مجله کنترل
ISSN 2008-8345
جلد ۵، شماره ۱، بهار ۱۳۹۰، صفحه ۶۳-۵۰

تقریب تابع ارزش عمل با استفاده از شبکه توابع پایه شعاعی برای یادگیری تقویتی

ولی درهمی[۱]، امید محرابی[۲]

[۱]استادیار دانشکده مهندسی برق و کامپیوتر، دانشگاه یزد، vderhami@yazduni.ac.ir
[۲]فارغ‌التحصیل کارشناسی ارشد مهندسی برق- کنترل، omidmehrabi62@yahoo.com

(تاریخ دریافت مقاله ۱۳۸۹/۱۰/۲۶، تاریخ پذیرش مقاله ۱۳۹۰/۲/۲۱)

چکیده: مشکل تنگنای ابعاد، یکی از چالش‌هایی است که کاربرد الگوریتم‌های یادگیری تقویتی گسسته را در مورد مسائل کنترلی واقعی که دارای فضای حالت و عمل بزرگ و یا پیوسته می باشند محدود نموده است. ترکیب روش‌های آموزشی گسسته با تقریب زننده‌های تابعی برای حل این مشکل چندی است مورد توجه محققان قرارگرفته است. در همین راستا در این مقاله یک الگوریتم جدید یادگیری تقویتی عصبی (NRL) بر مبنای معماری نقاد- تنها معرفی می‌گردد. الگوریتم مذکور از ترکیب الگوریتم یادگیری سارسا با شبکه عصبیRBF به عنوان یک تقریب زننده‌ی تابعی حاصل شده است و ما آن را "یادگیری سارسای عصبی" (NSL) می-نامیم. ورودی‌های شبکه جفت حالت و عمل‌های مسأله و خروجی آن تابع ارزش عمل تقریب زده شده می‌باشد. وزن‌های شبکه به صورت بر خط با توجه به الگوریتم ارائه شده تنظیم می‌گردند. به عنوان یک شرط لازم همگرایی ما همچنین وجود نقاط ایستای منطبق بر نقاط ثابت الگوریتم "تکرار تقریب ارزش عمل" برای NSL را اثبات می‌نماییم. نتایج شبیه سازی ارائه شده در مورد مسائل خودرو در کوهستان و آکروبات حاکی از عملکرد مناسب تر روش ارائه شده از لحاظ سرعت آموزش و کیفیت عملکرد می‌باشد.

کلمات کلیدی: یادگیری تقویتی عصبی، معماری نقاد-تنها، شبکه عصبی RBF، یادگیری سارسا، نقاط ایستا

Action Value Function Approximation Based on Radial Basis Function Network for Reinforcement Learning

Vali Derhami, Omid Mehrabi

Abstract: One of the challenges encountered in the application of classical reinforcement learning methods to real-control problems is the curse of dimensiality. In order to overcome this difficulty, hybrid algorithms that combine reinforcement learning with various function approximators have attracted many research interests. In this paper, a novel Neural Reinforcement Learning (NRL) scheme which is based on Sarsa learning and Radial Basis Function (RBF) network is proposed. The RBF network is used to approximate the Action Value Function (AVF) on-line. The inputs of RBF network are state-action pairs of system and its outputs are corresponding approximated AVF. As the necessary condition for the convergence of NSL to the optimal task performance, the existence of stationary points for NSL which coincide with the fixed points of Approximate Action Value Iteration (AAVI) are proved. The validity of the proposed algorithm is tested through simulation examples: mountain car control task, and acrobot problem. Overall results demonstrate that our algorithm can effectively improve convergence speed and the efficiency of experience exploitation.

Keywords: Neural reinforcement learning, Critic-only architecture, RBF neural network, Sarsa, stationary points.

۱- مقدمه

طراحی کنترلگر بهینه برای سیستم‌هایی با دینامیک پیچیده، جایی که داشتن یک تعامل هوشمندانه با یک محیط پویایی که دارای عدم قطعیت[۱] و نایقینی[۲] نیز هست، همواره به صورت کارا و مؤثر توسط روش‌های کنترل کلاسیک قابل انجام نیست. با توجه به اینکه عموماً نمی‌توان مقدار خروجی مطلوب کنترل کننده در این نوع سیستم‌ها را تعیین نمود، استفاده از روش‌های آموزشِ بدون ناظر برای طراحی و تنظیم پارامترهای کنترلگر ارجحیت دارد [۱]. یادگیری تقویتی[۳] (RL)، یک روش قوی مدرن برای آموزش رویِ خط استراتژی‌های کنترل از طریق تعامل با محیط است. در این روش سیستم تلاش می‌کند تا تقابلات خود با یک محیط پویا را از طریق سعی و خطا بهینه نماید. ایده بنیادی یادگیری تقویتی بر این اصل استوار است که اگر عملی منجر به بهبود عملکرد گردد میل به انجام آن عمل تشدید یا تقویت می‌گردد [۲، ۳]. این روش تنها با استفاده از یک معیار اسکالر راندمان، که سیگنال تقویت یا پاداش نامیده می‌شود، بدون نیاز به سرپرست قادر به آموزش عامل‌ها در محیط‌های پیچیده، ناقطعی و تصادفی[۴] می‌باشد و کنترلگر تنها با توجه به نقد عملکرد کنترل که توسط سیگنال تقویتی بیان می‌گردد، به جای یک خروجی مطلوب داده شده توسط سرپرست، آموزش می‌بیند. قابلیت‌های مذکور باضافه قدرت کاوش[۵] بالا در جهت یافتن پاسخ بهینه و عملکرد در زمان واقعی، آن را تبدیل به یک استراتژی کارآمد برای آموزش کنترلگرهای هوشمند نموده است.

در یادگیری تقویتی گسسته مقدار ارزشِ حالت (یا جفت حالت- عمل) در جدول ارزش[۶] ذخیره شده و در هر قدم که آن حالت (یا جفت حالت- عمل) ملاقات شوند به روز رسانی انجام می‌گیرد [۲]. به کارگیری الگوریتم‌های یادگیری تقویتی گسسته در مسائل کنترل با دو چالش عمده روبروست. ۱) با توجه به اینکه حتی مسائل کنترلی ساده نیز عمدتاً دارای فضای حالت و عمل بزرگ و یا پیوسته می‌باشند و از آنجا که تعداد پارامترهای قابل تنظیم در یادگیری تقویتی گسسته، رابطه مستقیمی با عدداصلی[۷] فضای متغیرهای حالت و عمل مسأله دارد، در نتیجه در مسائل کنترل با مشکل تنگنای ابعاد[۸] مواجه هستیم. ۲) در فرایند آموزش ممکن است بسیاری از حالت‌ها (جفت‌های حالت و عمل) ملاقات نشوند و در نتیجه مقادیر آنها در جدول ارزش به روز رسانی نگردد. واضح است که در این صورت عامل نمی‌تواند عملکرد مناسبی را در مواجهه با این حالت‌ها پس از آموزش داشته باشد. در حالی که اصولاً ما از یک الگوریتم یادگیر انتظار داریم که قدرت استدلال و تعمیم[۹] در مورد موارد ملاقات نشده را نیز دارا باشد. هر دو مشکل نامبرده شده به رویکرد استفاده از تقریب زننده‌های تابع برای تقریب تابع ارزش[۱۰] منجر شده است [۴، ۵]. بر این اساس محققین با ترکیب الگوریتم‌های یادگیری تقویتی گسسته با تقریب زننده‌های تابعی[۱۱]، الگوریتم‌های یادگیری تقویتی پیوسته را، ارائه داده‌اند. کاربردهای گسترده و عملکرد مطلوب شبکه‌های عصبی در کنترل و مسائل پیچیده و نیز مزایایی چون کنترل سیستم‌های غیرخطی با دقت دلخواه، مقاوم بودن و تحمل خطا، قابل یادگیر بودن (یعنی توانایی تنظیم وزن‌های شبکه)، قابلیت تعمیم، سرعت بالا به دلیل پردازش‌های موازی و قابلیت تقریب توابع باعث شده است تا محققین با ترکیب شبکه‌های عصبی به عنوان تقریب زننده با روش‌های یادگیری تقویتی، الگوریتم‌های یادگیری تقویتی عصبی[۱۲] (NRL) را ارائه دهند [۶، ۷]. شبکه‌های MLP و RBF[۱۳] از جمله پرکاربردترین شبکه‌های عصبی به کارگرفته شده بدین منظور بوده‌اند [۱۰-۸].

دومعماری معروف در زمینه الگوریتم‌های یادگیری تقویتی پیوسته عبارتند از: معماری عملگر- نقاد[۱۴] و معماری نقاد- تنها[۱۵]. قالب کلی اکثر پژوهش‌های انجام شده در این زمینه دارای معماری عملگر- نقاد است. این معماری دارای دو بخش عملگر و نقاد بوده و در آن بخش نقاد برای تقریب تابع ارزش و بخش عملگر برای تولید عمل استفاده می گردد [۱۴-۹]. با وجود کاربرد گسترده‌ی الگوریتم‌های عملگر-نقاد عصبی در بسیاری از مسائل توسط محققین، در مطالعات قبلی نشان داده شده است که الگوریتم‌های مذکور دارای فقدان کاوش مناسب می باشند. البته در تعدادی از مقالات [۱۲، ۱۵] برای افزایش درجه کاوش سیستم، خروجی شبکه را با یک مقدار تصادفی بر مبنای تابع توزیع چگالی احتمال گوسی با میانگین صفر جمع کرده و به فرآیند اعمال می کنند. علیرغم بهبودهای صورت گرفته بخاطر لحاظ کردن کاوش در کنترلگر، روش دارای مشکلات زیر می‌باشد: مقدار کاوش، تقریباً در همه وضعیت‌های سیستم یکسان، متقارن، حول یک عمل دارای ماکزیمم ارزش تخمینی و متناسب با مقدار متوسط پاداش‌ها است.

درمقابلِ معماری عملگر- نقاد، روش نقاد- تنها فقط دارای، یک بخش نقاد است که برای تقریب تابع ارزش عمل[۱۶] استفاده می‌شود و عمل نهایی با توجه به مقادیر ارزش تقریب زده شده با استفاده از یکی از روش‌های انتخاب عمل همچون شبه حریصانه[۱۷] و یا بیشینه نَرم[۱۸] تولید

[1]. Nondeterministic
[2]. Uncertainity
[3]. Reinfocement learning (RL)
[4]. Stochastic
[5].Exploration
[6]. Lookup table
[7].Cardinality
[8].Cures of dimensionality

[9]. Generalization
[10].Value function
[11]. Function approximators
[12]. Neural Reinforcement Learning (NRL)
[13]. Radial Basis Function
[14]. Actor-critic
[15]. Critic-only
[16]. Action Value Function
[17]. ∈-Greedy
[18]. Softmax

می‌گردد. این نحوه‌ی انتخاب عمل درجه کاوش بالا را باعث می‌شود [۱۶، ۱۷].

در مقاله [۱۸] از یک شبکه عصبی بازگشتی برای تقریب تابع ارزش عمل در روش یادگیری Q استفاده شده است. ورودی‌های شبکه زوج های حالت وعمل مسأله و خروجی شبکه تابع ارزش عمل تقریب زده شده می‌باشد. در [۱۹] از ترکیب یک شبکه MLP با یک لایه از توابع گوسی یک تقریب زننده نیمه محلی[1] معرفی شده است. ورودی‌های شبکه در این تقریب زننده بردار حالت مسأله می‌باشد. در [۲۰] از یک شبکه عصبی RBF چند ورودی چند خروجی به عنوان تقریب زننده تابع ارزش عمل استفاده شده است در این روش ورودی‌های شبکه بردار حالت مسأله بوده و خروجی‌های شبکه توابع ارزش عمل تقریب زده شده به ازای هر عمل گسسته می‌باشد به عبارت دیگر به تعداد عمل‌های گسسته از شبکه خروجی گرفته می‌شود. هیچ تحلیل ریاضی برای روش های بالا ارائه نشده است و بررسی‌های ما نشان داد که ساختار تقریب زننده و روش انتخاب عمل در الگوریتم‌های فوق به نوعی است که نمی‌تواند فرض‌های مربوط به لم‌ها وقضایای در این زمینه را برآورده کنند.

همچنین در سال‌های اخیر پژوهش‌هایی در مورد حل مسائل کنترل بهینه به صورت بر خط با استفاده از یادگیری تقویتی با فرض داشتن مدل کامل (جزئی) از دینامیک سیستم انجام شده است [۲۷-۲۴]. الگوریتم‌های ارائه شده در مقالات مذکور بر مبنای روش‌های برنامه ریزی پویا[2] می‌باشند. روش‌های برنامه ریزی پویا برای بدست آوردن سیاست بهینه[3] با فرض داشتن مدل کاملی از محیط به کار می‌روند. روش‌های مذکور اگرچه از لحاظ ریاضی قابل توجه‌اند ولی به خاطر احتیاج به مدل کاملی از محیط و همچنین حجم محاسبات گسترده، از اهمیت کمی در یادگیری تقویتی برخوردارند. در روش‌های یادگیری تقویتی مطلوب آن است که همان نتایج حاصل شده از روش‌های مذکور با محاسبات کمتر و بدون فرض داشتن یک مدل کامل از محیط حاصل گردد.

همانگونه که از مباحث بالا دریافت می‌شود عدم وجود تحلیل ریاضی یکی از مشکلات اکثر الگوریتم‌های یادگیری تقویتی پیوسته‌ای است که تاکنون توسط محققین ارائه شده‌اند. با این وجود در تعدادی از مقالات تحلیل‌های ریاضی مثبتی در مورد ترکیب الگوریتم‌های یادگیری تقویتی وابسته به سیاست با تقریب زننده‌های تابعی خطی موجود است. در [۲۱] وجود نقاط ایستا[4] در روش تفاضل موقتی[5] (TD) خطیِ بهره برنده از سیاست انتخابِ عمل بیشینه نَرم اثبات شده است. البته این اثبات تنها مربوط به تقریب تابع ارزش حالت است نه تابع ارزش عمل. بعلاوه، با توجه به آنکه در فرمول بیشینه نَرم مقدار ارزش عمل مورد نیاز است، در مقاله‌ی مذکور مقدار ارزش عمل در هر قدم زمانی، از مقدار تابع ارزش با فرض دانستن مدل محیط محاسبه شده است، در حالی که در مسائل یادگیری تقویتی معمولاً مدل محیط، ناشناخته فرض می‌گردد.

در مقالات [۲۲، ۲۳] از ترکیب روش سارسا[6] (که یک روش وابسته به سیاست است) با تقریب زننده‌های تابعی خطی (ما این ترکیب را سارسای خطی می‌نامیم) برای تقریب تابع ارزش عمل استفاده شده است. در [۱۶، ۲۲] همگرایی پارامترهای وزن سارسای خطی به یک ناحیه در صورت استفاده از یک سیاست ایستا[7] در همه رویدادها[8]، اثبات شده است. در [۲۳] همگرایی الگوریتم ارائه شده به پاسخ یکتا به شرط ایستا بودن سیاست‌ها در هر رویداد اثبات گردیده است. الگوریتم مذکور دارای دو ضعف عمده می‌باشد: ۱- نتایج قضیه، تضمینی در خصوص کیفیت سیاست نهایی که الگوریتم به آن همگرا می‌شود ارائه نمی‌دهد، ۲- با توجه به آنکه پس از تعیین هر سیاست جدید باید آموزش تا همگرایی وزن‌ها ادامه یابد، سرعت آموزش کند می‌باشد. در حالیکه در مسائل کنترل با آموزش رویِ خط، مطلوب آن است که در هر قدم زمانی سیاست انتخاب عمل به روز رسانی گردد. توجه شود که هیچ‌گونه پیاده سازی و شبیه سازی برای سه الگوریتم اشاره شده در بالا [۲۳-۲۱] ارائه نشده و در واقع دو چالش عمده برای پیاده سازی این الگوریتم‌ها وجود دارد: انتخاب تابع تقریب زننده تابعی خطی و نحوه‌ی انتخاب عمل، به نحوی که این دو بتوانند شرایط بیان شده در لم‌ها وقضایای بیان شده در مقالات مذکور را ارضاء نمایند.

در مقاله حاضر سعی داریم با ارائه یک ساختار تقریب زننده عصبی جدید و ترکیب آن با روش یادگیری تقویتی سارسا، یک الگوریتم جدید یادگیری تقویتی پیوسته‌که ما آن را یادگیری سارسای عصبی[9] (NSL)می‌نامیم مورد مطالعه و بررسی قرار دهیم. به عنوان شرط لازم همگرایی برای الگوریتم NSL در این مقاله ما همچنین وجود نقاط ایستای الگوریتم ارائه شده را که منطبق بر نقاط ثابت[10] الگوریتم تکرار تقریب ارزش عمل[11] است ثابت می‌نماییم. براساس تحقیقات ما تاکنون هیچ الگوریتم یادگیری تقویتی عصبی بر مبنای معماری نقاد- تنها با هدف تعیین بهترین عمل در هر حالت (استفاده در کنترل) که دارای تحلیل ریاضی باشد ارائه نشده است. لذا نتایج ارائه شده در این مقاله بعنوان اولین NRL با معماری نقاد- تنها است که دارای تحلیل ریاضی هم می‌باشد.

ساختار این نوشتار به این ترتیب صورت گرفته است. در بخش بعد مفاهیم پایه‌ای یادگیری تقویتی به طور مختصر توضیح داده می‌شوند.

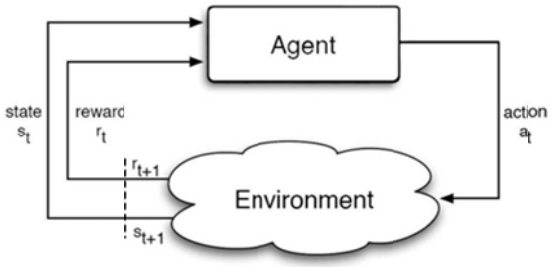
[1]. Semi-localised
[2]. Dynamic programming
[3]. Optimal policy
[4]. Stationary points
[5]. Temporal Difference (TD)
[6]. Sarsa learning
[7]. Stationary
[8]. Episodes
[9]. Neural Sarsa Learning(NSL)
[10]. Fixed points
[11]. Approximate Action Value Iteration

یادگیری سارسای عصبی در بخش سوم توضیح داده شده است. در بخش چهارم وجود نقاط ایستای منطبق بر نقاط ثابت الگوریتم تکرار تقریب ارزش عمل برای الگوریتم NSL ارائه شده در بخش سوم اثبات می‌گردد. در بخش چهارم مسائلی که برای ارزیابی عملی ایده‌ها مورد شبیه سازی قرار گرفته تشریح گشته و مختصری از نتایج شبیه سازی ارائه می‌گردد. نهایتاً بخش پنجم به طرح نتایج مهم، نتیجه‌گیری مطالب و پیشنهادات اختصاص دارد.

۲- مدل تصمیم گیری مارکف و یادگیری تقویتی

چارچوب ریاضی مرسوم محیط استفاده شده در اکثر مسائل RL "مدل تصمیم گیری مارکف[1]" (MDP) می‌باشد [۲۷]. دریک سیستم مارکف حالت بعدیِ محیط و پاداش دریافتی تنها به عمل و حالت قبلی عامل در محیط بستگی دارد. یک MDP یک چندتایی $\langle S, A, \psi, P, R \rangle$ است. جایی که $S = \{s_1, s_2, \dots, s_n\}$ یک مجموعه محدود حالت‌های گسسته محیط، $A = \{a_1, a_2, \dots, a_m\}$ مجموعه محدود عمل‌های گسسته عامل، $\psi \subseteq S \times A$ مجموعه زوج-های حالت و عمل قابل قبول و $P: \psi \times S \to [0,1]$ تابع احتمال گذر از حالت s_t به حالت بعدی $s_{t+1} = s'$ با انجام عمل a_t است به طوری که $\forall (s,a) \in \psi \ \sum_{s' \in S} P(s_t, a_t, s') = 1$ و $R: \psi \to R$ تابع پاداش دریافتی توسط عامل می باشد. در هر مرحله زمانی... $t = 0,1,2,$، عامل حالت جدیدی از محیط را دریافت می کند $s_t \in S$ و بر مبنای این حالت، عمل خود $a_t \in A(s_t)$ را بر اساس سیاست اتخاذ شده انجام می‌دهد. یک گام بعد یعنی در $t+1$، محیط یک پاداش عددی $r_{t+1} = r(s_t, a_t)$ بر حسب عمل عامل به وی می‌دهد و با احتمال $P(s_t, a_t, s')$ به حالت جدید $s_{t+1} = s'$ می رود (شکل ۱).

در یادگیری تقویتی عامل همواره می بایست مصالحه‌ایی بین دو هدف مخالف برقرار کند. از یک طرف کلیه‌ی جفت‌های حالت- عمل مسأله باید به حد کافی برای بدست آوردن دانش مورد کاوش قرار گیرند و از طرف دیگر تجربه‌های بدست آمده باید در انتخاب عمل بکار گرفته شوند. برای آموزش مؤثّر، عمل‌ها باید بگونه‌ای انتخاب شوند که محیط بطور مناسب آزموده شده و از جریمه‌ها نیز حتی الامکان اجتناب گردد. انجام کامل این دو کار بطور همزمان ممکن نیست. به منظور برقراری تعادل بین کاوش و بهره‌برداری از تجربیّات[2] در حین آموزش عامل می‌بایست مکانیزمی برای انتخاب عمل خود داشته باشد تا بدین وسیله تعیین نماید که در هر گام زمانی مناسب‌ترین عمل کدام است. قانونی که عامل با توجه به آن در هر حالت، عملی را برای اجرا انتخاب می‌کند سیاست می‌نامند. سیاست یا استراتژی عامل، $\pi: \psi \to [0,1]$ تابع احتمالی است که احتمال انتخاب شدن هر عمل



شکل ۱: چارچوب یادگیری تقویتی

را در هر حالت و با توجه به گام زمانی می‌دهد. به طور مثال $\pi_t(s,a) = p$ می‌گوید اگر عامل در زمان t در حالت s قرار گرفته باشد یا احتمال p عمل a را انتخاب می کند [۲].

در یادگیری تقویتی هدف عامل در قالب سیگنال پاداشی که از محیط دریافت می کند بیان می‌شود. در هر مرحله زمانی این پاداش به صورت عددی ساده بیان می‌شود $r_t \in R$. در بیانی ساده، هدف عامل بیشینه کردن مجموع این پاداش‌ها در بلند مدت[3] است و این لزوماً به معنای بیشینه کردن پاداش در هر مرحله نیست.

$$E\left\{\sum_{t=0}^{\infty} \gamma^t r_t\right\} \qquad (۱)$$

که در آن E بیانگر امید ریاضی و $\gamma \in [0,1]$ نرخ تنزیل می باشد که اهمیت نسبی پاداش‌های بلند مدت را نسبت به پاداش های کوتاه مدت تنظیم می‌کند. امید ریاضی کل پاداشی که عامل با شروع از حالت $s_t = s$ تحت سیاست π به دست می‌آورد ارزش یک حالت تحت سیاست π، $V^\pi(s)$، می‌نامند [۲، ۲۸]:

$$V^\pi(s) = E_\pi\left\{\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \middle| s_t = s\right\} \qquad (۲)$$

که درآن $\mathrm{E}_\pi\{\}$ نشان دهنده امید ریاضی است در صورتی که از سیاست π پیروی شود. به طور مشابه تابع دیگری به نام $Q^\pi(s,a)$ تعریف می‌شود که بیانگر امید ریاضی کل پاداشی است که عامل با شروع از حالت $s_t = s$، انجام عمل $a_t = a$ و سپس در پیش گرفتن سیاست π بدست می‌آورد [۲]:

$$Q^\pi(s,a) = E_\pi\left\{\sum_{k=0}^{\infty} \gamma^k r_{t+1+k} \middle| s_t = s, a_t = a\right\} \qquad (۳)$$

[1]. Markovian Decision Process (MDP)
[2]. Exploitation
[3]. Long term reward

تابع $Q^{\pi}(s,a)$ را تابع ارزش عمل برای سیاست π می‌نامند. در ادامه روش خطای تفاضل موقتی و یک تعمیم معروف آن که برای تخمین توابع ارزش عمل به کار می روند می‌آید.

۲-۱- یادگیری تفاضل موقتی

روش‌های TD به عنوان یکی از روش‌های پایه ای حل مسأله یادگیری تقویتی، ایده‌ای است که به عنوان هسته مرکزی در یادگیری تقویتی شناخته می‌شود. ساده‌ترین روش تفاضل موقتی که آن را $\mathrm{TD}(0)$ می نامند، برای تخمین تابع ارزش حالت به صورت زیر به کار می‌رود [۲]:

$$V(s_t) \leftarrow V(s_t) + \alpha_t[r_{t+1} + \gamma V(s_{t+1}) - V(s_t)] \qquad (۴)$$

که α نرخ آموزش، s_t بردار حالت، γ نرخ تنزیل[1] و r مقدار پاداش آنی است. در این رابطه عبارت داخل کروشه را خطای تفاضل موقتی می‌نامند.

۲-۱-۱- یادگیری سارسا

یک قدم مناسب جهت حصول به کاوش بالاتر، هنگام بکارگیری یادگیری تقویتی درکنترل، تخمین تابع ارزش عمل بجای تابع ارزشِ حالت می باشد. روش سارسا مقادیر ارزش عمل را تحت سیاست جاری تخمین می‌زند. فرمول به روز رسانی مقادیر ارزش عمل در روش سارسا به صورت زیر است [۲]:

$$Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \alpha_t[r_{t+1} + \gamma Q(s_{t+1},a_{t+1}) - Q(s_t,a_t)] \qquad (۵)$$

در روش سارسا گذر از یک جفت حالت- عمل به جفت حالت- عمل بعدی لحاظ می‌گردد. قاعده به روز رسانی فوق، از همه عناصر چند تایی $(s_t, a_t, r_{t+1}, s_{t+1}, a_{t+1})$ استفاده می‌کند، به همین دلیل با توجه به حروف این چند تایی روشِ مذکور *Sarsa* نامیده شده است.

مقادیر ارزش در هر دو الگوریتم شرح داده شده در بالا، تحت فرض‌های زیر به مقداربهینه همگرا می‌شوند [۳۲-۳۰]:

فرض ۱: محیط مسأله MDP غیر نوسانی[2]، کاهش ناپذیر[3] و با سیگنال های تقویت محدود است.

فرض ۲: نرخ آموزش مثبت، کاهشی و شرایط زیر را، ارضا می‌کند.

$$\sum_{t=0}^{\infty} \alpha_t = \infty, \qquad \sum_{t=0}^{\infty} {\alpha^2}_t \leq \infty \qquad (۶)$$

۳- یادگیری سارسای عصبی

یکی از قدرتمندترین شبکه‌های عصبی مورد استفاده درمسائل تخمین تابع، شبکه عصبی RBF است.از مزایای این شبکه می‌توان به سرعت همگرایی بالا، ساختار ساده و در عین حال همگرایی به جواب بهینه دقیق اشاره نمود. این شبکه، الگوی ورودی n بعدی را با استفاده از گره‌های لایه‌ی پنهان به یک الگوی خروجی m بعدی نگاشت می‌کند. بنابراین از این شبکه‌ها می‌توان در هر مسأله‌ای که حل آن نیاز به یک نگاشت از فضای ورودی به خروجی باشد استفاده کرد. شبکه عصبی RBF نرمالیزه شده[4] (NRBF) به وسیله نرمالیزه کردن توابع گوسی با جمع جبری خروجی کل توابع گوسی حاصل می‌شود. استفاده از شبکه NRBF در یادگیری تقویتی به واسطه‌ی حذف تأثیرات محدوده‌های توابع گوسی و عملکرد بهتر در تقریب توابع، مناسب‌تر است. در این مقاله یک ساختار تقریب زننده عصبی جدید مطابق با شکل (۲) که در آن از یک شبکه عصبی NRBF چهار لایه با یک نرون در لایه‌ی خروجی برای تقریب تابع ارزش عمل، پیشنهاد شده است.

لایه‌ی ورودی شبکه شامل $n+1$ نرون می‌باشد. $x(t) = (s_t, a_t)^T \in R^{n+1}$ که متشکل از جفت حالت و عمل‌های مسأله است. که در آن $S_t = (s_1, s_2, \dots, s_n)^T \in R^n$ بردار حالت‌های مسأله می‌باشد و همچنین $a_t \in A$ که $A = [a_1, a_2, \dots, a_M]$ بردار عمل‌های مسأله است.

لایه‌ی ورودی، بردار ورودی $x_t \in R^{n+1}$ را به هر یک از گره‌های لایه‌ی پنهان ارسال می‌کند. در شبکه‌ی RBF لایه‌ی پنهان از نرون‌ها با توابع فعالساز شعاعی تشکیل شده است. متداول‌ترین تابع فعالساز در این نوع از شبکه ها تابع انتقال گوسی است.

به ازای هر متغیر ورودی $x_i = (x_1, x_2, \dots, x_{n+1})$، k_i تابع پایه حالت و عمل گوسی به فرم زیر تعریف می‌گردد:

$$\varphi_{ip}(x_i) = exp\left(-\frac{\|x_i - c_{ip}\|^2}{\sigma_{ip}^2}\right), p = 1,2,\dots,k_i \qquad (۷)$$

که در آن c_{ip} را مرکز[5] گوسی می‌نامیم و σ_{ip} نیز مشخص کننده میزان گستردگی[6] گوسی است که در آن مقدار تابع به طور مشخص از صفر متفاوت است. $\|.\|$ نشان دهنده نرم اقلیدسی[1] است.

[1]. Discount factor
[2]. Aperiodic
[3]. Irreducible
[4]. Normalized Radial Basis Function (NRBF)
[5]. Center
[6]. Width

برای یک الگوی ورودی x خروجی j- امین گره $\varphi_j:\mathcal{R}^{n+1}\rightarrow\mathcal{R}$ در لایه‌ی پنهان برابر است با:

$$\varphi_j(x_1,x_2,\dots,x_{n+1})=\prod_{i=1}^{n+1}\varphi_{ip}=exp\left(-\sum_{i=1}^{n+1}\frac{\|x_i-c_{ip}\|^2}{\sigma_{ip}^2}\right),j=1,2,\dots,h \quad (۸)$$

نرمالیزه سازی در لایه‌ی سوم به صورت زیر انجام می‌گیرد:

$$\phi_j(x)=\frac{\varphi_j(x)}{\sum_{k=1}^{h}\varphi_k(x)},j=1,2,\dots,h \quad (۹)$$

خروجی شبکه $\hat{Q}:\mathcal{R}^{n+1}\rightarrow\mathcal{R}$ با استفاده از حاصلجمع وزندار خطی پاسخ‌های لایه‌ی پنهان در گره‌ی خروجی بدست می‌آید:

$$\hat{Q}_t(s_t,a_t)=\sum_{j=1}^{h}\phi_j(x)w_j(t) \quad (۱۰)$$

که $w_1,w_2,\dots,w_h$ پارامترهای وزن شبکه عصبی و $\phi_1,\phi_2,\dots,\phi_h$ توابع پایه ارزش عمل نرمالیزه شده روی فضای حالت- عمل می‌باشند [۲۳].

رابطه (۱۰) را می‌توان به فرم زیر به صورت برداری نوشت:

$$\hat{Q}_t(s_t,a_t)=\phi^T(s,a)w_t \quad (۱۱)$$

که $\phi^T(s,a)=[\phi_1(s,a),\dots,\phi_h(s,a)]$ ترانهاده ماتریس $\phi(s,a)$ می‌باشد. ماتریس فضای حالت-عمل Φ با ابعاد $|S|\times k|A|$ را به صورت زیر تعریف می‌کنیم:

$$\Phi=\begin{bmatrix}|&&|\\ \phi_1(s,a)&\dots&\phi_k(s,a)\\ |&&|\end{bmatrix}=\begin{bmatrix}\phi^T(s_1,a_1)\\ \vdots\\ \phi^T(s_N,a_M)\end{bmatrix} \quad (۱۲)$$

که S مجموعه حالت‌ها، A مجموعه عمل‌ها، $N=|S|$ و $M=|A|$ است. لذا داریم:

$$\hat{Q}_t=\Phi w_t \quad (۱۳)$$

بردار مراکز $c_r=[c_{i1},c_{i2},\dots,c_{ik_i}]$ برای k_i تابع پایه گوسی تعریف شده برای ورودی x_i به صورت زیر تنظیم می‌گردند:

$$c_{i1}=x_{i_{min}} \quad (۱۴)$$

$$c_{ip}=c_{i(p-1)}+\Delta c_i,\quad for\ p=2,3,\dots,k_i \quad (۱۵)$$

که Δc_i فاصله بین دو مرکز مجاور می‌باشد و بر طبق رابطه زیرتعیین می‌گردد:

$$\Delta c_i=\frac{x_{i_{max}}-x_{i_{min}}}{k_i-1} \quad (۱۶)$$

با این توصیف، در هر مسأله از $h=k_1\times k_2\times\dots\times k_{n+1}$ تابع پایه حالت و عمل بر روی فضای $n+1$ بعدی فضای حالت و عمل برای تقریب تابع ارزش عمل استفاده می‌گردد.

پارامترهای گستردگی گوسی نیز با توجه به نوع مسأله توسط طراح تنظیم می‌گردند. هدف آموزش، به روز رسانی رویِ خط مقادیر وزن شبکه عصبی w به گونه‌ای است که بهترین تقریب از تابع ارزش عمل صورت پذیرد.

مقدار خطای تفاضل موقتی ارزش عمل بین دو حالت متوالی به صورت زیر تقریب زده می‌شود:

$$\Delta\hat{Q}_t(s_t,a_t)=r_{t+1}+\gamma\hat{Q}_t(s_{t+1},a_{t+1})-\hat{Q}_t(s_t,a_t) \quad (۱۷)$$

در این مقاله از روش گرادیان نزولی برای تنظیم بردار وزن در جهت کاهش مربع خطای تابع ارزش عمل تقریب زده شده $E(t)=\frac{1}{2}\left(\Delta\hat{Q}_t\right)^2$ استفاده شده است. پارمترهای وزن شبکه عصبی بر طبق قاعده زنجیری به صورت زیر به روز رسانی می‌گردند:

$$\Delta w_j(t+1)=-\eta_t\frac{\partial E(t)}{\partial w_j(t)}=\eta_t\Delta\hat{Q}_t\frac{\partial Q(s_t,a_t)}{\partial w_j(t)}$$

$$=\eta_t\left(r_{t+1}+\gamma\hat{Q}_t(s_{t+1},a_{t+1})-\hat{Q}_t(s_t,a_t)\right)\frac{\partial Q(s_t,a_t)}{\partial w_j(t)} \quad (۱۸)$$

که در آن η_t نرخ یادگیری وزن‌های شبکه عصبی در بازه‌ی $[0,1]$ در زمان t می باشد. $\frac{\partial Q(s_t,a_t)}{\partial w_j(t)}$ از رابطه (۱۱) به صورت زیر محاسبه می‌گردد:

$$\frac{\partial Q(s_t,a_t)}{\partial w_j(t)}=\phi_j(x_t) \quad (۱۹)$$

[1]. Euclidean

با توجه به این نکته، فرمول به روز رسانی پارمترهای وزن شبکه عصبی به صورت زیر در می‌آید:

$$\Delta w_j(t+1) = \left[\eta_t \times \Delta\hat{Q}_t(s_t,a_t) \times \phi_j(x_t)\right] \quad (۲۰)$$

لازم به ذکر است که در رابطه (۱۷) مقدار $\hat{Q}_t(s_{t+1},a_{t+1})$ وابسته به سیاستی است که عامل با توجه به آن در حالت بعدی، انتخاب عمل می نماید به همین دلیل الگوریتم NSL پیشنهادی یک الگوریتم "وابسته به سیاست[1]" است.

رویه‌ی اجرای یادگیری بر اساس الگوریتم NSL به صورت زیر است:

۱- مشاهده‌ی حالت s_{t+1} و دریافت سیگنال تقویتی r_{t+1}.

۲- توقف آموزش، اگر الگوریتم به تعداد دفعات لازم موفقیت (رسیدن به هدف) و یا ماکزیمم تعداد رویدادها رسیده باشد.

۳- محاسبه‌ی مقدارارزش عمل تقریب زده شده $\hat{Q}_t(s_{t+1},a)$ برای کل عمل های ممکن $a \in A$ با استفاده ازشبکه عصبی RBF.

۴- محاسبه‌ی $\Delta\hat{Q}$ و به روز رسانی مقادیر وزن های شبکه عصبی با استفاده از فرمول‌های (۱۷) و (۲۰).

۵- انتخاب عمل با استفاده از یکی از روش‌های مرسوم انتخاب عمل.

در این مقاله با توجه به آنکه می‌خواهیم از نتایج و فرض‌های ارائه شده در این بخش در تحلیل NSL استفاده کنیم، برای انتخاب عمل از رابطه بیشینه نَرم به فرم زیر استفاده شده است:

$$prob(a_t = a_k) = \frac{exp(\hat{Q}(s_t,a_k)/T)}{\sum_{b\epsilon A} exp(\hat{Q}(s_t,b)/T)} \quad (۲۱)$$

که در آن $T > 0$ ضریب دما[2] نامیده می‌شود. معمولاً در ابتدای آموزش مقدار ضریب دما بزرگ و در حین آموزش هر چه به سمت جلو می رویم مقدار ضریب دما کاهش می‌یابد تا از تجربیات قبلی بیشتر استفاده گردد. گذشت زمان در قالب تعداد تلاش‌های انجام شده و تعداد مراحل به روز شدن جدول دانش عامل و یا وزن‌های شبکه نشان داده می‌شود.

روش بیشینه نرم دارای ویژگی‌های مطلوب زیر است:

۱- تابع توزیع احتمال عمل در فرمول بیشینه نَرم از توزیع بولتزمن که یک توزیع پیوسته است، تبعیت می‌کند. این ویژگی یک خاصیت مهم از لحاظ همگرایی در یادگیری تقویتی پیوسته می باشد [۲۳].

۲- مقدار احتمال انتخاب عمل در فرمول بیشینه نَرم به همه مقادیر ارزش وابسته است، این ویژگی امکان کاوش مناسب را فراهم می کند [۲].

۳- با تعیین مقدار ضریب دما می‌توان میزان کاوش و بهره‌برداری از تجربیّات را کنترل کرد.

۶- محاسبه‌ی مقدار ارزش عمل تقریب زده شده جدید $\hat{Q}_{t+1}(s_{t+1},a)$ برای کلیه عمل‌ها $a \in A$ با استفاده از فرمول (۱۱).

۷- اعمال عمل انتخاب شده به محیط.

۸- $t \leftarrow t+1$ و بازگشت به قدم اول.

۴- تحلیل ریاضی الگوریتم NSL

مهمترین نتایج این مقاله در این بخش ارائه می‌گردد که در آن ما وجود نقاط ایستای منطبق بر نقاط ثابت الگوریتم تکرار تقریب ارزش عمل را برای NSL اثبات می کنیم.

۴-۱- اثبات وجود نقاط ایستا برای الگوریتم NSL

همچنانکه نشان داده خواهد شد، NSL یک پیاده سازی از الگوریتم سارسای خطیِ بهره برنده از سیاست بیشینه نَرم می‌باشد. لذا در ابتدا شرایطی را که تحت آن وجود نقاط ایستا برای روش سارسای خطی وقتی که انتخاب عمل در آن بر اساس فرمول بیشینه نَرم انجام می‌پذیرد، را بیان نموده و سپس نشان می دهیم که این شرایط برای NSL نیز برقرار است.

در الگوریتم سارسای خطی از K تابع پایه برای تقریب تابع ارزشِ عمل $Q(s,a)$ در قدم t به فرم زیر استفاده می شود [۱۶]:

$$\hat{Q}_i(W) = \sum_{i=1}^{k} w_k f_k(i) = W^T F(i) \;\; \forall i \in (s,a) \quad (۲۲)$$

که w بردار وزن‌های قابل تنظیم و $F_i = \left(f_1(i),\dots,f_k(i)\right)$ توابع پایه حالت- عمل روی فضای حالت- عمل مسأله می‌باشند. ماتریس قطری D_π با ابعاد $|S|$. $|A| \times |S|$ شامل المان‌های احتمال حالت ماندگار هر حالت- عمل تحت سیاست π را در نظر بگیرید. با استفاده از روش

[1]. On-policy
[2]. Temperature factor

گرادیان نزولی برای تنظیم بردار وزن در جهت کاهش مجموع مربعات خطای $\left\|Q-\hat{Q}\right\|_{D_\pi}^2$ فرمول به روز رسانی بردار وزن پس از اجرای عمل a_t در حالت s_t برای روش سارسای خطی به صورت زیر حاصل می‌شود [۳۳]:

$$w_{t+1}=w_t+\alpha_t\phi^T(s_t,a_t)[r_{t+1}+\gamma\phi^T(s_{t+1},a_{t+1})w_t-\phi^T(s_t,a_t)w_t] \quad (۲۳)$$

که r_{t+1} پاداش آنی بعد از اجرای عمل a_t، s_{t+1} حالت بعدی، و a_{t+1} عمل بعدی می‌باشد.

لازم به ذکر است که برای تحلیل NSL از روابط و ویژگی‌های یک نسخه از الگوریتم "تکرار تقریب ارزش عمل" استفاده شده است. یک نسخه از الگوریتم مذکور در [۱۶] ارائه گردیده و وجود حداقل یک نقطه ثابت برای آن اثبات شده است.

لم۱: الگوریتم سارسای خطی(۲۳) تحت فرض‌های ۱ تا ۴ دارای نقاط ایستای منطبق بر نقاط ثابت روش تکرار تقریب ارزش عمل می‌باشد.

فرض ۳: توابع پایه حالت– عمل $\{F_i|\ i=1,2,\dots,k\}$ در رابطه‌ی (۲۲) مستقل خطی هستند.

فرض ۴: سیاست انتخاب عمل در هر حالت با استفاده از فرمول بیشینه نَرم سنتی به صورت زیر به دست می‌آید [۲، ۳۴]:

$$\pi_w^\beta(s,a)=\frac{\exp[\beta\ .\ \hat{Q}(s_t,a_t)]}{\sum_{b\epsilon A}\exp[\beta\ .\ \hat{Q}(s_t,b)]} \quad (۲۴)$$

که در آن β ضریب کاوش می‌باشد و به صورت عکس ضریب دما تعریف می شود. از آنجا که عمل ها بر طبق تابع ارزش عمل تقریب زده شده، حاصل می‌گردند احتمال انتخاب عمل به β و w وابسته است؛ به همین دلیل β و w در اندیکس π در روابط آورده شده‌اند.

قضیه ۱: الگوریتم یادگیری سارسای عصبی (NSL) ارائه شده، تحت فرض‌های ۱ و ۲ دارای نقاط ایستای منطبق بر نقاط ثابت روش تکرار تقریب ارزش عمل می‌باشد.

اثبات: کافی است نشان دهیم که NSL، یک پیاده سازی از الگوریتم سارسای خطی(۲۳) است. به سادگی می توان نشان داد که با جایگزینی رابطه‌ی (۱۱) در رابطه‌ی (۱۷) و (۲۰)، رابطه‌ی (۲۳) حاصل می شود. بنابراین قانون به روز رسانی وزن‌های الگوریتم NSL معادل با الگوریتم سارسای خطی است.

اکنون با توجه به لم ۱، اگر ما ثابت کنیم که شرایط ذکر شده در فرض‌های ۳ و ۴ برای NSL برقرار است، آنگاه نتایج لم ۱ هم برای NSL صادق است و قضیه اثبات شده است.

به عبارت دیگر باید ثابت شود که: الف) ستون‌های ماتریس Φ مستقل خطی هستند. ۲) تابع احتمال انتخاب عمل در NSL برابر با فرمول بیشینه نَرم سنتی می‌باشد.

الف) برای یک مجموعه گسسته محدود از حالت و عمل می‌توان ماتریس حالت و عمل Φ تعریف شده در رابطه‌ی (۱۲) را، برای NSL به صورت زیر بازنویسی کرد:

$$\Phi=\begin{bmatrix}\phi_1(s_1,a_1) & \cdots & \phi_h(s_1,a_1)\\ \phi_1(s_1,a_2) & \cdots & \phi_h(s_1,a_2)\\ & \vdots & \\ \phi_1(s_1,a_M) & \cdots & \phi_h(s_1,a_M)\\ & \vdots & \\ & \vdots & \\ \phi_1(s_N,a_1) & \cdots & \phi_h(s_N,a_1)\\ \phi_1(s_N,a_2) & \cdots & \phi_h(s_N,a_2)\\ & \vdots & \\ \phi_1(s_N,a_M) & \cdots & \phi_h(s_N,a_M)\end{bmatrix}=[\phi_1|\dots|\phi_h] \quad (۲۵)$$

است. RBF– امین گره i خروجی نرمالیزه شده $\phi_i(s_j,a_k)$ که متمایز RBFاز آنجا که بردار مراکز و پارامترهای گستردگی توابع دارای مرتبه Φ هستند، به سادگی می‌توان استدلال نمود که ماتریس NSLکامل یا به عبارت دیگر توابع پایه ارزش عمل در الگوریتم مستقل خطی هستند.

ب) از آنجا که استرتژی انتخاب عمل ما در این مقاله بیشینه نَرم انتخاب شده بود، بنابراین فرض ۴ نیز برقرار است. □

۵- شبیه سازی

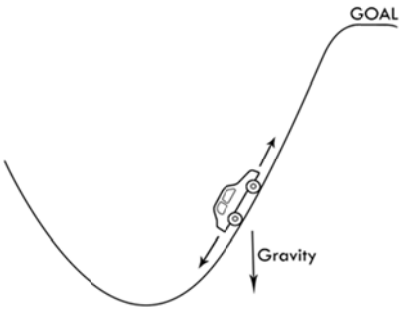
به منظور ارزیابی عملکرد استراتژی آموزشی ارائه شده و همچنین مقایسه آن با روش ارئه شده در مقاله [۲۰]، از این دو روش در حل مسائل حالت پیوسته‌ی خودرو در کوهستان و آکروبات استفاده می کنیم.

۵-۱- مسأله خودرو در کوهستان

مسأله خودرو در کوهستان[1] یکی از مسائل معروف در زمینه یادگیری تقویتی است، که در مقالات مختلف [۱۴، ۳۵] و همچنین مسابقات علمی مرتبط در این زمینه [۳۶] جهت ارزیابی الگوریتم‌های یادگیری تقویتی جدید از آن استفاده شده است. اتومبیلی را در سراشیبی یک جاده کوهستانی مانند شکل (۳) در نظر بگیرید. هدف راندن ماشین به بالای تپه می‌باشد، اما از آنجا که شتاب گرانش زمین قویتر از نیروی محرک موتور اتومبیل است، قادر نیست به سادگی از سطح شیب دار

[1].Mountain car problem

جاده حتی با تمام نیرو بالا رود. تنها راه حل ممکن در این شرایط بازگشت به عقب و گرفتن شتاب لازم جهت غلبه بر شتاب گرانش و عبور از شیب جاده می‌باشد. متغیرهای ورودی، موقعیت اتومبیل (x) و سرعت آن (v) می‌باشند. هدف از یادگیری، تنظیم روی خط وزن های یک کنترلگر عصبی به گونه‌ای است که بتواند نیروی کنترلی لازم (F) جهت راندن اتومبیل به بالای تپه از هر موقعیت و سرعت اولیه را در حداقل زمان فراهم نماید.



شکل ۳: مسأله خودرو در کوهستان

۵-۱-۱- دینامیک سیستم

معادلات حرکت اتومبیل به صورت زیر است [۲، ۳۵]:

$$v_{t+\tau} = \min\Big(0.07, \max\big(-0.07, v_t + 0.001 \times F_t + \mathrm{g} \times \cos(3x_t)\big)\Big)$$

$$x_{t+\tau} = \min\big(0.5, \max(-1.2, x_t + v_{t+1})\big) \qquad (۲۶)$$

که در آن $F \in \{-1,0,1\}$، کل عمل‌های ممکن قابل انجام در این مسأله می‌باشد. محدوده تغییرات موقعیت ماشین (x) و سرعت آن (v) به صورت زیردر نظر گرفته می‌شوند:

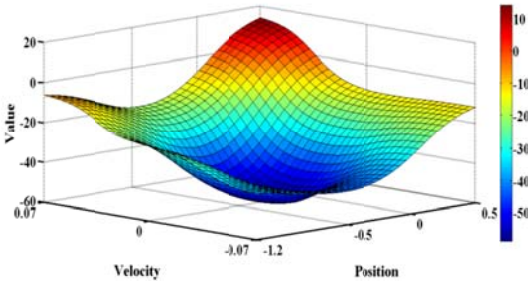
$$\{(x, v \in R^2) | -1.2 \le x \le 0.5, -0.07 \le v \le 0.07\} \qquad (۲۷)$$

هنگامی که موقعیت اتومبیل به نقطه‌ی $x_t = -1.2$ برسد، سرعت ماشین صفر شده و رسیدن به موقعیت $x = 0.5$ هدف عامل بوده و به عنوان موفقیت در نظر گرفته می‌شود.

سیگنال تقویتی در این مسأله به صورت زیر تعریف می‌گردد:

$$r(x, v) = \begin{cases} +1 & x = 0.5 \\ -1 & \text{otherwise} \end{cases} \qquad (۲۸)$$

شکل (۴) منحنی ارزشِ عمل با بالاترین ارزش $\max_a \left(\hat{Q}(s,a)\right)$ در مسأله خودرو در کوهستان را نشان می‌دهد.[۱] همانطور که از شکل مشخص است مقدار ارزش در نواحی نزدیک به هدف و نواحی سمت چپ مسیر حرکت ماشین بیشتر از قسمت های دیگر است.



شکل ۴: ارزشِ عمل با بالاترین ارزش در NSL

۵-۱-۲- جزئیات آموزش

در این مسأله ورودی‌های شبکه عصبی $x = [s,a]^T = [x,v,a]^T$ متشکل از موقعیت و سرعت عامل $s \in [x,v]$ باضافه تک تک عمل‌ها $a \in A = [-1,0,+1]$ در آن حالت می‌باشند. در این مسأله از $3 \times 3 \times 3$ تابع پایه حالت و عمل برای تقریب تابع ارزش عمل در روش NSL استفاده شده است. برای اینکه ارزیابی دقیق و عادلانه‌ای در مقایسه بین دو روش داشته باشیم تمامی شرایط در هر دو روش یکسان در نظر گرفته شده‌اند بنابراین در شبیه سازی روش ارائه شده در مقاله [۲۰] نیز ۳ تابع پایه گوسی بر روی هر یک از متغیرهای حالت ورودی تعریف شده است. در هر دو الگوریتم $\lambda = 0.99$ و همچنین نرخ نمونه برداری ۰.۰۲ ثانیه در نظر گرفته شده است. نتایج هر آزمایش متوسط ارزیابی انجام شده در ۳۰ اجرا است. در ابتدای هر اجرا مقادیر وزن‌های شبکه عصبی با مقدار اولیه صفر مقداردهی می شوند.

هر اجرا شامل دو بخش آموزش و تست می‌باشد. اگر تعداد رویدادها (منظور از یک رویداد شروع از نقطه آغاز و رسیدن به هدف می باشد) از ۲۰۰ بیشتر شود یا تعداد دفعات متوالی که عامل به هدف رسیده به ۴۰ برسد بخش آموزش پایان می‌یابد. شماره رویداد در پایان بخش آموزش به عنوان "معیار زمان آموزش" (LDI)[۲] در نظر گرفته می‌شود. هر رویداد از یک نقطه‌ی تصادفی آغاز می‌شود و زمانی پایان می‌پذیرد، که یا عامل به هدف برسد و یا تعداد قدم‌های انجام شده توسط عامل در یک رویداد از ۷۰۰ بیشتر شود. تعداد قدم‌های انجام شده در رسیدن به هدف در یک رویداد به عنوان معیار "تعداد قدم زمانی رسیدن به هدف" (LE)[۳] در نظر گرفته می‌شود. همچنین درصد موفقیت عامل در رسیدن به هدف به عنوان "معیار نرخ موفقیت[۴]" لحاظ می‌گردد. بخش تست شامل ۴۰ رویداد با شروع از یک نقطه تصادفی می‌باشد.

۱. این شکل توسط الگوریتم NSL تولید شده است.

2. Learning Duration Index
3. Length of Episode
4. Success Rate

جدول ۱: نتایج شبیه سازی برای مقادیر متفاوت نرخ آموزش اولیه و ضریب دمای اولیه

Initial Parameters	Method	Avg. LDI	Success Rate	LE	Success Rate (Test)	LE (Test)
$T_0 = 1.0$	NSL	47.83	89.75	147.34	98.25	67.42
$\eta_0 = 0.2$	method's [20]	51.73	90.85	143.74	97.33	77.12
$T_0 = 0.1$	NSL	44.26	93.07	128.11	98.33	68.66
$\eta_0 = 0.2$	method's [20]	45.16	92.62	130.55	97.11	76.58
$T_0 = 0.01$	NSL	50.30	89.86	165.28	82.93	170.32
$\eta_0 = 0.1$	method's [20]	51.14	89.44	164.41	81.92	175.41
$T_0 = 5.0$	NSL	53.50	85.29	180.87	99.16	64.05
$\eta_0 = 0.2$	method's [20]	53.06	84.55	184.52	97.83	71.03

LDI, Learning duration index; LE, Length of Episode; NSL, Neural Sarsa Learning.

نرخ یادگیری و ضریب دما مطابق با روابط زیر در هر مرحله آموزش به صورت زیر کاهش می‌یابند [۱۶]:

$$\eta_t = \begin{cases} \eta_{t-1}/1.001 & \text{if } t = 4k \\ \eta_{t-1} & \text{otherwise} \end{cases} \quad (۲۹)$$

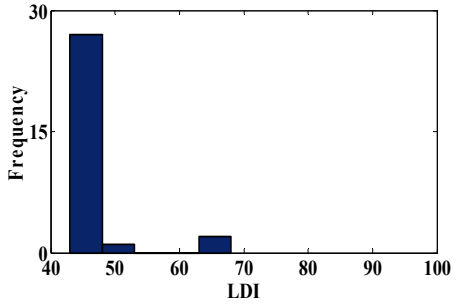
$$T_t = \begin{cases} T_{t-1} - (0.99)^t \times 0.4 \times T_t & if\ t = 4k, t < 25 \\ T_{t-1} - (0.99)^t \times 0.2 \times T_t & if\ t = 4k, t \geq 25 \\ T_{t-1} & otherwise \end{cases} \quad (۳۰)$$

که $t \geq 1$ شماره رویداد می‌باشد. همچنانکه از روابط (۲۹) و (۳۰) مشخص است مقدار ضریب دما و نرخ آموزش در طول هر رویداد ثابت است و در شروع هر رویداد طبق روابط مذکور مقدار آن‌ها تعیین می گردد.

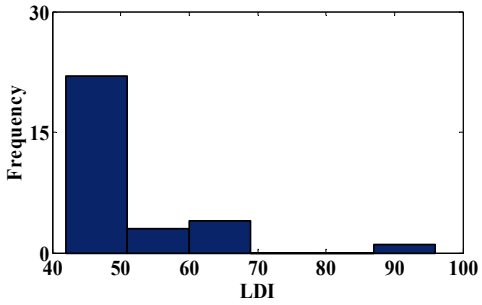
به منظور افزایش درجه کاوش و جلوگیری از گیر افتادن در مینیمم-های محلی، در صورتی که الگوریتم نتواند بعد از ۷۰ رویداد در بخش آموزش همگرا گردد ضریب آموزش روی مقدار اولیه و ضریب دما روی ۲۵٪ مقدار اولیه تنظیم می گردد.

۵-۱-۳- نتایج شبیه سازی

جدول (۱) نتایج ۴ سری آزمایش را بر روی هر دو روش برای مقدارهای اولیه متفاوت نرخ یادگیری و ضریب دما نشان می‌دهد. ستون های ۳ تا ۵ به ترتیب بیانگر معیار زمان آموزش، تعداد قدم های انجام شده در رسیدن به هدف و نرخ موفقیت، در بخش آموزش می‌باشند. ستون های ۶ و ۷ نیز به ترتیب بیانگر نرخ موفقیت و تعداد قدم های انجام شده در رسیدن به هدف در بخش تست هستند. همچنانکه نتایج شبیه سازی نشان می‌دهد، زمان آموزش وکیفیت عملکرد در مجموع در روش NSL بهتر از روش ارائه شده در [۲۰] می‌باشد. در حقیقت ساختار تقریب زننده در NSL به نوعی است که تقریب مناسب‌تری از ارزش عمل صورت می‌پذیرد. هیستوگرام LDI ها برای هر دو

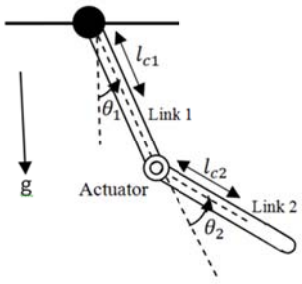


شکل ۵: هیستوگرام LDI ها در NSL



شکل۶: هیستوگرام LDI ها در [۲۰]

الگوریتم در شکل‌های (۵) و (۶) برای حالت $\eta_0 = 0.2$ و $T_0 = 1.0$ نشان داده شده است. همانگونه که از شکل‌ها مشخص است هیستوگرام های رسم شده نیز پراکندگی کمتر LDI ها در NSL را نشان می‌دهد که مؤید عملکرد بهتر روش NSL می باشد. نتایج ارائه شده در جدول (۱)، حاکی از وابستگی کیفیت و زمان آموزش در هر دو الگوریتم به مقدار اولیه ضریب دما است. هر چه ضریب دما کاهش می‌یابد سرعت آموزش سریع‌تر می‌شود. اما این کاهش ضریب دما، کیفیت آموزش را نیز کاهش می‌دهد. اگر ضریب دما خیلی کوچک شود، سیاست استفاده شده به سیاست حریصانه همگرا می‌شود. در این حالت زمان آموزش برای الگوریتم‌ها کوتاه شده و سیستم به سرعت به یکی از نقاط اکسترمم همگرا می گردد، لیکن به دلیل نبود



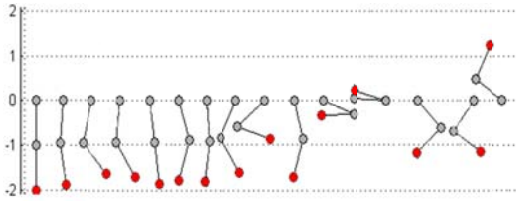
شکل ۷: مسأله آکروبات

کاوش کافی، کیفیت آموزش افت قابل ملاحظه‌ای می‌نماید. مقادیر بزرگ ضریب دما کاوش بالاتری را به سیستم می‌دهد، اما اگر این ضریب خیلی بزرگ باشد به علت فضای بزرگ مسأله، عامل دچار سردرگمی می‌شود و زمان آموزش و کیفیت آموزش افت می‌نمایند. از این رو یک مقدار میانی برای ضریب دما جواب مناسب تری داده است. باید توجه شود که در فرمول بیشینه نَرم، درجه کاوش به ضریب دما و اختلاف مقادیر ارزش عمل وابسته است. اگر اختلاف مقادیر ارزش عمل کوچک باشد، آنگاه انتخاب یک ضریب دمای متوسط باعث کاوش بالا می‌شود. همچنین اگر اختلاف مقادیر ارزش عمل بزرگ باشد، یک ضریب دمای نسبتاً کوچک می‌تواند باعث استفاده از تجربیّات بالایی گردد. با توجه به نکات فوق، مقدار اولیه ضریب دما باید به گونه‌ای انتخاب شود که اولاً در شروع آموزش که اختلاف بین مقادیر ارزش عمل کوچک است، میزان کاوش نسبتاً زیاد باشد، ثانیاً مقدار ضریب دما پس از حدود ۲۵ رویداد به حد کافی کوچک شده باشد به گونه‌ای که اگر اختلاف بین مقادیر ارزش عمل زیاد شده است، انتخاب عمل به سمت سیاست حریصانه نزدیک گردد.

۵-۲- مسأله آکروبات

آکروبات[1] در واقع یک روبات صفحه‌ای دو لینکی است (شکل۷)، که از نظر فیزیکی رفتار یک ژیمناستیک کار که با خم و راست نمودن کمر خود سعی می کند تا انرژی لازم را برای بالا کشیدن بدن خود به بالای میله ای که از آن آویزان شده را فراهم آورد، مدل می‌کند [۳۷، ۳۸]. در این مسأله هدف بالا آوردن نوک لینک دوم (پای ژیمناستیک کار) به اندازه یک واحد بالای نقطه‌ی آویزمی‌باشد. برای این منظور ربات باید حالت تاب خوردن گرفته تا بتواند انرژی لازم را بدین منظور به دست آورد. متغیرهای حالت مسأله، شامل زاوایای دو لینک و مشتقات‌آنها می‌باشد. در این مسأله سرعت زاویه‌ای لینک اول به بازه‌ی $\dot{\theta}_1 \in [-4\pi, 4\pi]$ و سرعت زاویه‌ای لینک دوم به بازه‌ی $\dot{\theta}_2 \in [-9\pi, 9\pi]$ محدود شده‌اند [۶]. شکل (۸) نمایی از مجموعه حرکات آکروبات برای رسیدن به هدف را با نرخ نمونه برداری ۰.۴ ثانیه نشان می‌دهد.

[1]. Acrobot



شکل ۸: نمایی از مجموعه حرکات آکروبات با نرخ نمونه برداری ۰.۴ ثانیه

جدول۲: پارامترهای مسأله آکروبات شبیه سازی شده

Parameter	Description	Value
l_1, l_2	Link lengths	1.0
lc_1, lc_2	Joint to mass center	0.5
m_1, m_2	Link masses	1.0
I_1, I_2	Link inertias	1.0
τ	Torque range	$\{-1, 0, 1\}$
g	Gravitational force	9.8
t_i	Integration time step	0.05
t_c	Control time step	0.2

۵-۲-۱- دینامیک سیستم

معادلات دینامیکی حاکم بر سیستم به شکل زیر می باشند [۶، ۳۸]:

$$\ddot{\theta}_1 = -{d_1}^{-1}\left(d_2\ddot{\theta}_2 + \phi_1\right) \quad (۳۱)$$

$$\ddot{\theta}_2 = \left({m_2 {l_{c2}}^2} + I_2 - \frac{{d_2}^2}{d_1}\right)^{-1}\left(\tau + \frac{d_2}{d_1}\dot{\phi}_1 - \dot{\phi}_2\right) \quad (۳۲)$$

که در آن

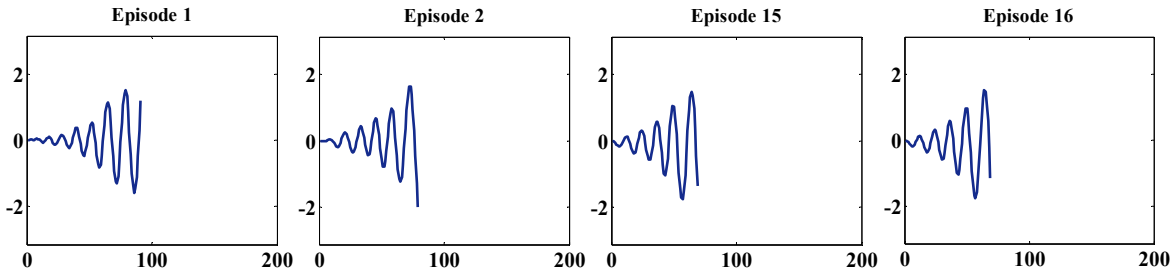
$$d_1 = m_1 {l_{c1}}^2 + m_2\left({l_1}^2 {l_{c2}}^2 + 2l_1 l_2 cos\theta_2\right) + I_1 + I_2 \quad (۳۳)$$

$$d_2 = m_2\left({l_{c2}}^2 + l_1 l_{c2} cos\theta_2\right) + I_2 \quad (۳۴)$$

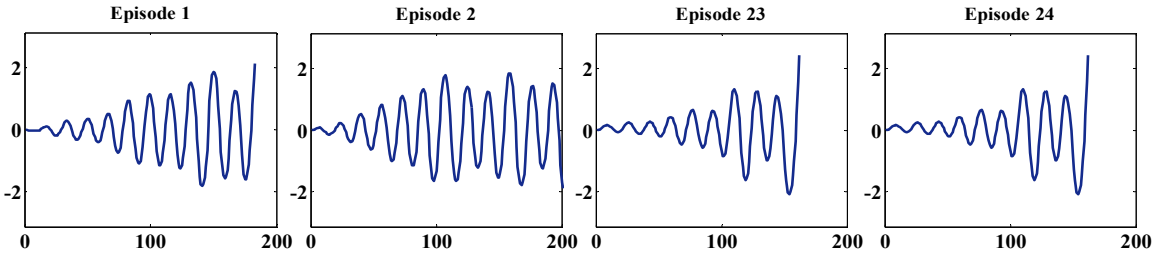
$$\dot{\phi}_1 = -m_2 l_1 l_{c2}{\dot{\theta}_2}^2 sin\theta_2 - 2m_2 l_1 l_{c2}\dot{\theta}_2\dot{\theta}_1 sin\theta_2 + (m_1 l_{c1} + m_2 l_1) g cos(\theta_1 - \pi/2) + \dot{\phi}_2 \quad (۳۵)$$

$$\dot{\phi}_2 = m_2 l_{c2} g cos(\theta_1 + \theta_2 - \pi/2) \quad (۳۶)$$

شایان ذکر است معادلات فوق تنها جهت شبیه‌سازی سیستم مورد استفاده قرار گرفته‌اند و از دید عامل (بخش کنترل) دینامیک سیستم ناشناخته می‌باشد.



شکل ۹: تغییرات زاویه‌ای لینک اول (θ_1) در روش NSL



شکل ۱۰: تغییرات زاویه‌ای لینک اول (θ_1) در روش [۲۰]

۵-۲-۲- جزئیات آموزش

ورودی‌های شبکه عصبی در این مسأله در روش NSL متشکل از زوایای دو لینک و مشتقات آنها باضافه‌ی تک تک عمل‌ها در آن حالت می‌باشد $x = [s,a]^T = [\theta_1, \theta_2, \dot{\theta}_1, \dot{\theta}_2, a]^T$. در این مسأله $3 \times 3 \times 3 \times 3 \times 3$ تابع پایه‌ی حالت و عمل برای تقریب تابع ارزش عمل در روش NSL استفاده شده است در روش [۲۰] نیز برای هر متغیر پیوسته‌ی حالت ورودی ۳ تابع پایه حالت گوسی در نظر گرفته شده است. پارامترهای سیگنال تقویتی، نرخ آموزش، ضریب دما مانند مثال قبل است. برای مشاهده تأثیر کنترلِ اعمالی، کنترلگر و مکانیسم آموزش در هر ۰.۲ ثانیه تحریک می‌شوند. تعریف کلیه پارامترهای استفاده شده و همچنین مقادیری که در شبیه سازی برای آنها در نظر گرفته شده در جدول (۲) آمده است. برای مقایسه دو الگوریتم تغییرات زوایای دو لینک در یک اجرا که شامل حداکثر ۵۰ رویداد با شروع از نقطه $(\theta_1,\theta_2,\dot{\theta}_1,\dot{\theta}_2) = (0,0,0,0)$ می‌باشد بررسی شده است. حداکثر تعداد قدم ها در یک رویداد در بخش آموزش ۲۰۰ لحاظ شده است. شکل‌های (۹) و (۱۰) تغییرات زاویه‌ای لینک اول (θ_1) و شکل‌های (۱۱) و (۱۲) تغییرات زاویه‌ای لینک دوم را در هر دو روش برای حالتی که $\eta_0 = 0.2$، $T_0 = 0.1$ و $\lambda = 0.9$ نشان می‌دهند.

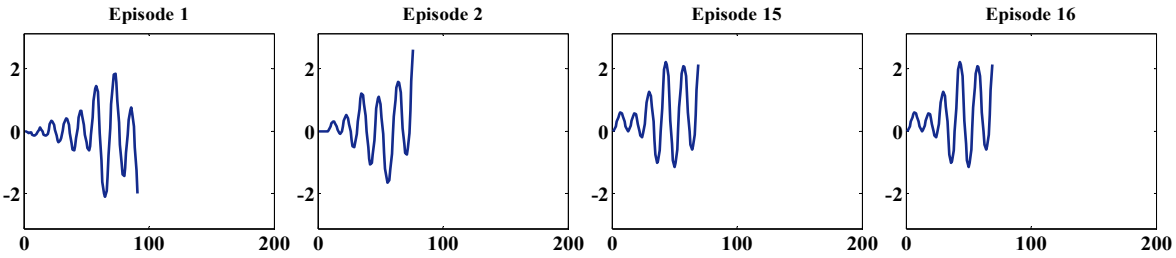
۵-۲-۳- نتایج شبیه سازی

همانگونه که از شکل‌ها مشخص است روش NSL هم از جهت سرعت همگرایی و هم از نظر سیاست نهایی که بدان همگرا شده است به مراتب بهتر از روش ارائه شده در [۲۰] عمل کرده است. در روش NSL به طور متوسط الگوریتم پس از ۱۵ رویداد همگرا شده است در حالی که این معیار در روش [۲۰] در حدود ۲۳ رویداد می‌باشد همچنین سیاست نهایی که در نهایت روش NSL بدان همگرا شده است به مراتب بهتر بوده و تعداد قدم‌های زمانی بسیار کمتری در رسیدن به هدف انجام شده است.

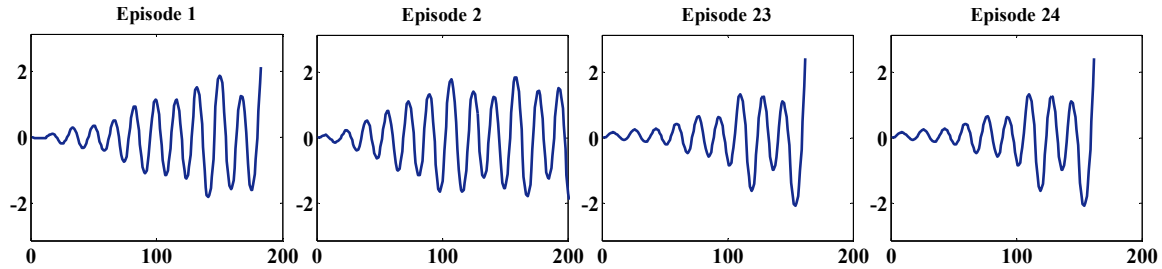
۶- نتیجه گیری و کار های آینده

در این مقاله یک الگوریتم جدید یادگیری تقویتی که از ترکیب روش‌های یادگیری تقویتی با شبکه‌های عصبی حاصل شده و در واقع تعمیمی از روش گسسته سارسا است معرفی شد. الگوریتم مذکور که NSLنامیده شد از لحاظ معماری دارای ساختار نقاد – تنها بوده و به منظور تنظیم وزن‌های یک شبکه عصبی به صورت بر خط به کار می رود. شبکه عصبی مورد نظر یک شبکه RBF است که به عنوان تقریب زننده‌ی تابع ارزش عمل به کار گرفته شد. از نتایج شبیه سازی انجام شده در مورد مسائل خودرو در کوهستان و آکروبات می‌توان نتیجه گرفت که به کارگیری معماری نقاد-تنها به خاطر پتانسیل کاوش بالاتر شانس رسیدن به جواب بهینه را زیادتر می‌نماید. هر چند نتایج شبیه سازی در مقایسه روش NSL با روش ارائه شده در مقاله [۲۰] در مورد مسأله خودرو در کوهستان به هم نزدیک بود. نتایج مربوط به مسأله آکروبات که مسأله‌ای بسیار سخت‌تر از مسأله خودرو در کوهستان است کارآیی بالاتر روش ارائه شده توسط ما را مشخص نمود.

با استفاده از نتایج مربوط به وجود نقاط ثابت برای روش تکرار تقریب ارزش عمل و همچنین تعریف تابع پایه حالت- عمل در الگوریتم NSL به نحوی که شرایط سارسای خطی بیان شده را ارضا نماید، وجود نقاط ایستای



شکل ۱۱: تغییرات زاویه‌ای لینک دوم (θ_2) در روش NSL



شکل ۱۲: تغییرات زاویه‌ای لینک دوم (θ_2) در روش [۲۰]

منطبق بر نقاط ثابت روش تکرار تقریب ارزش عمل برای NSL بعنوان اولین کار تحلیلی در این زمینه اثبات شد.

توجه شود که اثبات وجود نقاط ایستا لزوماً همگرایی روش را نتیجه نمی‌دهد، بلکه این یک شرط لازم برای همگرایی است. به طور کلی این نتیجه از چند جهت مهم می‌باشد: ۱- بر طبق تحقیقات و جستجوهایی که ما انجام دادیم، نتایج نظری ارائه شده برای NSL در این مقاله اولین نتایج مثبت ریاضی برای الگوریتم‌های NRL با معماری نقاد-تنها است، وقتی که سیاست در هر قدم زمانی به روز رسانی می شود، می‌باشد. ۲- NSL یک پیاده سازی سارسای خطی با استفاده از شبکه‌های عصبی است که در این پیاده سازی ما با ارائه یک ساختار تقریب زننده عصبی جدید توانستیم راهکاری برای رفع چالش، تعریف توابع پایه حالت- عمل ارائه دهیم. ۳- اثبات وجود نقاط ایستای منطبق بر نقاط ثابت روش تکرار تقریب ارزش عمل حاکی از امید بالا برای دست یافتن به راندمان مناسب می‌باشد. ۴- در الگوریتم NSL سیاست انتخاب عمل در هر قدم زمانی به روز رسانی می‌گردد که منجر به سرعت بالای آموزش در این روش می‌شود.

در آینده تلاش خواهیم نمود تا الگوریتمی عمل پیوسته بر مبنای ایده‌ی ارائه شده برای حل مسائلی که در آنها پیوستگی عمل مزیت بارزی نسبت به عمل گسسته دارد ارائه نماییم.

مراجع

[1] Gullapallia, V., 1991, “A comparison of supervised and reinforcement learning methods on a reinforcement learning task”, Proceedings of IEEE International Conference on Intelligent control, USA, pp. 394-399.

[2] Sutton, Richard S., and Barto, Andrew.G. “Reinforcement Learning: An Introduction”, Cambridge, MA: MIT Press, 1998.

[3] Sutton, R.S., Barto, A.G., andWilliams, R.J., 1992, “Reinforcement learning is direct adaptive optimalcontrol”, IEEE control systems magazine, pp.19-22.

[4] Wiering, M.A., 2004, “Convergence and divergence in standard and averaging reinforcement learning”, Proceedings of European Conference on Machine Learning, Italy, pp.477 – 488.

[5] Tsitsiklis, J. N., and Van Roy, B., 1997,“An analysis of temporal-difference learning with function approximation”, IEEE Transactions on AutomaticControl, Vol. 42, No. 5, pp. 674–690.

[6] Barto, A.G., Sutton, R.S., and Anderson, C.W., 1983, “Neuronlike adaptive elements that can solve difficult learning control problems”, IEEE Transactionson Systems, Man, and Cybernetics, Vol. 13, No.5.

[7] Potocnil, P., and Grabec, I., 2000, “Adaptive self-tuning neurocontrol”, Mathematics and Computers inSimulation, Vol.51, pp.201-207.

[8] Haykin, S., 1994, “Neural Networks, A Comprehensive Foundation”, NewYork: Macmillan.

[9] Guili, C., Wang, M., Hung, Z.J., andZhang, Z.F., 2009, “An actor-critic reinforcement learning algorithm based on adaptive RBF network”, Proceedings of the Eighth International Conference on Machine Learning and Cybernetics, pp.984-988.

[10] Si, J., and Wang, Y.T., 2001, “Online learning control by association and reinforcement”, IEEE Transactions on Neural Networks, Vol. 12, No.2, pp. 264-276.

[11] Konda, V.R., and Tsitsiklis, J.N., 2003,“On Actor-Critic Algorithms”, SIAM Journal on Control and Optimization, 42(4):1143–1166.

[12] Potocnil, P., and Grabec, I., 2000,"Adaptive self-tuning neurocontrol", Mathematics and Computers in Simulation, Vol.51, pp.201-207.

[13] Hwang, K.S., Tan, S.W., and Tsai, M. C., 2003, "Reinforcement Learning to Adaptive Control of Nonlinear Systems", IEEE Transactions on Systems, Man, and Cybernetics-Part B, Vol.33, No.3, pp.514-521.

[14] Bhatnagar, S.,Sutton, R.S., Ghavamzadeh, M.,and Lee, M., 2008,"Incremental natural actor-critic algorithms", Advances in Neural Information Processing Systems, MIT Press, Cambridge, MA, pp. 105–112.

[15] Hwang, K.S., Tan, S.W., and Tsai, M. C., 2003, "Reinforcement Learning to Adaptive Control of NonlinearSystems",IEEE Transactions on Systems, Man, and Cybernetics-Part B, Vol.33, No.3, pp.514-521.

[16] Derhami, V., Majd, V.J.,and Nili Ahmadabadi, M., 2008, "Fuzzy Sarsa learning and the proof of existence of its stationary points",Asian journal of control, John Wiley InterScience, Vol. 10, No. 5, pp.535-549.

[17] Jouffe, L., 1998, "Fuzzy inference system learning by reinforcement methods", IEEE Transactions on Systems, Man and Cybernetics-part C, Vol. 28, No. 3, pp. 338–355.

[18] Sun. W., Wang, X., and Yuhu, C., 2008, "Reinforcement Learning Method for Continuous State Space Based on Dynamic Neural Network", proceedings of the 7th World Congress on Intelligent Control and Automation, June 25 - 27, Chongqing, China.

[19] Cetina, V.U., 2008, "Multilayer Perceptrons with Radial BasisFunctions as Value Functions inReinforcement Learning", Eurpean symposium on Artificial Neural Networks- Advances in Computational Intelligence and Learning, Bruges, Belgium.

[20] FRÄMLING, K.,2008, "Light-weight reinforcement learning with function approximation for real-life control tasks", In: Proceedings of 5th International Conference on Informatics in Control, Automation and Robotics (ICINCO) , Funchal, Madeira, Portugal, pp. 127-134.

[21] De Farsis, D.P., and Van Roy, B.,2000,"On the existence of fixed points for approximate value iteration and temporal-difference learning", Journal of Optimal Theory and Appllication, Vol. 105, No. 3, pp. 25-36.

[22] Gordon, G. J., 2000, "Reinforcement Learning with function approximation converges to a region", Proc. 8th Int. Conf. Neural Inf. Process. Syst., Colorado, pp. 1040-1046.

[23] Perkins, T.J., and Precup, D., 2000, "A convergent form of approximate policy iteration", Proc. 9th Int. Conf.Neural Inf. Process. Syst., Singapore, pp. 1595–1602.

[24] Vamvoudakis, K. G., and Lewis, F. L., 2010, "Online actor-critic algorithm to solve the continuous-time infinite horizon optimal control problem", Automatica, vol. 46, pp. 878-888.

[25] Vrabie, D., and Lewis, F. L., 2008, "Adaptive optimal control algorithm for continuous-time nonlinear systems based on policy iteration", IEEE Proc. CDC08, pp. 73-79.

[26] D. Vrabie, Pastravanu, O., Abu-Khalaf, M., and Lewis, F.L., 2009, "Adaptive optimal control for continuous-time linear systems based on policy iteration", Automatica, vol. 45, no. 2, pp. 477-484.

[27] Al-Tamimi, A. Lewis, F.L., 2008, "Discrete-Time Nonlinear HJB Solution Using Approximate Dynamic Programming: Convergence Proof", IEEE Transactions on Systems, Man, and Cybernetics-Part B: Cybernetics, vol. 38, no. 4.

[28] Kalyanasundaram, S., Chong, E. K. P., and Shroff, N. B., 2004, "Markov decision processes with uncertain transition rates: sensitivity and max hyphen min control", Asian J. Control, Vol. 6, No. 2, pp.253–269.

[29] Kaelbling, L. P., Littman, M. L., and Moore, A. W., 1996, "Reinforcement learning: a survey", Journal of Artificial Intelligence Research, No. 4, pp. 237– 285.

[30] Watkins, C., andDayan, P., 1992, "Q-Learning", MachineLearning, Vol. 8, pp. 279–292.

[31] Singh, S., Jaakkola, T., Littman, M. L., and Szepesvari, C., 2000, "Convergence results for single-step on-policy reinforcement learning algorithms", Machine Learning, Vol. 39, pp. 287-308.

[32] Tsitsiklis, J. N., 1994, "Asynchronous stochastic approximation and Q-learning", Machine Learning, Vol. 16, pp.85–202.

[33] Baird, L. C., 1995, "Residual algorithms: Reinforcement learning with function approximation", Proceedings of 12th International Conference on Machine Learning, California, pp. 30–37.

[34] E. Alpaydin., 2004, "Introduction to Machine Learning", the MIT Press, Cambridge, Massachusetts.

[35] Derhami, V., Majd, V.J., andNili Ahmadabadi, M., 2010, "Exploration and Exploitation Balance Management in Fuzzy Reinforcement Learning", Fuzzy Sets and Systems, Elsevier, Vol. 161, No. 4, pp. 578-595.

[36] Second Annual Reinforcement Learning Competition http://rl-competition.org.

[37] Spong, M. W., 1995, "The swing up control problem for the acrobat", IEEE Control Systems Magazine, Vol. 15, pp. 49-55.

[38] Xin Xu, Dewen Hu, and Xicheng Lu, 2007, "Kernel-Based Least Squares Policy Iteration for Reinforcement Learning", IEEE Transactions on Neural Networks, Vol. 18, No. 4, pp. 973-992.