

- سیاست نرم: سیاست $\varepsilon-\text{greedy}$ نسبت به مقادیر ارزش action ها در state ها تنظیم می شود. میزان ε در اولین اپیزود برابر ۱ است و در هر طول اپیزود ثابت می ماند و با شروع اپیزود بعدی به صورت $\varepsilon \leftarrow \varepsilon * \exp(-episode\ number)$ به روز می شود.
- نرخ یادگیری α: مقدار نرخ یادگیری در طول اپیزود ثابت و از رابطه $\alpha = c(episode\ number)^{-\eta}$ و $0.5 < \eta < 1$ تعیین می شود که مقدار $c = 0.5, 0.5, 0.01$ و $\eta = 0.55, 0.8, 0.9$ به ترتیب به ازای $\lambda = 0\,,0.3\,,1$ منظور شده اند. میزان α برای هر جفت (state,action) در ابتدا یک است پس از اولین بار دیده شدن در طول یک اپیزود مقدار آن بروز شده و پس از آن هر چند بار دیگر که با آن در طول اپیزود روبرو شویم تغییر نمی کند و تا اپیزود بعد ثابت می ماند.

- Discount factor γ : مقدار DF برابر $\gamma = 0.9$ در نظر گرفته شده است.
- تعداد اپیزودها: تعداد اپیزودها ۱۰۰۰ در نظر گرفته شده است.
- $\lambda = 0 , 0.3 , 1$

در این شبیه سازی به ازای هر λ ، ۱۰۰ بار یادگیری از ابتدا انجام شده و متوسط نتایج نمایش داده می شوند.

میزان regret را مانند قبل به دست می آوریم.

نتیجه

همانطور که مشاهده می شود به ازای $\lambda = 0.3$ بهترین نتیجه گرفته شد.