

طراحی الگوریتمی مبتنی بر اتوماتاهای یادگیر برای حل مسئله کوله پشتی تصادفی

افشین مهرابی^۱، محمد رضا میبدی^۲

چکیده

در این مقاله یک الگوریتم مبتنی بر اتوماتاهای یادگیر برای حل مسئله کوله پشتی تصادفی، پیشنهاد میگردد. در این الگوریتم، شبکه‌ای از اتوماتاهای یادگیر که تشکیل یک حلقه را میدهند برای حل مسئله همکاری می‌نمایند. در این الگوریتم به تعداد آیتمها، اتوماتای یادگیر در نظر گرفته شده است که هر اتوماتا متناظر با یک کالا خواهد بود. هر اتوماتا انتخاب یا عدم انتخاب کالای متناظر، برای قرار گرفتن در کوله پشتی را مشخص میکند. نتایج شبیه سازی نشان داده است که الگوریتم پیشنهادی در مقایسه با الگوریتمهای موجود از کارایی بالاتر و هوشمندی بیشتری برخوردار است. نتایج شبیه سازیها همچنین نشان داده است که الگوریتم پیشنهادی در مسائل با اندازه های متفاوت دارای سرعت و درصد همگرایی بالایی می باشد.

کلمات کلیدی

مسئله کوله پشتی تصادفی، اتوماتاهای یادگیر، مسائل مشکل

Learning automata based algorithms for solving the stochastic knapsack problem

Afshin mehrabi; mohamad reza meybodi

ABSTRACT

In this article a learning automata based algorithm is proposed to solve the stochastic the stochastic knapsack problem with random weights and values. This algorithm involves a network of automata which cooperate to solve the problem.

Each item is represented by a learning automaton which determines whether that item is chosen to be placed in the knapsack. Simulation results show that in comparison to existing algorithms, this one is more efficient and intelligent. The proposed algorithms also execute quicker with a high percentage of convergence in dealing with various problems.

KEYWORDS

Stochastic knapsack problem, learning automata, NP complete problems

^۱ کارشناسی ارشد کامپیوتر - نرم افزار، دانشگاه آزاد اسلامی واحد قزوین، afshin.mehrabi@gmail.com

^۲ عضو هیات علمی دانشگاه صنعتی امیر کبیر دانشکده مهندسی کامپیوتر و فناوری، mmeybodi@aut.ac.ir

مسئله کولهپشتی تصادفی یکی از مسائل بهینه سازی است که کاربردهای فراوانی از قبیل تخصیص منابع، برنامه ریزی بودجه، تخصیص سهام و بارگیری محموله ها دارد. تا به اینجا مسائل کوله پشتی به این گونه مطرح بودند که در آنها لیست آیتمها مشخص، و ارزش و وزن مربوط به هر آیتم نیز از قبل تعیین شده بود. اما در اینجا تعریف جدیدی از مسئله ارائه میشود، که به آن مسئله کولهپشتی تصادفی میگویند. در این تعریف n شیئی معلوم و یک کوله پشتی موجود هستند که میخواهیم اشیاء را به گونه ای انتخاب کنیم که اولاً از ظرفیت کولهپشتی بیشتر نشود و ثانیاً بالاترین ارزش را در مجموع کوله پشتی داشته باشیم. در مسئله کولهپشتی تصادفی دیگر مقادیر یکی یا هردو پارامتر ارزش و وزن برای آیتمها از پیش تعیین شده و مشخص نمیشد در این مسئله ارزش و وزن آیتمها نامشخص است و زمانی میتوان به ارزش و وزن آیتم پی برد که آن آیتم درون کوله پشتی قرار بگیرد، [۳، ۲، ۷] در ضمن ظرفیت کوله پشتی معین و مشخص میباشد. اگر فرض کنیم که n شیئی موجود است و هر شیئی i دارای وزن W_i و ارزش C_i باشد به طوریکه W_i و C_i می توانند مقادیر تصادفی باشند و ظرفیت وزنی کوله پشتی، به میزان M باشد، در اینجا باید یک زیر مجموعه از اشیاء انتخاب شود به گونه ای که شرایط زیر برقرار شود:

$$\begin{aligned} \max \quad & E\left(\sum_{i=1}^n C_i x_i\right) \\ \text{Subject to} \quad & P\left(\sum_{i=1}^n W_i x_i \leq M\right) \geq 1 - \varepsilon \quad (1) \\ & x_i \in \{0,1\} \quad (i = 1,2,\dots,n) \end{aligned}$$

به دلیل اهمیت مسئله کوله پشتی تصادفی تاکنون الگوریتمهای متعددی برای حل آن ارائه شده است. این الگوریتمها را می توان به دو گروه کلی تقسیم کرد: الگوریتم های دقیق و الگوریتم های تقریبی. با توجه به اینکه مسئله کوله پشتی تصادفی یک مسئله NP-complete میباشد الگوریتمهای دقیق که معمولاً از روشهای شاخهوقید^۱ برنامه نویسی پویا^۲ یا روشهای ترکیبی استفاده میکنند [۴، ۶، ۸] در بدترین حالت دارای پیچیدگی نمایی هستند و برای استفاده در کاربردهای عملی مناسب نمی باشند. به همین دلیل الگوریتمهای تقریبی متعددی برای حل مسئله کوله پشتی ارائه شدهاند. از جمله الگوریتم های تقریبی، الگوریتمهای تکرار شونده^۳ هستند. در این روشها رسیدن به یک پاسخ بهینه تضمین نمی - شود اما در اغلب موارد جوابهای تقریبی قابل قبولی تولید میکنند. از جمله الگوریتمهای تکرار شونده میتوان به الگوریتم ژنتیکی [۱، ۲، ۵، ۹] اشاره کرد.

در این مقاله یک الگوریتم تکرار شونده مبتنی بر اتوماتاهای یادگیر برای حل مسئله کولهپشتی تصادفی پیشنهاد میشود. در این الگوریتم، شبکه ای از اتوماتاهای یادگیر که تشکیل یک حلقه را میدهند برای حل مسئله همکاری می نمایند. در این الگوریتم به تعداد آیتمها، اتوماتای یادگیر در نظر گرفته شده است که هر اتوماتا متناظر با یک کالا خواهد بود. هر اتوماتا انتخاب یا عدم انتخاب کالای متناظر برای قرار گرفتن در کوله پشتی را مشخص میکند. در این شبکه در هر مرحله یک اتوماتای یادگیر فعال شده و سپس یکی از اعمال خود را انتخاب میکنند. اعمال انتخاب شده در محیط اجرا و با توجه به نتیجه عمل، به آنها پاداش و یا جریمه داده میشود و در نتیجه بردار احتمالات اتوماتاهای یادگیر بروز میشود و در خاتمه الگوریتم، بهترین راه حل بدست آمده به عنوان جواب نهایی انتخاب میشود. نتایج شبیه سازی انجام گرفته کارایی الگوریتم پیشنهادی را در مقایسه با دیگر الگوریتمهای ارائه شده را نشان میدهد. ادامه مقاله به این صورت سازماندهی شده است، در بخش ۲ اتوماتاهای یادگیر به صورت اجمالی معرفی میگردد. در بخش ۳، الگوریتم پیشنهادی برای حل مسئله کوله پشتی تصادفی ارائه میشود. در بخش ۴، ارزیابی - های انجام شده را بررسی کرده و در بخش ۵، نتیجه را میتوان مشاهده کرد.

۲ - اتوماتاهای یادگیر

اتوماتای یادگیر یک مدل انتزاعی است که بطور تصادفی یک اقدام از مجموعه متناهی اقدامهای خود را انتخاب کرده و بر محیط اعمال می کند. محیط اقدام انتخاب شده توسط اتوماتای یادگیر را ارزیابی کرده و نتیجه ارزیابی خود را توسط یک سیگنال تقویتی به اتوماتای یادگیر اطلاع می دهد. اتوماتای یادگیر با دریافت سیگنال و با توجه به آخرین اقدام انجام شده، وضعیت داخلی خود را بروز کرده و اقدام بعدی خود را انتخاب می کند. شکل ۱، نحوه ارتباط بین اتوماتای یادگیر و محیط را نشان می دهد.



شکل ۱: ارتباط اتوماتای یادگیر با محیط

محیط را می توان توسط سه تایی $E = \{\alpha, \beta, c\}$ نشان داد که در آن $\alpha = \{\alpha_1, \alpha_2, \dots, \alpha_r\}$ مجموعه ورودیها، $\beta = \{\beta_1, \beta_2, \dots, \beta_r\}$ مجموعه خروجیها و $c = \{c_1, c_2, \dots, c_r\}$ مجموعه احتمالات جریمه می باشد. هرگاه 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، مجموعه β دارای تعداد متناهی عضو می باشد و در محیط از نوع S، تعداد اعضا مجموعه β نامتناهی است. c_i نشان دهنده احتمال نامطلوب بودن سیگنال تقویتی محیط در پاسخ به اقدام α_i می باشد. در یک محیط ایستا^۴ مقادیر c_i ها ثابت هستند، حال آنکه در یک محیط غیر ایستا^۵ این مقادیر در طی زمان تغییر می کنند. بر اساس اینکه تابع بروز رسانی وضعیت اتوماتای یادگیر (که با اطلاع از اقدام انتخاب شده و سیگنال تقویتی β ، وضعیت بعدی اتوماتای یادگیر را محاسبه می کند) ثابت یا متغیر باشد، اتوماتای یادگیر به دو دسته اتوماتای یادگیر با ساختار ثابت و اتوماتای یادگیر با ساختار متغیر تقسیم می گردند. در این مقاله از اتوماتای یادگیر با ساختار متغیر استفاده شده است که در ادامه معرفی می شود. اتوماتای یادگیر با ساختار متغیر توسط چهار تایی $\{\alpha, \beta, p, T\}$ نشان داده می شود که در آن $\alpha = \{\alpha_1, \alpha_2, \dots, \alpha_r\}$ مجموعه اقدامهای اتوماتای یادگیر، $\beta = \{\beta_1, \beta_2, \dots, \beta_r\}$ مجموعه ورودیهای اتوماتای یادگیر، $p = \{p_1, p_2, \dots, p_r\}$ بردار احتمال انتخاب هر یک از اقدامها و $T, p(n+1) = T[\alpha(n), \beta(n), p(n)]$ الگوریتم یادگیری اتوماتای یادگیر می باشد. الگوریتمهای یادگیری متنوعی برای اتوماتای یادگیر ارائه شده است که در ادامه یک الگوریتم یادگیری خطی برای اتوماتای یادگیر بیان می گردد. فرض کنید اتوماتای یادگیر در مرحله n اقدام a_i خود را انتخاب نموده و محیط ارزیابی خود را توسط سیگنال تقویتی $\beta(n)$ به اتوماتای یادگیر اعلام کند. با استفاده از الگوریتم یادگیری خطی، اتوماتای یادگیر بردار احتمال انتخاب اقدامهای خود را مطابق رابطه زیر تنظیم می کند.

الف- پاسخ مطلوب از محیط

$$\begin{aligned} p_i(n+1) &= (1-b)p_i(n) \\ p_j(n+1) &= \frac{b}{r-1} + (1-b)p_j(n) \quad \forall j, j \neq i \end{aligned} \quad (2)$$

ب- پاسخ نامطلوب از محیط

$$\begin{aligned} p_i(n+1) &= p_i(n) + a[1 - p_i(n)] \\ p_j(n+1) &= (1-a)p_j(n) \quad \forall j, j \neq i \end{aligned} \quad (3)$$

که a پارامتر پاداش و b پارامتر جریمه می باشد. اگر a و b با هم برابر باشند، الگوریتم L_{R-P} ، اگر b از a خیلی کوچکتر باشد، الگوریتم L_{R-P} و اگر b صفر باشد، الگوریتم L_{R-I} نام دارد [۱۰].

۳ حل مسئله کوله پشتی تصادفی توسط اتوماتاهای یادگیر

در این بخش الگوریتم پیشنهادی برای حل مسئله کوله پشتی تصادفی ارائه می گردد. در این الگوریتم به تعداد آیتنها، اتوماتای یادگیر در نظر گرفته شده است یعنی هر اتوماتا متناظر با یک کالا می باشد. هر اتوماتا دارای دو عمل است که یک عمل انتخاب کالای متناظر و عمل دیگر عدم انتخاب کالای متناظر می باشد. در ابتدا این دو عمل دارای احتمال برابر و مساوی 0.5 میباشند. اتوماتاها با همدیگر تشکیل یک حلقه را میدهند. در هر زمان فقط یک اتوماتا درون این شبکه فعال می باشد. اولین اتوماتا به صورت تصادفی یکنواخت انتخاب میشود از آن پس مسیر حلقه برخلاف حرکت عقربههای ساعت از چپ به راست طی میشود. زمانی که یک اتوماتا فعال میشود از بین دو عمل خود یکی را بنا به احتمال انتخاب میکند. هنگامی که یک اتوماتا عمل خود را انتخاب میکند اتوماتای همسایه سمت راست این اتوماتا فعال میشود و آن نیز به همین ترتیب یکی از اعمال خود را انتخاب میکند. طی شدن این حلقه تا زمانی که کولهپشتی پر شود و یا اینکه دیگر کالایی باقی نمانده باشد که وزن آن از گنجایش باقی مانده کوله کمتر باشد ادامه پیدا میکند. اعمال انتخاب شده در محیط اجرا میشود و با توجه به نتیجه اعمال، به آنها پاداش و یا جریمه داده میشود. الگوریتم پیشنهاد شده سعی میکند تا بر اساس یادگیری، هوش عملیاتی را بالا برده تا در نهایت از بین آیتنها با ارزشترین

آنها انتخاب شود. الگوریتم پیشنهادی را DLA-SKP^۹ می نامیم.

توجه شود که هر آیتم را فقط و فقط یک بار میتوان انتخاب کرد و درون کوله پشتی قرار داد. در این مسئله ارزش و وزن آیتمها نامشخص است و زمانی میتوان به ارزش و وزن آیتم پی برد که آن آیتم درون کوله پشتی قرار بگیرد در ضمن ظرفیت کوله پشتی مشخص میباشد. اولین آتوماتا به صورت تصادفی یکنواخت انتخاب شده و فعال میشود و یک عمل از مجموعه اعمال خود را به صورت تصادفی و مطابق بردار احتمال $p(n)$ ، انتخاب میکند. به عبارت دیگر در این مرحله آیتم متناظر با آتوماتا یا انتخاب می شود و یا انتخاب نمیشود. در صورتیکه عمل انتخاب کالا، برگزیده شود در صورتیکه وزن آیتم از ظرفیت کوله پشتی بیشتر نشود (بر اساس تجربیات گذشته، امید ریاضی وزن آیتم محاسبه میشود)

آن آیتم درون کوله پشتی قرار میگیرد. با قرار دادن آیتم درون کوله پشتی ارزش آیتم نمایان میشود. در این مرحله از تابع برازش $f(i) = \frac{C_i}{W_i}$ به منظور سنجش ارزش آیتم استفاده شده است و مقدار آن به مقدار برازش کل انتخابهای موجود در کولهپشتی، که آن را با $F(K)$ نشان میدهیم جمع بسته میشود.

$$F(K) = \sum \frac{C_i}{W_i} : i \in \{1, 2, \dots, r\} \quad (۴)$$

از این پس مسیر حلقه برخلاف حرکت عقربههای ساعت از چپ به راست طی میشود. زمانی که یک آتوماتا عمل خود را انتخاب میکند آتوماتای همسایه سمت راست این آتوماتا فعال میشود و آن نیز به همین ترتیب یکی از اعمال خود را انتخاب میکند و تا زمانی که حجم کوله پشتی پر نشده است یا اینکه دیگر کالایی باقی نمانده باشد که وزن آن از گنجایش باقی مانده کوله کمتر باشد این روند تکرار میشود. بعد از پر شدن کولهپشتی، مقدار $F(K)$ با میانگین برازش انتخابهای قبلی آتوماتاها در دفعات قبل که آن را با $AVF(K_n)$ ^{۱۱} نشان میدهیم مقایسه می - شود اگر مقدار $F(K)$ کوچکتر از میزان $AVF(K_n)$ باشد و یا اینکه وزن آیتم های درون کوله از حجم کوله بیشتر شود، به اعمال انتخاب شده توسط آتوماتاهای یادگیر جرمه و در غیر این صورت به اعمال انتخاب شده توسط آتوماتاهای یادگیر پاداش تعلق میگیرد. در پایان این مرحله میزان $AVF(K_n)$ دوباره محاسبه میشود که فرمول آن به قرار زیر است:

$$AVF(K_n) = \frac{\sum_{l=1}^n F(k_l)}{n} \quad (۵)$$

در اینجا n تعداد دفعات قبلی میباشد که کوله پشتی پر شده است و $F(K_l)$ مقدار برازش کل انتخابهای موجود در کولهپشتی در تکرار L ام میباشد. در نهایت اگر تعداد اجرای الگوریتم از یک تعداد از پیش تعیین شده بیشتر شود الگوریتم خاتمه می یابد. در الگوریتم، از حد آستانه $AVF(K_n)$ که در حقیقت میانگین برازش انتخابهای قبلی آتوماتاها میباشد، جهت دادن پاداش و جرمه استفاده شده است، که در آن مقدار برازش کل انتخابهای موجود در کولهپشتی از طریق تابع $F(K)$ ، (معادله شماره ۴) محاسبه شده است.

$$F(K) = \sum \frac{C_i}{W_i} : i \in \{1, 2, \dots, r\}$$

حال اگر این الگوریتم را تحت عنوان FB_DLA_SKP^{۱۲} نام ببریم، می توانیم الگوریتم دیگری تحت عنوان CB_DLA_SKP^{۱۳} طراحی کنیم که در آن از حد آستانه $AVC(K_n)$ ^{۱۴} (معادله شماره ۷) که در حقیقت میانگین ارزش انتخابهای قبلی آتوماتاها میباشد، جهت دادن پاداش و جرمه استفاده شود. در ضمن در این حالت مقدار ارزش کل انتخابهای موجود در کولهپشتی از طریق تابع $C(K)$ ، (معادله شماره ۶) محاسبه میشود.

$$C(K) = \sum C_i : i \in \{1, 2, \dots, r\} \quad (۶)$$

$$AVC(K_n) = \frac{\sum_{l=1}^n C(k_l)}{n} \quad (۷)$$

که در معادله شماره ۷، n تعداد دفعات قبلی میباشد که کوله پشتی پر شده است و $C(K_l)$ مقدار ارزش کل انتخابهای موجود در کوله - پشتی در تکرار L ام میباشد. توجه شود که در هر دو الگوریتم پیشنهادی، انتخاب اولین آتوماتا درون حلقه جهت فعال سازی، به صورت تصادفی انجام میپذیرد که در الگوریتمی مشابه الگوریتمهای ذکر شده، میتوان یک آتوماتای معین و مشخص را جهت فعالسازی انتخاب نمود، و از حالت انتخاب تصادفی خارج کرد که در ادامه در آزمایشها این دو روش را با یکدیگر مقایسه میکنیم.

۴- ارزیابی

الگوریتم پیشنهادی DLA-SKP، بر روی مجموعه آیت‌هایی با مشخصات متفاوت از لحاظ ارزش آیت‌ها، وزن آیت‌ها و حجم کوله‌پشتی آزمایش گردیده است و با الگوریتم ژنتیک که توسط Hiroko Ikenoue و Yasunari Yoshitomi روی مسئله کوله‌پشتی تصادفی ارائه گردیده است [۲]، مقایسه می‌شود.

به منظور ارزیابی از داده‌های که در [۲] ذکر شده است استفاده می‌شود، که در ادامه توضیح مختصری در مورد این داده‌ها، بیان می‌شود. در اینجا آیت‌ها به سه گونه ارائه شده‌اند. در گونه اول تعداد آیت‌ها برابر ۵ می‌باشد و حجم کوله‌پشتی برابر ۳۹ خواهد بود که به آن، گونه ((A)) می‌گوییم، در گونه دوم تعداد آیت‌ها برابر ۸ می‌باشد و حجم کوله‌پشتی برابر ۱۲۱ خواهد بود که به آن، گونه ((B)) می‌گوییم و در گونه سوم، تعداد آیت‌ها برابر ۱۰ می‌باشد و حجم کوله‌پشتی برابر ۱۵۰ خواهد بود که به آن، گونه ((C)) می‌گوییم. در گونه A، میانگین برآزش انتخاب بهینه برابر ۱۴/۰۶ و میانگین ارزش انتخاب بهینه برابر ۱۵۲/۷ می‌باشد. در گونه B، میانگین برآزش انتخاب بهینه برابر ۱۲/۹۷ و میانگین ارزش انتخاب بهینه برابر ۲۶۸ می‌باشد. در گونه C، میانگین برآزش انتخاب بهینه برابر ۱۱/۵۵ و میانگین ارزش انتخاب بهینه برابر ۴۰۴/۵ می‌باشد. مقادیر متغیرها بر اساس تابع توزیع احتمال گسسته که در جداول ۱، ۲ و ۳ نشان داده شده است، بدست می‌آید.

جدول (۱) تابع توزیع احتمال گونه ((A))

items	cost		Weight	
	value	probability	value	probability
Item۱	۳۲, ۵۲, ۶۶	۰.۲, ۰.۴, ۰.۴	۵, ۹, ۱۱	۰.۳, ۰.۲, ۰.۵
Item۲	۱۰, ۲۲, ۳۶	۰.۳, ۰.۵, ۰.۲	۸, ۱۵, ۲۰	۰.۶, ۰.۲, ۰.۲
Item۳	۵, ۱۵, ۲۵	۰.۷, ۰.۲, ۰.۱	۸, ۱۲, ۱۴	۰.۲, ۰.۵, ۰.۳
Item۴	۲۲, ۴۳, ۵۵	۰.۳, ۰.۲, ۰.۵	۱۰, ۱۴, ۱۸	۰.۸, ۰.۱, ۰.۱
Item۵	۴۵, ۵۶, ۷۸	۰.۲, ۰.۷, ۰.۱	۹, ۱۲, ۱۷	۰.۴, ۰.۳, ۰.۳

جدول (۲) تابع توزیع احتمال گونه ((B))

items	cost		weight	
	value	probability	value	probability
Item۱	۳۲, ۴۲, ۵۲	۰.۵, ۰.۲, ۰.۳	۳۹	۱
Item۲	۱۲	۱	۳, ۱۳, ۲۳	۰.۵, ۰.۲, ۰.۳
Item۳	۳۵, ۴۵, ۵۵	۰.۳, ۰.۲, ۰.۵	۶۸	۱
Item۴	۵۱, ۶۱, ۷۱	۰.۳, ۰.۲, ۰.۵	۲۰	۱
Item۵	۷۹, ۸۹, ۹۹	۰.۵, ۰.۲, ۰.۳	۲۱, ۳۱, ۴۱	۰.۳, ۰.۲, ۰.۵
Item۶	۲۲, ۳۲, ۴۲	۰.۵, ۰.۲, ۰.۳	۵, ۱۵, ۲۵	۰.۵, ۰.۲, ۰.۳
Item۷	۴۷	۱	۶۱, ۷۱, ۸۱	۰.۵, ۰.۲, ۰.۳
Item۸	۸۸	۱	۶, ۱۶, ۲۶	۰.۳, ۰.۲, ۰.۵

جدول (۳) تابع توزیع احتمال گونه ((C))

items	cost		weight	
	value	probability	value	probability
Item1	۱۰, ۱۵	۰.۳, ۰.۷	۴۲, ۷۰	۰.۶, ۰.۴
Item2	۱۵, ۲۵	۰.۴, ۰.۶	۳۵, ۵۷	۰.۵, ۰.۵
Item3	۱۲۰, ۱۲۵	۰.۴, ۰.۶	۳۰, ۳۶	۰.۴, ۰.۶
Item4	۲۸, ۳۶	۰.۵, ۰.۵	۳۴, ۳۷	۰.۴, ۰.۶
Item5	۱۲۵, ۱۲۸	۰.۵, ۰.۵	۳۲, ۳۸	۰.۶, ۰.۴
Item6	۱۵, ۵۰	۰.۷, ۰.۳	۵۰, ۶۶	۰.۵, ۰.۵
Item7	۱۲۰, ۱۳۰	۰.۷, ۰.۳	۳۵, ۳۹	۰.۵, ۰.۵
Item8	۲۰, ۳۰	۰.۴, ۰.۶	۳۳, ۵۵	۰.۶, ۰.۴
Item9	۱۴۰, ۱۴۵	۰.۶, ۰.۴	۷۲, ۷۷	۰.۵, ۰.۵
Item10	۱۳۰, ۱۳۵	۰.۵, ۰.۵	۷۰, ۷۳	۰.۴, ۰.۶

برای بررسی عملکرد الگوریتم پیشنهادی و مقایسه آن با الگوریتمهای موجود شش معیار زیر استفاده می شود :

- حاصلضرب احتمالات انتخاب بهینه از بین آیتها که می بایست به سمت یک میل کند.
- میانگین ارزش انتخابهای حاصل از اجرای الگوریتم. (Av-Cost)
- درصد اجراهای همگرا شده که با PC^{15} نشان میدهیم.
- میانگین تعداد تکرار برای اجراهای همگرا شده که با AVI^{16} نشان میدهیم.
- میانگین تعداد نمونهگیریهای انجام شده از کل آیتها که با Av-Sam No نشان می دهیم.
- میانگین تعداد نمونهگیریهای انجام شده از کل آیتها در هر تکرار که با Av-Sam PI¹⁷ نشان می دهیم.

آزمایشاتی جهت بررسی عملکرد الگوریتمهای پیشنهادی طراحی شده است. در این آزمایشات هر کدام از معیارهای معرفی شده مورد بررسی قرار گرفته است. نتایج حاصل از اجرای هر الگوریتم در دیگرام و جداولی که نشاندهنده میزان معیارهای ذکر شده است، نشان داده می شود. با توجه به تعدد حالات ممکن در آزمایشها، ما یک استاندارد ساده برای نامگذاری آنها ارائه کرده ایم. نام هر آزمایش از چهار قسمت که با خط تیره (-) از هم جدا شده اند تشکیل شده است. به ترتیب از سمت چپ به راست، هر بخش از این اسم به معنای زیر میباشد:

- نوع مجموعه آیتهای مورد استفاده در آزمایش (گونه $A=1$ ، گونه $B=2$ و گونه $C=3$)
- نوع الگوریتم از جهت نوع حد آستانه (حد آستانه $FB=AVF(K_n)$ ، حد آستانه $CB=AVC(K_n)$)
- نوع انتخاب اولین اتوماتا جهت فعالسازی (از پیش تعیین شده SS ، به صورت تصادفی RS)
- نوع الگوریتم یادگیر و میزان پارامتر پاداش و جریمه (A, B, C, D)

جدول (۴) میزان پارامتر پاداش و جریمه

شناسه	میزان پارامتر پاداش (a)		میزان پارامتر جریمه (b)
L_{ReP}	A	۰.۰۵	۰.۰۰۵
	B	۰.۰۸	۰.۰۰۲
L_{Ri}	C	۰.۰۵	.
	D	۰.۰۸	.

در ضمن شرط همگرایی در همه این آزمایشها، بزرگتر بودن حاصلضرب احتمالات انتخاب آیتهای انتخاب بهینه، از (۰.۹۵) میباشد. همچنین تعداد تکرار در حالت عادی ۲۵۰۰ میباشد که هر آزمایش را ۱۰ بار اجرا نموده و خروجی الگوریتمهای مختلف را در جداول و نمودارهایی که در ادامه مشاهده میشود لحاظ نموده ایم.

با مقایسه درصد اجراهای هگرا شده در روشهای مختلف میتوان نتیجه گرفت که انتخاب حد آستانه $AVC(K_n)$ بسیار کارایی بالاتری داشته است و تقریباً از لحاظ تعداد نمونه گیریها کارایی خوبی داشته و آن را کاهش داده است و در ضمن میانگین تعداد تکرار جهت همگرایی در این روش بسیار بهتر از روش حد آستانه $AVF(K_n)$ بوده است.

جدول (۵) جزئیات نتایج حاصل از اجرای الگوریتمها بروی مجموعه آیت‌های گونه ((B))

Method	PC	AVI	Av-Sam No	Av-Sam PI
۲-FB-RS-A	۳۰	۷۱۲.۳	۱۰۹۰۰.۲	۴.۳۶
۲-FB-RS-B	۸۰	۲۸۶.۳	۸۹۰۰.۶	۳.۵۶
۲-FB-RS-C	۹۰	۴۷۲.۵	۸۷۰۰.۳	۳.۴۸
۲-FB-RS-D	۵۰	۲۸۷	۱۰۵۵۰.۱	۴.۲۲
۲-FB-SS-A	.	.	۱۱۷۲۵	۴.۶۹
۲-FB-SS-B	۲۰	۳۶۳.۳	۱۱۰۲۵	۴.۴۱
۲-FB-SS-C	۸۰	۳۰۱.۲	۸۹۵۰.۶	۳.۵۸
۲-FB-SS-D	۴۰	۳۴۱.۲	۱۰۷۰۰.۲	۴.۲۸
۲-CB-RS-A	۱۰۰	۵۸۴	۸۵۵۰	۳.۴۲
۲-CB-RS-B	۹۰	۱۶۲	۸۵۷۵	۳.۴۳
۲-CB-RS-C	۹۰	۳۰۱.۲	۸۶۲۵.۳	۳.۴۵
۲-CB-RS-D	۷۰	۱۵۱.۴	۹۰۲۵.۱	۳.۶۱
۲-CB-SS-A	۷۰	۴۴۱.۴	۹۲۵۰.۲	۳.۷۰
۲-CB-SS-B	۷۰	۲۰۸.۲	۹۱۵۰.۸	۳.۶۶
۲-CB-SS-C	۴۰	۲۲۵.۵	۱۰۸۰۰.۴	۴.۳۲
۲-CB-SS-D	۴۰	۱۲۵	۱۰۹۷۵.۱	۴.۳۹

جدول (۶) جزئیات نتایج حاصل از اجرای الگوریتمها بروی مجموعه آیت‌های گونه ((A))

Method	PC	AVI	Av-Sam No	Av-Sam PI
۱-FB-RS-A	۷۰	۱۹۳۶	۶۰۵۰.۲	۲.۴۲
۱-FB-RS-B	۱۰۰	۳۱۷.۸	۵۲۷۵	۲.۱۱
۱-FB-RS-C	۹۰	۳۴۶.۲	۵۷۲۵.۱	۲.۲۹
۱-FB-RS-D	۱۰۰	۳۴۱.۸	۵۵۵۰.۴	۲.۲۲
۱-CB-RS-A	۱۰	۱۶۰.۸	۷۳۲۵.۲	۲.۹۳
۱-CB-RS-B	۱۰۰	۳۱۶.۶	۵۳۲۵.۶	۲.۱۳
۱-CB-RS-C	۸۰	۴۱۳.۲	۵۹۰۰.۴	۲.۳۶
۱-CB-RS-D	۹۰	۲۸۹.۴	۵۶۷۵.۱	۲.۲۷

جدول (۷) جزئیات نتایج حاصل از اجرای الگوریتمها بروی مجموعه آیت‌های گونه ((C))

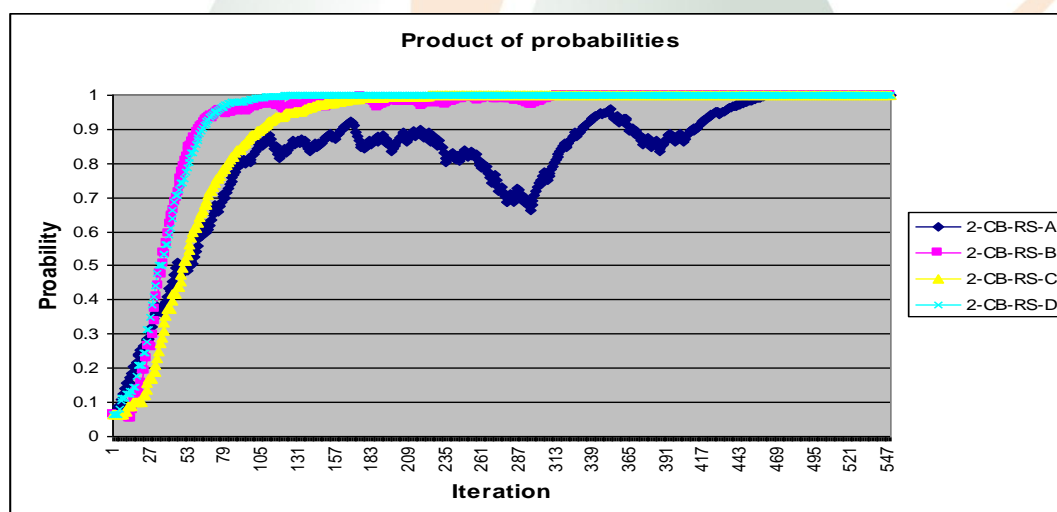
Method	PC	AVI	Av-Sam No	Av-Sam PI
۳-FB-RS-A	۶۰	۲۵۰.۸	۸۵۵۰.۲	۳.۴۲
۳-FB-RS-B	۷۰	۱۹۲	۸۷۷۵.۱	۳.۵۱
۳-FB-RS-C	۸۰	۲۴۰.۴	۸۶۵۰	۳.۴۶
۳-FB-RS-D	۴۰	۱۹۹.۲	۹۶۵۰.۶	۳.۸۶
۳-CB-RS-A	۱۰۰	۲۲۹	۹۰۵۰	۳.۶۲
۳-CB-RS-B	۸۰	۲۷۷	۸۷۰۰.۴	۳.۴۸
۳-CB-RS-C	۱۰۰	۲۵۲.۷	۸۹۵۰.۲	۳.۵۸
۳-CB-RS-D	۷۰	۱۶۸	۸۶۰۰.۶	۳.۴۴

از طرفی با دقت در نتایج بدست آمده میتوان متوجه شد که روش FB در گونه ۲، کارایی لازم را نداشته است چرا که در این گونه، آیتمهای ۴، ۵، ۶ و ۸ دارای بالاترین ارزشها بوده و جزء انتخاب بهینه می باشند در حالی که برازش ((ارزش به وزن)) آیتم ۲ در مواقعی بنا به احتمال از آیتم ۶ بیشتر میشود، لذا روش FB آیتم ۲ را به جای آیتم ۶ انتخاب میکند که این مورد بر خلاف هدف، که همان بیشینه شدن ارزش است، میباشد. پس به گونه‌ای می توان نتیجه گرفت که روش FB برای مواقعی که به وزن به عنوان هزینه، نگاه میشود و کاهش میزان آن، جزء هدف بوده و ارزشمند خواهد بود، مناسب میباشد و نه زمانی که به وزن فقط به عنوان یک مرز و محدودیت نگاه میشود.

از طرفی با مقایسه مقادیر مختلف پارامتر پاداش و جریمه مشاهده میشود که حالت B و C به نسبت بقیه حالات، بهتر نتیجه داده‌اند که این نشان میدهد که پاداش بیشتر و جریمه کمتر کارایی بالاتری دارد. در حالت B و C تعداد تکرار جهت همگرایی از حالت D بیشتر بوده است پس حالت D باعث تسریع در همگرایی می شود اما همین سرعت در همگرایی باعث میشود که در مواقعی، بروی انتخاب بهینه محلی همگرا شود و از انتخاب بهینه سراسری دور بماند. تعداد تکرار جهت همگرایی در حالت A از بقیه حالات بیشتر بوده و از طرف دیگر تعداد نمونه برداری معمولاً در این مورد از بقیه موارد بیشتر بوده است.

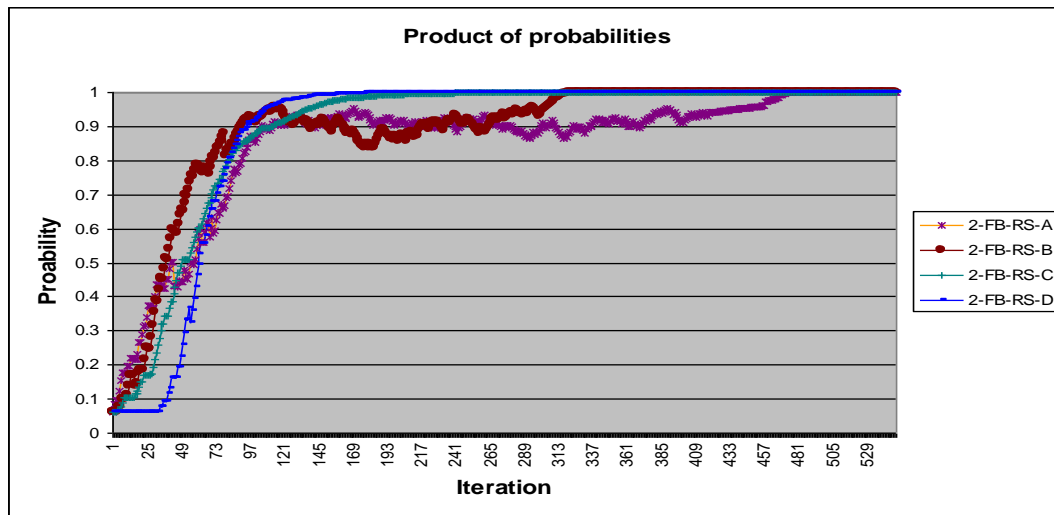
با مقایسه روش های RS و SS میتوان مشاهده کرد که روش RS از کارایی بسیار بالاتری برخوردار بوده است، چرا که روش SS در دام انتخاب بهینه محلی قرار میگیرد در حالی که روش RS به دلیل انتخاب تصادفی، کمتر دچار این مشکل میشود. از این رو نتایج روش SS، در آزمایشات بروی گونه های A و C نشان داده نشده است.

در ادامه حاصلضرب احتمالات انتخاب بهینه از بین آیتمها در طول اجرای الگوریتمهای پیشنهاد شده، مورد بررسی قرار گرفته شده است. مقدار حاصلضرب احتمالات میبایست با تکرارهای بیشتر و متوالی، به سمت یک میل کند. در اینجا به عنوان نمونه این آزمایش بروی آیتم های گونه B نشان داده شده است. همانطور که در شکلهای ۲ و ۳ قابل مشاهده است روش CB به خصوص در مواقعی که میزان پارامتر پاداش و جریمه به صورت B، C و D هستند بسیار بهتر و سریعتر از روش FB همگرا شده است و حاصلضرب احتمالات با سرعت بالایی به سمت یک سوق پیدا می کند.



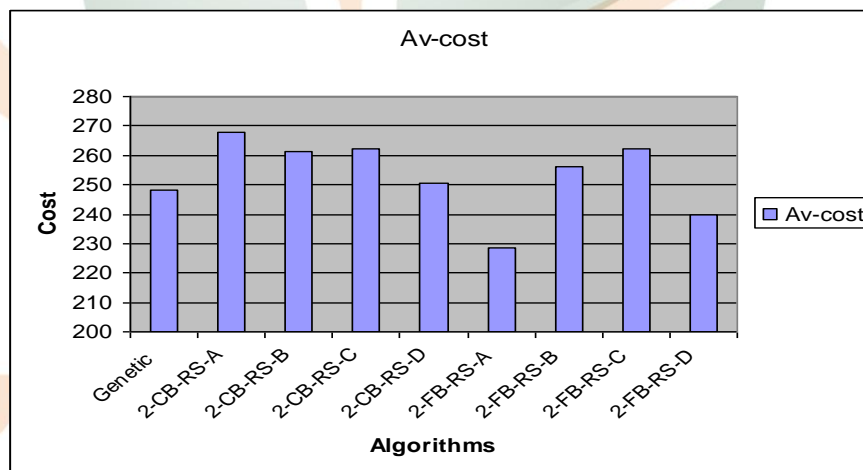
شکل ۲: حاصلضرب احتمالات انتخاب بهینه از بین آیتمها در الگوریتم های پیشنهادی، بروی مجموعه آیتم های گونه ((B))

کنفرانس داده کاوی ایران

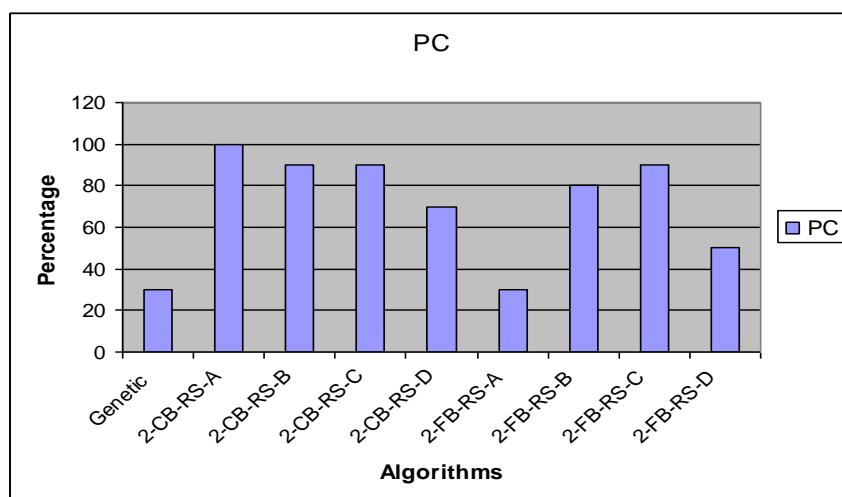


شکل ۳: حاصلضرب احتمالات انتخاب بهینه از بین آیت‌ها در الگوریتم‌های پیشنهادی، بروی مجموعه آیت‌های گونه (B)

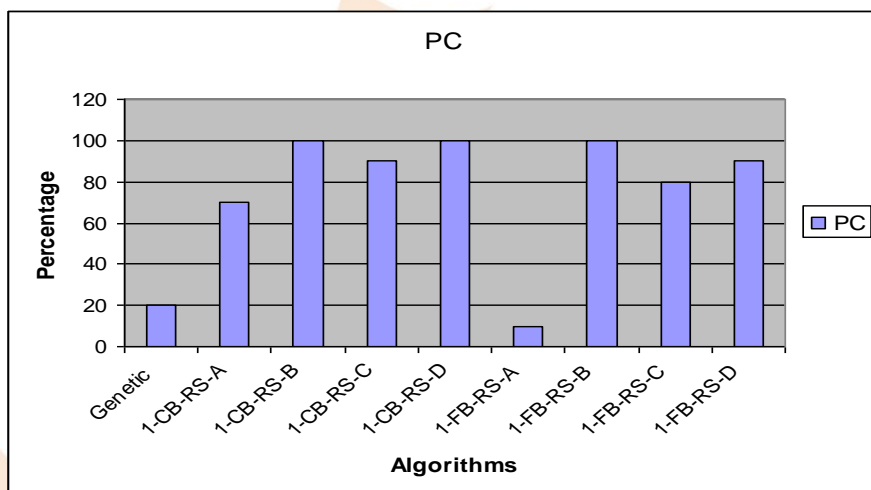
حال جهت بررسی عملکرد الگوریتم پیشنهادی در مقایسه با الگوریتم ژنتیک که توسط Hiroko Ikenoue و Yasunari Yoshitomi روی مسئله کوله پشتی تصادفی ارائه گردیده است [۲]، آزمایشی طراحی شده است. در این آزمایش معیار ارزش نمونه‌های انتخاب شده توسط الگوریتم در اجراهای متوالی و نیز درصد تکرارهایی که به جواب بهینه رسیدند مورد بررسی قرار گرفته است. در این آزمایش، در الگوریتم ژنتیک سائز جمعیت برابر ۵۰۰، تعداد نسل برابر ۱۵۰۰، احتمال Cross over برابر ۰/۶ و احتمال mutation برابر ۰/۱ در نظر گرفته شده است. از آنجایی که در این الگوریتم ژنتیک در هر بار اجرا، یک جواب با بیشترین تکرار در کل نسل‌ها را، به عنوان تنها جواب بهینه معرفی میکند، این الگوریتم بروی داده‌های یاد شده، ۱۰ مرتبه اجرا میشود و نتایج حاصل با نتایج الگوریتم‌های پیشنهاد شده که آنها نیز بروی داده‌های مشابه ۱۰ مرتبه اجرا شده‌اند، مقایسه شده است. نتایج در شکل‌های ۴، ۵، ۶ و ۷ قابل مشاهده است.



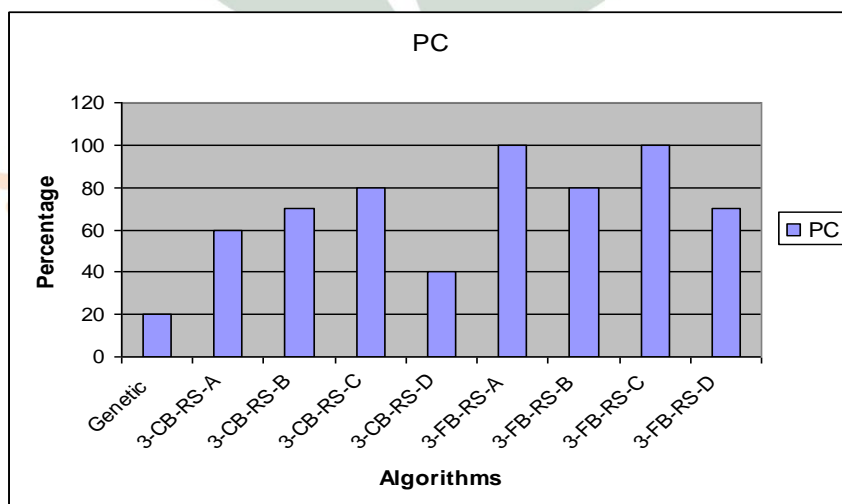
شکل ۴: مقایسه الگوریتم‌های پیشنهاد شده با الگوریتم ژنتیک بر اساس میانگین ارزش آیت‌های انتخاب شده، در آیت‌های گونه (B)



شکل ۵: مقایسه الگوریتمهای پیشنهاد شده با الگوریتم ژنتیک بر اساس درصد تکرارهایی که به جواب بهینه رسیدهاند، در آیتمهای گونه ((B))



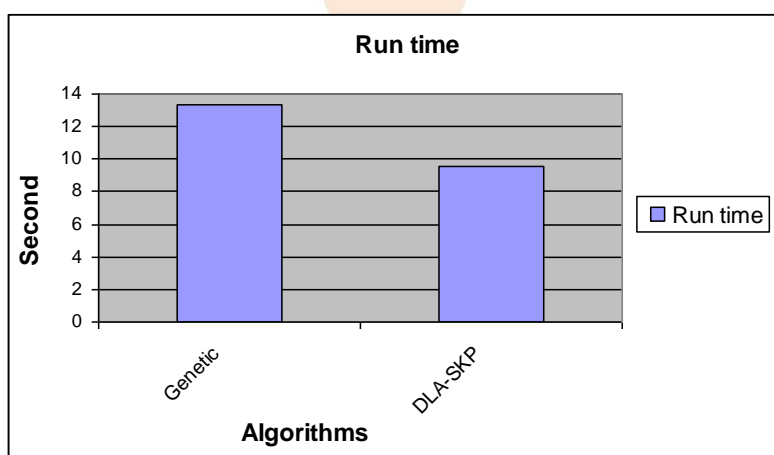
شکل ۶: مقایسه الگوریتمهای پیشنهاد شده با الگوریتم ژنتیک بر اساس درصد تکرارهایی که به جواب بهینه رسیدهاند، در آیتمهای گونه ((A))



شکل ۷: مقایسه الگوریتمهای پیشنهاد شده با الگوریتم ژنتیک بر اساس درصد تکرارهایی که به جواب بهینه رسیدهاند، در آیتمهای گونه ((C))

همانطور که مشاهده میشود از لحاظ ارزش نمونههای انتخاب شده و درصد همگرایی، الگوریتمهای پیشنهاد شده کارایی بالاتری به نسبت الگوریتم ژنتیک داشتهاند. نقطه ضعف الگوریتم ژنتیک در این است که ارزش نمونههای انتخاب شده در طول زمان بهبود نمییابد و کاملاً انتخابها به صورت تصادفی انجام میشود اما در مقابل، در الگوریتم پیشنهاد شده، افزایش هوشمندی در طول زمان کاملاً مشهود است و انتخابهای بعدی نسبت به انتخابهای قبلی، خیلی هوشمندانهتر انتخاب شده و از حالت تصادفی بودن خارج میشود و این برتری الگوریتم DLA_SKP را نشان میدهد.

در ادامه زمان اجرای الگوریتم پیشنهاد شده با زمان اجرای الگوریتم ژنتیک مقایسه شده است. این مقایسه بروی یک کامپیوتر شخصی با مشخصات RAM ۲GB, CPU ۳GHz و سیستم عامل Win-server ۲۰۰۳ انجام شده است. در شکل ۸، نتیجه مقایسه زمان اجرای الگوریتم های پیشنهاد شده با زمان اجرای الگوریتم ژنتیک در گونه C از مجموعه آیتها، قابل مشاهده است. همان طور که مشاهده می شود زمان اجرای الگوریتم های پیشنهاد شده به مراتب از زمان اجرای الگوریتم ژنتیک کمتر می باشد. همانند همین آزمایش بروی گونه های A و B نیز به انجام رسیده است که نتیجه بدست آمده بروی آن دو نیز تقریباً مانند این آزمایش می باشد و در آنها نیز زمان اجرای الگوریتم های پیشنهاد شده از الگوریتم ژنتیک بسیار کمتر میباشد.



شکل ۸: مقایسه الگوریتم پیشنهاد شده با الگوریتم ژنتیک بر اساس زمان اجرا، در آیتها گونه ((C))

۵ - نتیجه گیری

در این مقاله یک الگوریتم جدید مبتنی بر اتوماتاهای یادگیر برای حل مسئله کوله پشتی تصادفی پیشنهاد گردید. در این الگوریتم شبکه‌ای از اتوماتاهای یادگیر برای حل مسأله همکاری می‌نمایند. در این شبکه در هر مرحله یک اتوماتای یادگیر فعال شده و سپس یکی از اعمال خود را انتخاب میکنند. اعمال انتخاب شده در محیط اجرا و با توجه به نتیجه عمل، به آنها پاداش و یا جریمه داده میشود. الگوریتم پیشنهاد شده سعی میکند که با حداقل تعداد نمونهگیری از آیتها در شرایطی که توزیع احتمال ارزش یا وزن آیتها از قبل شناخته شده نیست بهترین آیتها را برای پر کردن کولهپشتی انتخاب نماید. الگوریتم پیشنهاد شده سعی میکند تا بر اساس یادگیری، هوش عملیاتی را بالا برده تا در نهایت با ارزش ترین آیتها انتخاب شود. به منظور بررسی کارایی، الگوریتم پیشنهادی بر روی دادههای استاندارد، آزمایش و نتایج آن بروی شیوههای مختلفی که ارائه شده است نشان داده شده است. نتایج بدست آمده با نتایج حاصل از الگوریتمهای موجود مقایسه شده است. نتایج مقایسه حاکی از هوشمندی بالای الگوریتم پیشنهادی در مقایسه با الگوریتمهای موجود می باشد.

مراجع

- [1] F. Tse Lin, "Solving the Knapsack Problem With Imprecise Weight Coefficients Using Genetic Algorithms", Department of Applied Mathematics, Chinese Culture University, Yangminshan, Taipei 111, Taiwan, Elsevier B.V., 2007.
- [2] Y. Yoshitomi, H. Ikenoue, T. Takeba, S. Tomita, "Genetic Algorithm in Uncertain Environments for Solving Stochastic Programming Problem", Journal of the Operations Research, Society of Japan, Volume 43, No. 2, June 2000.
- [3] D. P. Morton and R. K. Wood, "On a Stochastic Knapsack Problem and Generalizations", Operations Research Department Naval Postgraduate School Monterey, CA 93943, 1998.
- [4] A. J. K. Leyweg and J. d. Papastavrou, "The Dynamic and Stochastic Knapsack Problems", School of Industrial and System Engineering, Atlanta, 1998.

- [5] D. Whitley, "Cellular Genetic Algorithms", Colorado State University, Genetic Algorithms: Proceedings of the Fifth International Conference (GA93), 1993.
 - [6] W. G. Khoo Civilian, "Dynamic-Programming Approaches to Single-and Multi-Stage Stochastic Knapsack Problems For Portfolio Optimization", Master's Thesis, Naval Postgraduate School, Republic of Singapore, Master of Science in Operations Research, 2000.
 - [7] A. J. Kleywegt and J. D. Papastavrou, "The Dynamic and Stochastic Knapsack Problem with Random Sized Items", Oper. Res., Volume 49, No.1, Pages 26–41, 2001.
 - [8] F. M. Gimenez, R. Blanco and C. Azcarate, "Combining Linear Programming and Multi objective Evolutionary Computation for Solving a Type of Stochastic Knapsack Problem", Springer Berlin/Heidelberg, Volume 4403/2007, Pages 531-545, May, 2007.
 - [9] A. S. Anagun and T. Sarac, "Optimization of Performance of Genetic Algorithm for 0-1 Knapsack Problems Using Taguchi Method", Springer Berlin/Heidelberg, Volume 3982/2006, Pages 678-687, May, 2006.
 - [10] K. S. Narendra, M. A. L. Thathachar, "Learning Automata: An Introduction", Prentice Hall, 1989.
-

¹ Branch and cut

² Dynamic programming

³ Iterative

⁴ Stationary

⁵ Non-Stationary

⁶ Linear Reward-Penalty

⁷ Linear Reward epsilon Penalty

⁸ Linear Reward Inaction

⁹ Distributed learning automata algorithm for the stochastic knapsack problem

¹⁰ Fitness

¹¹ Average fitness

¹² Fitness based_ distributed learning automata for stochastic knapsack problem

¹³ Cost based_ distributed learning automata for stochastic knapsack problem

¹⁴ Average cost

¹⁵ percentage of convergence

¹⁶ Average of iteration

¹⁷ Average sampling number per iteration



کنفرانس داده کاوی ایران