یک معیار مبتنی بر واریانس برای ارزیابی یادگیری آتاماتای یادگیر در حل مسایل بهینه‌سازی گراف تصادفی

محمدرضا ملاخلیلی میبدی و محمدرضا میبدی

چکیده: در این مقاله به بررسی یک معیار جدید مقایسه‌ای برای تولید پاسخ محیط در حل مسایل بهینه‌سازی روی گراف‌های تصادفی به عنوان مدلی از شبکه‌های کامپیوتری توسط شبکه‌ای از آتاماتاهای یادگیر می‌پردازیم. این روش جدید به دلیل لحاظ‌کردن تقریبی از واریانس پاسخ‌های تولیدشده توسط شبکه آتاماتاهای یادگیر، قادر به انطباق بیشتری با محیط بوده و در نتیجه پاسخ‌های مناسب‌تری به اقدام‌های انجام‌شده توسط آتاماتاها در شبکه‌ای از آتاماتاهای یادگیر می‌دهد. روش جدید از طریق واردکردن یک مقدار نویز محاسبه‌شده، از ایستایی فرایند یادگیری و گیرافتادن آن در نقاط کمینه محلی جلوگیری کرده و باعث تسریع در فرایند یادگیری می‌شود. به کمک شبیه‌سازی‌ها نشان می‌دهیم این روش جدید در مقایسه با روش‌های فعلی که تا کنون مورد استفاده بوده است، هم به لحاظ سرعت همگرایی به جواب بهینه و هم به لحاظ قابلیت گریز از اثر واریانس وزن یال‌های گراف تصادفی- که باعث میل جواب نهایی به سمت کوچک‌ترین مقدار و نه مقدار میانگین می‌شود- عملکرد بهتری دارد.

کلیدواژه: **شبکه آتاماتاهای یادگیر، واریانس، همگرایی، درخت پوشای کمینه تصادفی، کوتاه‌ترین مسیر تصادفی.**

۱- مقدمه

آتاماتای یادگیر، ماشینی با حالات متناهی است که در تعامل با یک محیط تصادفی، سعی می‌کند که بهترین اقدام قابل انجام را مشخص کند. این مفهوم نخستین بار توسط Tsetlin و به منظور مدل‌سازی یادگیری بیولوژیکی ارائه شد و تاکنون کاربردهای متفاوتی را در حوزه‌های مختلف نظیر مسیریابی در شبکه‌های تلفن و داده‌ای [۱] تا [۴]، تشخیص الگو [۵]، جداسازی گراف [۶] و برنامه‌ریزی مسیر [۷] پیدا کرده است (دیگر کاربردهای آتاماتای یادگیر در [۸] تا [۱۱] ارائه شده‌اند). فرایند یادگیری را در آتاماتا به صورت خلاصه می‌توان به این صورت توصیف کرد: آتاماتا دارای مجموعه‌ای از اقدام‌ها است و در تعامل با یک محیط تصادفی قرار دارد. در هر زمان آتاماتا بایستی یکی از اقدام‌های خودش را انتخاب کند. زمانی که یک اقدام توسط آتاماتا انتخاب می‌شود، با احتمال مشخصی ممکن است توسط محیط پاداش داده شده یا جریمه شود. آتاماتای یادگیر، آتاماتایی است که اقدام بهینه را انتخاب می‌کند، اقدامی که کمترین احتمال جریمه‌شدن را داراست. بررسی رفتار آتاماتاها در محیط‌های ایستا، تاکنون بارها و به اشکال مختلف مورد بررسی قرار گرفته است.

یکی از کاربردهای آتاماتای یادگیر در نمونه‌گیری هدایت‌شده[1] برای حل مسایل بهینه‌سازی روی گراف‌های وزن‌داری است که در آن یال‌ها دارای وزن با توزیع احتمال ناشناخته است. در این مسایل، از آتاماتای یادگیر برای نمونه‌گیری از یال‌هایی که نقش مؤثرتری در مسأله مورد نظر دارند استفاده می‌شود. نشان داده شده که روش نمونه‌گیری مبتنی بر آتاماتای یادگیر در مقایسه با روش نمونه‌گیری استاندارد، می‌تواند با تعداد نمونه‌گیری کمتر، جواب بهینه را پیدا کند. اما روش‌های پیشنهادی از مشکل کندی همگرایی و وابستگی شدید به نرخ یادگیری رنج می‌برند. مطالعات متعددی برای حل این مشکل صورت گرفته که به کارگیری آنها در عمل با چالش‌هایی مواجه است.

در این مقاله یک روش کاملاً جدید و کارامد برای حل این مشکل و به خصوص در حل مسایل بهینه‌سازی روی گراف‌های تصادفی به عنوان مدلی از شبکه‌های کامپیوتری ارائه شده است. ادامه این مقاله به این صورت سازماندهی شده است: در بخش ۲ آتاماتای یادگیر معرفی شده و بخش ۳ به بررسی مدل گراف تصادفی اختصاص یافته است. در بخش ۴ به بررسی برخی از کارهایی که در این حوزه صورت گرفته اختصاص یافته است. در بخش ۵ به بررسی روش پیشنهادی می‌پردازیم، بخش ۶ به مقایسه معیار جدید پیشنهادی اختصاص یافته و در بخش پایانی نتایج مقاله ارائه شده است.

۲- آتاماتای یادگیر

آتاماتای یادگیر تصادفی یک واحد تصمیم‌گیرنده تطبیقی است که فرایند یادگیری در آن از طریق تعاملش با محیط صورت می‌گیرد. آتاماتای یادگیر، مجموعه‌ای از اقدام‌های قابل انجام دارد. این اقدام‌ها به تصادف و بر اساس یک بردار توزیع احتمال، انتخاب شده و به عنوان ورودی به محیط اعمال می‌شوند. محیط، اقدام انجام‌شده را به کمک یک سیگنال تقویتی بازخوردی، مورد ارزیابی قرار می‌دهد. آتاماتای یادگیر بر اساس سیگنال بازخوردی حاصل، بردار توزیع احتمال انتخاب اقدام‌ها را به روز رسانی می‌کند. هدف آتاماتا، پیداکردن اقدام بهینه در میان مجموعه اقدام‌های قابل انجام است، اقدامی که بیشترین پاداش را از محیط دریافت کند.

ارتباط آتاماتای تصادفی با محیط در شکل ۱ نشان داده شده است. از این مجموعه به همراه الگوریتم یادگیری تحت عنوان آتاماتای یادگیر تصادفی نام برده می‌شود. به این ترتیب آتاماتای یادگیر تصادفی را می‌توان با چهارتایی (۱) تعریف کرد

$$SLA \equiv \{\alpha, \beta, p, T, c\} \qquad (۱)$$

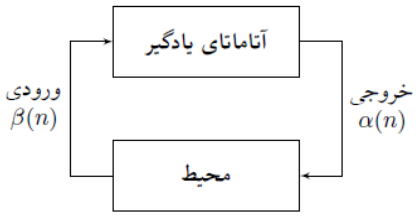
که $\alpha \equiv \{\alpha_1, \alpha_2, \ldots, \alpha_r\}$ مجموعه اقدام‌های آتاماتا/مجموعه ورودی‌های محیط، $\beta \equiv \{\beta_1, \beta_2, \ldots, \beta_m\}$ مجموعه ورودی‌های آتاماتا/ مجموعه

این مقاله در تاریخ ۸ اردیبهشت ماه ۱۳۹۵ دریافت و در تاریخ ۹ آذر ماه ۱۳۹۵ بازنگری شد.

محمدرضا ملاخلیلی میبدی، دانشگاه آزاد اسلامی، واحد میبد، گروه کامپیوتر، میبد، (mail: mollakhalili@maybodiau.ac.ir).

محمدرضا میبدی، آزمایشگاه محاسبات نرم، دانشکده مهندسی کامپیوتر و فناوری اطلاعات، دانشگاه صنعتی امیرکبیر، تهران (email: mmeybodi@aut.ac.ir).

1. Guided Sampling



شکل ۱: آتاماتای یادگیر و محیط.

خروجی‌های محیط، $p \equiv \{p_1, p_2, \ldots, p_r\}$ بردار احتمال اقدام‌های آتاماتا، $T \equiv \varphi \to \alpha$ الگوریتم یادگیری و $c \equiv \{c_1, c_2, \ldots, c_r\}$ مجموعه احتمالات جریمه و معرف محیط است.

الگوریتم یادگیری یک رابطه بازگشتی است که برای انجام تغییرات و به روز رسانی در بردار احتمال اقدام‌های آتاماتا در یک آتاماتای یادگیر تصادفی با ساختار متغیر مورد استفاده قرار می‌گیرد. فرض کنید یک آتاماتای یادگیر تصادفی ساختار متغیر در زمان k از میان مجموعه اقدام‌های α عمل $\alpha_i(k)$ را انتخاب کرده باشد. همچنین فرض کنید بردار احتمال انتخاب اقدام‌های آتاماتا را با $p(k)$ نمایش داده‌ایم. اگر a و b پارامترهایی باشند که به ترتیب میزان افزایش یا کاهش احتمالات اقدام‌ها را مشخص می‌کنند و r تعداد اقدام‌های قابل انجام توسط آتاماتای یادگیر باشد، بردار $p(k)$ توسط الگوریتم یادگیری خطی ارائه‌شده در روابط زیر به روز رسانی می‌شود. مقدار a را پارامتر پاداش و b را پارامتر جریمه می‌نامند

$$p_j(k+1) = \begin{cases} (1-a)p_j(k) + a & , \ j = i \\ (1-a)p_j(k) & , \ \forall j \neq i \end{cases} \qquad (۲)$$

$$p_j(k+1) = \begin{cases} (1-b)p_j(k) & , \ j = i \\ (1-b)p_j(k) + \dfrac{b}{r-1} & , \ \forall j \neq i \end{cases} \qquad (۳)$$

رابطه (۲) زمانی مورد استفاده قرار می‌گیرد که عمل $\alpha_i(k)$ منجر به دریافت پاداش از محیط شده باشد و (۳) زمانی مورد استفاده قرار می‌گیرد که این عمل به دریافت جریمه از محیط منجر شود. در (۲) و (۳) اگر $a = b$ باشد روابط یادگیری خطی را الگوریتم L_{R-P}، اگر $a \gg b$ باشد آن را $L_{R-\varepsilon P}$ و اگر $b = 0$ باشد آن را L_{R-I} می‌نامند.

۲-۱ آتاماتای یادگیر با مجموعه اقدام‌های متغیر

یک آتاماتای یادگیر با مجموعه اقدام متغیر، آتاماتایی است که در آن تعداد اقدام‌های موجود هر آتاماتا در طول زمان تغییر می‌کند [۱۲]. فرض کنید مجموعه اقدام‌های آتاماتا را با $\alpha \equiv \{\alpha_1, \alpha_2, \ldots, \alpha_r\}$ و مجموعه اقدام‌های قابل انجام توسط آتاماتا در زمان n را با $A(k) \subseteq \alpha$ نشان دهیم

$$A(k) \in \{A_1, A_2, \ldots, A_m\} \ , \ m = 2^{n-1} \qquad (۴)$$

انتخاب زیرمجموعه $A(k)$ از میان مجموعه اقدام‌های قابل انجام توسط آتاماتا، غالباً توسط شرایط بیرونی تحمیل می‌شود. می‌توان این گونه فرض کرد که این زیرمجموعه از اقدام‌های آتاماتا را که از این به بعد مجموعه اقدام‌های فعال آتاماتا نامیده می‌شود، توسط یک عامل بیرونی و با توزیع احتمال

$$\psi_i(k) = \{\psi_1(k), \psi_2(k), \ldots, \psi_m(k)\} \qquad (۵)$$

انتخاب می‌شود. اگر تعریف کنیم

$$\hat{p}_i(k) = \text{Prob}[\alpha(k) = \alpha_i \,|\, A(k)] \ , \ \alpha_i \in A(k) \qquad (۶)$$

$\hat{p}_i(k)$) احتمال انتخاب عمل از میان مجموعه اقدام‌های فعال به شرط آن که مجموعه اقدام‌های فعال آتاماتا برابر با $A(k)$ باشد) در این صورت

$$\hat{p}_i(k) = \frac{p_i(k)}{K(k)} \qquad (۷)$$

در (۷)، $K(k)$ مجموع احتمالات تمام اقدام‌های فعال (اقدام‌های عضو $A(k)$) است و داریم

$$K(k) = \sum_{\alpha_i \in A(k)} p_i(k) \qquad (۸)$$

که در آن

$$p_i(k) = \text{Prob}[\alpha(k) = \alpha_i] \qquad (۹)$$

احتمال انتخاب اقدام α_i در میان مجموعه تمام اقدام‌های آتاماتا (اعم از این که فعال باشد یا نباشد) است. بدین ترتیب نحوه انتخاب اقدام توسط آتاماتا یا مجموعه اقدام متغیر بدین صورت است که فرض کنید آتاماتا مجموعه اقدام‌های فعال $A(k)$ را داشته باشد. ضریب نرمال‌کننده $K(k) = \sum_{\alpha_i \in A(k)} p_i(k)$ برای این مجموعه از اقدام‌های فعال محاسبه شده و بردار احتمال انتخاب اقدام‌های آتاماتا مطابق با رابطه بالا نرمال‌سازی می‌شود (به گونه‌ای‌که مجموع احتمال انتخاب اقدام‌های فعال همچنان ۱ باشد). پس از این آتاماتا بر اساس بردار جدید $\hat{p}_i(k)$ یکی از اقدام‌های (فعال) خود را انتخاب کرده و به محیط اعمال می‌کند. در مرحله به روز رسانی نیز بردار $\hat{p}_i(k)$ مطابق با الگوریتم یادگیری مورد استفاده توسط آتاماتا به روز رسانی می‌شود و سپس با استفاده مجدد از رابطه بالا بردار $p_i(k) = \hat{p}_i(k).K(k)$ به دست می‌آید.

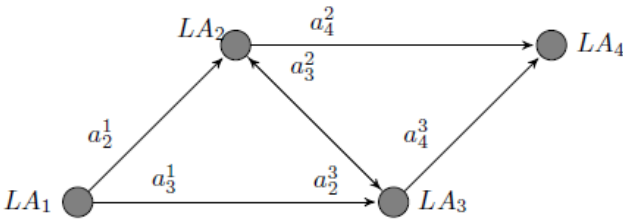
نشان داده شده است [۱۲] در صورتی که الگوریتم یادگیری مورد استفاده L_{R-I} باشد، این روش ویژگی‌های $\varepsilon-$optimality و absolute expediency را دارد.

۲-۲ آتاماتای یادگیر توزیع‌شده

آتاماتای یادگیر توزیع‌شده شبکه‌ای است از آتاماتاهای یادگیر که برای حل یک مسأله خاص با یکدیگر همکاری دارند. یک DLA در قالب یک گراف ارائه می‌شود و با چندتایی (V, E, T, v_0) تعریف می‌شود که در آن V مجموعه رئوس گراف، E مجموعه یال‌ها، T مجموعه‌ای از الگوریتم‌های یادگیر مورد استفاده توسط آتاماتاها در DLA و v_0 گره ریشه DLA است [۱۳].

در این شبکه از آتاماتاهای همکار در هر زمان تنها یک آتاماتا فعال است. تعداد اعمال قابل انجام توسط یک آتاماتا در DLA برابر است با تعداد آتاماتاهایی که به این آتاماتا متصل شده‌اند. انتخاب یک عمل توسط آتاماتا در این شبکه، باعث فعال‌شدن آتاماتای متصل‌شده به این آتاماتا و متناظر با این عمل می‌گردد. به عبارت معادل، انتخاب یک عمل توسط یک آتاماتا در این شبکه متناظر با فعال‌شدن یک آتاماتای دیگر در این شبکه است.

وجود یال (LA_i, LA_j) در این گراف بدان معناست که انتخاب عمل α_j^i توسط LA_i باعث فعال‌شدن LA_j می‌گردد. تعداد اعمال قابل انتخاب توسط LA_k برابر است با درجه خروجی این رأس در ساختار گراف DLA. بردار احتمالات مربوط به عمل‌های قابل انجام توسط آتاماتای LA_k به صورت $p^k = \{p_1^k, p_2^k, \ldots, p_{r_k}^k\}$ نمایش داده می‌شود و در این مجموعه عدد p_m^k نشان‌دهنده احتمال مربوط به عمل a_m^k است. انتخاب



شکل ۲: آتاماتای یادگیر توزیع‌شده.

عمل a_m^k توسط LA_k باعث فعال‌شدن LA_m می‌شود. r_k تعداد اعمال قابل انجام توسط آتاماتای LA_k را نشان می‌دهد. نمونه‌ای از یک DLA و نمادگذاری‌های مرتبط با آن در شکل ۲ نشان داده شده است.

۳- گراف تصادفی و مدل‌سازی شبکه‌های کامپیوتری

تعاریف متعددی از گراف تصادفی در متون یافت می‌شود و منظور ما از گراف تصادفی در این مقاله، یک گراف وزن‌دار جهت‌دار یا بدون جهت است که وزن یال‌ها در آن دارای یک توزیع احتمال ناشناخته است. به بیان فرمال، گراف تصادفی $G=(V,E,Q)$ با سه‌تایی G تعریف می‌شود که $V=\{v_1,v_2,\ldots,v_n\}$ مجموعه‌ای از رئوس، $E\subseteq V\times V$ مجموعه یال‌ها و ماتریس $Q_{n\times n}$ توزیع احتمال وزن یال‌ها است. برای هر یال $e_{(i,j)}\in E$ وزن آن، w_{ij} یک متغیر تصادفی مثبت فرض می‌شود که q_{ij} تابع چگالی احتمال آن است [۱۴] و در این مقاله q_{ij} ناشناخته فرض می‌شود.

مدل گراف تصادفی با این تعریف، ابزار مناسبی برای مدل‌سازی بسیاری از شبکه‌ها و از جمله شبکه‌های کامپیوتری است. برای حل مسایل بهینه‌سازی روی گراف‌های تصادفی، به کمک شبکه‌ای از آتاماتاها تاکنون روش‌های مختلفی ارائه شده است [۱۵] تا [۱۷].

علاوه بر این، استفاده از آتاماتای یادگیر توزیع‌شده در حل مسایل دیگری نیز مورد استفاده قرار گرفته است از جمله مدل‌سازی و پیش‌بینی حرکت کاربران در وب [۱۸] و [۱۹]، داده‌کاوی وب [۲۰] و رتبه‌بندی صفحات وب [۲۱]. آتاماتای یادگیر توزیع‌شده توسعه‌یافته [۲۲]، نسخه‌ای از آتاماتای یادگیر توزیع شده است که برخی کاستی‌های آن مرتفع شده و برای حل مسایل مختلفی مورد استفاده قرار گرفته است [۱۷]، [۲۳] و [۲۴].

۳-۱ مسأله درخت پوشای کمینه تصادفی

در گراف تصادفی فاقد جهت G زیرگراف $T=(V,E',Q)$ درخت پوشا نامیده می‌شود اگر:

الف) T یک زیرگراف متصل از G و شامل تمام رئوس G باشد.

ب) مجموعه یال‌های زیرگراف T زیرمجموعه‌ای از یال‌های گراف G بوده و تعداد آنها $|E'|=|V|-1=n-1$ باشد.

فرض کنید گراف تصادفی G دقیقاً r درخت پوشای تصادفی مطابق با شرایط گفته‌شده داشته باشد که آنها را با $T_G=\{\tau_1,\tau_2,\ldots,\tau_r\}$ نمایش می‌دهیم. از آنجا که وزن یال‌ها در گراف تصادفی یک متغیر تصادفی است، وزن درخت پوشای تصادفی نیز یک متغیر تصادفی خواهد بود. وزن درخت τ_i را با $W(\tau_i)$ و مقدار امید ریاضی (متوسط) وزن درخت τ_i را با $\bar{W}(\tau_i)$ نشان می‌دهیم. از میان این r درخت پوشا، درخت پوشای کمینه تصادفی، درختی است که کمترین مقدار متوسط وزن را داشته باشد [۱۶]. به عبارت دیگر، درخت پوشای کمینه تصادفی درخت τ^* است که

$$\bar{W}(\tau^*)=\min_{\forall \tau_i\in T_G}\{\bar{W}(\tau_i)\} \qquad (۱۰)$$

نسخه قطعی مسأله یافتن درخت پوشای کمینه (یعنی حالتی که یال‌های گراف دارای وزن قطعی باشند) راه حل‌های متفاوتی با زمان چندجمله‌ای دارد. یکی از این راه حل‌ها به راه حل کروسکال موسوم است [۲۵]. در این راه حل به شکل حریصانه، یالی با کمترین وزن در هر مرحله به جواب اضافه می‌شود به نحوی که موجب بروز پیدایش حلقه در زیرگراف نشود. این کار تا رسیدن تعداد یال‌ها به اندازه یک واحد کمتر از تعداد گره‌های گراف ادامه پیدا می‌کند. علاوه بر این، الگوریتم‌های متفاوتی برای حل مسأله MST در گراف‌های قطعی به صورت تقریبی و توزیع‌شده نیز ارائه شده است [۲۶] تا [۲۸]. نویسنده در [۲۵] فهرستی از مقالاتی را که به صورت تقریبی و توزیع‌شده به حل مسأله پرداخته‌اند ارائه نموده است.

۳-۲ مسأله کوتاه‌ترین مسیر تصادفی

گراف تصادفی جهت‌دار $G=(V,E,X)$ و گره‌های v_s و v_d را در نظر بگیرید. مسیر π_i از مبدأ v_s به مقصد v_d در گراف G را به عنوان ترتیب $\pi_i=\{i_1,i_2,\ldots,i_{m_i}\}$ تعریف می‌کنند که در آن $i_1=v_s$ و $i_{m_i}=v_d$ و

$$(i_j,i_{j+1})\in E \ , \ \forall 1\le j<m_i \qquad (۱۱)$$

متوسط هزینه مسیر π_i که آن را با C_{π_i} نمایش می‌دهیم با استفاده از (۱۲) تعریف می‌شود

$$C_{\pi_i}=\sum_{j=1}^{m_i-1}C_{i_ji_{j+1}} \qquad (۱۲)$$

در (۱۲) C_{ij} متوسط هزینه یال $e_{ij}\in E$ را نشان می‌دهد. فرض کنید $\Pi=\{\pi_1,\pi_2,\ldots,\pi_r\}$ نشان‌دهنده r مسیر مجزای بین v_s و v_d باشد. کوتاه‌ترین مسیر تصادفی، مسیری با کمترین متوسط هزینه تعریف می‌شود که آن را با π^* نشان می‌دهیم و

$$\pi^*=\min_{\pi_i\in\Pi}C_{\pi_i} \qquad (۱۳)$$

٤- مروری بر برخی از کارهای انجام‌شده در این زمینه

برای حل مسایل بهینه‌سازی روی گراف‌های تصادفی، نیازمند معیاری برای مقایسه پاسخ تولیدشده در هر مرحله از اجرای الگوریتم توسط آتاماتا، به منظور تعیین جریمه یا پاداش آتاماتاهای همکار در تولید پاسخ هستیم. در [۲۹] نویسنده برای نخستین بار در حل مسأله کوتاه‌ترین مسیر تصادفی مفهوم آستانه پویا را برای سنجش عملکرد آتاماتاها در یک DLA و تولید پاسخ به اقدام‌های آتاماتا معرفی کرده و بر اساس قضیه حد مرکزی صورتی از مقدار آستانه را ارائه داده که بارها و بارها در مطالعات مشابه مورد استفاده قرار گرفته است. در مطالعات متعددی که برای حل مسایل توسط شبکه‌ای از آتاماتاها روی گراف‌های تصادفی انجام شده است، نشان داده شده که استفاده از این مقدار آستانه پویا، مستلزم استفاده از نرخ‌های کوچک یادگیری است و نرخ کوچک یادگیری باعث افزایش تعداد تکرارهای لازم برای همگرایی الگوریتم و افزایش مدت زمان اجرای آن می‌شود.

برای تسریع در روند همگرایی و کاهش مدت زمان اجرا در الگوریتم‌هایی که مبتنی بر شبکه‌ای از آتاماتاهای یادگیر هستند دو دسته راهکار وجود دارد. راهکار اول آن است که از طریق ایجاد یک شوک، احتمال همگرایی الگوریتم را افزایش دهند. ایده کلی این است که با تزریق یک نویز در جواب پیداشده توسط سیستم یادگیر، زمینه خروج آن را از نقاط کمینه محلی فراهم می‌کنند. راهکار دوم بر مبنای نحوه تقسیم

پاداش میان آتاماتاهای یادگیر است. ایده اساسی آن است که میزان پاداشی که به آتاماتاها تخصیص داده می‌شود بایستی متناسب با میزان عملکردشان باشد.

در [۱۶] نویسنده الگوریتم‌هایی برای بهبود عملکرد الگوریتم یافتن درخت پوشای کمینه توسط آتاماتاهای یادگیر به کمک تغییر در نحوه تقسیم پاداش میان آتاماتاها ارائه داده است. این الگوریتم‌ها مبتنی بر شاخص‌های آماری توزیع وزن نمونه‌های گرفته‌شده توسط آتاماتاها است. این روش حداقل سه ایراد اساسی دارد:

- اولاً منطق آتاماتای یادگیر که سادگی محاسباتی آن است را با انجام محاسباتی نظیر میانگین، واریانس و انحراف معیار (که محاسبه آن مستلزم ضرب، تقسیم، توان و مجذور است) نمونه‌های گرفته‌شده توسط آتاماتای یادگیر، خدشه‌دار می‌کند.
- در عمل، به کارگیری آن به صورت آنلاین با چالش مواجه است. به عنوان مثال در شبکه‌های کامپیوتری، فرض این که یک مسیریاب میانی که هدایت یک بسته را بر عهده دارد، از زمان طی‌شدن بسته روی یک لینک مطلع می‌شود (معادل با این که یک آتاماتای یادگیر، وزن یال انتخاب‌شده را می‌داند) فرض درست و عملی نیست. در این مدل‌ها، تنها معیاری که برای مقایسه قابل استفاده است، مقدار تجمیع‌شده وزن یال‌های تشکیل‌دهنده زیرگراف جواب مسأله است [۳۰].
- انجام محاسباتی نظیر واریانس نمونه مستلزم حافظه‌ای برای ذخیره‌سازی نمونه‌ها است.

گرچه ممکن است بتوان برای ایرادهای ۱ و ۲ راهکارهایی ارائه داد (مثلاً استفاده از تخمین‌گرهای stochastic gradient برای سادگی محاسباتی و یا استفاده از روش‌های بازگشتی در محاسبه واریانس و میانگین برای عدم ذخیره‌سازی مقادیر) اما ایراد ۲ یک ایراد اساسی است که در نهایت به کارگیری این معیارها را در عمل با مشکل مواجه می‌کند.

در [۳۱] نویسنده، راهکاری برای تطبیق‌پذیری بیشتر نرخ یادگیری در محیط‌های پویا برای آتاماتای یادگیر ارائه داده است. راه حل ارائه‌شده، حدس رخداد تغییرات وسیع در محیط یادگیری از طریق برخی شاخص‌های آماری جواب‌های محیط به اقدام‌های انجام‌شده است.

۵- روش پیشنهادی

در الگوریتم‌های مبتنی بر یادگیری برای حل مسایل بهینه‌سازی در گراف‌های تصادفی، محیط می‌بایستی از وزن زیرگراف به دست آمده در هر مرحله به عنوان معیاری برای ارزیابی عملکرد سیستم یادگیر استفاده کند. از آنجا که وزن یال‌ها متغیرهایی تصادفی هستند واضح است که وزن زیرگراف تولیدشده توسط سیستم یادگیر نیز تصادفی خواهد بود. می‌توان دو نوع معیار برای سنجش معرفی کرد:

الف) سنجش نقطه‌ای: در این روش، محیط از یک مقدار آستانه پویا استفاده می‌کند. این مقدار، نوعی میانگین وزنی پویا است به این ترتیب که مقدار آن در هر دور یادگیری بر اساس (۱۴) تغییر می‌کند

$$W_{TH}(n) = \alpha W + (1-\alpha) W_{TH}(n-1) \qquad (14)$$

در (۱۴) مقدار وزن زیرگراف در مرحله nام با W و مقدار آستانه پویا با $W_{TH}(n)$ آمده و α یک مقدار عددی ثابت یا پویا است. رابطه (۱۴) در حقیقت نمونه‌ای ساده از رده‌ای از تخمین‌گرها موسوم به الگوریتم‌های recursive prediction error یا stochastic gradient است.

اگر $\alpha = 1/n$ فرض شود، (۱۴) دقیقاً مقدار متوسط مقادیر W را از ابتدا تا زمان n به عنوان مقدار آستانه پویا در نظر می‌گیرد. در روش سنجش نقطه‌ای این مقدار آستانه پویا به عنوان مرز قابل قبول بودن یا نبودن مورد استفاده قرار می‌گیرد. این معیار در [۲۹] معرفی شده و از آن پس در سایر مطالعات مربوط به بهینه‌سازی روی گراف‌های تصادفی مورد استفاده قرار گرفته است.

ب) روش سنجش بازه‌ای: در این روش می‌توان بر اساس مشخصه‌های آماری توزیع مقادیری که تاکنون برای W به دست آمده‌اند، یک بازه قابل قبول از حدودی که برای W به دست می‌آیند، به دست آورد. نمونه‌ای از این روش در [۳۲] مورد استفاده قرار گرفته است. به کارگیری روش [۳۲] علاوه بر نیاز محاسباتی بالا که به کارگیری آن در آتاماتای یادگیر را با چالش مواجه می‌کند به عنوان معیاری برای تعیین پاداش یا جریمه مجموعه اقدام‌ها نیز ناممکن است چرا که بر اساس قضایای نمونه‌گیری تنها قادر به تعیین یک بازه اطمینان از وزن یال‌های نمونه‌گیری شده است.

به کارگیری معیارهای نقطه‌ای غالباً ساده‌تر است اما لازم است تا تخمین درستی از نقطه معیار ارائه شود و این در حالی است که استفاده از معیارهای بازه‌ای دقیق‌تر است. ما در این مقاله به دنبال ارائه یک معیار نقطه‌ای هستیم که از مزایای معیارهای بازه‌ای برخوردار بوده و بتواند تخمین درستی از نقطه معیار ارائه دهد.

رابطه (۱۴) را مجدداً در نظر بگیرید. داریم

$$\begin{aligned} &W_{TH}(n) = \alpha W + (1-\alpha) W_{TH}(n-1) = \\ &W_{TH}(n-1) + \alpha (W - W_{TH}(n-1)) = \\ &W_{TH}(n-1) + \alpha \times Err \end{aligned} \qquad (15)$$

مقدار Err در (۱۵) نشان‌دهنده خطا است و خطا را می‌توان ناشی از ۲ عامل دانست: خطای ناشی از تخمین بد و خطای ناشی از محیط که این دو خطا را به ترتیب با E_e و E_r نشان می‌دهیم. رابطه (۱۵) را می‌توان به صورت (۱۶) نشان داد

$$W_{TH}(n) = W_{TH}(n-1) + \alpha \times E_r + \alpha \times E_e \qquad (16)$$

رابطه (۱۶) نشان می‌دهد که W_{TH} به مقدار واقعی‌اش همگرا می‌شود اما به اندازه $\alpha \times sdev(W)$ انحراف دارد. برای اندازه‌گیری تغییرات W، واریانس یک انتخاب معمول است (به دلیل خواص ریاضی آن) اما محاسبه آن مستلزم محاسبه توان دوم و مجذورگیری است که در عمل باعث پیچیدگی محاسباتی خواهد شد.

معیار دیگری که محاسبه آن ساده بوده و اندازه‌ای از تغییرات W به ما می‌دهد میانگین خطای پیش‌بینی یا انحراف میانگین است یعنی میانگین $|W - W_{TH}|$. از آنجا که

$$\begin{aligned} &mdev^2 = \\ &(\sum |W - W_{TH}|)^2 \geq \sum |W - W_{TH}|^2 = \delta^2 = sdev^2 \end{aligned} \qquad (17)$$

لذا $mdev$ از $sdev$ محافظه‌کارتر (بزرگ‌تر) است. غالباً رابطه ساده‌ای بین $mdev$ و $sdev$ وجود دارد. اگر خطاهای پیش‌بینی (یعنی $W - W_{TH}$) به شکل نرمال توزیع‌شده باشند $mdev = \sqrt{\pi/2} \times sdev = 1.25 \times sdev$ است. برای اکثر توزیع‌های معمول ضریب رسیدن از $sdev$ به $mdev$ در همین حدود است و بنابراین $mdev$ می‌تواند تخمین خوبی از $sdev$ به دست دهد، علاوه بر آن که محاسبه آن نیز ساده است.

اگر از یک تخمین‌گر stochastic gradient برای $mdev$ استفاده کنیم، خواهیم داشت

$$mdev(n) = mdev(n-1) + \beta(|W - W_{TH}| - mdev(n-1)) \qquad (18)$$

Input: Stochastic graph $G=(V,E,Q)$, Thresholds P_{POM}, K
Output: SMST $T=(V,X)$

1. Assign a LA to each node of Graph G
2. Let k be the stage number and initially set to 0
3. Let X, W_X denotes the constructed Spanning Tree and its weight
4. Let WTH_k be the dynamic threshold at stage k and initially set to 0
5. $X \leftarrow \phi$; $W_X \leftarrow 0$; $\alpha = 0.125$; $\beta = 0.25$;
6. **begin Algorithm**
7. **repeat**
8. Let P, A denote Passive and Active sets of LAs
9. $P \leftarrow V$, $A \leftarrow \phi$
10. **while** $|X| < |V| - 1$ and $P \neq \phi$ **do**
11. Let u be random node of graph G
12. $P \leftarrow P - \{u\}$, $A \leftarrow A + \{u\}$
13. e = select_action (u)
14. each automaton in A prunes its action-set for cycle avoidance
15. $X \leftarrow X + e$
16. $W_X \leftarrow W_X + W_e$
17. **end while**
18. $E = W_X - WTH_k$
19. $WTH_k = WTH_k + \alpha \times E$
20. $V = V + \beta(abs(Err) - V)$
21. **if** $|X| = |V| - 1$ **then**
22. **if** $W_X < (WTH_k + 4 \times V)/2$ **then**
23. reward the selected actions of LAs
24. **else**
25. penalize the selected actions of LAs
26. **end if**
27. **else**
28. Do nothing
29. **end if**
30. $k \leftarrow k + 1$
31. Enable all disables actions of $LA \in A$
32. $P \leftarrow V$, $A \leftarrow \phi$
33. **until** $(P(T) > P_{POM}$ or $k > K)$
34. **end Algorithm**

شکل ۳: الگوریتم ۱ نسخه بهبودیافته الگوریتم اکبری- میبدی [۱۶] برای حل مسأله درخت پوشای کمینه تصادفی.

به این ترتیب از (۱۴) و (۱۸) داریم

$$\begin{aligned} Err &= W - W_{TH}(n-1) \\ W_{TH}(n) &= W_{TH}(n-1) + \alpha \times Err \\ v(n) &= v(n-1) + \beta \times (|Err| - v(n-1)) \end{aligned} \qquad (19)$$

در (۱۹) مقدار v نشان‌دهنده انحراف از میانگین یا همان $mdev$ است. کران بالای پیشنهادی ما برای مسایلی که با کمترین هزینه سر و کار دارند $W_{TH}(n)/2 + f \times v(n)$ و کران پایین پیشنهادی در مسایلی که با بیشترین هزینه سروکار دارند $W_{TH}(n)/2 - f \times v(n)$ است.

۶- بررسی تجربی

برای بررسی تأثیر روش جدید، این معیار جدید را روی دو الگوریتم موجود به کار گرفته‌ایم: الگوریتم به کار رفته در حل مسأله کوتاه‌ترین مسیر تصادفی به کمک DLA، ارائه‌شده در [۱۴] و الگوریتم پیشنهادشده در [۱۶] برای حل مسأله درخت پوشای کمینه تصادفی.

برای شبیه‌سازی، هر یک از این الگوریتم‌ها را ابتدا با معیار معرفی‌شده توسط خود الگوریتم (همان معیار میانگین وزنی پویا، معرفی‌شده در (۱۴) با مقدار $\alpha = 1/n$) و سپس با معیار جدید معرفی‌شده به کار برده‌ایم. برای هر الگوریتم روی یک گراف ۲ مرحله شبیه‌سازی انجام داده‌ایم. در مرحله اول به منظور بررسی تأثیر معیارهای مقایسه‌ای در سرعت همگرایی، الگوریتم را به تعداد K بار اجرا کرده‌ایم. در هر دور احتمال انتخاب جواب بهینه $-$OSP$-$ یعنی حاصل ضرب احتمال انتخاب یال‌های تشکیل‌دهنده جواب بهینه را محاسبه کرده‌ایم. این کار را به تعداد noOfRuns بار اجرا کرده و سپس میانگین مقادیر OSP را محاسبه کرده‌ایم. الگوریتم مورد استفاده برای پاداش یا جریمه اقدام‌های آتاماتاها در تمام آزمایش‌هایی که در ادامه صورت گرفته است، الگوریتم خطی L_{R-I} می‌باشد.

در مرحله دوم و به منظور سنجش تعداد نمونه‌گیری‌ها و زمان لازم برای اجرای هر الگوریتم، هر الگوریتم را تا زمان رسیدن به جوابی با احتمال بیش از مقدار از پیش تعیین شده P_{TH} اجرا کرده‌ایم. در صورت یافت‌نشدن چنین جوابی، الگوریتم پس از K بار اجرا متوقف می‌شود. در صورتی که زیرگراف تعیین‌شده توسط الگوریتم در پایان K بار اجرا یا رسیدن به زیرگرافی با احتمال بیشتر از P_{TH} برابر با جواب بهینه باشد، آن دور از اجرا را همگرا گوییم و در غیر این صورت آن را واگرا می‌نامیم. این کار را به تعداد noOfRuns بار اجرا کرده و متوسط تعداد نمونه‌های گرفته‌شده در یک دور اجرا (AS)، متوسط تعداد نمونه‌هایی که از جواب بهینه گرفته شده است (OAS)، متوسط تعداد تکرارهای لازم (AI) و درصد همگرایی الگوریتم به جواب بهینه (PC) را محاسبه کرده‌ایم.

برای حل مسأله کوتاه‌ترین مسیر گراف‌های تصادفی Net۳ و Net۴ مورد استفاده قرار گرفته‌اند [۱۴]. Net۳ یک گراف تصادفی با ۱۰ رأس و ۲۱ یال است که در آن مسیر $\pi^* = ۱ \rightarrow ۴ \rightarrow ۹ \rightarrow ۱۰$، مسیر با کمترین وزن مورد انتظار میان رأس مبدأ $v_s = ۱$ و رأس مقصد $v_d = ۱۰$ است. مسأله یافتن درخت پوشای کمینه روی گراف تصادفی $Alex۱-a$ و $Alex۱-b$ حل شده است. در گراف $Alex۱-b$، درخت $\tau^* = \{(۱,۲)(۱,۴)(۴,۳)(۳,۶)(۶,۵)(۳,۷)(۷,۸)\}$ درخت پوشای کمینه تصادفی با وزن متوسط ۶۰٫۸ است [۳۳].

۶-۱ حل مسأله درخت پوشای کمینه تصادفی

برای این آزمایش، حل مسأله درخت پوشای کمینه را در گراف تصادفی مورد بررسی قرار داده‌ایم. روش استفاده‌شده برای این آزمایش، روشی است که در [۱۶] ارائه شده است. الگوریتم اصلی ارائه‌شده در این مقاله را اکبری- میبدی [۱۶] نامیده‌ایم. روش کار این الگوریتم بدین صورت است که مجموعه‌ای از آتاماتاها- هر آتاماتا متناظر با یکی از رئوس گراف تصادفی- با یکدیگر برای یافتن درخت پوشا همکاری می‌کنند. هر آتاماتا در یکی از وضعیت‌های فعال و غیر فعال قرار دارد. در ابتدای کار تمامی آتاماتاها غیر فعال هستند. در هر مرحله، یکی از آتاماتاهای غیر فعال انتخاب شده و بر اساس بردار احتمال انتخاب اقدام‌های خود، اقدامی را انتخاب می‌کند (در حقیقت یک یال انتخاب می‌شود). انتخاب آتاماتا به گونه‌ای صورت می‌گیرد که منجر به تشکیل حلقه نشود و برای این منظور، آتاماتاها دارای تعداد متغیر اقدام هستند و بعد از انتخاب اقدام یک آتاماتا، سایر آتاماتاهای فعال‌نشده موجود، بررسی شده و اقدام‌هایی که انتخاب آنها توسط آتاماتا منجر به بروز حلقه می‌شود، غیر فعال می‌شوند. انتخاب یک آتاماتای غیر فعال و فعال‌سازی آن کاملاً تصادفی انجام می‌گیرد.

نسخه بهبودیافته این الگوریتم با استفاده از شیوه پیشنهادی را الگوریتم ۱ نامیده‌ایم (شکل ۳).

جدول ۱: نتایج شبیه‌سازی برای مسأله درخت پوشای کمینه تصادفی (ALEX۱-A).

نرخ یادگیری	الگوریتم بهبودیافته (الگوریتم ۱)				Original-AM			
	AS	OAS	AI	PC	AS	OAS	AI	PC
۰/۰۰۶	۸۸۷۷۸	۷۷۱۱۸	۱۲۶۸۳	٪۱۰۰	۹۴۶۹۴	۸۰۰۸۷	۱۳۵۲۹	٪۱۰۰
۰/۰۰۷	۷۰۶۳۵	۶۱۱۱۴	۱۰۰۹۰	٪۱۰۰	۷۹۱۸۴	۶۶۹۰۶	۱۱۳۱۳	٪۱۰۰
۰/۰۰۸	۶۰۹۲۰	۵۲۵۳۸	۸۷۰۴	٪۱۰۰	۷۰۳۹۰	۵۹۵۴۴	۹۳۲۰	٪۱۰۰
۰/۰۰۹	۵۶۷۲۵	۴۹۲۹۰	۸۱۰۴	٪۱۰۰	۶۴۶۹۱	۵۴۳۶۰	۹۲۴۲	٪۱۰۰
۰/۰۱	۵۰۳۸۵	۴۳۸۴۹	۷۲۱۳	٪۱۰۰	۶۰۶۹۰	۵۱۲۳۴	۷۹۱۰	٪۱۰۰
۰/۰۲	۲۴۸۳۲	۲۱۴۰۷	۳۵۴۸	٪۱۰۰	۴۸۴۳۳	۴۱۵۹۸	۳۶۳۴	٪۹۰
۰/۰۳	۱۷۷۳۷	۱۵۳۱۴	۲۵۳۴	٪۱۰۰	۶۶۱۳۴	۵۶۴۹۶	۲۷۹۸	٪۷۵
۰/۰۰۴	۱۱۷۹۱	۱۰۰۵۳	۱۶۸۵	٪۱۰۰	۶۳۳۵۶	۵۴۲۶۷	۲۶۶۱	٪۵۵
۰/۰۵	۹۶۸۳	۸۲۲۶	۱۳۸۳	٪۱۰۰	۶۸۵۷۹	۵۶۶۵۹	۶۰۵۲	٪۲۰
۰/۰۶	۷۶۲۰	۶۴۳۷	۱۰۸۹	٪۱۰۰	۶۶۵۵۳	۵۳۲۶۴	۳۱۶۷	٪۲۵
۰/۰۷	۶۸۲۲	۵۷۹۷	۹۷۵	٪۱۰۰	۱۲۶۴۵۳	۱۰۰۵۳۶	۴۸۸۳	٪۱۰
۰/۰۸	۱۲۹۳۹	۱۰۸۹۵	۱۰۹۸	٪۸۵	۵۲۱۹۸	۴۰۹۱۲	۲۳۳۰	٪۲۵
۰/۰۹	۱۱۸۸۲	۱۰۳۸۵	۸۳۱	٪۸۵	۷۱۵۴۰	۵۳۵۹۳	۶۹۸۰	٪۵
۰/۱	۴۴۹۳	۳۷۰۷	۶۴۲	٪۱۰۰	۸۵۶۰۷	۶۷۴۶۶	۸۲۳۷	٪۵

جدول ۲: نتایج شبیه‌سازی برای مسأله درخت پوشای کمینه تصادفی (ALEX۱-B).

نرخ یادگیری	الگوریتم بهبودیافته (الگوریتم ۱)				Original-AM			
	AS	OAS	AI	PC	AS	OAS	AI	PC
۰/۰۰۷	۱۱۷۰۸۳	۱۰۲۱۸۷	۱۶۶۲۲	٪۱۰۰	۹۵۳۰۴	۸۱۲۶۵	۱۳۵۹۷	٪۹۹
۰/۰۰۸	۱۰۲۱۲۵	۸۹۲۵۰	۱۴۵۵۸	٪۱۰۰	۸۱۶۷۱	۶۹۶۸۳	۱۱۶۶۵	٪۱۰۰
۰/۰۰۹	۸۹۰۹۷	۷۷۷۶۹	۱۲۷۰۶	٪۱۰۰	۶۷۳۹۲	۵۶۸۶۳	۹۶۲۸	٪۱۰۰
۰/۰۱	۸۰۵۶۸	۷۰۳۳۴	۱۱۴۴۴	٪۱۰۰	۶۳۵۴۷	۵۳۷۹۶	۹۰۰۰	٪۹۹
۰/۰۲	۳۹۳۴۳	۳۴۱۶۵	۵۶۲۱	٪۱۰۰	۴۱۰۶۵	۳۴۶۵۶	۴۳۶۸	٪۹۱
۰/۰۳	۲۳۰۸۶	۱۹۷۹۵	۳۲۹۸	٪۱۰۰	۴۳۱۵۱	۳۶۳۱۳	۳۵۵۲	٪۷۷
۰/۰۰۴	۱۷۸۶۸	۱۵۳۲۶	۲۵۵۳	٪۱۰۰	۴۸۱۴۱	۴۰۹۹۹	۲۶۴۶	٪۶۹
۰/۰۵	۱۳۵۸۷	۱۱۵۵۸	۱۹۴۱	٪۱۰۰	۴۳۱۶۷	۳۵۵۹۴	۲۳۰۶	٪۵۰
۰/۰۶	۱۲۴۹۰	۱۰۵۴۸	۱۶۴۵	٪۹۸	۴۲۹۸۵	۳۴۷۳۵	۲۱۳۱	٪۴۲
۰/۰۷	۸۹۵۴	۷۴۸۹	۱۳۹۰	٪۹۷	۴۳۴۳۵	۳۴۲۹۲	۲۴۳۸	٪۳۱
۰/۰۸	۹۳۸۳	۷۹۱۷	۱۱۷۰	٪۹۸	۴۷۶۸۸	۳۸۴۶۹	۱۸۰۶	٪۲۹
۰/۰۹	۱۲۷۹۳	۱۰۸۴۴	۱۱۸۳	٪۹۱	۴۱۸۸۶	۳۱۵۶۷	۲۰۹۷	٪۲۳

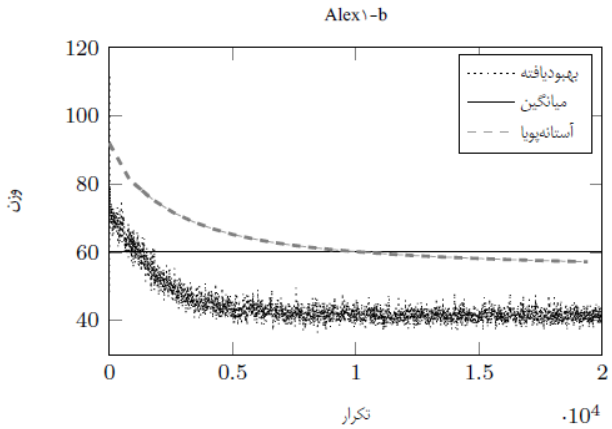
آزمایش ۱

در این آزمایش، کار یافتن درخت پوشای کمینه روی گراف‌های تصادفی $Alex۱-a$ و $Alex۱-b$ برای هر یک از دو الگوریتم اصلی و بهبودیافته تا رسیدن به درختی با احتمال بیش از ۹۰٪ و یا رسیدن به حداکثر تکرار $K=۲۰۰۰۰$ دور انجام شده است. متوسط‌گیری‌ها روی ۱۰۰ بار (noOfRuns) صورت گرفته است. نتایج مقایسه‌ای شاخص‌ها برای الگوریتم اصلی و بهبودیافته را در جداول ۱ و ۲ مشاهده می‌کنید.

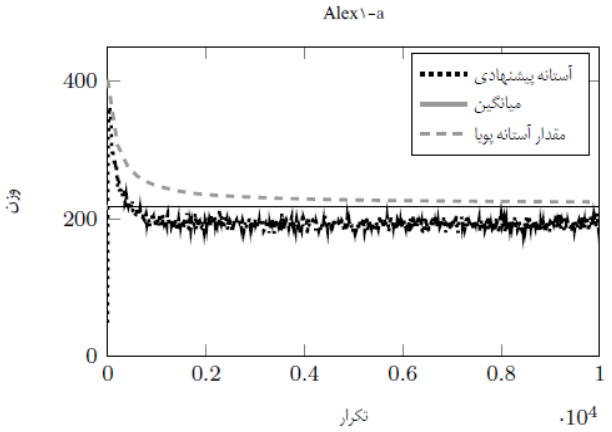
بر اساس نتایج مندرج در این جداول، نکات زیر قابل استخراج است:

۱) مقایسه ستون‌های مربوط به درصد همگرایی (PC) در جداول ۱ و ۲ نشان می‌دهد روش جدید پیشنهادی در مقایسه با روش اصلی، تقریباً مستقل از نرخ یادگیری همگرا است، گرچه با بزرگ‌شدن مقادیر نرخ یادگیری، در عملکرد الگوریتم مقداری ناپایداری مشاهده می‌شود. این درحالی است که در روش اصلی که مبتنی بر میانگین وزنی مقادیر است، همگرایی به نرخ یادگیری وابستگی زیادی دارد به نحوی که برای مقادیر کوچک نرخ یادگیری، همگرایی تضمین شده است اما در مقادیر بزرگ نرخ یادگیری، الگوریتم دچار ناپایداری است.

۲) مقایسه ستون‌های مربوط به میانگین تعداد تکرار لازم (AI) در دو الگوریتم در جدول ۱ نشانگر آن است که میانگین تعداد تکرارهای لازم برای همگرایی در الگوریتم اصلی پیشنهادی در [۱۶] همواره بیشتر از الگوریتم بهبودیافته پیشنهادی است. این اختلاف در مقادیر کوچک نرخ یادگیری حدود ۱۰٪ است که با افزایش مقدار نرخ یادگیری و ناپایدارشدن روش اصلی، این اختلاف تا ۹۰٪ نیز افزایش می‌یابد. نکته دیگری که در جدول ۱ قابل مشاهده می‌باشد آن است که حداقل متوسط تعداد تکرار لازم با اطمینان ۱۰۰٪ از همگرایی در روش اصلی، ۷۹۱۰ تکرار با متوسط ۶۰۶۹۰ نمونه است (در نرخ یادگیری ۰/۰۱) و این در حالی است که همین حداقل‌ها برای روش جدید پیشنهادی متوسط ۹۷۵ تکرار با متوسط نمونه‌گیری ۶۸۲۲ نمونه است (در نرخ یادگیری ۰/۰۷) که این خود،



شکل ۴: مقایسه مقدار آستانه پیشنهادی مورد استفاده توسط الگوریتم اکبری- میبدی [۱۶] و نسخه بهبودیافته آن. آستانه پویا هموارتر است اما مقدار آستانه پیشنهادی در الگوریتم بهبودیافته نوسان‌های بیشتری داشته و بسیار محتاطانه‌تر عمل می‌کند.

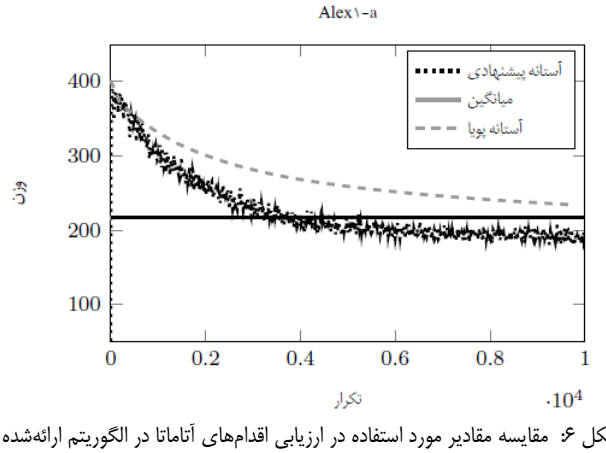


شکل ۵: مقایسه مقادیر مورد استفاده در ارزیابی اقدام‌های آتاماتا در الگوریتم ارائه‌شده در [۱۶] و الگوریتم ۱ پیشنهادی در مقایسه با میانگین وزن درخت پوشای بهینه (نرخ یادگیری ۰/۰۷ بوده است).

نشان‌دهنده کاهش بیش از ۸۰٪ در متوسط تعداد تکرارها و نمونه‌گیری‌ها است.

۳) نتایج بررسی داده‌های جدول ۱ نشان می‌دهد که الگوریتم اصلی پیشنهادی در [۱۶] برای مقادیر نرخ یادگیری کوچک‌تر از ۰/۰۲ همواره همگرا است و علاوه بر این در مقایسه با روش بهبودیافته، هم از نظر متوسط تعداد نمونه‌گیری‌ها و هم از نظر متوسط تعداد تکرار لازم برای رسیدن به جواب، عملکرد بهتری دارد، به نحوی که متوسط تعداد تکرارهای روش اصلی نسبت به روش بهینه‌شده کاهشی در حدود ۲۵٪ نشان می‌دهد. اما با افزایش نرخ یادگیری مجدداً مشاهده می‌شود الگوریتم پیشنهادی عملکرد بهتری دارد و تقریباً مستقل از نرخ یادگیری، همواره همگرا بوده و علاوه بر این، برای مقادیر بزرگ‌تر نرخ یادگیری (بزرگ‌تر از ۰/۰۲) همواره متوسط تعداد نمونه‌گیری‌های کمتری نسبت به روش اصلی لازم دارد. کاهش متوسط تعداد نمونه‌ها از حدود ۴۵٪ برای نرخ یادگیری ۰/۰۳ تا حدود ۸۰٪ برای نرخ یادگیری ۰/۰۸ متغیر است که یکی از دلایل این نوسان، ناپایداری الگوریتم اصلی در نرخ‌های یادگیری بزرگ‌تر است.

۴) بر اساس داده‌های جدول ۱ مشاهده می‌شود که حداقل متوسط تعداد تکرار لازم با اطمینان همگرایی ۱۰۰٪ در روش اصلی، ۹۶۲۸ تکرار با متوسط ۶۷۳۹۲ نمونه است (در نرخ یادگیری ۰/۰۹) و این در حالی است که همین حداقل‌ها برای روش جدید پیشنهادی،



شکل ۶: مقایسه مقادیر مورد استفاده در ارزیابی اقدام‌های آتاماتا در الگوریتم ارائه‌شده در [۱۶] و الگوریتم ۱ پیشنهادی در مقایسه با میانگین وزن درخت پوشای بهینه (نرخ یادگیری ۰/۰۷ بوده است).

متوسط ۱۹۴۱ تکرار با متوسط نمونه‌گیری ۱۳۵۸۷ نمونه است (در نرخ یادگیری ۰/۰۵) که این خود نشان‌دهنده کاهش حدود ۸۰٪ در متوسط تعداد تکرارها و نمونه‌گیری‌ها است.

در شکل ۴ مقادیر آستانه مقایسه‌ای در الگوریتم اصلی و الگوریتم بهبودیافته در مقایسه با مقدار وزن واقعی درخت پوشای کمینه تصادفی در $Alex۱-b$ نشان داده شده‌اند. به نظر می‌رسد دلیل عملکرد خوب الگوریتم بهبودیافته فرار از نقاط بهینه محلی با استفاده از تزریق نویز به کمک واریانس تخمینی است. علاوه بر این، فاصله بیشتر مقدار بهبودیافته پیشنهادی با مقدار واقعی میانگین وزن درخت پوشا در مقایسه با مقدار آستانه پویا باعث می‌شود روش بهبودیافته محتاطانه‌تر عمل کند.

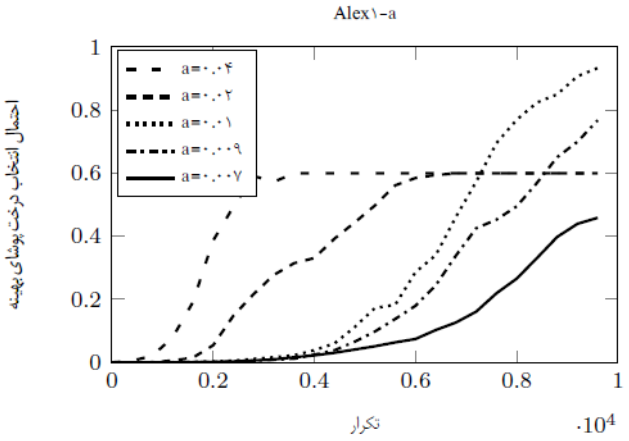
مقایسه‌ای مشابه نیز برای گراف تصادفی $Alex۱-a$ در شکل ۵ و شکل ۶ قابل مشاهده است. اثر نرخ یادگیری در مقادیر آستانه پویا و مقدار بهبودیافته پیشنهادی در این مقاله نیز در این شکل‌ها قابل مشاهده است.

آزمایش ۲

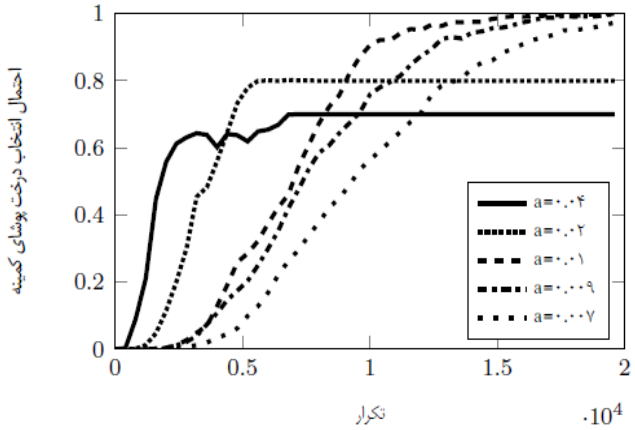
برای انجام آزمایش‌های این بخش از نمودارهایی استفاده کرده‌ایم که رفتار همگرایانه الگوریتم‌ها را به تصویر کشیده است. در این نمودارها هر نقطه با مختصات (x, y) بیانگر احتمال انتخاب (y) زیرگراف بهینه (در اینجا درخت پوشای کمینه) در گراف تصادفی در xامین تکرار الگوریتم است. مقایسه این نمودارها می‌تواند رفتار الگوریتم‌ها و نحوه همگرایی آنها به جواب بهینه را نشان دهد. از آنجا که محاسبات مورد استفاده در آتاماتای یادگیر بسیار ساده و فاقد پیچیدگی است، این نمودارها می‌تواند توصیفی از زمان اجرای الگوریتم نیز باشد.

برای این منظور در ابتدا مسأله درخت پوشای کمینه را روی گراف $Alex۱-a$ با استفاده از الگوریتم پیشنهادی در [۱۶] و نسخه بهبودیافته پیشنهادی در این مقاله (الگوریتم ۱) حل کرده‌ایم. حداکثر تعداد تکرار برای این گراف $K=۱۰۰۰۰$ در نظر گرفته شده است. در هر تکرار حاصل ضرب احتمال انتخاب اقدام‌های متناظر با یال‌های تشکیل‌دهنده درخت پوشای بهینه محاسبه شده است. در شکل‌های ۷ و ۸ نتایج تغییر در احتمال انتخاب درخت پوشای بهینه به ازای مقادیر مختلف نرخ یادگیری روی گراف $Alex۱-a$ در هر دو الگوریتم آمده است.

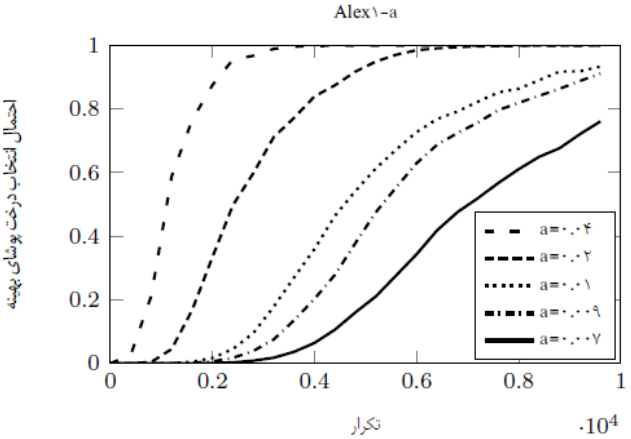
همان گونه که در این شکل‌ها مشاهده می‌شود سرعت همگرایی الگوریتم با معیار جدید پیشنهادی در این مقاله در مقایسه با مقدار آستانه پویای مورد استفاده در روش‌های قبلی بیشتر بوده و الگوریتم با معیار جدید تقریباً مستقل از نرخ یادگیری، همواره همگرا است، علاوه بر این که روش جدید پیشنهادی در مقایسه با روش قبلی روال یک‌نواتری دارد



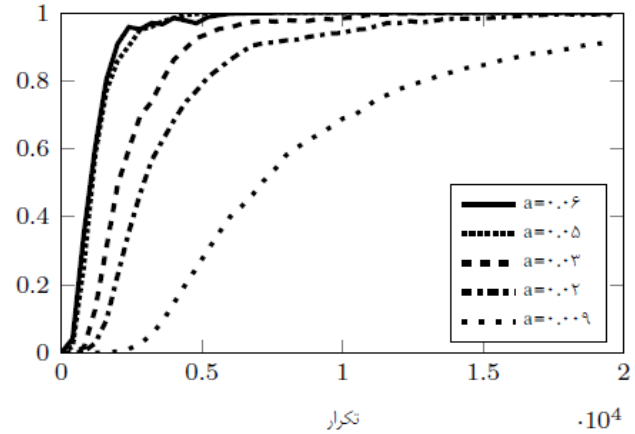
شکل ۷: نحوه تغییر احتمال انتخاب درخت پوشای کمینه در الگوریتم [۱۶] به ازای مقادیر مختلف نرخ یادگیری.



شکل ۹: نحوه تغییر احتمال انتخاب درخت پوشای کمینه در الگوریتم [۱۶] به ازای مقادیر مختلف نرخ یادگیری.



شکل ۸: نحوه تغییر احتمال انتخاب درخت پوشای کمینه در الگوریتم ۱ به ازای مقادیر مختلف نرخ یادگیری.



شکل ۱۰: نحوه تغییر احتمال انتخاب درخت پوشای کمینه در الگوریتم ۱ به ازای مقادیر مختلف نرخ یادگیری. (در متن ارجاع ندارد)

(به خصوص با بزرگ‌تر شدن مقادیر نرخ یادگیری). به بیان دیگر اگر (x_1, y_1) و (x_2, y_2) دو نقطه روی نمودار احتمال انتخاب جواب بهینه در روش جدید باشند و $x_2 > x_1$ باشد آن گاه تقریباً همواره $y_2 > y_1$ است. این نشان می‌دهد که با افزایش تعداد تکرارها روش پیشنهادی با احتمال بیشتری به جواب بهینه همگرا می‌شود. این ویژگی در مقادیر کوچک نرخ یادگیری برای الگوریتم مبتنی بر معیار آستانه پویا نیز برقرار است.

علاوه بر این شکل ۱۱ نتایج مقایسه‌ای رفتار هر دو معیار را به ازای مقادیر یکسان نرخ یادگیری نشان می‌دهد. با فرض ثابت‌بودن نرخ یادگیری، شیب میل به جواب بهینه در معیار جدید بیشتر است.

آزمایشی مشابه با آزمایش بالا روی گراف $Alex۱-b$ نیز صورت گرفته است. نمودارهای رفتار همگرایانه الگوریتم پیشنهاد مبتنی بر مقدار آستانه پویا پیشنهادی در [۱۶] و مقدار آستانه جدید پیشنهادی (الگوریتم ۱) به ازای مقادیر مختلف نرخ یادگیری به ترتیب در شکل ۹ و شکل ۱۰ نشان داده شده‌اند. نمودار مقایسه‌ای رفتار هر دو الگوریتم بر روی گراف $Alex۱-b$ با نرخ‌های یادگیری مشابه نیز در شکل ۱۲ نشان داده شده که برتری معیار پیشنهادی را در مقایسه با معیار آستانه پویا نشان می‌دهند.

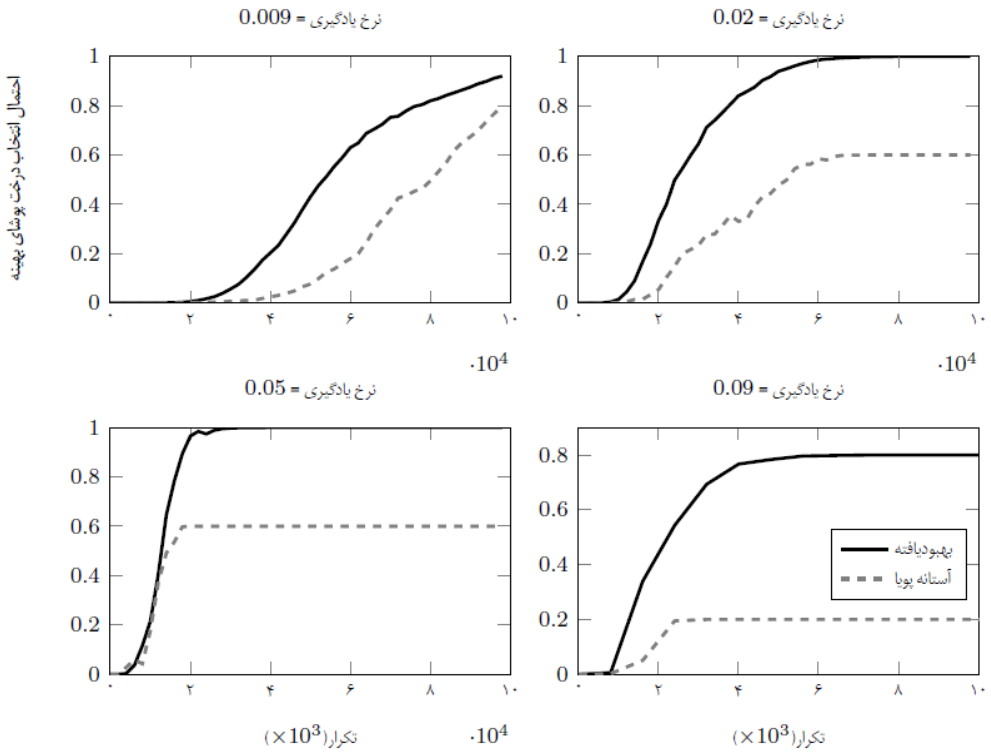
۶-۲ حل مسأله کوتاه‌ترین مسیر تصادفی

برای این بخش، حل مسأله یافتن کوتاه‌ترین مسیر را در گراف تصادفی مورد بررسی قرار داده‌ایم. روش استفاده‌شده برای این آزمایش، روشی است که در [۱۴] ارائه شده است. الگوریتم اصلی ارائه‌شده در این مقاله را الگوریتم بیگی- میبدی [۱۴] نامیده‌ایم. نسخه بهبودیافته این الگوریتم با استفاده از شیوه پیشنهادی را الگوریتم ۲ نامیده‌ایم (شکل ۱۹).

آزمایش ۳

در این آزمایش، کار یافتن کوتاه‌ترین مسیر روی گراف تصادفی Net۳ برای هر یک از دو الگوریتم اصلی و بهبودیافته تا رسیدن به مسیر با احتمال بیش از ۹۰٪ و یا رسیدن به حداکثر تکرار $K = ۲۰۰۰۰$ دور انجام شده و متوسط‌گیری‌ها روی ۱۰۰ بار صورت گرفته است. نتایج مقایسه شاخص‌ها برای الگوریتم اصلی و بهبودیافته را در جدول ۳ مشاهده می‌کنید. همان گونه که در جدول آمده است با بزرگ‌تر شدن نرخ یادگیری (بزرگتر از ۰/۰۱) درصد همگرایی الگوریتم اصلی کاهش می‌یابد. حداقل تعداد نمونه‌گیری از یال‌ها در روش جدید با درصد همگرایی ۱۰۰٪ تعداد ۳۹۶۴ نمونه‌گیری و با متوسط تکرار ۱۰۵۵ دور تکرار است (در نرخ یادگیری ۰/۰۱) و این در حالی است که حداقل تعداد نمونه‌گیری در روش پیشنهادی با اطمینان ۱۰۰٪ از همگرایی، ۷۹۲ نمونه‌گیری و با متوسط ۲۰۹ تکرار است (در نرخ یادگیری ۰/۰۵). این مقایسه نشان می‌دهد که الگوریتم پیشنهادی می‌تواند تا ۸۰٪ از تعداد نمونه‌گیری‌ها بکاهد (و به همین میزان در تعداد تکرارهای لازم برای همگرایی به جواب بهینه نیز کاهش وجود دارد).

مقایسه رفتار لحظه‌ای و نحوه همگرایی دو الگوریتم را نیز در شکل ۱۳ مشاهده می‌کنید. نمودارها بهبود عملکرد معیار پیشنهادی جدید به ازای مقادیر مختلف نرخ یادگیری را نشان می‌دهند.

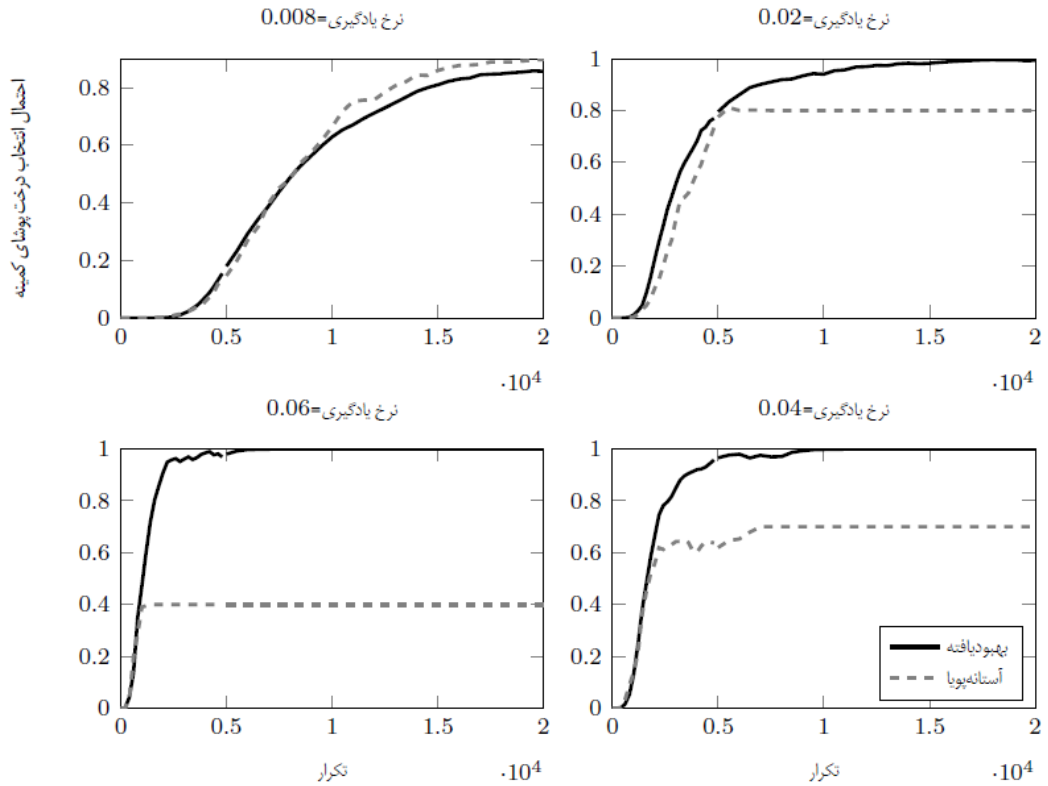


شکل ۱۱: مقایسه نحوه تغییر احتمال انتخاب درخت پوشای کمینه در گراف $Alex۱-a$ به ازای مقادیر مختلف نرخ یادگیری در نسخه اصلی (آستانه پویا) [۱۶] و نسخه بهبودیافته (الگوریتم ۱).

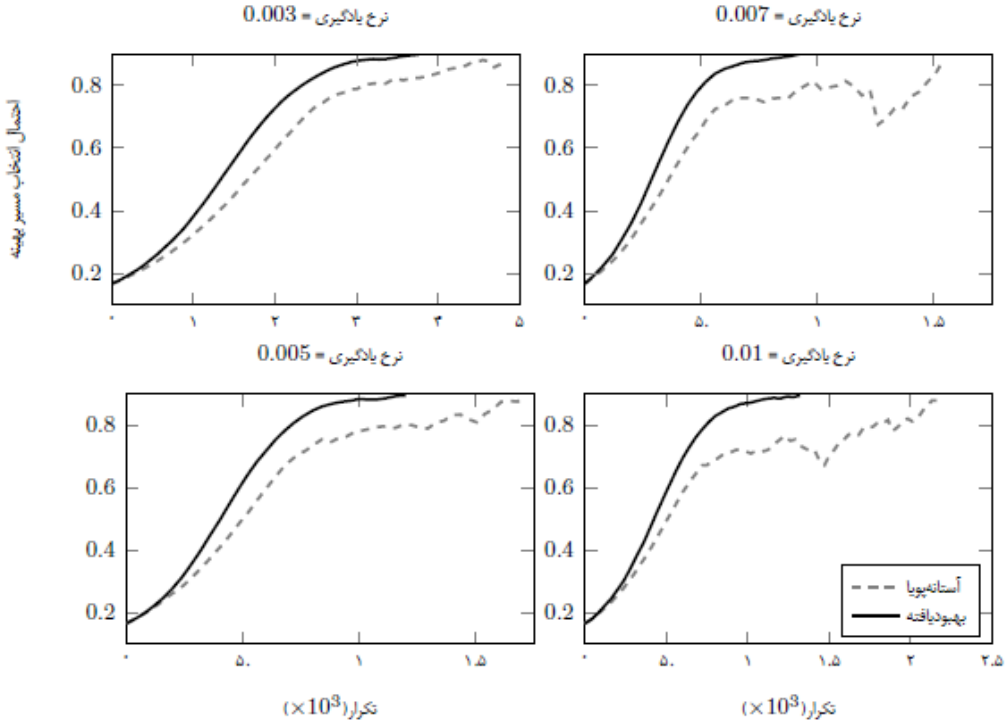
جدول ۳: نتایج شبیه‌سازی برای مسأله کوتاه‌ترین مسیر تصادفی (NET۳).

نرخ یادگیری	الگوریتم بهبودیافته (الگوریتم ۲)				الگوریتم اصلی بیگی- میبدی[۱۴]			
	AS	OAS	AI	PC	AS	OAS	AI	PC
۰/۰۰۲	۱۸۳۴۹	۹۸۳۵	۴۸۴۲	٪۱۰۰	۱۸۷۳۹	۸۹۶۹	۴۹۲۶	٪۱۰۰
۰/۰۰۳	۱۲۲۰۷	۶۵۲۸	۳۲۲۰	٪۱۰۰	۱۲۴۴۲	۵۹۴۷	۳۲۸۰	٪۱۰۰
۰/۰۰۴	۹۲۱۶	۴۹۳۰	۲۴۳۱	٪۱۰۰	۹۳۳۸	۴۴۶۲	۲۴۶۰	٪۱۰۰
۰/۰۰۵	۷۳۶۰	۳۸۹۸	۱۹۳۶	٪۱۰۰	۷۵۳۴	۳۵۸۵	۱۹۹۵	٪۱۰۰
۰/۰۰۶	۶۱۶۶	۳۲۶۷	۱۶۲۳	٪۱۰۰	۶۰۵۹	۲۸۹۸	۱۶۰۲	٪۱۰۰
۰/۰۰۷	۵۲۳۴	۲۷۸۱	۱۳۸۱	٪۱۰۰	۵۴۷۸	۲۶۰۱	۱۴۵۴	٪۱۰۰
۰/۰۰۸	۴۶۶۷	۲۴۴۸	۱۲۲۸	٪۱۰۰	۴۸۱۲	۲۲۵۲	۱۲۸۰	٪۱۰۰
۰/۰۰۹	۴۰۰۸	۲۱۱۸	۱۰۵۴	٪۱۰۰	۴۱۶۳	۱۹۶۶	۱۱۰۴	٪۱۰۰
۰/۰۱	۳۷۷۳	۱۹۶۷	۹۹۱	٪۱۰۰	۳۹۶۴	۱۸۶۵	۱۰۵۵	٪۱۰۰
۰/۰۲	۱۹۵۴	۱۰۰۳	۵۱۵	٪۱۰۰	۲۰۰۹	۸۹۷	۵۳۴	٪۹۶
۰/۰۳	۱۳۰۰	۶۵۹	۳۴۳	٪۱۰۰	۱۲۹۷	۵۸۵	۳۴۶	٪۹۷
۰/۰۴	۱۴۶۳	۴۹۰	۲۵۸	٪۱۰۰	۱۱۸۲	۴۶۰	۳۲۱	٪۸۹
۰/۰۵	۷۹۲	۳۹۰	۲۰۹	٪۱۰۰	۸۸۳	۳۵۴	۲۳۵	٪۹۰
۰/۰۶	۷۵۹	۳۳۸	۲۰۳	٪۹۶	۸۵۰	۳۱۷	۲۲۷	٪۸۳
۰/۰۷	۶۷۶	۳۰۴	۱۸۰	٪۹۶	۶۷۵	۲۴۳	۱۸۱	٪۸۴
۰/۰۸	۶۱۲	۲۵۶	۱۶۵	٪۹۵	۵۷۴	۲۰۷	۱۵۲	٪۸۰
۰/۰۹	۱۸۲۹	۲۲۳	۱۳۹	٪۹۱	۱۳۰۳	۱۹۹	۱۵۹	٪۷۱
۰/۱	۱۸۳۴	۲۰۰	۱۲۹	٪۸۸	۴۸۰	۱۶۳	۱۲۶	٪۷۴

رفتار الگوریتم اصلی و بهبودیافته برای یافتن کوتاه‌ترین مسیر تصادفی در شبکه Net۳ به ازای برخی از مقادیر مختلف نرخ یادگیری به ترتیب در شکل‌های ۱۴و ۱۵ نمایش داده شده است. نکته جالب توجه آن است که در هر دو روش با افزایش نرخ یادگیری شیب نمودارها افزایش می‌یابد و به بیان دیگر هر دو الگوریتم با سرعت بیشتری به سمت جواب بهینه حرکت می‌کند. اما در الگوریتم بهبودیافته این شیب با کاهش بارز تعداد تکرارها نیز همراه است در حالی که در الگوریتم اصلی، این افزایش شیب به علت ناپایداری در رفتار الگوریتم موجب کاهش چشمگیری در تعداد تکرارها نمی‌شود. در اینجا نیز مشاهده می‌شود معیار جدید پیشنهادی موجب رفتار هموارتر الگوریتم می‌شود.



شکل ۱۲: احتمال انتخاب درخت پوشای کمینه در گراف $Alex۱-b$ و تغییر آن در طول اجرای الگوریتم به ازای مقادیر مختلف نرخ یادگیری (نسخه اصلی و بهبودیافته الگوریتم اکبری- میبدی [۱۶]). با بزرگ‌تر شدن نرخ یادگیری احتمال همگرایی در روش اصلی کاهش می‌یابد.

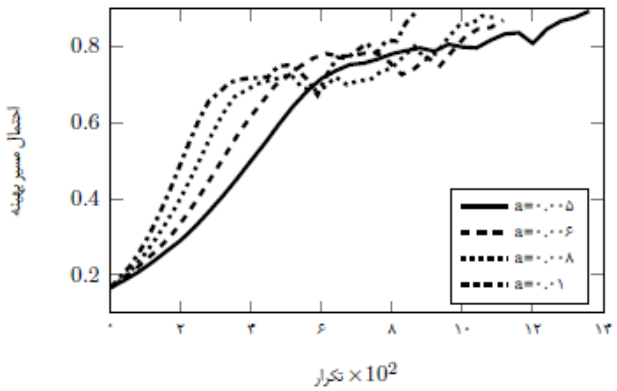


شکل ۱۳: احتمال انتخاب مسیر بهینه در گراف Net۳ و تغییر آن در طول اجرای الگوریتم به ازای مقادیر مختلف نرخ یادگیری (نسخه اصلی و بهبودیافته الگوریتم بیگی- میبدی [۱۴]). با بزرگ‌تر شدن نرخ یادگیری روند همگرایی در روش اصلی مختل می‌شود اما در روش بهبودیافته شیب میل جواب به جواب بهینه افزایش می‌یابد.

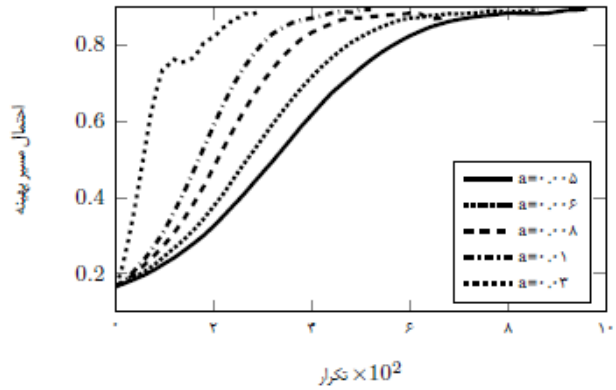
آزمایش ٤

در این آزمایش نیز همانند آزمایش قبلی مسأله یافتن کوتاه‌ترین مسیر روی گراف تصادفی Net۴ به کمک الگوریتم ارائه‌شده در [۱۴] و نسخه بهبودیافته (الگوریتم ۲) حل شده است. نتایج شاخص‌ها روی ۱۰۰ بار تکرار الگوریتم‌ها متوسط‌گیری شده و در جدول ۴ آورده شده است.

مقایسه مقادیر شاخص‌ها به ازای هر نرخ یادگیر، نشان‌دهنده بدتر بودن معیار پیشنهادی است اما اولاً الگوریتم پیشنهادی تقریباً به ازای تمامی مقادیر نرخ یادگیری همگرا است و ثانیاً حداقل تعداد تکرار لازم برای ۱۰۰٪ همگرایی در معیار پیشنهادی ۱۲۶۱ تکرار است و در معیار اصلی (آستانه پویا) ۳۸۶۵ تکرار (چیزی بیش از ۳ برابر تعداد تکرارهای لازم در روش پیشنهادی). به همین ترتیب مقایسه حداقل تعداد نمونه‌های لازم برای ۱۰۰٪ همگرایی نشان می‌دهد در روش پیشنهادی تعداد ۵۲۹۲



شکل ۱۴: احتمال انتخاب مسیر بهینه روی گراف Net۳ در الگوریتم اصلی بیگی- میبدی [۱۴] به ازای مقادیر مختلف نرخ یادگیری. با بزرگ‌تر شدن نرخ یادگیری شیب نمودار افزایش می‌یابد و رفتار الگوریتم ناپایدار است و افزایش نرخ یادگیری موجب کاهش بارز در تعداد تکرارها برای همگرایی نمی‌شود.



شکل ۱۵: احتمال انتخاب مسیر بهینه روی گراف Net۳ در الگوریتم ۲ (نسخه بهبودیافته الگوریتم بیگی- میبدی [۱۴]) به ازای مقادیر مختلف نرخ یادگیری. با افزایش نرخ یادگیری شیب نمودار افزایش می‌یابد و تعداد تکرارها برای رسیدن به آستانه‌ای قابل قبول از بهینگی نیز با افزایش نرخ یادگیری کاهش می‌یابد.

جدول ۴: نتایج شبیه‌سازی برای مسأله کوتاه‌ترین مسیر تصادفی (Net۴).

نرخ یادگیری	الگوریتم بهبودیافته (الگوریتم ۲)				الگوریتم اصلی بیگی- میبدی [۱۴]			
	AS	OAS	AI	PC	AS	OAS	AI	PC
۰/۰۰۳	۳۱۹۸۸	۲۱۴۷۲	۷۵۵۷	٪۱۰۰	۲۱۹۵۳	۱۲۹۲۶	۵۱۰۸	٪۱۰۰
۰/۰۰۴	۲۳۴۶۰	۱۵۷۳۴	۵۵۴۰	٪۱۰۰	۱۷۲۰۵	۱۰۳۲۹	۳۸۶۵	٪۱۰۰
۰/۰۰۵	۲۰۲۰۵	۱۳۶۱۱	۴۷۹۱	٪۱۰۰	۱۵۱۴۳	۸۹۵۶	۳۴۹۷	٪۹۶
۰/۰۰۶	۱۶۵۷۷	۱۱۱۴۶	۳۹۳۴	٪۱۰۰	۱۱۷۹۸	۶۸۲۳	۲۸۰۰	٪۹۷
۰/۰۰۷	۱۳۷۹۲	۹۲۴۱	۳۲۶۴	٪۱۰۰	۱۱۰۰۶	۶۴۳۴	۲۶۴۲	٪۹۴
۰/۰۰۸	۱۲۵۳۶	۸۴۳۵	۲۹۷۳	٪۱۰۰	۱۰۷۳۱	۶۳۴۸	۲۶۰۳	٪۹۰
۰/۰۰۹	۱۱۰۴۳	۷۴۵۸	۲۶۲۱	٪۱۰۰	۹۲۲۵	۵۴۶۶	۲۲۳۳	٪۹۲
۰/۰۱	۱۰۰۲۶	۶۶۴۱	۲۳۸۱	٪۱۰۰	۹۰۲۳	۵۳۴۶	۲۱۹۸	٪۹۰
۰/۰۲	۵۲۹۲	۳۵۵۳	۱۲۶۱	٪۱۰۰	۵۳۱۲	۳۰۰۳	۱۳۱۳	٪۸۲
۰/۰۳	۴۵۳۰	۳۰۰۱	۱۱۰۸	٪۹۵	۳۴۷۹	۱۷۵۲	۸۴۹	٪۷۹
۰/۰۴	۳۲۲۰	۲۰۲۰	۷۸۰	٪۹۳	۲۸۷۳	۱۳۴۱	۷۰۳	٪۷۰
۰/۰۵	۲۹۲۳	۱۷۷۶	۷۱۴	٪۸۷	۱۹۴۷	۹۱۴	۴۷۶	٪۷۷
۰/۰۶	۲۲۸۷	۱۴۴۰	۵۶۲	٪۹۴	۱۶۰۳	۷۳۵	۳۸۶	٪۷۲
۰/۰۷	۲۰۴۱	۱۱۷۶	۵۰۳	٪۸۸	۱۵۲۰	۶۷۰	۳۶۹	٪۶۳
۰/۰۸	۱۷۱۴	۹۵۸	۴۲۲	٪۸۱	۱۳۰۹	۶۱۰	۳۱۸	٪۶۳
۰/۰۹	۱۹۸۶	۱۱۴۷	۴۹۷	٪۸۴	۱۱۱۸	۴۴۸	۲۶۴	٪۵۹
۰/۱	۱۷۷۸	۹۴۰	۴۴۴	٪۷۵	۹۵۶	۳۸۲	۲۲۶	٪۶۰
۰/۲	۹۹۹	۳۶۹	۲۴۳	٪۵۷	۴۲۸	۱۵۶	۹۹	٪۵۲

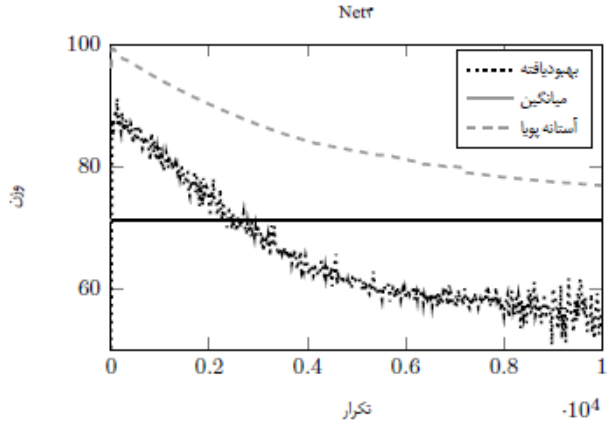
نمونه‌گیری از یال‌ها لازم است اما در روش اصلی ۱۷۲۰۵ مورد (باز هم چیزی بیش از ۳ برابر). مقادیر آستانه مورد استفاده برای ارزیابی اقدام‌های آتاماتا (در مقایسه با وزن درخت کمینه Net۴) در شکل ۱۶ نشان داده شده است.

در آزمایشی دیگر مسأله یافتن کوتاه‌ترین مسیر تصادفی روی گراف Net۴ توسط الگوریتم اصلی [۱۴] و الگوریتم بهبودیافته حل شد. در این آزمایش از هر دو الگوریتم خواسته شده که کار نمونه‌گیری برای یافتن مسیر بهینه را دقیقاً ۱۰۰۰۰ بار تکرار کنند. احتمال انتخاب مسیر بهینه در هر بار نمونه‌گیری اندازه‌گیری شد. این کار ۱۰ بار صورت گرفت و مقادیر متوسط احتمال انتخاب مسیر بهینه روی این ۱۰ بار محاسبه شد. نتایج این کار به ازای مقادیر مختلف نرخ یادگیری را در شکل ۱۷ و شکل ۱۸ مشاهده می‌کنید.

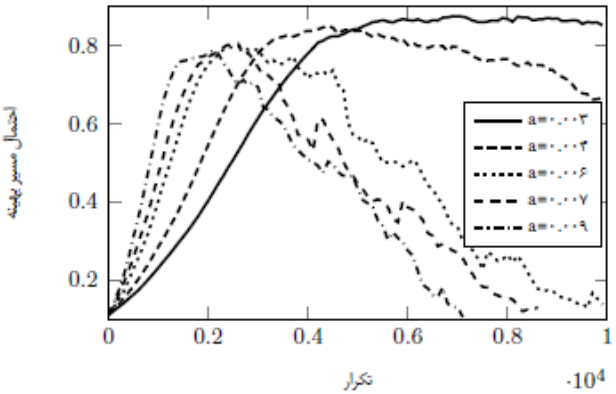
در شکل ۱۷ که اثر ارزیابی رفتار آتاماتاها به کمک مقدار آستانه پویا آمده است، مشاهده می‌شود با بزرگ‌تر شدن نرخ یادگیری و در تکرارهای بیشتر الگوریتم، جوابی غیر از جواب بهینه انتخاب شده و احتمال انتخاب جواب بهینه بعد از تعداد مشخصی نمونه‌گیری مجدداً کاهش می‌یابد اما در شکل ۱۸ مشاهده می‌شود که نه تنها همگرایی الگوریتم مبتنی بر معیار جدید پیشنهادی به نرخ یادگیری وابسته نیست، بلکه انجام جستجوی بیشتر در فضای جواب‌ها نیز منجر به واگرایی الگوریتم نمی‌شود.

۷- نتیجه‌گیری

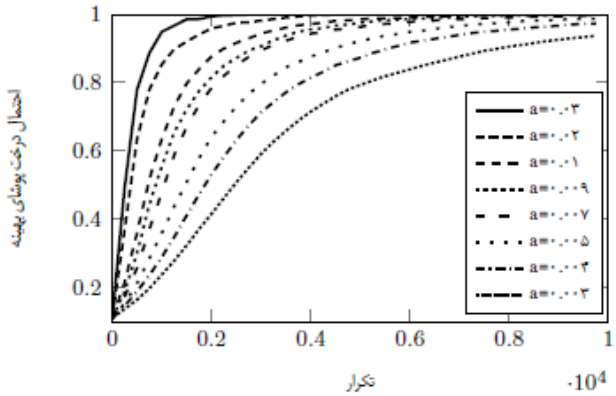
در این مقاله یک معیار جدید برای تعیین پاداش یا جریمه آتاماتاهای یادگیر در حل مسایل بهینه‌سازی روی گراف‌های تصادفی به کمک شبکه‌ای از آتاماتاهای یادگیر معرفی شد. نتایج شبیه‌سازی‌ها نشان داد که اولاً معیار جدید می‌تواند باعث عملکرد مستقل از نرخ یادگیری در آتاماتای یادگیر شود و ثانیاً در بررسی‌ها مشاهده شد دلیل عملکرد بهتر این معیار



شکل ۱۶: مقایسه مقادیر مورد استفاده در ارزیابی اقدام‌های آتاماتا روی گراف Net۴ در الگوریتم ارائه‌شده در [۱۴] و الگوریتم ۲ پیشنهادی در مقایسه با میانگین طول مسیر بهینه (نرخ یادگیری ۰٫۰۰۳ بوده است).



شکل ۱۷: احتمال انتخاب مسیر بهینه در گراف Net۴ با معیار آستانه پویا ([۱۴]). الگوریتم به نرخ یادگیری حساس بوده و با بزرگ‌تر شدن نرخ یادگیری در اثر تکرار بیشتر الگوریتم جوابی غیر از جواب بهینه انتخاب می‌شود.



شکل ۱۸: احتمال انتخاب مسیر بهینه در گراف Net۴ با معیار بهبودیافته (الگوریتم ۲). الگوریتم مستقل از نرخ یادگیری همواره همگرا است.

نسبت به معیار آستانه پویا که تاکنون مورد استفاده بوده است، احتمالاً به دلیل توانایی آن در فاصله‌گیری مناسب از مقدار میانگین واقعی است که در نتیجه آن سیستم یادگیر قابلیت فرار از نقاط کمینه محلی را پیدا می‌کند.

بررسی‌های متعدد روی دو نمونه از مسایل بهینه‌سازی روی گراف‌های تصادفی یعنی حل مسأله کوتاه‌ترین مسیر بین یک مبدأ و مقصد مشخص و حل مسأله یافتن درخت پوشای کمینه تصادفی، بهبود عملکرد الگوریتم‌های موجود فعلی را با معیار جدید نشان داد. معیار جدید به دلیل استفاده از تخمین‌گرهای ساده از پیچیدگی محاسباتی بسیار کمی برخوردار

Input: Stochastic graph $G=(V,E,Q)$, Thresholds $P_{POM}, K, s, d \in V$

Output: Shortest Path between s and d

1. Assign a LA to each node of Graph G
2. Let k be the stage number and initially set to 0
3. Let X, W_X denotes the constructed Path and its weight
4. Let WTH_k be the dynamic threshold at stage k and initially set to MAX
5. $X \leftarrow \phi$; $W_X \leftarrow 0$
6. **begin Algorithm**
7. **repeat**
8. Let P, A denote Passive and Active sets of LAs
9. $P \leftarrow V$, $A \leftarrow \phi$, $u1 = s$
10. **while** $u2 \neq d$ **do**
11. $P \leftarrow P - \{u\}$, $A \leftarrow A + \{u\}$
12. $e = (u1, u2) =$ select_action(u)
13. each automaton in A prunes its action-set for cycle avoidance
14. $X \leftarrow X + e$
15. $W_X \leftarrow W_X + W_e$
16. $P(T) \leftarrow p^{u_1}_{u_2} \times P(T)$
17. $u1 \leftarrow u2$
18. **end while**
19. **if** $|X|$ = path from s to d **then**
20. $E = W_X - WTH_k$
21. $WTH_k = WTH_k + \alpha \times E$
22. $V = V + \beta(abs(Err) - V)$
23. **if** $W_X < (WTH_k + 4 \times V)/2$ **then**
24. reward the selected actions of LAs
25. **else**
26. penalize the selected actions of LAs
27. **end if**
28. **else**
29. Do nothing
30. **end if**
31. $k \leftarrow k + 1$
32. Enable all disables actions of $LA \in A$
33. $P \leftarrow V$, $A \leftarrow \phi$
34. **until** $(P(T) > P_{POM}$ or $k > K)$
35. **end Algorithm**

شکل ۱۹: الگوریتم ۲ نسخه بهبودیافته الگوریتم بیگی- میبدی [۱۴] برای حل مسأله درخت پوشای کمینه تصادفی.

بوده و بر خلاف برخی از روش‌هایی که تاکنون برای تسریع عملکرد الگوریتم‌های پیشنهادی برای حل مسایل بهینه‌سازی روی گراف تصادفی ارائه شده است، هم با منطق ساده محاسباتی در آتاماتای یادگیر همخوانی دارد و هم با منطق کاربرد مدل شبکه‌های تصادفی در عمل- نظیر شبکه‌های داده‌ای- که در آنها تنها مقادیر تجمیع‌شده جواب مسأله مورد نظر روی شبکه‌های تصادفی (تأخیر end-to-end) برای بهینه‌سازی در اختیار است و شبکه شبیه یک جعبه سیاه در نظر گرفته می‌شود سازگار است.

در این مقاله یک ترکیب ساده خطی از مقادیر تخمینی میانگین و واریانس برای ارزیابی مورد استفاده قرار گرفت و تأثیر ضرایب استفاده‌شده در تخمین موضوع مطالعات آتی است.

مراجع

[1] K. S. Narendra and M. A. L. Thathachar, "Learning automata: a survey," *IEEE Trans. Syst. Man, Cyberentics*, vol. 14, no. 4, pp. 323-334, Jul. 1974.

[2] S. Lakshmivarahan, *Learning Algorithms Theory and Applications*, vol. 30, Springer-Verlag, New York, 1981.
[3] G. Santharam and P. S. Sastry, "A reinforcement learning neural network for adaptive control of Markov chains," *IEEE Trans.Syst. Man Cybern. Part A, Syst. Humans*, vol. 27, no. 5, pp. 588-600, Sept. 1997.
[4] B. J. Oommen and D. C. Y. Ma, "Deterministic learning automata solutions to the equipartitioning problem," *IEEE Trans. Comput.*, vol. 37, no. 1, pp. 2-13, Jan. 1988.
[5] B. J. Oommen and E. V De St Croix, "String taxonomy using learning automata," *IEEE Trans. Syst. Man, Cybern. Part B, Cybern. a Publ. IEEE Syst. Man, Cybern. Soc.*, vol. 27, no. 2, pp. 354-65, Jan. 1997.
[6] B. J. Oommen and E. V de St Croix, "Graph partitioning using learning automata," *IEEE Trans. Comput.*, vol. 45, no. 2, pp. 195-208, Feb. 1996.
[7] A. Tsoularis, C. Kambhampati, and K. Warwick, "Path planning of robots in noisy workspaces using learning automata," in *Proc. of the 1993 IEEE Int. Symp. on Intelligent Control,* pp. 560-564, Aug. 1993.
[8] T. J. Gordon, C. Marsh, and Q. H. Wu, "Stochastic optimal control of active vehicle suspensions using learning automata," *Proc. Inst. Mech. Eng. Part I J. Syst. Control Eng.*, vol. 207, no. 3, pp. 143-152, Aug. 1993.
[9] C. Marsh, T. J. Gordon, and Q. H. Wu, "Application of learning automata to controller design in slow-active automobile suspensions," *Veh. Syst. Dyn.*, vol. 24, no. 8, pp. 597-616, Sept. 1995.
[10] E. Ikonen and K. Najim, "Use of learning automata in distributed fuzzy logic processor training," *IEE Proc.Control Theory Appl.*, vol. 144, no. 3, pp. 255-262, May 1997.
[11] T. Aoki, T. Oka, T. Suzuki, and S. Okuma, "Acquisition of optimal action selection to avoid moving obstacles in autonomous mobile robot," in *Proc., IEEE Int. Conf. on Robotics and Automation*, vol. 3, pp. 2055-2060, Apr. 1996.
[12] M. A. Thathachar and B. Harita, "Learning automata with changing number of actions," *IEEE Trans. Syst. Man, Cybern.-Part A Syst. Humans*, vol. 17, no. 6, pp. 1095-1100, Nov. 1987.
[13] H. Beigy and M. Meybodi, *Intelligent Channel Assignment in Cellular Networks: A Learning Automata Approach*, Amirkabir University of Technology, Technical Report, 2004.
[14] H. Beigy and M. R. Meybodi, "Utilizing distributed learning automata to solve stochastic shortest path problems," *Int. J. Uncertainty, Fuzziness Knowledge-Based Syst.*, vol. 14, no. 5, pp. 591-615, Oct. 2006.
[۱۵] م. ر. ملاخلیلی میبدی و م. ر. میبدی، "یک الگوریتم جدید مبتنی بر آتاماتای یادگیر توزیع‌شده برای حل مسأله کوتاه‌ترین مسیر تصادفی،" *مجموعه مقالات ششمین کنفرانس سیستم‌های هوشمند، کرمان*، ۱۰ ص.، ۵-۴ آذر ۱۳۸۴.
[16] J. Akbari Torkestani and M. R. Meybodi, "Learning automata-based algorithms for solving stochastic minimum spanning tree problem," *Appl. Soft Comput.*, vol. 11, no. 6, pp. 4064-4077, Sep. 2011.
[17] A. Rezvanian and M. R. Meybodi, "A new learning automata-based sampling algorithm for social networks," *Int. J. Commun. Syst.*, vol. 30, no. 5, 21 pp., 25 Mar. 2015.
[۱۸] م. ر. ملاخلیلی میبدی و م. ر. میبدی، "روشی مبتنی بر آتاماتای یادگیر توزیع‌شده برای مدل‌سازی کاربران در محیط‌های فراپیوندی،" *مجموعه مقالات کنگره ملی مهندسی برق، کامپیوتر و فناوری اطلاعات*، ۵ ص.، ۱۹-۱۷ آبان ۱۳۹۱.
[۱۹] م. ر. ملاخلیلی‌میبدی و م. ر. میبدی، "استفاده از آتاماتای یادگیر توزیع‌شده در پیش‌بینی حرکت کاربران در وب،" *مجموعه مقالات سیزدهمین کنفرانس سالانه انجمن کامپیوتر ایران*، ۶ ص.، تهران، ۲۱-۱۹ اسفند ۱۳۸۶.
[۲۰] ع. برادران هاشمی و م. ر. میبدی، "داده‌کاوی استفاده از وب با استفاده از آتاماتای یادگیر توزیع‌شده،" *مجموعه مقالات دوازدهمین کنفرانس سالانه انجمن کامپیوتر ایران*، ۷ ص.، تهران، ۳-۱ اسفند ۱۳۸۵.
[۲۱] ب. اناری، م. ر. میبدی و ز. اناری، "ارائه روشی جدید برای رتبه‌بندی صفحات وب با استفاده از آتاماتای یادگیر توزیع‌شده،" *مجموعه مقالات سیزدهمین کنفرانس سالانه انجمن کامپیوتر ایران*، ۷ ص.، تهران، ۲۱-۱۹ اسفند ۱۳۸۶.
[22] M. R. Mollakhalili Meybodi and M. R. Meybodi, "Extended distributed learning automata: an automata-based framework for solving stochastic graph optimization problems," *Appl. Intell.*, vol. 41, no. 2, pp. 923-940, Oct. 2014.
[۲۳] م. ر. ملاخلیلی میبدی و م. ر. میبدی، "یک الگوریتم جدید مبتنی بر آتاماتای یادگیر توزیع‌شده توسعه‌یافته برای یادگیری پارامتری شبکه بیزی،" *نشریه مهندسی برق و مهندسی کامپیوتر ایران*، سال ۱۲، شماره ۲-ب، صص. ۱۲۶-۱۱۹، زمستان ۱۳۹۳.
[۲۴] م. ر. ملاخلیلی میبدی و م. ر. میبدی، "یادگیری ساختاری شبکه‌های بیزی: یک رهیافت مبتنی بر آتاماتای یادگیر،" *نشریه مهندسی برق و مهندسی کامپیوتر ایران*، سال ۱۴، شماره ۱-ب، صص. ۴۰-۲۷، بهار ۱۳۹۵.
[25] J. B. Kruskal, "On the shortest spanning sub tree of a graph and the traveling salesman problem," *American Mathematical Society*, vol. 7, no. 1, pp. 48-50, Feb. 1956.
[26] M. Elkin, "An unconditional lower bounds on the time-approximation tradeoffs for the distributed minimum spanning tree problem," *SIAM Journal on Computing*, vol. 36, no.2, pp.433-456, 2006.
[27] J. Garay, S. Kutten, and D. Peleg, "A sublinear time distributed algorithm for minimum-wight-spanning tree," *SIAM J. Comput.*, vol. 27, no. 1, pp. 302-326, 1998.
[28] M. Khan and G. Pandurangan, "A fast distributed approximation algorithm for minimum spanning trees," in *Proc. of the 20th Int. Symp. on Distributed Computing, DISC'06*, pp. 355-369, 2006.
[29] H. Beigy and M. R. Meybodi, "Solving stochastic shortest path problem using distributed learning automata," in *Proc. 6th Annual CSI Computer Conf., CSICC'01*, pp. 70-86, 2001.
[30] K. Liu, *Stochastic Online Learning for Network Optimization under Random Unknown Weights*, 18 pp.
[۳۱] م. ر. ملاخلیلی میبدی و م. ر. میبدی، "یک روش جدید مبتنی بر معیارهای آماری توزیع برای تنظیم خودکار نرخ یادگیری آتاماتای یادگیر در محیط‌های پویا،" *نشریه مهندسی برق و مهندسی کامپیوتر ایران*، سال ۱۱، شماره ۱-ب، صص. ۵۱-۴۳، تابستان ۱۳۹۲.
[۳۲] م. قربعلی‌پوردرو و م. ر. میبدی، "یافتن درخت پوشای مینیمم در گراف‌های تصادفی با استفاده از آتاماتاهای یادگیر،" *مجموعه مقالات چهاردهمین کنفرانس سالانه انجمن کامپیوتر ایران*، ۷ ص.، تهران، ۲۱-۲۰ اسفند ۱۳۸۷.
[33] K. R. Hutson and D. R. Shier, "Bounding distributions for the weight of a minimum spanning tree in stochastic networks," *Oper. Res.*, vol. 53, no. 5, pp. 879-886, Oct. 2005.

محمدرضا ملاخلیلی میبدی تحصیلات خود را در مقاطع کارشناسی، کارشناسی ارشد و دکترای مهندسی کامپیوتر به‌ترتیب در سال‌های ۱۳۸۰، ۱۳۸۲ و ۱۳۹۳ در دانشگاه‌های شهید بهشتی، صنعتی امیرکبیر و واحد علوم و تحقیقات تهران به پایان رسانده است. نام‌برده از سال ۱۳۸۰ تاکنون با آزمایشگاه محاسبات نرم دانشگاه صنعتی امیرکبیر همکاری تحقیقاتی داشته است. وی از سال ۱۳۸۲ به عضویت هیأت علمی دانشگاه آزاد اسلامی واحد میبد درآمده و در حال حاضر استادیار گروه مهندسی کامپیوتر این دانشگاه می‌باشد. زمینه‌های تحقیقاتی مورد علاقه ایشان عبارتند از: شبکه‌های کامپیوتری و شبکه‌های نرم‌افزاری، وب، محاسبات نرم و کاربردهای آن، یادگیری والگوریتم‌ها، گراف‌های تصادفی و شبکه‌های پیچیده.

محمدرضا میبدی تحصیلات خود را در مقاطع کارشناسی ارشد و دکتری علوم کامپیوتر به‌ترتیب در سال‌های ۱۳۵۹ و ۱۳۶۲ از دانشگاه اوکلاهمای آمریکا به پایان رسانده است. ایشان در فاصله‌ی سال‌های ۱۳۶۲ تا ۱۳۶۴ استادیار دانشگاه میشیگان غربی و بین سال‌های ۱۳۶۴ الی ۱۳۷۰ دانشیار دانشگاه ایالتی اوهایو در ایالات متحده آمریکا بوده است. پس از بازگشت به ایران به عضویت هیأت علمی دانشگاه صنعتی امیرکبیر تهران در آمده و در حال حاضر استاد دانشکده مهندسی کامپیوتر این دانشگاه و سرپرست آزمایشگاه محاسبات نرم این دانشکده می‌باشد. زمینه‌های تحقیقاتی مورد علاقه ایشان عبارتند از: الگوریتم‌های موازی، پردازش موازی، محاسبات نرم و کاربردهای آن، شبکه‌های کامپیوتری و مهندسی نرم افزار.