سیزدهمین کنفرانس ملی انجمن کامپیوتر ایران
جزیره کیش، خلیج فارس، ایران ۱۹ الی ۲۱ اسفند ۱۳۸۶
انجمن کامپیوتر ایران
Computer Society of Iran
دانشگاه صنعتی شریف
دانشکده مهندسی کامپیوتر
مقاله کامل

یک روش بهینه‌سازی ترکیبی (بهینه‌سازی حدی + CLA-EC)

آیدین خاتم نژاد پاکزاد
دانشکده مهندسی کامپیوتر و فناوری اطلاعات
دانشگاه صنعتی امیر کبیر
khatamnejad@gmail.com

محمد رضا میبدی
دانشکده مهندسی کامپیوتر و فناوری اطلاعات
دانشگاه صنعتی امیر کبیر
mmeybodi@aut.ac.ir

چکیده: الگوریتم بهینه‌سازی حدی تعمیم یافته برای مسائلی که بیش از یک نقطه بهینه دارند به خوبی عمل نکرده و و با احتمال بالایی در نقاط بهینه محلی گرفتار می‌شود. با این حال، این الگوریتم این ویژگی مهم را دارد که در صورت قرار گرفتن در اطراف نقطه بهینه، جواب را با دقت بالایی محاسبه می‌کند. از طرف دیگر مطالعات بر روی الگوریتم بهینه سازی CLA-EC نشان می‌دهند که این الگوریتم با وجود جستجوی مناسب دامنه مساله، جواب‌هایی با دقت بالا تولید نمی‌کند. در این مقاله یک الگوریتم بهینه سازی ترکیبی که آن را CLA-EC-EO می‌نامیم و از ترکیب الگوریتم بهینه سازی حدی تعمیم یافته و الگوریتم بهینه سازی CLA-EC حاصل می‌شود پیشنهاد می‌گردد . آزمایش‌ها نشان می‌دهد که الگوریتم پیشنهادی ویژگی‌های مطلوب هر دو الگوریتم پایه خود را دارا می‌باشد.

واژه های کلیدی: بهینه‌سازی حدی تعمیم یافته، CLA-EC، بهینه‌سازی.

۱- مقدمه

الگوریتم‌های مبتنی بر بهینه‌سازی حدی[1] بر خلاف دیگر الگوریتم‌های بهینه‌سازی نظیر الگوریتم ژنتیک که همواره از مجموعه‌ای از جواب‌ها (جمعیت) تشکیل شده‌اند، در هر لحظه تنها یک جواب در اختیار دارند و در هر مرحله این جواب را بهبود می‌بخشند؛ بهبود جواب با انتخاب یکی از اجزای جواب و جایگزین کردن مقدار آن با مقداری جدید انجام می‌شود. برای انتخاب جزء تعویضی، الگوریتم از شایستگی محلی اجزا استفاده می‌کند؛ بنابراین این الگوریتم‌ها علاوه بر تابع شایستگی کلی جواب، به تابعی برای ارزیابی شایستگی اجزای جواب (شایستگی محلی) نیز احتیاج دارند. واضح است که این تابع وابسته به نوع مساله بوده و برای هر مساله باید به صورت جداگانه تعریف و پیاده‌سازی شود.

الگوریتم بهینه‌سازی حدی تعمیم یافته[2]، تعمیم یافته یکی از الگوریتم‌های مشتق شده از الگوریتم بهینه‌سازی حدی به نام الگوریتم بهینه‌سازی حدی با پارامتر t است. این الگوریتم یک الگوریتم ابر اکتشافی است که نیاز الگوریتم بهینه‌سازی حدی به تابع ارزیابی شایستگی محلی اجزا را از بین می‌برد. از ویژگی‌های جالب این الگوریتم می‌توان به سرعت بالای همگرایی، دقت بالای جواب‌ها و تعداد پارامترهای کم آن (این الگوریتم تنها یک پارامتر به نام t برای تنظیم دارد) اشاره کرد. هر چند این الگوریتم جواب‌های بسیار دقیقی برای مسائل تولید می‌کند، احتمال گرفتار شدن آن در نقاط بهینه محلی بسیار بالا است. در واقع این الگوریتم دامنه مساله را به شکل خوبی جستجو نکرده و در نتیجه در اولین نقطه بهینه‌ای که پیدا می‌کند متوقف می‌شود. این مساله باعث شده است که استفاده از این الگوریتم تنها به مجموعه کوچکی از مسائل (مسائلی با تنها یک نقطه بهینه) محدود شود.

در این مقاله برای حل این مشکل پیشنهاد می‌شود که الگوریتم بهینه‌سازی حدی تعمیم یافته در کنار الگوریتم بهینه‌سازی دیگری به نام CLA-EC که اخیرا معرفی شده است قرار گیرد تا در شرایطی که به تنهایی از کارایی بالایی بر خوردار نیست بتواند با کمک گرفتن از CLA-EC عملکرد خود را بهبود بخشد. علت این پیشنهاد این است که بنا بر آزمایش‌های انجام شده بر روی الگوریتم CLA-EC، مشاهده شده است که این الگوریتم بر خلاف الگوریتم بهینه‌سازی حدی تعمیم یافته دامنه مساله را به شکل مطلوبی جستجو می‌کند اما با وجود اینکه محدوده جواب را درست تشخیص می‌دهد، جواب‌های دقیقی تولید نمی‌کند. در واقع نقطه‌های قوت هر کدام از این دو الگوریتم منطبق بر نقطه ضعف‌های الگوریتم دیگر است. بنابراین انتظار می‌رود که در صورت ترکیب این دو الگوریتم که آن را الگوریتم CLA-EC-EO می‌نامیم، بتوان ضعف‌های هر دو الگوریتم را مرتفع نمود.

[1] Extremal Optimization (EO)

[2] Generalized Extremal Optimization (GEO)

ادامه این مقاله بدین صورت سازماندهی شده است: در بخش ۲ بحران‌های خودسازمانده و بهینه‌سازی حدی و در بخش ۳ اتوماتای سلولی، اتوماتای یادگیر ، اتوماتای یادگیر سلولی و مدل CLA-EC به اختصار شرح داده می‌شوند. در بخش ۴ الگوریتم پیشنهادی معرفی می‌شود و دو بخش انتهایی اختصاص به ارایه نتایج آزمایش‌ها و نتیجه‌گیری دارد.

۲- بحران‌های خودسازمانده و بهینه‌سازی حدی

در این بخش ابتدا بحران‌های خود سازمانده و مدل بک-اسنپن به صورت خلاصه معرفی می‌شوند. سپس بهینه‌سازی حدی و بهینه‌سازی حدی با پارامتر t معرفی و نهایتا الگوریتم بهینه‌سازی حدی تعمیم یافته شرح داده می‌شود.

۲-۱ بحران خود سازمانده[1]

بحران خود سازمانده [۱] یکی از ویژگی‌های سیستم‌های طبیعی است که بسیاری از رفتارهای پیچیده این سیستم‌ها را توصیف می‌کند. این مفهوم اولین بار توسط بک[2] برای توصیف رفتار مدل توده شن معرفی شد. در حال حاضر از این مفهوم در شبیه‌سازی و پیش‌بینی زلزله و رانش‌زمین، شبیه‌سازی آتش‌سوزی در جنگل‌ها، بررسی انتشار بیماری‌ها، تغییرات آب و هوا و بررسی بازار خرید و فروش کالاها استفاده می‌شود.

سیستم‌هایی که دارای ویژگی بحران خود سازمانده هستند معمولا از یک حالت پایدار شروع به کار کرده و در طول زمان به یک نقطه بحرانی[3] نزدیک می‌شوند. در این نقطه، کوچکترین تغییری در وضعیت، باعث می‌شود که سیستم با یک رفتار سریع و وسیع که با نام بهمن[4] شناخته می‌شود، خود را به یک حالت پایدار برساند. این سیستم‌ها تا مرز آشفتگی رفته ولی توسط بهمن به حالت عادی بر می‌گردند. تعداد و وسعت بهمن‌ها از قانون توان[5] پیروی می‌کند:

$$y = ax^k \qquad (۱)$$

در این رابطه x وسعت بهمن (با توجه به نوع سیستم می‌تواند واحدهای مختلفی داشته باشد)، a و k مقادیر ثابت که برای هر سیستم به صورت جداگانه تعریف می‌شود و y تعداد دفعات وقوع بهمن است. به عنوان مثال اگر زمین لرزه را به عنوان بهمن در نظر بگیریم، تعداد زمین لرزه‌ها با بزرگی‌های مختلف از این قانون پیروی می‌کند.

۲-۲ مدل بک- اسنپن[6]

مدل بک-اسنپن مدلی برای نمایش تکامل گونه‌های موجودات است که بر اساس مفهوم بحران خود سازمانده ایجاد شده است. این مدل از آرایه‌ای از سلول‌ها تشکیل شده است که در آن هر سلول نماینده یک گونه جانوری است؛ ابتدا و انتهای آرایه سلول‌ها در کنار هم قرار دارند به نحوی که آرایه صورت یک حلقه در می‌آید. در این مدل به هر گونه جانوری یک شایستگی اختصاص داده می‌شود که نشان دهنده شایستگی و مقدار تطابق آن با محیط است؛ این شایستگی به صورت عددی در $[0,1]$ در هر سلول نگهداری می‌شود.

روش کار الگوریتم مدل بک-اسنپن به این صورت است که در ابتدا همه سلول‌ها با عددی اتفاقی مقداردهی می‌شوند. سپس تا توقف الگوریتم در هر مرحله، سلولی که کمترین شایستگی را دارد (در صورتی که چندین سلول با کمترین شایستگی وجود داشتند یکی از آنها به صورت اتفاقی انتخاب می‌شود) به همراه سلول‌های چپ و راست آن مجددا به صورت اتفاقی مقداردهی می‌شوند. توقف الگوریتم می‌تواند به صورت دستی انجام شود و یا با تعریف آستانه‌ای به خود الگوریتم محول شود.

اگر در مرحله جاری، مجموعه‌ای از کوچکترین شایستگی در هر کدام از مراحل قبل را تشکیل دهیم، بزرگترین عضو این مجموعه شایستگی بحرانی نامیده می‌شود. واضح است که شایستگی بحرانی همواره مقدار ثابتی ندارد و در طول اجرا به صورت صعودی تغییر پیدا می‌کند. آزمایش‌هایی که بر روی این مدل انجام شده است نشان می‌دهد که شایستگی بحرانی دارای حدی برابر ۰.۶۶۷ است که با گذشت زمان به سمت آن میل می‌کند.

۲-۳ بهینه‌سازی حدی

بهینه‌سازی حدی [۲] الگوریتمی اکتشافی است که از مدل بک-اسنپن الهام گرفته شده است. همانطور که از نام آن مشخص است این الگوریتم برای حل مسائل بهینه‌سازی استفاده می‌شود. در این الگوریتم سلول‌های مدل بک-اسنپن برابر اجزای جواب مساله قرار داده می‌شوند. جواب در هر مرحله (تکرار حلقه اصلی الگوریتم) با جایگزینی مقدار بدترین سلول (سلول با کمترین شایستگی) با مقداری دیگر بهبود داده می‌شود. در این الگوریتم بر خلاف مدل بک-اسنپن، با تغییر یک سلول (گونه در مدل بک-استپن) سلول‌های همسایه آن تغییر داده نمی‌شوند. بارزترین تفاوت الگوریتم بهینه‌سازی حدی با دیگر الگوریتم‌های اکتشافی نظیر الگوریتم ژنتیک، نیاز الگوریتم به دانستن شایستگی سلول‌ها (شایستگی محلی) علاوه بر شایستگی کلی جواب است. در واقع این ویژگی الگوریتم بهینه‌سازی حدی، هسته مرکزی و عامل کار آن است. الگوریتم بهینه‌سازی حدی برای حل مسائلی نظیر تقسیم گراف

---

[1] Self-Organized Criticality (SOC)
[2] Bak
[3] Critical Point
[4] Avalanche
[5] Power Law
[6] Bak-Sneppen Model

به دو زیر گراف، رنگ آمیزی گراف با سه رنگ و فروشنده دوره‌گرد مورد استفاده قرار گرفته و جواب‌های مطلوبی ارائه داده است.

۴-۲ بهینه‌سازی حدی با پارامتر τ

هرچند الگوریتم بهینه‌سازی حدی جواب‌های مطلوبی برای مسائل به کار گرفته شده ارائه می‌دهد، با این حال احتمال گرفتار شدن آن در نقاط بهینه محلی و متوقف شدن آن در مکانی به جز جواب مساله زیاد است. برای حل این مشکل پارامتر جدیدی به نام t به الگوریتم بهینه‌سازی حدی اضافه شده و الگوریتم جدیدی به نام بهینه‌سازی حدی با پارامتر t [۲] معرفی شده است؛ تفاوت این الگوریتم جدید و الگوریتم بهینه‌سازی حدی به مرحله انتخاب سلول تعویضی مربوط می‌شود. در این الگوریتم برای پیدا کردن سلول تعویضی، ابتدا سلول‌ها بر اساس شایستگی و به صورت صعودی مرتب می‌شوند. سپس به هر سلول مقداری برابر P_k که از رابطه ۲ قابل محاسبه است اختصاص داده می‌شود؛ در این رابطه k برابر ترتیب سلول در فهرست مرتب شده است:

$$P_k \propto k^{-t} \qquad (۲)$$

پس از نسبت دادن مقادیر P_k به سلول‌ها، سلول تعویضی به اینصورت انتخاب می‌شود که:

۱. در حلقه‌ای به طور اتفاقی یکی از سلول‌ها انتخاب می‌شود.
۲. عددی به صورت اتفاقی بین ۰ و ۱ انتخاب می‌شود.
۳. در صورتی که عدد تولید شده کوچکتر از P_k سلول باشد سلول به عنوان سلول تعویضی انتخاب می‌شود و حلقه پایان می‌پذیرد، در غیر اینصورت حلقه تکرار می‌شود.

بهینه‌سازی حدی با پارامتر t مانند الگوریتم بهینه‌سازی حدی برای حل مسائلی نظیر تقسیم گراف به دو زیر گراف، رنگ آمیزی گراف با سه رنگ و فروشنده دوره‌گرد به کار گرفته شده است و جواب‌های مطلوب‌تری از الگوریتم مادر خود ارائه داده است. همچنین نتایج بدست آمده توسط این الگوریتم هم از نظر دقت و هم از نظر سرعت، کاملا قابل رقابت با الگوریتم‌هایی نظیر شبیه‌سازی تابکاری[1] و الگوریتم ژنتیک نشان داده است.

۵-۲ بهینه‌سازی حدی تعمیم یافته

همانطور که مطرح شد استفاده از الگوریتم بهینه‌سازی حدی نیازمند تعریف تابعی برای محاسبه شایستگی محلی سلول‌های جواب است و علاوه بر آن چگونگی تعویض سلول تعویضی به شکل مساله وابسته است. بنابراین نمی‌توان از آن به راحتی و بدون نیاز به پیاده‌سازی مجدد، برای حل دامنه وسیعی از مسائل استفاده کرد. به این منظور الگوریتم بهینه‌سازی حدی تعمیم یافته بر اساس الگوریتم بهینه‌سازی حدی با پارامتر t معرفی شده است.

الگوریتم بهینه‌سازی حدی تعمیم یافته [۳] یک الگوریتم ابر اکتشافی مانند الگوریتم ژنتیک و شبیه‌سازی تابکاری است. از این الگوریتم می‌توان برای حل دسته وسیعی از مسائل استفاده کرد.

ساختار الگوریتم بهینه‌سازی حدی تعمیم یافته کاملا شبیه الگوریتم بهینه‌سازی حدی با پارامتر t است. برای استفاده از این الگوریتم، ابتدا متغیرهای مساله باید به یک رشته دودویی نگاشت شوند؛ به عنوان مثال اگر مساله پیدا کردن نقطه بهینه یک تابع دو بعدی باشد، ابتدا متغیرهای x و y باید به صورت دودویی درآمده و سپس در کنار هم قرار داده شوند تا جواب به شکل آرایه‌ای از بیت‌ها در آید. به این ترتیب هر سلول در الگوریتم بهینه‌سازی حدی تعمیم یافته معادل یک بیت از جواب خواهد بود؛ بنابراین تعداد سلول‌ها به دقت مورد نیاز و اندازه دامنه مساله بستگی دارد. واضح است که دامنه مقادیر مجاز برای سلول‌ها در الگوریتم اخیر، ۰ یا ۱ می‌باشد و از این نظر در مقایسه با الگوریتم بهینه‌سازی حدی بسیار ساده‌تر است.

مساله دیگری که در الگوریتم بهینه‌سازی حدی تعمیم یافته مطرح است، محاسبه شایستگی سلول‌ها است. همانطور که در بخش‌های اشاره شد، محاسبه شایستگی محلی مهمترین مرحله در الگوریتم بهینه‌سازی حدی است. در دو الگوریتم بهینه‌سازی حدی و بهینه‌سازی حدی با پارامتر t، شایستگی محلی به صورت تابعی وابسته به مساله تعریف می‌شود؛ در الگوریتم بهینه‌سازی حدی تعمیم یافته روشی برای محاسبه شایستگی سلول‌ها ارائه شده که مستقل از نوع و شکل مساله است. طبق این روش در هر مرحله بر اساس جواب جاری و به تعداد سلول‌های آن، جواب‌هایی تولید می‌شود که به عنوان فرزندان جواب جاری شناخته می‌شوند. این جواب‌ها از هم متمایز بوده و هر کدام تنها در مقدار یک سلول با جواب جاری اختلاف دارند. به عنوان مثال اگر رشته دودویی جواب جاری برابر ۰۱۰۰۱ باشد، پنج فرزند با مقادیر ۰۱۰۰۰، ۰۱۰۱۰، ۰۱۱۰۱، ۰۰۰۰۱ و ۱۱۰۰۱ برای آن تولید می‌شود. بعد از تولید شدن جواب‌های جدید، شایستگی هر کدام از آنها بوسیله تابع شایستگی کلی محاسبه می‌شود؛ شایستگی هر سلول برابر خواهد بود با شایستگی فرزندی از جواب جاری که در مقدار آن سلول با جواب جاری اختلاف دارد منهای شایستگی جواب جاری. به این ترتیب شایستگی محلی اجزای جواب بدون نیاز به تابعی برای محاسبه شایستگی محلی و تنها بوسیله تابع شایستگی کلی مساله محاسبه می‌شوند. لازم به ذکر است که خروجی تابع شایستگی کلی برابر مقدار تابع در نقطه (جواب) مورد نظر است. همچنین باید توجه داشت که برای توابعی که مقدار بهینه یک نقطه ماکزیمم است شایستگی هر سلول باید منفی در نظر گرفته شود.

مهمترین مزیت الگوریتم بهینه‌سازی حدی تعمیم یافته در مقایسه با الگوریتم‌های اکتشافی دیگر، دارا بودن تنها یک پارامتر (t) است و از

[1] Simulated Annealing

این نظر استفاده از آن بسیار ساده می‌باشد. سرعت هم‌گرا شدن این الگوریتم بسیار بالا بوده و همچنین نحوه حرکت این الگوریتم به سمت جواب به گونه‌ای است که در صورت رسیدن به جواب، جواب را با دقت بسیار بالایی تولید و محاسبه می‌کند. با این حال، علی رقم تمهیداتی که برای جلوگیری از گرفتار شدن الگوریتم در نقاط بهینه محلی در نظر گرفته شده است، این الگوریتم برای مسائلی که دارای نقاط بهینه محلی زیاد و پراکنده هستند، خوب عمل نکرده و جواب‌های مطلوبی ارائه نمی‌دهد.

۳- اتوماتای سلولی، اتوماتای یادگیر، اتوماتای یادگیر سلولی و مدل CLA-EC

در این بخش از مقاله اتوماتای سلولی، اتوماتاهای یادگیر و اتوماتای یادگیر سلولی به اختصار شرح داده شده و سپس مدل تکاملی CLA-EC که بر مبنای اتوماتای یادگیر سلولی ایجاد شده معرفی خواهد شد.

۳-۱ اتوماتای سلولی[1]

اتوماتای سلولی [۴][۵][۶][۷] یک مدل ریاضی برای سیستم‌هایی است که از اجزا و مولفه‌های ساده تشکیل شده، پویا بوده، با مرور زمان تغییر کرده و بر اساس ارتباط محلی بین اجزای خود فعالیت می‌کنند. اجزای سیستم در مدل اتوماتای سلولی بوسیله مجموعه‌ای منظم از سلول‌ها نمایش داده می‌شوند که در آن هر سلول معادل یکی از اجزای سیستم است. در هر قدم زمانی به هر سلول مقداری از مجموعه‌ای متناهی نسبت داده می‌شود که نشان دهنده وضعیت سلول است. این مقداردهی بر اساس مقدار جاری سلول، سلول‌های همسایه آن و قانون تعریف شده برای اتوماتای سلولی انجام می‌شود؛ از آنجایی که قانون در نظر گرفته شده تنها مقدار سلول و همسایه‌های آن را مورد استفاده قرار می‌دهد، قانون محلی نیز نامیده می‌شود.

هر چند اتوماتای سلولی امکانات زیادی را برای شبیه‌سازی و تعریف سیستم‌های گوناگون در اختیار قرار می‌دهد با این حال استفاده از آن برای شبیه‌سازی سیستم‌های طبیعی، بزرگ و ناشناخته با یک اشکال بزرگ همراه است و آن تعیین فرم قطعی قوانین است؛ زیرا در اغلب سیستم‌ها وجود نویز و عدم قطعیت مانع از تعیین قوانینی قطعی برای سیستم می‌شود. البته راهکار احتمالی کردن قوانین برای حل این مشکل پیشنهاد شده است، ولی با توجه به اینکه تعیین مقادیر احتمالات قوانین برای سیستم‌های ناشناخته مشکلی بزرگ است استفاده از این مدل به راحتی امکان‌پذیر نیست.

۳-۲ اتوماتای یادگیر[2]

اتوماتای یادگیر ماشینی است که می‌تواند تعداد محدودی عمل را انجام دهد؛ هرگاه این ماشین عملی را انتخاب می‌کند، عمل انتخاب شده توسط محیط ارزیابی شده و نتیجه آن به صورت یک سیگنال بازخوردی مثبت (در صورت مناسب بودن عمل) یا منفی (در صورت نامناسب بودن عمل) به اتوماتا بازگردانده می‌شود. مقدار این سیگنال در انتخاب اعمال بعدی تاثیر می‌گذارد. هدف این فرایند این است که اتوماتا بعد از گذشت مدتی به سمت مناسب‌ترین عمل خود در محیط میل کرده و یا به عبارت دیگر یاد بگیرد که کدام عمل بهترین عمل است. برای اطلاعات بیشتر در باره اتوماتاهای یادگیر میتوان به [۸][۹] مراجعه نمود.

۳-۳ اتوماتای یادگیر سلولی[3]

در [۱۰] مدلی متشکل از دو مفهوم اتوماتای سلولی و اتوماتای یادگیر ارائه شده است که با نام اتوماتای یادگیر سلولی شناخته می‌شود. در این مدل هر سلول در اتوماتای سلولی با یک اتوماتای یادگیر جایگزین شده است. به این ترتیب این مدل نه تنها امکان استفاده از چندین اتوماتای یادگیر را به صورت توام فراهم می‌کند بلکه مشکل تعیین فرم قطعی قوانین در اتوماتای سلولی را نیز مرتفع می‌کند.

نحوه عملکرد و به روز رسانی سلول‌های اتوماتای یادگیر سلولی شباهت بسیار زیادی به اتوماتای سلولی دارد با این تفاوت که قبل از انجام اعمال به روز رسانی، ابتدا هر کدام از اتوماتاهای یادگیر عملی را انتخاب می‌کنند. سپس، عمل انتخاب شده توسط هر سلول در کنار اعمال انتخاب شده توسط اتوماتاهای همسایه آن به وسیله قانون محلی اتوماتای سلولی بررسی و نتیجه آن به عنوان پاسخ محیط و سیگنال بازخوردی به سلول برگردانده می‌شود. در نهایت، اتوماتای یادگیر هر سلول با توجه به سیگنال بازخوردیی که از محیط دریافت کرده است، ساختار خود را تصحیح کرده و آموزش می‌بینید. برای اطلاعات بیشتر در باره اتوماتای سلولی یادگیر می‌توان به مراجع [۱۱][۱۲][۱۳][۱۴] مراجعه نمود.

۳-۴ مدل CLA-EC

CLA-EC [۱۵] یک الگوریتم تکاملی است که از ترکیب CLA و مفاهیم مطرح در محاسبات تکاملی به وجود آمده و از زمان معرفی تا امروز در مسائل مختلف مورد استفاده قرار گرفته است [۱۶][۱۷][۱۸][۱۹]. ساختار استفاده شده در الگوریتم CLA-EC از چندین ژنوم تشکیل شده است که هر کدام در یکی از سلول‌های یک اتوماتای یادگیر سلولی قرار گرفته‌اند. هر ژنوم از دو مولفه رشته ژنومی و مدل ژنومی تشکیل شده است؛ رشته ژنومی که به صورت رشته‌ای از

[1] Cellular Automata (CA)

[2] Learning Automata

[3] Cellular Learning Automata (CLA)

بیت‌ها می‌باشد معادل یکی از راه‌حل‌های میانی مساله است و مدل ژنومی نیز مجموعه‌ای از اتوماتاهای یادگیر می‌باشد که بر اساس تجارب گذشته خود و ژنوم‌های دیگر آموزش می‌بینند. به این ترتیب فرآیند تکاملی به طریقی هدایت می‌شود تا شایستگی رشته ژنومی بهبود یافته و به سمت جواب مساله حرکت کند. نحوه انتساب اتوماتاهای یادگیر مدل ژنومی به بیت‌های رشته ژنومی به گونه‌های مختلفی قابل طرح است. برای مثال می‌توانیم رشته ژنومی را به دسته‌های دو بیتی تقسیم کرده و به ازای هر دسته یک اتوماتای یادگیر در نظر بگیریم. به این ترتیب با فرض دودویی بودن فضای جستجو، برای یک رشته ژنومی به طول n (با فرض زوج بودن)، تعداد $n/2$ اتوماتای یادگیر، هرکدام با چهار عمل خواهیم داشت. در این مثال اگر یک اتوماتای یادگیر عمل ۳ را انتخاب کند به معنای آن است که بیت ۱ و ۲ دسته متناظر آن مقادیر ۱ و۱ را می‌پذیرند. در این مقاله با فرض مستقل بودن متغیرهای مساله، ساده‌ترین شکل مدل ژنومی که در آن برای هر بیت در رشته ژنومی یک اتوماتای یادگیر اختصاص داده می‌شود را در نظر می‌گیریم؛ هر اتوماتای یادگیر مقدار بیت متناظر با خود را تعیین می‌کند.

همانطور که اشاره شد ژنوم‌ها در الگوریتم CLA-EC در سلول‌های یک اتوماتای یادگیر سلولی در کنار هم قرار گرفته‌اند. بنابراین با فرض همگام بودن این اتوماتا، سلول‌ها به طور همزمان رشته‌های ژنومی خود و همسایگان خود را مورد بررسی قرار داده و بر اساس تابع شایستگی از میان آنها تعدادی را به عنوان رشته‌های ژنومی مناسب انتخاب می‌کند. این شیوه انتخاب دو طرفه نمی‌باشد؛ به عبارت دیگر اگر یک ژنوم، ژنوم دیگری را به عنوان یکی از کاندیدهای خود انتخاب نماید، تضمینی برای انتخاب شدن این ژنوم از طرف ژنوم مقابل وجود نخواهد داشت. ژنوم بر اساس ژنومهایی که انتخاب نموده است، یک بردار سیگنال تقویتی ساخته و به اتوماتاهای یادگیر خود می‌دهد.

برای تولید رشته ژنومی جدید سلول، در هر گام اتوماتاهای یادگیر مدل ژنومی سلول، عملی را انتخاب و مقدار جدیدی برای بیت‌های متناظر خود در رشته ژنومی اعلام می‌کنند. در صورتی که رشته ژنومی جدید شایستگی بیشتری نسبت به رشته ژنومی قبلی داشته باشد جایگزین رشته قدیمی سلول می‌شود، در غیر اینصورت تغییری در رشته ژنومی سلول ایجاد نمی‌شود. این عمل برای تمامی سلول‌های اتوماتای یادگیر سلولی انجام شده و به این ترتیب جمعیت جدیدی تولید می‌شود. این بخش از الگوریتم معادل یادگیری از تجارب قبلی ژنوم و حفظ آنها در نسل‌های بعدی (ژنوم‌های بعدی) و یا به بیان دیگر معادل تاثیر یادگیری بر تکامل از دیدگاه تئوری بالدوین می‌باشد. یک قانون CLA-EC از دو بخش تشکیل شده است: استراتژی انتخاب و استراتژی تولید سیگنال تقویتی. منظور از استراتژی انتخاب برای یک سلول، نحوه انتخاب تعداد مشخصی از همسایه‌های سلول است که برای تولید سیگنال تقویتی استفاده می‌شوند و استراتژی تولید سیگنال تقویتی روشی است که سیگنال تقویتی بر اساس سلول‌های انتخاب شده ایجاد می‌شود [۱۸].

۴- الگوریتم پیشنهادی (CLA-EC-EO)

همانطور که قبلا اشاره شد، بهینه‌سازی حدی تعمیم یافته ممکن است در یکی از نقاط بهینه محلی گرفتار شود ولی این الگوریتم این ویژگی مهم را دارد که در صورت قرار گرفتن در اطراف نقطه بهینه، جواب را با دقت بالایی پیدا می‌کند. از طرف دیگر نتایج گزارش شده برای الگوریتم CLA-EC نشان می‌دهد که این الگوریتم با وجود جستجوی مناسب دامنه مساله، جواب‌هایی با دقت بالا تولید نمی‌کند. در این بخش یک الگوریتم ترکیبی به نام CLA-EO-EC که دارای مزایای هردو الگوریتم بهینه‌سازی حدی تعمیم یافته و CLA-EC می‌باشد پیشنهاد می‌گردد. در الگوریتم CLA-EC که در این مقاله استفاده شده است، همسایه‌های یک سلول تمامی همسایه‌های آن سلول تا یک شعاع مشخص و خود سلول را شامل می‌شود. بطور مثال همسایگی با شعاع یک به انتخاب سه سلول منتهی می‌شود. برای تولید سیگنال تقویتی هر اتوماتای یادگیر سلول (ژنوم)، تمامی بیت‌ها در رشته‌های ژنومی سلول‌های انتخاب شده بررسی می‌شوند. اگر تعداد بیت‌های هم مقدار با بیت متناظر با اتوماتای یادگیر در سلول مورد نظر، از نصف تعداد سلول‌های انتخاب شده بیشتر باشد، اتوماتای یادگیر تشویق و در غیر اینصورت جریمه می‌شود.

در الگوریتم پیشنهادی، وظیفه جستجوی دامنه مساله به عهده CLA-EC و وظیفه دقیقتر شدن در جواب‌ها به عهده الگوریتم بهینه‌سازی حدی است. به بیان ساده‌تر، در الگوریتم پیشنهادی ابتدا فضای مساله توسط CLA-EC جستجو می‌شود و هر جا احتمال حضور جواب وجود داشته باشد (شایستگی سلول بیشتر از سلول‌های همسایه‌اش باشد) بهینه‌سازی حدی وارد عمل شده و محل مورد نظر را برای یافتن جواب بهتر با دقت بیشتری جستجو می‌کند. سپس، الگوریتم CLA-EC مجددا کنترل را بر عهده گرفته و جستجو ادامه پیدا می‌کند. لازم به ذکر است که جستجوی دامنه توسط CLA-EC قابل جایگزینی با جستجوی اتفاقی نیست، زیرا CLA-EC به شکلی هدفمند فضا را جستجو کرده و الگوریتم بهینه‌سازی حدی را به شکل بهتری به سمت نقاط بهینه هدایت می‌کند. مراحل کلی الگوریتم CLA-EO-EC به شرح زیر است:

۱. مقداردهی و تولید جمعیت اولیه
۲. برای هر سلول (ژنوم)، اگر شایستگی سلول انتخاب شده از همه همسایه‌هایش بیشتر بود:
   ۲-۱. با توجه به رشته ژنومی سلول، رشته ژنومی جدید توسط الگوریتم بهینه‌سازی حدی تعمیم یافته تولید می‌شود. (رشته ژنومی سلول به عنوان یک جواب در نظر گرفته شده، فرزندان آن تولید و یکی از آنها با توجه به رابطه ۲ انتخاب می‌شود.)
   ۲-۲. در صورتی که شایستگی ژنوم جدید بیشتر از شایستگی ژنوم سلول جاری باشد، رشته جدید جایگزین ژنوم سلول

شده و اتوماتاهای یادگیر سلول مقداردهی مجدد می‌شوند.

۳. در غیر این صورت:

۳-۱. سیگنال‌های تقویتی اتوماتاهای یادگیر توسط الگوریتم CLA-EC ایجاد شده و سپس بر اساس این سیگنال‌های تقویتی اتوماتاهای یادگیر جریمه شده و یا پاداش می‌گیرد.

۳-۲. یک رشته ژنومی جدید توسط اتوماتاهای یادگیر در هر سلول تولید می‌شود. اگر این رشته ژنومی شایسته‌تر از رشته ژنومی خود سلول باشد، جایگزین رشته ژنومی سلول می‌شود.

۴. تا محقق شدن شرط توقف مرحله ۲ تکرار می‌شود.

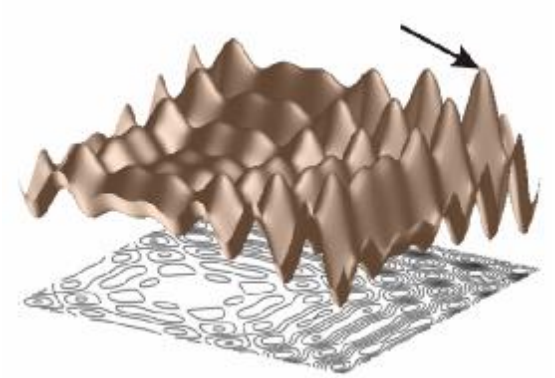
۵. بهترین جواب اجرا به عنوان خروجی برگردانده می‌شود.

در این الگوریتم، مرحله ۲ مربوط به بخش بهینه‌سازی حدی و مرحله ۳ مربوط به بخش CLA-EC الگوریتم می‌باشند.

۵- نتایج آزمایش‌ها

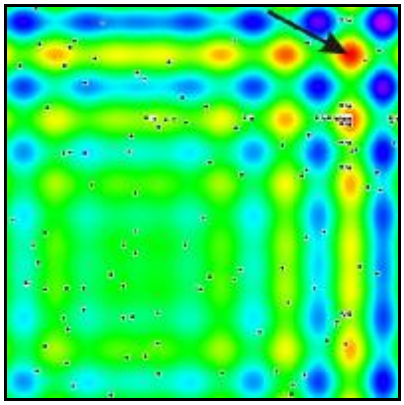
در این بخش ابتدا به بررسی رفتار الگوریتم CLA-EO-EC و مقایسه آن بارفتارهای الگوریتم‌های بهینه‌سازی حدی تعمیم یافته و CLA-EC می‌پردازیم. سپس نتایج اجرای این الگوریتم‌ها بر روی چند مساله بهینه‌سازی ارائه می‌شود.

برای بررسی رفتار الگوریتم‌ها از تابع Bumps که در جدول ۱ و شکل ۱ نمایش داده شده استفاده شده است. این تابع دارای نقاط بهینه محلی زیادی می‌باشد. ماکزیمم این تابع در شکل ۳ توسط یک پیکان نشان داده شده است

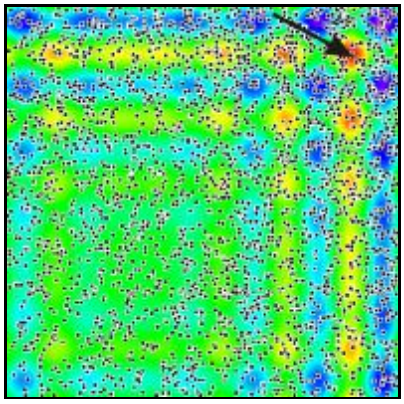


شکل (۱) : نمای سه بعدی تابع Bumps

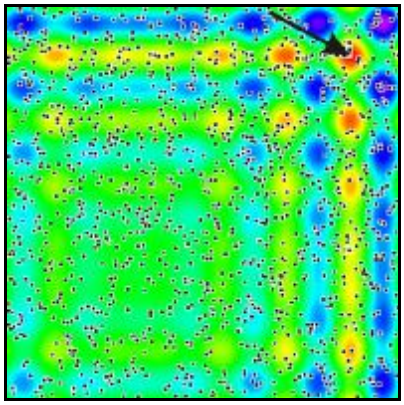
شکل‌های ۲، ۳ و ۴ نتایج اجرای الگوریتم‌های بهینه‌سازی حدی تعمیم یافته، CLA-EC و CLA-EC-EO بر روی تابع Bumps را بعد از ۳۰۰۰ ارزیابی نمایش می‌دهند. در این اشکال که تابع را از جهت محور z ها (از بالا) نشان می‌دهد، نقطه بهینه مساله با یک فلش مشخص شده است. همانطور که در شکل۲ نشان داده شده است، جواب‌های تولید شده توسط الگوریتم بهینه‌سازی حدی تعمیم یافته گستردگی بسیار کمی دارند و الگوریتم با رسیدن به اولین نقطه بهینه بر روی آن متمرکز شده و نهایتا در همان نقطه متوقف می‌شود. این تابع یک نمونه از توابعی است که الگوریتم بهینه‌سازی حدی تعمیم یافته عمکرد مناسبی برای آن ندارد. شکل۳ نحوه عملکرد الگوریتم CLA-EC را برای تابع Bumps نشان می‌دهد. همانطور که مشاهده می‌شود جواب‌های تولید شده توسط این الگوریتم پراکندگی بسیار زیاد و مناسبی دارند و تمامی دامنه مساله را پوشش می‌دهند و چگالی جواب‌ها در اطراف نقاط بهینه بخصوص نقطه بهینه عمومی بیشتر از سایر نقاط است.



شکل (۲) : رفتار الگوریتم بهینه‌سازی حدی برای تابع Bumps



شکل (۳) : رفتار الگوریتم CLA-EC برای تابع Bumps



شکل (۴) : رفتار الگوریتم CLA-EO-EC برای تابع Bumps

شکل ۴ جواب‌های تولید شده توسط الگوریتم CLA-EO-EC را برای تابع Bumps نشان می‌دهد. همانطور که مشاهده می‌شود پراکندگی جواب‌های تولید شده توسط این الگوریتم کمتر از الگوریتم CLA-EC است. علت این اتفاق به رفتار الگوریتم بهینه‌سازی حدی تعمیم یافته مربوط می‌شود که جواب‌هایی با فاصله‌های بسیار کم و نزدیک به هم تولید می‌کند؛ الگوریتم CLA-EC-EO در محدوده نقاطی که احتمال حضور جواب بهینه در آن وجود دارد با دقت بیشتری به جستجو می‌پردازد.

بررسی شکل‌های ۲ تا ۴ همچنین نشان می‌دهد که رفتار الگوریتم CLA-EO-EC شباهت بیشتری به الگوریتم CLA-EC دارد تا به الگوریتم بهینه‌سازی حدی تعمیم یافته. دلیل این امر را می‌توان در نحوه انتخاب بین مراحل ۲ و ۳ در الگوریتم CLA-EC-EO جستجو کرد. از آنجایی که از بین سلول‌های یک همسایگی تنها یک سلول بهترین است، قسمت بهینه‌سازی حدی الگوریتم فقط یکبار برای کل همسایگی اجرا می‌شود؛ به عنوان مثال اگر شعاع همسایگی برابر ۱۰ باشد، به ازای هر بار اجرای قسمت مربوط به بهینه‌سازی حدی (مرحله ۲)، ۲۰ بار الگوریتم CLA-EC (مرحله ۳) اجرا می‌شود. بنابراین شباهت الگوریتم CLA-EC-EO و CLA-EC توجیه‌پذیر و منطقی است.

برای مقایسه الگوریتم‌ها از هفت تابع Bumps، Squashed Frog، Langermann، CPF1، Ackley's Path، Easom و De Jong F2 که در جدول ۱ آمده است استفاده شده است. از بین این توابع چهار تابع Langermann، Ackley's Path، Easom و De Jong F2 جزو توابع استاندارد و پرکاربرد برای مقایسه الگوریتم‌های بهینه‌سازی هستند [۲۰][۲۱]. علاوه بر چهار تابع فوق از سه تابع CPF1، Bumps و Squashed Frog که در [۲۲] معرفی شده اند نیز برای بررسی الگوریتمها استفاده شده است. این هفت تابع از نظر شکل و تعداد نقاط بهینه در سه گروه زیر قرار می‌گیرند:

۱. توابعی با چندین نقطه بهینه که یک یا گروهی از این نقاط جواب مساله‌اند مانند توابع Bumps، Squashed Frog و Langermann.
۲. توابعی با یک نقطه بهینه که شیب تابع در همه نقاط به سمت آن میباشد مانند توابع: CPF1و Ackley's .Path
۳. توابعی با یک نقطه بهینه که محل آن از طریق دنبال کردن شیب تابع پیدا نمی‌شود مانند توابع Easom و De Jong F2.

آزمایشهای برای تعداد تکرارهای۱۰۰۰، ۱۰۰۰۰ و ۳۰۰۰۰ که به ترتیب با نامهای تعداد تکرار کم، متوسط و زیاد به آنها رجوع می‌شود انجام شده است. به ازای هر تعداد تکرار، هر الگوریتم‌ها ۱۰۰ بار بر روی هر یک از توابع آزمایشی اجرا شده و جواب‌های تولید شده به عنوان داده آماری برای مقایسه استفاده شده‌اند. برای این ازمایشها پارامتر t برای الگوریتم بهینه‌سازی حدی تعمیم یافته برابر ۱.۲۵، پارامترهای جریمه و پاداش برای الگوریتم‌های CLA-EC و CLA-EC-EO برابر ۰.۰۱ و شعاع همسایگی برابر ۶۰ در نظر گرفته شده است. همچنین جمعیت اولیه تمامی الگوریتم‌ها برابر ۱۰۰ انتخاب شده است.

شکل‌های ۵ تا ۱۱ نمودارهای توزیع جواب‌ها در نزدیکی جواب بهینه (منتهی الیه سمت چپ محور افقی) را نشان می‌دهند. در این نمودارها محور افقی نشان دهنده مقدار جواب و محور عمودی نشان دهنده تعداد جواب‌ها می‌باشد. به عنوان مثال، شکل ۵-ب نشان می‌دهد که الگوریتم بهینه‌سازی حدی تعمیم یافته ۳ بار، الگوریتم CLA-EC ۱۴ بار و الگوریتم CLA-EC-EO ۲۲ بار از ۱۰۰ اجرا به جواب بهینه همگرا شده‌اند. لازم به ذکر است که برای بالا بردن خوانایی نمودارها تنها ۱۰ جواب نزدیک به نقطه بهینه نشان داده شده و دقت محور عمودی نیز بر اساس بیشترین تعداد جواب‌ها در هر نمودار تنظیم شده است.

بررسی نمودارها نشان که الگوریتم CLA-EC-EO از هر دو الگوریتم پایه‌ای خود (CLA-EC و بهینه‌سازی حدی تعمیم یافته) برای توابعی که در این مقاله استفاده شده است بهتر عمل کرده است. توجه کنید که الگوریتم بهینه‌سازی حدی تعمیم یافته فقط برای مسایل CPF1 و Ackley's Path و الگوریتم CLA-EC فقط برای مسایل Bumps و Longermann به خوبی عمل می‌کنند. همچنین مشاهده می‌شود که الگوریتم پشنهادی برای توابعی که دو الگوریتم پایه جواب‌های خوبی تولید نکرده‌اند، نیز دارای عملکرد خوبی می‌باشد.

۶- نتیجه‌گیری

در این مقاله یک الگوریتم بهینه‌سازی ابر اکتشافی ترکیبی که از ترکیب الگوریتم‌های بهینه‌سازی حدی تعمیم یافته و CLA-EC حاصل شده است معرفی گردید. این الگوریتم ترکیبی مزایای هر دو الگوریتم بهینه‌سازی حدی تعمیم یافته و CLA-EC را دارا می‌باشد؛ نتایج آزمایش‌ها بر روی مسایل مختلف نشان داد که الگوریتم پیشنهادی هم از دقت الگوریتم بهینه‌سازی حدی تعمیم یافته و هم از قابلیت جستجوی بالای الگوریتم CLA-EC برخوردار است.

سپاسگزاری

این کار تحقیقاتی توسط مرکز تحقیقات مخابرات ایران حمایت مالی شده است که بدین وسیله از این مرکز سپاسگزاری می‌شود.

مراجع

[1] Tucotte, D. L., **Self-Organized Criticality**, Rep. Prog. Phys., Vol. 62, 1999, pp. 1377-1429

[2] Boettcher, S. and Percus,A. G., **Extremal Optimization: An Evolutionary Local-Search Algorithm**, http://arxiv.org/abs/cs.NE/0209030

[3] Fabiano Luis de Sousa and Valeri Vlassov and Fernando Manuel Ramos, **Centralized Extremal Optimization for**

Solving Complex Optimal Design Problems, Lecture Notes in Computer Science, Vol. 2723, 2003, pp.375-276

[۴] محمد شیبانی، **اتوماتای یادگیر سلولی، انواع و کاربردهای آن**، سمینار کارشناسی ارشد، دانشکده مهندسی کامپیوتر و فناوری اطلاعات، دانشگاه صنعتی امیر کبیر، ۱۳۸۴.

[5] Wolfram, S., **Cellular Automata**, Los Alamos Science, vol. 9, pp. 2-21, Fall 1983.

[6] Wolfram, S., **Universality And Cmplexity In Cellular Automata**, Physica D, no. 10, pp. 1-35, January 1984.

[۷] یاسر مهدوی فر، **اتوماتای سلولی و کاربردهای آن**، سمینار کارشناسی ارشد، دانشکده مهندسی کامپیوتر و فناوری اطلاعات، دانشگاه صنعتی امیر کبیر، ۱۳۸۳

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

[9] Thathachar, M. A. L. and Sastry, P. S., **Varieties of Learning Automata: An Overview**, IEEE Transaction on Systems, Man, and Cybernetics-Part B: Cybernetics, Vol. 32, No. 6, PP. 711-722, 2002.

[۱۰] محمدرضا میبدی، حمید بیگی و مسعود طاهرخانی، **اتوماتای یادگیر سلولی**، در مجموعه مقالات ششمین کنفرانس انجمن کامپیوتر ایران، ص ۱۶۳-۱۵۳، ۱۳۷۹.

[11] Beigy, H. and Meybodi, M. R., **A Mathematical Framework for Cellular Learning Automata**, Advances on Complex Systems, Vol. 7, Nos. 3-4, pp. 295-320, September/December 2004

[12] Beigy , H. and Meybodi, M. R., **Open Synchronous Cellular Learning Automata**, Advances in Complex Systems, 2007, to appear .

[13] Beigy, H. and Meybodi, M. R., **Asynchronous Cellular Learning Automata**, Automatica, Journal of International Federation of Automatic Control, 2007, to appear

[14] Rastegar, R., Meybodi, M. R. and Hariri, A., **A New Fine Grained Evolutionary Algorithm based on Cellular Learning Automata**, International Journal of Hybrid Intelligent Systems, IOS Press, Volume 3, Number 2, pp. 83-98, 2006.

[15] Rastergar, R. and Meybodi, M. R., **A New Evolutionary Computing Model Based on Cellular Learning Automata**, IEEE conference on Cybernetics and Intelligent Systems 2004 (CIS2004), Singapore, December 2004.

[16] Rastegar, A., Arasteh, A. R., Harriri. A. and Meybodi, M. R., **A Fuzzy Clustering Algorithm Using Cellular Learning Automata based Evolutionary Algorithm**, Proceedings of Fourth International Conference on Hybrid Intelligent Systems(HIS'04), pp. 310-314, Japan, Kitakyushu, Dec. 2004.

[۱۷] رضا رستگار، **تخصیص کانال در شبکه‌های سلولی مخابراتی با استفاده از اتوماتای یادگیر سلولی**، پایان نامه کارشناسی ارشد، دانشکده مهندسی کامپیوتر و فناوری اطلاعات، دانشگاه صنعتی امیر کبیر، ۱۳۸۳

[18] Masoodifar, B., Meybodi, M. R. and Rastegar, R., **Asynchronous CLA-EC**, Proceedings of 11th Annual CSI Computer Conference of Iran, Fundamental Science Research Center (IPM), Computer Science Research Lab., Tehran, Iran, pp. 447-458, Jan. 24-26, 2006

[19] Hariri, A., Rastegar, R., Navi, K., Zamani, M. S. and Meybodi, M. R., **Cellular Learning Automata based Evolutionary Computing (CLA-EC) for Intrinsic Hardware Evolution**, Proceedings of NASA/DoD Conference on Evolvable Hardware (EH'05), pp. 294-297, Washington DC, USA, June 29-July 1, 2005.

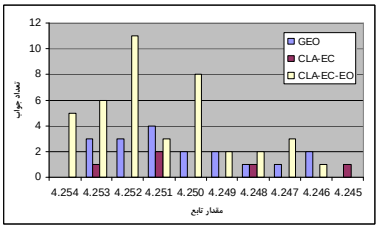
[20] Silagadze, Z. K. **Finding Two-Dimensional Peaks**, http://arxiv.org/abs/physics/0402085

[21] **GEATbx: Example Functions (Single And Multi-Objective Functions) 2 Parametric Optimization**, http://www.geatbx.com/docu/fcnindex-01.html

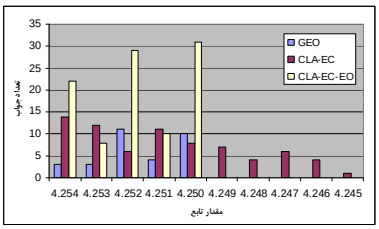
[22] **Optimization Algorithm Toolkit (OAT)**, http://optalgtoolkit.sourceforge.net/index.html

جدول (۱) : توابع آزمایشی استفاده شده

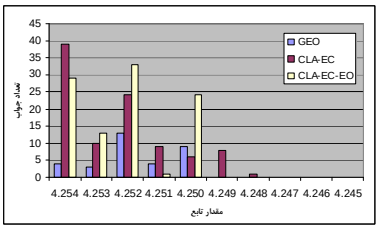
تابع	رابطه تابع	بازه مقایر ورودی
Bumps	$f(x)=x\times\sin(4\times\Pi\times x)-y\times\sin(4\times\Pi\times y+\Pi)+1$	$-1\le x,y\le 2$
Squashed Frog	$t=(x-0.1)^2+(y-0.2)^2$ $f(x)=1+\sqrt[4]{t}-\cos(5\times\Pi\times\sqrt{t})$	$-2\le x,y\le 2$
Langermann	$f(x)=-\sum_{i=1}^{m}c_i\left(e^{-\frac{\left\|\bar{x}-A(i)\right\|^2}{\Pi}}\times\cos\left(\Pi\times\left\|\bar{x}-A(i)\right\|^2\right)\right)$	$0\le x,y\le 10$ برای مشاهده مقادیر A و c به [20] و [21] مراجعه شود. $\bar{x}$ برداری با مولفه‌ها x و y است.
CPF1	$f(x)=\lvert x\rvert\lvert y\rvert+\lvert x\rvert+\lvert y\rvert$	$-10\le x,y\le 10$
Ackley's Path	$f(x)=-a\times e^{-b\sqrt{\frac{x^2+y^2}{2}}}-e^{\frac{\cos(c\times x)+\cos(c\times y)}{2}}+a+e$	$-32.768\le x,y\le 32.768$ $a=20,b=0.2,c=2\Pi$
Easom	$f(x)=-\cos(x)\times\cos(y)\times e^{-\left((x-\Pi)^2+(y-\Pi)^2\right)}$	$-100\le x,y\le 100$
De Jong F2	$f(x)=100\times\left(y-x^2\right)^2+\left(1-x^2\right)$	$-2.048\le x,y\le 2.048$



(الف) ۱۰۰۰ تکرار

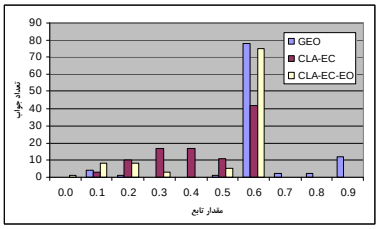


(ب) ۱۰۰۰۰ تکرار

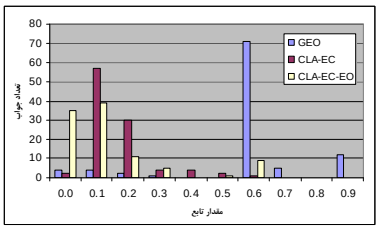


(ج) ۳۰۰۰۰ تکرار

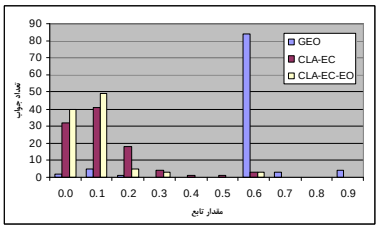
شکل (۵): مقایسه الگوریتم‌ها برای تابع Bumps



(الف) ۱۰۰۰ تکرار

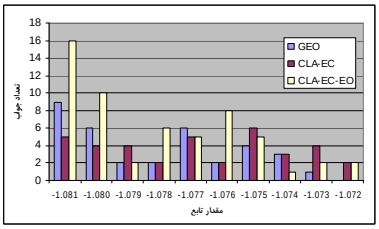


(ب) ۱۰۰۰۰ تکرار

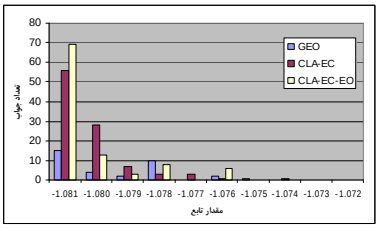


(ج) ۳۰۰۰۰ تکرار

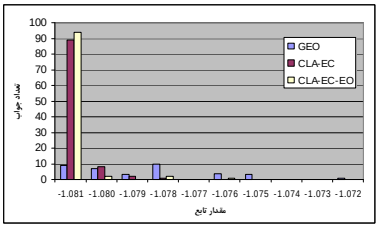
شکل (۶): مقایسه الگوریتم‌ها برای تابع Squashed Frog



(الف) ۱۰۰۰ تکرار

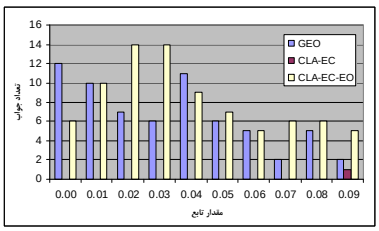


(ب) ۱۰۰۰۰ تکرار

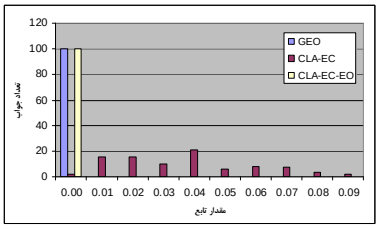


(ج) ۳۰۰۰۰ تکرار

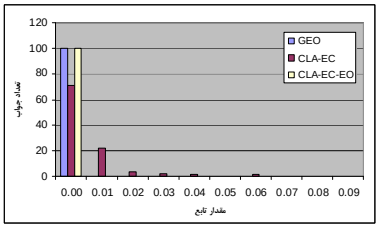
شکل (۷): مقایسه الگوریتم‌ها برای تابع Langermann



(الف) ۱۰۰۰ تکرار

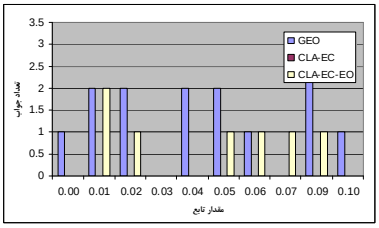


(ب) ۱۰۰۰۰ تکرار

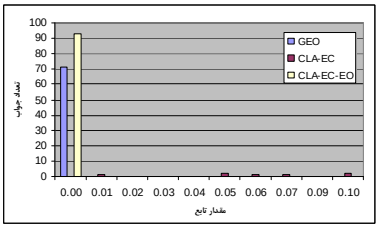


(ج) ۳۰۰۰۰ تکرار

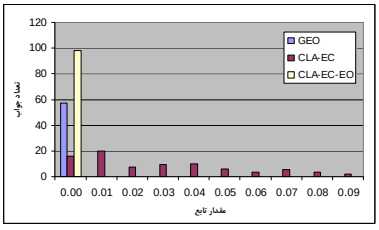
شکل (۸): مقایسه الگوریتم‌ها برای تابع CPF1



(الف) ۱۰۰۰ تکرار

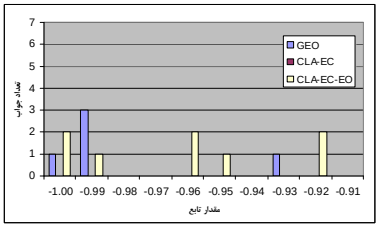


(ب) ۱۰۰۰۰ تکرار

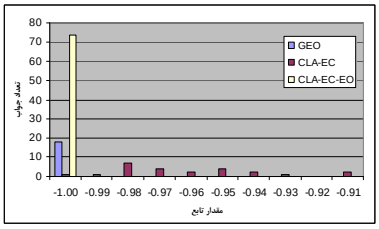


(ج) ۳۰۰۰۰ تکرار

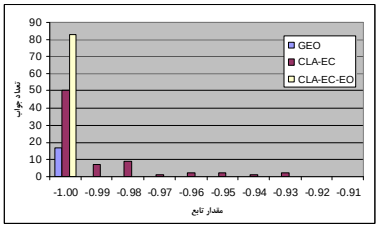
شکل (۹): مقایسه الگوریتم‌ها برای تابع Ackley's Path



(الف) ۱۰۰۰ تکرار

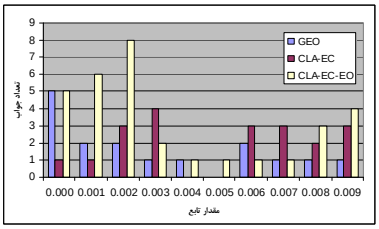


(ب) ۱۰۰۰۰ تکرار

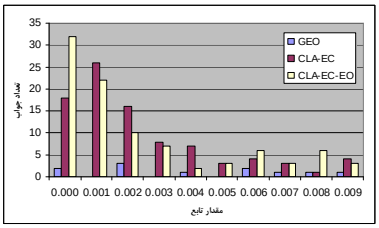


(ج) ۳۰۰۰۰ تکرار

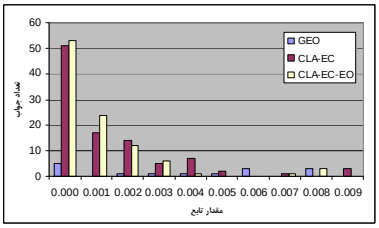
شکل (۱۰): مقایسه الگوریتم‌ها برای تابع Easom



(الف) ۱۰۰۰ تکرار



(ب) ۱۰۰۰۰ تکرار



(ج) ۳۰۰۰۰ تکرار

شکل (۱۱): مقایسه الگوریتم‌ها برای تابع De Jong F2