کنفرانس ملی فناوری، انرژی و داده با رویکرد مهندسی برق و کامپیوتر
National Conference of Technology, Energy and Data on Electrical & Computer Engineering
کونفرانسی نەتەوەیی فەناوەری ەێز و دەیتا بە روانگە ئەندازیاری کارەبا و کومپیتر

اکتساب مهارت در یادگیری تقویتی و الگوریتم‌های آن

مریم زارع[۱] و علیرضا خلیلیان[۲*]
[۱] کارشناس نرم‌افزار، دانشگاه فنی و حرفه‌ای، دانشکده فنی دکتر شریعتی
zare.maryam1992@gmail.com
[۲*] دانشجوی دکتری نرم‌افزار، دانشگاه اصفهان
khalilian@eng.ui.ac.ir

چکیده: یادگیری تقویتی یکی از حوزه‌های یادگیری ماشین است که هدف آن بهبود رفتار عامل هوشمند بر اساس سیگنال‌های تقویتی است که از محیط دریافت می‌کند. تنها مسیر اطلاع‌رسانی به عامل در یادگیری تقویتی، از راه سیگنال پاداش یا جریمه می‌باشد. سیگنال پاداش به عامل می‌فهماند که آیا تصمیم مناسبی گرفته است یا خیر. عامل موظف است با در دست داشتن این اطلاعات یاد بگیرد که بهترین عمل کدام است. یکی از مشکلات یادگیری تقویتی این است که با پیچیده‌تر شدن محیط، تعداد پارامترهای تصمیم‌گیری افزایش می‌یابد و زمان یادگیری نیز بیشتر می‌شود. تنظیم درست پارامترها اولین قدم در کاهش سرعت یادگیری است. هدف از این مقاله، مروری بر ادبیات یادگیری تقویتی، مفاهیم اصلی، روش‌ها و الگوریتم‌های آن و مفهوم پاداش شکل‌دهی شده است. به‌منظور مشاهده و بررسی تاثیر برخی پارامترها در اجرای الگوریتم‌ها روی محیط‌های مختلف، همچنین نتیجه استفاده از پاداش شکل‌دهی شده، برخی از الگوریتم‌های یادگیری تقویتی در قالب نرم‌افزار شبیه‌ساز طراحی و پیاده‌سازی شده است. سپس آزمایش‌هایی روی چند محیط محک همچون maze و شش اتاقه انجام شده و نتایج گزارش شده‌اند.

کلمات کلیدی: یادگیری تقویتی، پاداش ساختگی، یادگیری کیو، سارسا، R-max.

۱- مقدمه

یادگیری، درحالت کلی به سه دسته تقسیم می‌شود: الف- یادگیری نظارت‌شده: در این روش یک معلم یا ناظر وجود دارد که بهترین عمل را در هر وضعیت می‌داند و توصیه‌هایی را برای تصحیح شیوه‌ی عملکرد عامل ارائه می‌دهد [۱]. ب- یادگیری غیر نظارت‌شده: در این نوع یادگیری عامل یادگیرنده، می‌تواند تشخیص دهد که آن‌چه دریافت کرده را به نوعی به آن‌چه پیش‌تر دیده است ربط دهد. در این نوع یادگیری هدف تنها دسته‌بندی ورودی‌هاست [۲]. ج- یادگیری تقویتی: در این نوع یادگیری بازخوردی به صورت عبارات کمکی مثبت (پاداش) یا منفی (جریمه) به عامل یادگیرنده داده می‌شود [۱]. در یادگیری تقویتی هیچ‌گاه به عامل گفته نمی‌شود که عمل صحیح در هر وضعیت چیست و فقط به وسیله‌ی معیاری به عامل گفته می‌شود که یک عمل چقدر خوب یا چقدر بد است [۱].

یادگیری تقویتی الگوریتم‌های متعددی دارد که در هر کدام باید پارامترهای بسیاری تنظیم شوند. به‌علاوه هرچه محیط بزرگتر می‌شود، زمان یادگیری تقویتی نیز بیشتر می‌گردد. یکی از راه‌های حل مشکل اول، برپایی آزمایش‌های گوناگون روی محیط‌هایی با اندازه‌های متفاوت است تا اثر هر پارامتر بررسی گردد. ضمن اینکه تاکنون برای مقایسه عملکرد روش‌های جدید با هر روش پیشنهادی، هر شخصی خودش الگوریتم‌ها را پیاده‌سازی کرده است.

این مقاله یادگیری تقویتی، مفاهیم و الگوریتم‌های آن‌را معرفی می‌کند. سپس مشخصات نرم‌افزار شبیه‌ساز طراحی شده برای آزمایش‌های یادگیری را بیان می‌کند و در انتها نیز نتیجه اجرای شبیه‌سازی روی دو محیط محک با اندازه‌های مختلف گزارش شده است.

ساختار ادامه مقاله به این شرح است: در بخش دوم یادگیری تقویتی، الگوریتم کلی آن، مفاهیم تابع ارزش و پاداش و محیط معرفی می‌شود. بخش‌های سوم و چهارم خاصیت مارکوف و فرایند تصمیم‌گیری ماکوف را تشریح می‌کنند. بخش پنجم به کاربردهای یادگیری تقویتی اختصاص دارد. در بخش‌های ششم و هفتم هم به‌ترتیب روش‌ها و الگوریتم‌های یادگیری تقویتی مورد بحث قرار می‌گیرد. بخش هشتم مفهوم پاداش شکل‌دهی شده را معرفی می‌کند. در بخش نهم مشخصات نرم‌افزار شبیه‌ساز ارائه می‌شود. بخش دهم نتایج آزمایش‌های صورت‌گرفته را گزارش می‌کند. بخش یازدهم هم به نتیجه‌گیری اختصاص پیدا کرده است.

۲- یادگیری تقویتی[۱]

در یک مسئله یادگیری تقویتی با عاملی رو به رو هستیم که از طریق سعی و خطا با محیط تعامل کرده و یاد می‌گیرد تا عملی بهینه را برای رسیدن به هدف انتخاب کند [۳].

در این نوع یادگیری هیچ ناظر خارجی وجود ندارد و عامل به‌تنهایی با محیط تعامل کرده، یاد می‌گیرد و تجربه کسب می‌کند و پاداشی دریافت می‌کند [4].

در یادگیری تقویتی عامل‌ها به حسگرهایی مجهز شده‌اند که می‌توانند ویژگی‌های قطعی محیط را دریافت کنند. این ویژگی‌ها فضای حالت[۲] عامل یادگیرنده را تشکیل می‌دهند. سپس در هر بازه زمانی عامل با انجام عملیاتی، محیط را تحت تاثیر قرار می‌دهد. بنابراین عامل ورودی‌های مختلف را در بازه زمانی بعدی با توجه به عمل[۳] قبلی دریافت می‌کند. علاوه‌بر ورودی جدید، عامل یادگیرنده در هر زمان یک سیگنال تقویتی که میزان مطلوبیت کنش قبلی را نشان می‌دهد، دریافت می‌کند که به آن پاداش می‌گویند و با توجه به این که کنش مناسب بوده یا خیر، این مقدار می‌تواند مثبت یا منفی باشد [۳].

به طور کلی، تمام عامل‌های یادگیری تقویتی هدف آشکاری دارند. می‌توانند محیط خود را درک کنند، اعمالی را انتخاب کنند تا در محیط‌شان تاثیر بگذارند. با محیط خود تعامل دارند، عملی انجام می‌دهند و پاداشی دریافت می‌کنند. با وجود این ممکن است محیط پیرامون خود را به‌طور کامل نشناسند. از آن‌جایی که عامل با محیط خود در تعامل است، پس اعمال او در محیط و وضعیت‌های بعدی تاثیرگذار است. پس عامل باید به طور متناوب محیطش را نظارت کند و به‌درستی واکنش نشان دهد [4].

۲-۱- الگوریتم یادگیری تقویتی

الگوریتم کلی در یادگیری تقویتی به شرح زیر است:

۱. مشاهده حالت فعلی
۲. تصمیم گرفتن برای انجام کنش
۳. انجام کنش
۴. دریافت سیگنال تقویتی
۵. مشاهده حالت جدید
۶. یادگیری از تجربیات
۷. تکرار [۳]

۲-۲- اجزای اصلی یادگیری تقویتی

یادگیری تقویتی چهار جزء اصلی و اساسی دارد که در ادامه هر کدام بیان شده اند:

۲-۲-۱- سیاست[۴]

سیاست یا راهبرد عامل، شیوه‌ی رفتار کردن عامل را تعریف می‌کند. در واقع سیاست، نگاشتی بین وضعیت دریافت شده از محیط و اعمال قابل انجام در آن وضعیت است. این نگاشت به عامل می‌گوید که در مواجهه با حالات مختلف، چه عمل یا اعمالی را انجام دهد. پیروی از یک سیاست خوب، قطعا عامل را به نتیجه‌ای مناسب خواهد رساند [۱]. به بیانی دیگر، سیاست، π، تابع احتمالی است که احتمال انتخاب شدن هر کنش را در هر حالت و با توجه به گام زمانی می‌دهد. به‌طور مثال، $\pi_t(s,a)=p$، می‌گوید که اگر عامل در زمان t و در حالت s قرار گرفته باشد، با احتمال p، کنش a را انتخاب می‌کند [۵]. علاوه براین، در بعضی موارد، سیاست می‌تواند یک جدول ساده جستجو یا فرآیندهای سنگین جستجو باشد. سیاست، هسته‌ی یادگیری تقویتی بوده و به‌تنهایی برای تعیین رفتار عامل کافی است [4].

۲-۲-۲- تابع پاداش[۵]

این تابع، وضعیت یا (وضعیت-کنش) دریافت شده از محیط را به یک سیگنال عددی به نام پاداش نگاشت می‌دهد. این تابع با پاداش یا جریمه، تعیین می‌کند که کدام عمل برای عامل خوب و کدام عمل بد است. هدف اصلی عامل این است که پاداش هایی که در طولانی مدت به دست می‌آورد، بیشینه کند. همچنین ممکن است از این تابع به عنوان اصلی برای تعویض سیاست استفاده شود. مثلا اگر یک عمل انتخاب شده توسط سیاستی پاداش کمی به همراه داشته باشد، این سیاست ممکن است عوض شود تا در آینده عمل دیگری در آن وضعیت انتخاب گردد [4].

۲-۲-۳- تابع ارزش[۶]

ارزش یک حالت برابر است با مجموع مقادیر پاداش دریافتی با شروع از آن حالت و پیروی از سیاست مشخصی که به یک حالت پایانی (هدف) ختم شود [4].

تابع ارزش عبارت است از نگاشتی میان حالت و ارزش آن که می‌تواند توسط هر تقریب‌زننده تابع نظیر یک شبکه عصبی تخمین زده شود [۶].

تابع دیگری به نام $Q^{\pi}(s,a)$ وجود دارد که بیان‌گر مجموع پاداش‌هایی است که عامل با شروع از حالت s و انجام کنش a و در پیش گرفتن سیاست π به‌دست می‌آورد. این تابع، تابع ارزش-کنش برای سیاست π نام دارد [۵].

۲-۲-۴- محیط[۷]

عامل یادگیرنده با سعی و خطا با یک محیط پویا درگیر شده و یاد می‌گیرد که برای هر وضعیت چه عملی انجام دهد. در واقع، هر چیزی که خارج از عامل است و عامل با آن تعامل دارد، محیط نامیده می‌شود.

- این محیط باید قابل مشاهده و یا حداقل تا قسمتی قابل مشاهده باشد.
- مشاهده محیط ممکن است از طریق خواندن اطلاعات یک حسگر و توضیح نمادین و ... باشد.
- در حالت ایده آل عامل باید به‌طور کامل قادر به مشاهده محیط باشد، زیرا اغلب تئوری‌ها بر اساس این فرض بنا شده‌اند [۶].

۳- خاصیت مارکوف[۸]

در یادگیری تقویتی، عامل بر اساس سیگنال دریافتی از محیط، که به آن حالت محیط گفته می‌شود، تصمیم می‌گیرد. فرض کنید سیگنال s_t نشان‌دهنده حالت محیط در لحظه t است.

مطلوب این است که s_t تمام داده‌های مفید مربوط به حال و گذشته را در خود خلاصه کند. برای نیل به این هدف، چیزی فراتر از یک درک آنی لازم است، ولی این به مفهوم داشتن تمام تاریخ گذشته نیز نمی‌باشد. به سیگنال حالتی که چنین باشد، گفته می‌شود، دارای خاصیت مارکوف است. به‌عنوان مثال، چیدمان مهره‌ها روی صفحه شطرنج دارای خاصیت مارکوف است. زیرا با وجود این‌که تمام حرکت‌هایی که از اول بازی تا حال شده است را به بازیکن نمی‌دهد ولی تمام داده‌های مفید برای ادامه بازی را در خود دارد؛ از سیگنال حالت نباید انتظار رود تمام داده‌های محیط را برای تصمیم‌گیری عامل نمایش دهد؛ برای مثال، اگر عامل به تماس‌های دریافتی پاسخ می‌دهد، نباید انتظار داشت اطلاعاتی در مورد تماس گیرنده داشته باشد [۵].

۴- فرآیند تصمیم‌گیری مارکوف[۹]

به مسئله یادگیری تقویتی که در آن خاصیت مارکوف برقرار باشد، فرآیند تصمیم‌گیری مارکوف گویند. اگر فضای حالات و مجموعه کنش‌های ممکن متناهی باشد، به فرآیند تصمیم‌گیری، مارکوف متناهی گفته می‌شود [4].

به عبارت دیگر، فرآیند تصمیم‌گیری مارکوف (MDP)، مدلی برای تصمیم‌گیری‌های ترتیبی می‌باشد و برای زمانی مورد استفاده است که خروجی‌ها به صورت غیر قطعی باشند. منظور از تصمیم‌گیری‌های ترتیبی این است که کارایی عامل به مجموعه‌ای از تصمیمات که به‌صورت متوالی اتخاذ کرده است، بستگی دارد و تنها وابسته به تصمیم فعلی او نخواهد بود.

فرآیند تصمیم‌گیری مارکوف، در حالت کلی به صورت یک مجموعه شش‌تایی $(S, A, T, Next, R, \alpha)$ نشان داده می‌شود که S، مجموعه وضعیت‌های ممکن در محیط، A، مجموعه کنش‌های ممکن برای عامل در هر مرحله از تصمیم‌گیری، T، نمایان‌گر احتمال گذر عامل از وضعیت s به s' است، به شرط انتخاب عمل a از مجموعه اعمال ممکن، $Next$، : نشان‌دهنده مجموعه وضعیت‌هایی است که می‌توان در یک قدم با احتمال غیر صفر و انتخاب عمل a به آن رسید، $R(s,a)$، نمایان‌گر مقدار پاداشی است که عامل به ازای انجام عمل a در وضعیت s، از محیط دریافت می‌کند و α، یک فاکتور کاهنده [۶].

۵- کاربردهای یادگیری تقویتی

از گذشته تا به‌حال از یادگیری تقویتی در زمینه‌های مختلفی استفاده شده است که مثال‌هایی از آن در این‌جا آورده شده است:

- **سامانه‌های چندعامله**: سامانه‌های چندعامله، به‌عنوان الگویی از هوش مصنوعی توزیع‌شده هستند. این سامانه‌ها با ساختاری متشکل از چند عامل که رفتارهای عامل‌های مختلف باید در یک محیط به سمت یک هدف جامع باهم هماهنگ شوند، بیان می‌شوند. همچنین، کنترل سامانه‌های چندعامله به‌نوعی هم متمرکزشده و هم توزیع‌شده است. فرآیند تصمیم‌گیری هر عامل باید به‌دست خودش انجام شود که البته این تصمیم‌گیری روی مشاهداتش و دانش‌هایی که درباره محیط و عامل‌های دیگر بنا شده است، انجام می‌گیرد.
- **بازی‌ها**: از یادگیری تقویتی برای چندین بازی اصلی مثل شطرنج و تخته نرد که در جهان مطرح هستند، استفاده شده است. در سال ۱۹۵۹ یک سامانه ارائه شد که به چکرزها یاد داده می‌شود که از راه بازی کردن با خودشان بازی کنند. این روش بعدها با ایده یادگیری به نام TD-Learning[۱۰] ترکیب شد و توانست قدرت بازی چکرزها را افزایش دهد.
- **هدایت کردن ربات**: هدایت خودکار ربات‌ها در محیط‌های ناشناخته یک بستر آزمایش برای تحقیقات زیاد در این زمینه یادگیری است. در سال ۱۹۸۳ یک ربات قادر شد به کمک این یادگیری به اهداف تعیین شده با دوری کردن از موانع برسد.
- **زمان‌بندی**: یک استفاده دیگر از یادگیری تقویتی، زمان‌بندی در کارهاست که نخست در سال ۱۹۹۶ تحقیق شد و با موفقیت دیدگاه TD-Learning در مسائل زمان-بندی، این روش یادگیری در این زمینه هم مطرح شد.
- **مدیریت وزارت**: سامانه‌های پشتیبان، تصمیم برای سرمایه-گذاران و مشتری‌ها به شدت در سال‌های پیش مورد مطالعه بود. این سامانه‌ها سعی دارند که سود برگشتی برای سرمایه-گذاران را بیشینه کنند. دیدگاه اخیر از یادگیری تقویتی استفاده کرد که به طور کامل ریسک سرمایه‌گذاری را از بین برد.
- **مسیریابی**: مسیریابی در شبکه هایی با شرایط پویا می‌توانند با Q-routing به‌خوبی انجام گیرد که یک روش یادگیری تقویتی توزیع‌شده است که از Q-Learning، که در ادامه بحث خواهد شد، ناشی می‌شود [۷].

۶-روش‌های یادگیری تقویتی

روش‌های یادگیری تقویتی به سه دسته تقسیم می‌شود:

- **برنامه‌ریزی پویا**[۱۱]: به مجموعه الگوریتم‌هایی گفته می‌شود که با در درست داشتن مدل کاملی از محیط به‌عنوان فرآیند تصمیم‌گیری مارکوف می‌توانند سیاست بهینه را محاسبه کنند.

- **مونت کارلو**[۱۲]: این روش نیازی به شناخت کامل محیط ندارد و مسئله یادگیری تقویتی را بر اساس میانگین بازده‌ها حل می‌کنند. برای این منظور (به‌دست آوردن بازده مناسب و درست)، مونت کارلو فقط بر روی وظایف اپیزودیک تعریف می‌شود؛ بنابراین یک وظیفه به چند اپیزود تقسیم می‌شود و هر اپیزودی در نهایت به‌پایان می‌رسد و فقط در انتهای یک اپیزود است که تخمین مقادیر و ارزش‌ها صورت گرفته و سیاست‌ها انتخاب می‌شوند.
- **تفاضل زمانی**[۱۳]: هسته‌ی اصلی یادگیری تقویتی است. این روش ترکیبی از برنامه‌ریزی پویا و روش مونت کارلو است؛ یعنی عامل مانند روش مونت کارلو، می‌تواند بدون نیاز به مدلی از محیط پویا، به‌صورت مستقیم از تجربه‌ها یاد بگیرد و همانند برنامه‌ریزی پویا براساس یادگیری‌های دیگرش و بدون انتظار برای رسیدن به یک نتیجه نهایی برآورد خود را به‌روز رسانی نماید [۸].

۷- الگوریتم‌های یادگیری تقویتی

در ادامه سه الگوریتم یادگیری تقویتی به تفصیل بیان می‌شوند:

۷-۱- الگوریتم سارسا[۱۴]

این الگوریتم در سال ۱۹۹۴ مطرح گردید. همان‌طور که از نام الگوریتم مشخص است (حالت، کنش، پاداش، حالت، کنش)، عامل در حالت جاری s_t، کنش a_t را انتخاب می‌کند و با دریافت پاداش r، به حالت جدید s_{t+1} می‌رود و کنش a_{t+1} را انتخاب می‌کند $(s_t, a_t, r_{t+1}, s_{t+1}, a_{t+1})$.

محاسبه تابع ارزش- کنش و به‌روز رسانی آن مطابق فرمول (۱) می‌باشد:

(۱)

$$Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \alpha[r_{t+1} + \gamma Q(s_{t+1},a_{t+1}) - Q(s_t,a_t)]$$

این به‌روز رسانی پس از هر حالت غیر پایانی s_t صورت می‌گیرد. اگر حالت s_{t+1} حالت پایانی باشد، مقدار تابع $Q(s_{t+1},a_{t+1})$ تغییری نکرده و مقدار اولیه خود را حفظ خواهد کرد. این قانون برای هر پنج تایی $(s_t, a_t, r_{t+1}, s_{t+1}, a_{t+1})$ که در نهایت منجر به انتقال از حالت- کنش به حالت- کنش بعدی می‌گردد، استفاده می‌شود [۶]. در این رابطه، متغییر نرخ یادگیری α، تعیین می‌کند که تا چه میزان اطلاعات به‌دست آمده جدید بر اطلاعات قدیمی ترجیح داده شود. مقدار صفر باعث می‌شود که عامل چیزی یاد نگیرد و مقدار یک باعث می‌شود که عامل فقط اطلاعات جدید را ملاک قرار دهد. همچنین متغیر نرخ تنزیل γ، اهمیت پاداش‌های آینده را تعیین می‌کند. مقدار صفر باعث می‌شود که عامل ماهیت فرصت‌طلبانه گرفته و فقط پاداش‌های فعلی را مدنظر قرار دهد. در حالی‌که مقدار یک عامل را ترغیب می‌کند، برای یک دوره زمانی طولانی برای پاداش تقلا کند [۹].

روال زیر الگوریتم سارسا را نشان می‌دهد:

1. Initialize $Q(s,a)$ arbitarily
2. Repeat (for each episode)
3. Initialize s
4. Choose a from s using policy derived from Q (e.g, ε-greedy)
5. Repeat (for each step of episode)
6. Take a action, observe r, s'
7. Choose a' from s' using policy derived from Q (e.g, ε-greedy)
8. $Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \alpha[r_{t+1} + Q(s_{t+1},a_{t+1}) - Q(s_t,a_t)$
9. $s \leftarrow s'$; $a \leftarrow a'$;
10. Until s is terminal [4].

شرح الگوریتم:

- به ازای هر کنش a، Q را مقداردهی اولیه کن (معمولا صفر)
- در هر اپیزود تکرار کن
  - حالت فعلی s را مشاهده کن
  - کنش a را از حالت فعلی s انتخاب کن (با سیاست جاری)
  - تا مشاهده حالت پایانی تکرار کن
    - کنش a را اجرا کن، پاداش r، حالت جدید s' را مشاهده کن
    - یک کنش ممکن از حالت جدید s' را انتخاب کن (با سیاست جاری)
    - مقدار Q را برای s و a به‌روز کن
    - حالت جدید s' را به‌عنوان حالت فعلی s و کنش جدید a' را به‌عنوان کنش فعلی a در نظر بگیر.

باید توجه داشت که الگوریتم سارسا با احتمال یک به سیاست بهینه همگرا خواهد شد و تابع ارزش- کنش $Q(s,a)$ برای هر جفت حالت- کنش در تعداد تکرار نامتناهی به‌دست خواهد آمد [۶].

۷-۲- الگوریتم یادگیری کیو[۱۵]

در الگوریتم کنترلی و غیر برخط یادگیری کیو، به هر زوج حالت- کنش یک مقدار $Q(s,a)$ نسبت داده می‌شود. این مقدار عبارت است از مجموع پاداش‌های دریافتی، وقتی عامل از حالت s شروع و کنش a را انجام داده و در ادامه سیاست موجود را پیروی کرده باشد و تا زمانی که به مقدار بهینه همگرا شود، با استفاده از رابطه (۲) به‌روز رسانی می‌شود. لازم به ذکر است که این الگوریتم نیازی به داشتن مدلی از محیط ندارد. همچنین در هر اپیزود، زمانی‌که s_{t+1}، یک حالت نهایی باشد، مقدار تابع Q برای آن هیچ‌گاه به‌روز نمی‌شود و مقدار اولیه خود را حفظ می‌کند [۶].

(۲)

$$Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \alpha[r_{t+1} + \gamma \max_{a'} Q(s_{t+1},a_{t+1}) - Q(s_t,a_t)]$$

روال زیر الگویتم یادگیری کیو را نشان می‌دهد:

1. Initialize $Q(s,a)$ arbitarily
2. Repeat (for each episode)
3. Initialize s
4. Repeat (for each step of episode)
5. Choose a from s using policy derived from Q (e.g, ε-greedy)
6. Take action a, observe r, s'
7. $Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \alpha[r_{t+1} + \gamma \max_{a'} Q(s_{t+1},a_{t+1}) - Q(s_t,a_t)$
8. $s \leftarrow s'$;
9. Until s is terminal;

شرح الگوریتم:

- به ازای هر کنش a، Q را مقداردهی اولیه کن (معمولا صفر)
- در هر اپیزود تکرار کن
  - حالت فعلی s را مشاهده کن
  - تا مشاهده حالت پایانی تکرار کن
    - کنش a را از حالت فعلی s انتخاب کن (با سیاست جاری)
    - کنش a را اجرا کن، پاداش r و حالت جدید s' را مشاهده کن
    - مقدار Q را برای s و a به‌روز کن
    - حالت جدید s' را حالت فعلی s در نظر بگیر

۷-۳- الگوریتم R-max

الگوریتم R-max، یک روش استاندارد و غیر برخط تفاضل زمانی بوده و به الگوریتم یادگیری کیو خیلی نزدیک است و از دو سیاست تشکیل می‌شود: یک سیاست رفتاری و یک سیاست تخمین، به‌علاوه تابع ارزش- کنش و متوسط پاداش دریافتی عامل. عامل از سیاست رفتاری برای تولید تجربه و دانش استفاده می‌کند، مانند سیاست ε- حریصانه با توجه به تابع ارزش- کنش. از نمونه‌های سیاست تخمین هم می‌توان به سیاست حریصانه باتوجه به تابع ارزش- کنش اشاره کرد. لازم به ذکر است که این الگوریتم می‌تواند متوسط پاداش بهینه (ρ) را در مدت زمان طولانی به‌دست آورد. در الگوریتم R-max عامل همواره مدلی کامل ولی نه‌چندان دقیق از محیط را در خود نگه می‌دارد و بر اساس این مدل سیاست بهینه را انتخاب می‌کند. عامل در این‌گونه مسائل به‌دنبال بیشینه کردن پاداش در هر گام زمانی است؛ تمامی کنش‌ها در هر حالت بیش‌ترین مقدار پاداش را به‌دست عامل می‌دهد. روال زیر الگوریتم R-max را به صورت کامل نشان می‌دهد. β هم مانند α پارامتر نرخ یادگیری می‌باشد [4].

1. Initialize ρ and $Q(s,a)$, for all s,a, arbitarily
2. Repeat forever:
3. $s \leftarrow$ current state
4. Choose action a in s using behavior policy, (e.g, ε- greedy)
5. Take action a, observe r, s'
6. $Q(s,a) \leftarrow Q(s,a) + \alpha[r - \rho + \max_{a'} Q(s',a') - Q(s,a)]$
7. If $Q(s,a) = \max_a Q(s,a)$, then:
8. $\rho \leftarrow \rho + \beta[r - \rho + \max_{a'} Q(s',a') - \max_a Q(s,a)]$

شرح الگوریتم:

- به ازای هر حالت s و کنش a، Q و ρ را مقداردهی اولیه کن (معمولا صفر)
- تا بی‌نهایت انجام بده
  - حالت فعلی s را مشاهده کن
  - کنش فعلی a را با استفاده از سیاست رفتاری انتخاب کن
  - کنش a را اجرا کن، پاداش r، حالت جدید s' را مشاهده کن
  - مقدار Q را برای حالت s و کنش a به‌روز کن
  - اگر مقدار $Q(s,a)$ برابر مقدار بیشینه موجود بود،
    - مقدار ρ را به‌روز کن

۸- پاداش شکل‌دهی شده[۱۶]

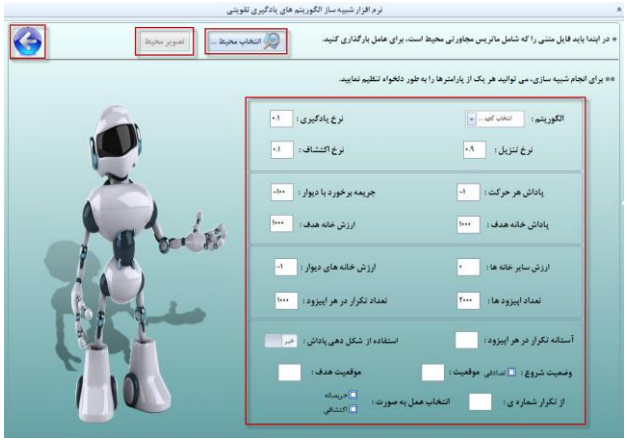
در اغلب مسائل مبتنی بر هدف، گرچه مجموع پاداش دریافتی عامل بعد از هر کنش تغییر می‌کند اما بیش‌ترین پاداش بعد از رسیدن به هدف داده می‌شود و چنین تابع پاداش‌هایی با بازخورد همراه با تاخیر، سرعت یادگیری را کاهش می‌دهند. از این‌رو پاداش شکل‌دهی شده می‌تواند تاثیر به‌سزایی در افزایش سرعت یادگیری عامل داشته باشد.

پاداش شکل‌دهی شده به معنی دادن یک بازخورد مصنوعی ایجاد شده توسط طراح به‌غیر از پاداش محیط، به عامل، به‌منظور افزایش نرخ سرعت یادگیری است. لازم به ذکر است گرچه پاداش شکل‌دهی شده ابزار قدرتمندی در افزایش سرعت یادگیری عامل است، اما به همان نسبت تعریف غلط آن می‌تواند در گمراه نمودن عامل تاثیر زیادی داشته باشد [۶].

۹- نرم‌افزار شبیه‌ساز

نرم‌افزار شبیه‌سازی که برای اجرای الگوریتم‌های یادگیری تقویتی از جمله، الگوریتم سارسا، کیو و R-max، بر روی محیط‌های مختلف و به‌همراه تنظیم پارمترهای موثر در اجرا، طراحی شده است، دارای مشخصاتی به این شرح می‌باشد: با اجرای نرم‌افزار، ابتدا فرمی مشابه شکل ۱ نمایش داده شده که در این فرم کاربر، نخست فایل متنی شامل ماتریس مجاورت گراف محیط مورد نظر خود را در نرم‌افزار

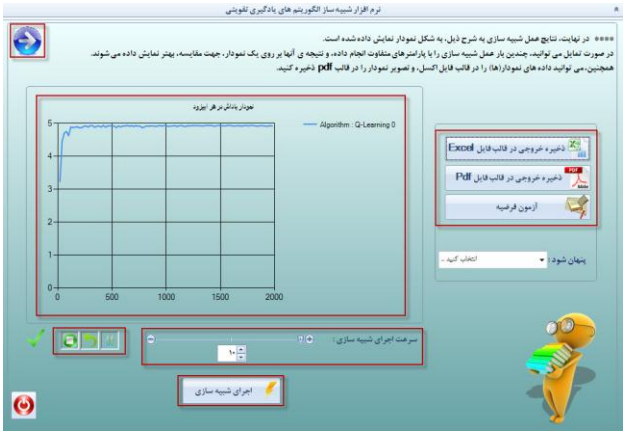
بارگذای کرده و سپس کلیه پارامترهای مورد نیاز را تنظیم و به مرحله بعد هدایت می‌شود.



شکل (۱): بارگذاری فایل متنی و تنظیم پارمترهای شبیه‌سازی

مرحله دوم، مروری بر روی پارامترهای تنظیم‌شده خواهد بود. در صورتی که موردی از قلم افتاده و یا پارامتری نیاز به تنظیم مجدد داشته باشد، به مرحله قبل بازگشته و تنظیمات مورد نظر خود را انجام می‌دهد (شکل ۲). در نهایت با مشاهده فرم نهایی، با کلیک دکمه اجرای شبیه‌سازی، الگوریتم مورد نظر بر روی محیط انتخابی اجرا و نتیجه‌ی آن در قالب نمودار پاداش- اپیزود، بر روی صفحه ترسیم می‌گردد.

شکل (۲) : مروری بر روی کلیه پارامترهای تنظیم‌شده



شکل (۳): اجرای شبیه‌سازی و سایر امکانات نرم‌افزار

برای بررسی‌های بیشتر، می‌توان داده‌های شبیه‌سازی را در قالب فایل اکسل و همچنین می‌توان تصویر نمودار را به شکل فایل pdf ذخیره کرد. امکان دیگری که در نرم‌افزار تعبیه شده است، امکان انجام آزمون فرض آماری بر روی دو نمونه داده به انتخاب کاربر است (شکل ۳). این آزمون بررسی می‌کند که آیا برتری نمونه داده اول بر نمونه داده‌ی دیگر از نظر آماری معنی‌دار هست یا خیر. و این معنی‌دار بودن را پس از انجام آزمون با چاپ درصد اطمینان به اطلاع کاربر می‌رساند.

۱۰- آزمایش‌ها و ارزیابی

اجرای الگوریتم یادگیری کیو بر روی محیط ناوبری maze و همچنین محیط ناوبری شش اتاقه نشان داده شده است. محیط ناوبری محیطی است که در آن عامل شی‌ای را ازنقطه‌ای به نقطه‌ی دیگر انتقال می‌دهد. این الگوریتم به‌همراه تنظیم پارامترها به‌صورت $\alpha = 0.1$، $\gamma = 0.9$، $\varepsilon = 0.1$، تعداد اپیزودها برابر ۲۰۰۰ و تعداد تکرار در هر اپیزود حداکثر برابر ۱۰۰۰، یک بار بدون استفاده از پاداش ساختگی و بار دیگر با تاثیر این پاداش به شکل رابطه‌ی (۳) بر روی هریک از این محیط‌ها اجرا و برای مقایسه به‌ترتیب به شکل نمودار پاداش- اپیزود در شکل ۴ و ۵ نمایش داده شده‌اند.

(۳)

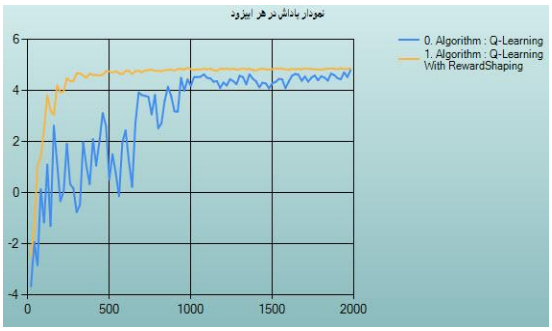
$$Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \alpha[r_{t+1} + \max_a Q(s_{t+1},a_{t+1}) - Q(s_t,a_t) + \gamma\max_a Q(s_{t+1},a_{t+1}) - Q(s_t,a_t)]$$

طبق آنچه در شکل‌ها مشاهده می‌شود سرعت همگرایی با استفاده از پاداش ساختگی به‌طور قابل ملاحظه‌ای بهبود یافته است. برای تعیین اینکه آیا این افزایش سرعت از لحاظ آماری معنی‌دار است، از یک روش آماری به‌نام *آزمون فرض آماری برای تفاضل دو میانگین* استفاده شده است. با استفاده از فرمول (۴)، مقدار z را محاسبه کرده و در جدولی از مقادیر بحرانی جستجو می‌کنیم. با محاسبه روی داده‌های آزمایش، مقادیر ۱۵/۱۵ و ۱۳/۷۳ به‌ترتیب در محیط‌های maze و شش اتاقه برای z محاسبه شده‌اند. نتیجه می‌شود که فرض صفر با احتمال حدود ۹۹ درصد برای هر دو محیط رد می‌شود. رد فرض صفر

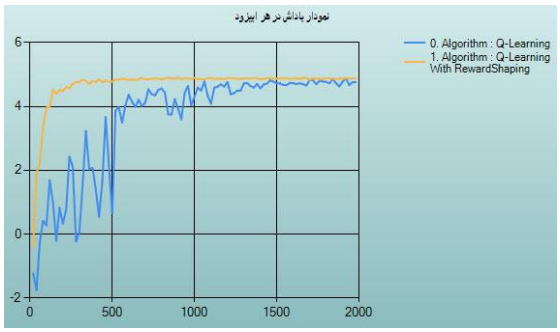
دلالت بر این دارد که بهبود سرعت یادگیری به‌دست آمده با به‌کارگیری پاداش ساختگی از لحاظ آماری معنی دار است.

(۴)

$$Z = \frac{(\overline{x_1} - \overline{x_2}) - \delta}{\sqrt{\frac{\sigma_1^2}{n_1} + \frac{\sigma_2^2}{n_2}}}$$



شکل (۴) : مقایسه میانگین پاداش دریافتی در الگوریتم کیو بر روی محیط maze، بدون پاداش ساختگی (نمودار آبی‌رنگ) و با پاداش ساختگی (نمودار نارنجی‌رنگ)



شکل (۵): مقایسه میانگین پاداش دریافتی در الگوریتم کیو بر روی محیط شش اتاقه، بدون پاداش ساختگی (نمودار آبی‌رنگ) و با پاداش ساختگی (نمودار نارنجی‌رنگ)

۱۱- نتیجه‌گیری

در این مقاله، مروری بر ادبیات یادگیری تقویتی، مفاهیم اصلی، روش‌ها و الگوریتم‌های یادگیری تقویتی و مفهوم شکل‌دهی پاداش در اجرای الگوریتم‌ها انجام شد. سپس الگوریتم یادگیری کیو بر روی محیط‌های محکی چون شش اتاقه و **maze** یک‌بار بدون پاداش شکل‌دهی‌شده و بار دیگر با اعمال پاداش شکل‌دهی شده اجرا و نتایج اجرا در قالب نمودار پاداش- اپیزود نمایش داده شد.

مراجع

[۱] کلامی، مصطفی، "**یادگیری تقویتی: روش‌ها و کاربردها**"، سمینار دوره‌ای گروه کنترل، دانشگاه صنعتی خواجه نصیرالدین طوسی، تهران، ایران، فروردین ۱۳۸۸.

[۲] خرامان، یونس، "**یادگیری ماشین**"، گزارش تحقیق، موسسه آموزش عالی اشراق واحد بجنورد، بجنورد، ایران، مهر ۱۳۸۹.

[۳] مومن، حسن، "**یادگیری ماشین و یادگیری بیزی**"، گزارش تحقیق، موسسه آموزش عالی اشراق واحد بجنورد، بجنورد، ایران، مهر ۱۳۸۹.

[4] R. S. Sutton and A. G. Barto, "**Reinforcement Learning An Introduction**", Cambridge: MT press, 1998.

[۵] عطریان‌فر. حامد، "**یادگیری تقویتی**"، گزارش تحقیق، دانشگاه صنعتی شریف، تهران، ایران.

[۶] مرعشی، مریم، "**کسب مهارت در یادگیری تقویتی فعال توسط عامل‌های خودمختار**"، پایان‌نامه کارشناسی ارشد، دانشگاه صنعتی امیرکبیر، تهران، ایران، مهر ۱۳۹۱.

[۷] فرحناکیان، فهیمه، "**یادگیری تقویتی**"، ماهنامه هوش مصنوعی و ابزار دقیق، تهران، ایران، شماره۶، سال اول، اسفند ۱۳۸۶.

[۸] جمشیدی، نیلوفر، "**مروری بر الگوریتم‌های یادگیری تقویتی و پیاده‌سازی الگوریتم یادگیری کیو روی چند محیط محک**"، رساله کارشناسی، دانشکده فنی دکتر شریعتی، تهران، ایران، بهار ۱۳۹۲.

[۹] "**یادگیری کیو**"، [برخط]، قابل دسترسی در: http://fa.wikipedia.org/ .

زیرنویس‌ها

[1] Reinforcement Learning
[2] state
[3] Action
[4] Policy
[5] Reward function
[6] Value function
[7] Environment
[8] Markov Property
[9] Markov Decision Process
[10] Temporal Difference
[11] Dynamic Programming
[12] Mont Carlo
[13] Temporal Difference
[14] SARSA (State, Action, Reward, State, Action)
[15] Q-Learning
[16] Shaped Reward