ctionmark[۱]

۶۳۴روش‌های بازیگر_ منتقدalgorithm.۶

دانشگاه صنعتی شریف

دانشکده‌ی علوم ریاضی

پایان‌نامه‌ی کارشناسی ارشد

رشته علوم کامپیوتر

عنوان:

دسته بندی الگوریتم های یادگیری تقویتی

نگارش:

حسین یوسفی زاده

استاد راهنما:

دکتر دانشگر

دی ۱۳۹۹


بسم الله الرحمن الرحیم

به نام خدا

دانشگاه صنعتی شریف

دانشکده‌ی علوم ریاضی

پایان‌نامه‌ی کارشناسی ارشد

عنوان: دسته بندی الگوریتم های یادگیری تقویتی

نگارش: حسین یوسفی زاده

کمیته‌ی ممتحنین

استاد راهنما: دکتر دانشگر امضاء:

استاد مشاور: استاد مشاور امضاء:

استاد مدعو: استاد ممتحن امضاء:

تاریخ:

چکیده

تئوری یادگیری تقویتی ،(RL) به تدریج به یکی از فعال ترین حوزه های تحقیقاتی در یادگیری ماشین و هوش مصنوعی تبدیل شده، که ریشه در دیدگاه های روانشناختی و علوم اعصاب درباره رفتار حیوانات و انسان دارد. یادگیری تقویتی تلاش می کند به این سوال پاسخ دهد: چه کار کنیم که بیشترین پاداش یا کمترین هزینه نصیبمان شود؟ اینکه چگونه عوامل هوشمند می توانند کنترل خود را روی محیط بهینه کنند نیز در حوزه یادگیری تقویتی قرار می گیرد. عوامل هوشمند با کاری دشوار روبرو می شوند: آنها باید مدل های کارآمدی از محیط را با استفاده از ورودی های حسی بدست آورند و از این مدل ها برای تعمیم تجربه گذشته به موقعیت های جدید استفاده کنند. به نظر می رسد انسان و سایر حیوانات این مشکل را از طریق ترکیب هماهنگ یادگیری تقویتی و سیستم های حسی سلسله مراتبی حل می کنند. در این پایان نامه، به بررسی روش های مبتنی بر یادگیری تقویتی در حالت گسسته می‌پردازیم و الگوریتم های مهم آن را خواهیم دید. نهایتا چند کاربرد دیگر را بررسی خواهیم کرد.

کلیدواژه‌ها: یادگیری تقویتی

فهرست مطالب

فصل ۱

مقدمه

یک روش یادگیری ماشین تعریف می‌شود که مربوط به نحوه اقدام یک عامل هوشمند [۱] در محیط براساس هدفی مشخص است. یادگیری تقویتی عبارت است از یادگیری اینکه عامل هوشمند چه کاری باید انجام دهد(نحوه انتخاب اقدامات برحسب موقعیت) تا به حداکثر پاداش برسد. این روش تمامی مسائل یک عامل هدفمند را صریحا در تعامل با یک محیط نامشخص بررسی می‌کند. به عامل هوشمند گفته نمی‌شود که چه کارهایی را انجام دهد ، اما درعوض باید کشف کند که کدام اقدامات، بیشترین پاداش را به همراه دارد. در جالب ترین و چالش برانگیزترین موارد ، اقدامات ممکن دارد نه تنها بر پاداش فوری بلکه در وضعیت بعدی محیط، و از طریق آن، بر کلیه پاداش های بعدی تأثیر بگذارد. این دو ویژگی (جستجوی آزمون و خطا و پاداش تأخیری) دو ویژگی مهم تمییز دهنده یادگیری تقویتی از روش های متداول یادگیری ماشین هستند. اقدامات عامل هوشمند، می‌تواند بر وضعیت آینده محیط تأثیر بگذارد یادگیری تقویتی یک رویکرد محاسباتی برای درک و خودکار کردن یادگیری و تصمیم‌گیری هدفمند است. یادگیری تقویتی، یادگیری از طریق تعامل است که چگونه می‌توان برای رسیدن به یک هدف رفتار کرد. عامل یادگیری تقویتی و محیط، در طی مراحل زمانی گسسته یا پیوسته با یکدیگر تعامل دارند.

[1] Agent

۱ـ۱ تعریف مسئله

مسئله‌ی **یادگیری تقویتی** در اصل یک مسئله بهینه سازی است. هدف اصلی مسئله، به حداکثر رساندن پاداشی است که از محیط دریافت می‌شود

تعریف دقیق‌تر این مسئله را در فصل دوم خواهیم دید.

۱ـ۲ اهمیت موضوع

یادگیری تقویتی در بسیاری از رشته ها مانند نظریه بازی، نظریه کنترل، تحقیق در عملیات، نظریه اطلاعات، بهینه سازی مبتنی بر شبیه سازی، سیستم های چند عاملی، هوش انبوه و آمار مورد مطالعه قرار می‌گیرد. در ادبیات تحقیق و کنترل عملیات، یادگیری تقویتی را برنامه ریزی تقریبی پویا [۲] یا برنامه ریزی عصبی پویا [۳] می‌نامند. مسائل مورد بررسی در یادگیری تقویتی در نظریه کنترل بهینه [۴] نیز مورد بررسی قرار گرفته است، که بیشتر مربوط به وجود و توصیف راه حل های بهینه و الگوریتم های محاسبه دقیق آنهاست، و کمتر مربوط به یادگیری یا تقریب، به ویژه در غیاب یک مدل ریاضی از محیط. در اقتصاد و نظریه بازی، ممکن است از یادگیری تقویتی برای توضیح چگونگی ایجاد تعادل، استفاده شود.

۱ـ۳ فضای حالت‌ها و عمل‌ها

۱ـ۴ خط مشی

۱ـ۵ سیگنال پاداش

terms in formalized is agent the of goal or purpose the learning، reinforcement In agent. the to environment the from passing reward، the called signal، special a of agent's the Informally، R. ؟ Rt number، simple a is reward the step، time each At

[2] Approximate Dynamic Programming
[3] Neuro-dynamic Programming
[4] Optimal Control Theory

maximiz- means This receives. it reward of amount total the maximize to is goal clearly can We run. long the in reward cumulative but reward، immediate not ing hypothesis reward the as idea informal this state

همه ی آنچه به عنوان هدف مدنظرداریم می‌تواند به صورت بیشینه سازی مقدار میانگین یک سیگنال عددی بیان شود.

۱ـ۶ تابع ارزش

۱ـ۷ محیط

۱ـ۸ مسئله اکتشاف و بهره برداری

یکی از چالش هایی که در یادگیری تقویتی برخلاف سایر روش‌های یادگیری وجود دارد ، رقابت بین اکتشاف و بهره برداری است. برای به دست آوردن پاداش زیاد ، یک عامل یادگیری تقویتی باید کارهایی را ترجیح دهد که در گذشته انجام داده و در تولید پاداش موثرتر بوده است. اما برای کشف چنین اعمالی، باید اقداماتی را امتحان کند که قبلاً انتخاب نکرده است. این عامل برای به دست آوردن پاداش مجبور است از آنچه قبلاً تجربه کرده است بهره برداری کند، اما همچنین برای انتخاب اقدامات بهتر در آینده باید به کاوش بپردازد. اکتشاف و بهره برداری هیچکدام به تنهایی در رسیدن به هدف، کارا نیست.

۱ـ۹ اهداف تحقیق

۱ـ۱۰ ساختار پایان‌نامه

این پایان‌نامه شامل ـــ فصل است. در فصل ...

فصل ۲

مفاهیم اولیه یادگیری تقویتی

کنترل بهینه [۱] در اواخر دهه ۱۹۵۰ برای توصیف مسئله طراحی یک کنترل‌گر برای به حداقل رساندن اندازه‌گیری رفتار سیستم *دینامیکی*[۲] در طول زمان مورد استفاده قرارگرفت. یکی از رویکردهای این مسئله در اواسط دهه ۱۹۵۰ توسط ریچارد بلمن[۳] و دیگران از طریق گسترش نظریه قرن نوزدهم همیلتون[۴] و جاکوبی[۵] توسعه یافت. این رویکرد از مفاهیم حالت یک سیستم دینامیکی و یک *تابع ارزش*[۶] برای تعریف یک معادله تابعی استفاده می‌کند؛ که اکنون *معادله بلمن*[۷] نامیده می‌شود. مجموعه روش‌های حل مسائل کنترل بهینه به کمک معادله بلمن به عنوان برنامه‌ریزی پویا شناخته می‌شود. همچنین بلمن نسخه گسسته از مسئله کنترل بهینه را که تحت عنوان *فرایندهای تصمیم‌گیری مارکوف*[۸] شناخته می‌شود، معرفی کرد. رونالد هوارد (۱۹۶۰) روش Policy Iteration را برای MDP ها طراحی کرد. همه این‌ها عناصر اساسی در تئوری و الگوریتم‌های *یادگیری تقویتی*[۹] مدرن هستند. در این فصل، با فرایندهای تصمیم‌گیری مارکوف آشنا خواهیم‌شد و چهار عنصر اصلی یادگیری تقویتی، یعنی *خط‌مشی*[۱۰]، *سیگنال پاداش*[۱۱]، تابع ارزش و محیط[۱۲] را دقیقا تعریف خواهیم‌کرد. همچنین برخی روش‌های کلاسیک در

[1] Optimal control
[2] Dynamical system
[3] Richard Bellman
[4] Hamilton
[5] Jacobi
[6] Value function
[7] Bellman equation
[8] Markov decision process(MDP)
[9] Reinforcement learning (RL)
[10] Policy
[11] Reward signal
[12] Environment

یادگیری تقویتی را معرفی می‌کنیم.

۲ـ۱ دینامیک عامل ـ محیط

حوزه یادگیری تقویتی دو بازیگر اصلی دارد؛ عامل و محیط. موجود تصمیم گیرنده و آموزنده را عامل یادگیری، یا به اختصار، عامل می‌نامند. قسمتی که عامل با آن تعامل دارد(هر چیزی خارج از عامل)، محیط نامیده می‌شود. در ادبیات کنترل بهینه، معمولا به جای واژه‌های عامل و محیط، از کنترل‌کننده [۱۳] و سیستم کنترل شده [۱۴] استفاده می‌شود. عامل و محیط به طور مداوم با یکدیگر ارتباط برقرار می‌کنند. عامل انتخاب می‌کند که چه اقدامی انجام دهد، محیط به این اقدامات پاسخ می‌دهد و موقعیت جدیدی را به عامل ارائه می‌دهد. محیط، همچنین مقادیر عددی ویژه ای به نام پاداش به عامل برمی‌گرداند، که عامل به دنبال به حداکثر رساندن آن است.

به طور خاص، عامل و محیط در یک توالی زمانی گسسته تعامل می‌کنند، $t = ۰, ۱, ۲, ۳, ...$، در هر مرحله t، عامل، وضعیت محیط $S_t \in \mathcal{S}$ را دریافت می‌کند، و بر اساس آن یک عمل $A_t \in \mathcal{A}$ را انتخاب می‌کند. در گام بعدی، عامل به عنوان نتیجه عمل خود، یک پاداش عددی $R_{t+۱} \in \mathcal{R}$ دریافت می‌کند و خود را در حالت جدید $S_{t+۱}$ می‌یابد. دینامیک عامل ـ محیط را می‌توان به شکل یک دنباله از حالت‌ها، عمل‌ها و پاداش‌ها به صورت زیر نمایش داد:

$$S_۰, A_۰, R_۱, S_۱, A_۱, R_۲, S_۲, A_۲, R_۳, ...$$

.

۲ـ۲ فرایند تصمیم‌گیری مارکوف

یادگیری تقویتی از چارچوب رسمی *فرایندهای تصمیم‌گیری مارکوف* [۱۵] (MDP) برای تعریف تعامل بین یک عامل یادگیری و محیط توسط حالت‌ها، اقدامات و پاداش استفاده می‌کند. مدل MDP یک مدل کلاسیک از تصمیم‌گیری متوالی است، جایی که اقدامات نه تنها بر پاداش‌های فوری، بلکه بر موقعیت‌ها

[13] Controller
[14] Controlled Systen
[15] Marcov Decision Process

و حالت‌های بعدی و به تبع آن بر پاداش‌های آینده تأثیر می‌گذارد. MDP یک فرم ایده‌آل ریاضی از مسئله یادگیری تقویتی است که برای آن تئوری‌های دقیقی بیان شده‌است. MDP متناهی، یک MDP با مجموعه حالت‌های متناهی است. بیشتر نظریه‌های فعلی یادگیری تقویتی، محدود به MDP های متناهی است، اما روش‌ها و ایده‌ها به طور کلی بیان می‌شوند.

تعریف ۲ـ۱ (فرایند تصمیم‌گیری مارکوف) *فرایند تصمیم‌گیری مارکوف، یک ۴ تایی*

$$\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{P} \rangle$$

است که

- $\mathcal{S}$ *بیانگر مجموعه تمام* ***حالت‌ها****ست،*
- $\mathcal{A}$ *بیانگر مجموعه تمام* ***عمل****هاست،*
- $\mathcal{R} \subseteq \mathbb{R}$ *بیانگر مجموعه* ***پاداش****هاست،*
- $\mathcal{P}$ ***هسته احتمال انتقال***[۱۶] $\mathcal{P} : \mathcal{S} \times \mathcal{A} \to \Pi(\mathcal{S} \times \mathcal{R})$*، تابعی است که دینامیک MDP را مشخص می‌کند.*

هسته احتمال انتقال یا تابع انتقال $\mathcal{P}$، هر دوتایی حالت‌ـعمل (s, a) که $s \in \mathcal{S}$ و $a \in \mathcal{A}$، را به یک توزیع احتمال روی دوتایی‌هایی به شکل (s', r) نسبت می‌دهد. s' بیانگر حالت بعدی و r بیانگر پاداش این انتقال است. به ازای هر دو حالت $s, s' \in \mathcal{S}$ و هر عمل $a \in \mathcal{A}$ و هر پاداش $r \in \mathcal{R}$ احتمال رسیدن به حالت s' و دریافت پاداش r با انتخاب عمل a در حالت s، یک عدد حقیقی متعلق به بازه $[۰, ۱]$ است که آن را به شکل $p(s', r|s, a)$ نمایش می‌دهیم:

$$p(s', r|s, a) \triangleq Pr\{S_t = s', R_t = r|S_{t-۱} = s, A_{t-۱} = a\}.$$

نامگذاری فرایند تصمیم‌گیری مارکوف اشاره به این موضوع دارد که این سیستم‌ها دارای **ویژگی مارکوف** [۱۷] هستند، بدین معنا که تابع انتقال آن تنها به حالت فعلی سیستم و آخرین عمل انجام شده وابسته است، و نسبت به حالت‌ها و اعمال قبلی مستقل است.

[16] Transition Probability Kernel
[17] Marcov Property

۲ـ۳ عناصر اصلی یادگیری تقویتی

چهار عنصر کلیدی و اصلی مسائل یادگیری تقویتی عبارت است از: *خط‌مشی*[۱۸]، *سیگنال پاداش*[۱۹]، *تابع ارزش*[۲۰] و *محیط*[۲۱]

۲ـ۳ـ۱ خط‌مشی

خط‌مشی[۲۲] نحوه رفتار عامل یادگیری را در یک زمان خاص، مشخص می‌کند و هسته اصلی رفتار یک عامل یادگیری تقویتی است. به بیان دیگر، خط‌مشی، نگاشتی از حالت‌های مدل شده از محیط به عملی است که باید در آن حالت انجام شود. خط‌مشی به تنهایی برای تعیین رفتار عامل کافی است. ممکن است خط‌مشی عامل یادگیری، یک تابع ساده یا جدول جستجو باشد، یا ممکن است شامل محاسبات پیچیده‌ای مانند فرآیند جستجو باشد؛ همچنین خط‌مشی‌ها ممکن است احتمالاتی باشند.

تعریف ۲ـ۲ (خط‌مشی احتمالاتی ثابت) *یک خط‌مشی احتمالاتی ثابت*[۲۳] *(یا به طور خلاصه خط‌مشی ثابت)* $\pi : S \to \Pi(A)$ *تابعی است که هر حالت را به یک توزیع احتمال روی فضای عمل* $\mathcal{A}$ *می‌نگارد. برای سادگی، احتمال انتخاب عمل* a *در حالت* s *تحت خط‌مشی* π *را به شکل* $\pi(a|s)$ *نشان می‌دهیم.*

تعریف ۲ـ۳ *می‌گوییم خط‌مشی* π *در یک MDP دنبال می شود هرگاه*

$$A_t \sim \pi(.|S_t), \quad t \in \mathbb{N}.$$

۲ـ۳ـ۲ سیگنال پاداش

در مسئله یادگیری تقویتی، تصمیمات عامل یادگیری، توسط سیگنال پاداش جهت‌دهی می‌شود. در هر گام، محیط، یک عدد حقیقی به عنوان پاداش برای عامل یادگیری تقویتی ارسال می‌کند. تنها هدف عامل، به حداکثر رساندن کل پاداش دریافتی از محیط در طولانی مدت است. سیگنال پاداش اتفاقات

[18] Policy
[19] Reward Signal
[20] Value Function
[21] Environment
[22] Policy
[23] Stationary probabilistic policy

خوب و بد را برای عامل مشخص می‌کند و مبنای اصلی تغییر خط‌مشی است سیگنال پاداش می‌تواند تابعی تصادفی از وضعیت محیط و عمل انجام شده باشد.

۲ـ۳ـ۳ عایدی و تابع ارزش

همانطور که سیگنال پاداش نشان می‌دهد که انجام چه عملی در هر گام خوب است، تابع ارزش مشخص می‌کند که کدام عمل در طولانی مدت بهتر است. در واقع تابع ارزش نشانگر مطلوبیت طولانی مدت حالت‌ها با در نظر گرفتن حالت‌هایی است که در پی خواهند داشت. ارزش حالت s، تحت خط‌مشی π، یا $v_\pi(s)$، مجموع میزان پاداشی است که عامل، با شروع از s و دنبال کردن خط‌مشی π، می‌تواند انتظار داشته باشد در آینده کسب کند. پاداش‌ها به یک معنا اولیه هستند، در حالی که ارزش‌ها، به عنوان پیش‌بینی پاداش‌ها، ثانویه هستند. بدون پاداش هیچ ارزشی وجود ندارد و تنها هدفِ تخمین ارزش‌ها، دستیابی به پاداش بیشتر است. با این وجود، این تابع ارزش است که هنگام تصمیم‌گیری و ارزیابی به آن توجه می‌کنیم. تعیین ارزش، بسیار دشوارتر از تعیین پاداش است. ارزش‌ها باید از توالی مشاهداتی که یک عامل در طول عمر خود انجام می‌دهد، برآورد شوند. مهمترین مولفه بیشتر الگوریتم‌های یادگیریِ تقویتی که در این فصل و فصل آینده معرفی خواهیم کرد، روشی برای تخمین کارآمد تابع ارزش است.

تعریف ۲ـ۴ *عایدی آینده کاهشی* [۲۴] *یا به اختصار، عایدی* [۲۵]*، در زمان t به شکل*

$$G_t \triangleq \sum_{t'=t}^{T} \gamma^{t'-t} R_{t'}$$

تعریف می‌شود که T زمانی است که اپیزود به اتمام می‌رسد. اگر مسئله مستمر باشد آنگاه $T = \infty$

از تعریف بالا نتیجه می‌شود

$$\begin{aligned} G_t =& R_t + \sum_{t'=t+1}^{T} \gamma^{t'-t} R_{t'} \\ =& R_t + \gamma \sum_{t'=t+1}^{T} \gamma^{t'-(t+1)} r_{t'} \\ =& R_t + \gamma G_{t+1}. \end{aligned} \tag{۱ـ۲}$$

[24] Future Discounted Return
[25] Return

تعریف ۲ـ۵ (تابع ارزش حالت) *ارزش حالت s تحت خط‌مشی π یا $v_\pi(s)$ به شکل امیدریاضی عایدی، با شروع از s و دنبال کردن خط‌مشی π تعریف می‌شود.*

$$v_\pi(s) \triangleq \mathbb{E}_\pi\left[G_t|S_t=s\right] = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty}\gamma^k R_{t+k+1}|S_t=s\right]$$

تابع v_π را تابع ارزش حالت [۲۶] *مربوط به خط‌مشی π می‌نامیم. اگر s یک حالت نهایی باشد آنگاه $v_\pi(s)=0$*

تعریف ۲ـ۶ (تابع ارزش عمل) *ارزش عمل a در حالت s تحت خط‌مشی π یا $q_\pi(s,a)$ به شکل امیدریاضی عایدی، با شروع از s و انتخاب عمل a و سپس دنبال کردن خط‌مشی π تعریف می‌شود.*

$$q_\pi(s,a) = \mathbb{E}_\pi\left[G_t|S_t=s,A_t=a\right] = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty}\gamma^k R_{t+k+1}|S_t=s,A_t=a\right]$$

تابع q_π را تابع ارزش عمل [۲۷] *مربوط به خط‌مشی π می‌نامیم.*

۲ـ۳ـ۴ مدل محیط

مدلِ محیط رفتار محیط را تقلید می‌کند، یا به طور کلی‌تر، اجازه می‌دهد تا در مورد نحوه رفتار محیط، پیشبینی کارآمدی داشته‌باشیم. از مدل‌ها برای برنامه‌ریزی و انتخاب در روند تصمیم‌گیری با در نظر گرفتن شرایط احتمالی آینده بدون تجربه واقعی آن‌ها استفاده می‌شود. روش‌هایی که برای حل مشکلات یادگیری تقویتی از مدل‌ها و برنامه‌ریزی‌ها استفاده می‌کنند، روش‌های *مبتنی بر مدل* [۲۸] نامیده می‌شوند. در مقابل این روش‌ها، روش‌های *بدون مدل* [۲۹] هستند که از هیچ گونه شبیه سازی یا ابزاری برای پیشبینی رفتار محیط استفاده نمی‌کنند و معمولا روند یادگیری آن‌ها بر اساس تجربه واقعی در محیط و آزمون و خطا است.

[26] State Value Function
[27] Action Value Function
[28] Model Based
[29] Model Free

۲ـ۴ برنامه‌ریزی پویا

در بخش قبل، با مفاهیم کلیدی یادگیری تقویتی آشنا شدیم. در این بخش به یکی از مهم ترین روش‌های کلاسیک در حل مسئله یادگیری تقویتی، یعنی *برنامه ریزی پویا* [۳۰]خواهیم پرداخت. برنامه‌ریزی پویا مجموعه‌ای از الگوریتم‌ها است در صورت وجود مدل کاملی از محیط در قالب یک MDP می‌توانند برای محاسبه بهترین خط‌مشی (خط‌مشی‌ای که بیشترین عایدی را به دست می‌دهد) استفاده شوند. الگوریتم‌های کلاسیک برنامه‌ریزی پویا به دلیل فرض مدل کاملی از محیط و همچنین هزینه محاسباتی زیادشان، به لحاظ عملی چندان قابل استفاده نیستند اما به لحاظ نظری مهم هستند. قبل از آن که به روش‌های برنامه ریزی پویا بپردازیم، لازم است با مفهوم خط‌مشی بهینه و تابع ارزش بهینه آشنا شویم.

۲ـ۴ـ۱ خط‌مشی و تابع ارزش بهینه

حل کردن مسئله یادگیری تقویتی، به معنی پیدا کردن خط‌مشی ای است که بیشترین پاداش را در طول زمان موجب می‌شود. به چنین خط‌مشی ای، *خط‌مشی بهینه* [۳۱] گفته می‌شود. برای MDP های متناهی، می‌توانیم خط‌مشی بهینه را به صورت زیر تعریف کنیم:

تعریف ۲ـ۷ *می‌گوییم خط‌مشی π **بهتر یا مساوی** خط‌مشی π' است یا $\pi \geqslant \pi'$ هرگاه برای هر $s \in S$*

$$v_\pi(s) \geqslant v_{\pi'}(s).$$

می‌توان نشان داد که برای هر MDP متناهی، حداقل یک خط‌مشی وجود دارد که بهتر یا مساوی هر خط‌مشی دیگری باشد [۱]. به چنین خط‌مشی‌ای **خط‌مشی بهینه** گفته می‌شود. ممکن است بیش از یک خط‌مشی بهینه وجود داشته باشد. تمام خط‌مشیهای بهینه را با نماد π_* نمایش می‌دهیم. تابع ارزش متناظر با همه خط‌مشیهای بهینه یکسان است و برابر با *تابع ارزش بهینه* [۳۲] است که با نماد v_* نمایش داده شده و به شکل زیر تعریف می‌شود:

$$v_*(s) \triangleq \max_\pi v_\pi(s).$$

[30] (DP)Dynamic Programming
[31] Optimal Policy
[32] Optimal Value Function

همچنین تمام خط‌مشی‌های بهینه تابع ارزشِ عمل مشترکی دارند که آن را با نماد q_* نمایش می‌دهیم و به شکل زیر تعریف می‌کنیم:

$$q_*(s,a) \triangleq max_\pi q_\pi(s,a).$$

می‌توانیم q_* را برحسب v_* به شکل زیر بنویسیم

$$q_*(s,a) = \mathbb{E}[R_{t+1} + \gamma v_*(S_{t+1})|S_t = s, A_t = a]. \quad (۲\_۲)$$

۲_۴_۲ معادله بلمن

ویژگی اساسی توابع ارزش که در طول یادگیری تقویتی و برنامه‌ریزی پویا استفاده می‌شوند، این است که آن‌ها در یک رابطه بازگشتی موسوم به *معادله بلمن* [۳۳] صدق می‌کنند. معادله بلمن رابطه‌ای بین ارزش یک حالت و ارزش‌های حالت‌های بعدی آن را بیان می‌کند.

فرض کنید π یک خط‌مشی دلخواه باشد و $s \in \mathcal{S}$. با استفاده مستقیم از تعریف ۲_۵ داریم

$$\begin{aligned} v_\pi(s) \triangleq & \mathbb{E}_\pi[G_t|S_t = s] \\ = & \mathbb{E}_\pi[R_{t+1} + \gamma G_{t+1}|S_t = s] \\ = & \sum_a \pi(a|s) \sum_{s'} \sum_r p(s', r|s, a)[r + \gamma \mathbb{E}_\pi[G_{t+1}|S_{t+1} = s']] \\ = & \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a)[r + \gamma v_\pi(s')]. \end{aligned} \quad (۳\_۲)$$

به رابطه ۲_۳ **معادله بلمن** گفته می‌شود. این معادله، رابطه ای بین ارزش یک حالت و ارزش حالت‌های بعدی آن را مشخص می‌کند. مشابه این رابطه را می‌توان برای تابع ارزش عمل نیز تحقیق کرد. فرض کنید $s \in \mathcal{S}$ و $a \in \mathcal{A}$. داریم

[33] Bellman Equation

$$
\begin{aligned}
q_\pi(s,a) \triangleq & \mathbb{E}_\pi[G_t|S_t=s,A_t=a] \\
=& \mathbb{E}_\pi[R_{t+1}+\gamma G_{t+1}|S_t=s,A_t=a] \\
=& \sum_{s',r} p(s',r|s,a)\sum_{a'}\pi(a'|s')[r+\gamma \mathbb{E}_\pi[G_{t+1}|S_{t+1}=s',A_{t+1}=a']] \\
=& \sum_{s',r} p(s',r|s,a)\sum_{a'}\pi(a'|s')[r+\gamma q_\pi(s',a')] \quad \forall s\in(S) \qquad (۴\_۲)
\end{aligned}
$$

رابطه ۲ـ۴ به معادله بلمن مربوط به تابع ارزش عمل معروف است.

۲ـ۴ـ۳ بهینگی و معادله بهینگی بلمن

از آنجا که v_* تابع ارزش یک خط‌مشی است، بنابراین باید در شرایط معادله ۲ـ۳ صدق کند. در این حالت خاص، معادله بلمن را می‌توان به فرم ویژه ای نوشت که با نام *معادله بهینگی بلمن* [۳۴] شناخته می‌شود.

$$
\begin{aligned}
v_*(s) =& \max_{a\in\mathcal{A}(s)} q_{\pi_*}(s,a) \\
=& \max_a \mathbb{E}_{\pi_*}[G_t|S_t=s,A_t=a] \\
=& \max_a \mathbb{E}_{\pi_*}[R_{t+1}+\gamma G_{t+1}|S_t=s,A_t=a] \\
=& \max_a \mathbb{E}[R_{t+1}+\gamma v_*(S_{t+1})|S_t=s,A_t=a] \\
=& \max_a \sum_{s',r} p(s',r|s,a)[r+\gamma v_*(s')]. \qquad (۵\_۲)
\end{aligned}
$$

معادله ۲ـ۵ بیانگر این واقعیت است که ارزش یک حالت تحت یک خط‌مشی بهینه، برابر با امیدریاضی عایدی، برای بهترین عمل از آن حالت است. به طور مشابه برای تابع ارزش عمل بهینه، معادله بهینگی بلمن به شکل زیر خواهد بود:

$$
\begin{aligned}
q_*(s,a) =& \mathbb{E}\left[R_{t+1}+\gamma \max_{a'} q_*(S_{t+1},a')|S_t=s,A_t=a\right] \\
=& \sum_{s',r} p(s',r|s,a)\left[r+\gamma\max_{a'} q_*(s',a')\right]. \qquad (۶\_۲)
\end{aligned}
$$

[34] Bellman Optimality Equation

۲ـ۴ـ۴ برنامه‌ریزی پویا

ایده اصلی DP و به طور کلی یادگیری تقویتی، استفاده از تابع ارزش حالت یا عمل برای سازماندهی یک الگوریتم جستجو برای پیدا کردن خط‌مشی بهینه است. v_* یا q_* است که در معادلات بهینگی بلمن صدق می‌کند:

$$v_*(s) = \max_a \sum_{s',r} p(s', r|s, a)[r + \gamma v_*(s')] \tag{۲ـ۷}$$

$$q_*(s, a) = \sum_{s',r} p(s', r|s, a)[r + \gamma \max_{a'} q_*(s', a')] \tag{۲ـ۸}$$

اگر دینامیک محیط کاملاً معلوم باشد معادله ۲ـ۷ و یک دستگاه معادلات خطی با اندازه $|\mathcal{S}|$ است که راه‌حل آن سرراست است. اما اگر تعداد حالت‌ها زیاد پاید، ممکن است این روش برای اهداف ما، عملی نباشد. روش‌های تکراری مناسب‌ترین روش‌ها هستند. یک دنباله $v_۰, v_۱, v_۲, ...$ از توابع ارزش تقریبی را در نظر بگیرید که هرکدام نگاشتی از $\mathcal{S}$ به $\mathbb{R}$ هستند. تقریب اولیه $v_۰$ به طور دلخواه انتخاب می‌شود(به جز در حالت‌های پایانی که باید صفر باشد). هر تقریب به عنوان یک قانون بروزرسانی به وسیله معادله بلمن از روی تقریب قبلی برای v_π بدست می‌آید.

$$\begin{aligned} v_{k+۱}(s) &\triangleq \mathbb{E}_\pi[R_{t+۱} + \gamma v_k(S_{t+۱})|S_t = s] \\ &= \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a)[r + \gamma v_k(s')] \end{aligned} \tag{۲ـ۹}$$

برای هر $s \in \mathcal{S}$. بر اساس معادله بلمن می‌توان درستی این تساوی را برای v_π نوشت و واضح است که $v_k = v_\pi$ نقطه ثابتی برای این قانون بروز رسانی است. در واقع، می‌توان نشان داد که در حالت کلی دنباله $\{v_k\}$ وقتی که $k \to \infty$ و تحت همان شرایط که وجود v_π را تضمین می‌کند، به مقدار v_π همگرا می‌شود. این الگوریتم را *ارزیابی خط‌مشی گام به گام* [۳۵] یا به طور خلاصه ارزیابی خط‌مشی [۳۶] می‌نامند.

[35] Iterative policy evaluation
[36] Policy Evaluation

الگوریتم ۱ الگوریتم ارزیابی خط مشی

ورودی: خط مشی π

۱: تا رسیدن به همگرایی تکرار کن:

۲: برای هر $s \in \mathcal{S}$:

۳: $V(s) \longleftarrow \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a) \left[r + \gamma V(s')\right]$

خروجی: $V \approx v_\pi$

۲ـ۴ـ۵ بهبود خط‌مشی

هدف ما برای محاسبه تابع ارزش یک خط‌مشی کمک به یافتن خط‌مشی‌های بهتر است. فرض کنید ما تابع ارزش v_π را برای یک خط‌مشی معین [۳۷] دلخواه π محاسبه کرده ایم. برای یک حالت s می‌خواهیم بدانیم که آیا باید خط‌مشی را برای انتخاب قطعی یک عمل $a \neq \pi(s)$ تغییر دهیم یا خیر. ما می‌دانیم که پیروی از خط‌مشی فعلی از حالت s چقدر خوب است، اما آیا تغییر به یک خط‌مشی جدید بهتر است یا بدتر؟ یکی از راه‌های پاسخ به این سوال در نظر گرفتن انتخاب عمل a در حالت s و پس از آن پیروی از خط‌مشی π است. ارزش این شیوه عملکرد برابر است با:

$$q_\pi(s, a) = \sum_{s',r} p(s', r|s, a) \left[r + \gamma v_\pi(s')\right]$$

اگر این عبارت از $v_\pi(s)$ بزرگتر باشد، بدین معناست که انتخاب قطعی عمل a در حالت s و سپس دنبال کردن خط‌مشی π بهتر از این است که همواره خط‌مشی π را دنبال کنیم. این گزاره را می‌توان به فرم کلی تر در قالب یک قضیه بیان کرد.

قضیه‌ی ۲ـ۱ (قضیه‌ی بهبود خط‌مشی) *فرض کنید π و π' دو خط‌مشی معین باشند که برای هر $s \in S$*

$$q_\pi(s, \pi'(s)) \geqslant \pi(s) \qquad (۲ـ۱۰)$$

در این صورت

$$v_{\pi'}(s) \geqslant v_\pi(s) \qquad (۲ـ۱۱)$$

[37] Deterministic

همچنین اگر نامساوی ۲ـ۱۰ برای یکی از حالت ها، به صورت اکید برقرار باشد آنگاه ۲ـ۱۱ نیز به صورت اکید برقرار است.

درستی قضیه بالا را می‌توان با استفاده از تعاریف به روشنی بررسی کرد؛

$$v_\pi(s) \leqslant q_\pi(s,\pi'(s)) = \qquad \mathbb{E}\left[R_{t+1} + \gamma v_\pi(S_{t+1})|S_t = s, A_t = \pi'(a)\right]$$

$$=\mathbb{E}_{\pi'}\left[R_{t+1} + \gamma v_\pi(s_{t+1})|S_t = s\right]$$

$$\leqslant\mathbb{E}_{\pi'}[R_{t+1} + \gamma q_\pi(S_{t+1}, \pi'(S_{t+1}))|S_t = s]$$

$$=\mathbb{E}_{\pi'}[R_{t+1} + \gamma\mathbb{E}_{\pi'}[R_{t+2} + \gamma v_\pi(S_{t+2})]|S_t = s]$$

$$\leqslant\mathbb{E}_{\pi'}[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 v_\pi(s_{t+3})|S_t = s]$$

$$\vdots$$

$$\leqslant\mathbb{E}_{\pi'}[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 R_{t+4} + \ldots|S_t = s]$$

$$\leqslant v_{\pi'}(s)$$

الگوریتم ۲ الگوریتم بهبود خط مشی

ورودی: خط مشی π و تابع $V \approx v_\pi$

۱: برای هر $s \in \mathcal{S}$:

۲: $\pi(s) \longleftarrow arg\max_a \sum_{s',r} p(s',r|s,a)\left[r + \gamma V(s')\right]$

خروجی: $V \approx v_\pi$

۲ـ۴ـ۶ الگوریتم تکرار خط‌مشی

قضیه ۲ـ۱ یک شرط کافی برای اینکه خط‌مشی π' بهتر یا مساوی خط‌مشی π باشد را بیان می‌کند. این قضیه می‌تواند ما را به روشی برای بهبود یک خط‌مشی، سوق دهد. منظور ما از بهبود خط‌مشی، روشی برای دستیابی به یک خط‌مشی بهتر، از روی یک خط‌مشی ضعیف تر است.

فرض کنید π' خط مشی معینی باشد که نسبت به تابع ارزش عمل q_π

$$\begin{aligned}\pi'(s) &\triangleq arg\max_a q_\pi(s,a)\\ &= arg\max_a \mathbb{E}\left[R_{t+1} + \gamma v_\pi(S_{t+1})|S_t = s, A_t = a\right]\\ &= arg\max_a \sum_{s',r} p(s',r|s,a)\left[r + \gamma v_\pi(s')\right].\end{aligned}$$

با استفاده از قضیه ۲ ـ ۱ می توان نشان داد

$$v_{\pi'}(s) \geqslant v_\pi(s).$$

بنابراین با استفاده از تابع ارزش خط‌مشی π می‌توان به یک خط‌مشی بهتر رسید [۱]. می توانیم با استفاده از ارزیابی خط‌مشی، روی π' به تخمینی از $v_{\pi'}$ برسیم؛ سپس می‌توانیم $v_{\pi'}$ را محاسبه کرده و مجدداً آن را بهبود بخشیم تا خط‌مشی بهتری داشته باشیم. با ادامه دادن این روند، می‌توانیم دنباله‌ای از خط‌مشی‌ها و تابع ارزش‌ها بهدست آوریم که به صورت یکنوا در حال بهبود هستند:

$$\pi_0 \longrightarrow v_{\pi_0} \longrightarrow \pi_1 \longrightarrow v_{\pi_1} \longrightarrow \pi_2 \longrightarrow \cdots \longrightarrow \pi_* \longrightarrow v_*$$

می‌توان نشان داد که این دنباله از خط‌مشی‌ها و توابع ارزش، به خط‌مشی و تابع ارزش بهینه همگرا می‌شود. به این روش، روش *تکرار خط‌مشی*[۳۸] گفته می‌شود.

۲ ـ ۴ ـ ۷ الگوریتم تکرار ارزش

یک اشکال در روش تکرار خط‌مشی این است که هر یک از تکرارهای آن، شامل ارزیابی خط‌مشی است که به خودی خود یک محاسبه زمان‌بر است که نیاز به رفت و برگشت‌های متعدد روی مجموعه حالت دارد. اگر ارزیابی خط‌مشی بصورت گام به گام تکراری انجام شود، همگرایی فقط در حد اتفاق می‌افتد. بنابراین یا باید منتظر همگرایی دقیق بمانیم و یا می‌توانیم قبل تر از آن متوقف شویم. در واقع، عملیات ارزیابی خط‌مشی را می‌توان پس از یک بار رفت و برگشت روی فضای حالت (یک بروزرسانی در هر حالت)، بدون از دست دادن تضمین همگرایی به خط‌مشی بهینه، متوقف کرد. این الگوریتم، *تکرار ارزش*[۳۹] نامیده می‌شود. این الگوریتم را می‌توان به عنوان یک عملیات بروزرسانی ساده نوشت که ترکیبی از بهبود خط‌مشی و مراحل ارزیابی خط‌مشی کوتاه شده است.

[38] Policy iteration
[39] Value Iteration

۲ـ۴ـ۸ درهم تنیدگی PE و PI و الگوریتم‌های GPI

یک اشکال عمده در روشهای DP که تا کنون بحث شده‌است، این است که آنها شامل عملیات‌هایی روی تمام مجموعه حالت‌ها هستند. اگر فضای حالت بسیار بزرگ باشد، حتی یک بار رفت و برگشت می‌تواند بسیار پرهزینه باشد. الگوریتم‌های ناهمزمان DP الگوریتم‌های تکرار شونده درجا هستند که مقید به رفت و برگشت سیستماتیک روی تمام مجموعه حالت‌ها نیستند. این الگوریتم‌ها ارزش حالت‌ها را بدون هیچ ترتیب منظمی و با استفاده از هر تخمینی از ارزش حالت‌های دیگر که در دسترس باشد، به روز می‌کنند. ارزش برخی حالت‌ها ممکن است قبل از اینکه ارزش دیگران یکبار به روز شود، چندین بار به روز شوند. با این حال، برای همگرایی صحیح، یک الگوریتم ناهمزمان باید بروزرسانی مقادیر همه حالت‌ها را ادامه دهد و نمی‌تواند حالتی را نادیده بگیرد. الگوریتم‌های ناهمزمان DP امکان انعطاف‌پذیری زیادی را در سیستم فراهم می‌کنند و در بسیاری از مسائل، تنها گزینه قابل اجرا به حساب می‌آیند.

۲ـ۵ روش های یادگیری تفاوت زمانی

برخلاف روش های برنامه‌ریزی پویا، روش های یادگیری تفاوت زمانی (TD) می توانند مستقیماً از تجربه خام بدون در اختیار داشتن مدلی از دینامیک محیط، یاد بگیرند. مانند DP، روش های TD، ارزش ها حالت (حالت_عمل) را بر اساس تخمین های حالت (حالت_عمل) های دیگر به روز می کنند.

همانطور که دیدیم، ارزش یک حالت، امید ریاضی عایدی، با شروع از آن حالت است. بنابراین ، یک روش آشکار برای تخمین ارزش حالت s از طریق تجربه، میانگین گرفتن تمام عایدی های تجربه شده در هر بار عبور از s است. با تجربه بیشتر و مشاهده عایدی های بیشتر، میانگین آن ها باید به $v_\pi(s)$ همگرا شود.

روش های تفاوت زمانی در ساده ترین حالت، بعد از عبور از هر حالت، ارزش آن حالت را با استفاده از ارزش حالت بعدی به عنوان تخمینی از عایدی، بلافاصله به روزرسانی می کنند

$$V(S_t) \longleftarrow V(S_t) + \alpha \left[R_{t+1} + \gamma V(S_{t+1}) - V(S_t)\right]$$

الگوریتم ۳ الگوریتم TD(۰) برای تخمین v_π

ورودی: خط مشی π

۱: تابع $V(s)$ را با مقادیر دلخواه مقداردهی اولیه کن. برای هر گام در اپیزود

۲: عمل a را با دنبال کردن خط مشی π انتخاب کن

۳: پاداش r و حالت جدید s' را مشاهده کن

۴: $V(s) \longleftarrow V(s) + \alpha\left[r + \gamma V(s') - V(s)\right]$

۵: $s \longleftarrow s'$

۶:

۲ـ۵ـ۱ Q-learning

ایده اصلی در روش Q-learning ، تخمین تابع مقدار عمل $Q^*(s,a)$ با استفاده از معادله بلمن به عنوان یک بروزرسانی تکراری،

$$Q_{i+1}(s,a) = \mathbb{E}[r + \gamma \max_{a'} Q_i(s',a')|s,a]$$

چنین الگوریتم‌های تکرار مقداری به تابع ارزشِ عمل بهینه همگرا می‌شوند، $Q_i \longrightarrow Q^*$ وقتی $i \longrightarrow \infty$ ، این رویکرد کلی کاملا غیر عملی است. زیرا تابع ارزشِ عمل برای هر دنباله، به طور جداگانه و بدون هیچ گونه تعمیم برآورد می‌شود. در عوض، معمولاً از یک تخمین‌گر توابع (مثل شبکه عصبی) برای تخمین تابع ارزشِ عمل استفاده می‌شود. در فصل سوم با این روش بیشتر آشنا خواهیم شد.

فصل ۳

نتایج اخیر

۳ـ۱ روش‌های مبتنی بر مدل و بدون مدل

یکی از مهم‌ترین نقاط انشعاب در الگوریتم‌های RL این است که آیا عامل به یک مدل از محیط دسترسی دارد یا توانایی آموختن مدلی از محیط را دارد؟ منظور از مدل محیط، تابعی است که انتقال و پاداش هر حالت_عمل را پیش‌بینی می‌کند. نقطه قوت داشتن مدل این است که به عامل اجازه می‌دهد با برآورد قبلی، طیف وسیعی از گزینه‌های ممکن را پیش بینی کند و به صراحت در مورد گزینه‌های خود تصمیم بگیرد. سپس عامل می‌تواند نتایج حاصل از برنامه‌ریزی قبلی را در قالب یک خط‌مشی بیاموزد یک نمونه مشهور از این روش AlphaZero است. در عمل، چنانچه دستیابی به مدلی از محیط امکان‌پذیر و عملی باشد، معمولا از روش‌های مبتنی بر مدل استفاده می‌شود. زیرا می‌تواند باعث بهبود قابل توجه‌ای در کارایی نمونه نسبت به روش‌های بدون مدل شود. اصلی ترین نقطه ضعف این روش‌ها این است که معمولا یک مدل کامل از محیط در دسترس عامل نیست و مدل کاملاً از طریق تجربه یاد گرفته می‌شود. الگوریتم‌هایی که از یک مدل استفاده می‌کنند، روش‌های مبتنی بر مدل و آن‌هایی که از چنین مدلی استفاده نمی‌کنند، بدون مدل نامیده می‌شوند. روش‌های بدون مدل از دستاوردهای بالقوه روش‌های مبتنی بر مدل چشم‌پوشی می‌کنند، اما پیاده‌سازی و تنظیم آنها آسان‌تر است. به همین خاطر، روش‌های بدون مدل از محبوبیت بیشتری برخوردار بوده و به طور گسترده‌تری توسعه و آزمایش شده‌اند.

۳ـ۲ روش‌های بدون مدل

۳ـ۲ـ۱ روش‌های مبتنی بر ارزش

در روش‌های یادگیری تقویتی بدون مدل مبتنی بر ارزش تابع ارزش عمل با استفاده از یک تخمین‌گیر تابع [۱] ، مانند شبکه عصبی ، نشان داده می‌شود. فرض کنید $Q(s,a;\theta)$ یک تابع ارزش عمل تقریبی با پارامتر θ باشد الگوریتم‌های مختلفی برای بروزرسانی θ وجود دارد الگوریتم $Q-learning$ یکی از نمونه‌های چنین الگوریتمی‌است که هدف آن تقریب مستقیم تابع ارزشِ عمل بهینه $Q^*(s,a) \approx Q(s,a:\theta)$ است. در $Q-learning$ یک مرحله‌ای، پارامترهای θ از تابع ارزشِ عمل با به حداقل رساندن تابع هزینه به شکل مرحله به مرحله آموخته می‌شوند، به شکلی که تابع هزینه iام به شکل

$$L_i(\theta_i) = \mathbb{E}\left(r + \gamma \max_{a'} Q(s',a';\theta_{i-1}) - Q(s,a:\theta_i)\right)^2$$

تعریف می‌شود که s' حالتی است که بعد از حالت s دیده می‌شود.

روش‌های Q-learning

خانواده روش‌های Q-learning تلاش می‌کنند مستقیما تابع ارزش عمل‌ـحالت بهینه $Q^*(s,a)$ را تخمین بزنند. آن‌ها به طور معمول از یک تابع هدف مبتنی بر معادله بلمن استفاده می‌کنند. این بهینه‌سازی تقریباً همیشه به صورت off-policy انجام می‌شود، به این معنی که هر بروزرسانی می‌تواند از داده‌های جمع‌آوری شده در هر نقطه استفاده کند، بدون اینکه در نظر بگیرد نحوه انتخاب عامل برای کشف محیط در هنگام بدست آوردن داده‌ها چگونه بوده است. خط‌مشی مربوطه از طریق ارتباط بین Q^* و π^* بدست می‌آید:

عامل بعد از یادگرفتن تابع $Q_\theta(s,a)$ به طوری‌که $Q_\theta(s,a) \approx Q^*(s,a)$می‌تواند عمل بهینه در حالت s را به به صورت زیر محاسبه کند

$$a(s) = arg\max_a Q_\theta(s,a).$$

از جمله الگوریتم‌های Q-learning می‌توان به موارد زیر اشاره کرد:

[1] Function approximator

- روش کلاسیک DQN که حوزه یادگیری تقویتی ژرف[۲] را عمیقا ارتقا بخشید.
- روش C۵۱ که توزیعی روی عایدی را می‌آموزد که امیدریاضی آن Q^* است.

روش DQN

معمولا برای تقریب زدن توابع ارزش در یادگیری تقویتی، از یک تابع خطی استفاده می‌شود. اما گاهی اوقات از یک تقریب عملکرد غیرخطی به جای آن، مانند یک شبکه عصبی استفاده می‌شود. شبکه‌های عصبی با عنوان شبکه Q[۳] شناخته می‌شوند. شبکه Q را می‌توان با کمینه ساختن دنباله‌ای از توابع هزینه به شکل $L_۱(\theta_۱), L_۲(\theta_۲), L_۳(\theta_۳), ...$ آموزش داد؛ به طوری‌که

$$L_i(\theta_i) = \mathbb{E}\left[(y_i - Q(s,a;\theta_i))^۲\right]$$

و

$$y_i = \mathbb{E}[r + \gamma \max_{a'} Q(s',a';\theta_{i-۱})|s,a].$$

با مشتق گرفتن از تابع هزینه نسبت به پارامترهای θ_i خواهیم داشت:

$$\nabla_{\theta_i} L_i\theta_i = \mathbb{E}\left[\left(r + \gamma \max a' Q(s',a';\theta_{i-۱}) - Q(s,a;\theta_i)\right) \nabla_{\theta_i} Q(s,a;\theta_i)\right].$$

به جای محاسبه امیدریاضی کامل در گرادیان فوق، غالباً از نظر محاسباتی، بهینه‌سازی تابع هزینه با نزول گرادیان تصادفی [۴] راه‌حل بهتری است. اگر در هر مقطع زمانی، وزن‌ها بروزرسانی شود و امیدریاضی با یک نمونه از توزیع خط‌مشی رفتار [۵] جایگزین شود، الگوریتم ۱۲ Q-learning را نشان می‌دهد.

توجه داشته باشید که ۱۲ یک الگوریتم بدون مدل است: این کار وظیفه یادگیری تقویتی را مستقیماً با استفاده از نمونه‌های شبیه ساز E بدون ساختن صریح تخمین E حل می‌کند. استراتژی حریصانه، $a = \max_a Q(s,a;\theta)$ ، در حالی که یاد می‌گیرد که کاوش کافی در فضای حالت را تضمین کند.!!!!!!!!!!!!!!! . توزیع رفتار اغلب توسط یک استراتژی Greed انتخاب می‌شود که استراتژی حریصانه را با احتمال ۱ دنبال می‌کند و یک اقدام تصادفی با احتمال ϵ !!!!!!!!!!!!!!!

[2] Deep reinforcement learning
[3] Q-Network
[4] stochastic gradient descend
[5] behavior policy

الگوریتم ۴ الگوریتم Q-learning با replay Experience

۱: حافظه replay D را مقدار دهی اولیه کن

۲: تابع ارزشِ عمل Q را با وزن‌های تصادفی مقداردهی اولیه کن

۳: برای هر اپیزود $M...۱$:

۴: دنباله $d_۱ = \{S_۱\}$ و کدینگ $\phi_۱ = \phi(d_۱)$ را مقداردهی اولیه کن

۵: برای $t = ۱...T$:

۶: با احتمال ϵ یک عمل تصادفی a_t را انتخاب کن، در غیر این صورت $a_t = \max_a Q^*(\phi(d_t), a; \theta)$ را انتخاب کن

۷: عمل a_t را انجام بده و حالت $S_{t+۱}$ و پاداش R_t را مشاهده کن

۸: قرار بده $d_{t+۱} = d_t, a_t, S_{t+۱}$ و $\phi_{t+۱} = \phi(d_{t+۱})$

۹: تجربه $(\phi_t, A_t, R_t, \phi_{t+۱})$ را در D ذخیره کن

۱۰: یک نمونه تصادفی از تجربه‌های $(\phi(j), A_j, R_j, \phi_{j+۱})$ از انبار تجربیات D انتخاب کن

۱۱: قرار بده $y_j = \begin{cases} \mathrm{r}_j & \phi_{j+1}\ terminal \\ \mathrm{r}_j & \text{otherwise} \end{cases}$

۱۲: یک گام از نزول گرادیان را برای تابع هزینه $(y_j - Q(\phi_j, a_j; \theta))^۲$ انجام بده

روش C۵۱

روش DDPG

گسسته سازی فضای عمل یک روش برای سازگارکردن و انطباق روش‌های یادگیری تقویتی عمیق نظیر DQN با دامنه‌های پیوسته می‌یاشد. با این حال، این روش محدودیت‌های زیادی دارد، مخصوصا مشکل نفرین ابعاد. نفرین ابعاد بیانگر این است که تعداد عمل‌ها به صورت نمایی با تعداد درجات آزادی افزایش پیدا می‌کند.

در روشی جدید یک الگوریتم بازیگر_منتقد مستقل از خط‌مشی، مستقل از مدل [۶] با استفاده از تخمین‌گر تابع عمیق ارائه می‌دهند که می‌تواند خط‌مشی‌ها را در فضاهای عمل پیوسته با ابعاد بالا یاد بگیرد.

این روش بر اساس الگوریتم گرادیان خط‌مشی معین [۷] است که آن را گرادیان خط‌مشی معین عمیق [۸] می نامیم. در ان روش، رویکرد بازیگر_منتقد را با بینش شبکه Q عمیق [۹] ترکیب می‌کنیم.

الگوریتم DPG تابع عمل پارامتری $\mu(s|\theta^{\mu})$ را نگهداری می‌کند که با نگاشت معین حالت‌ها به یک عمل خاص، سیاست فعلی را مشخص می‌کند.

همانند روش یادگیری Q در اینجا نیز منتقد $Q(s,a)$ با استفاده از معادله بلمن آموخته می‌شود. بازیگر با پیروی از اعمال قاعده زنجیره‌ای روی عایدی چشم‌داشتی از توزیع شروع J نسبت به پارامترهای بازیگر به روز می‌شود.

$$\begin{aligned}\nabla_{\theta^{\mu}} J =& \mathbb{E}_{s_t\sim\rho^{\beta}}[\nabla_{\theta^{\mu}} Q(s,a|\theta^{Q})|_{s=s_t,a=\mu(s_t|\theta^{\mu})}] \\ =& \mathbb{E}_{s_t\sim\rho^{\beta}}[\nabla_{\theta^{\mu}} Q(s,a|\theta^{Q})|_{s=s_t,a=\mu(s_t)} \nabla_{\theta^{\mu}} \mu(s|\theta^{\mu})|_{s=s_t}]\end{aligned} \qquad (۱\_۳)$$

در این روش نیز، مانند روش DQN از انبار تجربه استفاده می شود. در هر مرحله، بازیگر و منتقد با نمونه‌برداری یکنواخت از انبار به روز رسانی می‌شوند. از آنجا که DDPG الگوریتمی مستقل از خط‌مشی می‌باشد، انبار تجربه می تواند بزرگ باشد، که به الگوریتم اجازه می‌دهد تا از یادگیری مجموعه‌ای از انتقال‌های ناهمبسته بهره‌مند شود.

[6] A model-free, off-policy actor-critic algorithm
[7] Deterministic policy gradient (DPG)
[8] Deep DPG (DDPG)
[9] Deep Q Network (DQN)

۳ـ۲ـ۲ روش‌های مبتنی بر خط‌مشی

در این قسمت روش‌هایی را در نظر می‌گیریم که برای دستیابی به خط‌مشی بهینه، به جای استفاده از تابع ارزشِ عمل یا ارزشِ حالت، یک خط‌مشی پارامتری‌شده [۱۰] را می‌آموزد. با این حال ممکن است برای یادگیری پارامترهای خط‌مشی از یک تابع ارزش استفاده شود، اما تابع ارزش برای انتخاب عمل مورد نیاز نیست. ما از نماد $\theta \in \mathbb{R}^{d'}$ برای بردار پارامتر خط‌مشی استفاده می‌کنیم. بر خلاف روش‌های مبتنی بر ارزش، روش‌های مبتنی بر خط‌مشی مستقیماً تابع خط‌مشی $\pi_\theta(a|s)$ را تخمین می‌زنند و پارامترهای θ را با استفاده از **صعود گرادیان** [۱۱] روی یک مقیاس عملکرد $J(\pi_\theta)$ [۱۲] یا به طور مستقیم و یا با بیشینه‌سازی تخمین‌های محلی از $J(\pi_\theta)$ بروزرسانی می‌کنند. این روش تقریبا همیشه به صورت on-policy عمل می‌کند.

همانطور که در ادامه خواهیم دید، می‌توان از توابع مختلفی برای مقیاس عملکرد J استفاده نمود. یک انتخاب بدیهی $J(\pi_\theta) = \mathbb{E}[R_t]$ است. هدف این روش‌ها بیشینه کردن تابع J است. در عبارت

$$\theta_{t+1} = \theta_t + \alpha \nabla \hat{J}(\theta_t)$$

$\nabla \hat{J}(\theta_t)$، تخمینی احتمالاتی است که امیدریاضی آن گرادیان J را نسبت به پ θ_t تخمین می‌زند.

به روش‌هایی که چنین الگویی را برای محاسبه خط‌مشی بهینه دنبال می‌کنند، روش‌های *گرادیان خط‌مشی* [۱۳] می‌گوییم. دسته‌ای از روش‌های گرادیان خط‌مشی وجود دارند که تلاش می‌کنند تخمینی از تابع ارزش را نیز محاسبه کنند. به چنین روش‌هایی، ***بازیگرـ منتقد*** [۱۴] گفته می‌شود که ***بازیگر*** [۱۵] اشاره به خط‌مشی آموخته شده و ***منتقد*** [۱۶] اشاره به تابع ارزش آموخته شده (معمولا یک تابع ارزشِ حالت) دارد.

قضیه‌ی ۳ـ۱ (گرادیان خط‌مشی)

$$\nabla J(\pi_\theta) \propto \sum_s \mu(s) \sum_a q_\pi(s,a) \nabla_\theta \pi_\theta(a|s)$$

[10] Parameterized
[11] Gradian ascent
[12] Performance Measure
[13] Policy Gradient
[14] Actor-Critic
[15] Actor
[16] Critic

الگوریتم ۵ الگوریتم DDPG

۱: پارامترهای θ_μ و θ_Q به ترتیب مربوط به بازیگر $\mu(s;\theta_\mu)$ و منتقد $Q(s,a;\theta_Q)$ را مقداردهی اولیه کن.

۲: پارامترهای توابع هدف μ' و Q' را با وزن‌های $\theta_\mu \longrightarrow \theta_{\mu'}$ و $\theta_Q \longrightarrow \theta_{Q'}$ مقداردهی اولیه کن

۳: حافظه تکرارها R را بساز برای هر اپیزود M...۱

۴: یک تابع نویز تصادفی $\mathbb{N}$ بساز

۵: حالت اولیه $S_۱$ را مشاهده کن برای $t = ۱...T$

۶: عمل $a_t = \mu(s_t;\theta_\mu) + \mathbb{N}_t$ را بر اساس خط‌مشی فعلی و نویز اکتشاف، انتخاب کن و حالت بعدی $S_{t+۱}$ و پاداش R_t را مشاهده کن.

۷: تجربه $(s_t, a_t, r_t, s_{t+۱})$ را در انبار تجربه R ذخیره کن

۸: یک نمونه به اندازه N از تجربه‌های $(s_i, a_i, r_i, s_{i+۱})$ از انبار تجربه R انتخاب کن

۹: وزن‌های منتقد θ_Q را با درنظر گرفتن تابع هزینه $L = \frac{۱}{N}\sum_i (y_i - Q(S_i, A_i;\theta_Q))^۲$ بروزرسانی کن

۱۰: وزن‌های بازیگر θ را با استفاده از گرادیان خط‌مشی نمونه

$$\nabla_{\theta_\mu} J \approx \frac{۱}{N}\sum_i \nabla_a Q(s,a;\theta_Q)|_{s=s_i, a=\mu(s_i)} \nabla_\mu \mu(s;\theta_\mu)|_{S_i}$$

به روزرسانی کن

۱۱: وزن‌های توابع هدف را به شکل

$$\theta_{Q'} = \tau\theta_Q + (۱-\tau)\theta_{Q'}\theta_{\mu'} = \tau\theta_\mu + (۱-\tau)\theta_{\mu'}$$

بروزرسانی کن

۱۳:

۱۴:

که μ یک توزیع احتمال روی S است که متناسب با تعداد دفعاتی است که حالت s با دنبال کردن خط‌مشی π_θ تکرار می‌شود.

می‌توان نشان داد

$$\nabla J(\pi_\theta) = \mathbb{E}_\pi \left[R_t \frac{\nabla_\theta \pi_\theta(a|S_t)}{\pi_\theta(a|S_t)} \right]$$

بنابراین در هر گام $\left[R_t \frac{\nabla_\theta \pi_\theta(a|S_t)}{\pi_\theta(a|S_t)} \right]$ یک تخمین‌گر ناااریب از $\nabla J(\pi_\theta)$ خواهد بود[۱]. پس می‌توان در هر گام θ را به شکل زیر بروزرسانی کرد

$$\theta_{t+1} = \theta_t + \alpha R_t \frac{\nabla_\theta \pi_\theta(a|S_t)}{\pi_\theta(a|S_t)} = \theta_t + \alpha R_t \nabla_\theta log \pi_\theta(a|S_t)$$

چند نمونه از روش‌های بهینه‌سازی خط‌مشی به شرح زیر است.

روش‌های بازیگر منتقد [۱۷] که الگوریتم گرادیان افزایشی را مستقیما برای بیشینه سازی $J(\pi\theta)$ به کار می‌برند.

روش Proximal Policy Optimization که!!!!!!!!!!

روش‌های بازیگر ـ منتقد

یک نمونه از روش‌های بهینه‌سازی خط‌مشی، خانواده REINFORCE از الگوریتم‌های یادگیری تقویتی است [۲]. الگوریتم استاندارد REINFORCE پارامترهای θ را در جهت $\nabla_\theta \ log \ \pi(a_t|s_t \ : \ \theta)R_t$ بروزرسانی می‌کند که یک تخمین ناااریب از $\nabla_\theta \mathbb{E}[R_t]$ است. می‌توان با تفریق یک تابع آموخته شده روی حالت‌ها، $b_t(s_t)$ از R_t، واریانس این تخمین را کاهش داد بطوری‌که ناااریب باقی بماند. به چنین تابعی **پایه** گفته می‌شود. در نتیجه گرادیان به شکل $\nabla_\theta \ log \ \pi(a_t|s_t;\theta)(R_t - b_t(s_t))$ خواهد بود. معمولا از تخمینی آموخته‌شده از تابع ارزش به عنوان پایه استفاده می‌شود، $b_t(s_t) \approx v_\pi(s_t)$، که منجر به تخمینی با واریانس بسیار کوچک‌تر از گرادیان خط‌مشی می‌شود؛ در حالیکه تخمین ناااریب باقی می‌ماند و درنتیجه عملیات یادگیری با سرعت بیشتری انجام می‌شود. این روش می‌تواند به شکل معماری **بازیگر ـ منتقد** تعبیر شود که خط‌مشی π **بازیگر** و پایه b_t **منتقد** است. اگر از تخمین یک تابع ارزشِ حالت به عنوان پایه استفاده کنیم $b_t(s) = v_{\pi_{\theta_t}}(s)$ عبارت $R_t - b_t(s)$ می‌تواند به شکل تخمینی از **مزیت** [۱۸] عمل a_t در

[17] Actor-Critic
[18] Advantage

حالت s_t یا $A(a_t, s_t) = Q(a_t, s_t) - v(s_t)$ تعبیر شود. چراکه R_t تخمینی از $Q_\pi(a_t, s_t)$ و b_t تخمینی از $v_\pi(s_t)$ است. در این صورت به این روش Advantage Actor-Critic یا A۲C گفته می‌شود.

الگوریتم ۶ الگوریتم $Advantage\ Actor-Critic$ [۱]

ورودی: یک پارامتری‌شده مشتق پذیر از خط‌مشی $\pi_\theta(a|s)$

ورودی: یک پارامتری‌شده مشتق‌پذیر از تابع ارزشِ حالت $v_\omega(s)$

۱: پارامترهای خط‌مشی $\theta \in \mathbb{R}^{d'}$ و تابع ارزشِ حالت $\omega \in \mathbb{R}^d$ را مقداردهی اولیه کن

۲: تکرار کن:

۳: حالت اولیه S را بساز

۴: $۱ \longrightarrow I$

۵: تا وقتی S حالت نهایی نیست:

۶: $A \sim \pi_\theta(.|S)$

۷: عمل A را انجام بده و حالت S' و پاداش R را مشاهده کن

۸: $R + \gamma v_\omega(S') - v_\omega(S) \longrightarrow \delta$

۹: $\omega + \alpha^\omega I \delta \nabla_\omega v_\omega(S) \longrightarrow \omega$

۱۰: $\theta + \alpha^\theta I \delta \nabla_\theta \, ln\, \pi_\theta(A|S) \longrightarrow \theta$

۱۱: $\gamma I \longrightarrow I$

۱۲: $S' \longrightarrow S$

روش TRPO

تعریف ۳ـ۱ *اگر $\rho_\pi : S \longrightarrow \mathbb{R}$ تابع فرکانس تخفیف‌دار دیده شدن حالت‌ها باشد. یعنی*

$$\rho_\pi(s) = P(S_۰ = s) + \gamma P(S_۱ = s) + \gamma^۲ P(S_۲ = s) + ...$$

که دنباله $S_۰, S_۱, S_۲$ خط‌مشی π را دنبال می‌کند.

اگر π و π' دو خط‌مشی باشند و $J(\pi) = \mathbb{E}_\pi[R_\cdot]$ می‌توان نشان داد

$$J(\pi') = J(\pi) + \sum_s \rho_{\pi'}(s) \sum_a \pi'(a|s) A_\pi(s,a)$$

که $A_\pi(s,a) = Q_\pi(s,a) - V_\pi(s)$ تابع مزیت عمل a در حالت s باشد. وابستگی پیچیده $\rho'_\pi(s)$ در طرف راست تساوی به π' بهینه‌سازی مستقیم را مشکل می‌کند. برای حل این مشکل شولمن و همکارانش مقیاس عملکرد دیگری، $L_\pi(\pi')$، را معرفی می‌کند و نشان می‌دهد که اگر π و π' به اندازه کافی به یکدیگر نزدیک باشند، افزایش $L_\pi(\pi')$ همواره با افزایش J همراه خواهدبود [۳].

$$L_\pi(\pi') = J(\pi) + \sum_s \rho_\pi(s) \sum_a \pi'(a|s) A_\pi(s,a) \qquad (۲\_۳)$$

در عبارت ۳_۲ L_π از تابع فرکانس ρ_π به جای $\rho_{\pi'}$ استفاده می‌کند.

قضیه‌ی ۳_۲ *فرض کنید $\alpha = D^m_{TV}ax(\pi_{old}, \pi_{new})$ باشد که*

$$D^m_{TV}ax(\pi, \pi') = \max_s D_{TV}(\pi(.|s)||\pi'(.|s)$$

و $D_{TV}(p||q)$ دیورژانس total variation بین دو بردار p و q باشد

$$D_{TV}(p||q) = \frac{۱}{۲} \sum_i |p_i - q_i|$$

در این صورت

$$J(\pi_{new}) \geqslant L_{\pi_{old}}(\pi_{new}) - \frac{۴\epsilon\gamma}{(۱-\gamma)^۲}$$

که $\epsilon = \max_{s,a} |A_\pi(s,a)|$.

با توجه به قضیه ۳_۲ و نامعادله $D_{TV}(p||q)^۲ \leqslant D_{KL}(p||q)$ که $D_{KL}(p||q)$ برابر با دیورژانس KL دو بردار p و q است[۳]. می‌توان نتیجه گرفت

$$J(\pi') \geqslant L_\pi(\pi') - C\ D^m_{KL}ax(\pi, \pi')$$

که

$$C = \frac{۴\epsilon\gamma}{(۱-\gamma)^۲} \qquad (۳\_۳)$$

رابطه ۳ـ۳ نشان می‌دهد که می‌توان یک دنباله صعودی از خط‌مشی‌ها داشت به طوری که

$$J(\pi_۰) \leqslant J(\pi_۱) \leqslant J(\pi_۲) \leqslant ...$$

چرا که فرض کنید $M_i(\pi) = L_{\pi_i}(\pi) - C\ D^m_{KL}ax(\pi_i, \pi)$ در این صورت

$$J(\pi_{i+۱}) \geqslant M_i(\pi i + ۱)$$

$$J(\pi_i) = M_i(\pi_i)$$

بنابراین

$$J(\pi_{i+۱}) - J(\pi_i) \geqslant M_i(\pi_{i+۱}) - M_i(\pi_i)$$

می‌توان نتیجه گرفت با بیشینه کردن M_i در هر گام می‌توان اطمینان حاصل کرد که مقیاس عملکرد واقعی J غیرنزولی خواهد بود.

الگوریتم ۷ الگوریتم $PolicyIteration$ با مقیاس عملکرد L_π

۱: خط‌مشی $\pi_۰$ را مقداردهی اولیه کن

۲: برای $i = ۰, ۱, ...$ تکرار کن:

۳: همه مزیت‌های $A_{\pi_i}(s, a)$ را محاسبه کن

۴: $\pi_{i+۱} = arg\max_\pi [L_{\pi_i}(\pi) - C\ D^m_{KL}ax(\pi_i, \pi)]$ که

$$C = (۴\epsilon\gamma)/(۱ - \gamma)^۲$$

و

$$L_{\pi_i}(\pi) = J(\pi_i) + \sum_s \rho_{\pi_i}(s) \sum_a \pi(a|s) A_{\pi_i}(s, a)$$

اگر $\hat{A}_t$ تخمین مزیت $A_{\pi_t}(S_t, A_t)$ باشد که در گام t محاسبه می‌شود، می‌توان نشان داد که در روش TRPO مقیاس عملکرد L_π در هر گام به شکل؛

$$\mathbb{E}_t \left[\frac{\pi_{\theta_{new}}(A_t|S_t)}{\pi_{\theta_{old}}(A_t|S_t)} \hat{A}_t \right]$$

خواهد بود.

روش PPO

در روش TRPO دیدیم که بیشینه‌سازی مقیاس عمکلرد L_π ساده‌تر از مقیاس عملکرد J است ولی در عوض الگوریتم صعود گرادیان تنها مجاز به اعمال تغییرات کوچک در خط‌مشی است. یک راه دیگر برای کنترل تغییرات خط‌مشی استفاده از تابع CLIP است. در روش TRPO دیدیم که تابع مقیاس عملکرد، در گام t به شکل زیر است؛

$$L_{\pi_{old}}(\pi_{new}) = \mathbb{E}_{\approx}\left[\frac{\pi_{\theta_{new}}(A_t|S_t)}{\pi_{\theta_{old}}(A_t|S_t)}A_\pi(S_t,A_t)\right]$$

فرض کنید $r_t(\theta_{new})$ نسبت احتمالات $r_t(\theta_{new}) = \frac{\pi_{\theta_{new}}(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}$ باشد. بنابراین

$$L_{\pi_{old}}(\pi_{new}) = \mathbb{E}_t\left[\frac{\pi_{\theta_{new}}(A_t|S_t)}{\pi_{\theta_{old}}(A_t|S_t)}A_\pi(S_t,A_t)\right] = \mathbb{E}_t\left[r_t(\theta_{new})A_{\pi_{old}}(S_t,A_t)\right].$$

مقیاس عملکرد $L^{CLIP}(\theta)$ را به شکل زیر تعریف می‌کنیم

$$L^{CLIP}(\theta) = \mathbb{E}_t\left[min(r_t(\theta)\hat{A}_t, clip(r_t(\theta), ۱-\epsilon, ۱+\epsilon)\hat{A}_t\right]$$

که ϵ یک ابرپارامتر [۱۹] مثلا $\epsilon = ۰/۲$ است. اولین عبارت داخل min همان مقیاس عملکرد روش TRPO است. در عبارت دوم مقادیر بزرگتر از $۱+\epsilon$ یا کوچکتر از $۱-\epsilon$ در $r_t(\theta)$ به ترتیب به $۱+\epsilon$ و $۱-\epsilon$ تغییر پیدا کرده‌اند تا تغییرات بزرگ خط‌مشی را کنترل کنند. نهایتا عبارت کوچکتر از میان این دو انتخاب خواهد شد. بنابراین اگر مقیاس عملکرد clip نشده (عبارت اول) کوچکتر یا مساوی با حالت clip شده (عبارت دوم) باشد، L^{CLIP} دقیقا همان مقیاس عملکرد L_π خواهد بود. در غیر این صورت مقیاس عملکرد clip شده انتخاب می‌شود تا از تغییرات بزرگ خط‌مشی جلوگیری شود.

۳ـ۲ـ۳ مقایسه روش بهینه‌سازی خط‌مشی و Q-learning

نقطه قوت اصلی روش‌های بهینه‌سازی خط‌مشی، اصولی بودن آنهاست ، به این معنا که شما مستقیماً چیزی که می‌خواهید را بهینه‌سازی می‌کنید. در نتیجه این روش‌ها قابل اتکا و باثبات هستند. در مقابل، روش‌های Q-learning با یادگیری تابع Q، مقیاس عملکرد را به طور غیر مستقیم بهینه می‌کند. حالت‌های زیادی برای این نوع یادگیری وجود دارد که به شکست منتهی می‌شود، بنابراین این روش‌ها ثبات کمتری دارند [۱]. روش‌های Q-learning می‌توانند از داده‌ها به طور موثرتری نسبت به تکنیک‌های بهینه‌سازی خط‌مشی استفاده کنند.

[19] Hyperparameter

تعامل بین بهینه‌سازی خط‌مشی و Q-learning بهینه‌سازی خط‌مشی و Q-learning ناسازگار نیستند(و به نظر می‌رسد تحت برخی شرایط، معادل آن باشد) و طیف وسیعی از الگوریتم‌ها وجود دارد که بین دو حد این طیف وجود دارند. الگوریتم‌هایی که در این طیف قرار دارند قادرند از نقاط قوت طرفین طیف استفاده کنند.

به طور مثال، DDPGالگوریتمی‌است که یک خط‌مشی قطعی و یک تابع Q را یاد می‌گیرد، به طوری که از هریک برای بهبود دیگری استفاد می‌کند. روش SAC، از خط‌مشی‌های تصادفی، تنظیم آنتروپی [۲۰] و چند ترفند دیگر برای یادگیری و کسب امتیاز بالاتر از DDPG در محک‌های استاندارد [۲۱] استفاده می‌کند.

۳ـ۳ روش‌های مبتنی بر مدل

۳ـ۳ـ۱ روش مدل جهان

مدل جهانی [۲۲] می‌تواند به سرعت و به روشی بدون نظارت آموزش ببیند تا یک بازنمایی از محیط را بیاموزد. سپس با استفاده از ویژگی‌های استخراج شده از مدل جهان به عنوان ورودی به یک عامل، می‌توان یک خط‌مشی ساده و فشرده را آموخت که می‌تواند وظیفه مورد نیاز را حل کند. حتی می‌توانیم عامل را کاملاً در داخل محیط رویایی خود که توسط مدل جهانی آن ایجاد شده، آموزش دهیم و این خط‌مشی آموخته شده را به محیط واقعی انتقال دهیم. در بسیاری از مسائل یادگیری تقویتی مبتنی بر مدل، عامل به مدل قدرتمندی از دینامیک محیط دسترسی دارد.

اکثر رویکردهای مبتنی بر مدل موجود در یادگیری تقویتی، مدلی از محیط را یاد می‌گیرند، اما همچنان در محیط واقعی آموزش می‌بینند. در این روش، ما همچنین می‌توانیم یک محیط مصنوعی را کاملاً جایگزین محیط واقعی کنیم و خط‌مشی عامل خود را فقط در داخل محیط مصنوعی آموزش دهیم و درنهایت خط‌مشی آموخته شده را به محیط واقعی انتقال دهیم.

[20] entropy regularization
[21] Standard Benchmark
[22] World Models

پیوست آ

مطالب تکمیلی

پیوست‌های خود را در صورت وجود می‌توانید در این قسمت قرار دهید.

مراجع

[1] R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[2] R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

[3] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz. Trust region policy optimization. In *International conference on machine learning*, pages 1889–1897, 2015.

واژه‌نامه

الف

ابتکاری ..... heuristic
ارزش ..... worth
ارضاپذیری ..... satisfiability
استراتژی ..... strategy
ائتلاف ..... coalition

ب

بارگذاری ..... loading
بازی ..... game
برچسب ..... label
برنامه‌ریزی خطی ..... linear programming
برنامه‌ریزی صحیح ..... integer programming
بسته‌بندی ..... packing
بهترین پاسخ ..... best response
بیشینه ..... maximum

پ

پالت ..... pallet
پایداری ..... robustness
پشتیبان ..... support
پوسته‌ی محدب ..... convex hull
پوش بالایی ..... upper envelope
پوششی ..... covering

ت

تبدیل تصویری ..... projective transformation
تعادل ..... equlibrium
تعدیل ..... relaxation
تقاطع ..... intersection
تقسیم‌بندی ..... partition
تکاملی ..... evolutionary
توزیع‌شده ..... distributed

ج

جست‌وجوی جامع ..... brute-force
جست‌وجوی عمق‌اول ..... Depth-First Search

جعبه ..... bin

چ

چاله ..... sink

ح

حرکت ..... action

خ

خودخواهانه ..... selfish
خوشه ..... clique

د

دودویی ..... binary
دوگان ..... dual
دو ماتریسی ..... bimatrix

ر

رأس ..... vertex
رفتار ..... behaviour
رنگ‌آمیزی ..... coloring

ز

زمان‌بندی ..... scheduling
زیست‌شناسی ..... biology

س

ساختی ..... constructive
سود ..... pay off, utility

ش

شبه‌چندجمله‌ای ..... quasi-polynomial
شبه‌مقعر ..... quasi-concave

ص

صوری ..... formal

ع

عاقل ..... rational
عامل‌ـ‌محور ..... agent-based
عمل ..... action

غ

غائب ..... missing
غیرمتمرکز ..... decentralized
غیرمعمول ..... degenerate

ق

قابل انتقال ..... transferable
قاموسی ..... lexicographically
قوی ..... strong

ک

کمینه ..... minimum

م

مجموع زیرمجموعه‌ها ..... subset sum
مجموعه ..... set
محور ..... pivot
مختلط ..... mixed
مخفی ..... hidden
مستوی ..... affine
مسطح ..... planar
منطقی ..... reasonable
موازی ..... parallel

ن

نتیجه‌ی نهایی ..... outcome
نش ..... Nash
نقطه‌ثابت ..... fixed point
نگارخانه‌ی هنر ..... art gallery
نگهبان ..... gaurd
نمایه ..... profile
نوبتی ..... round-robin

و

وجه ..... facet

هـ

هزینه‌ی آشوب ..... price of anarchy (POA)
هزینه‌ی اجتماعی ..... social cost
هزینه‌ی پایداری ..... price of stability (POS)

ی

یال ..... edge
یکریختی ..... isomorphism

Abstract

We present a standard template for typesetting theses in Persian. The template is based on the XePersian package for the LaTeX typesetting system. This write-up shows a sample usage of this template.

Keywords: Thesis, Typesetting, Template, XePersian

Sharif University of Technology

Department of Computer Engineering

M.Sc. Thesis

A Standard Template for Typesetting Theses in Persian

By:

Hamid Zarrabi-Zadeh

Supervisor:

Dr. Supervisor

September 2017