پژوهش درعلوم ومهندسی شماره ۲۶، جلد ۱، مرداد ۱۳۹۶ ، ص ۱۸-۱
ISSN: 1394-0040
http://jomea.iseas.ir

مجله علمی تخصصی پژوهش درعلوم ومهندسی (سال سوم)
شماره ۲۶ / جلد ۱ / مرداد ۱۳۹۶ / ص ۱۸- ۱

الگوریتم بهینه‌سازی دسته ذرات با تعریف سرعت مبتنی بر توزیع گوسی همبسته

حمید نصیری ، محمد مهدی عبادزاده

دانشجوی دکترای دانشکده مهندسی کامپیوتر و فناوری اطلاعات، دانشگاه صنعتی امیرکبیر، تهران
دانشیار دانشکده مهندسی کامپیوتر وفناوری اطلاعات دانشگاه صنعتی امیرکبیر، تهران

چکیده: امروزه در علوم مختلف با مسائل بهینه‌سازی گوناگونی سروکار داریم که برای حل آن‌ها روش‌های متعددی پیشنهاد شده است. یکی از این روش‌ها استفاده از الگوریتم‌های تکاملی و به ویژه الگوریتم بهینه‌سازی دسته ذرات می‌باشد. الگوریتم بهینه‌سازی دسته ذرات که برای اولین بار در سال ۱۹۹۵ توسط کندی و ابرهارت معرفی شد، یک تکنیک جدید و پیشرو در حل مسائل بهینه‌سازی می‌باشد. این الگوریتم با توجه به سادگی و کارآمدی‌اش توجه بسیاری از محققان را به خود جلب کرده است. در طی سالیان گذشته محققان زیادی برروی بهبود الگوریتم PSO کار کرده‌اند و با استفاده از روش‌های مختلف سعی در افزایش کارایی این الگوریتم داشته‌اند. یکی از این روش‌ها حذف مولفه سرعت ذرات می‌باشد. در این مقاله ما یک الگوریتم بهینه‌سازی دسته ذرات جدید ارائه کرده‌ایم که مولفه سرعت ذرات را در الگوریتم بهینه‌سازی دسته ذرات استاندارد حذف کرده و به جای آن از یک توزیع گاوسی برای تولید سرعت ذره استفاده می‌کند. در روش ارائه شده توزیع حرکت بهترین تجربه شخصی ذرات با یک توزیع گاوسی مدل شده و توسط الگوریتم یاد گرفته می‌شود و سپس از آن برای تولید سرعت ذره استفاده می‌شود. تفاوت روش پیشنهادی با سایر روش‌های موجود در یادگیری توزیع گاوسی برای تولید سرعت ذره می‌باشد. علاوه براین در سایر روش‌ها مولفه مکان ذره نیز به همراه مولفه سرعت ذره حذف می‌شود، اما در روش پیشنهادی این مقاله هر ذره مانند الگوریتم PSO استاندارد دارای مکان مختص به خود می‌باشد. آزمایشات ارزیابی و مقایسه روش فوق با سایر الگوریتم‌های پیشنهاد شده که در این دسته قرار می‌گیرند، نشان می‌دهد که روش پیشنهادی به نسبت سایر روش‌ها از دقت بالایی برخوردار است.

واژگان کلیدی: الگوریتم‌های تکاملی، بهینه‌سازی دسته ذرات، بهینه‌سازی دسته ذرات گاوسی.

پژوهش درعلوم ومهندسی شماره ۲۶، جلد ۱، مرداد ۱۳۹۶ ، ص ۱۸-۱
ISSN: 1394-0040
http://jomea.iseas.ir

مقدمه

امروزه در بسیاری از کاربردهای مهندسی با مسائل بهینه‌سازی سروکار داریم که محققان برای حل آن‌ها از روش‌های مختلفی بهره می‌گیرند. یکی از این روش‌ها الگوریتم‌های فراابتکاری و به ویژه الگوریتم‌های تکاملی می باشد. الگوریتم‌های تکاملی در سال‌های اخیر جایگاه مناسبی در بین پژوهشگران بدست آورده‌اند. از سال ۱۹۴۸ که ایده استفاده از جست‌وجوی ژنتیکی[1] یا جست‌وجوی تکاملی[2] توسط آلن تورینگ مطرح شد [۱] تا به حال الگوریتم‌های تکاملی بسیاری پیشنهاد شده‌اند که مهمترین آن‌ها در بین سال‌های ۱۹۶۰ تا ۱۹۷۵ معرفی شدند. برنامه‌ریزی تکاملی (EP) [۲] در سال ۱۹۶۶ توسط فوگل، اوونز و والش معرفی شد. هلند در سال ۱۹۷۳ الگوریتم ژنتیک (GA) [۳] را پیشنهاد کرد. در همان سال رشنبرگ الگوریتم استراتژی تکامل (ES) [۴] را ارائه کرد.

یکی دیگر از الگوریتم‌های فراابتکاری که به صورت گسترده مورد توجه محققان قرار گرفته است، الگوریتم بهینه‌سازی دسته ذرات[3] (PSO) می‌باشد. این الگوریتم که برای اولین بار در سال ۱۹۹۵ توسط کندی و ابرهارت [۵] معرفی شد؛ یک الگوریتم مبتنی بر جمعیت است که از مشاهده رفتار اجتماعی و گروهی ماهی‌ها و پرندگان الهام گرفته شده است. سادگی و کارآمدی این الگوریتم سبب شده است که توجه بسیاری از پژوهشگران به سوی آن جلب شود. PSO اگرچه در دسته الگوریتم‌های هوش جمعی قرار می‌گیرد، اما شباهت زیادی به الگوریتم‌های تکاملی دارد به نحوی که برخی محققان آن را جزو الگوریتم‌های تکاملی به حساب می‌آورند[۶].

الگوریتم PSO شامل دسته‌ای از ذرات می‌باشد که هرکدام از آن‌ها یک جواب ممکن را مشخص می‌کنند. در تشابه با الگوریتم‌های تکاملی، «دسته[4]» معادل «جمعیت[5]» و «ذره[6]» معادل «موجود[7]» می‌باشد. ذرات در فضای جست‌وجوی چند بعدی پرواز می‌کنند و مکان هر ذره با توجه به تجربیات خودش و تجربیات همسایگانش به‌روزرسانی می‌شود [۷]. برخلاف الگوریتم‌های تکاملی در PSO عملیات انتخاب (انتخاب والدین و انتخاب بازماندگان) وجود ندارد. این بدان معنا است که هیچ یک از ذرات حذف نمی‌شوند و تنها مکان و سرعت هر ذره تغییر می‌کند. علاوه بر این در الگوریتم PSO خبری از عملگرهای بازترکیبی و جهش نیست. هریک از مراحل فوق در الگوریتم‌های تکاملی پارامترهای خاص خودشان را دارند که در الگوریتم بهینه‌سازی دسته ذرات با توجه به اینکه این مراحل حذف شده‌اند، نیاز به مشخص کردن پارامترهای مربوط به آن‌ها نمی‌باشد. در نتیجه پیاده‌سازی آسان، پارامترهای کم و توانایی بالا در فرار از اکسترمم‌های محلی و همگرایی سریع از مزایای این الگوریتم می‌باشد [۸].

[1] Genetically Search
[2] Evolutionary Search
[3] Particle Swarm Optimization
[4] Swarm
[5] Population
[6] Particle
[7] Individual

پژوهش درعلوم ومهندسی شماره ۲۶، جلد ۱، مرداد ۱۳۹۶ ، ص ۱۸-۱
ISSN: 1394-0040
http://jomea.iseas.ir

همان‌طور که در بالا اشاره شد؛ الگوریتم PSO شامل دسته‌ای از ذرات می‌باشد که هریک از آن‌ها یک جواب ممکن مسئله را مشخص می‌کنند. هر یک از این ذرات دارای سرعت مختص به خود می‌باشند. در معادلات به‌روزرسانی سرعت و مکان ذرات، سرعت هر ذره در پارامتر وزن اینرسی[8] ضرب می‌شود. بسیاری از کارهایی که برای بهبود الگوریتم PSO انجام شده است برروی پارامتر وزن اینرسی تمرکز دارد. در روش پیشنهاد شده در این مقاله پارامتر وزن اینرسی حذف شده و همچنین به جای بردار سرعت از یک عدد تصادفی با توزیع نرمال استفاده شده است. به نحوی که میانگین و کوواریانس این توزیع با توجه به بهترین تجربه شخصی هر ذره به‌روزرسانی می‌شود. در ادامه این مقاله در بخش دوم الگوریتم بهینه‌سازی دسته ذرات شرح داده می‌شود. در بخش سوم به مرور برخی بهبودهای انجام شده برروی الگوریتم PSO می‌پردازیم. روش پیشنهادی مقاله در بخش چهارم شرح داده شده است. پس از آن در بخش پنجم روش پیشنهادی را با سایر روش‌ها مقایسه کرده و در نهایت در بخش ششم به جمع‌بندی و نتیجه‌گیری در رابطه با روش پیشنهادی می‌پردازیم.

الگوریتم بهینه‌سازی دسته ذرات

الگوریتم PSO شامل دسته‌ای از ذرات می‌باشد که هرکدام از آن‌ها یک جواب ممکن را مشخص می‌کنند. این ذرات در فضای جست‌وجوی D بعدی مسئله پرواز می‌کنند و مکان هر ذره با توجه به تجربیات خودش و همسایگانش به‌روزرسانی می‌شود. الگوریتم بهینه‌سازی دسته‌ذرات دارای دو نسخه متفاوت می‌باشد که تفاوت اصلی آن‌ها در اندازه همسایگی بین ذرات می‌باشد. این دو نسخه عبارتند از: Global Best PSO (gbest PSO) و Local Best PSO (lbest PSO). در ادامه این بخش به توضیح gbest PSO می‌پردازیم که در آن همسایگی برای هر ذره شامل تمام ذرات می‌باشد.lbest PSO در [۷] به تفصیل شرح داده شده است.

در gbest PSO هر ذره دارای مکان و سرعت مختص به خود می‌باشد که در هر مرحله به‌روزرسانی می‌شوند. $x_i(t)$ مکان ذره i ام را در فضای جست‌وجو در لحظه t مشخص می‌کند. سرعت ذره در لحظه $t+1$ از جمع سرعت ذره در لحظه t، ضریبی از اختلاف مکان فعلی ذره با بهترین تجربه شخصی ذره و ضریبی از اختلاف مکان فعلی ذره با بهترین تجربه گروه بدست می‌آید (t بازه‌های زمانی گسسته را مشخص می‌کند). همچنین مکان ذره در لحظه $t+1$ با اضافه کردن سرعت جدید ذره به مکان فعلی آن حاصل می‌شود. در نتیجه در هر مرحله سرعت و مکان ذرات با توجه به روابط (۱) و (۲) به‌روزرسانی می‌شوند [۹].

$$v_{ij}(t+1) = v_{ij}(t) + R_{1ij}c_1\left(P_{ij} - x_{ij}(t)\right) + R_{2ij}c_2\left(P_{gj} - x_{ij}(t)\right) \quad (۱)$$

$$x_i(t+1) = x_i(t) + v_i(t+1) \quad (۲)$$

[8] Inertia Weight

پژوهش درعلوم ومهندسی شماره ۲۶، جلد ۱، مرداد ۱۳۹۶ ، ص ۱۸-۱
ISSN: 1394-0040
http://jomea.iseas.ir

که $v_{ij}(t+1)$ نشان‌دهنده بعد j ام از بردار سرعت مربوط به ذره i ام در لحظه $t+1$ است. R_{1ij} و R_{2ij} دو عدد تصادفی در بازه $[0,1]$، c_1 و c_2 ضرایب شتاب[9]، P_i بهترین مکانی است که ذره i ام تا به حال پیدا کرده است و P_g بهترین مکانی است که تمامی ذرات در فضای جست‌وجو یافته‌اند. همچنین $x_i(0)\sim U(x_{min}, x_{max})$، که x_{min} و x_{max} به ترتیب حد پایین و حد بالای فضای جست‌وجو می‌باشند.

رابطه (۱) را می‌توان به سه مولفه اصلی تجزیه کرد که عبارتند از: ۱- مولفه اینرسی[10] ۲- مولفه شناختی[11] ۳- مولفه اجتماعی[12]، که در ادامه به شرح هریک از آن‌ها می‌پردازیم. در رابطه (۱)، $v_{ij}(t)$ سرعت قبلی ذره را مشخص می‌کند. این مولفه در واقع حافظه‌ای است که جهت قبلی پرواز هر ذره را نگه می‌دارد. به این مولفه حافظه می‌توان به چشم ممانی نگاه کرد که از تغییر جهت ناگهانی ذرات جلوگیری می‌کند و سبب می‌شود ذرات تمایل بیشتری به حفظ جهت فعلی خود داشته باشند. از این رو به آن مولفه اینرسی نیز می‌گویند. از عبارت $R_{1ij}c_1\left(P_{ij}-x_{ij}(t)\right)$ در رابطه (۱) به عنوان مولفه شناختی یاد می‌شود. این عبارت دانشی است که ماحصل تجربیات شخصی ذره می‌باشد. به بیان دیگر مولفه شناختی متناسب است با فاصله ذره از بهترین تجربه‌ای که تا به حال داشته است. عبارت $R_{2ij}c_2\left(P_{gj}-x_{ij}(t)\right)$ در رابطه (۱) مولفه اجتماعی را مشخص می‌کند. این عبارت دانشی است که در اثر تبادل اطلاعات یک ذره با سایر ذرات دسته بدست آمده است. مولفه اجتماعی متناسب است با فاصله ذره از بهترین مکانی که تمام ذرات در فضای جست‌وجو یافته‌اند [۷]. الگوریتمی که از هر سه مولفه فوق استفاده نماید، الگوریتم PSO کامل[13] نامیده می‌شود.

کندی در [۱۰] مدل‌های سرعت مختلف الگوریتم PSO را مورد بررسی قرار داد و آن‌ها را به سه دسته مدل‌های شناختی محض[14]، اجتماعی محض[15] و از خود گذشته[16] تقسیم‌بندی کرد. مدل‌های شناختی محض، مولفه اجتماعی را از معادله سرعت (رابطه (۱)) کنار می‌گذارند و با استفاده از رابطه (۳) سرعت ذره را به‌روزرسانی می‌کنند. به عبارت دیگر در حالتی که $c_2=0$ و $c_1\neq 0$ باشد. الگوریتم PSO کامل به مدل شناختی محض تبدیل می‌شود.

$$v_{ij}(t+1)=v_{ij}(t)+R_{1ij}c_1\left(P_{ij}-x_{ij}(t)\right) \qquad (۳)$$

مدل‌های شناختی محض به نسبت الگوریتم بهینه‌سازی دسته ذرات کامل، احتمال شکست بالاتری داشته و تمایل به یک جست‌وجوی محلی در ناحیه‌ای که ذرات مقداردهی اولیه شده‌اند، دارند. این مدل‌ها برای رسیدن به یک جواب خوب به تعداد تکرار بیشتری نیاز دارند و زمانی که ثابت c_1 کوچک باشد با شکست مواجه می‌شوند. اگرچه ضعف عملکرد این مدل‌ها در

[9] Acceleration Coefficients
[10] Inertia Component
[11] Cognitive Component
[12] Social Component
[13] Full PSO
[14] Cognition Only Model
[15] Social Only Model
[16] Selfless

پژوهش درعلوم ومهندسی شماره ۲۶، جلد ۱، مرداد ۱۳۹۶ ، ص ۱۸-۱
ISSN: 1394-0040
http://jomea.iseas.ir

[۱۱] تایید شده است. اما در محیط‌های پویا و در روش‌های Niching با موفقیت مورد استفاده قرار گرفته‌اند [۱۲]. مدل‌های اجتماعی محض، مولفه شناختی را از معادله سرعت حذف و با استفاده از رابطه (۴) سرعت را به‌روزرسانی می‌نمایند.

$$v_{ij}(t+1) = v_{ij}(t) + R_{2ij}c_2\left(P_{gj} - x_{ij}(t)\right) \qquad (۴)$$

در حالتی که $c_1 = 0$ و $c_2 \neq 0$ باشد، الگوریتم PSO کامل به مدل اجتماعی محض تبدیل می‌شود. در این مدل‌ها ذرات تمایلی به بازگشت به بهترین مکان قبلی خود ندارند و تمام آن‌ها به سوی بهترین مکان در همسایگی‌شان جذب می‌شوند. کندی در [۱۰] به صورت تجربی نشان داد که مدل‌های اجتماعی محض، سریع‌تر و کارآمدتر از مدل‌های شناختی محض و همچنین الگوریتم PSO کامل عمل می‌کنند. این مدل‌ها در محیط‌های پویا نیز نتایج قابل قبولی بدست آورده‌اند [۱۲].

دسته سوم مدل‌های سرعت، مدل‌های از خود گذشته می‌باشند. این مدل‌ها مشابه مدل‌های اجتماعی هستند. با این تفاوت که همسایگی مربوط به بهترین تجربه گروهی فقط از همسایگی ذره انتخاب می‌شود. به عبارت دیگر خود ذره نمی‌تواند بهترین ذره در همسایگی شود و برای محاسبه P_{gj} خود ذره در نظر گرفته نمی‌شود. کندی نشان داد که مدل‌های از خود گذشته در تعداد کمی از مسائل سریع‌تر از مدل‌های اجتماعی محض هستند [۱۰]. با این وجود این مدل‌ها در محیط‌های پویا و متغیر عملکرد ضعیفی دارند [۱۲].

یکی از جنبه‌های مهمی که کارآمدی و دقت یک الگوریتم بهینه‌سازی را مشخص می‌کند؛ مصالحه بین اکتشاف[17] و استخراج[18] می‌باشد. اکتشاف، توانایی الگوریتم جست‌وجو برای گشتن نواحی مختلف فضای جست‌وجو برای یافتن یک نقطه بهینه مناسب می‌باشد. در مقابل اکتشاف، استخراج قرار دارد. استخراج در واقع توانایی جست‌وجو اطراف یک ناحیه امیدبخش برای بهبود یک جواب کاندید می‌باشد. الگوریتم بهینه‌سازی خوب است که بتواند تعادل مناسبی بین این دو هدف که در تناقض با یکدیگر قرار دارند، برقرار کند. در الگوریتم PSO میزان اکتشاف یا استخراج به معادله به‌روزرسانی سرعت وابسته است. در نسخه اولیه الگوریتم PSO مولفه سرعت ذرات به خصوص برای ذراتی که از بهترین تجربه گروه و بهترین تجربه شخصی فاصله زیادی دارند، به سرعت افزایش می‌یابد. در نتیجه مکان ذرات تغییرات زیادی خواهد داشت و این امر سبب می‌شود که ذرات از محدوده فضای جست‌وجو خارج شده و واگرا شوند [۷]. برای کنترل اکتشاف سراسری ذرات، سرعت آن‌ها محدود شده و در صورتی که از یک مقدار بیشینه عبور کند برابر مقدار بیشینه سرعت قرار می‌گیرد [۱۳]. یکی دیگر از راه‌های ایجاد تعادل بین استخراج و اکتشاف استفاده از وزن اینرسی می‌باشد.

وزن اینرسی اولین بار توسط شی و ابرهارت [۱۴] به عنوان مکانیزمی برای کنترل توانایی اکتشاف و استخراج ذرات و همچنین مکانیزمی برای حذف محدودیت سرعت، پیشنهاد شد [۱۵]. اگرچه وزن اینرسی به خوبی توانست توانایی اکتشاف و استخراج ذرات را کنترل نماید ولی نتوانست محدودیت سرعت را به طور کامل از بین ببرد. وزن اینرسی (w) در واقع ممان ذرات را از طریق وزن‌دهی میزان مشارکت سرعت‌های قبلی کنترل می‌کند. به عبارت دیگر وزن اینرسی کنترل می‌کند که چه

[17] Exploration
[18] Exploitation

پژوهش درعلوم ومهندسی شماره ۲۶، جلد ۱، مرداد ۱۳۹۶ ، ص ۱۸-۱
ISSN: 1394-0040
http://jomea.iseas.ir

تعداد از جهت‌های پرواز قبلی برروی سرعت جدید تاثیر بگذارند. براساس الگوریتمی که شی و ابرهارت پیشنهاد دادند، سرعت ذرات با استفاده از رابطه (۵) به‌روزرسانی می‌شود.

$$v_{ij}(t+1) = w\, v_{ij}(t) + R_{1ij}c_1\left(P_{ij} - x_{ij}(t)\right) + R_{2ij}c_2\left(P_{gj} - x_{ij}(t)\right) \quad (۵)$$

مقدار w برای اطمینان از همگرایی و همچنین برقراری مصالحه بهینه بین اکتشاف و استخراج از اهمیت بالایی برخوردار است [۷]. مقدار زیاد w جست‌وجوی سراسری را بهبود می‌بخشد. در حالی که مقدار کم آن باعث جست‌وجوی محلی می‌شود. با تغییر وزن اینرسی به صورت پویا، توانایی جست‌وجو نیز به صورت پویا تنظیم می‌شود [۹]. روش‌هایی که مقدار w را به صورت تطبیقی تنظیم می‌کنند، معمولاً با یک مقدار زیاد اینرسی شروع کرده و با گذشت زمان آن را کاهش می‌دهند. این امر باعث می‌شود تا ذرات در گام‌های اولیه الگوریتم به اکتشاف بپردازند و هرچه زمان می‌گذرد، گرایش بیشتری به استخراج پیدا کنند. در اینجا لازم است به رابطه مهم بین مقادیر w وثابت‌های شتاب اشاره کنیم. انتخاب مقدار w باید به نحوی انجام گیرد که با مقادیر c_1 و c_2 در ارتباط باشد. ون دن برگ و انگلبرت [۱۶ و ۱۷] نشان دادند در صورتی که w در رابطه (۶) صدق کند، همگرایی ذرات تضمین می‌شود.

$$w > \frac{1}{2}(c_1 + c_2) - 1 \quad (۶)$$

کارهای گذشته

از همان ابتدا که الگوریتم PSO توسط کندی و ابرهارت ارائه شد، محققان بسیاری در جهت بهبود این الگوریتم گام برداشتند. در این بخش به معرفی برخی از کارهایی که سبب بهبود الگوریتم بهینه‌سازی دسته ذرات شده‌اند، می‌پردازیم. به طور کلی روش‌های ارائه شده برای بهبود الگوریتم PSO را می‌توان به هفت دسته تقسیم کرد که عبارتند از: ۱- بهبود وزن اینرسی، ۲- تطبیق ضرایب شتاب، ۳- اصلاح ساختار توپولوژیک، ۴- ترکیب با روش‌های جست‌وجوی دیگر، ۵- الگوریتم‌های بهینه‌سازی دسته ذرات موازی، ۶- الگوریتم‌های بهینه‌سازی دسته ذرات چند دسته‌ای و ۷- روش‌های مبتنی بر حذف سرعت، که در ادامه به معرفی برخی از آن‌ها می‌پردازیم.

بهبود وزن اینرسی

همان‌طور که در بخش قبلی نیز به آن اشاره شد، یکی از راه‌های بهبود الگوریتم PSO، تغییر وزن اینرسی به صورت پویا می‌باشد. روش‌هایی که وزن اینرسی را به صورت پویا تغییر می‌دهند در پنج دسته قرار می‌گیرند که عبارتند از: ۱- تنظیم تصادفی، ۲- کاهش خطی، ۳- کاهش غیرخطی، ۴- اینرسی تطبیقی فازی، و ۵- افزایش اینرسی، که در ادامه به بررسی هریک از این دسته‌ها می‌پردازیم.

پژوهش درعلوم ومهندسی شماره ۲۶، جلد ۱، مرداد ۱۳۹۶ ، ص ۱۸-۱
ISSN: 1394-0040
http://jomea.iseas.ir

تنظیم تصادفی: در هر تکرار یک وزن اینرسی متفاوت به صورت تصادفی انتخاب می‌شود. یک راه برای این کار نمونه‌برداری از یک توزیع گاوسی است. به عنوان مثال $w\sim N(0.72,\sigma)$ که σ آنقدر کوچک در نظر گرفته می‌شود تا اطمینان حاصل شود که w عمدتاً کوچکتر از یک است. پنگ و دیگران [۱۸] از رابطه (۷) برای تولید w تصادفی استفاده کردند.

$$w = c_1 r_1 + c_2 r_2 \qquad (۷)$$

کاهش خطی: یک وزن اینرسی اولیه بزرگ (معمولاً ۰,۹) به صورت خطی به یک مقدار کوچک کاهش می‌یابد (معمولاً ۰,۴). ابرهارت و شی [۲۰-۱۹] روشی را پیشنهاد دادند که از طریق رابطه (۸)، w را به صورت خطی از یک مقدار اولیه w_{max} به یک مقدار نهایی w_{min} کاهش می‌دهد.

$$w(iter) = \frac{iter_{max} - iter}{iter_{max}}(w_{max} - w_{min}) + w_{min} \qquad (۸)$$

کاهش غیر خطی: یک وزن اینرسی اولیه بزرگ به صورت غیر خطی به یک مقدار کوچک کاهش می‌یابد. روش‌های کاهش غیر خطی به نسبت روش‌های کاهش خطی، زمان اکتشاف کمتری دارند ولی زمان استخراجشان بیشتر است. روش‌های کاهش غیر خطی برای فضاهای جست‌وجوی هموار مناسب‌تر هستند. پرام و دیگران [۲۱] از رابطه (۹) برای کاهش غیرخطی وزن اینرسی استفاده کردند.

$$w(t+1) = \frac{(w(t) - 0.4)(n_t - t)}{n_t + 0.4} \qquad (۹)$$

که در آن $w(0) = 0.9$ و n_t حداکثر تکرار الگوریتم می‌باشد.

اینرسی تطبیقی فازی: وزن اینرسی به صورت پویا براساس مجموعه‌ها و قوانین فازی تنظیم می‌شود. شی و ابرهارت [۲۲] یک سیستم فازی برای تطبیق اینرسی طراحی کردند که بخش‌های آن عبارتند از: الف) دو ورودی یکی برای مشخص کردن بهترین ذره و دیگری مقدار فعلی وزن اینرسی، ب) یک خروجی برای مشخص کردن تغییر در وزن اینرسی، پ) سه مجموعه فازی با نام‌های LOW، MEDIUM و HIGH که به ترتیب با توابع تعلق چپ مثلثی، مثلثی و راست مثلثی پیاده‌سازی شده‌اند. ت) ۹ قانون فازی که توسط آن‌ها تغییرات وزن اینرسی محاسبه می‌شود. عبارت مقابل مثالی از این قوانین را نشان می‌دهد: اگر بهترین شایستگی کم است و مقدار فعلی وزن اینرسی نیز کم است، آنگاه تغییر در وزن اینرسی زیاد خواهد بود [۱۳ و ۲۲].

افزایش اینرسی: وزن اینرسی به صورت خطی از ۰,۴ به ۰,۹ افزایش می‌یابد [۲۳]. روش‌های کاهش خطی و غیرخطی که در بالا به آن‌ها اشاره شد، شباهت بسیاری به تغییر پارامتر دما در الگوریتم شبیه‌سازی ذوب فلزات[19] دارند [۲۴ و ۲۵].

[19] Simulated Annealing

پژوهش درعلوم ومهندسی شماره ۲۶، جلد ۱، مرداد ۱۳۹۶ ، ص ۱۸-۱
ISSN: 1394-0040
http://jomea.iseas.ir

تطبیق ضرایب شتاب

برای تطبیق ضرایب شتاب (c_1 و c_2) روش‌های متعددی ارائه شده است.ضرایب شتاب متغیر با زمان اولین بار توسط راتناویرا و همکارانش [۲۶] ارائه شد. آن‌ها برای تغییر c_1 و c_2 از روابط (۱۰) و (۱۱) استفاده کردند.

$$c_1 = (c_{1f} - c_{1i})\frac{iter}{MAXITR} + c_{1i} \qquad (۱۰)$$

$$c_2 = (c_{2f} - c_{2i})\frac{iter}{MAXITR} + c_{2i} \qquad (۱۱)$$

که در آن c_{1i}، c_{1f}، c_{2i} و c_{2f} ثابت هستند، $iter$ شماره تکرار فعلی و $MAXITR$ حداکثر تکرار مجاز الگوریتم است. در روش ارائه شده توسط راتناویرا و همکارانش پارامتر c_1 از ۲,۵ تا ۰,۵ و پارامتر c_2 از ۰,۵ تا ۲,۵ تغییر می‌کند. از دیگر کارهای انجام شده در این زمینه می‌توان به [۲۷] و [۲۸] اشاره کرد. هو و همکارانش [۲۸] روشی را پیشنهاد کردند که در آن هر سه پارامتر w، c_1 و c_2 به صورت تطبیقی تغییر می‌کنند.

اصلاح ساختار توپولوژیک

انگیزه معرفی ساختار شبکه اجتماعی در PSO، بهبود تنوع ذرات با استفاده از کنترل همسایه‌ها می‌باشد [۲۹ و ۳۰]. در الگوریتم PSO ذرات درون یک همسایگی با تبادل اطلاعات در مورد موفقیت هر ذره درون آن همسایگی با یکدیگر ارتباط برقرار می‌کنند. سپس تمام ذرات به سمت مکانی که فکر می‌کنند بهتر است، حرکت می‌کنند. عملکرد PSO به شدت به ساختار توپولوژیک آن وابسته است. کندی و مندز [۳۱] ساختارهای توپولوژیک مختلفی را ارائه کردند که توپولوژی حلقه و توپولوژی ون نیومن مثال‌هایی از آن‌ها می‌باشند. اگرچه تحقیقات بسیاری با استفاده از توپولوژی‌های مختلف انجام شده است؛ اما هیچ یک از این ساختارها برای تمام مسائل بهترین نیست. به طور کلی ساختارهای تماماً متصل برای مسائل تک قله‌ای و ساختارهای با اتصالات کمتر برروی مسائل چندقله‌ای عملکرد بهتری دارند [۷]. از جمله کارهای دیگر انجام شده در زمینه اصلاح ساختار توپولوژیک می‌توان به [۳۲] اشاره کرد.

ترکیب با روش‌های جست‌وجوی دیگر

از آنجایی که روش‌های جست‌وجوی مختلف توانایی‌های متفاوتی دارند، یک ایده طبیعی ترکیب PSO با روش‌های جست‌وجوی دیگر می‌باشد. یک راه سرراست برای این کار ترکیب PSO با الگوریتم‌های دیگر مانند الگوریتم ژنتیک [۳۲]، تکامل تفاضلی[20] [۳۳] و الگوریتم کلونی مورچه‌ها[21] [۳۴ و ۳۵] می‌باشد. ایده مهم دیگر ترکیب PSO با روش‌های جست‌وجوی محلی می‌باشد [۳۶ و ۳۷].

[20] Differential Evolution
[21] Ant Colony Algorithm

پژوهش درعلوم ومهندسی شماره ۲۶، جلد ۱، مرداد ۱۳۹۶ ، ص ۱۸-۱
ISSN: 1394-0040
http://jomea.iseas.ir

الگوریتم PSO موازی

اولین موازی‌سازی برای PSO توسط شوت و همکارانش [۳۸] انجام شد. آن‌ها یک پیاده‌سازی همگام از این الگوریتم را با روش پایه و پیرو[۲۲] با نام PSPSO ارائه کردند. در این روش پردازشگر پایه همه محاسبات الگوریتم به جز ارزیابی تابع هدف را انجام می‌دهد و ارزیابی تابع هدف بر عهده پردازشگر پیرو می‌باشد. الگوریتم آن‌ها برروی مسئله بهینه‌سازی یک سیستم بیومکانیک تست شد [۳۹] و نشان داده شده که عملکرد آن وقتی که زمان لازم برای ارزیابی تابع هدف برای ذرات مختلف به طور قابل توجهی متفاوت می‌باشد، کاهش می‌یابد، که علت این امر همگام بودن الگوریتم می‌باشد. از جمله کارهای دیگری که در زمینه موازی‌سازی الگوریتم انجام شده است می‌توان به [۴۰-۴۳] اشاره کرد.

الگوریتم PSO چند دسته‌ای

در این روش‌ها که به آن‌ها روش‌های تعاونی[۲۳] نیز می‌گویند، مسئله به زیربخش‌هایی شکسته می‌شود و پس از بهینه‌سازی هریک از آن‌ها جواب‌های بدست آمده با هم ترکیب می‌شوند تا جواب مسئله بهینه‌سازی بدست آید. از جمله اولین کارهای انجام شده در این زمینه در [۴۴] گزارش شده است که در آن زیردسته‌ها با یکدیگر همکاری می‌کنند تا مسائل بهینه‌سازی با ابعاد بالا را حل کنند. لی و یائو [۴۵] یک الگوریتم تعاونی مبتنی بر گروه‌بندی پویا ارائه کردند که از گروه‌بندی تصادفی برای تعیین زیردسته‌ها استفاده می‌کند. در الگوریتم ارائه شده توسط آن‌ها از ساختار همسایگی حلقه استفاده شده است.

روش‌های مبتنی بر حذف سرعت

ایده دیگری که می‌تواند موجب بهبود الگوریتم بهینه‌سازی دسته ذرات شود حذف مولفه سرعت ذرات می‌باشد. به این ترتیب که مولفه سرعت ذرات حذف شده و به جای آن از یک توزیع آماری برای تولید مکان جدید ذرات استفاده می‌شود. از جمله کارهای انجام شده می‌توان به الگوریتم Bare Bone PSO که توسط کندی [۴۶] پیشنهاد شد اشاره کرد. در این الگوریتم در هر گام مکان جدید هر ذره در هر یک از ابعاد به صورت تصادفی از یک توزیع گاوسی انتخاب می‌شود که میانگین این توزیع، میانگین بهترین تجربه شخصی ذره و بهترین تجربه گروه و انحراف معیار آن برابر با فاصله بهترین تجربه شخصی ذره و بهترین تجربه گروه می‌باشد. رابطه (۱۲) فرمول به‌روزرسانی مکان ذرات در این الگوریتم را نشان می‌دهد.

$$x_{t+1} = \mathcal{N}\left(\frac{P_t + g_t}{2}, |P_t - g_t|\right) \qquad (۱۲)$$

در رابطه (۱۲) مشخص است که عبارت سرعت از فرمول به‌روزرسانی مکان ذره حذف شده است و مکان جدید ذره با استفاده از یک توزیع گاوسی تولید می‌شود.

[22] Master-Slave
[23] Cooperative

پژوهش درعلوم ومهندسی شماره ۲۶، جلد ۱، مرداد ۱۳۹۶ ، ص ۱۸-۱
ISSN: 1394-0040
http://jomea.iseas.ir

محققان دیگر برای حذف مولفه سرعت از توزیع‌های لوی[۲۴] و کوشی به جای توزیع گاوسی استفاده کرده‌اند. توزیع لوی فرم کلی‌تری از توزیع گاوسی و کوشی دارد. شکل این توزیع توسط پارامتر α قابل کنترل است. اگر $\alpha = ۲$ باشد، توزیع لوی به یک توزیع گاوسی تبدیل می‌شود و در صورتی که $\alpha = ۱$ باشد معادل توزیع کوشی خواهد بود. از آن‌جایی که دو توزیع لوی و کوشی دنباله پهنی دارند، به نسبت توزیع گاوسی توانایی بیشتری در فرار از مینیمم‌های محلی دارند و عملکرد بهتری را ارائه می‌دهند [۴۹-۴۷]. در روش ارائه شده توسط سکرست و لامونت [۵۰] به جای نمونه‌برداری حول نقطه میانی بهترین تجربه شخصی و بهترین تجربه گروهی، از یک توزیع گاوسی استفاده شده و با احتمال P حول بهترین تجربه گروهی و با احتمال $1 - P$ حول بهترین تجربه شخصی نمونه‌برداری می‌شود. در نتیجه ذرات به جای اینکه فقط حول نقطه میانی بهترین تجربه گروهی و بهترین تجربه شخصی اکتشاف نمایند، می‌توانند در فضای جست‌وجوی وسیع‌تری این کار را انجام دهند.

لی و یائو [۴۵] در روش پیشنهادیشان از هر دو توزیع کوشی و گاوسی برای نمونه‌برداری استفاده کردند. در نتیجه الگوریتم ارائه شده توسط آن‌ها علاوه بر اینکه به خوبی همگرا می‌شود، قابلیت اکتشاف بالایی نیز دارد. در روش فوق مکان بعدی هر ذره با احتمال P با استفاده از توزیع کوشی نمونه‌برداری و با احتمال $1 - P$ از توزیع گاوسی برای تولید مکان بعدی ذره استفاده می‌شود که احتمال P را کاربر تعیین می‌نماید. لازم به ذکر است که P را می‌توان برابر ۰.۵ قرار داد. در نتیجه در نیمی از اوقات از توزیع کوشی و در نیمی دیگر از توزیع گاوسی برای نمونه‌برداری استفاده می‌شود.

روش پیشنهادی اگرچه در دسته روش‌های مبتنی بر حذف سرعت قرار می‌گیرد، اما یک فرق اساسی با سایر روش‌های این دسته دارد و آن این است که مولفه مکان ذره را حذف نمی‌کند. در روش‌هایی که در بالا توضیح داده شد، مولفه‌های سرعت و مکان ذره حذف شده و مکان ذره در هر لحظه با نمونه‌برداری از یک تابع توزیع بدست می‌آید. در روش ارائه شده در این مقاله مانند الگوریتم بهینه‌سازی دسته ذرات استاندارد مکان فعلی هر ذره نگهداری می‌شود و به جای مولفه سرعت از یک توزیع گاوسی استفاده می‌شود. به بیان دیگر در روش‌های قبلی، مکان هر ذره با استفاده از یک توزیع گاوسی بدست می‌آمد، اما در روش پیشنهادی سرعت ذره با توزیع گاوسی بدست آمده و در نهایت با اضافه کردن سرعت به مکان قبلی ذره، مکان جدید آن حاصل می‌شود. در بخش بعدی به شرح تفصیلی روش پیشنهادی می‌پردازیم.

روش پیشنهادی

همان‌طور که در بخش قبلی نیز اشاره شد، در روش پیشنهادی مقاله حاضر مولفه سرعت از الگوریتم PSO استاندارد حذف شده و به جای آن از یک توزیع گاوسی برای نمونه‌برداری و تعیین سرعت هر ذره استفاده می‌شود. در روش پیشنهادی لی و یائو [۴۵] که در بخش قبلی به مرور آن پرداختیم، از یک توزیع گاوسی استفاده می‌شود که میانگین این توزیع میانگین بهترین تجربه شخصی ذره و بهترین تجربه گروه و انحراف معیار آن برابر با فاصله بهترین تجربه شخصی ذره و بهترین تجربه گروه می‌باشد. اما در روش پیشنهادی به ازای هر ذره توزیع حرکت Pbest آن ذره یادگرفته می‌شود و از آن برای تولید سرعت ذره استفاده می‌شود. برای این کار فرض می‌کنیم توزیع حرکت Pbest هریک از ذرات یک توزیع نرمال با میانگین $\bar{P}$ ، ماتریس

[24] Levy

پژوهش درعلوم ومهندسی شماره ۲۶، جلد ۱، مرداد ۱۳۹۶ ، ص ۱۸-۱
ISSN: 1394-0040
http://jomea.iseas.ir

کوواریانس Σ_P و مسئله n بعدی باشد. در نتیجه فرمول تابع چگالی به فرم رابطه (۱۳) خواهد بود و برای بدست آوردن پارامترهای توزیع Pbest ذرات از روش متوسط گیری پنجره ای استفاده می کنیم.

$$f(p) = \frac{1}{(2\pi)^{\frac{n}{2}}|\Sigma_P|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}\left((P-\bar{P})^T \Sigma_P^{-1}(P-\bar{P})\right)\right) \quad (۱۳)$$

$$\bar{P} = (1-\alpha)\bar{P}_{old} + \alpha P \quad (۱۴)$$

$$\Sigma_P = \gamma((1-\alpha)\Sigma_{P_{old}} + \alpha(P-\bar{P})(P-\bar{P})^T) \quad (۱۵)$$

حال با شروع از یک مقدار اولیه برای میانگین و ماتریس کوواریانس ذرات هربار که Pbest یکی از ذرات بهتر شد از طریق روابط (۱۴) و (۱۵) میانگین و کوواریانس ذره را به‌روزرسانی می‌کنیم. پس از یادگیری میانگین و کوواریانس توزیع حرکت Pbest هریک از ذرات برای تولید سرعت ذره به صورت تصادفی از این توزیع استفاده می‌کنیم. برای این کار از یک توزیع گاوسی چند متغیره استفاده می‌شود که ماتریس کوواریانس آن برابر با کوواریانس توزیع حرکت Pbest ذره است و بردار میانگین آن در همه ابعاد مقدار صفر دارد. در نتیجه مکان ذرات با استفاده از رابطه (۱۶) به‌روزرسانی خواهد شد.

$$x_i(t+1) = x_i(t) + V_{randi} + R_{1i}c_1\left(P_i - x_i(t)\right) + R_{2i}c_2\left(P_g - x_i(t)\right) \quad (۱۶)$$

که در آن $x_i(t)$ مکان ذره i ام در لحظه t، V_{randi} بردار سرعت است که با استفاده از توزیع گاوسی به صورت تصادفی تولید شده است، R_{1i} و R_{2i} اعداد تصادفی در بازه $[0,1]$ می‌باشند، P_i بهترین تجربه شخصی ذره i و P_g بهترین تجربه گروه ذرات است. جهت ارزیابی عملکرد روش پیشنهادی در بخش بعدی آن را با سایر روش‌های ارائه شده و همچنین الگوریتم بهینه سازی دسته ذرات استاندارد مقایسه می‌نماییم.، با توجه به اینکه روش پیشنهادی مقاله در دسته روش‌های مبتنی بر حذف سرعت قرار می‌گیرد، در بخش بعد آن را با روش‌هایی که در این دسته جای می‌گیرند، مقایسه می‌نماییم.

آزمایشات ارزیابی

همان‌طور که در بخش قبل نیز اشاره شد برای بررسی عملکرد روش پیشنهادی، آن را با الگوریتم بهینه‌سازی دسته ذرات استاندارد و برخی از روش‌هایی که در دسته روش‌های مبتنی بر حذف سرعت قرار می‌گیرند، مقایسه می‌کنیم. برای مقایسه از ۷ تابع که قبلاً در [۴۹] به عنوان توابع ارزیابی مورد استفاده قرار گرفته‌اند؛ استفاده شده است. توابع مورد استفاده در جدول (۱) آمده‌اند. همچنین حد پایین و بالای فضای جست‌وجوی توابع در جدول (۲) قابل مشاهده می‌باشد.

پژوهش درعلوم ومهندسی شماره ۲۶، جلد ۱، مرداد ۱۳۹۶ ، ص ۱۸-۱
ISSN: 1394-0040
http://jomea.iseas.ir

جدول ۱- توابع ارزیابی برای مقایسه الگوریتم‌ها

$$f_1(x) = \sum_{i=1}^{N} x_i^2$$

$$f_2(x) = \sum_{i=1}^{N} (\sum_{j=1}^{i} x_j)^2$$

$$f_3(x) = \sum_{i=1}^{N-1} \{100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2\}$$

$$f_4(x) = \sum_{i=1}^{N} x_i \sin(\sqrt{|x_i|})$$

$$f_{5(x)} = \sum_{i=1}^{N} \{x_i^2 - 10\cos(2\pi x_i) + 10\}$$

$$f_6(x) = -20\exp\left\{-0.2\sqrt{\frac{1}{N}\sum_{i=1}^{N} x_i^2}\right\} -$$

$$\exp\{\frac{1}{N}\sum_{i=1}^{N}\cos(2\pi x_i)\} + 20 + e$$

$$f_7(x) = \frac{1}{4000}\sum_{i=1}^{N} x_i^2 - \prod_{i=1}^{N}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$$

جدول ۲- حد پایین و بالای فضای جست‌وجوی توابع ارزیابی

f_1, f_2	f_3	f_4	f_5	f_6	f_7
± 100	± 30	± 500	± 5.12	± 32	± 600

توابع f_1، f_2 و f_3 توابعی تک قله‌ای هستند و توابع f_4 تا f_7 توابعی چند قله‌ای می‌باشند. هر الگوریتم ۱۰۰ بار برروی هر تابع اجرا شده است و ابعاد مسئله برای تمام توابع ۳۰ بعدی در نظر گرفته شده است. تعداد ذرات دسته برای تمامی الگوریتم‌ها یکسان و برابر ۲۰ می‌باشد و از توپولوژی مش برای تعیین بهترین تجربه گروه استفاده شده است. به ازای هراجرای الگوریتم مکان اولیه ذرات به صورت تصادفی در فضای جست‌وجو مشخص شده است. هر بار اجرای الگوریتم تا ۳۰۰۰ تکرار ادامه یافته و بهترین شایستگی در این ۳۰۰۰ تکرار تعیین شده است و در نهایت میانگین بهترین شایستگی به ازای این ۱۰۰ تکرار گزارش شده است.

الگوریتم‌هایی که روش پیشنهادی با آن‌ها مقایسه شده است، عبارتند از: الگوریتم بهینه‌سازی دسته ذرات استاندارد، Gaussian PSO [۵۰]، Gaussian Bare Bone [۴۶] و Levy PSO [۴۷]. علاوه بر این در [۴۷] از توزیع لوی در Bare Bone PSO استفاده شده که روش پیشنهادی با آن نیز مقایسه شده است. جزئیات روش فوق در [۴۷] قابل مشاهده است. برای الگوریتم بهینه‌سازی دسته ذرات استاندارد از روش کلرک و کندی [۵۱] استفاده شده است. میانگین بهترین شایستگی به ازای هرمسئله و هر الگوریتم به همراه انحراف معیار آن در جدول (۳) آمده است.

پژوهش درعلوم ومهندسی شماره ۲۶، جلد ۱، مرداد ۱۳۹۶ ، ص ۱۸-۱
ISSN: 1394-0040
http://jomea.iseas.ir

جدول ۳- نتایج حاصل از مقایسه الگوریتم‌های مختلف برروی ۷ تابع ارزیابی

الگوریتم	f_1	f_2	f_3	f_4	f_5	f_6	f_7
PSO (انحراف معیار)	2.68E-11 (1.57E-10)	17.46 (25.99)	49.06 (37.10)	-8514.35 (447.179)	92.55 (21.62)	2.37 (1.91)	.087 (.236)
Gaussian PSO	2.50E-4 (.002)	91.98 (341.97)	36.35 (35.48)	-7478.38 (566.15)	125.33 (28.57)	11.40 (3.19)	.579 (1.094)
Gaussian Bare Bones	2.43E-41 (1.20E-40)	10.55 (15.56)	33.03 (30.14)	-8503.11 (495.83)	79.25 (19.35)	1.27 (1.21)	.019 (.028)
Levy PSO	1.80E-11 (9.93E-11)	22.16 (20.14)	43.62 (37.90)	**-9478.15 (421.14)**	47.56 (16.98)	.398 (2.20)	.029 (.040)
Levy Bare Bones	8.27E-44 (3.05E-43)	4.24 (3.74)	42.05 (38.30)	-9233.43 (440.61)	62.61 (16.86)	**.012 (.116)**	.013 (.014)
روش پیشنهادی	**6.26E-78 (2.29E-78)**	**4.40E-79 (1.95E-78)**	**16.32 (13.18)**	-8384.69 (524.66)	**45.07 (10.84)**	.043 (.068)	**8.86E-3 9.69E-3**

همان‌طور که مشاهده می‌شود روش پیشنهادی به نسبت سایر روش‌ها در توابع f_1، f_2، f_3، f_5 و f_7 عملکرد بهتری داشته است. به ازای تابع f_4 روش Levy PSO بهترین نتیجه را بدست آورده است و در نهایت به ازای تابع f_6 الگوریتم Levy Bare Bones که با جایگزین کردن توزیع لوی به جای توزیع گاوسی در Gaussian Bare Bone حاصل شده است، بهترین عملکرد را از خود نشان داده است. با توجه به نتایج بدست آمده و با مقایسه پاسخ‌های بدست آمده توسط روش پیشنهادی با سایر روش‌ها به نظر می‌رسد که الگوریتم پیشنهادی بهترین عملکرد را برروی توابع تک قله‌ای (توابع f_1، f_2 و f_3) داشته است و در این مسائل پاسخ‌های بدست آمده توسط آن اختلاف چشم‌گیری با سایر روش‌ها دارد. در رابطه با توابع چند قله‌ای (توابع f_4 تا f_7) عملکرد روش پیشنهادی قابل قبول بوده و اختلاف چندانی با سایر روش‌ها ندارد.

جمع‌بندی و نتیجه‌گیری

در این مقاله یک روش بهینه‌سازی دسته ذرات جدید پیشنهاد شد که در دسته روش‌های بهبود الگوریتم PSO با حذف مولفه سرعت قرار می‌گیرد. در این روش مولفه سرعت از الگوریتم PSO استاندارد حذف شده و به جای آن از یک توزیع گاوسی برای نمونه برداری و تعیین سرعت هر ذره استفاده می‌شود. برای این کار به ازای هر ذره توزیع حرکت Pbest آن ذره یاد گرفته می‌شود و از آن برای تولید سرعت ذره استفاده می‌شود. آزمایشات ارزیابی برروی مسائل مختلف نشان می‌دهند که روش ارائه شده در مجموع به نسبت سایر روش‌های مبتنی بر حذف سرعت از عملکرد بهتری برخوردار است. لازم به ذکر است که روش پیشنهادی این مقاله بهترین عملکرد را برروی توابع تک قله‌ای دارد و در این گونه توابع با اختلاف زیادی از سایر روش‌ها پیشی گرفته است. علاوه بر این عملکرد آن برروی توابع چند قله‌ای قابل قبول بوده و قابل مقایسه با سایر روش‌ها می‌باشد.

مراجع

- [1] A. M. Turing, "Intelligent machinery, a heretical theory," The Turing Test: Verbal Behavior as the Hallmark of Intelligence, vol. 105, 1948.
- [2] L. J. Fogel, A. J. Owens, and M. J. Walsh, "Artificial intelligence through simulated evolution," 1966.
- [3] J. H. Holland, "Genetic algorithms and the optimal allocation of trials," SIAM Journal on Computing, vol. 2, pp. 88-105, 1973.
- [4] I. Rechenberg, "Evolution Strategy: Optimization of Technical systems by means of biological evolution," Fromman-Holzboog, Stuttgart, vol. 104, 1973.
- [5] R. Eberhart and J. Kennedy, "Particle swarm optimization, proceeding of IEEE International Conference on Neural Network," Perth, Australia, pp. 1942-1948, 1995.
- [6] V. Kachitvichyanukul, "Comparison of three evolutionary algorithms: GA, PSO, and DE," Industrial Engineering and Management Systems, vol. 11, pp. 215-223, 2012.
- [7] A. P. Engelbrecht, Computational intelligence: an introduction: John Wiley & Sons, 2007.
- [8] R. Chiong, Nature-inspired algorithms for optimisation vol. 193: Springer, 2009.
- [9] A. Nickabadi, M. M. Ebadzadeh, and R. Safabakhsh, "A novel particle swarm optimization algorithm with adaptive inertia weight," Applied Soft Computing, vol. 11, pp. 3658-3670, 2011.
- [10]J. Kennedy, "The particle swarm: social adaptation of knowledge," in Evolutionary Computation, 1997., IEEE International Conference on, 1997, pp. 303-308.
- [11]A. Carlisle and G. Dozier, "Adapting particle swarm optimization to dynamic environments," in International conference on artificial intelligence, 2000, pp. 429-434.

- [12]R. Brits, A. P. Engelbrecht, and F. Van den Bergh, "A niching particle swarm optimizer," in Proceedings of the 4th Asia-Pacific conference on simulated evolution and learning, 2002, pp. 692-696.
- [13]R. Eberhart, P. Simpson, and R. Dobbins, Computational intelligence PC tools: Academic Press Professional, Inc., 1996.
- [14]Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on, 1998, pp. 69-73.
- [15]Y. Shi, "Particle swarm optimization: developments, applications and resources," in evolutionary computation, 2001. Proceedings of the 2001 Congress on, 2001, pp. 81-86.
- [16]D. Van and F. Bergh, "An Analysis of Particle Swarm Optimizers [PH. D thesis]," Pretoria: Natural and Agricultural Science Department, University of Pretoria, 2001.
- [17]F. Van Den Bergh and A. P. Engelbrecht, "A study of particle swarm optimization particle trajectories," Information sciences, vol. 176, pp. 937-971, 2006.
- [18]J. Peng, Y. Chen, and R. Eberhart, "Battery pack state of charge estimator design using computational intelligence approaches," in Battery Conference on Applications and Advances, 2000. The Fifteenth Annual, 2000, pp. 173-177.
- [19]R. C. Eberhart and Y. Shi, "Comparing inertia weights and constriction factors in particle swarm optimization," in Evolutionary Computation, 2000. Proceedings of the 2000 Congress on, 2000, pp. 84-88.
- [20]Y. Shi and R. C. Eberhart, "Empirical study of particle swarm optimization," in Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on, 1999, pp. 1945-1950.
- [21]T. Peram, K. Veeramachaneni, and C. K. Mohan, "Fitness-distance-ratio based particle swarm optimization," in Swarm Intelligence Symposium, 2003. SIS'03. Proceedings of the 2003 IEEE, 2003, pp. 174-181.
- [22]Y. Shi and R. C. Eberhart, "Fuzzy adaptive particle swarm optimization," in Evolutionary Computation, 2001. Proceedings of the 2001 Congress on, 2001, pp. 101-106.
- [23]Y.-L. Zheng, L.-H. Ma, L.-Y. Zhang, and J.-X. Qian, "On the convergence analysis and parameter selection in particle swarm optimization," in Machine Learning and Cybernetics, 2003 International Conference on, 2003, pp. 1802-1807.
- [24]S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," science, vol. 220, pp. 671-680, 1983.
- [25]R. H. J. M. Otten and L. P. P. P. v. Ginneken, The annealing algorithm: Kluwer, B.V., 1989.

- [26]A. Ratnaweera, S. K. Halgamuge, and H. C. Watson, "Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients," IEEE Transactions on evolutionary computation, vol. 8, pp. 240-255, 2004.
- [27]R. Cheng and M. Yao, "Particle swarm optimizer with time-varying parameters based on a novel operator," Applied Mathematics & Information Sciences, vol. 5, pp. 33-38, 2011.
- [28]M. Hu, T. Wu, and J. D. Weir, "An adaptive particle swarm optimization with multiple adaptive methods," IEEE Transactions on Evolutionary computation, vol. 17, pp. 705-720, 2013.
- [29]P. N. Suganthan, "Particle swarm optimiser with neighbourhood operator," in Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on, 1999, pp. 1958-1962.
- [30]J. Kennedy, "Small worlds and mega-minds: effects of neighborhood topology on particle swarm performance," in Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on, 1999, pp. 1931-1938.
- [31]J. Kennedy and R. Mendes, "Population structure and particle swarm performance," in Evolutionary Computation, 2002. CEC'02. Proceedings of the 2002 Congress on, 2002, pp. 1671-1676.
- [32]J. Robinson, S. Sinton, and Y. Rahmat-Samii, "Particle swarm, genetic algorithm, and their hybrids: optimization of a profiled corrugated horn antenna," in Antennas and Propagation Society International Symposium, 2002. IEEE, 2002, pp. 314-317.
- [33]B. Niu and L. Li, "A novel PSO-DE-based hybrid algorithm for global optimization," Advanced intelligent computing theories and applications. With aspects of artificial intelligence, pp. 156-163, 2008.
- [34]N. Holden and A. A. Freitas, "A hybrid particle swarm/ant colony algorithm for the classification of hierarchical biological data," in Swarm Intelligence Symposium, 2005. SIS 2005. Proceedings 2005 IEEE, 2005, pp. 100-107.
- [35]P. Shelokar, P. Siarry, V. K. Jayaraman, and B. D. Kulkarni, "Particle swarm and ant colony algorithms hybridized for improved continuous optimization," Applied mathematics and computation, vol. 188, pp. 129-142, 2007.
- [36]J.-J. Liang and P. N. Suganthan, "Dynamic multi-swarm particle swarm optimizer with local search," in Evolutionary Computation, 2005. The 2005 IEEE Congress on, 2005, pp. 522-528.
- [37]Z.-H. Zhan, J. Zhang, Y. Li, and H. S.-H. Chung, "Adaptive particle swarm optimization," IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), vol. 39, pp. 1362-1381, 2009.

ISSN: 1394-0040
http://jomea.iseas.ir

- [38]J. F. Schutte, J. A. Reinbolt, B. J. Fregly, R. T. Haftka, and A. D. George, "Parallel global optimization with the particle swarm algorithm," International journal for numerical methods in engineering, vol. 61, p. 2296, 2004.
- [39]J. A. Reinbolt, J. F. Schutte, B. J. Fregly, B. I. Koh, R. T. Haftka, A. D. George, et al., "Determination of patient-specific multi-joint kinematic models through two-level optimization," Journal of biomechanics, vol. 38, pp. 621-626, 2005.
- [40]B. I. Koh, A. D. George, R. T. Haftka, and B. J. Fregly, "Parallel asynchronous particle swarm optimization," International Journal for numerical methods in engineering, vol. 67, pp. 578-595, 2006.
- [41]G. Venter and J. Sobieszczanski-Sobieski, "Parallel particle swarm optimization algorithm accelerated by asynchronous evaluations," Journal of Aerospace Computing, Information, and Communication, vol. 3, pp. 123-137, 2006.
- [42]V. Kalivarapu, J.-L. Foo, and E. Winer, "Synchronous parallelization of particle swarm optimization with digital pheromones," Advances in Engineering Software, vol. 40, pp. 975-985, 2009.
- [43]V. K. Kalivarapu and E. H. Winer, "Asynchronous parallelization of particle swarm optimization through digital pheromone sharing," Structural and Multidisciplinary Optimization, vol. 39, pp. 263-281, 2009.
- [44]F. Van den Bergh and A. P. Engelbrecht, "A cooperative approach to particle swarm optimization," IEEE transactions on evolutionary computation, vol. 8, pp. 225-239, 2004.
- [45]X. Li and X. Yao, "Cooperatively coevolving particle swarms for large scale optimization," IEEE Transactions on Evolutionary Computation, vol. 16, pp. 210-224, 2012.
- [46]J. Kennedy, "Bare bones particle swarms," in Swarm Intelligence Symposium, 2003. SIS'03. Proceedings of the 2003 IEEE, 2003, pp. 80-87.
- [47]T. J. Richer and T. M. Blackwell, "The Lévy particle swarm," in Evolutionary Computation, 2006. CEC 2006. IEEE Congress on, 2006, pp. 808-815.
- [48]X. Yao, Y. Liu, and G. Lin, "Evolutionary programming made faster," IEEE Transactions on Evolutionary computation, vol. 3, pp. 82-102, 1999.
- [49]C.-Y. Lee and X. Yao, "Evolutionary programming using mutations based on the Lévy probability distribution," IEEE Transactions on Evolutionary Computation, vol. 8, pp. 1-13, 2004.
- [50]B. R. Secrest and G. B. Lamont, "Visualizing particle swarm optimization-Gaussian particle swarm optimization," in Swarm Intelligence Symposium, 2003. SIS'03. Proceedings of the 2003 IEEE, 2003, pp. 198-204.

- [51]M. Clerc and J. Kennedy, "The particle swarm-explosion, stability, and convergence in a multidimensional complex space," IEEE transactions on Evolutionary Computation, vol. 6, pp. 58-73, 2002.