# CLA-PSO: یک مدل ترکیبی برای بهینه سازی

محمد شیبانی  محمد رضا میبدی
آزمایشگاه محاسبات نرم
دانشکده مهندسی کامپیوتر و فناوری اطلاعات
دانشگاه صنعتی امیرکبیر
تهران ایران
meybodi@ce.aut.ac.ir  sheybani@ce.aut.ac.ir

**چکیده : در این مقاله یک مدل جدید که از ترکیب اتوماتای یادگیر سلولی و مدل حرکت دسته جمعی ذرات حاصل میشود برای بهینه سازی پیشنهاد میگردد. در مدل پیشنهادی در هر سلول از اتوماتای یادگیرسلولی یک جمعیت از ذرات قرار داده میشود. اتوماتای یادگیر در هر سلول وظیفه تنظیم رفتار ذرات جمعیت ان سلول و برقراری موازنه بین فرایند جستجوی سراسری و جستجوی محلی را بر عهده دارد. برای انجام این وظیفه، اتوماتای یادگیر موجود در هر سلول از تجربیات خود و اتوماتای یادگیر سلولهای همسایه بهره میبرد. نتایج آزمایشها بر روی مسایل نمونه نشان میدهد که روش ارائه شده از عملکرد بهتری در مقایسه با مدل PSO استاندارد برخوردار است.**

**کلید واژه**- حرکت دسته جمعی ذرات، اتوماتای یادگیر سلولی

## ۱- مقدمه

حرکت دسته جمعی ذرات[1] یک تکنیک کارا برای حل مسایل بهینه سازی است که بر مبنای قوانین احتمال و بر اساس جمعیت کار میکند. در این روش هر یک از اعضای جمعیت که ذره نام دارند سعی میکنند با تنظیم مسیر خود و حرکت به سمت بهترین تجربه شخصی و بهترین تجربه جمعی گروه به سوی راه حل نهایی حرکت کنند. از آنجا که در این الگوریتم ذرات به تدریج به سمت بهترین راه حل پیدا شده تا به حال میل میکنند اگر این راه حل، یک بهینه محلی باشد ذرات همگی به سمت آن میروند و الگوریتم استاندارد PSO راهکاری برای خروج از این بهینه محلی ارائه نمیدهد. این بزرگترین مشکل PSO استاندارد است که سبب میشود در حل مسایل چند قله ای مخصوصا با فضای حالت بزرگ ناتوان باشد.

یکی از راه حلها برای مقابله با مشکل بهینه های محلی در الگوریتم حرکت دسته جمعی ذرات، استفاده از جهش[2] است [۶][۷][۸]. در [۶] و [۷] از جهش گاوسی در PSO استاندارد و همچنین نسخه هایی تغییر یافته آن استفاده شده است و با آزمایش بر روی توابعی حداکثر با دو بعد نشان داده شده است که راه حلهای ارایه شده پاسخهای بهتری در مقایسه با PSO ستاندارد تولید میکنند. در [۸] برای جهش از توزیع کوشی استفاده شده است. به این ترتیب که هر ذره با احتمال $P_{mutate}$ میتواند جهش پیدا کند. هرگاه ذره ای برای جهش انتخاب شد، هر یک از اجزای بردار آن ذره به احتمال $1/d$ که $d$ ابعاد مساله است، ممکن است دچار جهش شوند. برای جهش دادن هر جزء از یک ذره یک عدد به صورت تصادفی از توزیع کوشی انتخاب میشود و با مقدار جزء مربوطه از بردار ذره جمع میشود. روش ارائه شده برای برخی مسایل با فضای حالتهای بزرگ، جواب مناسبی داشته است.

در الگوریتم حرکت دسته جمعی ذرات استاندارد، در محاسبه سرعت ذره در گام بعد، کل سرعت فعلی ذره محاسبه میشود. در واقع سرعت ذره در هر گام از دو قسمت تشکیل میشود که قسمت اول سرعت فعلی ذره و قسمت دوم مربوط به دنبال کردن بهترین تجربه شخصی و بهترین تجربه گروه است. بدون قسمت دوم الگوریتم حالت یک جستجوی سراسری کورکورانه را خواهد داشت

و بدون قسمت اول، الگوریتم به جستجویی محلی در نزدیکی بهترین ذره تبدیل خواهد شد که در رسیدن به قسمتهای زیادی از فضای جستجو ناتوان خواهد بود. الگوریتم حرکت دسته جمعی ذرات با ترکیب این دو قسمت، سعی میکند که به نوعی تعادل را بین جستجوی سراسری و محلی ایجاد نماید.

در [۵] پارامتری با نام وزن میانی تعریف شده است. وزن میانی در واقع ضریبی از سرعت فعلی ذره است که در تعیین سرعت ذره در گام بعد مورد استفاده قرار میگیرد. با تنظیم این پارامتر میتوان تعادل بین جستجوی محلی و جستجوی سراسری برقرار نمود. در برخی از کاربردها وزن میانی ثابت در نظر گرفته شده است. ولی معمولا وزن میانی در ابتدا بایستی مقدار بالاتری داشته و با گذشت زمان مقدار آن به صورت خطی کاهش یابد و به این ترتیب ذرات در ابتدا فرایند جستجو بیشتر سعی در شناسایی محلهای جدید از فضای جستجو دارند ولی با گذشت زمان میل بیشتری به دنباله روی پیدا میکنند. کاهش مقدار وزن میانی به صورت خطی روشی بسیار ساده است که توانایی بکارگیری کامل دانش اولیه راجع به مساله و همچنین استفاده از بازخورد سیستم را ندارد.

برخی از محققان سعی کرده اند تا با افزودن نویز به حرکات ذرات در الگوریتم استاندارد PSO، باعث حرکات متنوعتری شوند و بدین صورت مشکل بهینه های محلی را حل کنند[۹][۱۰]. برخی دیگر از راه حلها برای حل مشکل بهینه های محلی از طریق ترکیب الگوریتم PSO با الگوریتمهای دیگر مانند الگوریتمهای ژنتیک[۳] و روش بالا رفتن در امتداد شیب[۴] و الگوریتم سرد کردن تدریجی[۵] حاصل شده است[۱۱][۱۸].

یک اتوماتای یادگیر[۳]، ماشینی است که میتواند تعدادی متناهی عمل را انجام دهد. هر عمل انتخاب شده توسط یک محیط احتمالی ارزیابی میشود و نتیجه ارزیابی در قالب سیگنالی مثبت یا منفی به اتوماتای یادگیر داده میشود و اتوماتا 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 و به یا عبارتی دیگر 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 شبکه های عصبی [۱۴][۱۵] و الگوریتمهای ژنتیک [۱۶][۱۷] و همچنین حرکت دسته جمعی ذرات باینری[۱۲] اشاره نمود. در [۱۲] مدل PSO باینری بر پایه اتوماتای یادگیر ارائه شده است. در این روش از یک اتواتای یادگیر در هر بعد ذره استفاده شده است. هر کدام از این اتوماتاهای یادگیر دارای دو عمل ۰ و ۱ میباشد و به عنوان مغز متفکر ذره عمل میکند و حرکت آن را در فضای حالت محدود به ۰ و ۱ کنترل و راهبری مینماید. این مدل نتایج بهتری در مقایسه با مدل باینری استاندارد[۲] در حل مسایل نمونه داشته است.

اتوماتای یادگیر سلولی، مدلی برای سیستمهایی است که از اجزاء سادهای تشکیل شدهاند و رفتار هر جزء بر اساس رفتار همسایگانش و نیز تجربیات گذشتهاش تعیین و اصلاح میشود. اجزاء ساده تشکیل دهنده این مدل، از طریق تعامل با یکدیگر می-توانند رفتار پیچیدهای از خود نشان دهند. هر اتوماتای یادگیر سلولی، از یک اتوماتای سلولی تشکیل شدهاست که هر سلول آن به یک یا چند اتوماتای یادگیر مجهز میباشد که وضعیت این سلول را مشخص میسازد. یک قانون محلی در محیط حاکم است و این قانون تعیین میکند که آیا عمل انتخاب شده توسط یک اتوماتا در یک سلول باید پاداش داده شود و یا جریمه شود. عمل دادن پاداش و یا جریمه منجر به بروز درآوردن ساختار اتوماتای یادگیر سلولی برای رسیدن به یک هدف مشخص میگردد.

در [۱۹]، یک مدل محاسبات تکاملی بر پایه اتوماتای یادگیر سلولی معرفی شده است که ترکیبی از مدل محاسبات تکاملی با اتوماتای یادگیر سلولی میباشد. در این مدل هر ژنوم از جمعیت به یک سلول اتوماتای یادگیر سلولی منتسب میشود و هر سلول نیز مجهز به یک مجموعه اتوماتای یادگیر است. مجموعه اعمالی که توسط مجموعه اتوماتای یک سلول انتخاب میشوند، ژنوم سلول را شکل میدهند. بر اساس یک قانون محلی یک بردار سیگنال تقویتی تولید میشود و به مجموعه اتوماتای سلول داده میشود. بر اساس سیگنالهای دریافتی هر اتوماتای یادگیر ساختار درونی خود را با استفاده از یک الگوریتم یادگیری به روز میکند.پروسه انتخاب اعمال و به روز کردن ساختار داخلی تا ملاقات شرایطی از پیش تعیین شده، ادامه پیدا میکند. این مدل برای حل مسایلی با فضای حالت پیچیده میتواند مناسب باشد.

در این مقاله مدل جدیدی با نام CLA-PSO پیشنهاد شده است که ترکیبی از اتوماتای یادگیر سلولی و مدل حرکت دسته جمعی ذرات میباشد. در این مدل، در هر سلول از اتوماتای یادگیر سلولی یک جمعیت از ذرات قرار دارد. اتوماتای یادگیر هر سلول، وظیفه تنظیم رفتار ذرات در ان سلول را بر عهده دارد. تنظیم رفتار ذرات به این معنی است که با استفاده از اتوماتای یادگیر در هر گام تعیین میگردد که ذرات به مسیر فعلی ادامه دهند و یا به دنباله روی از بهترین ذرات پیدا شده تاکنون بپردازند. در واقع وظیفه برقراری تعادل بین جستجوی سراسری و جستجوی محلی که در [۵] بر عهده ضریب میانی گذاشته شده است در این مدل بر عهده

اتوماتاهای یادگیر در سلولها میباشد. استفاده از اتوماتای یادگیردارای دو مزیت عمده میباشد: اولا میتوان از دانش موجود در تعیین روند تغییرات وزن میانی استفاده نمود و ثانیا این روند با گرفتن بازخورد از اجرای الگوریتم اصلاح گردد.

برای تنظیم رفتار ذرات، اتوماتای یادگیر در هر سلول از تجربیات خود و تجربیات اتوماتاهای یادگیر در سلولهای همسایه استفاده میکند. بدین ترتیب نوعی همکاری، بین جمعیتهایی یک سلول و سلولهای همسایه شکل میگیرد که باعث عملکرد بهتر مدل میگردد. اتوماتای یادگیر سلولی یک مدل ذاتا موازی میباشد و این توازی ذاتی به CLA-PSO نیز به ارث میرسد. به عبارت دیگر جمعیتهای سلولها میتوانند به صورت موازی کار کنند. با توجه به این مساله، مدل ارائه شده به صورت بالقوه قابلیت اجرا شدن در زمانی برابر با زمان مدل استاندارد حرکت دسته جمعی ذرات را داراست.

ادامه این گزارش بدین صورت سازماندهی شده است. بخش ۲ به معرفی مدل PSO میپردازد. در بخش۳ اتوماتای سلولی، اتوماتاهای یادگیر و اتوماتای یادگیر سلولی به اختصار شرح داده میشود. در بخش ۴ مدل پیشنهادی ارائه میشود. بخش پنجم اختصاص به ارایه نتایج شبیه سازیها دارد و بخش پایانی نتیجه گیری میباشد

## ۲- حرکت دسته جمعی ذرات

حرکت جمعی ذرات(PSO)، یک تکنیک بهینه سازی احتمالی است که بر مبنای جمعیت کار میکند. این روش در سال ۱۹۹۵ توسط دکتر ابرهارت و دکتر کندی ارائه شد [۱] و ایده اصلی آن از رفتار دسته جمعی ماهیها یا پرندگان به هنگام جستجوی غذا الهام گرفته شده است. گروهی از پرندگان در فضایی به صورت تصادفی دنبال غذا میگردند. تنها یک تکه غذا در فضای مورد بحث وجود دارد. هیچ یک از پرندگان محل غذا را نمیدانند. یکی از بهترین استراتژیها میتواند دنبال کردن پرنده ای باشد که کمترین فاصله را تا غذا داشته باشد. این استراتژی در واقع جانمایه الگوریتم PSO است.

در الگوریتم PSO، هر راه حل که به آن یک ذره گفته میشود، معادل یک پرنده در الگوری حرکت جمعی پرندگان میباشد. هر ذره یک مقدار شایستگی دارد که توسط یک تابع شایستگی محاسبه میشود. هر چه ذره در فضای جستجو به هدف- غذا در مدل حرکت پرندگان- نزدکتر باشد، شایستگی بیشتری دارد.. همچنین هر ذره دارای یک سرعت است که هدایت حرکت ذره را بر عهده دارد. هر ذره با دنبال کردن ذرات بهینه در حالت فعلی، به حرکت خود در فضای مساله ادامه میدهد.

آغاز کار PSO به این شکل است که گروهی از ذرات(راه حلها) به صورت تصادفی به وجود می آیند و با به روز کردن نسلها سعی در یافتن راه حل بهینه مینمایند. در هر گام، هر ذره با استفاده از دو بهترین مقدار به روز میشود. اولین مورد، بهترین موقعیتی است که تا کنون ذره موفق به رسیدن به آن شده است. موقعیت مذکور با نام pbest شناخته و نگهداری میشود. بهترین مقدار دیگری که توسط الگوریتم مورد استفاده قرار میگیرد، بهترین موقعیتی است که تا کنون توسط جمعیت ذرات بدست آمده است. این موقعیت با gbest نمایش داده میشود. در برخی ویرایشهای PSO، ذره، قسمتهایی از جمعیت را که همسایگان توپولوژیکی اش هستند، انتخاب میکند و تنها آنها را در اعمال خود دخیل میکند که در این حالت به بهترین راه حل محلی که با lbest نشان داده میشود، به جای gbest مورد استفاده قرار میگیرد.

پس از یافتن بهترین مقادیر، سرعت و مکان هر ذره با استفاده از معادلات (۱) و (۲) به روز میشود.

| | |
|---|---|
| v[] = v[] +<br>c1 * rand() * (pbest[] - position []) +<br>c2 * rand() * (gbest[] - position[]) | (۱) |
| position[] = position[] + v[] | (۲) |

در معادلات (۱) و (۲)، v[] سرعت ذره و position [] محل فعلی ذره هستند که هر دو آرایه هایی به طول تعداد ابعاد مساله میباشند. Rand () یک عدد تصادفی در بازه (0, 1) است و C1 و C2 نیز فاکتورهای یادگیری هستند. معمولا C1=C2=2 در نظر گرفته میشود. سرعت ذرات در هر بعد به یک مقدار Vmax محدود میشوند. اگر مجموع شتابها باعث شوند که سرعت در یک بعد از Vmax بیشتر شود، مقدار سرعت در آن بعد برابر با Vmax قرار میگیرد. شبه کد الگوریتم PSO در شکل (۱) مشاهده میشود.

سمت راست معادله (۱) از سه قسمت تشکیل شده است که قسمت اول، سرعت فعلی ذره است و قسمتهای دوم و سوم تغییر سرعت ذره و چرخش آن به سمت بهترین تجربه شخصی و بهترین تجربه گروه را به عهده دارند. اگر قسمت اول را در این معادله در نظر نگیریم، آنگاه سرعت ذرات تنها با توجه به موقعیت فعلی و بهترین تجربه ذره و بهترین تجربه جمع تعیین میشود. به این ترتیب، بهترین ذره جمع، در جای خود ثابت میماند و سایرین به سمت آن ذره حرکت میکنند. در واقع حرکت دسته جمعی ذرات

بدون قسمت اول معادله(۱)، پروسه ای خواهد بود که طی آن فضای جستجو به تدریج کوچک میشود و جستجویی محلی حول بهترین ذره شکل میگیرد. در مقابل اگر فقط قسمت اول معادله (۱) را در نظر بگیریم، ذرات راه عادی خود را میروند تا به دیواره محدوده برسند و به نوعی جستجویی سراسری را انجام میدهند[۵].

```
For each particle
   Initialize particle
End For
Do
   For each particle
      Calculate fitness value of the particle fp
      /*updating  particle's best fitness value so far)*/
       If fp is better than pBest
          set current value as the new pBest
   End For
    /*updating  population's best fitness value so far)*/
   Set gBest to the best fitness value of all particles
   For each particle
      Calculate particle velocity according equation (1)
      Update particle position according equation (2)
   End For
While maximum iterations OR
        minimum error criteria is not attained
```

**شکل ۱- شبه کد الگوریتم PSO**

در معادله (۱) با ترکیب این دو عامل سعی شده است تا موازنه ای بین جستجوی محلی و سراسری برقرار گردد. در [۵]، یک پارامتر جدید توسط دکتر شی و دکتر ابرهارت که از ارائه دهندگان الگوریتم اولیه PSO بود، ارائه گردید. این پارامتر جدید وزن میانی نامیده و با w نشان داده شد. هدف از معرفی این پارامتر برقراری موازنه بهتر و قابل تنظیم بسته به نوع مساله، بین جستجوی محلی و جستجوی سراسری است. به این منظور در معادله (۱)، این ضریب در سرعت اولیه ذره ضرب میشود و به عبارتی تنها قسمتی از سرعت اولیه به سرعت آینده ذره منتقل میشود و به این ترتیب معادله (۱) به شکل معادله (۳) تغییر میکنند.

$$V[] = w * v[] + c1 * rand() * (pbest[] - position[]) + c2 * rand() * (gbest[] - position[]) \quad (۳)$$

وزن میانی میتواند یک ضریب ثابت، یک تابع خطی با زمان و یا حتی یک تابع غیرخطی با زمان نیز باشد. در بسیاری از کاربردها وزن میانی به صورت تابعی خطی با زمان در نظر گرفته میشود که تابعی نزولی است. به این ترتیب در ابتدا قسمت بیشتری از سرعت فعلی ذره در سرعت آینده اش دخالت داده میشود و با گذشت زمان، این میزان کاهش می یابد. به عبارت بهتر، در ابتدا ذرات میل بیشتری به حرکات انفجاری و تجربه های تازه دارند و با گذشت زمان این میل جای خود را به دنباله روی بیشتر از بهترینها میدهد. این روش در بسیاری موارد میتواند مشکل گیرافتادن در بهینه های محلی را حل کند.

مدل باینری PSO برای حل مسایل گسسته، در سال ۱۹۹۷ توسط ارائه دهندگان الگوریتم استاندارد، معرفی شد[۲]. در این مدل موقعیت هر ذره در هر بعد با یکی از دو مقدار ۰ یا ۱ مشخص میگردد. به این ترتیب، ذره در یک فضای محدود به ۰ و ۱ حرکت میکند و سرعت ذره در هر بعد برابر با احتمال یک بودن موقعیت ذره در آن بعد خواهد بود. بروز رسانی سرعت همچنان مطابق با معادله (۱) انجام میشود. سپس ابتدا سرعت بدست آمده در هر بعد با استفاده از تابع سیگموید به بازه [۰,۱] منتقل میگردد و آنگاه موقعیت جدید ذره در هر بعد مطابق با معادله (۴) محاسبه میگردد.

$$\text{If } (rand() < sigmoid(v_{id}(t+1)) \text{ then } x_{id}(t+1) = 1 \text{ else } xid(t+1) = 0 \quad (۴)$$

که ()rand یک مقدار تصادفی در بازه [۰,۱] است.

## ۳- اتوماتای سلولی، اتوماتای یادگیر و اتوماتای یادگیر سلولی[۶]

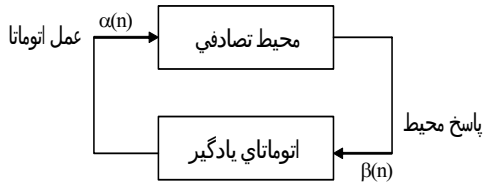
### ۳-۱- اتوماتای سلولی

اتوماتای سلولی [۲۰] یک مدل ریاضی برای سیستمهایی است که در آنها چندین مؤلفه ساده برای تولید الگوهای پیچیده با هم همکاری می‌کنند. در اتوماتای سلولی یک مجموعه منظم از سلولها وجود دارد که هر کدام می‌توانند با چند مقدار مختلف که تعدادشان متناهی است، مقداردهی شوند. این سلولها به صورت همگام و در زمانهای گسسته بر طبق یک قانون محلی بهنگام‌رسانی می‌شوند. محلی بودن به این معناست که در تعیین مقدار جدید هر سلول، سلولهایی که در همسایگی وی هستند تاثیرگذار هستند و سلولهای دورتر، تاثیری ندارند. شبکه سلولها می‌تواند ابعاد متفاوتی داشته باشند و یک, دو و یا بیشتر بعد داشته باشند.

ویژگیهای اتوماتای سلولی را به اختصار میتوان به صورت زیر بیان نمود: فضا و زمان به صورت گسسته پیش میروند. اتوماتا همگن است. عمل به روزرسانی به صورت همگام انجام میشود و قوانین بر اساس همسایه های هر سلول تعریف میشوند. از مشکلات مهم اتوماتای سلولی تعیین فرم قطعی قوانین است. زیرا در اغلب سیستمها نویز و عدم قطعیت وجود دارند که سیستم را تحت تاثیر قرار میدهند. لذا تعیین فرم قطعی قوانین در این سیستمها کاری مشکل و در برخی موارد غیر ممکن است.

## ۳-۲- اتوماتای یادگیر

اتوماتـای یـادگیر[۳]، ماشـینی اسـت کـه میتوانـد تعـدادی متناهی عمل را انجام دهد. هر عمل انتخاب شده توسط یک محیط احتمالی ارزیابی میشود و نتیجه ارزیابی در قالب سیگنالی مثبت یـا منفی به اتوماتا داده میشود و اتوماتا از این پاسخ در انتخـاب عمـل بعدی تاثیر میگیرد. هدف نهایی این است که اتوماتا یاد بگیرد تـا از بین اعمال خود بهترین عمل را انتخاب کند. بهتـرین عمـل، عملـی است که احتمال دریافت پـاداش از محـیط را بـه حـداکثر برسـاند. کارکرد اتوماتای یادگیر در تعامل با محـیط، در شـکل(۲) مـشاهده میشود.




**شکل (۲) : ارتباط بین اتوماتای یادگیر و محیط**

اتوماتای یادگیر با ساختار متغیر را میتوان توسط چهارتایی $\{\alpha,\beta,p,T\}$ نشان داد که $\alpha=\{\alpha_1,...,\alpha_r\}$ مجموعه عملهای اتوماتا، $\beta=\{\beta_1,...,\beta_m\}$ مجموعه ورودیهای اتوماتا، $p=\{p_1,..,p_r\}$ بردار احتمال انتخاب هریک از عملها و $p(n+1)=T[\alpha(n),\beta(n),p(n)]$ الگوریتم یادگیری می باشد. الگوریتم زیریک نمونه از الگوریتمهای یادگیری خطی است. فرض کنید عمل $\alpha_i$ در مرحله $n$ام انتخاب شود.

- پاسخ مطلوب

$$\begin{aligned} p_i(n+1)&=p_i(n)+a[1-p_i(n)] \\ p_j(n+1)&=(1-a)p_j(n) \qquad \forall j \;\; j\neq i \end{aligned} \qquad (۵)$$

- پاسخ نامطلوب

$$\begin{aligned} p_i(n+1)&=(1-b)p_i(n) \\ p_j(n+1)&=(b/r-1)+(1-b)p_j(n) \quad \forall j \;\; j\neq i \end{aligned} \qquad (۶)$$

در روابط (۵) و (۶)، $a$ پارامتر پاداش و $b$ پارامتر جریمه می باشند. با توجه به مقادیر $a$ و $b$ سه حالت زیر را می تـوان در نظـر گرفت. زمانیکه $a$ و $b$ با هم برابـر باشـند، الگـوریتم را $L_{RP}$ مـی نامیم، زمانیکه $b$ از $a$ خیلی کوچکتر باشد، الگـوریتم را $L_{R\varepsilon P}$ مـی نـامیم. و زمانیکـه $b$ مـساوی صـفر باشـد الگـوریتم را $L_{RI}$ مینامیم[۴]. شکل (۳) شبه کد اتوماتای یادگیر با ساختار متغیـر را نشان میدهد.

```
Initialize p to [1/s,1/s,…,1/s]  /*s is the number of actions*/
While not done
    Select an action i based on the probability vector p
    Evaluate action and return a reinforcement signal β
    Update probability vector using learning rule
End While
```

**شکل (۳) : اتوماتای یادگیر با سختار متغیر**

## ۳-۳- اتوماتای یادگیر سلولی

بسیاری از مسایل را نمیتوان با استفاده از یک اتوماتای یادگیر تکی حل کرد بلکه قدرت اصلی اتوماتای یادگیر زمانی آشکار میشود که آنها به صورت دسته جمعی بکار روند. با توجه بـه ایـن مـساله و ضعفهای عنوان شده برای اتوماتای سلولی، در [۲۱] با ترکیـب ایـن دو مدل، مدل جدیدی با نام اتوماتای یادگیر سـلولی ایجـاد گردیـد. در ادامه تعریف فرمال اتوماتای یادگیر سلولی[۲۲] ارائه شده است.

**تعریف** (اتوماتای یادگیر سلولی) : اتوماتای یادگیر سلولی d بعـدی یک چندتایی $CLA=(Z^d,\phi,A,N,F)$ است به طوریکه:

- $Z^d$ یک شبکه از d تایی های مرتـب از اعـداد صـحیح می باشد. این شبکه می تواند یک شبکه متناهی، نیمه متنـاهی یـا متناهی باشد.
- $\phi$ یک مجموعه متناهی از حالتها می باشد.
- A، یک مجموعه از اتوماتاهای یادگیر (LA) است که هر یک از آنها به یک سلول از اتوماتای سلولی نسبت داده میشود.
- $N=\{\overline{x}_1,...,\overline{x}_{\overline{m}}\}$ یک زیر مجموعه متناهی از $Z^d$ می باشد که بردار همسایگی نامیده می شود.
- $F:\underline{\phi}^{\overline{m}}\rightarrow\underline{\beta}$ قـانون محلـی CLA مـی باشـد بـه طوریکه $\underline{\beta}$ مجموعه مقادیری است که می تواند به عنوان سـیگنال تقویتی پذیرفته شود.

در اتوماتای یادگیر سلولی مـی‌تـوان از سـاختارهای مختلفـی برای همسایگی استفاده نمود. در حالت کلی هر مجموعـه مرتـب از سلولها را می‌توان به عنوان همسایه در نظر گرفت.

عملکرد اتوماتای یادگیر سلولی را می‌توان به شرح زیـر بیـان کرد. در هر لحظه هر اتوماتای یادگیر در اتوماتای یادگیر سلولی یک عمل از مجموعه اعمال خود را انتخاب می‌کند. این عمل می‌تواند بر اساس مشاهدات قبلی و یا به صورت تـصادفی انتخـاب شـود. عمـل انتخاب شده با توجه به اعمال انتخاب شده توسط سلولهای همسایه

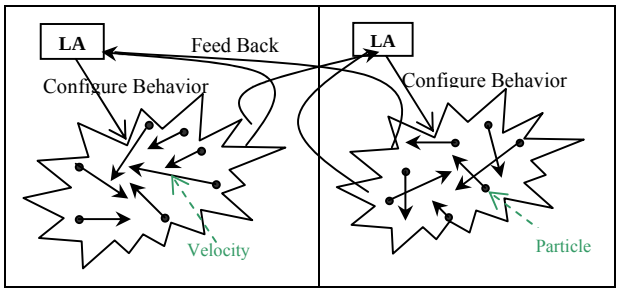
و قانون حاکم بر اتوماتای یادگیر سلولی پاداش داده و یا جریمه می‌شود. با توجه به اینکه عمل انتخاب شده پاداش گرفته و یا جریمه شده است, اتوماتا رفتار خود را تصحیح کرده و ساختار داخلی اتوماتا بهنگام می‌گردد. معمولاً عمل بروزرسانی تمام اتوماتا به صورت همزمان انجام می‌شود. بعد از بروزرسانی، هر اتوماتای یادگیر در اتوماتای یادگیر سلولی دوباره یک عمل از مجموعه اعمال خود را انتخاب کرده و انجام می‌دهد. فرآیند انتخاب عمل و دادن پاداش و یا جریمه تا زمانی که سیستم به حالت پایدار برسد و یا یک معیار از قبل تعریف شده برقرار شود, ادامه می‌یابد.

## ۴- CLA-PSO

در این بخش یک مدل جدید به نام CLAPSO معرفی شده است که تر کیبی از اتوماتای یادگیر سلولی و حرکت دسته جمعی ذرات میباشد. در این مدل، هر سلول اتوماتای یادگیر سلولی دارای یک جمعیت از ذرات و همچنین یک اتوماتای یادگیر است. مانند مدل حرکت دسته جمعی ذرات، هر یک از ذرات دارای یک بردار موقعیت و یک بردار سرعت هستند. اتوماتای یادگیر هر سلول دارای دو عمل «دنباله روی» و «ادامه مسیر فعلی» میباشد و وظیفه تنظیم رفتار ذرات همان سلول را بر عهده دارد.

تنظیم رفتار ذرات به این معنی است که با استفاده از اتوماتای یادگیر در هر گام تعیین میگردد که ذرات به مسیر فعلی ادامه دهند و یا به دنباله روی از بهترین ذرات پیدا شده تا کنون بپردازند. در واقع وظیفه برقراری تعادل بین جستجوی سراسری و جستجوی محلی که در [۵] بر عهده ضریب میانی گذاشته شده است، در الگوریتم پیشنهادی توسط اتوماتای یادگیر انجام میشود که دو مزیت مهم دارد: اولا میتوان از دانش موجود مساله در تعیین روند تغییرات وزن میانی استفاده نمود و ثانیا این روند با گرفتن بازخورد از اجری الگوریتم اصلاح میگردد.

برای تنظیم رفتار ذرات، اتوماتای یادگیر هر سلول، از تجربیات خود و اتوماتاهای یادگیر سلولهای همسایه، استفاده میکند. بدین ترتیب نوعی همکاری، بین جمعیتهایی یک سلول و سلولهای همسایه شکل میگیرد که باعث عملکرد بهتر آنها میگردد. شکل (۴)، نمای کلی یک CLAPSO با دو سلول را نشان میدهد.




**شکل (۴)- CLA-PSO با دو سلول**

روش کار الگوریتم پیشنهادی به این صورت است که در ابتدا، تعداد سلولها و نوع همسایگی اتوماتای یادگیر سلولی مشخص میشود و موقعیت و سرعت ذرات و همچنین احتمال انتخاب اعمال اتوماتای سلولها مقداردهی اولیه میشوند. سپس تا زمانیکه حداکثر تعداد گامها انجام گردد و یا هدف مورد نظر حاصل شود، مراحل زیر تکرار میشوند:

۱- اتوماتای یادگیر یکی از اعمالشان را با توجه به بردار احتمال اعمال، انتخاب میکنند.

۲- با توجه به عمل انتخاب شده توسط اتوماتای یادگیر هر سلول، نحوه بروزرسانی سرعت ذرات در آن سلول مشخص میشود و ذرات، سرعت و موقعیت خود را به روز میکنند.

۳- بر اساس نتایج به روزرسانی موقعیت ذرات و همچنین مقایسه وضعیت بهترین ذره جمعیت سلول با بهترین ذره های سلولهای همسایه، عمل اتوماتای یادگیر ، ارزیابی میشود و بردار احتمال انتخاب اعمال اتوماتای یادگیر اصلاح میشود.

همانطور که در مرحله دوم اشاره شد، عملی که اتوماتای یادگیر سلولها در هر گام برمیگزینند، تعیین کننده شیوه بروز کردن سرعت ذرات در آن گام میباشند. در صورت انتخاب عمل «دنباله روی»، تنها دنبال کردن بهترین تجربه شخصی و بهترین تجربه گروهی، در به روز نمودن سرعت ذرات مد نظر قرار خواهند گرفت و از سرعت فعلی ذرات صرفنظر میشود. در این صورت بروزرسانی سرعت ذرات مطابق با فرمول (۷) انجام میگردد. در صورت انتخاب عمل «ادامه مسیر فعلی»، سرعت جدید ذرات برابر با سرعت فعلی آنها خواهد بود و ذره همچنان مسیری که میرفت ادامه خواهد داد.

$$V[] = c1 * rand() * (pbest[] - position[]) + c2 * rand() * (gbest[] - position[]) \quad (۷)$$

در واقع انتخاب عمل «دنباله روی»، انجام یک جستجوی محلی را در پی خواهد داشت و انتخاب عمل «ادامه مسیر فعلی»، باعث جستجوی سراسری و کشف قسمتهایی ناشناخته از فضای جستجو میگردد و وظیفه اتوماتای یادگیر بکارگرفته شده، یادگیری

توزیع احتمال مطلوب و برقراری تعادل بین جستجوی سراسری و جستجوی محلی است.

برای ارزیابی عمل انتخاب شده، به این صورت عمل میشود که موقعیت هر ذره در اثر انجام یک گام، با موقعیت قبلیش مقایسه میشود. چنانچه درصد مشخصی از جمعیت موقعیتشان بهبود یافته بود، عمل انتخاب شده مثبت ارزیابی میشود و در غیر اینصورت عمل انتخاب شده منفی ارزیابی خواهد شد.

اگر عمل انتخاب شده، «دنباله روی» باشد، برای مثبت ارزیابی شدن عمل، بایستی موقعیت $cf_{imp}$ درصد جمعیت بهبود یافته باشد و چنانچه عمل انتخاب شده، «ادامه مسیر فعلی» باشد، بهبود یافتن موقعیت $cg_{imp}$ درصد جمعیت برای مثبت ارزیابی شدن عمل لازم است.

برای تعیین مقادیر $cf_{imp}$ و $cg_{imp}$، در ابتدا gbest جمعیت سلول با gbest جمعیتهای سلولهای همسایه مقایسه میشود. هر چه در این مقایسه، gbest جمعیت سلول وضعیت بهتری داشته باشد، استنباط میگردد که بهترین ذره جمعیت به نسبت بهترین ذرات جمعیتهای سلولهای همسایه در قسمت مناسبتری از فضای جستجو قرار گرفته است و مقدار $cf_{imp}$ کمتر و مقدار $cg_{imp}$ بیشتر قرار داده میشود. در واقع سعی میگردد تا عمل «دنباله روی» تشویق گردد. بر عکس، چنانچه در مقایسه انجام شده، gbest وضعیت بدتری داشته باشد، سعی میشود تا با تشویق عمل «ادامه مسیر فعلی»، قسمتهای بهتری از فضای جستجو کشف شوند و به همین دلیل مقدار $cf_{imp}$ بیشتر و مقدار $cg_{imp}$ کمتر قرار داده میشود.

برای استفاده از الگوریتم، مجموعه ای از پارامترها باید تنظیم شوند. ابتدا باید شرایطی مشخص و دقیق تعیین گردند که کیفیت gbest جمعیت سلول در مقایسه با سلولهای همسایه، توسط آنها سنجیده میشود. برای مثال اگر سلولی ۸ همسایه داشته باشد و m تعداد سلولهای همسایه ای باشدکه gbest جمعیتشان بدتر از gbest جمعیت سلول مورد بحث باشد، آنگاه، یک مجموعه شرایط میتوانند به صورت مجموعه معادلات (۸) باشند.

$$Stat = \begin{cases} good & m > 6 \\ avg & 3 \le m \le 6 \\ bad & m < 3 \end{cases} \qquad (۸)$$

پس از تعیین این شرایط، برای هر دسته وضعیت، باید مقادیر دقیق $cf_{imp}$ و $cg_{imp}$ مشخص گردند. مثلا میتوان برای دسته خوب، $cf_{imp}$ را برابر با ۳۰ و $cg_{imp}$ را برابر با ۸۰ قرار داد. در صورت تنظیم پارامترها به صورت گفته شده الگوریتم به این صورت عمل خواهد کرد: اگر بهترین ذره جمعیت سلول در مقایسه با بهترین ذرات جمعیتهای سلولهای همسایه، حداقل از ۶ تای آنها بهتر باشد، آنگاه، اگر عمل «دنباله روی» انتخاب شود، فقط کافیست تا موقعیت ۳۰ درصد از ذرات را بهبود دهد تا مثبت ارزیابی شود ولی اگر عمل «ادامه مسیر فعلی» انتخاب شود، برای مثبت ارزیابی شدن، باید موقعیت ۸۰ درصد ذرات را بهبود دهد.

پارامترهای گفته شده را بسته به نوع مساله و شکل فضای جستجو، تعداد سلولها و نوع همسایگی و همچنین نوع اتوماتای یادگیر میتوان تنظیم نمود. شکل (۵) شبه کد الگوریتم ارائه شده را نشان میدهد.

```
Initialize the LA
For each particle
   Initialize particle
End For
Do
   For Each cell do in parallel
      The LA selects an action ac
      For each particle
         Calculate fitness value of the particle fp
         /*updating  particle's best fitness value so far)*/
          If fp is better than pBest
             set current value as the new pBest
      End For
       /*updating  population's best fitness value so
far)*/
      Set gBest to the best fitness value of all particles
      For each particle
         if  ac is "follow the best"
              Calculate particle velocity according
               equation (7)
         Update particle position according equation (2)
      End For
      Update LA's probability vector
   End for
While maximum iterations OR
         minimum error criteria is not attained
```

**شکل (۵)- شبه کد الگوریتم PSO-LA**

یکی از مزایای الگوریتم پیشنهادی ایجاد موازنه بین جستجوی سراسری و جستجوی محلی میباشد. این موازنه با توجه به نوع مساله و شکل فضای جستجو برقرار میگردد. مزیت دیگر الگوریتم پیشنهادی این است که بدلیل ایجاد رقابت ما بین جمعیتهای مختلف که در سلولها پراکنده شده اند اگر در یک سلول احساس شود که بهترین ذره ان سلول در محل مناسبی از فضای جستجو قرار نگرفته است و نسبت به بهترین ذرات جمعیتهای سلولهای همسایه وضعیت بدتری دارد، عمل «ادامه مسیر فعلی» تشویق میشود و بدین وسیله تلاش میگردد تا با یافتن قسمتهایی جدید از

فضای جستجو این عقب ماندگی جبران گردد. این مکانیزم میتواند الگوریتم را از بهینه های محلی خارج نموده و مشکل گیرافتادن در بهینه های محلی را کاهش دهد.

## ۵- نتایج شبیه سازیها

آزمایشات بر روی چهار تابع تست استاندارد صورت گرفته است که معمولا به عنوان معیار سنجش الگوریتمهای بهینه سازی مورد استفاده قرار میگیرند. توابع استفاده شده، توابع آکلی[۷]، روزنبراک[۸]، رستریجین[۹] و اسفیر[۱۰] میباشند که به ترتیب توسط معادلات (۹) تا (۱۲) تعریف شده اند.

توابع فوق همگی دارای بهینه سراسری با مقدار ۰ هستند. آزمایشات با مقادیر n برابر با ۱۰، ۲۰ و ۳۰ صورت گرفته است. اتوماتای یادگیر سلولی با ۹ سلول و همسایگی مور مورد استفاده قرار گرفته است و هر سلول با ۸ سلول دیگر همسایه است. از مجموعه معادلات (۸) برای تعیین وضعیت gbest جمعیت سلول استفاده شده است که سه دسته خوب، متوسط و بد را تعیین میکند. برای دسته خوب، $cf_{imp}$ برابر با ۲۵ و $cg_{imp}$ را برابر با ۷۵ قرار داده شده است. برای دسته متوسط هردو مقدار $cf_{imp}$ و $cg_{imp}$ برابر با ۷۵ و برای دسته بد $cf_{imp}$ برابر با ۷۵ و $cg_{imp}$ را برابر با ۲۵ قرار داده شده است.

| | |
|---|---|
| $f(x)=20+e-20e^{-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2}}-e^{\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)}$ | (۹) |
| $f(x)=\sum_{i=1}^{n-1}(100(x_{i+1}-x_i^2)^2+(x_i-1)^2)$ | (۱۰) |
| $f(x)=\sum_{i=1}^{n}(x_i^2-10\cos(2\pi x_i)+10)$ | (۱۱) |
| $f(x)=\sum_{i=1}^{n}x_i^2$ | (۱۲) |

تعداد جمعیت ۲۵ و تعداد گامها ۱۰۰۰ بوده است. ضرایب پاداش و جریمه برای اتوماتای یادگیر $L_{RP}$ و نیز ضریب پاداش برای اتوماتای یادگیر $L_{RI}$ برابر با ۰٫۰۱ در نظر گرفته شده است. آزمایشات ۳۰ بار تکرار شده اند و بهترین و میانگین نتایج سی بار اجرا ارائه شده است. نتایج بدست آمده در جدولهای۱ تا ۴ مشاهده میشوند. همانطور که در جداول (۱) تا (۴) مشاهده میشود، استفاده از وزن میانی کارایی الگوریتم حرکت دسته جمعی ذرات را بهبود میدهد. همچنین الگوریتم CLA-PSO عملکرد بهتری نسبت به الگوریتم حرکت دسته جمعی ذرات حتی زمانیکه از پارامتر وزن میانی استفاده میکند داشته است. این برتری به ویژه در مورد توابع روزنبراک و اسفیر بیشتر میباشد.

**جدول ۱- نتایج مقایسه دو الگوریتم PSO و CLA-PSO برای تابع آکلی**

| ابعاد | الگوریتم | بهترین | متوسط |
|---|---|---|---|
| ۱۰ | PSO-Std | ۰٫۵۹ | ۲٫۵ |
| | PSO-I.W | ۰٫۰۸ | ۲٫۴۹ |
| | $CLA\text{-}PSOL_{RP}$ | ۰٫۰۱۱ | ۰٫۰۱۶ |
| | $CLA\text{-}PSOL_{RI}$ | ۰٫۰۰۹ | ۰٫۰۱ |
| ۲۰ | PSO-Std | ۴٫۳۲ | ۶٫۰۹ |
| | PSO-I.W | ۳٫۷۶ | ۵٫۳۶ |
| | $CLA\text{-}PSOL_{RP}$ | ۰٫۴۸ | ۱٫۰۸ |
| | $CLA\text{-}PSOL_{RI}$ | ۰٫۴۳ | ۱٫۱۲ |
| ۳۰ | PSO-Std | ۷٫۴۶ | ۹٫۱۱ |
| | PSO-I.W | ۶٫۵۱ | ۸٫۳۴ |
| | $CLA\text{-}PSOL_{RP}$ | ۴٫۸۰ | ۵٫۲۷ |
| | $CLA\text{-}PSOL_{RI}$ | ۴٫۹ | ۵٫۳۶ |

**جدول ۲- نتایج مقایسه دو الگوریتم PSO و CLA-PSO برای تابع روزنبراک**

| ابعاد | الگوریتم | بهترین | متوسط |
|---|---|---|---|
| ۱۰ | PSO-Std | ۱۴٫۳۹ | ۳۵۲٫۸۶ |
| | PSO-I.W | ۰٫۴۳ | ۱۱٫۳۳ |
| | $CLA\text{-}PSOL_{RP}$ | ۰٫۲۱ | ۸٫۵۲ |
| | $CLA\text{-}PSOL_{RI}$ | ۰٫۵۶ | ۵٫۴ |
| ۲۰ | PSO-Std | ۵۸۶۴٫۰۶ | ۳۱۶۷۴٫۲۶ |
| | PSO-I.W | ۱۴٫۷۲ | ۳۵٫۲۴ |
| | $CLA\text{-}PSOL_{RP}$ | ۸٫۴۱ | ۲۷٫۸۲ |
| | $CLA\text{-}PSOL_{RI}$ | ۱۰٫۱ | ۲۹٫۹۶ |
| ۳۰ | PSO-Std | ۵۲۰۵۷٫۶۴ | ۴۱۲۵۱۵٫۰۴ |
| | PSO-I.W | ۲۹٫۱۵ | ۸۹٫۰۰ |
| | $CLA\text{-}PSOL_{RP}$ | ۲۱٫۷ | ۶۲٫۳۱ |
| | $CLA\text{-}PSOL_{RI}$ | ۲۲٫۵۱ | ۶۱٫۹ |

**جدول ۳- نتایج مقایسه دو الگوریتم PSO و CLA-PSO برای تابع رستریجین**

| ابعاد | الگوریتم | بهترین | متوسط |
|---|---|---|---|
| ۱۰ | PSO-Std | ۴٫۹۷ | ۱۰٫۱۴ |
| | PSO-I.W | ۳٫۶۵ | ۸٫۶۹ |
| | $CLA\text{-}PSOL_{RP}$ | ۱٫۰۵ | ۳٫۰۶ |
| | $CLA\text{-}PSOL_{RI}$ | ۰٫۰۹ | ۲٫۸۳ |
| ۲۰ | PSO-Std | ۱۰٫۰۵ | ۲۸٫۲۱ |
| | PSO-I.W | ۱۰٫۹۹ | ۲۸٫۹۳ |
| | $CLA\text{-}PSOL_{RP}$ | ۶٫۰۷ | ۹ |
| | $CLA\text{-}PSOL_{RI}$ | ۷٫۳۳ | ۸٫۹۹ |

| | | | |
|---|---|---|---|
| ۳۰ | PSO-Std | ۲۸,۴۹ | ۵۷,۱۰ |
| | PSO-I.W | ۲۴,۲۵ | ۳۴,۱۶ |
| | $CLA\text{-}PSOL_{RP}$ | ۱۳,۱ | ۱۶,۰۳ |
| | $CLA\text{-}PSOL_{RI}$ | ۹,۴۷ | ۱۴,۴۱ |

**جدول ۴- نتایج مقایسه دو الگوریتم PSO و CLA-PSO برای تابع اسفیر**

| ابعاد | الگوریتم | بهترین | متوسط |
|---|---|---|---|
| ۱۰ | PSO-Std | ۰,۰۰۹ | ۰,۳۳ |
| | PSO-I.W | ۰,۰۰۰۴ | ۰,۰۰۲ |
| | $CLA\text{-}PSOL_{RP}$ | ۵,۴۷e-۵ | ۹,۳۸e-۵ |
| | $CLA\text{-}PSOL_{RI}$ | ۶,۱۳e-۵ | ۰,۰۰۰۱ |
| ۲۰ | PSO-Std | ۰,۶۹ | ۹,۲۱ |
| | PSO-I.W | ۰,۰۰۸ | ۷,۰۷ |
| | $CLA\text{-}PSOL_{RP}$ | ۰,۰۰۱۲ | ۰,۰۰۱۹ |
| | $CLA\text{-}PSOL_{RI}$ | ۰,۰۰۱۱ | ۰,۰۰۱۹ |
| ۳۰ | PSO-Std | ۲,۱۱ | ۱۵,۰۸ |
| | PSO-I.W | ۰,۰۲ | ۵,۸۲ |
| | $CLA\text{-}PSOL_{RP}$ | ۰,۰۰۴ | ۰,۰۰۷ |
| | $CLA\text{-}PSOL_{RI}$ | ۰,۰۰۵ | ۰,۰۰۷ |

## ۶- نتیجه گیری

در این مقاله یک مدل جدید برای بهینه سازی که بر پایه دو مدل حرکت دسته جمعی ذرات و اتوماتاای یادگیر میباشد پیشنهاد گردید. در این مدل در هر سلول اتوماتای یادگیر سلولی یک جمعیت از ذرات قرار دارد. اتوماتای یادگیر در هر سلول، به مثابه مغز متفکر و کنترل کننده حرکات ذرات در ان سلول عمل میکند. اتوماتای یادگیر هر سلول دارای دو عمل دنبال کردن بهترینها و یا ادامه مسیر فعلی میباشد. با انتخاب یک عمل در هر گام نحوه بروز رسانی سرعت ذرات تعیین میشود. نتایج شبیه سازیها نشان داد که مدل ارائه شده پاسخهای بهتری در مقایسه با مدل PSO استاندارد و همچنین مدل PSO با استفاده از وزن میانی تولید میکند.

## مراجع


[1] Kennedy, J. and Eberhart, R. C., "*Particle Swarm Optimization*", Proceedings of IEEE International Conference on Neural Networks, Piscataway, NJ, pp. 1942-1948, 1995.

[2] Kennedy, J., Eberhart, R. C., "*A Discrete Binary Version of the Particle Swarm Algorithm*", Proceedings of the International Conference on Systems, Man, and Cybernetics, IEEE Service Center, Piscataway, NJ, pp. 4104-4108, 1997.

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

[4] 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.

[5] Shi, Y. and Eberhart, R. C., "*A Modified Particle Swarm Optimizer*", IEEE International Conference on Evolutionary Computation, Anchorage, Alaska, USA, 1998.

[6] Hagashi, N. and Iba, H., "*Particle Swarm Optimization with Gaussian Mutation*" Proceedings of the IEE swarm intelligence symposium, Indianapolis, Indiana, USA, pp. 72-79, 2003.

[7] Secrest, B. R. and Lamont, G. B., "*Visualizing Particle Swarm Optimization – Gausssian Particle Swarm Optimization*", Proceedings of the IEEE Swam intelligence Symposium, Indianapolis, Indiana, USA, pp. 198-204, 2003.

[8] Stacey, A., Jancic, M., and Grundy, I., "*Particle Swarm Optimization with Mutation*", Proceedings of the Congress on Evolutionary Computation, p.1425-1430, 2003.

[9] Riget, J. and Vesterstroem, J.S., "*A Diversity Guided Particle Swarm Optimizer - the ARPSO*", Department of Computer Science, University of Aarhus, Tech. Rep. No. 2002-02, 2002.

[10] Kiink, T., Vesterstroem, J. S. and Riget, J., "*Particle Swam Optimization with Spatial Particle Extension*", Proceedings of the lEEE Congress on Evolutionary Computation, pp. 1474-1479, 2002.

[11] Krink, T. and Lovhjerg, M., "*The Life Cycle Model: Combining Particle Swarm Optimization, Genetic Algorithms and Hill Climbers*" Proceedings of Parallel Problem Solving from Nature VI1 (PPSN 2002), pp. 621-630, 2002.

[12] Rastegar, R., Meybodi, M.R., K. Badie, "*A New Discrete Binary Particle Swarm Optimization based on Learning automata*", Proceedings of the International Conference on Machine Learning and Applications, ICMLA '04, pp. 456-462, 2004.

[13] Shi, Y. and Eherhart, R.C., "*Empirical Study of Particle Swarm Optimization*", Proceedings of the IEEE Congress 011 Evolutionary Computation, pp. 1945-1950, 1999.

[14] Meybodi, M. R. and Beigy, H., "*A Note on Learning Automata Based Schemes for Adaptation of BP Parameters*", Journal of Neurocomputing, Vol. 48, No. 4, pp. 957-974, October 2002.

[15] Beigy, H. and Meybodi, M.R., "*A Learning Automata Based Algorithm for Determination of Minimum Number of Hidden Units for Three Layers Neural Networks*", Journal of Amirkabir, Vol. 12, No. 46, pp. 111-136, 2001.

[16] Munetomi, M., Takai, Y., and Sato, Y., "*StGA: An Application of Genetic Algorithm to Stochastic Learning Automata*", Syst. Comput. Jpn., Vol. 27, PP. 68-78, 1996.

[17] Howell, M. N., Gordon, T. J., and Brandao, F. V., "*Genetic Learning Automata for Function Optimization*", IEEE Transaction on Systems, Man, and Cybernetics-Part B: Cybernetics, Vol. 32, No. 6, 2002.

[18] Wang, X.H. and Li, J.J., "*Hybrid Particle Swarm Optimization with Simulated Annealing*", Proceedings of the Third International Conference on Machine Learning and Cybernetics, Shanghai, China, 2004.

[19] Rastegar, R. and Meybodi M. R., "*A New Evolutionary Computing Model based on Cellular Learning Automata*", Proceedings of IEEE conference on Cybernetics and Intelligent Systems 2004 (CIS2004), Singapore, 2004.

[۲۰] محمدرضا میبدی, حمید بیگی و مسعود طاهرخانی, « اتوماتای یادگیر سلولی», در مجموعه مقالات ششمین کنفرانس انجمن کامپیوتر ایران, ص ۱۶۳-۱۵۳, ۱۳۷۹.

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

[22] 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, 2004.

---

1 Particle Swarm Optimization (PSO)
2 Mutation
3 Genetic Algorithms
4 Hill Climbing
5 Simulated Annealing
6 Cellular Learning Automata
7 Ackley
8 Rosenbrock
9 Rastrigin
10 Sphere