یک الگوریتم CLA-EC همکارانه جدید برای محیط های پویا

مژده خاکسارمنشاد[۱]،محمدرضا میبدی[۲]

چکیده

بسیاری از تحقیقات در محاسبات تکاملی روی بهینه سازی مسائل ایستا بحث می کند.در حالی که بسیاری از مسائل بهینه سازی دنیای واقعی پویا هستند و روش های بهینه سازی نیاز است، که قادر باشد به طور پیوسته راه حل را بامحیط در حال تغییر وفق دهد. محاسبات تکاملی بر پایه آتاماتای یادگیر سلولی(CLA-EC) یک مدل محاسبات تکاملی است که از ترکیب آتاماتای یادگیر سلولی (CLA) و مدل محاسبات تکاملی (EC) به وجود آمده است.در این مقاله چهار ورژن از مدل CLA-EC برای یک نوع از مسائل بهینه سازی پویا که مسئله قله های متحرک نامیده می شود، بکار گرفته شده است. سپس یک رویکرد جدید معرفی شده است،در این رویکرد یک الگوریتم یک الگوریتم ژنتیک با جهش وفقی با یک الگوریتم CLA-EC همکاری می کند. برای نشان دادن تاثیر الگوریتم معرفی شده، آزمایشات مثعددی طراحی شد.این آزمایشات نشان می دهد الگوریتم معرفی شده ، نسبت به رویکردهایی که تا کنون برای محیط های پویا طراحی شده است دارای دقت بالاتری است.

کلمات کلیدی

محاسبات تکاملی، محیط های پویا، آتاماتای یادگیرسلولی، CLA-EC، مسئله قله های متحرک، CLA-EC جستجو/حافظه.

A New Cooperative CLA-EC Algoritm for Dynamic Environment

Mojdeh khaksar manshad; mohammadreza meybodi

ABSTRACT

Most research in evolutionary computation focuses on optimization of static, non-changing problems. Many real-world optimization problems, however, are dynamic, and optimization methods are needed that are capable of continuously adapting the solution to a changing environment. Cellular Learning Automata based Evolutionary Computing (CLA-EC) is an evolutionary computing model obtained by combining cellular learning automata (CLA) model and evolutionary computing (EC) model. This paper uses four version of CLA-EC model for a kind of dynamic optimization problem that called moving peaks problem. Then we proposed one new approach, in this approach a self adaptive genetic algorithm and a CLA-EC cooperating to solve a problem. In order to show the effectiveness of the proposed CLA-EC extensive computer experimentation were conduced. It has been shown that the proposed CLA-EC outperforms existing approaches reported for dynamic environments in terms of accuracy.

KEYWORDS

Evolutionary Computation, Dynamic Environments, Cellular Learning Automata, CLA-EC, Moving Peaks Problem, New Cooperative CLA-EC.

1. مقدمه

هر گاه یک تغییر در هدف بهینه سازی ، نمونه مسئله یا بعضی تغییرات محدودیت [۱] یک مسئله بهینه سازی رخ دهد ، ممکن است بهینه آن مسئله تغییر کند.اگر این حالت رخ دهد وفق دادن راه حل با راه حل قدیمی ضروری است.از آن جا که الگوریتم های تکاملی به طور رایج در حال تکامل هستند،و ذاتا به طور پیوسته در حال وفق دادن هستند، یک کاندیدای مناسب به نظر می آیند. مشکل اصلی الگوریتم های تکاملی استاندارد، این هست که آن ها عاقبت به یک بهینه [۲] همگرا[۳] می شوند و بنابراین تنوع [۴] لازم را برای اکتشاف [۵] به صورت کارا از دست می

[۱] دانشجوی کارشناسی ارشد دانشگاه آزاد اسلامی واحد قزوین، دانشکده برق، رایانه و فن آوری اطلاعات، پست الکترونیک: khaksar_mojdeh@yahoo.com
[۲] عضو هیئت علمی دانشگاه امیرکبیر، دانشکده کامپیوتر، پست الکترونیک: mmeybodi@aut.ac.ir

دهند.بنابراین هنگامی که جمعیت الگوریتم تکاملی همگرا شد ، توانایی اش برای وفق پذیری با تغییرات محیط را از دست می دهد. روش هایی برای مناسب ساختن الگوریتم های تکاملی برای محیط های پویا وجود دارد. برخی از روش هایی که به صورت گسترده استفاده شده اند عبارتند از:روش های شروع دوباره[۱،۲،۳،۴،۵،۶،۷] ،جهش وفقی [۸،۹،۱۰،۱۱،۱۲،۱۳،۳۰،۳۱]و حافظه اضافی[۱۴،۱۵،۱۶،۱۷،۱۸،۱۹،۲۰] ، انتخاب اصلاحی[۶] [۲۱،۲۲،۲۳،۲۴،۲۵،۲۶]، روش های چندجمعیتی[۷][۲۷،۲۸،۲۹]. همه محیط های پویا یکسان نیستند و محیط های پویای متفاوت نیاز به روش های بهینه سازی متفاوت دارند.معیارهای لازم برای تشخیص این که یک مسئله محک مناسب است ، عبارت است از: باید این امکان را داشته باشد تا بسیاری از متغیرهای محیطی را تغییر دهد،نهایتاً باید محک هایی برای انکد باینری و مقادیر حقیقی وجود داشته باشد،آن ها باید برای پیاده سازی و توصیف ساده باشند،آن ها برای تحلیل باید ساده باشند،باید از لحاظ محاسباتی کارا باشند. جفت کردن پویای بیت [۸]، سهمی متحرک[۹]، مسئله کوله پشتی با زمان متغیر [۱۰]، تابع قله های متحرک [۱۱]، مسائل برنامه ریزی [۱۲]، قله های در حال نوسان [۱۳]،مسائلی هستند که به عنوان مسائل محک مناسب شناخته شده اند.

تاکنون CLA-EC برای مسائل بهینه سازی استاتیک به کار گرفته شده است. بعضی از اشکالات CLA-EC استاندارد سرعت همگرایی پایین و دقت کم برای بعضی از مسائل بهینه سازی است.و همین طور CLA-EC توانایی دنبال کردن تغییرات محیط های پویا را ندارد.بنابراین نیاز است تا تغییراتی در CLA-ECایجاد شود تا توانایی دنبال کردن تغییرات را داشته باشد.یکی از دلایل این عدم توانایی از دست رفتن تنوع جمعیت بعد از گذشت چند نسل است.در این مقاله هدف اصلی طراحی الگوریتم های بهینه سازی مبتنی بر CLA-EC برای محیط های پویا است الگوریتم های طراحی شده در این مقاله با الگوریتم های جهش وفقی و الگوریتم های تکاملی مبتنی بر حافظه مورد مقایسه قرار خواهد گرفت. در همین راستا دو معیار بهینگی یعنی «کارایی برون خطی» و»خطای برون خطی » مورد توجه قرار گرفته است. مسائل محکی که در این مقاله مورد ارزیابی قرار می گیرد مسئله قله های متحرک خواهد بود.تمام پیاده سازی ها در محیط MATLAB انجام خواهد شد.

مسئله محک قله های متحرک شامل m قله در یک فضای پارامتر مقدار حقیقی n بعدی است که چشم انداز شایستگی به عنوان ماکزیمم روی همه تابع های قله ای تعریف می شود و به صورت زیر فرموله می شود.

$$f(\vec{x},t) = \max(B(\vec{x}), \max P(\vec{x}, h_i(t), w_i(t), \vec{p}_i(t)))$$
$$i = 1...m \qquad (1)$$

که $B(\vec{x})$ نا متغیر با زمان که پایه واساس چشم انداز است و p تابعی است که شکل قله را تعیین می کند، که هر یک از m قله پارامترهای متغیر با زمان شامل ارتفاع h،عرضw،و مکان p است.بعد از هر Δe ارزیابی ، ارتفاع،عرض و مکان هر قله تغییر می کند.ارتفاع و عرض هر قله به وسیله اضافه کردن متغیر گوسین تصادفی تغییر می کند.مکان هر قله به وسیله بردار v با طول ثابت s حرکت خواهد کرد.بنابراین پارامتر s اجازه کنترل شدت یک تغییر را می دهد، Δe فرکانس تغییر را تعیین می کند.یک پارامتر جدید λ، تعیین می کند که تغییر مکان قله چه مقدار بستگی به حرکت قبلی اش دارد.اگر λ=۰ باشد هر حرکت کاملاً تصادفی است،و برای λ=۱ قله همیشه در جهت یکسان حرکت خواهد کرد. فرمول هایی که باعث تغییر یک قله می شوند به صورت زیر هستند.

$$\sigma \in N(0,1)$$
$$h_i(t) = h_i(t-1) + height_severity.\sigma \qquad (2)$$
$$w_i(t) = w_i(t-1) + width_severity.\sigma$$
$$\vec{p}_i(t) = p_i(t-1) + \vec{v}_i(t)$$

کنفرانس داده کاوی ایران

2. الگوریتم تکاملی مبتنی بر آتاماتای یادگیر سلولی

الگوریتم های تکاملی از جمله الگوریتم مطرح شده در این مقاله امکان جستجو را در هر فضای جستجوی گسسته متناهی دلخواه دارند، اما به منظور ساده کردن ارائه الگوریتم فرض می کنیم فضای جستجوی مورد نظر یک فضای متناهی دودویی باشد. بنابراین مساله بهینه سازی به فرم زیر قابل طرح است.

$$\max\{f(\underline{X}) \mid \underline{X} \in B^n\} \qquad (3)$$

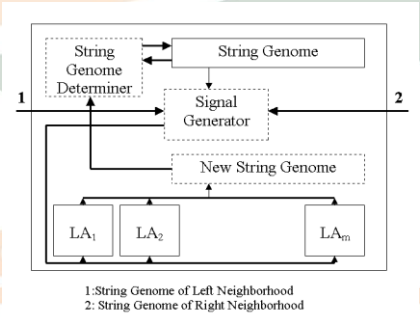
به طوریکه $f(.)$ یک تابع حقیقی و $B^n = \{0,1\}^n$ فضای جستجوی دودویی مورد نظر است. فرض کنید هر ژنوم مورد بحث در این الگوریتم دارای دو مولفه رشته ژنومی و مدل ژنومی باشد. رشته ژنومی همان راه حل های میانی مساله مورد نظر می باشند. مدل ژنومی متشکل از

تعدادی اتوماتای یادگیر می باشد که بر اساس تجارب گذشته خود و ژنومهای دیگر آموزش می بینند. و بدین ترتیب فرایند تکامل به سمتی هدایت می شود تا ارزش رشته ژنومی بر اساس تابع ارزیابی بهبود یابد. نحوه انتساب اتوماتاهای یادگیر مدل ژنومی به بیتهای رشته ژنومی به گونه های مختلفی قابل طرح است. برای مثال می توانیم رشته ژنومی را به دسته های دو بیتی تقسیم کرده و به ازای هر دسته یک اتوماتای یادگیر در نظر بگیریم. به این ترتیب با فرض دودویی بودن فضای جستجو و رشته ژنومی به طول n (با فرض زوج بودن)، تعداد n/۲ اتوماتای یادگیر، هرکدام با چهار عمل خواهیم داشت. برای مثال در این حالت 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 می پذیرند. در این پایان نامه با فرض مستقل بودن متغیرهای مساله ساده ترین شکل مدل ژنومی را در نظر می گیریم. به عبارت دیگر به ازای هر بیت در رشته ژنومی یک اتوماتای یادگیر در مدل ژنومی خواهیم داشت که مقدار آن بیت را تعیین می کند. تا به اینجا ساختار مورد نیاز در الگوریتم را شرح دادیم. همانطور که آشکار است این ساختار مانند ساختار سلول در اتوماتای یادگیر سلولی است.

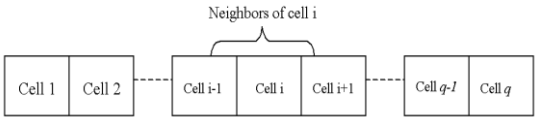
حال برای ادامه بحث یک اتوماتای یادگیر سلولی ، $CLA(\underline{L}_1,...,\underline{L}_k)$ ، با k سلول که هر سلول مجهز به n اتوماتای یادگیر است در نظر بگیرید. در هر سلول یک رشته به طول n نشان دهنده حالت آن سلول می باشد که در الگوریتم مورد نظر ما این رشته همان رشته ژنومی می باشد که در قسمت قبل آن را شرح دادیم. واضح است که فضای حالت هر سلول 2^n می باشد. با فرض سنکرون بودن اتوماتای یادگیر سلولی، در زمان t هر سلول، i ، رشته های ژنومی خود و همسایگان خود را مورد بررسی قرار داده و بر اساس تابع ارزیابی از میان آنها دسته ای را به عنوان ژنومهای مناسب انتخاب می کند. این فرایند معادل انتخاب زوج در طبیعت می باشد. این شیوه انتخاب دو طرفه نمی باشد. به عبارت دیگر اگر یک ژنوم، ژنوم دیگری را به عنوان یکی از کاندیدهای خود انتخاب نماید، تضمینی برای انتخاب شدن از طرف ژنوم مقابل وجود نخواهد داشت. البته این روش معادل عینی در بین سیستمهای طبیعی ندارد. ژنوم بر اساس ژنومهایی که انتخاب نموده است، بردار سیگنال تقویتی را ساخته و به اتوماتاهای یادگیر خود می دهد. این مکانیسم در واقع معادل جفت گیری ژنوم مورد نظر می باشد. نکته ای که باید به آن توجه داشته باشیم این است که نحوه انتخاب ژنومها و تولید بردار تقویتی که قانون اتوماتای یادگیر سلولی را نشان می دهد، از پارامترهای بسیار مهم الگوریتم می باشد. پس از آن برای تولید X_{t+1}^i ، هر کدام از اتوماتاهای یادگیر مقدار مورد نظر خود را برای بیت متناظر خود در ژنوم اعلام می کنند. در صورتی که ژنوم جدید، new_{t+1}^i ، ارزش بیشتری نسبت به ژنوم قبلی داشته باشد، جایگزین آن می شود، در غیر این صورت سلول رشته ژنومی قبلی را حفظ می کند.

$$X_{t+1}^i = \begin{cases} X_t^i & f(X_t^i) > f(new_{t+1}^i) \\ new_{t+1}^i & f(X_t^i) \le f(new_{t+1}^i) \end{cases} \qquad (4)$$

این بخش از الگوریتم معادل یادگیری از تجارب قبلی ژنوم و حفظ آنها در نسلهای بعدی می باشد. این مکانیزم را می توان به نوعی معادل تاثیر یادگیری بر تکامل از دیدگاه تئوری بالدوین دانست. درشکل(۱ ((نمایی از یک سلول الگوریتم در یک شبکه تک بعدی (شکل(۲() دیده می شود.



شکل (1) نمایی از یک سلول در الگوریتم تکاملی مبتنی بر آتاماتای یادگیرسلولی



شکل (2) نمایی از یک شبکه سلولی استفاده شده الگوریتم تکاملی مبتنی بر آتاماتای یادگیرسلولی

قانون تعریف شده برای اتوماتای یادگیر سلولی مورد استفاده در الگوریتم پیشنهادی از دو بخش تشکیل شده است. بخش اول استراتژی انتخاب می باشد. منظور از استراتژی انتخاب سلول i ام، نحوه انتخاب تعداد مشخص Se، $1 \le Se \le m$، رشته ژنومی از میان m ژنوم همسایه، P^i، می باشد. در استراتژی مورد استفاده، Se ژنوم از بهترین ژنومهای P^i را انتخاب می کنیم وآن را P^i_{Se} می نامیم.

بخش دوم قانون استراتژی تولید سیگنال تقویتی با استفاده از مجموعه P^i_{Se} می باشد. بردار سیگنال تقویتی به روشهای گوناگون قابل محاسبه است. در این بخش به شرح روش مورد استفاده در این پایان نامه می پردازیم. فرض کنید $\overline{\beta}^i_t = (\beta^{i,1}_t,...,\beta^{i,n}_t)$ بردار مورد نظر باشد. در این صورت هرگاه $\overline{\beta}^i_t \in \{0,1\}^n$ محیط از نوع P می باشد. در چنین محیطی مقدار ۱ به عنوان جریمه و ۰ به عنوان پاداش در نظر گرفته می شود. در محیط Q، $\overline{\beta}^i_t$ می تواند به طور گسسته یک بردار از بردارهای محدود در فاصله $[0,1]^n$ و در محیط از نوع S، بردار تصادفی در فاصله $[0,1]^n$ باشد. در ادامه کار فرض ما بر P بودن محیط است.

اگر $X^i_t = (X^{i,1}_t,...,X^{i,n}_t)$ رشته ژنومی سلول i ام در زمان t باشد. برای هر متغیر $X^{i,j}_t$، $1 \le j \le n$ و به ازای مقادیر $k = 0,1$، $N_{i,j}(k)$ را به صورت زیر محاسبه می کنیم.

$$N_{i,j}(k) = \sum_l \delta_l (X^{i,j}_t = k \mid P^i_{Se}) \qquad (5)$$

به طوریکه اگر متغیر $X^{i,j}_t$ در i امین سلول، مقدار k را داشته باشد $\delta_l(X^{i,j}_t = k | P^i_{Se}) = 1$ و در غیر این صورت برابر صفر است. سپس با استفاده از $N_{i,j}(k)$ مقدار $\beta^{i,j}_t$ را بدست می آوریم.

$$\begin{aligned} \beta^{i,j}_t &= u(N_{i,j}(1) - N_{i,j}(0)) \quad \text{If} \quad X^{i,j}_t = 0 \\ \beta^{i,j}_t &= u(N_{i,j}(0) - N_{i,j}(1)) \quad \text{If} \quad X^{i,j}_t = 1 \end{aligned} \qquad (6)$$

که $u(.)$ تابع پله است. قانونی که در بالا آن را شرح دادیم، به فرم قوانین کلی می باشد. در شکل(۳(کد مجازی الگوریتم ارائه شده است.

```
Initialize.
While not done do
   For each cell i in CLA do in parallel
      Generate a new string genome
      Evaluate the new string genome
      If f(new string genome)> f(old string genome) then
         Accept the new string genome
      End if
      Select Se cells from neighbors of cell i
      Generate the reinforcement signal vector
      Update LAs of cell i
   End parallel for
End while
```

شکل (3) الگوریتم تکاملی مبتنی بر اتوماتای یادگیر سلولی

الگوریتم CLA-EC دارای مدل های متفاوتی است که هر یک از مدل ها در زیر توضیح داده شده است:

CLA-ECهمکارانه[۳۳]: کارایی CLA-EC بستگی زیادی به نرخ های یادگیری آتاماتای یادگیر در سلول های CLA-EC دارد.اگر نرخ یادگیری آتاماتای یادگیر بالا انتخاب شود، معمولاً الگوریتم به سرعت با بهره برداری از جواب های موجود به سمت جواب حرکت می کند ولی به دلیل این که فضای جستجو به طور کامل کاوش نمی شود ممکن است در نقاط بهینه محلی به دام بیفتد.در مقابل اگر نرخ یادگیری پایین انتخاب شود به دلیل این که کاوش در فضای جستجو به طور کامل صورت می پذیرد پاسخ به دست آمده توسط الگوریتم از دقت خوبی برخوردار است ولی در عوض سرعت همگرایی پایین است.یک روش برای حل مشکل فوق الذکر استفاده از چندین CLA-EC با نرخ های یادگیری متفاوت است.CLA-ECهای همکار از طریق به اشتراک گذاشتن جواب های میانی هم از سرعت بالا و هم از دقت بالا بهره می برد. CLA-EC همکارانه شامل دو CLA-EC با مشخصات کاملا یکسان ولی نرخ های یادگیری متفاوت برای آتاماتای یادگیر مستقر در سیستم می باشد.هر CLA-EC

در CLA-EC همکار به طور دوره ای جواب میانی به دست آمده در هر سلول را با جواب میانی سلول متناظر در CLA-EC دیگر ،تعویض می کند.به این ترتیب یکی از CLA-EC ها عمل کاوش و دیگری عمل بهره برداری را انجام می دهد.همکاری CLA-ECها حرکت سریع و دقیق به سمت جواب بهینه را باعث می شود.

CLA-EC ناهنگام[۳۲]: در CLA-EC همگام تمام سلول ها به طور همزمان فعال می شوند و هر سلول با مقادیر سلول های همسایه در تکرار قبلی سر و کار دارد و وضعیت نهایی سلول ها فقط به آتاماتاهای یادگیر بستگی دارد.با استفاده از بروز رسانی نا همگام وضعیت نهایی به ترتیب فعال سازی سلول ها نیز وابسته خواهد بود.همچنین در روش های به روزرسانی نا همگام برخلاف روش به روز رسانی همگام،هر سلول با مقادیر سلول های همسایه در تکرار فعلی سر و کار دارد.روش های فعال سازی به دو گروه روش های مبتنی بر زمان[۱۴] و روش های مبتنی بر گام[۱۵] تقسیم بندی می شوند. در روش های مبتنی بر گام ترتیب فعال سازی با استفاده از یکی از روش های زیر و یا هر روش دیگری تعیین می شود.در صورتی که در روش های مبتنی بر زمان ، زمان فعال سازی به طور صریح مشخص می شود.

ترتیب ثابت:سلول ها بر اساس یک ترتیب ثابت و از پیش تعیین شده فعال می شوند.

ترتیب تصادفی:در این روش ترتیب فعال سازی به این صورت ساخته می شود که اولین سلول به طور تصادفی از میان n سلول انتخاب می شود.سپس دومین سلول به طور تصادفی از میان n-۱ سلول باقی مانده انتخاب شده و به همین ترتیب ادامه می یابد تا تمام سلول ها انتخاب شوند.نکته ای که باید به آن توجه کرد این است که هنگام انتخاب ، تمام سلول های باقی مانده شانس یکسانی برای انتخاب شدن دارند.به عبارت دیگر،انتخاب بر اساس توزیع یکنواخت بدون جایگذاری است.

ترتیب جدید تصادفی:در این روش برای هر تکرار از یک ترتیب فعال سازی تصادفی بر مبنای توزیع یکنواخت با جایگذاری استفاده می شود.در این روش بر خلاف روش ترتیب تصادفی ترتیب فعال سازی در تمام تکرار ها یکسان نمی باشد.

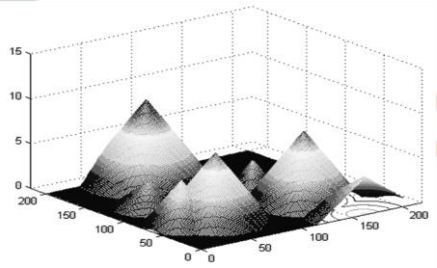
ترتیب یکنواخت:در این روش انتخاب سلول برای فعال سازی به طور تصادفی با توزیع یکنواخت انجام می گیرد.به عبارت دیگر از یک انتخاب تصادفی یکنواخت با جایگذاری استفاده می شود.در k تک گام ممکن است بعضی از سلول ها بیش از یکبار فعال شوند و بعضی اصلا فعال نشوند.در این روش بر خلاف روش های قبلی ،برای انتخاب سلول بعدی به حافظه اضافی نیاز نمی باشد.

CLA-EC بازترکیبی[۳۳]: در طول عملیات CLA-EC استاندارد ،آتاماتاهای یادگیر موجود در هر سلول،بردارهای احتمال خود را تغییر می دهند تا رشته های ژنومی محلی خوبی را برای نسل بعدی تولید کنند.فرض می شود که همه بیت های این رشته ژنومی محلی خوب بوده و برای یادگیری آتاماتای یادگیر استفاده خواهد شد.در اینجا رشته هایی وجود دارند که در حالت کلی کیفیت بالایی ندارند ولی جواب های جزیی خوبی را نگه می دارند. CLA-ECاستاندارد مکانیزمی برای جابه جایی رشته های جزیی میان سلول هایی که حاوی این رشته ها هستند ندارد. برای حل این مشکل از بازترکیب میان چند رشته در سلول ها استفاده شده است. در ضمن این الگوریتم به جستجوی بهتر فضا کمک می کند.

3. معیارهای کارایی

جدول (1) پارامترهای تابع قله های متحرک

Parameter	Value
number of peaks *m*	۱۰
Frequency of change *f*	every ۵۰۰۰ evaluations
height severity	۷.۰
width severity	۱.۰
peak shape	cone
shift length *s*	۱.۰
number of dimensions *D*	۵
cone height range *H*	[۳۰.۰, ۷۰.۰]
cone width range *W*	[۱, ۱۲]
cone standard height *I*	۵۰.۰
Search space range *A*	[۰, ۱۰۰]



شکل (4) نمایی از قله های متحرک

در محیط های پویا برای فهمیدن این که کدام رویکرد بر رویکرد دیگر اولویت دارد ، تنها لازم است که همگرایی را نمایش دهیم(بهترین و متوسط شایستگی هر نسل) و آن ها را به صورت بصری مقایسه کنیم.بسته به مسئله مورد نظر یکی از اندازه گیری های شمارشی ممکن است انتخاب خوبی برای مقایسه باشد.

کارایی درون خطی

به صورت میانگین همه ارزیابی ها بر روی کل اجرا محاسبه می شود.

$$x = \frac{1}{T}\sum_{t=1}^{T} e_t \qquad (7)$$

کارایی برون خطی

x^* به عنوان بهترین میانگین بدست آمده در هر مرحله محاسبه می شود.

$$x^* = \frac{1}{T}\sum_{t=1}^{T} e_t^* \; with \; e_t^* = \max\{e_1, e_2, \ldots, e_t\} \qquad (8)$$

معمولا فرض می شود که بهینه سازی در محیط شبیه سازی شده انجام می شود و تنها بهترین راه حل به دنیای واقعی وارد می شود.کارایی برون خطی باید تنها به افرادی رسیدگی کند که تا آخرین تغییر در محیط ارزیابی شده اند.

$$x' = \frac{1}{T}\sum_{t=1}^{T} e_t' \; with \; e_t' = \max\{e_\tau, e_{tau+1}, \ldots, e_t\} \qquad (9)$$

τ being the last time step<t

که در آن یک تغییر در محیط روی داده است.البته این نیازمند این است که تغییرات محیطی برای مشاعده کننده شناخته شده باشد.

4. آزمایشات انجام شده

یک استراتژی تکاملی کاملاً رایج تطبیق نرخ جهش بعد از تغییر محیط است.حرکت به این صورت است که نرخ جهش باید بعد از یک تغییر در محیط افزایش یابد تا به جمعیت همگرا شده کمک کند تا خارج گردد و جستجو از ابتدا آغاز شوددر این مقاله در قسمت اول به حل مساله قله های متحرک با استفاده از الگوریتم ژنتیک وفقی می پردازیم و سپس مساله قله های متحرک را با انواع مدل های CLA-EC حل کرده و به مقایسه جواب های بدست آمده می پردازیم.سپس برای بهبود عملکرد الگوریتم های CLA-EC به معرفی مدل حافظه/جستجو می پردازیم.

برای آزمایشات گزارش شده در این مقاله الگوریتم های CLA-EC طراحی شده با الگوریتم های ژنتیک با جهش وفقی مورد مقایسه قرار می گیرند.در اینجا برای الگوریتم ژنتیک مورد استفاده از کدینگ حقیقی،جایگزینی تولیدی با نخبه گرایی ۱ و تقاطع دو نقطه ای با احتمال ۰.۶ و جهش وفقی با احتمال $\frac{1}{n}$ که n تعداد ابعاداست (n=۵) استفاده می کنیم. در الگوریتم CLA-EC از ۱۰۰ سلول استفاده می شود.هر یک از توابع دارای ۵ بعد هستند و برای انکد کردن هر بعد از ۲۷ بیت استفاده شده است، که این ، به این معنی است که هر سلول دارای ۱۳۵ آتاماتای یادگیر است.برای سهولت در ارئه از نماد CLA-EC(LRI(a,b),r,se,q) برای اشاره کردن به الگوریتم CLA-EC با q سلول،شعاع همسایگی r ،تعداد سلول انتخابی se ،آتاماتای یادگیر با پارامترهای پاداش a وجریمه b است.در آزمایشات انجام شده توسط CLA-EC، از یک اتوماتای یادگیر سلولی خطی دارای ۱۰۰ سلول و همسایگی با شعاع یک دارای دور استفاده شده است. همچنین از اتوماتای یادگیر LRI با نرخ پاداش ۰.۰۱(یعنی نرخ جریمه صفر است) استفاده شده است. تعداد گام در هر دو الگوریتم، ۵۰۰۰ تکرار است. پیاده سازی ها روی ۳۰ اجرا میانگین گرفته شده است.از آن جایی که برای توابع شایستگی پویا گزارش بهترین راه حل به دست آمده ، مناسب نخواهد بود ، در این جا میانگین خطای برون خطی و کارایی برون خطی گزارش شده است. CLA-EC همکارانه شامل دو CLA-EC خطی ساده $CLA-EC_L$ و $CLA-EC_H$ هر کدام با شعاع همسایگی ۱ می باشد. $CLA-EC_H$ شامل آتاماتاهای یادگیر L_{RI} با نرخ یادگیری بالا و $CLA-EC_L$ شامل آتاماتاهای یادگیر L_{RI} با نرخ یادگیری پایین می باشد.طول هر دوره ۱۰ تکرار در نظر گرفته شده است. هر CLA-EC همکارانه با qسلول ، شامل دو CLA-EC ساده با q/۲ سلول است و پس از گذشت هر دوره زمانی از پیش تعیین شده،هر $CLA-EC_H$ سلول X_i خود را با سلول X_i متناظر در $CLA-EC_L$ معاوضه می کند.

در مرحله دوم از آزمایشات انواع مدل های دیگر CLA-EC را در محیط های پویا استفاده کردیم. همان طور که مشاهده می شود،کارایی الگوریتم CLA-ECاستاندارد و CLA-EC نا همگام شبیه به هم است و هر دو بعد از چند نسل در بهینه های محلی گیر افتاده و توانایی خروج از آن را ندارند،هر چند که در[۳۲]در بیشتر موارد کارایی CLA-EC ناهمگام در مسائل با محیط ایستا بالاتر از CLA-ECاستاندارد بوده است.و همین طور نشان داده شده است که در مسائل ایستا با این که CLA-EC استاندارد سریعتر ازCLA-EC ناهمگام همگرا می شود ولی برای این کار به تعداد سلول بیشتری نیاز داردو در غیر این صورت در بهینه های محلی گیر می افتد.ولی در مسائل با محیط پویا کارایی هر دو الگوریتم CLA-ECاستاندارد و ناهمگام مثل یکدیگر است و هیچ یک توانایی وفق پذیری با تغییرات محیط را ندارند.

الگوریتم دیگری که در این جا مورد مقایسه قرار گرفته است ، الگوریتم CLA-EC بازترکیبی است. این الگوریتم نسبت به الگوریتم CLA-EC استاندارد و CLA-EC نا همگام کارایی بالاتری داشته و در نتیجه دارای خطای کمتری است. دلیل این امر ،عملگر بازترکیبی است که به الگوریتم CLA-EC اضافه شده است.عملگر بازترکیبی قدرت اکتشاف الگوریتم را بالا می برد و همین طور اجازه می دهد که ساختارهای جزیی میان کروموزوم های افراد جابه جا شود.این تغییر در الگوریتم CLA-EC باعث می شود تا فضا به خوبی جستجو شود و الگوریتم در بهینه خای محلی گیر نیفتد

بهترین الگوریتمی که در میان الگوریتم های CLA-EC وجود دارد، الگوریتم CLA-EC همکارانه است. این الگوریتم دارای قدرت اکتشاف بسیار خوبی است و با تغییرات محیطی خود را وفق می دهد.البته در نسل های ابتدایی این الگوریتم نسبت به الگوریتم ژنتیک با جهش وفقی بد تر عمل می کند ولی در نسل های انتهایی از لحاظ کارایی به الگوریتم ژنتیک با جهش وفقی می رسد و کم کم کارایی آن بالاتر نیز می رود.

5. الگوریتم CLA-EC همکارانه بهبودیافته

با وجود این که الگوریتم CLA-EC همکارانه تا اندازه ای از به دام افتادن در مینیمم محلی جلوگیری می کند ولی باز هم احتمال به دام افتادن در مینیمم محلی بعد از گذشت هر نسل افزایش می یابد.زیرا CLA-EC با نرخ یادگیری پایین($CLA-EC_L$)نیز، هنگامی که جمعیت موجود در CLA-EC دیگر را در اختیار می گیرد ، آتاماتاهای موجود در $CLA-EC_L$ با استفاده از این جمعیت جدید وضعیت خود را به روز رسانی می کنند، و سعی می کنند به سمت وضعیت آتاماتاهای یادگیر با نرخ یادگیری بالا($CLA-EC_H$) به پیش بروند. البته چون نرخ یادگیری آن ها پایین است در تعداد نسل کم نمی توانند مشکلی ایجاد کنند، در نتیجه یک جمعیتی را به صورت تصادفی درست می کنند و در دوره تعویض بعدی در اختیار $CLA-EC_H$ قرار می دهند.در نتیجه آتاماتای یادگیر $CLA-EC_H$ کمی از خطر به دام افتادن در بهینه های محلی نجات می یابندو سعی می کنند در فضای جدید به دست آمده هم جستجویی داشته باشند، ولی این نجات دادن فقط برای یکبار آن هم در هنگام تعویض جمعیت حاصل می شودو باز بعد از گذشت چند نسل دوباره $CLA-EC_H$ با استفاده از آتاماتای یادگیر خودبه جمعیتی مشابه جمعیت قبلی خود می رسدو باز این روند تکرار می شود.حال اگر الگوریتم نتواند بعد از گذراندن تعداد نسل معینی به جواب برسد،به طور ناخواسته وضعیت آتاماتاهای یادگیر موجود در $CLA-EC_L$ هم به سمت وضعیت آتاماتاهای یادگیر موجود در $CLA-EC_H$میل کرده و $CLA-EC_L$هم دیگر قادر نیست با تولید تصادفی جمعیت روی فضای راه حل ها ، یک جستجوی عمومی را به راه بیاندازد.در نتیجه با استفاده از قدرت یادگیری آتاماتاهای خود سعی می کند جمعیتی مشابه با جمعیت موجود در $CLA-EC_H$را تولید کند.بدین ترتیب به تدریج از خاصیت نجات بخشی آن کاسته می شودو دوباره همان مشکل به دام افتادن در مینیمم های محلی به پیش می آید.

در[۳۵]به جای این که از یک CLA-EC با نرخ یادگیری پایین به عنوان یک همکار برای CLA-EC با نرخ یادگیری بالا استفاده شود، از یک الگوریتم ژنتیک استاندارد به عنوان همکار استفاده شده است تا با تنظیم پارامتر جهش،احتمال تولید جمعیت تصادفی و جستجوی عمومی روی کل فضای راه حل های مسئله را برای همیشه تضمین کند.این الگوریتم ژنتیک بر روی جمعیت موجود در یک شبکه سلولی عمل خواهد کرد،با این تفاوت که در این جا سلول ها دیگر کاری به وضعیت سلول های همسایه اشان ندارندو صرفا بر مبنای الگوریتم ژنتیک به روزرسانی می شوند.در این جا نیز مانند مدل همکارانه هر دو الگوریتم هر چند نسل یکبار جواب های میانی خود را به اشتراک می گذارند.در نتیجه CLA-EC با نرخ یادگیری بالا می تواند مشابه قبل با سرعت در فضای جواب ها حرکت کند و هرگز در بهینه های محلی به دام نخواهد افتاد.این الگوریتم جدید علاوه بر این که ما را از بهینه های محلی نجات می دهد، دارای سرعت بیشتری نسبت به الگوریتم CLA-EC ساده یا همکارانه نیز می باشد.چون در این الگوریتم ، جمعیت موجود در الگوریتم ژنتیک هم می توانند در حین جستجو در فضای عمومی با سرعت بیشتری نسبت به CLA-EC با نرخ یادگیری پایین به سمت جواب حرکت کنند.نکته قابل توجه دیگر در مورد این الگوریتم این است که ، با استفاده از تعداد جمعیت کم ، خیلی سریعتر از الگوریتم همکارانه و ساده به جواب می رسد.زیرا با استفاده از پارامترهای بازترکیبی و جهش نیز می تواند شانس خود را در رسیدن به جواب آزمایش کند.همچنین وابستگی این الگوریتم نسبت به مقدار پارا متر نرخ یادگیری نیز از مدل های مشابه قبلی کمتر می باشد.این الگوریتم را در زیر مشاهده می کنید:

```
Initialize
While not done do
    t=t+۱
```

```
For each cell i in CLA-EC_H do
    Generate new genome
    Evaluate new genome
    If f(new genome)>f(old genome) then
        Accept new genome
    End if
    Select se cells from neighbor
    Generate the reinforcement signal vector
    Update Las
    Generate new population for with GA
    If (t mod N) =۰ then
        Exchange the genome of cell i in GA and
        CLA-EC_H
    End for
End while
```

شکل (5) الگوریتم CLA-EC همکارانه بهبودیافته

6. رهیافت پیشنهادی

همان طور که در ۱قسمت ۵ بیان شد ، برای بهبود عملکرد CLA-EC همکارانه می توان به جای همکار با نرخ یادگیری پایین ، از یک الگوریتم ژنتیک استاندارد استفاده کرد. همان طور که در[۳۵]نشان داده شده است این الگوریتم نسبت به الگوریتم CLA-ECهمکارانه بهتر عمل می کند.اما این الگوریتم برای مسائل با محیط ایستا طراحی شده است و الگوریتم ژنتیک استاندارد در مسائل با محیط پویا خوب عمل نکرده و در همان نسل های اولیه در بهینه محلی می افتد.به همین جهت الگوریتم همکارانه ای که برای مسائل پویا طراحی کردیم، از یک CLA-EC با نرخ یادگیری بالا تشکیل شده است و از یک الگوریتم ژنتیک با جهش وفقی به عنوان همکار دیگر به جای CLA-EC با نرخ یادگیری پایین استفاده کردیم. این الگوریتم ژنتیک بر روی جمعیت موجود در یک شبکه سلولی عمل خواهد کرد،با این تفاوت که در این جا سلول ها دیگر کاری به وضعیت سلول های همسایه اشان ندارندو صرفا بر مبنای الگوریتم ژنتیک به روزرسانی می شوند.در این جا نیز مانند مدل همکارانه هر دو الگوریتم هر چند نسل یکبار جواب های میانی خود را به اشتراک می گذارند.در نتیجه CLA-EC با نرخ یادگیری بالا می تواند مشابه قبل با سرعت در

SEA	CLA-EC	Cooperative CLA-EC	New Cooperative CLA-EC	Recombinative CLA-EC	Uniform CLA-EC
۱۶.۷۳۵۸۲	۴۵.۱۵۶۵۷	۱۳.۴۱۴۷	۲.۷۳۳۹۵	۳۹.۹۷۳۸	۴۴.۹۷۶۹۸

فضای جواب ها حرکت کند و هرگز در بهینه های محلی به دام نخواهد افتاد.

نرخ جهش و بازترکیبی مورد استفاده در الگوریتم ژنتیک با جهش وفقی به ترتیب برابر ۰.۰۱و۰.۸ در نظر گرفته شده است. CLA-EC(LRI(۰.۰۱,۰),SEA,۱,۲,۱۰۰) نشان دهنده الگوریتم CLA-EC همکارانه است ، که یک همکار CLA-ECبا نرخ یادگیری ۰.۰۱ و شعاع همسایگی یک و همکار دیگر یک الگوریتم ژنتیک وفقی است، که الگوریتم مجموعا بر روی ۱۰۰ سلول کار می کند یعنی هر همکار بر روی ۵۰ سلول کار می کند.در جدول(۲(خطای برون خطی الگوریتم ژنتیک با جهش وفقی و انواع مدل های CLA-EC مورد مقایسه قرار گرفته است. همان طور که مشاهده می شود الگوریتم CLA-EC همکارانه بهبود یافته دارای کمترین خطا است.

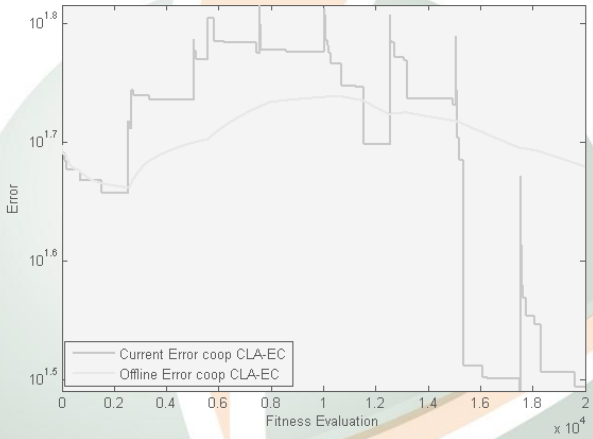
جدول (2) مقایسه خطای برون خطی روی تابع قله متحرک با الگوریتم ژنتیک با جهش وفقی و انواع مدل های CLA-EC در ۵۰۰۰ تکرار

7. بررسی سرعت همگرایی در الگوریتم CLA-EC همکارانه بهبود یافته و الگوریتم ژنتیک با جهش وفقی

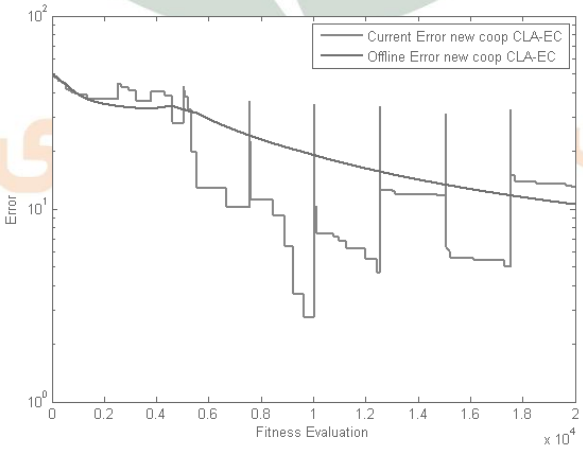
در این بخش هدف این است که با بررسی سرعت همگرایی به این نتیجه برسیم کدام الگوریتم پس از بروز تغییراتی در سیستم می تواند سریعتر بهترین پاسخ خود را تولید نماید و آنرا در اختیار سیستم قرار دهد که دارای کمترین خطا باشد. این پارامتر در سیستم های پویا به دلیل شرایط خاص این محیط ها از اهمیت بالایی برخوردار است اما به دلیل پیچیدگی هایی که دارد کمتر مورد بررسی قرار گرفته است.

برای این منظور از میزان خطای جاری استفاده شده است. در هربار گردش برنامه این میزان خطا در متغیری ذخیره شده سپس این مقادیر بصورت نموداری نشان داده خواهند شد. هر چه این مقدار خطا کمتر بوده و به صفر نزدیکتر باشد به آن معنی است که قله یافت شده به بلندترین قله نزدیکتر است، حال هر الگوریتمی که بتواند با شیب تندتری به کمترین مقدار تولیدی توسط الگوریتم برسد می توان نتیجه گرفت سرعت این الگوریتم بهتر بوده و می تواند بعد از بروز تغییرات جوابهای بهتری را با سرعت بیشتر تولید نماید.

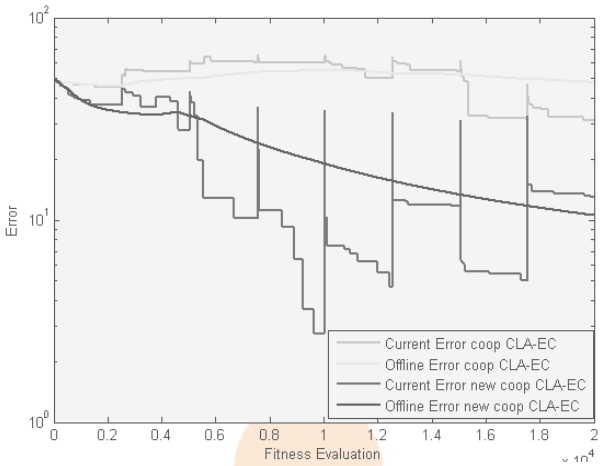
درشکل(۶(تاشکل(۸(به بررسی خطا در الگوریتم های SEA وNew Cooperative CLA-EC پرداخته ایم. در هر یک از نمودارها میزان خطای جاری و Offline در یک گردش بیست هزار بار برای ارزیابی برازش تابع نشان داده شده است، پالس های نامنظم نشان دهنده ایجاد یک تغییر و حرکت برای رسیدن به جواب است تا بروز تغییر بعدی، این عمل در هر ۵۰۰۰ بار ارزیابی تابع یکبار رخ می دهد که بصورت بازه هایی در شکل ها نشان داده شده است.



شکل (6) ارزیابی خطا در الگوریتم cooperative CLA-EC با F=۵۰۰۰ برای بیست هزار ارزیابی اول



شکل (7) ارزیابی خطا در الگوریتم new cooperative CLA-EC با F=۵۰۰۰ برای بیست هزار ارزیابی اول



شکل (8) مقایسه خطای الگوریتم های new cooperative CLA-EC وcooperative CLA-EC در F=۵۰۰۰ برای بیست هزار ارزیابی اول

8. نتیجه گیری

در این مقاله مدل محاسبات تکاملی بر پایه اتوماتای یادگیر سلولی معرفی شد و مورد بررسی قرار گرفت. نتایج آزمایشات صورت گرفته بر روی این مدل نشان دادند که مدل CLAEC در حل مسایلی با فضای حالت بزرگ به ویژه زمانی که فضای جستجو خلوت باشد، عملکرد مناسبی ندارد و نسبت به الگوریتمهای ژنتیک بدتر عمل میکند.

نکته قابل توجه این است که ایجاد تغییرات گسترده در ساختار ژنومها در هر گام، به دلیل ماهیت این مدل است. در الگوریتمهای ژنتیک با انتخاب روشهای درست بازترکیبی و جهش، سعی میشود که تغییرات ژنومها، بسیار نرم و به آهستگی صورت گیرند. به عنوان مثال، در روش بازترکیبی تک نقطه ای، بخش از ژنومهای والد، ثابت میمانند و فقط قسمتی از ژنوم تغییر میکند. ولی در مدل CLAEC، پیش از همگرا شدن اتوماتای یادگیر سلولها، در هر گام کل ژنوم به صورت تصادفی تغییر میکند. این مساله باعث میشود که همگرا شدن الگوریتم نرخی بسیار کند داشته باشد و در مقایسه با الگوریتم ژنتیک بدتر عمل کند.مطالعات انجام شده نشان داد که الگوریتم CLA-EC استاندارد ، CLA-EC بازترکیبی، CLA-EC آسنکرون دارای عملکرد خوبی نیستند ولی CLA-EC همکارانه به دلیل توازنی که میان خاصیت اکتشاف و بهره برداری ایجاد می کند،دارای کارایی بالاتر است.الگوریتم های CLA-EC استاندارد، بازترکیبی و آسنکرون به دلیل این که نرخ همگرایی اشان بسیار کند قدرت وفق پذیری با تغییرات محیط را ندارند.ولی در الگوریتم CLA-EC همکارانه نرخ همگرایی افزایش پیدا کرده و نسبتا بهتر می توانند با تغییرات محیط تطبیق پیدا کنند.در واقع CLA-ECهمکارانه شامل یک حافظه ضمنی است که باعث نگه داری تنوع در جمعیت و تطبیق بهتر با تغییرات می شود. CLA-EC همکارانه معرفی شده نسبت به CLA-EC همکارانه ابتدایی دارای این خصوصیت است که عملکرد آن وابستگی زیادی به تنظیم نرخ یادگیری و تعداد سلول ها ندارد، بنابراین به راحتی قادر است خود را با تغییرات محیط تطبیق دهد.

9. مراجع

[1] Louis, S. J., and XU, Z., "*Genetic Algorithms for Open Shop Scheduling and Re-scheduling*," In M. E. Cohen and D. L. Hudson , Editors, ISCA ۱۱th International Conference on Computers and Their Application, pp. ۹۹-۱۰۲, ۱۹۹۶.

[2] Cartwright, H. M., and Tusen, A. L., "*Genetic Algorithms and Flow Shop Scheduling: Towards the Development of a Real-time Process Control System*," T. C. Fogarty, Editor, AISB Workshop on Evolutionary Computing, vol. ۸۶۵ of LNCS, pp. ۲۷۷-۲۹۰, Springer Verlag, ۱۹۹۴.

[3] Bierwirth, C., and Kopfer, H., "*Dynamic Task Scheduling with Genetic Algorithms in Manufacturing Systems*," Technical Report, Department of Economics, University of Bremen, Germany, ۱۹۹۴.

[4] Bierwirth, C., Kopfer, H., Mattfeld, D. C., and Rixen, I., "*Genetic Algorithm Based Scheduling in a Dynamic Manufacturing Environment*," Proceedings of IEEE Conference on Evolutionary Computation, IEEE press, ۱۹۹۵.

کنفرانس داده کاوی ایران

[5] Pico, C. A. G., and Wainwright, R. L., "*Dynamic Scheduling of Computer Tasks Using Genetic Algorithms*," Proceedings of First IEEE Conference on Evolutionary Computation, vol. ۲, pp. ۸۲۹-۸۳۳, ۱۹۹۴.

[6] Fang, H. -L., Ross, P., and Corn, D., "*A Promising Genetic Algorithm Approach to Job-shop Scheduling, and Open-shop Scheduling Problems*," In S. Forrest, Editor, Fifth International Conference on Genetic Algorithms, pp. ۳۷۵-۳۸۲, ۱۹۹۳.

[7] Krishnakumar, K., "*Micro Genetic Algorithms for Stationary and Non-stationary Function Optimization*," In intelligent control and adaptive systems, of the SPIE, vol. ۱۱۹۶, pp. ۲۸۹-۲۹۶, ۱۹۸۹.

[8] Cobb, H. G., "*An Investigation into the Use of Hyper Mutation as an Adaptive Operator in Genetic Algorithms Having Continuous, Time-dependent Non stationary*," Technical Report AIC-۹۰-۰۰۱, proc in. Naval Research Laboratory, Washington, USA, ۱۹۹۰.

[9] Grefenstette, J. J., "*Genetic Algorithms for Changing Environments*," In R. Maenner and B. Manderick, Editors, Parallel Problem Solving From Nature ۲, pp. ۱۳۷-۱۴۴, North Holland,۱۹۹۲.

[10] Cobb, H. G., and Grefenstette, J. J., "*Genetic Algorithms for Tracking Changing Environment*," Proceedings of ۵th International Conference on Genetic Algorithms, pp. ۵۲۳-۵۳۰, ۱۹۹۳.

[11] Vavak, F., Jukes, K. A., and Fogarty, T. C., "*Performance of a Genetic Algorithms with Variable Local Search Range Relative to Frequency for the Environmental Changes*," Proceedings of Conference on Genetic Programming, ۱۹۹۸.

[12] Vavak, F., Fogarty, T. C., and Jukes, K., "*Learning the Local Search Range for Genetic Control of Dynamic Systems*," Proceedings of Pre-Conference Workshop on Evolutionary Computing and Machine Learning, pp. ۱۴۳-۱۵۰, ۱۹۹۶.

[13] Back, T., "*On the Behavior of Evolutionary Algorithms in Dynamic Environments*," Proceedings of IEEE International Conference on Evolutionary Computation, pp. ۴۴۶-۴۵۱.IEEE, ۱۹۹۸.

[14] Ng, K. P., and Wong, K. C., "*A New Diploid Scheme and Dominance Change Mechanism for Non-stationary Function Optimization*," Proceedings of ۶th international conference on genetic algorithms, pp. ۱۵۹-۱۶۶, ۱۹۹۵.

[15] Ryan, C., "*Diploidy Without Dominance*," In J. T. Alander, Editor, Third Nordic Workshop Mon Genetic Algorithms, pp. ۶۳-۷۰, ۱۹۹۷.

[16] Dasgupta, D., and McGregor, D. R., "*Non stationary Function Optimization Using the Structured Genetic Algorithm*," In R. Manner and B. Manderick, Editors, Parallel Problem Solving From Nature, pp. ۱۴۵-۱۵۴, Elsevier science publisher, ۱۹۹۲.

[17] Dasgupta, D., "*Incorporating Redundancy and Gene Activation Mechanisms in Genetic Search*," In L. chambers, Editor, Practical Handbook of Genetic Algorithms, vol. ۲, pp. ۳۰۳-۳۱۶, CRC Press, ۱۹۹۵.

[18] Ohkura, K., and Ueda, K., "*Adaptation in Dynamic Environment Using Genetic Algorithms with Redundant Representation and Additional Genetic Operators*," In Dagli and Cihan, Editors, Intelligent Engineering Systems through Artificial Neural Networks, pp. ۲۹۱-۲۹۶, ۱۹۹۴.

[19] Trojanowski, K., and Michalewicz, Z., "*Searching for Optima in Non-stationary Environments*," Proceedings of Congress on Evolutionary Computation, vol. ۳, pp. ۱۸۴۳-۱۸۵۰, IEEE, ۱۹۹۹.

[20] Trojanowski, K., and Michalewicz, Z., "*Searching for Optima in Non-stationary Environments*," Proceedings of Congress on Evolutionary Computation, vol. ۳, pp. ۱۸۴۳-۱۸۵۰, IEEE, ۱۹۹۹.

[21] Goldberg, D. E., and Richardson, J., "*Genetic Algorithms With Sharing for Multimodal Function Optimization*," Proceedings of Second International Conference on Genetic Algorithms, pp. ۴۱-۴۹, ۱۹۸۷.

[22] Andersen, H. C., "*An Investigation in to Genetic Algorithms, and the Relationship between Speciation and the Tracking of Optima in Dynamic Functions*," Honours thesis, Queensland university of technology, Brisbane, Australia, November ۱۹۹۱.

[23] Hocaglu, C., and Sanderson, A. C., "*Planning Multi-paths Using Speciation in Genetic Algorithms*," Proceedings of ۳rd IEEE Intl. Conf. on Evolutionary Computation, pp. ۳۷۸-۳۸۳, ۱۹۹۶.

[24] Cedeno, W., and Vemuri, V. R., "*On the Use of Niching for Dynamic Landscapes*," Proceedings of International Conference on Evolutionary Computation, IEEE, ۱۹۹۷.

کنفرانس داده کاوی ایران

[25] Liles, W., and Jong, D., "*The Usefulness of Tag Bits in Changing Environments*," Proceedings of Congress on Evolutionary Computation, vol. ۳, pp. ۲۰۵۴-۲۰۶۰. IEEE, ۱۹۹۹.

[26] Spears, W., "*Simple Subpopulation Schemes*," Proceedings of evolutionary programming conference, pp. ۲۹۶-۳۰۷, World Scientific, ۱۹۹۴.

[27] Oppacher, F., and Wineberg, M., "*The Shifting Balance Genetic Algorithm: Improving the GA in a Dynamic Environment*," Proceedings of Genetic and Evolutionary Computation Conference, vol. ۱, pp. ۵۰۴-۵۱۰, ۱۹۹۹.

[28] Wineberg, M., and Oppacher, F., "*Enhancing the GA's Ability to Cope with Dynamic Environments*," Proceedings of Genetic and Evolutionary Computation Conference, pp. ۳-۱۰, ۲۰۰۰.

[29] Ursem, R. k., "*Multinational GA Optimization Techniques in dynamic Environments*," Proceedings of Genetic and Evolutionary Computation Conference, pp. ۱۹-۲۶, ۲۰۰۰.

[30] Stephens, C. R., Olmedo, I. G., Vargas, J. M., and Waelbroeck, H., "*Self-adaptation in Evolving Systems*," In Artificial life, ۴(۲), ۱۹۹۸.

[31] Weiker, K., and Weiker, N., "*Dynamic Rotation Partial Visibility*," Proceedings of congress on evolutionary computation, pp. ۱۱۲۵-۱۱۳۱, ۲۰۰۰.

[32] Masoodifar, B., Meybodi, M. R., and Rastegar, R., "*Asynchronous CLA-EC*," Proceedings of ۱۱th Annual CSI Computer Conference of Iran, Fundamental Science Research Center (IPM), Computer Science Research Lab., Tehran, Iran, pp. ۴۴۷-۴۵۸, Jan. ۲۴-۲۶, ۲۰۰۶.

[33] Masoodifar, B., Meybodi, M. R., and Hashemi, M., "*Cooperative CLA-EC*," Proceedings of ۱۳th Annual CSI Computer Conference of Iran, Shahid Beheshti University, Tehran, Iran, pp. ۵۵۸-۵۵۹, Feb. ۲۰-۲۲, ۲۰۰۷.

[34] Jafarpourand, B., and Meybodi, M. R., "*Recombinative CLA-EC*," Proceedings of Annual IEEE International Conference on Tools with Artificial Intelligence (ICTAI ۲۰۰۷), pp. ۴۱۵-۴۲۲, Achaia, Greece, Oct. ۲۰۰۷.

[35] محمدجواد فتاحي حسن آباد و محمدمهدي عبادزاده « **بهبود مدل CLA-EC با استفاده از الگوريتم ژنتيك**» اولين كنفرانس ملي كاربردهاي مهندسي نرم افزار، لاهيجان، 1388.

زيرنويس ها

[۱] Restriction
[۲] Optimum
[۳] Converge
[۴] Diversity
[۵] Exploring
[۶] Modifying selection
[۷] Multi-population approaches
[۸] Dynamic bit-matching
[۹] Moving parabola
[۱۰] Time-varying knapsack problem
[۱۱] Moving peaks function
[۱۲] Scheduling problem
[۱۳] Oscillating peaks
[۱۴] Time-driven
[۱۵] Step-driven

کنفرانس داده کاوی ایران