Archive of SID

علوم و مهندسی کامپیوتر
نشریه علمی پژوهشی انجمن کامپیوتر ایران
مجلد ۱۱، شماره ۱ (الف)، ۱۳۹۲
صفحات ۶۱-۶۷
مقاله عادی

یک مدل تلفیقی مبتنی بر یادگیری تقویتی و استدلال مبتنی بر مورد برای افزایش کارایی سیستم‌های چندعامله

سارا اسفندیاری[۱] بهروز معصومی[۱] محمدرضا میبدی[۲]

[۱]دانشکده مهندسی کامپیوتر و فناوری اطلاعات، دانشگاه آزاد اسلامی واحد قزوین، قزوین، ایران
[۲]دانشکده مهندسی کامپیوتر و فناوری اطلاعات، دانشگاه صنعتی امیرکبیر، تهران، ایران

چکیده

در این مقاله، روشی با استفاده از استدلال مبتنی بر مورد، برای افزایش سرعت الگوریتم‌های یادگیری تقویتی در سیستم‌های چندعامله ارائه شده است. در روش پیشنهادی، مدل تلفیقی جدید با استفاده از سیستم‌های استدلال مبتنی بر مورد و یک تابع بهبودیافته‌ی جدید برای انتخاب عمل پیشنهاد شده است که، موجب تسریع درالگوریتم‌های مبتنی بر Q-Learning شده است. روش مذکور برای حل مسئله بازی‌های مارکوف همکارانه به عنوان یکی از مدل‌های سیستم‌های چندعامله مبتنی بر مارکوف استفاده شده است. نتایج بدست آمده از آزمایش‌ها، نشان می‌دهند که روش پیشنهادی ارائه شده نرخ همگرایی و سرعت یادگیری را افزایش می‌دهد.

کلمات کلیدی: الگوریتم Q-Learning، بازی‌های مارکوف، توزیع بولتزمن، سیستم‌های استدلال مبتنی بر مورد، سیستم‌های چندعامله.

۱- مقدمه

استدلال مبتنی بر مورد[۱] (CBR) یک روش حل مساله مبتنی بر دانش است که بر پایه استفاده مجدد از تجربیات پیشین عمل می‌کند و از تحقیقات علوم شناختی پدیدار شده است [۱]. در این روش فرض بر این است مسائل مشابه می‌توانند راه‌حل‌های مشابهی داشته باشند. بنابراین، ممکن است مسائل جدید با روش‌های تجربه شده در مسائل قبلی قابل حل باشند. یک سیستم چندعامله[۲]، شامل مجموعه‌ای از عامل‌های هوشمند و خودمختار است که هر یک به طور جداگانه با محیطی اشتراکی در ارتباط هستند، تا بتوانند به هدف مشخصی برسند. با توجه به این که عامل‌ها، در سیستم‌های چند عامله، با مسئله کمبود یا فقدان اطلاعات درباره محیط مواجه هستند و دانش کاملی درباره‌ی محیط وجود ندارد و معمولاً محیط نیز ناشناخته است، استفاده از الگوریتم‌های یادگیری تقویتی، از اهمیت بالایی برخوردار است [۲].

الگوریتم‌های یادگیری تقویتی، در هر مرحله‌ی زمانی، به عامل اجازه می‌دهند که بر اساس مشاهدات خود از محیط، یک عمل را انجام دهد و به حالت جدیدی وارد شود، سپس یک سیگنال پاداش که نشان‌دهنده‌ی کیفیت عمل انتخاب شده است، به عامل داده می‌شود [۳]. تاکنون برای مدل‌سازی سیستم‌های چندعامله، مدل‌های مختلفی پیشنهاد شده است؛ مدل فرآیند تصمیم گیری مارکوف چند عامله[۳] (MMDP) یکی از این مدل‌ها است [۴]. این مدل، توسعه‌ای از فرآیندهای تصادفی مارکوف با چندین عامل بوده و در تحقیقات یادگیری تقویتی بسیاری به کار رفته‌اند. برای بدست آوردن راه‌حل بهینه در بازی‌های مارکوفی، الگوریتم‌های یادگیری تقویتی مختلفی به‌کار رفته است. هو و ولمن الگوریتمی برای حل بازی‌های مارکوف کلی ارائه دادند. آنها الگوریتمی به نام Nash-Q را پیشنهاد دادند که تحت شرایط خاص به سیاست تعادل نش همگرا می‌شود [۵]. معصومی و همکاران در سال ۲۰۱۲ الگوریتمی برای حل بازی‌های مارکوف همکارانه ارائه کردند که از اتوماتای یادگیر و مفهوم آنتروپی استفاده می‌کرد [۶]. در سال ۲۰۰۰ الگوریتم Distributed Q-Learning برای عامل‌های مستقل معرفی شده است.

www.SID.ir

Archive of SID

ثابت شده است که این الگوریتم به سمت تعادل نش همگرا می‌شود، با این‌حال در این الگوریتم از هیچ‌گونه مکانیزم هماهنگی بین عامل‌ها استفاده نشده است [۷]. در سال ۲۰۰۴، الگوریتمی به نام FMQ معرفی شده است که مقادیر Q هر عمل در استراتژی توزیع بولتزمن را توسط یک تابع ابتکاری، تغییر می‌دهد و بدین ترتیب باعث همگرایی زودتر به سمت پاسخ بهینه می‌شود [۸]. در [۹]، الگوریتمی به نام Hysteretic Q-Learning، معرفی شده است که با اضافه کردن پارامتر جدیدی به روش FMQ، باعث بهبود عملکرد این الگوریتم شده است. در [۱۰]، الگوریتمی به نام CAQL، که براساس الگوریتم Q-Learning عمل می‌کند، معرفی شده است. در [۱۱]، یک روش مبتنی بر الگوریتم Q-Learning مطرح شده است.

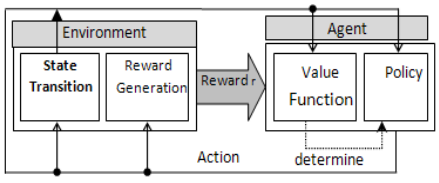
یکی از راه‌های افزایش سرعت الگوریتم‌های یادگیری تقویتی، بهبود تابع انتخاب عمل است. در [۱۲، ۱۳] نمونه‌هایی از این توابع به صورت مکاشفه ای ارائه شده‌اند. این توابع برای انتخاب عمل در هر حالت به کار برده شده‌اند. با وجود اینکه این روش‌ها به صورت موفقیت‌آمیزی، برای حل بازی‌های مارکوف، مورد استفاده قرار گرفته‌اند، ولی هنوز مسئله‌ی استفاده‌ی مجدد از تجربیات قبلی عامل‌ها در حل مسائل جدید، در این روش‌ها نادیده گرفته شده است. از آنجاییکه در سیستم‌های چندعامله، محیط ناشناخته است و عامل بایستی از طریق مشاهده، دانش خود را درباره‌ی محیط افزایش دهد، بنابراین مسئله‌ی حفظ و استفاده‌ی مجدد از دانش کسب شده‌ی قبلی، موجب افزایش سرعت یادگیری می‌شود.

در این مقاله، به منظور افزایش سرعت یادگیری سیاست بهینه برای بازی‌های مارکوف در حالت عامل‌های مستقل، یک روش تلفیقی جدید به نام CB-BHAQL[۴] پیشنهاد می‌شود. که در آن، از یک تابع بهبود یافته برای انتخاب عمل و تکنیک استدلال مبتنی بر مورد به عنوان ابزاری برای افزایش سرعت یادگیری، استفاده شده است. نتایج شبیه‌سازی‌های انجام شده بر روی محیط Grid World نشان می‌دهند که الگوریتم پیشنهادی، نسبت به روش‌های موجود کارایی بهتری از نظر سرعت دارد. در ادامه مقاله، ابتدا در بخش ۲ مفاهیم اولیه و در بخش ۳ روش پیشنهادی شرح داده می‌شود و در بخش ۴ ارزیابی الگوریتم ارائه شده ودر بخش ۵ بررسی رفتار الگوریتم و تحلیل آن و بخش ۶ نتیجه‌گیری است.

۲- مفاهیم اولیه

۲-۱- یادگیری تقویتی

یک عامل یادگیر تقویتی، رفتارش را از طریق تعامل با یک محیط ناشناخته و مشاهده‌ی نتایج اعمالش، تعیین می‌کند [۵]. ایده‌ی یادگیری تقویتی در شکل ۱ آمده است. در شکل ۱، ابتدا عامل، حالت فعلی سیستم (S) را دریافت می‌کند. با استفاده از یک تابع تصمیم ساز (Policy) عمل a مشخص می‌شود و عامل پس از اعمال عمل a بر روی محیط، پاداش r را دریافت می‌کند. سپس با استفاده از مقادیر a و s و r مقدار تابع یادگیر تقویتی توسط تابع ارزیابی به روزرسانی می‌شود. الگوریتم‌های یادگیری تقویتی، سعی می‌کنند که سیاست‌هایی را برای نگاشت حالت‌ها به عمل‌هایی که هر عامل باید در آن حالت انجام دهد، پیدا کنند.



شکل ۱- مدل یادگیری تقویتی

در الگوریتم‌های یادگیری تقویتی، معمولاً محیط به صورت یک فرآیند تصادفی مارکوف[۵] (MDP) مدل می‌شود. یک MDP یک چهارتایی به صورت $< S, A, T, r >$ است که در آن S مجموعه متناهی از حالات و A مجموعه‌ای از عملیات قابل دسترس برای عامل و $T: S \times A \times S \rightarrow [0,1]$ تابع انتقال از حالت فعلی به حالت بعدی و $r: S \times A \rightarrow R$ تابع پاداش است. هدف، پیدا کردن سیاستی به صورت $\pi: S \rightarrow A$ است به گونه‌ای که میانگین پاداش دریافتی در طول زمان بیشینه گردد. برای هر خط مشی نظیر π که عامل می‌تواند دنبال کند، بر روی وضعیت‌ها تابعی به نام تابع ارزیابی[۶] تعریف می‌شود. الگوریتم Q-Learning یکی از تکنیک‌هایی است که برای تابع ارزیابی استفاده می‌شود. شبه کد الگوریتم Q-Learning در شکل ۲ آمده است. در این الگوریتم‌ها برای هر عمل a در هر حالت S از مقدار ارزش آن عمل Q(s,a) مطابق رابطه ۱ استفاده می‌شود. در رابطه ۱، α نرخ یادگیری و $\gamma \in [0, 1]$ فاکتور کاهش است. الگوریتم هنگامی به پایان می‌رسد که سیاست بهینه برای مدت زمان معینی تغییر نکند.

$$Q(S,a) = (1-\alpha)Q(S,a) + \alpha(r + \gamma \max_a Q(S',a')) \quad (۱)$$

معمولاً برای انتخاب عمل در هر حالت (قسمت Policy)، از روش توزیع بولتزمن (رابطه‌ی ۲) استفاده می‌شود.

$$\pi(S) = \arg\max\left(\frac{e^{\frac{Q(S,t)}{\tau}}}{\sum_{i=1}^{m} e^{\frac{Q(S,t)}{\tau}}}\right) \quad (۲)$$

که در آن، m تعداد اعمال مجاز برای حالت S و τ ثابت است و Q(S,a) مقدار تابع ارزیابی حالت S را هنگامی که عمل a انجام می‌گیرد، نشان می‌دهد. در کاربردهای واقعی، تعیین مقدار دقیق τ برای همگرا شدن به سیاست بهینه، کار مشکلی است، از این رو در این مقاله، برای انتخاب عمل در هر حالت، رابطه‌ی ۳ پیشنهاد شده است. در رابطه‌ی۳، m تعداد اعمال مجاز برای حالت S و n(S,a) تعداد دفعاتی است که تاکنون عمل a انتخاب شده است و Q(S,a) مقدار تابع ارزیابی حالت S را هنگامی که عمل a انجام می‌گیرد، نشان می‌دهد.

$$\pi(S) = \arg\max\left(\frac{e^{n(S,a)Q(S,a)}}{\sum_{i=1}^{m} e^{n(S,a)Q(S,a)}}\right) \quad (۳)$$

Initialize $Q_t(S,a)$ arbitrarily
Repeat (for each episode)
  Initialize S randomly
  Repeat (for each step)
    Select an action a_i using EQ (2)
    Execute the action a, observe r(s, a), new state S'
    Update the Value of $Q_t(S,a)$ according
      to EQ (1)
    $S \leftarrow S'$
  Until S is theTerminal state
Until Some Stopping Criterion Criteria is reached.

شکل ۲- الگوریتم Q-Lerning

۲-۲- بازی‌های مارکوف

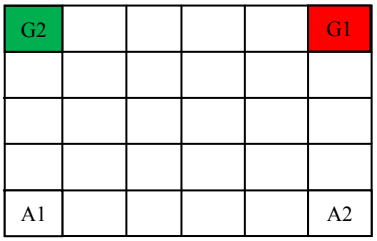
بازی مارکوف، تعمیم یافته مسئله تصمیم‌گیری مارکوف (MDP) به حالت

www.SID.ir

Archive of SID

چندعامله است. یکی از محیط های استفاده شده برای بازی‌های مارکوف چندعامله، بازی Grid World است که در [۵] معرفی شده است. در این بازی، دو عامل مستقل از مراحل ردیف پایین شروع به بازی می‌کنند و سعی می‌کنند که سلول هدفشان را در مراحل بالایی، پیدا کنند. یک عامل فقط می‌تواند در یک لحظه، به داخل یک سلول، حرکت کند. چهار عمل ممکن برای هر عامل وجود دارد: بالا، پایین، چپ، راست. اگر دو عامل سعی کنند که به خانه‌های یکسانی وارد شوند، بجز حالت هدف، به سلول قبلی خود بازگشت خورده و هر دو عامل، ۱- واحد جریمه می‌شوند. به محض اینکه یکی از عامل‌ها به حالت هدف رسید، بازی به پایان می‌رسد و عاملی که به حالت هدف رسیده است ۱۰۰+ واحد پاداش دریافت می‌کند.

هدف یک عامل این است که با مینیمم تعداد حرکات بتواند به حالت هدف برسد. یک سیاست (استراتژی)، دنباله‌ی اعمال انجام گرفته از حالت شروع به حالت پایان است. کوتاه‌ترین مسیری که اجازه دهد تا یک عامل بتواند هر چه زودتر به هدف برسد، یک استراتژی بهینه است. شکل ۳ نمونه‌ای از این بازی است. استراتژی بهینه در شکل ۳ شامل ۹ حرکت است.



شکل ۳- نمونه‌ای از بازی Grid World

۲-۳- استدلال مبتنی بر مورد

تکنیک استدلال مبتنی برمورد (CBR) از تجربیات بدست آمده‌ی قبلی (Case) برای حل مسائل جدید استفاده می‌کند. در سیستم‌های استدلال مبتنی بر مورد، تجربیات بدست آمده از حل مسائل، درون پایگاه موارد[۷] (CB) ذخیره می‌شود. در این سیستم‌ها، برای حل مسئله جدید (C_{new})، شبیه‌ترین موارد به C_{new} از درون پایگاه موارد (CB)، استخراج می‌شود و از راه‌حل‌های ارائه شده توسط موارد استخراجی، برای حل مسئله جدید C_{new} استفاده می‌شود. اگر مورد مشابهی یافت نشد، C_{new} به عنوان مورد جدید وارد پایگاه موارد می‌شود [۱]. برخلاف تکنیک‌های سنتی مبتنی بر دانش، CBR بر روی تجربه حل مساله خاصی تمرکز می‌کند که برگرفته از موارد جمع‌آوری شده در پایگاه مورد است. این موارد، تجربه خاصی را در یک دامنه حل مساله نشان می‌دهند. باید توجه داشت که CBR یک راه‌حل قطعی را پیشنهاد نمی‌کند بلکه فرضیات و نظراتی را برای عبور از فضای راه‌حل، ارائه می‌کند.

۳- روش پیشنهادی

در این بخش، یک روش جدید (الگوریتم) به منظور افزایش سرعت همگرایی در بازی‌های مارکوف، موسوم به CB-BHAQL ارائه شده است. در الگوریتم پیشنهادی از استدلال مبتنی بر مورد و نیز تابع جدیدی برای انتخاب عمل در هر حالت به منظور افزایش سرعت همگرایی به سمت سیاست بهینه، استفاده شده است. در اکثر الگوریتم‌های ارائه شده برای یادگیری تقویتی که از روش یادگیری Q استفاده کرده‌اند، فرآیند یادگیری و سرعت آن فقط بستگی به رابطه یادگیری Q به کاررفته در آن است و همیشه مسئله‌ی استفاده مجدد از یادگیری های قبلی عامل‌ها نادیده گرفته شده است. به نظر می آید اگر در هر بار تکرار الگوریتم، دانش یادگرفته شده‌ی عامل از محیط (مقدار Q)، در جایی حفظ شود و از آن برای ادامه فرآیند یادگیری استفاده شود، سرعت یادگیری الگوریتم نیز افزایش پیدا می کند از اینرو در این مقاله الگوریتم یادگیری Q با مدل استدلال مبتنی بر مورد تلفیق شده است. حل یک مسئله با استدلال مبتنی بر مورد شامل مراحل: ایجاد توصیفی از مسئله، اندازه‌گیری میزان شباهت مسئله کنونی با مسائل قبلی حل شده ذخیره شده در پایگاه موارد، بازیابی یک یا چندین مسئله مشابه از داخل پایگاه موارد، سعی برای استفاده مجدد از راه‌حل ارائه شده توسط مورد بازیابی شده برای حل مسئله کنونی، می‌باشد.

ساختار مورد استفاده شده در الگوریتم پیشنهادی، یک دو تایی به صورت Case=<Prob,Sol> است که Prob توصیف کننده مسئله و Sol راه‌حل ارائه شده برای مسئله است. توصیف کننده مسئله (Prob) حاوی مشخصات هر حالت است که به صورت Prob(S)={m, <Up, Down, Right, Left>, index} تعریف می شود M تعداد اعمال هر حالت و مجموعه‌ی <Up, Down, Right,Left> اعمال مجاز برای هر حالت و index اندیس هر حالت است. راه‌حل ارائه شده برای مسئله $Sol(S) =< \vec{E}, V >$ است که در آن بردار $\vec{E}$ به صورت $\vec{E} = (\vec{E}[1], \vec{E}[2],..., \vec{E}[m])$ لیستی از تجربیات جمع‌آوری شده از محیط توسط عامل برای حالت S بوده و هر بردار $\vec{E}$ شامل یک چهارتایی $e =< a_i, n_i, Q_i, \pi_i >$ است که a_i عمل مجاز برای حالت S و n_i تعداد دفعاتی که عمل a_i بروزرسانی شده است و Q_i مقدار تخمین زده شده توسط رابطه‌ی ۱ و π_i احتمال وقوع عمل a_i که توسط رابطه‌ی ۳ تخمین زده می‌شود. V مجوز استفاده از راه‌حل ارائه شده توسط مورد بازیابی شده است و اگر هر یک از اعمال حالت S حداقل یک بار انتخاب شده باشند، است. در این‌صورت از راه‌حل مورد بازیابی شده برای حل مسئله جدید می‌توان استفاده کرد.

در الگوریتم پیشنهادی هر بار که عامل وارد حالت جدیدی می‌شود، مشابه‌ترین مورد به حالت جدید را از درون پایگاه موارد استخراج می‌کند و در صورت دارا بودن مجوز (V=True)، از مورد بازیابی شده، برای تعیین حالت بعدی استفاده می‌کند. برای بازیابی موارد مشابه حالت فعلی، از الگوریتم نزدیکترین همسایه[۸] استفاده شده است. فاصله اقلیدسی مورد جدید، با هر یک از موارد موجود در داخل پایگاه موارد (CB) طبق رابطه‌ی ۴ محاسبه شده و مشابه‌ترین مورد (c) بازیابی می‌شود و در صورت دارا بودن مجوز (V=True)، از راه‌حل مورد بازیابی شده برای حل مسئله جدید استفاده می‌شود. الگوریتم پیشنهادی در شکل ۴ آمده است.

$$\pi NN(S) = \operatorname{argmax}_{c \in CB} \mathrm{Sim}(C.\mathrm{Prob}, S.\mathrm{Sol}) = \operatorname{argmax}_{c \in CB} \mathrm{dist}(C.\mathrm{Prob}, S.\mathrm{Sol}) \quad (۴)$$

۴- ارزیابی الگوریتم پیشنهادی

برای ارزیابی کارایی الگوریتم CB-BHAQL از یک بازی Grid World مطابق شکل ۳ با ۳۰ حالت و دو عامل مستقل A_1 و A_2 که دارای هدف‌های جداگانه‌ای هستند، استفاده شده است. هدف ما این است که نشان دهیم آیا الگوریتم CB-BHAQL که تلفیقی از CBR و QL[۹] است و در آن از تابع بهبودیافته‌ی جدیدی برای انتخاب عمل استفاده شده است، عملکرد بهتری نسبت به دو الگوریتم QL و CBR دارد؟ به همین منظور، الگوریتم CB-BHAQL را با دو الگوریتم ۱) الگوریتم Q-Learning ۲) الگوریتم Boltzmann-CBR که شبه کد آن شبیه شکل ۴ است و تنها تفاوت آن در ردیف ۵ از شکل ۴ است که برای انتخاب عمل به جای استفاده از تابع بهبودیافته‌ی جدید (رابطه‌ی ۳) از توزیع

www.SID.ir

Archive of SID

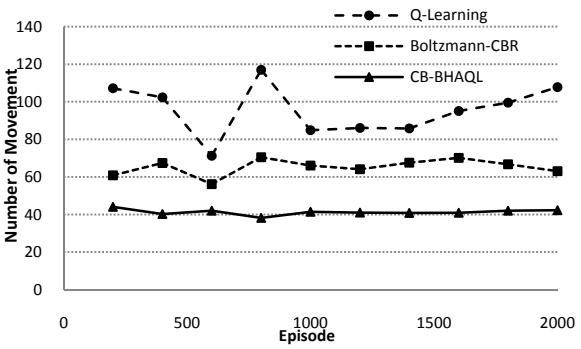
بولتزمن (رابطه‌ی ۲) استفاده شده است.

1. Let t be the global time, n be the number of agents, γ the discount factor, $CB_i=\phi$ an empty case base for each Agent
   Set $s = s' \in S$ to the initial state of the system.
2. Repeat
3. (a) Set S=S'
4. (b) **for all** agent $i \in [1 \dots n]$ do
   if $CB_i=\phi$ or add case_criterion(s) is true
   $CB = CB \cup C$ with c.Prob=s and
   c.Sol=empty_solution (i)
5. **for each** j=Sol(s).m do
   Compute Sol (s). E[j]. π_i according to **Eq(3)**
   Set index x_i the Maximum value of them.
6. Select elementary action $Sol(s).E[x_i].a_i$.
7. Observe Successor state $s' \in S$ and reward $r \in R$.
8. **for all** agents $i \in [1 \dots n]$ do
   a. **Retrieve** nearest neighbor according to Eq (4) of state s'.
   b. **Set** Learning Rate $\alpha_i = \frac{1}{1+Sol(s).E[x_i].n_i}$
   c. **Set** $Sol(s).E[x_i].Q_i$ according to **Eq(1)**.
   d. Increment $Sol(s).E[x_i].n_i$ by one.
   e. **Resort** detrimentally the experience list Q in Sol(s).E
9. **end for**
10. **end for**
11. Until **Stop_Criterion ()** becomes true.

شکل ۴- شبه کد الگوریتم پیشنهادی CB-BHAQL

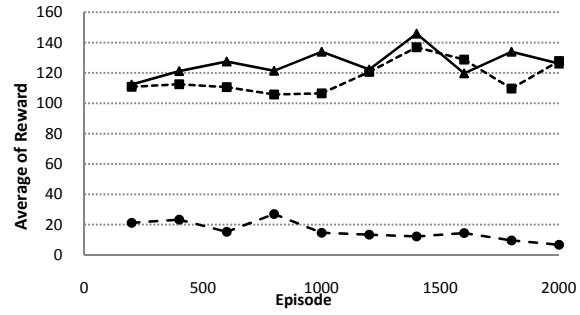
در آزمایش‌های انجام گرفته فرض شده است که، ۱۰۰۰ بار الگوریتم‌ها اجرا شده و میانگین نتایج برای الگوریتم ها به دست آمده‌اند. پارامترهای $\tau = 0.05$ و $\gamma = 0.7$ در نظر گرفته شده‌اند. شکل ۵ و ۶ و ۷ نتایج شبیه‌سازی را نشان می‌دهد.

آزمایش ۱. مقایسه کارایی الگوریتم پیشنهادی با سایر الگوریتم‌ها از نظر تعداد حرکات لازم برای رسیدن به هدف. برای این منظور، تعداد حرکات لازم برای همگرایی الگوریتم در مقایسه با الگوریتم های دیگر مورد آزمایش قرار می گیرد. نمودار شکل ۵ سه الگوریتم را از لحاظ تعداد حرکات لازم برای همگرایی به سمت سیاست بهینه، مقایسه می‌کند. هر چه تعداد حرکات لازم برای رسیدن به سیاست بهینه، کمتر باشد، الگوریتم کاراتری خواهیم داشت. با توجه به شکل۵ مشاهده می‌شود که الگوریتم یادگیری Q به تنهایی به بیشترین تعداد حرکات برای همگرایی نیازمند است. حال اگر الگوریتم یادگیری Q را با تکنیک CBR تلفیق نماییم (الگوریتم Boltzman CBR) که در آن از تابع بولتزمن (رابطه‌ی ۲) برای انتخاب عمل عامل‌ها در هر مرحله استفاده شده است، تعداد حرکات لازم برای همگرایی کاهش پیدا کرده است و این بدلیل استفاده مجدد عامل‌ها از دانش یادگرفته شده در مراحل قبل است. حال اگر در الگوریتم Boltzman CBR به جای تابع بولتزمن، از تابع پیشنهادی (رابطه‌ی ۳) استفاده کنیم (الگوریتم پیشنهادی CB-BHAQL) تعداد حرکات لازم برای همگرایی به طرز چشمگیری کاهش می‌یابد و این به دلیل ۱- تلفیق الگوریتم یادگیری Q با تکنیک CBR برای استفاده مجدد عامل‌ها از تجربیات بدست آمده از محیط و ۲- استفاده از تابع پیشنهادی برای انتخاب عمل عامل‌ها در هر حالت است. زیرا در تابع پیشنهادی از هیچ‌گونه پارامتر خارجی برای انتخاب عمل استفاده نشده است این در حالی است که در توزیع بولتزمن از پارامتر خارجی T شده است. استفاده از پارامتر خارجی و انتخاب مقدار نادست برای آن در اکثر مسائل عملی مشکل است و در صورت انتخاب نادرست مقدار آن، موجب عدم همگرایی الگوریتم یادگیری می‌شود. علاوه بر آن در محیط‌های پویا و ناشناخته، عملی که به تعداد بیشتری انتخاب شده است و به تعداد بیشتری، درستی یا نادرستی آن مورد آزمایش قرار گرفته است، گزینه مناسبی برای انتخاب است . این در حالی است که نباید شانس انتخاب اعمال دیگر را نیز از آنها گرفت و بایستی به طریقی بین آنها تعادل ایجاد کرد. از اینرو ماهیت تابع پیشنهادی ما (رابطه‌ی ۳) این است که احتمال انتخاب عملی که دارای بیشترین سابقه‌ی بروزرسانی است، بیشتر است و در عین حال شانس انتخاب اعمال دیگر نیز از دست نرفته است و هر عملی به نسبت تعداد بروز رسانی‌اش و مفید بودنش، دارای شانس انتخاب جداگانه‌ای است. همه‌ی این عوامل موجب شده است که الگوریتم پیشنهادی CB-BHAQL نسبت به الگوریتم‌های دیگر به تعداد حرکات خیلی کمتری برای همگرایی نیاز داشته باشند.



شکل ۵- مقایسه سه الگوریتم از لحاظ تعداد حرکات لازم برای رسیدن به هدف در ۱۰۰۰ بار اجرای الگوریتم‌ها

آزمایش ۲. مقایسه کارایی الگوریتم پیشنهادی با سایر الگوریتم‌ها از نظر میانگین پاداش تجمعی بدست آمده در هر اپیزود. شکل ۶ الگوریتم‌ها را از لحاظ میانگین پاداش تجمعی بدست آمده در هر اپیزود، مقایسه می‌کند. هر چه میانگین پاداش تجمعی بدست آمده برای الگوریتمی بیشتر باشد، الگوریتم کاراتری خواهیم داشت. بهتربن حالت زمانی است که هر دو عامل، همزمان به حالت هدف رسیده و ۲۰۰+ پاداش دریافت کنند. بر طبق شکل ۶ الگوریتم یادگیری Q بدلیل استفاده کردن از تابع بولتزمن برای انتخاب عمل و نیز عدم استفاده از تجربیات قبلی عامل‌ها پاداش کمتری بدست آوره است این در حالی است که استفاده از تکنیک CBR به طرز چشمگیری موجب بهبود پاداش‌های تجمعی بدست آمده شده است و الگوریتم CB-BHAQL با تعداد حرکات کمتر و بدست آوردن پاداش بیشتر، به سمت یک راه‌حل بهینه همگرا می‌شود و برتری الگوریتم پیشنهادی را نسبت به دو الگوریتم دیگر نشان می‌دهد.

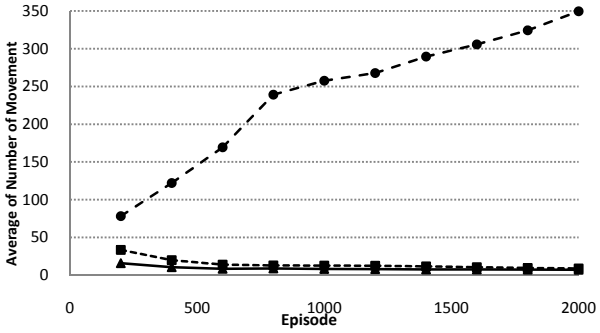


شکل ۶- مقایسه سه الگوریتم از لحاظ میانگین پاداش‌های بدست آمده در ۱۰۰۰ بار اجرای الگوریتم‌ها

www.SID.ir

Archive of SID

آزمایش ۳. مقایسه کارایی الگوریتم پیشنهادی با سایر الگوریتم‌ها از نظر میانگین تعداد حرکات انجام شده، در ۱۰۰۰ بار اجرای الگوریتم‌ها. شکل ۷ نتایج را نشان می دهد. با توجه به شکل دیده می‌شود که عامل‌ها هنگام استفاده از الگوریتم یادگیری Q به طور میانگین به تعداد حرکت بسیار زیادی برای همگرایی نیاز دارند این بدین دلیل است که در هر بار تکرار الگوریتم آموخته‌های قبلی عامل‌ها از محیط از بین می‌رود و عامل مجبور است دانش خود از محیط را از ابتدا کسب کند این در حالی است که اگر آموخته‌های قبلی عامل‌ها از محیط در هربار تکرار الگوریتم در جایی ذخیره شود و از آن در تکرارهای بعدی نیز استفاده شود (الگوریتم CBR-Boltzmann) میانگین تعداد حرکت‌های لازم برای همگرایی به طور بسیار چشمگیری کاهش پیدا می‌کند و طبق شکل ۷ در الگوریتم پیشنهادی CB-BHAQL عامل برای یادگیری سیاست بهینه، به تعداد حرکات بسیار کمتری نسبت به دو الگوریتم قبلی نیاز دارد.



شکل ۷- مقایسه سه الگوریتم از لحاظ میانگین تعداد حرکات انجام شده در ۱۰۰۰ بار اجرای الگوریتم‌ها

۵- بررسی رفتار الگوریتم و تحلیل آن

در این بخش تحلیلی برای نحوه عملکرد الگوریتم پیشنهادی ارائه می شود. که در آن برتری تابع π_2 (S) (رابطه‌ی ۳) نسبت به تابع π_1 (S) (رابطه‌ی ۲) به دو طریق ۱- آزمایش و ۲- تحلیل ریاضی ارائه گردیده است. می‌خواهیم نشان دهیم که در روش پیشنهادی تابع $\pi_1(S) = e^{(Q/\tau)} / \sum_j e^{Qj/\tau}$ نسبت به تابع $\pi_2(S) = e^{nQ} / \sum_j e^{nj\,Qj}$ با سرعت بیشتری به سمت پاسخ بهینه همگرا می‌شود. به عبارت دیگر، آهنگ تغییر π_2 (S) نسبت به Q بیشتر از آهنگ تغییر π_1 (S) نسبت به Q است.

۵-۱- بررسی با استفاده از آزمایش‌های مختلف

برای نشان دادن برتری رفتار تابع انتخاب عمل پیشنهادی، الگوریتم CB-BHAQL برای حالت S_0 و عمل a_1 با توجه به مقادیر مختلف n مورد بررسی قرار گرفته و نتایج زیر بدست آمده است.

نتیجه‌ی ۱: با توجه به نمودار شکل‌های ۸ و ۹ و۱۰ می‌بینیم که با افزایش n رشد π_2 (S) خیلی سریعتر از رشد π_1 (S) است.

نتیجه‌ی ۲: نمودار شکل ۹ و۱۰ و ۱۱ نشان می‌دهد که با افزایش n، مقدار تابع Q(S,a) در رابطه‌ی ۱، افزایش می‌یابد.

نتیجه‌ی ۳: نمودار شکل ۱۲ نشان می‌دهد که با افزایش مقدار Q(S,a)، مقدار تابع π_1 (S) افزایش می‌یابد.

از آنجایی که همواره $\lim_{t\to\infty} n_t(S,a) = \infty$ است، طبق نتیجه‌ی ۲، مقدار Q_t (S,a) افزایش می‌یابد و طبق نتیجه‌ی ۱، با افزایش n، تابع π_2 (S) سریعتر از π_1 (S) رشد می‌کند. با توجه به مطالب قبلی، نتیجه می‌شود که بایستی با افزایش مقدار Q_t (S,a) تابع π_2 (S) سریعتر از تابع π_1 (S) رشد کند. نمودار شکل ۱۲ نتیجه‌ی بدست آمده را تایید می‌کند.

۵-۲- تحلیل ریاضی

برای سادگی محاسبات توابع π_1 (S) و π_2 (S) را به ترتیب به صورت رابطه‌ی ۵ و ۶ بازنویسی می‌کنیم.

$$\pi_1(S) = e^{\frac{Q}{\tau}} \qquad (۵)$$

$$\pi_2(S) = e^{nQ} \qquad (۶)$$

آهنگ تغییر π_1 (S) نسبت به Q با پارامتر $\tau = 0.05$ در رابطه‌ی ۷ آمده است.

$$\Delta\pi_1(S)/\Delta Q = \frac{1}{\tau}e^{\frac{Q}{\tau}} = \frac{1}{0.05}e^{\frac{Q}{0.05}} = 20e^{20Q} \qquad (۷)$$

آهنگ تغییر π_2 (S) نسبت به Q در رابطه‌ی ۸ آمده است.

$$\Delta\pi_2(S)/\Delta Q = d\pi_2(S)/dn \times dn/dQ$$
$$\Delta\pi_2(S)/\Delta Q = Qe^{nQ} \times dn/dQ \qquad (۸)$$

با مقایسه‌ی رابطه‌ی ۷ و ۸ نتایج زیر بدست می‌آید.

تابع $\Delta\pi_1(S)/\Delta Q$ همواره مثبت است.

در رابطه‌ی ۶، بدلیل مثبت بودن Q مقدار تابع $\Delta\pi_1(S)/\Delta Q$ همواره مثبت است. و تابع $\Delta\pi_2(S)/\Delta Q$ همواره مثبت است.

طبق رابطه‌ی ۱ و نمودار شکل ۹ با افزایش n مقدار Q (S,a) همواره افزایش می‌یابد. بنابراین $dn/dQ > 0$ و از طرفی $n > 0$ و $Q > 0$ است. در نتیجه، $\Delta\pi_2(S)/\Delta Q$ همواره مثبت است. آهنگ رشد $\Delta\pi_1(S)/\Delta Q$ نسبت به $\Delta\pi_2(S)/\Delta Q$ کمتر است.

در رابطه‌ی ۶، e^{20Q} در مقدار ثابت ۲۰ ضرب شده است، این در حالی است که در رابطه‌ی ۷، e^{nQ} در متغیر Q ضرب شده است. با افزایش n، مقدار Q افزایش می‌یابد. بنابراین، آهنگ رشد $\Delta\pi_1(S)/\Delta Q$ نسبت به $\Delta\pi_2(S)/\Delta Q$ کمتر است. نمودار شکل ۱۳ نیز نتیجه‌ی بدست آمده را تایید می‌کند.

۶- نتیجه‌گیری

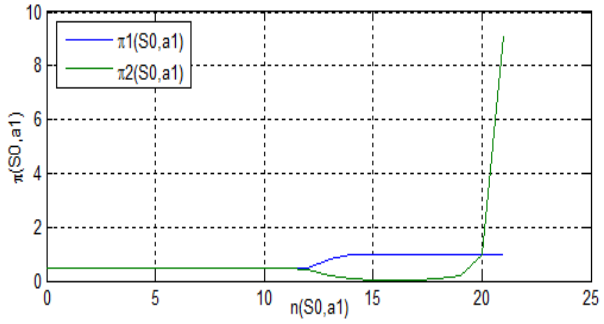
در این مقاله، مدل تلفیقی جدیدی به نام CB-BHAQL برای حل بازی‌های مارکوفی بر اساس یادگیری تقویتی و سیستم‌های مبتنی بر مورد، که در آن از تابع جدیدی برای انتخاب عمل در هر حالت استفاده شده است، ارائه گردید. نتایج بدست آمده، با الگوریتم‌های موجود مقایسه شد. بر طبق نتایج بدست آمده، الگوریتم CB-BHAQL در مقایسه با الگوریتم‌های Q-Learning و Boltzmann CBR، بدلیل استفاده از تکنیک CBR و نیز تابع پیشنهادی جدید، هم از لحاظ سرعت همگرایی به پاسخ بهینه و هم از لحاظ میانگین پاداش تجمعی بدست آمده و نیز تعداد حرکات لازم برای همگرایی به سمت سیاست بهینه، از کارایی بسیار خوبی برخوردار است.

به‌طور کلی اگر در الگوریتم‌های مبتنی بر یادگیری تقویتی Q برای انتخاب عمل عامل‌ها در هر حالت به جای توزیع بولتزمن از تابع پیشنهادی جدید استفاده شود سرعت یادگیری الگوریتم‌های مبتنی بر یادگیری Q بسیار افزایش خواهد

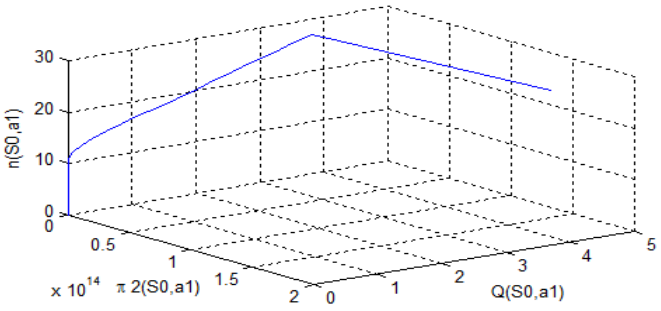
www.SID.ir

Archive of SID

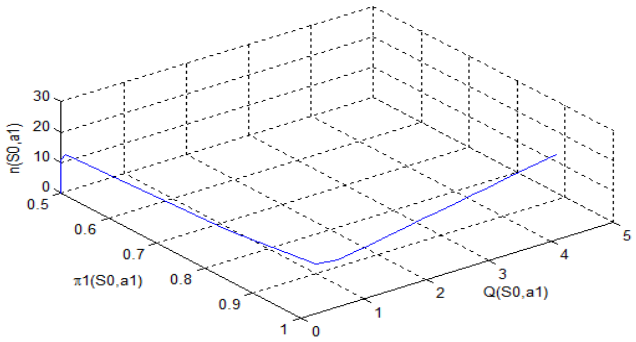
یافت. با اینحال به نظر می‌رسد که تابع پیشنهادی جدید، عمل خود را فقط بر اساس حالت فعلی عامل انتخاب می‌کند و اگر بتوان این تابع را طوری گسترش داد که برای انتخاب عمل عامل‌ها بجز حالت فعلی، حالت‌های همسایه‌ی حالت فعلی را هم در نظر بگیرد نتایج بهتری بدست می‌آید.



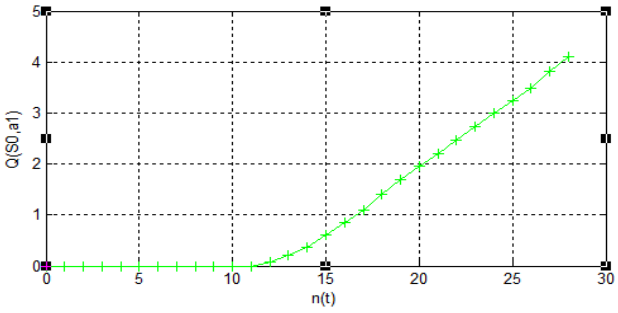
شکل ۸- بررسی رشد π_2 (S) و π_1 (S) بر حسب مقادیر مختلف n



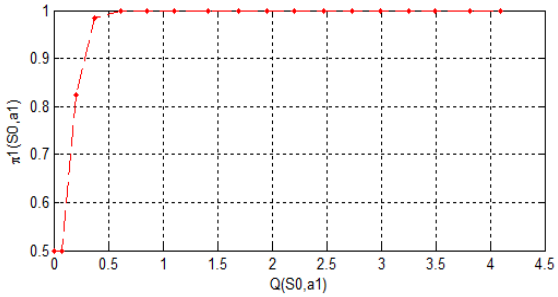
شکل ۹- بررسی رشد π_2 (S) بر حسب مقادیر مختلف n



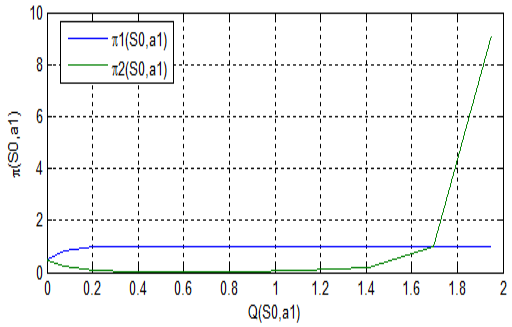
شکل ۱۰- بررسی رشد π_1 (S) بر حسب مقادیر مختلف n



شکل ۱۱- بررسی رشد Q(S,a) بر حسب مقادیر مختلف n



شکل ۱۲- بررسی رشد π_1 (S) بر حسب مقادیر مختلف Q(S,a)



شکل ۱۳- بررسی رشد π_2 (S) و π_1 (S) بر حسب مقادیر مختلف Q(S,a)

مراجع

[1] R. A. C. Branchi, R. Raquel, and R. L. D. Mantaras, "Imroving Reinforcement Learning by using Case Based Heuristics," *Proc, Int'l Conf. Case Based Learning (ICCBR)*, pp. 75-89, 2009.

[2] Y. Shoham, and K. Leyton-Brown, "Multiagent Systems: Algorithmic, Game theoretic and Logical Foundation," Cambridge University Press, 2009.

[3] C. Boutilier, "Sequential optimality and coordination in multi-agent systems," *Proc, Int'l Conf. Artificial intelligence*, vol. 99, no. 1, pp. 478-485, 1999.

[4] B. Masoumi, and M. R. Meybodi, "Learning Automata based Multi-agent System Algorithms for Finding Optimal Policies in Markov Games," *Asian Journal of Control*, vol. 14, no. 4, pp. 1-16, 2012.

[5] J. Hu, and M. Wellman, "Nash Q-Learning for General-Sum Stochastic Games," *Journal of Machine Learning Research*, vol. 4, no. 2, pp. 1039-1069, 2003.

[6] B. Masoumi, M. R. Meybodi, and F. Abtahi, "Learning Automata based Algorithms for Finding Optimal Policies in Fully Cooperative Markov Games," *Proc, Conf. Przeglad Elektrotechniczny,* pp. 280-289, 2012.

[7] M. Laur, and M. Reidmiller, "An Algorithm for Distributed Reinforcement Learning in Cooperative Multi-agent Systems," *Proc, Int'l Conf. Machine Learning*, pp. 535-542, 2000.

www.SID.ir

Archive of SID

[8] S. Kapetanakis, and D. Kudenko, "Reinforcement Learning of Coordination in Hetergenous Cooperative Multi-agent Systems," *Proc, IEEE Int'l Conf. Autonomous Agents and Multiagent Systems (AAMAS)*, pp. 1258-1259, 2004.

[9] L. Matignon, G. J. Lauent, and N. L. Front-Piat, "Hysteretic Q-Learning: An Algorithm for Decentralized Reinforcement Learning in Cooperative Multi-agent Teams," *Proc, IEEE Int'l Conf. Intelligence Robots and Systems (IROS)*, pp. 64-69, 2007.

[10] F. S. Melo, and M. I. Ribeiro, "Reinforcement Learning with Function Approximation for Cooperative Navigation Tasks," *Proc, IEEE Int'l Conf. Robotics and AUtomation*, pp. 3321-2237, 2008.

[11] M. Lauer, and M. Riedmiller, "Reinforcement Learning for Stochastic cooperative Multi-agent Systems," *Proc, IEEE Int'l Conf. Autonomous Agents and Multiagent Systems (AAMAS)*, pp. 1514-1515, 2004.

[12] R. A. C. Bianchi, C. H. C. Ribeiro, and C. Costa, "Accelerating autonomous learning by using heuristic selection of actions," *Journal of Heuristis*, vol. 2, no. 2, pp. 135-168, 2008.

[13] R. A. C. Bianchi, C. H. C. Ribeiro, and C. Costa, "Heuristic selection of actions in Multiagent Reinforcement Learning," *Proc, IEEE Int'l Conf. Artificil Intelligence*, pp. 690-695, 2007.

سارا اسفندیاری در سال ۱۳۸۵ مدرک کارشناسی مهندسی کامپیوتر- نرم‌افزار خود را از دانشگاه آزاد اسلامی تهران مرکزی و در سال ۱۳۸۹ مدرک کارشناسی‌ارشد مهندسی کامپیوتر خود را از دانشگاه آزاد اسلامی واحد قزوین دریافت کرد. زمینه‌های علمی مورد علاقه ایشان عبارتند از: سیستم‌های چند عامله، یادگیری در سیستم‌های چند عامله داده کاوی و الگوریتم‌های موازی.

آدرس پست‌الکترونیکی ایشان عبارت است از:

sara.esfandiari@gmail.com

بهروز معصومی در سال ۱۳۷۴ مدرک کارشناسی مهندسی کامپیوتر- نرم‌افزار خود را از دانشگاه شهید بهشتی تهران و در سال ۱۳۷۷ مدرک کارشناسی‌ارشد مهندسی کامپیوتر خود را از دانشگاه آزاد اسلامی واحد تهران جنوب دریافت کرد. ایشان در سال ۱۳۸۹ موفق به اخذ درجه دکترا در مهندسی کامپیوتر- نرم‌افزار از دانشگاه آزاد اسلامی علوم و تحقیقات تهران گردید. نامبرده از سال ۱۳۷۷ تاکنون، عضو هیات علمی دانشکده مهندسی کامپیوتر و فناوری اطلاعات دانشگاه آزاد اسلامی قزوین می‌باشند. زمینه‌های علمی مورد علاقه ایشان عبارتند از: سیستم‌های چند عامله، یادگیری در سیستم‌های چند عامله، طراحی سیستم‌های پایگاه داده و محاسبات نرم.

آدرس پست‌الکترونیکی ایشان عبارت است از:

masoumi@qiau.ac.ir

محمدرضا میبدی تحصیلات خود را در مقاطع کارشناسی و کارشناسی‌ارشد اقتصاد به‌ترتیب در سال‌های ۱۳۵۲ و ۱۳۵۶ از دانشگاه شهید بهشتی و در مقاطع کارشناسی‌ارشد و دکتری علوم کامپیوتر به‌ترتیب در سال‌های ۱۳۵۹ و ۱۳۶۲ از دانشگاه اوکلاهمای آمریکا به پایان رسانده است و هم‌اکنون استاد دانشکده مهندسی کامپیوتر دانشگاه صنعتی امیرکبیر می‌باشد. نامبرده قبل از پیوستنش به دانشگاه صنعتی امیرکبیر در سال‌های ۱۳۶۲ الی ۱۳۶۴ استادیار دانشگاه میشیگان غربی و در سال‌های ۱۳۶۴ الی ۱۳۷۰ دانشیار دانشگاه اوهایو در ایالات متحده آمریکا بوده است. زمینه‌های تحقیقاتی مورد علاقه ایشان عبارتند از: الگوریتم‌های موازی، پردازش موازی، محاسبات نرم و کاربردهای آن، شبکه‌های کامپیوتری و مهندسی نرم‌افزار.

آدرس پست‌الکترونیکی ایشان عبارت است از:

mmeybodi@aut.ac.ir

اطلاعات بررسی مقاله:

تاریخ ارسال: ۹۰/۴/۱

تاریخ اصلاح: ۹۲/۵/۳

تاریخ قبول شدن: ۹۲/۵/۲۶

نویسنده مرتبط: دکتر بهروز معصومی، دانشکده مهندسی کامپیوتر و فناوری اطلاعات، دانشگاه آزاد اسلامی قزوین، قزوین، ایران.

[1] Case Base Reasoning
[2] Multiagent System
[3] Multi Agent Markov Decision Process
[4] Case Base _ Best Heuristically Accelerated Q-Learning
[5] Markov Decision Process
[6] Evaluation Function
[7] Case Base
[8] Nearest Neighbour
[9] Q-Learning

www.SID.ir