The 2nd Lahijan's National Conference on Software Engineering, October 31, Novermer 1, 2012, Lahijan Branch, Islamic Azad University, Lahijan, Iran

دومین کنفرانس ملی مهندسی نرم افزار لاهیجان، ۱۰ و ۱۱ آبان ۱۳۹۱، دانشگاه آزاد اسلامی واحد لاهیجان

روش نوینی برای بهینه سازی در محیط های پیوسته و ایستا مبتنی بر الگوریتم جهش قورباغه

[۱] سمیه رنجکش، دانشکدهٔ برق، رایانه و فناوری اطلاعات دانشگاه آزاد اسلامی قزوین، s.ranjkesh@yahoo.com

[۲] محمدرضا میبدی، عضو هیئت علمی دانشکدهٔ مهندسی کامپیوتر و فناوری اطلاعات، دانشگاه صنعتی امیرکبیر، تهران، mmeybodi@aut.ac.ir

[۳] بهروز معصومی ،عضو هیئت علمی دانشکده برق-رایانه دانشگاه آزاد اسلامی واحد قزوین، masoumi_b@yahoo.com

چکیده

در این مقاله یک الگوریتم جدید که از ترکیب اتوماتای یادگیر و الگوریتم جهش قورباغه‌ها (SFLA) حاصل می‌شود، برای بهینه‌سازی در محیط‌های پیوسته و ایستا، پیشنهاد می‌گردد. در الگوریتم پیشنهادی LA به عنوان تصمیم گیرنده عمل می‌کند و نوع حرکت قورباغه‌ها را با توجه به شرایط دسته در فضای مسئله تعیین می‌کند. الگوریتم پیشنهادی به همراه SFLA استاندارد و دو نسخهٔ سراسری و محلی الگوریتم بهینه سازی دستهٔ ذرات در فضای ۳۰ بُعدی بر روی شش تابع شایستگی استاندارد آزمایش شده اند. نتایج آزمایشات نشان می‌دهند که الگوریتم پیشنهادی از کارایی بسیار مناسبی برخوردار است.

کلمات کلیدی

الگوریتم جهش قورباغه ، بهینه سازی، اتوماتای یادگیر

Frog Leaping Algorithm, Optimization, Learning Automata

۱. مقدمه

بهینه سازی یکی از مهمترین مسائل موجود در علوم مهندسی و ریاضیات بشمار می رود. اهمیت و کاربرد فراوان بهینه سازی باعث شده که دانشمندان بسیاری در زمینه‌های مختلفِ آن به تحقیق بپردازند. بهینه سازی مسئله ای است که همه افراد در زندگی روزمره اشان با آن سر و کار دارند. از برنامه ریزی برای برنامهٔ کاری روزانه گرفته تا انتخاب مسیرِ منزل تا محل کار، همگی به نوعی جزء مسائل بهینه سازی هستند که به راحتی قابل حل می باشند. مفهوم بهینه‌سازی بدین صورت است که در بین پارامترهای یک تابع به دنبال مقادیری باشیم که تابع را کمینه یا بیشینه می‌نمایند. کلیه مقادیر مناسب جهت این امر را، راه حلهای ممکن و بهترین مقدار از این مقادیر را راه حل بهینه می‌نامند. مسائل بهینه‌سازی که هدف، کمینه کردن تابع شایستگی می‌باشد را مسائل کمینه سازی می‌نامند و مسائلی را که هدف آنها بیشینه کردن تابع شایستگی است را مسائل بیشینه سازی می‌گویند. الگوریتم‌های بهینه‌سازی هر دو نوع مسائل بیشینه‌سازی و کمینه‌سازی را پوشش می-دهند.هنگامی که پارامترها و محدودیت های مسائل بهینه سازی کم باشند، می توان آنها را به سادگی حل کرد اما با افزایش محدودیت ها و پارامترها، حل این مسائل بسیار سخت می شود و تبدیل به مسائل NP-hard می شوند.

یکی از معروف ترین روش های حل این مسائل الگوریتم های هوش جمعی می باشند[۱]. تمامی این الگوریتم ها جزء روش‌های Meta-heuristic هستند. معروف ترین الگوریتم های هوش جمعی شامل الگوریتم بهینه سازی دستهٔ ذرات[۲]، الگوریتم بهینه سازی کلونی زنبورها[۳]، الگوریتم جهش قورباغه‌ها[۴]، الگوریتم بهینه سازی کلونی مورچه ها[۵] و الگوریتم دستهٔ ماهی های مصنوعی[۶] می باشد.

الگوریتم جهش قورباغه ها (SFLA) در سال ۲۰۰۳ توسط Eusuff و Lansey ارائه گردید[۴]. این الگوریتم برگرفته شده از الگوریتم بهینه سازی دستهٔ ذرات و الگوریتم های ممتیک می باشد. این الگوریتم رفتار قورباغه ها را در تالاب ها برای یافتن غذا شبیه سازی می کند. در این الگوریتم از مزایای الگوریتم های ممتیک مبتنی بر ژنتیک و رفتارهای اجتماعی ذرات در PSO استفاده شده است. از الگوریتم SFLA در کاربردهای مختلفی از جمله مدیریت پروژه [۷]، بهینه سازی توابع گسسته [۸]، خوشه بندی داده ها[۹]، بهینه سازی توابع پیوسته[۱۰]، توزیع بار اقتصادی[۱۱] و غیره استفاده شده است. یک اتوماتای یادگیر [۱۲]، ماشینی است که می‌تواند تعدادی متناهی عمل را انجام دهد. هر عمل انتخاب شده توسط یک محیط احتمالی ارزیابی می‌شود و نتیجهٔ ارزیابی در قالب سیگنالی مثبت یا منفی به اتوماتای یادگیر داده می‌شود و اتوماتای یادگیر از این پاسخ در انتخاب عمل بعدی استفاده می‌کند و بدین ترتیب به سمت انتخاب عملی که بیشترین پاداش را از محیط می‌گیرد میل می‌کند. به عبارت بهتر، اتوماتا عملی را که بیشترین پاداش را از محیط دریافت می‌کند یاد می‌گیرد. اتوماتای یادگیر برای بهبود قدرت یادگیری بسیاری از الگوریتم‌ها مورد استفاده قرار گرفته است که از آن جمله می‌توان به شبکه‌های عصبی [۱۳]، الگوریتم‌های ژنتیک [۱۴] و حرکت دسته جمعی ذرات [۱۵] اشاره نمود.

در این مقاله یک الگوریتم جدید که از ترکیب اتوماتای یادگیر و الگوریتم جهش قورباغه‌ها (SFLA) حاصل می‌شود، برای بهینه‌سازی در محیط‌های پیوسته و ایستا، پیشنهاد می‌گردد. در الگوریتم پیشنهادی LA به عنوان تصمیم گیرنده عمل می‌کند و نوع حرکت قورباغه‌ها را با توجه به شرایط دسته در فضای مسئله تعیین می‌کند. الگوریتم پیشنهادی به همراه SFLA استاندارد و دو نسخهٔ سراسری و محلی الگوریتم بهینه سازی دستهٔ ذرات در فضای ۳۰ بُعدی بر روی شش تابع شایستگی استاندارد آزمایش شده اند. نتایج آزمایشات نشان می‌دهند که الگوریتم پیشنهادی از کارایی بسیار مناسبی برخوردار است.

ادامهٔ این مقاله به صورت زیر سازماندهی شده است: بخش دوم به معرفی الگوریتم SFLA استاندارد می‌پردازد، در بخش سوم اتوماتای یادگیر به

اختصار معرفی شده است، در بخش چهارم الگوریتم پیشنهادی SFLA-LA شرح داده می‌شود، بخش پنجم نتایج آزمایشات را ارائه می‌کند و بخش پایانی به نتیجه‌گیری می‌پردازد.

۲. الگوریتم بهینه سازی جهش قورباغه ها (SFLA)

الگوریتم جهش قورباغه ها در سال ۲۰۰۳ توسط Eusuff و Lansey ارائه گردید[۴]. این الگوریتم بر اساس قوانین احتمالی، جستجوی تصادفی و جمعیت کار می کند. روند اجرای این الگوریتم برگرفته شده از رفتار قورباغه ها در تالاب ها برای یافتن غذا می باشد. در مسائل بهینه سازی، موقعیت هایی که دارای شایستگی بیشتری هستند از مقدار غذای بیشتری برخوردارند و هدف قورباغه ها یافتن غذای بیشتر است. بدین ترتیب قورباغه ها با اجرای روند SFLA سعی می کنند به غذای بیشتری دست یابند.

در ابتدای اجرای SFLA، یک جمعیت از قورباغه ها وجود دارد که به صورت تصادفی در فضای مسئله پخش می شوند. در SFLA هر قورباغه نشان دهندهٔ یک راه حل شدنی در فضای مسئله می باشد. پس از تعیین موقعیت قورباغه ها، مقدار شایستگی آنها سنجیده می شود. موقعیت قورباغهٔ iام در فضای Dبُعدی برابر $x_i=(x_{i,1}, x_{i,2},\ldots, x_{i,D})$ می باشد و مقدار غذای موجود در موقعیت این قورباغه که همان مقدار شایستگی آن می باشد برابر $f(x_i)$ می باشد.

مطابق با سایر الگوریتم های هوش جمعی، SFLA نیز به صورت تکرارشوند کار می کند و در هر تکرار مراحل زیر انجام می شوند:

۱. ابتدا اعضای جمعیت بر اساس مقدار شایستگی شان، مرتب می شوند (بهترین قورباغه در اندیس یک قرار می گیرد).

۲. قورباغه ها به m گروه که ممپلکس نام دارند تقسیم می شوند. هر ممپلکس شامل n قورباغه خواهد بود. نحوهٔ تقسیم بندی جمعیت به m ممپلکس بدین ترتیب است که اولین قورباغه به اولین ممپلکس، دومین قورباغه به دومین ممپلکس و mامین قورباغه به mامین ممپلکس داده خواهد شد. سپس (m+1)امین قورباغه به ۱امین ممپلکس داده می شود و به همین ترتیب فرآیند دسته بندی ادامه می یابد تا در هر ممپلکس nقورباغه قرار بگیرند.

۳. پس از دسته بندی قورباغه ها در m ممپلکس، هر ممپلکس روند زیر را itr بار تکرار می کند.در هر ممپلکس موقعیت بهترین و بدترین قورباغه از نظر مقدار شایستگی پیدا می شود که به ترتیب برابر با X_b و X_w می باشد. سپس یک موقعیت کاندید با استفاده از روابط زیر بدست می آید:

$$D = Rand.(X_b - X_W) \qquad (۱)$$

$$X'_W = X_W + D\,, \qquad D_{\min} \le D \le D_{\max} \qquad (۲)$$

که در روابط ۱و۲، rand یک تابع تولید عدد تصادفی با توزیع یکنواخت در بازهٔ [0,1] می باشد و برابر با طول گام جابه‌جایی است که مقدار آن در هر یک از ابعاد مسئله باید در محدودهٔ $[D_{min},D_{max}]$ باشد. پس از بدست آوردن موقعیت X'_W ، مقدار شایستگی آن سنجیده می شود و در صورتی که مقدار شایستگی آن بهتر از موقعیت قبلی باشد، X_w به موقعیت جدید خود حرکت می کند در غیر این صورت روابط ۱ و ۲ مجدداً اجرا می‌شوند با این تفاوت که به جای موقعیت X_b، از موقعیت بهترین قورباغه در میان تمام ممپلکس ها یعنی X_g استفاده می شود. اگر پس از این مرحله نیز مقدار شایستگی موقعیت بدست آمده از رابطهٔ۲ بهتر از موقعیت قبلی X_w نبود، این قورباغه مقداردهی اولیه می شود یعنی موقعیت جدید آن در فضای مسئله به صورت تصادفی تعیین می شود. پس از انجام مرحلهٔ ۳ توسط تمام ممپلکس ها (به تعداد itr بار)، شرط پایان الگوریتم سنجیده می شود و در صورتی که این شرط برقرار نشود، مجدداً تمام قورباغه ها بدون در نظر گرفتن اینکه عضو کدام ممپلکس بوده اند در یک دسته قرار می گیرند (shuffling) و مراحل ۱ تا ۳ را اجرا می کنند.

۳. اتوماتای یادگیر

اتوماتای یادگیر[۱۲]، ماشینی است که می‌تواند تعداد متناهی عمل را انجام دهد. هر عمل انتخاب شده توسط یک محیط احتمالی ارزیابی می‌شود و نتیجهٔ ارزیابی در قالب سیگنالی مثبت یا منفی به اتوماتا داده می‌شود و اتوماتا از این پاسخ در انتخاب عمل بعدی تأثیر می‌گیرد. هدف نهایی این است که اتوماتا یاد بگیرد تا از بین اعمال خود بهترین عمل را انتخاب کند. بهترین عمل، عملی است که احتمال دریافت پاداش از محیط را به حداکثر برساند. اتوماتای یادگیر در تعامل با محیط، در شکل ۱ نشان داده شده است.

شکل ۱- ارتباط بین اتوماتای یادگیر و محیط

محیط را می‌توان توسط سه‌تایی $E \equiv \{\alpha, \beta, c\}$ نشان داد که در آن $\alpha \equiv \{\alpha_1, \alpha_2, ..., \alpha_r\}$ مجموعهٔ ورودی‌ها، $\beta \equiv \{\beta_1, \beta_2, ..., \beta_m\}$ مجموعهٔ خروجی‌ها و $c \equiv \{c_1, c_2, ..., c_r\}$ مجموعهٔ احتمال‌های جریمه می‌باشد. هرگاه β مجموعه‌ای دوعضوی باشد، محیط از نوع P می‌باشد. در چنین محیطی $\beta_1 = 1$ به عنوان جریمه و $\beta_2 = 0$ به عنوان پاداش در نظر گرفته می‌شود. در محیط از نوع Q، $\beta(n)$ می‌تواند به طور گسسته یک مقدار از مقادیر موجود در بازهٔ [۰،۱] باشد و در محیط از نوع S، $\beta(n)$ متغیری تصادفی در بازهٔ [۰،۱] است. c_i احتمال اینکه عمل α_i نتیجهٔ نامطلوب داشته باشد می‌باشد. در محیط ایستا مقادیر c_i بدون تغییر می‌مانند، حال آنکه در محیط غیرایستا این مقادیر در طی زمان تغییر می‌کنند.

اتوماتای یادگیر با ساختار متغیر را می‌توان توسط چهارتایی $LA \equiv \{\alpha, \beta, p, T\}$ نشان داد که $\alpha \equiv \{\alpha_1, \alpha_2, ..., \alpha_r\}$ مجموعه عمل-های اتوماتا، $\beta \equiv \{\beta_1, \beta_2, ..., \beta_m\}$ مجموعه ورودی‌های اتوماتا، $p \equiv \{p_1, p_2, ..., p_r\}$ بردار احتمال انتخاب هر یک از اعمال اتوماتای یادگیر و $p(n+1) = T[\alpha(n), \beta(n), p(n)]$ الگوریتم یادگیری می‌باشد. الگوریتم زیر یک نمونه از الگوریتم‌های یادگیری خطی است. فرض کنید عمل α_i در مرحلهٔ nام انتخاب شود:

-پاسخ مطلوب

$$\begin{aligned} p_i(n+1) &= p_i(n) + a[1 - p_i(n)] \\ p_j(n+1) &= (1-a)p_j(n), \forall j \mid j \neq i \end{aligned} \qquad (۳)$$

-پاسخ نامطلوب

$$p_i(n+1) = (1-b)p_i(n)$$
$$p_j(n+1) = (\frac{b}{r-1}) + (1-b)p_j(n), \forall j \mid j \neq i \quad (۴)$$

در روابط ۳ و ۴، a پارامتر پاداش و b پارامتر جریمه می‌باشند. با توجه به مقادیر a و b سه حالت مختلف را می‌توان در نظر گرفت. زمانیکه a و b با هم برابر باشند، الگوریتم را L_{RP} می‌نامیم، زمانیکه b از a خیلی کوچکتر باشد الگوریتم را $L_{R\&P}$ می‌نامیم و زمانیکه b مساوی صفر باشد الگوریتم L_{RI} است.

۴. الگوریتم پیشنهادی

در اینجا یک روش ترکیبی جدید بر اساس الگوریتم SFLA و LA پیشنهاد می‌گردد که در آن، LA به عنوان مغز متفکر الگوریتم SFLA عمل می‌کند. در الگوریتم SFLA هر قورباغه برای انجام هر جابه‌جایی در فضای مسئله بدین ترتیب عمل می کند که ابتدا تلاش می کند به سمت بهترین قورباغهٔ ممپلکس خودش حرکت کند و در صورت عدم موفقیت سعی می‌کند به دنبال بهترین قورباغهٔ موجود در تمام دسته حرکت کند و در صورت شکست در انجام این حرکت به یک نقطهٔ تصادفی در فضای مسئله حرکت می‌کند.

در الگوریتم SFLA، حرکت بدترین قورباغهٔ هر ممپلکس به دنبال بهترین هم‌ممپلکس‌آش باعث انجام جستجو برای یافتن موقعیت‌های بهتر توسط ممپلکس و بالا بردن توانایی جستجوی سراسری جهت یافتن بهترین قله می شود. این در حالی است که حرکت قورباغه ها به دنبال بهترین قورباغهٔ دسته باعث جمع شدن دسته در اطراف بهترین موقعیت پیدا شده توسط الگوریتم و افزایش جستجوی محلی در اطراف آن می گردد. از طرفی حرکت تصادفی قورباغه های شکست خورده در فضای مسئله باعث فرار از بهینه های محلی و در نتنیجه افزایش توانایی جستجوی سراسری می گردد.

الگوریتم SFLAبا انجام سه عمل ذکر شده سعی می‌کند جستجوی سراسری و محلی را در حد قابل قبول انجام دهد اما نمی‌تواند با توجه به شرایط دسته در فضای مسئله یکی از این دو توانایی را تقویت یا تضعیف کند. برای مثال ممکن است شرایط مسئله به گونه ای باشد که انجام یک جستجوی محلی قوی برای رسیدن به نتایج بهتر لازم باشد در حالی که برخی از قورباغه ها به دنبال بهترین قورباغهٔ ممپلکس خودشان بروند و به نوعی جستجوی سراسری را انجام دهند. در واقع در چنین حالتی بخشی از پتانسیل الگوریتم تلف می شود.برای رفع این مسئله و بهبود نتایج الگوریتم یک الگوریتم جدید پیشنهاد می گردد که در آن LA به عنوان تصمیم گیرنده عمل می‌کند و نوع حرکت قورباغه‌ها را با توجه به شرایط دسته در فضای مسئله تعیین می‌کند.

روند الگوریتم پیشنهادی که آنرا SFLA-LA می‌نامیم بدین ترتیب است:در الگوریتم پیشنهادی یک LA وجود دارد که دارای دو عمل می‌باشد. عمل اول شامل دنباله‌روی بدترین قورباغهٔ هر ممپلکس از بهترین قورباغهٔ ممپلکس و در صورت شکست، انجام یک حرکت تصادفی در فضای مسئله است و عمل دوم دنباله‌روی بدترین قورباغهٔ هر ممپلکس از بهترین قورباغهٔ دسته و در صورت شکست انجام یک حرکت تصادفی در فضای مسئله می‌باشد.

در واقع با اجرای رفتار اول سعی می‌شود تا توانایی جستجوی سراسری الگوریتم بالاتر رود و با انجام رفتار دوم الگوریتم سعی می کند توانایی جستجوی محلی خو را تقویت کند. این اتوماتا در هر تکرار از اجرای الگوریتم تعیین می‌کند که قورباغه ها چگونه عمل کنند.

برای هر قورباغه که قصد حرکت در فضای مسئله را دارد (یعنی بدترین قورباغهٔ ممپلکس) اتوماتا با توجه به بردار احتمال خود یک عمل را انتخاب می کند سپس قورباغهٔ مورد نظر این عمل را انجام میدهد.

در صورتی که قورباغه پیشرفت کند عمل انجام شده جایزه می گیرد و در صورتی که به صورت تصادفی در فضای مسئله حرکت کند، عمل انجام شده جریمه می شود. بدین ترتیب با انتخاب و اجرای اعمال و با توجه به جواب های دریافتی، اتواماتا یاد می گیرد که چه عملی در شرایط کنونی دسته بهتر است. بدین ترتیب الگوریتم هوشمند شده و در هر تکرار، بر اساس شرایط مسئله، تصمیم می‌گیرند که چگونه عمل کند.

ساختار اتوماتای یادگیر به کار رفته در الگوریتم پیشنهادی به صورت متغیر است، محیط از نوع P در نظر گرفته شده است و از الگوریتم یادگیری $p(n+1) = T[\alpha(n), \beta(n), p(n)]$ استفاده شده است. مقدار a و b با هم برابرند و الگوریتم از نوع LA_{RP} می‌باشد.

۵. نتایج آزمایشات

آزمایشات بر روی شش تابع استاندارد صورت گرفته است که معمولاً به عنوان معیار سنجش الگوریتم‌های بهینه‌سازی در فضاهایی پیوسته و ایستا مورد استفاده قرار می‌گیرند. توابع استفاده شده به همراه محدودهٔ متغیرهایشان و نتیجهٔ قابل قبول در جدول ۱ نشان داده شده‌اند. شایان ذکر است که مقدار بهینهٔ تمام این توابع برابر با صفر است.

آزمایشات در فضای ۳۰ بُعدی انجام شده است و نتایج حاصل از الگوریتم پیشنهادی با SFLA استاندارد، نسخهٔ سراسری بهینه‌سازی دستهٔ ذرات و نسخهٔ محلی این الگوریتم مقایسه شده است.

در الگوریتم SFLA و الگوریتم پیشنهادی تعداد ممپلکس ها برابر ۱۰ و اندازهٔ جمعیت قورباغه ها برابر ۳۰ در نظر گرفته شده است. طولD می تواند حداکثر مساوی ۰،۱ برابر طول فضای مسئله در هر بُعد باشد. همچنین مقدار پارامتر پارامتر itr برابر ۱۰ در نظر گرفته شده است. مقادیر ذکر شده برای پارامترهای دو الگوریتم SFLA و الگوریتم پیشنهادی بر اساس آزمایشات متعددی که در زمینه های مختلف انجام شده، تعیین گردیده است. . ساختار اتوماتای یادگیر به کار رفته در الگوریتم پیشنهادی به صورت متغیر است.

جدول۱- شش تابع محک استفاده شده در این مقاله

Test Function	Search Space	Global f_{min}	Name
$f_1(x)=\sum_{i=1}^{D} x_i^2$	$[-100,100]^D$	0	Sphere
$f_2(x)=\sum_{i=1}^{D}\lvert x_i\rvert+\prod_{i=1}^{D}\lvert x_i\rvert$	$[-10,10]^D$	0	Schwefel P2.22
$f_3(x)=\frac{\pi}{D}\left\{\begin{array}{l}10\sin^2(\pi y_1)\\ +\sum_{i=1}^{D-1}(y_i-1)^2\left[1+10\sin^2(\pi y_{i+1})\right]\\ +(y_D-1)^2\end{array}\right\}+\sum u(x_i,10,100,4)$ Where $y_i=1+\frac{1}{4}(x_i+1)$, $u(x_i,a,k,m)=\left\{\begin{array}{ll}k(x_i-a)^m, & x_i>a\\ 0, & -a\le x_i\le a\\ k(-x_i-a)^m & x_i<-a\end{array}\right.$	$[-50,50]^D$	0	Generalized Penalized
$f_4(x)=\sum_{i=1}^{D}\left(\lfloor x_i+0.5\rfloor\right)^2$	$[-100,100]^D$	0	Step
$f_5(x)=\sum_{i=1}^{n}\left(\frac{x_i^2}{4000}\right)-\prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$	$[-600,600]^D$	0	Griewank
$f_6(x)=20+e-20e^{-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2}}-e^{\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)}$	$[-32,32]^D$	0	Ackly

محیط از نوع P در نظر گرفته شده است و از الگوریتم یادگیری $p(n+1)=T[\alpha(n),\beta(n),p(n)]$ استفاده شده است. مقدار a و b با هم برابرند و الگوریتم از نوع LA_{RP} می‌باشد. در الگوریتم پیشنهادی ضرایب پاداش و جریمه یعنی a و b برای اتوماتاهای یادگیر برابر ۰،۰۱ در نظر گرفته شده است. آزمایشات ۵۰ بار تکرار شده‌اند و هر بار تا ۲۰۰۰۰۰ ارزیابی شایستگی ادامه یافته اند و متوسط نتایج، بهترین و انحراف‌معیار آنها در جدول۲ در فضای ۳۰بُعدی آورده شده است.

همانطور که نتایج جدول۲ نشان می دهد، نتایج بدست آمده از الگوریتم پیشنهادی در تمام موارد بهتر از الگوریتم SFLA استاندارد می باشد. دلیل بهبود کارایی الگوریتم پیشنهادی نسبت به SFLA استاندارد، کنترل رفتار جستجوی قورباغه ها در طول فرآیند بهینه سازی می باشد. در واقع با کمک LA ، هر یک از ممپلکس ها می‌توانند با توجه به موقعیت قورباغه ها در فضای مسئله و نسبت به پیشرفت فرآیند بهینه سازی، جستجوی محلی و یا جستجوی سراسری را انجام دهند. در واقع وظیفهٔ اصلی LA، برقراری توازن میان توانایی های جستجوی محلی و سراسری و استفادهٔ به موقع از آنها است. با توجه به نتایج بدست آمده از الگوریتم ها، الگوریتم پیشنهادی تنها در دو موردِ توابع گریوانک و آکلی بهترین نتیجه را کسب نکرده است. نتیجهٔ بدست آمده در تابع آکلی تقریباً برابر با الگوریتم GPSO است و اختلاف نتایج بسیار ناچیز است. اما نتیجهٔ بدست آمده در تابع گریوانک اختلاف زیادی با نتیجهٔ بدست آمده از الگوریتم LPSO دارد. دلیل افت کارایی الگوریتم پیشنهادی در این تابع شکل خاص فضای مسئلهٔ این تابع است که فرار از بهینه های محلی را در آن بسیار سخت می کند. الگوریتم LPSO بدلیل جستجوی سراسری بسیار قوی که به واسطهٔ نوع همسایگی حلقه ای ذرات انجام میدهد می تواند از بهینه های محلی این تابع خارج شود.

جدول۲- مقایسهٔ بهترین و متوسط(با انحراف معیار) نتایج حاصل از ۵۰ بار اجرای الگوریتم‌ها بر روی توابع تست در فضای ۳۰ بُعدی

Function	Criteria	GPSO	LPSO	SFLA	LA-LA
Sphere	Best	4.56e-034	1.31e-010	9.42e-021	1.20e-031
	Mean	7.72e-031	3.72e-009	4.44e-018	1.65e-049
	Std-Dev	2.22e-029	3.59e-009	1.72e-017	3.63e-029
Schwefel P2.22	Best	20.00	0.0001	1.94e-014	6.26e-19
	Mean	256.21	82.97	9.59e-011	2.77e-017
	Std-Dev	512.63	96.43	3.00e-010	6.69e-017
Generalized Penalized	Best	1.11e-031	1.64e-009	3.70e-021	1.57e-032
	Mean	0.04	5.39e-005	8.37e-017	2.49e024
	Std-Dev	0.11	0.02	1.94e-016	1.10e-023
Step	Best	0	0	0	0
	Mean	0	0	0	0
	Std-Dev	0	0	0	0
Griewank	Best	2.22e-016	2.00e-007	3.34e-013	8.88e-016
	Mean	0.14	7.43e-006	0.21	0.1750
	Std-Dev	0.14	9.02e-006	0.26	0.1466
Ackly	Best	6.22e-015	8.39e-006	1.00e-010	2.04e-014
	Mean	1.20e-014	4.18e-005	1.14e-007	4.51e-014
	Std-Dev	3.00e-015	2.09e-005	4.47e-007	3.47e-014

۶. نتیجه گیری

در این مقاله یک الگوریتم جدید که از ترکیب اتوماتای یادگیر سلولی و الگوریتم جهش قورباغه‌ها (SFLA) حاصل می‌شود، برای بهینه‌سازی در محیط‌های پیوسته و ایستا، پیشنهاد می‌گردد. در الگوریتم پیشنهادی هر ممپلکس از قورباغه ها در یک سلول از اتوماتای یادگیر سلولی قرار می گیرند. اتوماتای یادگیر موجود در هر سلول به عنوان مغز متفکر

ممپلکس عمل می کند و استراتژی حرکت وجستجو را تعیین می‌کند. الگوریتم پیشنهادی به همراه SFLA استاندارد و دو نسخهٔ سراسری و محلی الگوریتم بهینه سازی دستهٔ ذرات در فضای ۳۰ بُعدی بر روی شش تابع شایستگی استاندارد آزمایش شده اند. نتایج آزمایشات نشان می‌دهند که الگوریتم پیشنهادی از کارایی بسیار مناسبی برخوردار است.

۷. مراجع

[1] Michalewicz. Z and Fogel. D, 2000, "*How to Solve It: Modern Heuristics*", Springer-Verlag، Berlin.

[2] Kennedy. J and Eberhart. R, 1995, "*Particle Swarm Optimization*", in IEEE International Conference on Neural Networks, Vol. 4, pp. 1942-1948, Perth.

[3] Karaboga. D and Akay. B, 2009, "*A Comparative Study of Artificial Bee Colony Algorithm*", in Elsevier Journal on Applied Mathematics and Computation.

[4] Eusuff. M and Lansey. K, 2003, "*Optimization of Water Distribution Network Design Using the Shuffled Frog Leaping Algorithm*", in Journal of Water Resource Plan and Management, Vol. 3, pp. 10-25.

[5] Dorigo. M, Birattari. M, Stutzle. T, 2006, "*Ant Colony Optimization*", in IEEE Computational Intelligent Magazine, Vol. 1, pp. 28-39.

[6] Li. L. X, Shao. Z. J and Qian. J. X, 2002, "*An Optimizing Method Based on Autonomous Animate: Fish Swarm Algorithm*", In Proceeding of System Engineering Theory and Practice, Vol. 11, pp. 32-38.

[7] Elbeltagi. E, Hegezy. T and Grierson. D, 2007, "*A Modified Shuffled Frog-Leaping Optimization Algorithm: Applications to Project Management*", in Journal of Taylor & Francis on Structure and Infrastructure Engineering, Vol. 3, pp. 53-60.

[8] Vkil Baghmisheh. M. T, Madani. K and Navarbaf. A, 2011, "*A Discrete Shuffled Frog Optimization Algorithm*", in journal of Springer on Artificial Intelligence Review.

[9] Amiri. B, Fathian. M and Maroosi. A, 2009, "*Application of Shuffled Frog-Leaping Algorithm on Clustering*", in journal of Springer on The International Journal of Advanced Manufacturing Technology, pp. 199-209.

[10] Li. X, Lou. J, Chen. M. R and Wang. N, 2010, "*An Improved Shuffled Frog-Leaping Algorithm with Extremal Optimisation for Continuous Optimisation*", in journal of Elsevier on Information Sciences.

[11] Chahkandi Nejad. H, Jahani. R and Sarlak.GH. R, 2011, "*Applying Shuffled Frog Leaping Algorithm for Economic Load Dispatch of Power System*", in American Journal of Scientific Research, pp. 82-89.

[12] Narendra. K. S and Thathachar. M. A. L, 1989, "*Learning Automata: An Overview*", Prentice Hall.

[13] Meybodi. M. R, Beigy. H and Taherkhani. M, 2003, "*Cellular Learning Automata and Its Applications*", Journal of Science and Technology, University of Sharif, No. 25, pp. 54-77.

[14] Beigy. H and Meybodi. M. R, 2004, "*A Mathematical Framework for Cellular Learning Automata", Advances on Complex Systems*, Vol. 7, No. 3, pp. 1-25.

[15] Howell. M. N, Gordon. T. J, and Brandao. F. V, 2002, "*Genetic Learning Automata for Function Optimization*", IEEE Transaction on Systems, Man, and Cybernetics-Part B: Cybernetics.

[16] Sheybani. M and Meybodi. M. R, "*PSO-LA: A New Model for Optimization*", In Proc. of CSICC06, 2006.