یافتن درخت پوشای مینیمم در گرافهای تصادفی با استفاده از اتوماتاهای یادگیر

مهدی قربعلی پور درو محمدرضا میبدی
دانشکده مهندسی کامپیوتر و فناوری اطلاعات
دانشگاه صنعتی امیرکبیر
تهران ایران
ghorbalipoor@aut.ac.ir,mmeybodi@aut.ac.ir

چکیده: در این مقاله یک الگوریتم مبتنی بر اتوماتاهای یادگیر برای یافتن درخت پوشای مینیمم در یک گراف تصادفی (درختی پوشا با کمترین هزینه مورد انتظار) پیشنهاد میشود. فرض بر این است که تابع توزیع وزن یالها از قبل شناخته شده نیست و فقط به نمونه های توزیع یالها دسترسی داریم. هدف الگوریتم یافتن درخت پوشای مینیمم با حداقل تعداد نمونه گیری از یالهای گراف می باشد. در الگوریتم پیشنهادی در هر تکرار اتوماتای یادگیر یالی از گراف را برای نمونه گیری نامزد می کند و سپس بر اساس کمی تجزیه و تحلیل آماری مشخص می شود که آیا از آن یال باید نمونه گرفته شود یا نه. با انتخاب مناسب پارامترهای اتوماتاهای یادگیر، الگوریتم پیشنهادی قادر است که درخت پوشای مینیمم را با احتمالی نزدیک ۱ پیدا کند. به منظور ارزیابی الگوریتم پیشنهادی، تعداد نمونه های گرفته شده توسط الگوریتم با تعداد نمونه های مورد نیاز به روش نمونه گیری استاندارد مقایسه شده است. نتایج آزمایشی نشان داده که تعداد نمونه های گرفته شده توسط الگوریتم پیشنهادی به مراتب کمتر از تعداد نمونه های گرفته شده به روش نمونه گیری استاندارد می باشد.

کلمات کلیدی: گراف تصادفی، درخت پوشای مینیمم، اتوماتاهای یادگیر

۱- مقدمه

درخت پوشای مینیمم یک گراف وزندار ، درخت پوشایی است که بین همه درختهای پوشای آن گراف دارای کمترین وزن باشد. وزن تخصیص داده شده به یک یال می تواند هزینه یال، زمان پیمایش یال، طول و یا غیره بنا به محتوای مورد نظر باشد. درختهای پوشای مینیمم دارای کاربردهای فراوانی در طراحی سیستمهای فیزیکی می باشند. همچنین در کاربردهایی نظیر آنالیز خوشه بندی آماری[1]، شناسایی گفتار و پردازش تصاویر مورد استفاده هستند[1]. در این مقاله می خواهیم الگوریتمی برای یافتن درخت پوشای مینیمم در گرافهای تصادفی ارائه دهیم.

گراف همبند و غیر جهتدار $G =< V, E >$ که درآن V مجموعه رئوس گراف با $|V| = n$ و E مجموعه یالهای گراف با $|E| = m$ را در نظر بگیرید. در صورتیکه وزن هر یال $e \in E$ به احتمال یک برابر ثابت c_e باشد گوییم که گراف G یک گراف قطعی است. در صورتیکه وزن هر یال $e \in E$ به طور قطع معلوم نباشد و دارای وزن احتمالی با متغیر تصادفی X_e و تابع چگالی احتمال f_e باشد، گراف را تصادفی می نامیم. توجه کنید که گراف قطعی حالت خاصی از گراف تصادفی است که در آن برای هر یال $e \in E$ رابطه $f_e(X_e = c_e) = 1$ برقرار است. در گراف تصادفی G معمولا فرض می شود که متغیرهای تصادفی X_{e_i} -ها نسبت به هم مستقل می باشند. در این مقاله نیز ما این فرض را در نظر می گیریم. متغیر تصادفی X_T را نشان دهنده وزن درخت پوشای T از گراف G در نظر می گیریم و آنرا مجموع متغیرهای تصادفی نشان دهنده وزن یالهای موجود در T تعریف می کنیم، یعنی اگر $\{e_1, e_2, ..., e_{n-1}\}$ یالهای درخت پوشای T باشند در این صورت داریم $X_T = \sum_{i=1}^{n-1} X_{e_i}$.

در حالتی که وزن یالها قطعی باشد الگوریتمهای حریصانه ای چون کروسکال(۱۹۵۶)، پریم-دایجکسترا (پریم ۱۹۵۷، دایجکسترا ۱۹۵۹) و سولین(بروفکا) (سولین۱۹۷۷) ارائه شده است که در اصل مشهورترین الگوریتمهای ارائه شده برای حل مساله درخت پوشای مینیمم یعنی یافتن درخت پوشایی چون T از G با کمترین مجموع وزن، می باشند [2]. نسخه موازی شده این الگوریتمها نیز ارائه شده است: پیچیدگی نسخه موازی شده الگوریتم کروسکال با $p = \lceil \log m \rceil$ پردازنده برابر با $O(m)$ می باشد. پیچیدگی

[1] Statistical cluster analysis

الگوریتم پریم با p پردازنده که به صورت hypercube آرایش شده اند برابر است با $\theta(n^2/p)+\theta(n\log p)$ وپیچیدگی نسخه موازی شده الگوریتم سولین با p پردازنده برابر است با $O(\log n(n^2/p+n/p+n+p))$ [2].

حال حالتی را در نظر بگیریدکه وزن یالها احتمالی باشند. فرض کنید که $\xi=\{T_1,T_2,...,T_M\}$ کلاس همه درختهای مختلف G باشد. در صورتیکه G گرافی کامل باشد بنا به فرمول کیلی داریم $|\xi|=n^{n-2}$، در نتیجه در حالت کلی خواهیم داشت $|\xi|\le n^{n-2}$. $T^*\in\xi$ را طوری در نظر می گیریم که برای هر $T\in\xi$ ، مقدار امید ریاضی X_{T^*} کوچکتر مساوی امید ریاضی X_T باشد یعنی $\forall T\in G, T\neq T^*(E(X_{T^*})\le E(X_T))$. هدف ما در این مقاله یافتن T^* می باشد با این فرض که هیچکدام از توابع f_e برای ما شناخته شده نیستند و فقط نمونه ایی از توزیع f_e را داریم . T^* را درخت پوشای مینیمم می گوییم.

قرار می دهیم $\gamma=\min_{T_i\in\xi}\{X_{T_i}\}$. مساله ای که در اکثر مقاله ها مورد بررسی قرار گرفته است یافتن تقریبی از امید ریاضی و تابع چگالی احتمال متغیرتصادفی γ می باشد. توجه کنید که در حالت کلی $\gamma\neq T^*$ و $E(\gamma)\le E(T^*)$ و برای گرافهای قطعی $\gamma=T^*$ می باشد. Kulkarni [3] با استفاده از نتایجی از زنجیرهای مارکف با زمان پیوسته، تابع چگالی احتمال γ را در حالتی که X_{e_i} ها مستقل ازهم و به طور یکسان به صورت نمایی توزیع شده اند را مورد بررسی قرار داد. سپس Jain و Mamer [4] مساله را در حالتی که X_{e_i} ها مستقل ازهم ولی به صورتهای مختلف توزیع شده اند را مورد بررسی قرار دادند. تعدادی از نویسندگان حالتی را در نظر گرفته اند که X_{e_i} ها مستقل و به صورت متغیرهای تصادفی گسسته مشخص شده اند(ولی نه الزاما به صورت یکسان توزیع شده اند). Alexopoulos و Jacobson [5] روشی از پارتیشن بندی را برای محاسبه و به دست آوردن باندی برای مقادیر خاص توزیع γ در گرافهایی با X_{e_i} های مسقل و به طور گسسته توزیع شده ولی نه الزاما به طور یکسان توزیع شده را ارائه کرده اند. همچنین اینها نشان دادند که محاسبه دقیق مقادیر توزیع γ، مساله ای $NP-Hard$ می باشد. Huston و Shier [6] باندهای بالا وپایین مناسبی بر ای γ و $E(\gamma)$ در حالتیکه X_{e_i} ها متغیرهای تصادفی مستقل گسسته باشند به دست آورده اند. همچنین این دو مولف در[1] برای حالتی که X_{e_i} ها مستقل و هرکدام تعداد متناهی مقادیر گسسته دارند، رابطه بین MST های مختلف درحالیکه وزن یالها تغییر می کند را مورد بررسی قرار داده اند.

مسالۀ درخت پوشای مینیمم یکی از مسایل بهینه سازی کلاسیک د ر شبکه های ارتباطی است که به عنوان راه حلی موثر در مسیر یابی های چند پخشی مورد استفاده قرار می گیرد. با توجه به آنکه مشخصه های شبکه های ارتباطی عموما مشخصه هائی تصادفی، غیر قابل پیش بینی و تغیرپذیر با زمان هستند بنابراین به کار گیری درخت پوشای کمینه در موضوعات ارتباطی شبکه های کامپیوتری بایستی مبتنی بر مشخصه های تصادفی شبکه های مذکور بررسی شود. هنگامی که در یک شبکۀ تصادفی وزن هر یک از یالهای گراف یک متغیر تصادفی گسسته با مؤلفه های آماری نامعین باشد تعیین درخت پوشای کمینه کار چندان آسانی نمی باشد. در این شرایط عامل یادگیر بایستی در یک محیط غیر قطعی و نامعین طی یک فرایند یادگیری پویا و تکرار پذیر نسبت به شناسایی مشخصه های آماری محیط و توزیع حاکم بر وزن یالهای گراف اقدام نماید و بر اساس توزیع و متوسط وزنی یالها، درخت پوشای کمینه را شناسایی نماید.

اتوماتای یادگیر و اتوماتای یادگیر توزیع شده برای حل مسائل مختلفی به طور موفقیت آمیز به کار رفته است، مساله های کوتاهترین مسیر بین دو راس [7]، بین یک راس و سایر راسها [8] و بین هر دو راس دلخواه [9] در یک گراف تصادفی ازجمله این کارهاست.

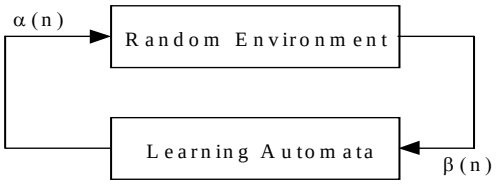
در این مقاله یک الگوریتم مبتنی بر اتوماهاتای یادگیر به منظور یافتن درخت پوشای مینیمم در یک گراف تصادفی که در آن تابع توزیع وزن یالهای گراف تصادفی از قبل شناخته شده نمی باشد و فقط نمونه هایی از توزیع وزنها در اختیار می باشد ارائه شده است. در الگوریتم پیشنهادی در هر تکرار اتوماتای یادگیر یالی از گراف را برای نمونه گیری نامزد می کند و سپس بر اساس کمی تجزیه و تحلیل آماری مشخص می شود که آیا از آن یال باید نمونه گرفته شود یا نه. با انتخاب مناسب پارامترهای اتوماتاهای یادگیرالگوریتم پیشنهادی قادر است که درخت پوشای مینیمم را با احتمالی نزدیک ۱ پیدا کند. به منظور ارزیابی الگوریتم پیشنهادی، تعداد نمونه های گرفته شده توسط الگوریتم پیشنهادی با تعداد نمونه های مورد نیاز به روش نمونه گیری استاندارد مقایسه شده است. نتایج آزمایشی نشان داده که تعداد نمونه های گرفته شده توسط الگوریتم پیشنهادی به مراتب کمتر از تعداد نمونه های گرفته شده به روش نمونه گیری استاندارد می باشد.

ادامۀ مقاله به صورت زیر سازماندهی شده است. در بخش ۲ اتوماتای یادگیر معرفی شده و الگوریتم یادگیری خطی برای این نوع اتوماتا بیان شده است. در بخش ۳ الگوریتم پیشنهادی برای یافتن درخت پوشای مینیمم در گرافهای تصادفی ارائه شده و در

بخش ۴ نتایج حاصل از آزمایشات ارائه گردیده و بخش آخر نیز نتیجه گیری مقاله می باشد.

۲- اتوماتای یادگیر

اتوماتای یادگیر یک مدل انتزاعی است که تعداد محدودی عمل را می تواند انجام دهد. هر عمل انتخاب شده توسط محیطی احتمالی ارزیابی شده و پاسخی به اتوماتای یادگیر داده می شود. اتوماتای یادگیر از این پاسخ استفاده نموده وعمل خود را برای مرحله بعد انتخاب می کند. شکل ۱ ارتباط بین اتوماتای یادگیر و محیط را نشان می دهد.



شکل ۱: ارتباط بین اتوماتای یادگیر و محیط

محیط را می توان توسط سه تایی $E \equiv \{\alpha, \beta, c\}$ نشان داد که در آن $\alpha \equiv \{\alpha_1, \alpha_2, ..., \alpha_r\}$ مجموعه ورودیها، $\beta \equiv \{\beta_1, \beta_2, ..., \beta_m\}$ مجموعه خروجیها و $c \equiv \{c_1, c_2, ..., c_r\}$ مجموعه احتمالهای جریمه می باشد. هر گاه β مجموعه دو عضوی باشد ، محیط از نوع P می باشد. در چنین محیطی $\beta_1 = 1$ به عنوان جریمه و $\beta_2 = 0$ به عنوان پاداش در نظر گرفته می شود. در محیط از نوع Q ، $\beta(n)$ می تواند به طور گسسته یک مقدار از مقادیر محدود در فاصله [0,1] و در محیط از نوع S، $\beta(n)$ متغیر تصادفی در فاصله [0,1] است. c_i احتمال اینکه عمل α_i نتیجه نامطلوب داشته باشد می باشد. در محیط ایستا مقادیر c_i بدون تغییر می مانند ، حال آنکه در محیط غیر ایستا این مقادیر در طی زمان تغییر می کنند. اتوماتاهای یادگیربه دو گروه با ساختار ثابت و با ساختار متغیر تقسیم بندی میگردند. در ادامه به شرح مختصری درباره اتوماتای یادگیر با ساختار متغیر که در این مقاله از آنها استفاده شده است می پردازیم.

اتوماتای یادگیر با ساختار متغیر : یک اتوماتای یادگیر با ساختار متغیر توسط ۴ تائی $\{\alpha, \beta, p, T\}$ نشان داده می شود که در آن $\alpha \equiv \{\alpha_1, \alpha_2, \cdots, \alpha_r\}$ مجموعه عملهای اتوماتا، $\beta \equiv \{\beta_1, \beta_2, \cdots, \beta_m\}$ مجموعه ورودیهای اتوماتا، $p \equiv \{p_1, p_2, \cdots, p_r\}$ بردار احتمال انتخاب هر یک از عملها، و $p(n+1) = T[\alpha(n), \beta(n), p(n)]$ الگوریتم یادگیری می باشد. در این نوع از اتوماتاها، اگر عمل α_i در مرحله n ام انتخاب شود و پاسخ مطلوب از محیط دریافت نماید، احتمال $p_i(n)$ افزایش یافته و سایر احتمالها کاهش می یابند. و برای پاسخ نامطلوب احتمال $p_i(n)$ کاهش یافته و سایر احتمالها افزایش می یابند. در هر حال، تغییرات به گونه ای صورت می گیرد تا حاصل جمع $p_i(n)$ ها همواره ثابت و مساوی یک باقی بماند. الگوریتم زیر یک نمونه از الگوریتمهای یادگیری خطی برای اتوماتای یادگیر با ساختار متغیر می باشد.

الف- پاسخ مطلوب

$$p_i(n+1) = (1-a)\,p_i(n) + a$$
$$p_j(n+1) = (1-a)\,p_j(n) \quad \forall j \quad j \neq i$$

ب- پاسخ نامطلوب

$$p_i(n+1) = (1-b)\,p_i(n)$$
$$p_j(n+1) = (1-b)\,p_j(n) + \frac{b}{r-1} \quad \forall j \quad j \neq i$$

در روابط فوق، پارامتر پاداش و a پارامتر پاداش و b پارامتر جریمه می باشد. با توجه به مقادیر a و b سه حالت را می توان در نظر گرفت. زمانیکه a و b با هم برابر باشند، الگوریتم را L_{RP} [2] می نامیم. زمانیکه b از a خیلی کوچکتر باشد، الگوریتم را $L_{R\varepsilon P}$ [3] می نامیم. زمانیکه b مساوی صفر باشد، الگوریتم را L_{RI} [4] می نامیم.

۳- الگوریتم پیشنهادی

قبل از ارائه الگوریتم ابتدا قضیه آماری زیر را که یکی از ارکان الگوریتم را تشکیل می دهد بیان می کنیم که از [10] گرفته شده است.

قضیه: اگر $\bar{x}$ مقدار میانگین یک نمونه تصادفی به اندازه n از جامعه ای نرمال با واریانس معلوم δ^2 باشد آنگاه

$$p\left(\bar{x} - z_{\alpha/2}.\frac{\delta}{\sqrt{n}} < \mu < \bar{x} + z_{\alpha/2}.\frac{\delta}{\sqrt{n}}\right) = 1-\alpha \;.$$

اگر z متغیر تصادفی نرمال استاندارد باشد، $z_{\alpha/2}$ نقطه ای است که به ازای آن داریم $p\left(z_{\alpha/2} < z\right) = \alpha/2$.

باید توجه نمود که بنا به قضیه حد مرکزی می توان از قضیه بالا برای نمونه های تصادفی از جامعه های غیر نرمال نیز استفاده کرد مشروط بر آنکه تعداد نمونه ها حداقل ۳۰ تا باشد یعنی $n \geq 30$؛

---

[2] Linear Reward Penalty
[3] Linear Reward Epsilon Penalty
[4] Linear Reward Inaction

در این حالت به جای δ نیز می توانیم مقدار انحراف معیار نمونه ای را قرار دهیم.

اگر در رابطه بالا $z_{\alpha/2}$ را به ترتیب برابر با ۱,۹۶ ، ۲,۵۸ و۴ قرار دهیم آنگاه احتمال متناظر به ترتیب برابر خواهد بود با ۰,۹۵۰۰ ، ۰,۹۹۰۲ و ۰,۹۹۹۹ .

فرض کنید که تعدادی یال با وزنهای تصادفی داریم که از هر یال حداقل ۳۰ نمونه گرفته ایم و میانگین نمونه ای و واریانس نمونه ای را حساب کرده ایم. حال اگر یالی بین این یالها پیدا شود که مقدار $\bar{x}+z_{\alpha/2}.\frac{\delta}{\sqrt{n}}$ آن کوچکتر مساوی مقدار $\bar{x}-z_{\alpha/2}.\frac{\delta}{\sqrt{n}}$ بقیه یالها باشد آنگاه حداقل به احتمال $1-\alpha$ می توان گفت که مقدار مورد انتظار وزن آن کوچکتر مساوی مقدار مورد انتظار وزن بقیه یالهاست. با این ایده الگوریتم پیشنهادی خود را شرح می دهیم.

ابتدا یک شبکه از اتوماتاهای یادگیر متناظر با گراف مساله ایجاد می کنیم به این صورت که هر راس را به یک اتوماتای یادگیر مجهز می کنیم که اقدامهای آن متناظر با انتخاب یالهای متصل به آن راس است.

مراحل الگوریتم پیشنهادی به شرح زیر است:

گام ۱. از هر یال گراف تصادفی ۳۰ نمونه می گیریم و میانگین نمونه ای و واریانس نمونه ای آن را حساب می کنیم.

گام ۲. با استفاده از میانگینهای نمونه ای به دست آمده در گام ۱، درخت پوشای مینیمم اولیه را به دست می آوریم که همان درخت پوشای مینیمم فعلی ما می باشد.

گام ۳. یکی از اتوماتاهای یادگیر به تصادف فعال می شود تا یکی از یالهای موجود در درخت پوشای فعلی را برای نمونه گیری کاندیدا کند، این یال را با Ec نشان می دهیم. (توجه کنید که اقدامهایی از اتوماتا که متناظر با انتخاب یالی از گراف است که در درخت پوشای مینیمم فعلی وجود ندارد را غیر فعال در نظر می گیریم.)

گام ۴. Ec از درخت پوشای مینیمم فعلی حذف می شود تا دو زیر درخت ایجاد شود. کلیه یالهایی که می توانند این دو زیردرخت را به هم اتصال دهند را در مجموعه ConnectionSet می ریزیم.

گام ۵. اگر مجموعه ConnectionSet بیش از یک عضو داشته باشد به گام ۶ می رویم در غیر این صورت تنها عضو مجموعه را که Ec است به درخت پوشای مینیمم فعلی اضافه می کنیم و بدون آن که نمونه ای از این یال بگیریم به انتخاب اتوماتای فعال شده پاداش می دهیم و به گام ۷ می رویم.

گام ۶. بین یالهای مجموعه ConnectionSet یالی را می یابیم که دارای کمترین مقدار میانگین نمونه ای باشد که آن را با Emin نشان می دهیم.

حالت اول: اگر Emin همان Ec باشد به انتخاب اتوماتای فعال شده پاداش می دهیم و در صورتی که یالی در مجموعه ConnectionSet موجود باشد که مقدار $\bar{x}-z_{\alpha/2}.\frac{\delta}{\sqrt{n}}$ آن کوچکتر از مقدار $\bar{x}+z_{\alpha/2}.\frac{\delta}{\sqrt{n}}$ یال Ec باشد از یال Ec نمونه می گیریم و مقدار میانگین نمونه ای و واریانس نمونه ای آن را به روز می کنیم. آنگاه به گام ۷ می رویم.

حالت دوم: اگر Emin متفاوت از Ec باشد انتخاب اتوماتای فعال شده را جریمه می کنیم و در صورتی که مقدار $\bar{x}+z_{\alpha/2}.\frac{\delta}{\sqrt{n}}$ یال Emin بزرگتر از مقدار $\bar{x}-z_{\alpha/2}.\frac{\delta}{\sqrt{n}}$ یال Ec باشد از یال Ec نمونه می گیریم و واریانس نمونه ای و میانگین نمونه ای آن را به روز می کنیم، یال Emin را به درخت پوشای مینیمم فعلی اضافه می کنیم و به گام ۷ می رویم.

گام ۷. برای هر اتوماتای یادگیر موجود در شبکه مجموع احتمالات اقدامهای متناظر با انتخاب یالهای موجود در درخت پوشای مینیمم فعلی حساب می شود. سپس این مجموعها را در هم ضرب می کنیم که آن را احتمال انتخاب درخت پوشای مینیمم فعلی می نامیم. اگر این احتمال از مقدار آستانه ای بیشتر باشد و یا تعداد تکرارها از عدد از قبل تعیین شده ای بیشتر باشد الگوریتم خاتمه می یابد در غیر این صورت به گام ۳ می رویم.

در پیاده سازیهایی که ما انجام دادیم مقدار $z_{\alpha/2}$ را برابر با ۱,۹۶، مقدار آستانه احتمال را برابر با ۰,۹۵ و حداکثر تعداد تکرارهای الگوریتم را ۵۰۰۰۰ در نظر گرفتیم. توجه داشته باشید که هر چه مقدار a (پارامتر پاداش اتوماتا) کوچکتر یا $z_{\alpha/2}$ را بزرگتر در نظر بگیریم دقت همگرایی الگوریتم بالاتر می رود ولی در عوض نیاز به نمونه گیری بیشتری داریم.

۴-نتایج آزمایشی

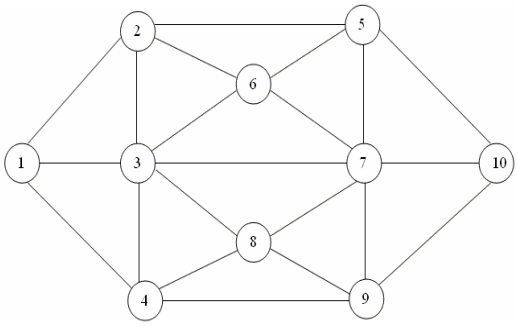
ابتدا به قضیه زیر توجه کنید[11].

قضیه: فرض کنید $x_1, x_2, ..., x_n$ دنباله ای از متغیر های تصادفی مستقل با توزیع یکسان باشند، بگونه ای که $E(x_i) = \mu$ و $Var(x_i) = \sigma^2$. با فرض آنکه متوسط μ با اطمینان $1-\delta$ در بازۀ $\bar{x} \pm \frac{\sigma}{\sqrt{n\delta}}$ قرار داشته باشد. آنگاه بازاء هر مقدار دلخواه وبه قدر کافی کوچک برای ε، وجود دارد n_0ای بگونه ای که بازاء تمامی مقادیر $n \geq n_0$ خواهیم داشت:

$$\forall \varepsilon \; \exists n_0 \; \forall n \geq n_0 \qquad p\{|\bar{x}_n - \mu| \prec \varepsilon\} \succ 1-\delta$$

برای بدست آوردن تعداد نمونه گیری مورد نیاز از هر یال گراف تصادفی G، طوری که بتوان با احتمال $1-\delta$ تضمین کرد که انحراف میانگین نمونه های اخذ شده از یال گراف از مقدار مورد انتظار وزن آن یال کمتر از ε باشد (ε خطای نمونه گیری خوانده می شود) می توانیم از قضیه بالا استفاده کنیم. این روش را نمونه گیری استاندارد می نامیم.

گراف تصادفی که برای آزمایش در نظر گرفته ایم از منبع [7] اقتباس شده و نمودار آن در شکل ۲ و تابع توزیع احتمال وزن یالهای گراف در جدول ۳ آمده است. این گراف از این جهت که مقدار مورد انتظار(امید ریاضی) وزن یالهای آن بسیار نزدیک به هم هستند که کار شناسایی درخت پوشای مینیمم را دشوار می کند انتخاب شده است.



شکل ۲: گراف تصادفی G

ابتدا به کمک روش نمونه گیری استاندارد حداقل تعداد نمونه های مورد نیاز از هریال، به گونه ای که متوسط نمونه های اخذ شده با احتمال (اطمینان) $1-\delta$در بازۀ $(\mu-\varepsilon, \mu+\varepsilon)$ قرار داشته باشد، تعیین می گردد. سپس تعداد نمونه گیریهای مورد نیاز در این روش با تعداد نمونه های اخذ شده توسط الگوریتم پیشنهادی مقایسه می گردد و نشان داده می شود که تعداد نمونه گیری های انجام گرفته توسط الگوریتم پیشنهادی به مراتب کمتر از تعداد نمونه گیریهای انجام شده به روش استاندارد می باشد.

در تمامی آزمایشات مقدار $\varepsilon = 0.001$در نظر گرفته شده است. بازۀ اطمینان بین ۵۰٪ تا ۹۹٪ تغییر داده شده و برای هر مقدار تعداد نمونه های مورد نیاز از یالها تعیین گردیده است. به ازای هر کدام از مقادیر اطمینان مجموع کل نمونه های گرفته شده از گراف محاسبه شده است. جدول ۱ این اطلاعات را نشان می دهد. این جدول تعداد نمونه های مورد نیاز از گراف را در نمونه گیری استاندارد نشان می دهد.

جدول ۱: متوسط تعداد کل نمونه ها در نمونه گیری استاندارد

Confidence Interval	Total Number of Samples
50	5105
55	5065
60	5398
65	5304
70	5747
75	5522
80	5595
85	6050
90	6844
92.5	7062
95	7632
97.5	9181
99	12797

در الگوریتم پیشنهادی برای کلیه اتوماتا های یادگیر موجود در شبکه مقدار پارامتر پاداش را برابر در نظر گرفتیم و از الگوریتم یادگیری L_{R-I} استفاده کردیم. به ازای مقادیر مختلف پاداش، الگوریتم پیشنهادی را ۱۰۰ بار اجرا کردیم و تعداد متوسط کل نمونه های گرفته شده از گراف (AvgGraphSamples) ، تعداد متوسط نمونه های گرفته شده از درخت پوشایی که الگوریتم به آن همگرا شده (AvgTreeSamples) و تعداد دفعات همگرا شده به درخت پوشای مینیمم (ConvRuns) را در جدول ۲ آورده ایم. الگوریتم زمانی خاتمه می یابد که احتمال انتخاب درخت پوشای انتخاب شده در یک تکرار به بیش از ۰,۹۵ برسد.

جدول ۲ : متوسط تعداد نمونه های گرفته شده و درصد همگرایی در الگوریتم پیشنهادی

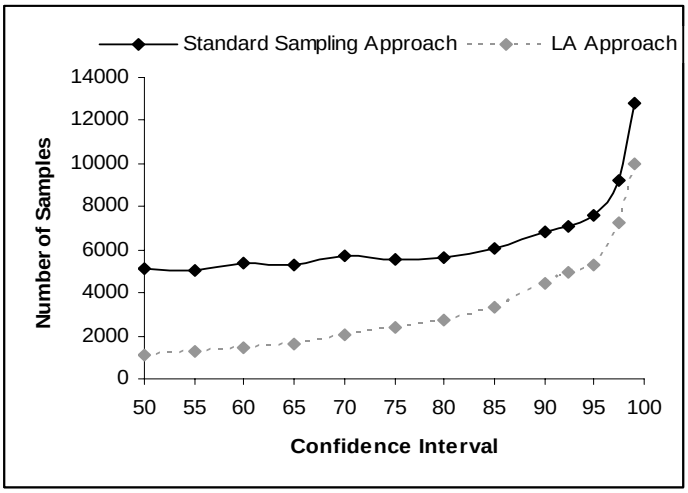
Learning Parameter	AvgGraph Samples	AvgTree Samples	ConvRuns
0.002	12103	5869	100
0.003	9961	4758	99
0.004	8486	4019	98
0.006	6664	3141	97
0.008	5261	2384	95

0.01	4693	2128	90
0.02	3061	1312	81
0.04	2101	847	72
0.06	1728	692	66
0.08	1503	620	64
0.10	1394	557	58
0.20	997	415	50
0.30	894	357	44
0.40	831	324	43
0.50	807	338	41

نمودار ۱ تعداد نمونه های مورد نیاز الگوریتم پیشنهادی را در مقایسه با تعداد نمونه های مورد نیاز در روش نمونه گیری استاندارد نشان می دهد. این مقایسه حاکی از آن است که تعداد نمونه گیری های انجام گرفته توسط الگوریتم پیشنهادی به مراتب کمتر از تعداد نمونه گبریهای انجام شده به روش استاندارد می باشد.

جدول ۳ : تابع توزیع احتمال وزن یالهای گراف G

Edge	Length				Probability			
(1, 2)	3.00	5.30	7.40	9.40	0.20	0.20	0.30	0.30
(1, 3)	3.50	6.20	7.90	8.50	0.30	0.30	0.20	0.20
(1, 4)	4.20	16.10	6.90	8.90	0.20	0.30	0.20	0.30
(2, 3)	9.50	2.30	3.60	4.50	0.20	0.20	0.30	0.30
(2, 5)	2.60	4.10	5.50	9.00	0.20	0.20	0.40	0.20
(2, 6)	5.80	7.00	8.50	9.60	0.30	0.30	0.20	0.20
(3, 4)	2.10	3.20	4.50	6.80	0.20	0.20	0.30	0.30
(3, 6)	6.80	7.70	8.50	9.60	0.40	0.10	0.10	0.40
(3 ,7)	6.50	7.20	8.30	9.40	0.50	0.20	0.20	0.10
(3, 8)	5.90	7.80	8.60	9.90	0.40	0.30	0.10	0.20
(4, 8)	7.00	8.00	8.80	9.40	0.40	0.20	0.20	0.20
(4, 9)	1.10	2.20	3.50	4.30	0.20	0.30	0.40	0.10
(5, 6)	0.60	1.50	3.90	5.80	0.20	0.20	0.30	0.30
(5, 7)	3.200	4.80	6.70	8.20	0.20	0.20	0.30	0.30
(5, 10)	6.30	7.80	8.40	9.10	0.20	0.20	0.40	0.20
(6, 7)	2.10	4.80	6.60	7.50	0.20	0.40	0.20	0.20
(7, 8)	1.60	2.80	5.20	6.00	0.20	0.30	0.30	0.20
(7, 9)	3.50	4.00	5.00	7.70	0.20	0.30	0.30	0.20
(7, 10)	1.60	3.40	8.200	9.30	0.20	0.20	0.20	0.40
(8, 9)	1.70	4.90	6.50	7.80	0.10	0.20	0.40	0.30
(9, 10)	4.60	6.40	7.600	8.90	0.40	0.10	0.20	0.30



نمودار ۱ : مقایسه بین تعداد نمونه های گرفته شده توسط الگویتم پیشنهادی و روش نمونه گیری استاندارد

۵- نتیجه گیری

در این مقاله ابتـدا مـساله درخـت پوشـای مینـیمم در گرافهـای تصادفی شرح داده شد و سپس یک الگـوریتم مبتنـی بـر اتوماتـای یادگیر برای یافتن درخت پوشای مینـیمم در یـک گـراف تـصادفی ارائه شد. به منظور ارزیابی الگوریتم پیشنهادی، تعداد نمونـه هـای گرفته شده توسط این الگوریتم با تعداد نمونـه هـای موردنیـاز بـه روش نمونه گیری استاندارد مقایسه گردیـد و نـشان داده شـد کـه تعداد نمونه های گرفته شده توسط الگوریتم پیشنهادی به مراتب از تعداد نمونه های گرفته شده به روش نمونه گیری اسـتاندارد کمتـر است.

مراجع

[1] K. Hutson and D. Shier, “Minimum spanning trees in networks with varying edge weights”, Annals of Operations Research, Vol. 146, No. 1, pp. 3-18, 2006.

[2] H. Ahrabian and A. Nowzari-Dalini, "Parallel Algorithms for Minimum Spanning Tree Problem", Intern. J. Computer Math., Vol. 79, No. 4, pp. 441–448, 2002.

[3] V. G. Kulkarni, “Minimum Spanning Trees in Undirected Networks with Exponentially Distributed Arc Weights”, Networks, Vol. 16, pp. 111-124, 1986.

[4] Jain, and J. W. Mamer, “Approximations for the Random Minimal Spanning Tree with Application to Network Provisioning”, Operations Research, Vol. 36, pp. 575–584, 1988.

[5] Alexopoulos and J. Jacobson, “State Space Partition Algorithms for Stochastic Systems with Applications to Minimum Spanning Trees”, Networks, Vol. 35, pp. 118-138, 2000.

[6] K. Hutson and D. Shier, "Bounding Distributions for the Weight of a Minimum Spanning Tree in Stochastic Networks", Operations Research, Vol. 53, pp. 879-886, 2005.

[7] H. Beigy and M. R. Meybodi, “Utilizing Distributed Learning Automata to Solve Stochastic Shortest Path Problems”, International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, Vol. 14, No. 5, pp. 591-615, 2006.

[8] S. Misra and B. G. Oommen, "Dynamic Algorithms for the Shortest Path Routing Problem: Learning Automata-based Solutions", IEEE Transactions on Systems, Man and Cybernetics, Vol. SMC-35(B), pp. 1179-1192, June 2005.

[9] S. Misra and B. J. Oommen, "An Efficient Dynamic Algorithm for Maintaining All-Pairs Shortest Paths in Stochastic Networks", IEEE Transactions on Computers, Vol. 55, No. 6, June 2006.

[10] J. E. Freund, "Mathematical Statistics", Fifth Edition, Prentice-Hall, 1992.

[11] Papoulis, "Probability, Random Variables, and Stochastic Processes", Third Edition, New York, McGraw-Hill, 1991.