ارائه یک الگوریتم چندجمعیتی مبتنی بر ازدحام ذرات برای حل مسائل بهینه‌سازی پویا

صمد نجاتیان[۱،۲]، استادیار؛ وحیده رضایی[۲،۳]، استادیار؛ حمید پروین[۴،۵]، استادیار

۱- دانشکده مهندسی برق - واحد یاسوج - دانشگاه آزاد اسلامی - یاسوج - ایران - samad.nej.2007@gmail.com
۲- باشگاه پژوهشگران و نخبگان - واحد یاسوج - دانشگاه آزاد اسلامی - یاسوج - ایران
۳- دانشکده ریاضی - واحد یاسوج - دانشگاه آزاد اسلامی - یاسوج - ایران - vahidehrezaie@gmail.com
۴- دانشکده مهندسی کامپیوتر - دانشگاه آزاد اسلامی - فارس - نورآباد ممسنی - ایران - parvin@iust.ac.ir
۵- باشگاه پژوهشگران و نخبگان - واحد نورآباد ممسنی - دانشگاه آزاد اسلامی - فارس - ایران

چکیده: بسیاری از مسائل بهینه‌سازی در دنیای واقعی پویا می‌باشند. در این مسائل بهینه سراسری و بهینه‌های محلی در طول زمان تغییر می کنند. نشان داده‌شده که استفاده از الگوریتم‌های یادگیر تقلید از طبیعت برای مواجهه با این مسائل مناسب هستند. در میان الگوریتم‌های مختلف بهینه‌سازی برای محیط‌های پویا در سال‌های اخیر الگوریتم بهینه‌سازی گروه ذرات (PSO) توجه زیادی را به خود جلب کرده است. در این مقاله یک الگوریتم مبتنی بر PSO برای محیط‌های پویا ارائه شده‌است. این الگوریتم یک روش چندجمعیتی است که ذرات به دو دسته خنثی و کوانتومی تقسیم می‌شوند. تولید اولیه جمعیت در این‌روش بر اساس نظریه آشوب صورت می‌گیرد. نشان داده‌شده که روش‌های چندجمعیتی برای حفظ تنوع ذرات در محیط مناسب هستند. در این‌روش تولید زیرجمعیت‌ها به‌صورت تطبیقی صورت می‌گیرد. در این‌روش از عملگر کنترل ذرات خنثی استفاده‌شده است. این عملگر نواحی متروکه و بد را برای ذرات خنثی شناسایی می‌نماید. هم‌چنین در این‌روش به‌جای عملگر ضدهمگرایی که در روش مشابه mQSO معرفی شده‌است، از یک عملگر دیگر استفاده‌شده که کارآیی مناسب‌تری را از خود نشان داده است. در این‌روش برای بهبود جستجوی محلی در هر زیرجمعیت از یک روش تپه‌نوردی بهبودیافته استفاده شده است. آزمایش‌ها مختلفی بر روی روش پیشنهادی انجام‌گرفته است. نتایج این آزمایش‌ها نشان می‌دهد که الگوریتم پیشنهادی از سایر الگوریتم‌های مشابه در این زمینه مانند FMSO و mQSO10(5+5^{q}) و Cellular PSO و Multi Swarm PSO در تمام محیط‌های ارزیابی‌شده نتایج بهتری را به‌دست می‌آورد.

واژه‌های کلیدی: بهینه‌سازی ازدحام ذرات، مسائل بهینه‌سازی پویا، محک قله‌های متحرک، خطای برون‌خطی، چندجمعیتی.

Introducing a Multi Population Algorithm based on PSO for Solving Dynamic Optimization Problems

S. Nejatian[1,2], Assistant Professor; V. Rezaie[2,3], Assistant Professor; H. Parvin[4,5], Assistant Professor

1- Department of Electrical Engineering, Yasooj Branch, Islamic Azad University, Yasooj, Iran, Email: samad.nej.2007@gmail.com
2- Young Researchers and Elite Clubs, Yasooj Branch, Islamic Azad University, Yasooj, Iran
3- Department of Mathematic, Yasooj Branch, Islamic Azad University, Yasooj, Iran, Email: vahidehrezaie@gmail.com
4- Department of Computer Engineering, Noorabad Mamasani Branch, Islamic Azad University, Noorabad Mamasani, Fars, Iran, Email: parvin@iust.ac.ir
5- Young Researchers and Elite Clubs, Noorabad Mamasani Branch, Islamic Azad University, Noorabad Mamasani, Fars, Iran

Abstract: Many real-world problems are dynamic, requiring an optimization algorithm which is able to continuously track the changing optima over time. Typical examples include recognition of the moving objects in changing background; constructing financial trading models in various changing market conditions; data mining in continuously updating databases; scheduling problems with dynamic available resources; vehicle routing in traffic networks of dynamic traffic flow, etc. which requires the optimization algorithms to be able to find and track the changing optima efficiently over time. Among various algorithms for dynamic optimization, particle swarm optimization algorithms (PSOs) are attracting more and more attentions in recent years. We have proposed three algorithms based on PSO for dynamic environments. The proposed algorithms are the multi-swarm algorithms. In the traditional research it is shown that the multi swarm algorithms are suitable for creating diversity in the environments. In this paper, we propose an adaptive strategy for dynamic envirounments. The adaptive strategy increases convergence speed of the algorithm. The proposed algorithm uses several suitable operators for increasing the efficiency of the algorithm. One of the operators in the proposed algorithm is the trajectory control operator. The proposed algorithm has the best results rather than the state-of-the-art. The experimental study on the Moving Peaks Benchmark shows that the proposed approach is superior to several other well-known and state-of-the-art algorithms in dynamic environments.

Keywords: Particle Swarm Optimization, dynamic optimization problems, moving peaks benchmark, offline error, multi swarm.

تاریخ ارسال مقاله: ۱۳۹۵/۱۱/۳۰
تاریخ اصلاح مقاله: ۱۳۹۶/۰۴/۱۷ و ۱۳۹۶/۰۵/۲۳
تاریخ پذیرش مقاله: ۱۳۹۶/۰۷/۰۴
نام نویسنده مسئول: وحیده رضایی
نشانی نویسنده مسئول: ایران - یاسوج - دانشگاه آزاد اسلامی واحد یاسوج - دانشکده ریاضی.

۱- مقدمه

در دنیای واقعی معمولاً با مسائل پیچیده‌ای روبرو هستیم که این مسائل نیازمند بهینه‌سازی هستند. در واقع با بهینه‌نمودن این مسائل می‌توان موضوعاتی ازجمله زمان و هزینه را نیز بهینه نمود. به‌طورکلی می‌توان مسائل بهینه‌سازی را در دو گروه اساسی قرار داد: ۱) مسائل بهینه‌سازی ایستا ۲) مسائل بهینه‌سازی پویا.

در مسائل بهینه‌سازی ایستا، بهینه مسئله ثابت بوده و تغییر نمی‌کند، بنابراین ردیابی بهینه این مسائل تا حدودی کار آسانی هست. در مسائلی که ماهیت پویایی دارند بهینه مسئله در طول زمان تغییر می‌نماید و بنابراین ردیابی این بهینه قابل‌تغییر برای الگوریتم کار دشواری خواهد بود. با توجه به این‌که اکثر مسائل موجود در دنیای حقیقی پویا هستند، درنتیجه در این محیط‌ها نیاز به الگوریتم‌هایی است که علاوه‌بر یافتن بهینه متغیر، بتوانند به‌نحو مطلوبی این بهینه متغیر را دنبال نمایند. بنابراین در این محیط‌ها الگوریتم‌هایی مطلوب خواهند بود که میانگین خطای بهترین راه‌حل یافت‌شده در آن‌ها در هر لحظه از زمان کم‌ترین مقدار باشد. هرگاه یک تغییر در هدف بهینه‌سازی، نمونه مسئله یا بعضی تغییرات محدودیت یک مسئله بهینه‌سازی رخ دهد، ممکن است بهینه آن مسئله تغییر کند. اگر این حالت رخ دهد وفق دادن یک راه‌حل با راه‌حل قدیمی ضروری است. یک روش استاندارد برای روبه‌رو شدن با این پویایی، برخورد با هر تغییر به‌عنوان یک مسئله بهینه‌سازی جدید است که باید از ابتدا حل شود. این‌روش بعضی مواقع غیرعملی است، به‌دلیل این‌که حل یک مسئله از ابتدا، بدون استفاده مجدد از اطلاعات گذشته بسیار وقت‌گیر است. در بهینه‌سازی مسائل پیچیده، استفاده از روش‌های بهینه‌سازی دقیق غیرممکن است، بنابراین از روش‌های جستجوی تصادفی برای رسیدن به یک پاسخ نزدیک به بهینه استفاده می‌شود. در واقع در این‌روش‌ها پاسخ‌های مناسب برای حل مسئله، در یک بازه زمانی قابل‌قبول به‌دست خواهند آمد، اما هیچ تضمینی جهت به‌دست آوردن بهترین پاسخ وجود ندارد. در میان روش‌های جستجوی تصادفی الگوریتم‌های تکاملی که برگرفته از طبیعت هستند از جایگاه ویژه‌ای برخوردار هستند. با توجه به پیچیدگی بسیاری از مسائل بهینه‌سازی با ماهیتی پویا، توسعه‌هایی برای بهبود کارآیی این الگوریتم‌ها در حل مسائل بهینه‌سازی پویا انجام گرفته است. از آن‌جا که الگوریتم‌های تکاملی به‌طور رایج در حال تکامل هستند، یک کاندیدای مناسب برای حل مسائل بهینه‌سازی به‌نظر می‌آیند. یکی از الگوریتم‌های تکاملی مبتنی بر هوش جمعی، الگوریتم ازدحام ذرات (PSO) می‌باشد. این الگوریتم کارآیی مناسب خود را در محیط‌های ایستا نشان داده است [۱۳]. اما استفاده از این الگوریتم به‌تنهایی در محیط‌های پویا نمی‌تواند کارآیی مناسبی را از خود نشان دهد. در محیط‌های پویا سه مشکل اساسی وجود دارد:

اولین مشکل شناسایی زمان تغییرات محیط می‌باشد تا الگوریتم بتواند به‌موقع پاسخ مناسبی به این تغییرات از خود نشان دهد. دومین مشکل موجود در محیط‌های پویا حافظه منسوخ‌شده ذرات می‌باشد، زیرا با تغییر محیط بهترین تجربه شخصی یک ذره و بهترین تجربه جمعی یک گروه ممکن است دیگر معتبر و صحیح نباشد و درصورت عدم اصلاح این اطلاعات نادرست الگوریتم منحرف می‌گردد. سومین مشکل و اساسی‌ترین مشکل موجود در محیط‌های پویا مشکل از دست دادن تنوع می‌باشد، زیرا متنوع‌سازی یک گروه همگرا شده برای یافتن بهینه متحرک و سپس هم‌گرایی آن به بهینه موردنظر شدیداً کارایی الگوریتم را کاهش می‌دهد.

الگوریتم‌های ازدحام ذرات استاندارد به‌تنهایی قادر به ردیابی پاسخ‌های یک مسئله بهینه‌سازی با ماهیتی پویا نمی‌باشد، بنابراین، باید این الگوریتم را با یک سازوکاری مناسب ترکیب نمود تا بتواند به‌طور کارآیی، خود را با محیط در حال تغییر وفق داده و پاسخ‌های مسئله را ردیابی نمایند. یکی از چالش‌های اصلی برای حل مسائل بهینه‌سازی پویا توسط الگوریتم‌های تکاملی حفظ تنوع درحین اجرای الگوریتم می‌باشد. مشکل اصلی الگوریتم‌های تکاملی استاندارد، این است که عاقبت به یک بهینه هم‌گرا می‌شوند و بنابراین تنوع لازم را برای اکتشاف در محیط جدید از دست می‌دهند. هنگامی‌که جمعیت الگوریتم تکاملی هم‌گرا شد، توانایی‌اش برای وفق‌پذیری با تغییرات محیط را از دست می‌دهد. در این‌جا سعی شده تا با ترکیب یک استراتژی مناسب با الگوریتم ازدحام ذرات مدلی ارائه شود که به یک موازنه در جستجو برسد، به‌طوری‌که دچار هم‌گرایی زودرس نشده و هم‌چنین تنوع لازم را برای اکتشاف درحین اجرای الگوریتم داشته باشد.

محیط‌های پویا محیط‌هایی هستند که در آن‌ها بهینه‌های سراسری و محلی در گذر زمان جابه‌جا می‌شوند. نشان داده شده است که استفاده از الگوریتم‌های یادگیر الهام گرفته از طبیعت، در مواجهه با این محیط‌ها مناسب است. بهینه‌سازی جمعی ذرات، یکی الگوریتم‌های موفق در این زمینه است که مبتنی بر رفتار پرندگان در طبیعت می باشد.

در این مقاله با ارائه یک الگوریتم بهینه‌سازی مبتنی بر ازدحام ذرات چندجمعیتی، یک الگوریتم مناسب و کارا برای حل مسائل بهینه‌سازی پویا ارائه شده است. الگوریتم ازدحام ذرات دارای خصوصیاتی ازجمله سرعت هم‌گرایی بالا، حساس نبودن به مقادیر اولیهٔ پرندگان (ذرات)، انعطاف‌پذیری و تحمل‌پذیری خطا می‌باشد که آن را برای حل مسائل بهینه‌سازی قابل‌قبول می‌کند. در این‌روش از چندین راهکار مناسب برای حفظ تنوع، افزایش سرعت هم‌گرایی و به‌طورکلی افزایش کارآیی استفاده شده‌است. روش پیشنهادی در این مقاله یک روش چندجمعیتی است که با چندین راه‌کار مناسب دیگر ترکیب شده‌است. در این‌روش از تئوری آشوب برای تولید جمعیت اولیه ذرات استفاده شده‌است. روش پیشنهادی از عملگرهای مناسبی استفاده نموده تا اکثر چالش‌های پیش روی مسائل بهینه‌سازی پویا را رفع نماید.

هدف اصلی در محیط‌های پویا و به‌دنبال آن مسائل بهینه‌سازی پویا ردیابی نقطه (نقاط) بهینه و سپس یافتن هر چه دقیق‌تر آن(ها) می‌باشد. از این‌رو مسئله‌ای که حائز اهمیت است سرعت ردیابی و دنبال کردن بهینه(ها) می‌باشد. به‌عبارتی بایستی بتوان قبل از ایجاد تغییر محیطی بعدی، تخمین هر چه بهتری از جواب(های) بهینه داشت. به‌همین دلیل سعی بر آن است تا با بهره‌گیری از الگوریتم‌های هوشمند و تکاملی که الهام‌گرفته از تکامل طبیعی‌اند، بتوان اهداف مطرح‌شده را برآورده کرد. چرا که تکامل طبیعی یک فرآیند پیوسته‌ای از سازگاری با محیط است. بنابراین حل مسئله بهینه‌سازی پویا در دنیای واقعی یکی از اساسی‌ترین چالش‌ها محسوب می‌شود. در مسائل پویا، چالش‌های اساسی و مهمی پیش روی یک الگوریتم بهینه‌سازی می‌باشد. برخی از این چالش‌ها می‌توانند شامل مشکل به‌روزکردن حافظه افراد جمعیت بعد از هر تغییر در محیط، حفظ تنوع[1] در محیط بعد از هم‌گرایی افراد به بهینه موردنظر، ظرفیت محدود حافظه، شناسایی زمان تغییرات در محیط باشند.

هدف اصلی این پژوهش ارائه الگوریتمی کارا و مناسب برای حل مسائل بهینه‌سازی پویا می‌باشد. در این الگوریتم سعی شده‌است ابتدا یک الگوریتم مبتنی بر ازدحام ذرات برای حل مسائل بهینه‌سازی پویا ارائه شود. سپس در این‌روش با تنظیم مناسب پارامترها در بخش آزمایش‌ها، بهترین نتایج به‌دست آورده شده است. از آن‌جایی‌که حل مسائل بهینه‌سازی در محیط‌های پویا کار آسانی نمی‌باشد، بنابراین در این پژوهش با آزمایش‌های مختلف سعی‌شده تا یک مسئله مهم، که شبیه‌ساز مناسبی برای مسائل بهینه‌سازی پویاست را حل نمود.

در این مقاله نوآوری روش پیشنهادی به‌صورتی است که در ادامه تشریح شده است:

در این مقاله یک رویکرد جدید برای حل مسائل بهینه‌سازی پویا با استفاده از الگوریتم ازدحام ذرات چندجمعیتی ارائه شده‌است که این‌روش پیشنهادی به‌شرح زیر است:

۱- ارائه یک راه‌کار کنترلی جدید برای ذرات خنثی در فضای مسئله

۲- راه‌کار جدیدی برای روش پیشنهادی ارائه‌شده که جایگزین عملگر ضدهمگرایی در روش‌های مشابه شده‌است.

۳- نحوه چندجمعیتی و تولید اولیه جمعیت در این‌روش نسبت به روش‌های مشابه کاملاً متفاوت می‌باشد.

این مقاله شامل ۵ بخش است:

بخش اول به مقدمه و کلیات موضوع می‌پردازد. بخش دوم مقاله مروری بر کارهای گذشته دارد. بخش سوم روش پیشنهادی را تشریح می‌کند. بخش چهارم به ارزیابی روش پیشنهادی و نتایج تجربی می پردازد. بخش پنجم نتیجه‌گیری و کارهای آتی را بیان می‌کند.

۲- مروری بر کارهای گذشته

در زمینه مسائل بهینه‌سازی پویا روش‌های مختلفی توسط محققین ارائه شده‌است. هر کدام از این‌روش‌ها به‌نوعی برای رفع چالش‌های اساسی مسائل بهینه‌سازی در محیط‌های پویا پیاده‌سازی شده‌اند. درمیان روش‌های ارائه‌شده روش‌های ترکیبی از جایگاه ویژه‌ای برخوردار هستند. در ادامه برخی از این‌روش‌ها آورده شده‌اند.

لانگ[2] و دامسترسیو[3] [۱] یک الگوریتم به‌نام CESO را پیشنهاد دادند، به‌طوری‌که این‌روش از دو جمعیت با اندازه برابر برای شناسایی و دنبال‌کردن بهینه متحرک در محیط‌های پویا استفاده می‌کند که یکی از این جمعیت‌ها مسئول حفظ تنوع در فضای جستجو می‌باشد و جمعیت دیگر مسئول پیداکردن بهینه سراسری می‌باشد، حفظ تنوع در جمعیت با به‌کار بردن الگوریتم تفاضلات تکاملی مبتنی بر ازدحام[4] که برای بهینه‌سازی چندهدفه به‌کار می‌رود، صورت می‌پذیرد درحالی که ازدحام ذرات مستقیماً برای پیدا کردن بهینه سراسری مورد استفاده قرار می‌گیرد. یکی از مشکلات اصلی در محیط‌های پویا همگرایی زودرس به یک بهینه محلی می‌باشد. برای غلبه بر این مشکل CESO از جمعیت CRDE برای نگه‌داشتن یک مجموعه از بهینه‌های محلی و سراسری در کل فرآیند جستجو استفاده می‌کند. در این‌روش جمعیت swarm برای پیداکردن بهینه سراسری در طول کل فرآیند جستجو به‌کار می‌رود. هر دو جمعیت swarm و CRDE از رفتار طبیعی خود استفاده می‌کنند و هیچ سازوکار اضافی به رفتار آن‌ها اضافه نمی‌شود. در الگوریتم CESO هر دو جمعیت ابتدا مقداردهی شده و مورد ارزیابی قرار می‌گیرند. در شروع هر مرحله یک آزمایش برای شناسایی تغییر در محیط صورت می‌گیرد و برای این کار بهترین موقعیت یافت‌شده در جمعیت CRDE مورد ارزیابی مجدد قرار می گیرد، اگر تغییر در میزان برازندگی آن ایجادشده باشد مشخص می شود که تغییر در محیط رخ داده است.

در [۲] یک الگوریتم چندجمعیتی مبتنی بر الگوریتم کرم شب‌تاب را برای حل مسائل بهینه‌سازی پویا ارائه شده‌است. در این‌روش از نگاشت آشوب برای تولید جمعیت اولیه استفاده شده‌است. در این‌روش از روش چندجمعیتی در الگوریتم کرم شب‌تاب استفاده شده تا تنوع در سطح مناسبی حفظ شود.

هو و ابرهارت[5] یک روش مبتنی بر بهینه‌سازی جمعی ذرات به‌نام RPSO برای محیط‌های پویا ارائه کرده‌اند که در آن هر وقت تنوع لازم از بین می‌رود از یک جابه‌جایی تصادفی ذرات بهره می‌گیرند [۳].

در مرجع [۴] صادقی و همکارانش یک الگوریتم مبتنی بر ازدحام ذرات برای حل مسائل بهینه‌سازی پویا ارائه نموده‌اند. در این‌روش برای ردیابی سریع بهینه تغییر یافته از یک حافظه صریح استفاده شده است. در این‌روش از راه‌کار مناسبی برای به‌روزرسانی حافظه استفاده شده‌است. در این الگوریتم بهترین راه‌حل‌های گذشته که خیلی قدیمی نباشند در حافظه برای ردیابی بهینه در آینده استفاده شده‌است.

در سال ۲۰۰۹، نویسندگان مرجع [۵] الگوریتم ممتیکی ارائه دادند که در آن از مفهوم تپه‌نوردی جهت جستجوی محلی قوی‌تر استفاده شده‌است. هم‌چنین از دو متد ایجاد تنوع برای حل مشکل هم‌گرایی در مسائل بهینه‌سازی پویا، بهره گرفته شد. چارچوب

الگوریتم ممتیک براساس الگوریتم ژنتیک به این‌صورت است که جمعیتی از افراد تولید و ارزیابی می‌شوند. سپس بهترین فرد جمعیت انتخاب و توسط متدهای به کار گرفته‌شده برای جستجوی محلی، بهبود می‌یابد. در نهایت جمعیت جدید از بهترین افراد بین والدها و فرزندان ایجاد شده و سپس بهترین فرد ارزیابی و توسط جستجوی محلی بهبود می‌یابد. همچنین در الگوریتم ممتیک عملگرهای جستجوی محلی نقش مهمی در استخراج بهینه در فرآیند تکامل به عهده می‌گیرند. یکی از متدهای جستجوی محلی تپه‌نوردی است که در آن فرآیند جستجو از فرد جاری به فرد کاندید درصورت بهتربودن فرد کاندید انتقال می‌یابد. در این مقاله از دو راهکار برای تپه‌نوردی استفاده شده‌است:

۱- تپه‌نوردی بر پایه جفت‌گیری حریصانه

۲- تپه‌نوردی براساس جهش تند

در نوع اول تپه‌نوردی بهترین فرد به عنوان یک والد در عملگر جفت‌گیری و والد دوم از طریق انتخاب به روش چرخ رولت از بین جمعیت انتخاب می‌شوند. سپس بر روی این دو جفت‌گیری یکنواخت انجام‌شده تا یک فرزند تولید شود. درصورتی‌که فرزند بهتر از بهترین فرد در جمعیت قبلی باشد، جایگزین بهترین فرد می‌شود. در این‌جا از پارامتری به‌نام PC_{LS} استفاده‌شده که احتمال انجام جفت‌گیری در جستجوی محلی است. هر چقدر این پارامتر کوچک‌تر باشد، ناحیه‌ای اطراف بهترین فرد مورد جستجو قرار می‌گیرد. یعنی در فضای کوچک‌تری اطراف بهترین فرد، به‌دنبال راه‌حل بهتر می‌باشد. در نوع دوم چندین بیت در کروموزوم بهترین فرد تغییر می‌یابد. این بیت‌ها به‌طور تصادفی و به تعداد nm_{LS} انتخاب می‌شوند. اگر فرد جدید تولید شده بهتر از بهترین فرد در جمعیت باشد، در آن‌صورت جایگزین بهترین فرد خواهدشد. در این‌جا هر چقدر nm_{LS} بزرگ‌تر باشد، در فضای بزرگ‌تری اطراف بهترین فرد جستجوی محلی ادامه خواهد یافت. در نهایت هرکدام از دو نوع تپه‌نوردی بر اساس میزان رشد الگوریتم تعیین می‌شوند. سپس برای ایجاد تنوع در جمعیت از دو روش نگاشت دوگان سازگار[۶] و مهاجران تصادفی به‌راه‌اندازی‌شده[۷] استفاده می‌گردد.

رضوانیان و میبدی در سال ۲۰۱۰ [۶] براساس الگوریتم خودکار یادگیرنده و ترکیب با الگوریتم ایمنی مصنوعی، الگوریتمی ارائه دارند تا در رویکرد افزایش تنوع جمعیتی گام دیگری در محیط‌های پویا بردارند. الگوریتم ایمنی یک الگوریتم الهام‌گرفته شده از سیستم ایمنی بدن می‌باشد. از نظر تئوریک سیستم ایمنی بدن می‌تواند تغییرات محیطی را به‌خوبی مدیریت کند و در مقابل آن واکنش نشان دهد. با این وجود الگوریتم‌های ایمنی مصنوعی قادر به دنبال‌کردن تغییرات در محیط‌های پویا نمی‌باشند. بنابراین نیازمند تعدادی راه‌کار برای شناسایی و پاسخ به تغییرات هستند. خودکار یادگیرنده از مجموعه محدودی «عمل»[۸] درست شده که در هر مرحله یکی از این اعمال انتخاب شده و بر روی محیط اثر می‌گذارند. انتخاب هر عمل براساس یک «بردار احتمال عمل»[۹] در هر وضعیت[۱۰] می‌باشد. پس از این‌که عمل بر روی محیط اثرگذاشت، محیط به‌دنبال آن سیگنالی با توزیع احتمال نامشخص به خودکار یادگیرنده برمی‌گرداند. سپس براساس آن سیگنال و با استفاده از الگوریتم یادگیری، خودکار بردار احتمال عمل را به‌روزرسانی می‌کند. در این الگوریتم سعی شده تا با سازگاری پارامترها با محیط‌های پویا، به ایجاد نوعی تنوع در جمعیت کمک کند. در الگوریتم ایمنی عملگر جهش تنها عملگر به‌کاربرده شده بوده که توسط پارامتری به نام احتمال نرخ جهش p_m، به‌کارگیری یا عدم به‌کارگیری این عملگر بر روی سلول‌های ایمنی مشخص می‌شود. نتایج نشان داده که این الگوریتم توانسته به‌طور کارا بهینه را دنبال کند (البته با کمی نوسان).

در مرجع [۷] یک الگوریتم به‌نام EA-KDTree پیشنهادشده است. در این‌روش نواحی اکتشافی و غیراکتشافی به‌صورت تطبیقی از هم مجزا می‌شوند. الگوریتم نواحی جستجو را به چندین ناحیه تقسیم می کند. در این‌روش نواحی برای تشخیص بهینه متغیر پوشش داده می شوند. در این‌روش راه‌کار ساده‌ای استفاده شده‌است. مطابق این راه‌کار هر ناحیه برای پیدا نمودن بهینه مورد تخمین قرار می‌گیرد و ساختار داده ویژه‌ای به‌نام KD-Tree استفاده شده تا نواحی جستجو را به خاطرسپاری نماید و سرعت هم‌گرایی افزایش یابد.

در سال ۲۰۱۰، نویسندگان مرجع [۸] برای افزایش تنوع جمعیت از راه‌کار خود-سازگار کردن پارامتر نرخ جابه‌جایی[۱۱] در به‌کارگیری مهاجران در جمعیت برای رویارویی با مسائل بهینه‌سازی پویا استفاده کردند. به‌این‌ترتیب برای مهاجران مختلف، در مراحل مختلف فرآیند تکامل و تحت محیط‌های مختلف نرخ جابه‌جایی برای الگوریتم تغییر می‌کند. درصورتی‌که مهاجران اثر مثبتی بر روی الگوریتم بگذارند، باید نرخ جابه‌جایی را افزایش داد تا بتواند بهینه‌های جدید را موقع تغییر شناسایی و درصورتی‌که هیچ‌گونه تغییری در محیط رخ ندهد، نرخ جابه‌جایی باید کاهش یافته تا بتواند سرعت هم‌گرایی بالایی در جمعیت ایجاد کند. بنابراین برای خود-سازگار کردن نرخ جابه‌جایی در گام اول باید تخمینی از اثر مهاجران جاری بر الگوریتم (یعنی مثبت یا منفی بودن اثر مهاجران) داشت. نتایج نشان داده‌شده کارآیی الگوریتم پیشنهادی می‌تواند کارآیی مهاجران ایجادشده را در بسیاری از موارد بهبود بخشد.

روش جستجوی باکتریای جهت داده‌شده با ازدحام ذرات در مرجع [۹] پیشنهاد شده‌است. در این‌روش از ترکیب الگوریتم جستجوی باکتریای با ازدحام ذرات برای جایابی بهینه خازن‌ها و ژنراتورهای توزیع‌شده در شبکه‌های توزیع استفاده شده‌است.در این پژوهش با استفاده از الگوریتم ارائه‌شده مسئله چندهدفه جایابی خازن‌ها و ژنراتورها در شبکه توزیع شده بهینه می‌شود. تابع هدف در نظر گرفته‌شده شامل کاهش هزینه تلفات و بهبود پروفیل ولتاژ شبکه می باشد. در این‌روش ابتدا مکان بهینه خازن‌ها با استفاده از تحلیل شاخص ولتاژ مشخص شده و سپس با استفاده از الگوریتم پیشنهادی مقدار

توان تولیدی خازن‌ها محاسبه می‌گردد. روش ACPSO ترکیبی از دو الگوریتم APSO [۱۲] و CPSO [۱۳] می‌باشد. در این‌روش از مزایای ساختار تطبیقی APSO و ساختار تعاونی CPSO بهره گرفته شده‌است. ساختار تطبیقی APSO باعث می‌شود در هر مرحله از اجرای الگوریتم پارامترها با مناسب‌ترین مقدار خود، معادله سرعت را به‌روزرسانی نمایند. ساختار تعاونی در این‌روش باعث می‌شود: ۱) برای حل مسئله‌های با ابعاد بالا مفید باشد، ۲) با افزایش تنوع جمعیت از به دام افتادن در بهینه محلی جلوگیری کرده و نرخ هم‌گرایی را بهبود دهد، ۳) برخلاف دیگر روش‌ها که برخی از پارامترها را بهتر و برخی را بدتر می‌نمایند این‌روش در هر مرحله کلیه ابعاد مسئله را بهبود می‌دهد [۹].

محمدپور و پروین در مرجع [۱۰] یک الگوریتم ژنتیک آشوب‌گونه با ترکیب حافظه برای حفظ راه‌حل‌های مناسب و خوشه‌بندی جمعیت در داخل حافظه و جمعیت اصلی ارائه نموده‌اند. در این‌روش برخلاف الگوریتم ژنتیک استاندارد که جمعیت اولیه به‌صورت تصادفی ایجاد می شود، از تئوری آشوب برای ایجاد جمعیت اولیه استفاده شده‌است. در این‌روش از خوشه‌بندی مبتنی بر میانگین برای افزایش تنوع استفاده شده‌است. در این‌روش مرکز هر خوشه برابر میانگین افراد درون هر خوشه می‌باشد، و ایده موردنظر بدین‌صورت است که افراد (کروموزوم‌ها) در جمعیت اصلی و حافظه خوشه‌بندی می‌شوند. تعداد خوشه‌ها در جمعیت اصلی و جمعیت حافظه باهم برابرند. حافظه مورد استفاده در این‌روش از نوع حافظه صریح مستقیم می‌باشد. حافظه مورد استفاده طبق نظریه آشوب و براساس تابع نگاشت لجیستیکی مقداردهی اولیه می‌شود. اندازه حافظه مورد نظر (*mem_size*) برابر $0/1 \times N$ می‌باشد. که N اندازه جمعیت یا همان تعداد افراد جمعیت می‌باشد. به‌روزرسانی حافظه در بازه‌های زمانی تصادفی صورت می‌گیرد، بدین‌صورت که اگر به‌روزرسانی فعلی حافظه در بازه زمانی $rand(5,10)$ صورت گیرد، به‌روزرسانی بعدی در زمان $t + rand(5,10)$ می‌باشد. اندازه حافظه برای ذخیره‌سازی اطلاعات محدود می‌باشد بنابراین باید یک راه‌کار مناسب جهت به‌روزرسانی حافظه اتخاذ نمود. برای به‌روزرسانی حافظه از روش متفاوتی استفاده شده‌است. در این‌روش به هر یک از افراد موجود در حافظه یک مشخصه به‌نام سن اختصاص داده می‌شود. افراد در مرحله اول ورودشان به حافظه دارای سن صفر هستند و در هر نسل یک واحد به سن آن‌ها اضافه می‌گردد. در این‌روش زمانی‌که حافظه پر شود و نیاز به جایگزینی داشته باشد، سن افراد طبق یک رابطه ترکیب خطی از سن واقعی افراد و کارآیی آن‌ها محاسبه می‌شود. در این‌روش بازیابی از حافظه زمانی صورت می گیرد که محیط دچار تغییر شود. نحوه بازیابی از حافظه بدین‌صورت است که بهترین فرد از حافظه جایگزین بدترین فرد از جمعیت اصلی می‌شود. زمان تغییر در محیط بدین‌صورت شناسایی می‌شود که اگر شایستگی برای یکی از افراد جمعیت در ارزیابی مجدد تغییر نماید دراین‌صورت الگوریتم تغییر در محیط را شناسایی نموده و باید بازیابی از حافظه انجام گیرد.

کاموسی، هاشمی و میبدی در سال ۲۰۱۰ الگوریتم Multi-SwarmPSO را ارائه داده‌اند[۱۱]. آن‌ها در این‌روش از یک گروه والد و چند گروه فرزند در الگوریتم PSO استفاده کرده‌اند. گروه والد برای کاوش فضای جستجو و گروه فرزندان جهت استخراج‌کردن فضای جستجو به‌کار گرفته می‌شوند و هر دو گروه از الگوریتم بهینه‌سازی تجمعی ذرات با همسایگی محلی استفاده می‌کنند. به این‌صورت که ذرات به‌جای این‌که از بهترین ذره گروه یاد بگیرند، از بهترین ذره موجود در گروه مربوطه‌اش یاد می‌گیرند. در این الگوریتم ابتدا گروه والد مقداردهی اولیه شده و به جستجو در فضا می‌پردازد و در هر تکرار الگوریتم سرعت و موقعیت ذرات به‌روزرسانی می‌شود. سپس فاصله هر ذره از گروه والد با بهترین ذرات گروه‌های فرزند سنجیده و درصورتی که فاصله کمتر از r باشد جای مجذوب‌کننده (بهترین ذره در گروه فرزند) گروه فرزند با ذره گروه والد عوض شده تا جستجوی محلی حول آن ذره گروه والد ادامه یابد. سپس به‌جای ذره گروه والد یک ذره دیگر مقداردهی می‌شود. زمانی‌که برای همه ذرات گروه والد این فاصله سنجیده شد و ذرات دوباره مقداردهی شدند بهترین ذره گروه والد ارزیابی می‌شود.

رضازاده، میبدی و نائبی در سال ۲۰۱۱ [۱۲] از مفهوم چندگروهی در الگوریتم PSO استفاده کردند. دراین‌جا از رویکرد وزن تطبیقی، خوشه‌بندی فازی و اجرای جستجوی محلی برروی بهترین گروه، استفاده شده‌است. برای دنبال‌کردن بهینه هم از ذرات موج‌گونه بهره برده‌اند. دراین‌جا دو نوع گروه وجود دارد: یک گروه آزاد و دیگری گروه هم‌گرا. در ابتدا گروه‌ها به‌طور تصادفی ایجادشده که هر گروه شامل تعدادی ذرات PSO است. در هر مرحله موقعیت و سرعت ذرات به‌روزرسانی می‌شود.

در این الگوریتم از همسایگی کوچک ذرات برای افزایش تنوع و کاهش سرعت هم‌گرایی استفاده می‌گردد. در ضمن اجرا اگر تعداد گروه‌های آزاد از یک حد آستانه‌ای کمتر شد یک گروه جدید اضافه و اگر بیشتر از حد آستانه شد، بدترین گروه حذف می‌گردد. سپس بهترین گروه در فضای جستجو را پیدا کرده و جستجوی محلی حول بهترین موقعیت بهترین گروه انجام می‌گیرد. چون این گروه احتمال بیشتری دارد که به بهینه سراسری نزدیک باشد. همچنین از یک وزن تطبیقی برای سرعت‌گرفتن ذرات در هر مرحله استفاده می‌شود. از نکاتی‌که می‌توان به آن اشاره کرد این‌است‌که در این الگوریتم به‌دلیل استفاده از تعداد کمتری از ذرات نسبت به سایر الگوریتم‌ها، تعداد ارزیابی‌ها کاهش می‌یابد. همچنین به‌دلیل استفاده از وزن تطبیقی، شاهد هم‌گرایی سریع گروه به بهینه و به‌تبع آن پیداکردن جواب‌های بهتر بعد از رخ دادن تغییرات در محیط در مراحل اولیه اجرای الگوریتم خواهد بود.

بنابه‌این‌که پایه و اساس روش پیشنهادشده در این مقاله، الگوریتم ازدحام ذرات می‌باشد، بنابراین در این بخش الگوریتم ازدحام ذرات به اختصار تشریح می‌شود.

الگوریتم ازدحام ذرات (PSO) [۱۳] با یک گروه از جواب‌های تصادفی شروع به‌کار می‌کند. سپس برای یافتن جواب بهینه در فضای مسئله با به‌روزکردن موقعیت و سرعت هر ذره به جستجو می‌پردازد. هر ذره به‌صورت چندبعدی (بسته به طبیعت مسئله) با دو مقدار v_i^d، x_i^d که به‌ترتیب معرف مکان و سرعت مربوط به بعد d–ام از i– امین ذره هستند، تعریف می‌شود. در هر مرحله از حرکت جمعیت، هر ذره با توجه به دو مقدار بهترین به‌روز می‌شود. اولین مقدار بهترین، بهترین جواب از لحاظ شایستگی است که تاکنون برای هر ذره به‌طور جداگانه به‌دست آمده است. این مقدار بهترین برای هر فرد[۱۴] است و $pbest_i$ نامیده می‌شود. مقدار بهترین دیگر که توسط PSO به‌دست می‌آید، بهترین مقداریست که تاکنون توسط تمام ذره‌ها در میان جمعیت به‌دست آمده است، این مقدار بهترین سراسری[۱۵] است و $gbest$ نام دارد. پس از یافتن دو مقدار $pbest_i$ و $gbest$ هر ذره سرعت و مکان جدید خود را مطابق روابط ۱ و ۲ به‌روز می‌کند.

$$v_i^{d'} = wv_i^d + n_1 r_1\left(x_{pbest_i}^d - x_i^d\right) + n_2 r_2 (x_{gbest}^d - x_i^d) \quad (۱)$$

$$x_i^{d'} = x_i^d + v_i^{d'} \quad (۲)$$

$x_i^{d'}$ و x_i^d به‌ترتیب مکان فعلی و مکان قبلی مربوط به بعد d–ام از i–امین ذره می‌باشند. $v_{i'}^{d'}$ و v_i^d سرعت فعلی و سرعت قبلی مربوط به بعد d–ام از i–امین ذره هستند. به‌طوری‌که $w \in (0,1)$ و وزن اینرسی[۱۶] می‌باشد، n_1 و n_2 ضرایب شتاب[۱۷]، r_1 و r_2 اعداد تصادفی با توزیع یکنواخت در بازه (۱و۰) می‌باشند. برای جلوگیری از واگرایی الگوریتم، مقدار نهایی سرعت هر ذره در بازه $[-V_{MAX}, V_{MAX}]$محدود می‌شود. w، $n_1 r_1$ و $n_2 r_2$ از پارامترهای PSO هستند و هم‌گرایی الگوریتم وابسته به مقدار این پارامترهاست. معمولاً مقدار n_1، n_2 عددی بین ۱/۵ تا ۲ می‌باشد. هم‌گرایی شدیداً به مقدار w وابسته است و بهتر است به‌صورت پویا تعریف شود.

الگوریتم mQSO در مرجع [۱۴] ارائه شده‌است. این‌روش به الگوریتم ازدحام ذرات مبتنی بر ذرات کوانتوم[۱۸] معروف است. در این‌روش جمعیت به تعدادی از زیرگروه‌ها تقسیم می‌شود که به نقاط مختلف فضای جستجو رانده می‌شوند و هر گروه دارای تعدادی از ذرات کوانتوم می‌باشند که تنوع در گروه را تأمین می‌کند. در این الگوریتم، کل جمعیت به چند گروه تقسیم می‌شوند و شامل سه عمل‌گر تنوع با نام‌های ذرات کوانتوم، دفع و ضدهم‌گرایی می‌باشد. ذرات کوانتوم به عنوان ابزاری برای حفظ سطح خاصی از تنوع در یک گروه ارائه شده‌اند و آن‌ها از مدل‌های اتمی تأثیر می‌پذیرند. ذرات کوانتوم در موقعیت‌های تصادفی قرار می‌گیرند تا تنوع گروه‌ها را حفظ کنند. عمل‌گر دفع هرگاه دو گروه هم‌پوشانی پیدا می‌کنند، گروه بدتر را مقداردهی اولیه مجدد می‌کند. عمل‌گر ضدهم‌گرایی زمانی‌که تمام گروه‌ها هم‌گرا می شوند، گروه بدتر را مقداردهی اولیه مجدد می‌کند. از مزایای این‌روش به حفظ تنوع با راه‌کارهای ذرات کوانتوم و ضدهم‌گرایی می‌توان اشاره نمود. ازجمله معایب این‌روش می‌توان کاهش شدید کارآیی با افزایش شدت تغییرات در محیط و افزایش ابعاد مسئله را نام برد.

الگوریتم چندجمعیتی Adaptive mQSO در مرجع [۱۵] ارائه شده‌است. در این‌روش تعداد گروه‌ها از ابتدا معین نمی‌باشد و با تغییر در محیط و پیداکردن قله‌های جدید تعداد گروه‌ها افزایش می‌یابد. از آن‌جایی‌که تعداد قله‌ها از قبل معلوم نمی‌باشد، تعداد گروه‌ها نمی‌تواند به‌راحتی قابل‌تعیین باشد. به‌این‌دلیل در این‌روش، یک راهکار خودانطباقی برای تعداد گروه‌ها ارائه شده‌است. این راهکار کاملاً ساده می‌باشد. گروه‌های چندتایی درصورتی نیاز به یک گروه جدید دارند که تمام گروه‌های فعلی به قله موردنظر خود همگرا شده‌باشند ازطرف دیگر درصورتی‌که بسیاری از گروه‌های آزاد، ارزیابی‌های تابع را جذب کرده باشند بایستی یکی از آن‌ها حذف شود. اگر بیش از یک گروه آزاد وجود داشته باشد، انتخاب برای حذف به‌صورت اختیاری است و بدترین گروه‌های آزاد حذف می‌شوند. ازجمله معایب این‌روش این است که کارآیی الگوریتم در ابعاد بالا کاهش یافته و هم‌چنین با کاهش تعداد قله‌ها در فضای مسئله کارآیی نیز برای این‌روش کاهش می‌یابد.

در روش FMSO از یک گروه والد به‌عنوان یک گروه پایه برای شناسایی نواحی امیدبخش استفاده شده‌است و از گروه‌های فرزند برای جستجوی محلی استفاده می‌شود. هر گروه فرزند ناحیه مربوط به خود را جستجو می‌کند.

در این‌روش به‌نوعی یک توازن میان جستجوی محلی و جستجوی سراسری به‌وجود آمده‌است. در روش FMSO ناحیه جستجو به شکل یک دایره به مرکزیت بهترین ذره گروه در نظر گرفته‌می‌شود. هر ذره که دارای فاصله کم‌تری از شعاع دایره باشد (فاصله نزدیک‌تری تا بهترین ذره که در مرکز قرار دارد) به آن گروه فرزند تعلق می‌یابد. در FMSO هر گروه فرزند دارای یک شمارنده خطا می‌باشد، که وظیفه آن کنترل مقدار برازندگی بهترین موقعیت یافت‌شده گروه فرزند می‌باشد. اگر در هر مرحله بهبودی در مقدار برازندگی آن ایجاد نشده باشد، مقدار شمارش‌گر خطا یک واحد افزایش پیدا می‌کند و درصورتی‌که مقدار شمارش‌گر خطا به بیشینه مقدار خود برسد، درآن‌صورت از یک عملگر جهش برروی بهترین ذره برای پرش به موقعیت جدید استفاده می‌شود. تعداد گروه‌های فرزند با بهبود بهترین موقعیت یافت‌شده گروه والد افزایش پیدا می‌کند ولی مقدار آن از ماکزیمم مقدار تعیین‌شده برای آن تجاوز نمی‌نماید درصورتی‌که تعداد گروه‌های فرزند به بیشینه مقدار آن برسد گروه جدید جایگزین آن گروه فرزند که دارای شمارنده خطای بزرگ‌تری است می‌گردد. این‌روش در فرکانس‌های تغییرات پایین کارآیی چندانی از خود نشان نمی‌دهد و این می‌تواند یکی از عیوب این‌روش محسوب شود. همچنین در این‌روش با افزایش شدت تغییرات در محیط کارآیی افت محسوسی می‌نماید [۱۶].

مورد دیگر استفاده از خودکار سلولی توسط هاشمی و میبدی در سال ۲۰۱۱ [۱۷] بود که به الگوریتم Cellular PSO معروف می‌باشد.

در این‌روش از الگوریتم PSO برای حرکت افراد جمعیت در خودکار سلولی بهره گرفته شده‌است. در این‌جا نیز نحوه ایجاد تنوع همانند الگوریتم کلونی مورچه سلولی انجام می‌گیرد. با این تفاوت که سعی شده تا با بهره‌گیری از تبادل اطلاعات بین سلول‌های همسایه تنوع بین سلول‌ها حفظ شود. بنابراین در الگوریتم PSO به‌جای این‌که از بهترین موقعیت موجود در جمعیت در به‌روزرسانی سرعت ذره بهره برده شود، از بهترین موقعیت پیداشده در همسایگان سلول استفاده می‌شود.

در سال ۲۰۱۴ نویسندگان مرجع [۱۸] یک روش چندجمعیتی تطبیقی مبتنی بر ازدحام ذرات باعنوان AMSO ارائه نموده‌اند. در این‌روش تولید چندجمعیتی مطابق یک راه‌کار تطبیقی صورت می گیرد. در این‌روش برای تطبیقی ساختن جمعیت براساس تغییرات محیطی از یک روش خوشه‌بندی استفاده شده‌است. در این‌روش تمام جمعیت‌ها از عملگر مشابهی برای جستجو در فضای مسئله استفاده می کنند. در این الگوریتم راهکار با عنوان ازدحام بیش از حد استفاده شده است. این راه‌کار بیان می‌کند که اگر ازدحام ایجادشده منجر به ارضای برخی از معیارها شود، جمعیت‌های غیرضروری حذف‌شده و درنتیجه باعث صرفه‌جویی در محاسبات می‌شود. در این الگوریتم از نرخی به‌نام *drop rate* استفاده می‌شود تا زمانی‌که تغییری در محیط رخ دهد تنوع در فضای مسئله حفظ شود. در این الگوریتم به‌منظور معرفی تعداد مناسبی از ذرات فعال که در هر محیط خاص مورد نیاز است، یک روش تطبیقی ارائه شده‌است. این‌روش تطبیقی با توجه به اطلاعات جمع‌آوری‌شده از کل جمعیت ذرات و آخرین تغییر در محیط ارائه شده‌است. این الگوریتم دارای سه مرحله اساسی است:

۱- آماده‌سازی مجدد برای چندجمعیتی تطبیقی

۲- چندجمعیتی تطبیقی

۳- پیاده‌سازی الگوریتم به‌وسیله الگوریتم ازدحام ذرات

در این‌روش برای تولید جمعیت‌ها، ابتدا فضای مسئله به چندین زیرناحیه تقسیم می‌شود. در این‌روش برای تقسیم‌بندی زیرنواحی از خوشه‌بندی سلسله‌مراتبی اتصال‌کوتاه استفاده شده‌است. تولید تطبیقی زیرجمعیت‌ها در این‌روش با روش ارائه‌شده در این مقاله کاملاً متفاوت است. در این‌روش برای حفظ تنوع در فضای مسئله از خوشه‌بندی استفاده شده‌است ولی در روش پیشنهادی این مقاله از خوشه‌بندی استفاده نشده‌است. یکی از عیوب روش‌هایی که از خوشه‌بندی استفاده می‌کنند، افزایش پیچیدگی محاسباتی الگوریتم به‌دلیل استفاده از روش‌های خوشه‌بندی است.

در سال ۲۰۱۱ نویسندگان مرجع [۱۹] روشی با عنوان mQSOE را ارائه نموده‌اند. این الگوریتم نسخه توسعه‌یافته الگوریتمی است که در مرجع [۱۴] ارائه شده‌است. این الگوریتم همانند الگوریتم پیشنهادی این مقاله از الگوریتم mQSO به‌عنوان الگوریتم پایه استفاده نموده و نویسندگان آن‌را توسعه داده‌اند. در این‌روش راه‌کارهایی در سطح دسته ذرات برای افزایش کارآیی الگوریتم ارائه شده‌است. در این الگوریتم ابتدا هر دسته به دو گروه تقسیم می‌شود. این تقسیم‌بندی بر اساس کیفیت ذرات در مواجهه با چالش از دست‌دادن تنوع و کنترل تعداد دسته‌های فعال درحین اجرای قوانین فازی انجام می‌گیرد. پس روش ارائه‌شده در این مقاله شامل دو مرحله اساسی می‌باشد. یکی ایجاد راه‌کاری برای مواجهه با مسئله از دست‌رفتن تنوع بعد از تغییر محیط و دوم کنترل تعداد دسته‌های فعال درحین اجرای قوانین فازی. روش ارائه‌شده در این مقاله از جهات مختلفی با روش ارائه شده در مرجع [۱۴] متفاوت است. راه‌کار ایجاد چندجمعیتی در روش پیشنهادی به‌صورت تطبیقی بوده و سازوکار متفاوتی نسبت به روش ارائه‌شده در مرجع [۱۴] دارد. راه‌کارهای افزایش تنوع در روش پیشنهادی با روش ارائه شده در مقاله [۱۴] نیز متفاوت می‌باشد.

در سال ۲۰۱۳ نویسندگان مرجع [۲۰] الگوریتمی چندجمعیتی مبتنی بر ازدحام ذرات با عنوان FTMPSO ارائه نموده‌اند. در این‌روش برای بهبود کارایی الگوریتم، یک جستجوی محلی براساس بهره‌برداری تطبیقی ذرات در اطراف بهترین موقعیت یافت‌شده و همچنین یک راه‌کار با عنوان بیداری-خواب[19] استفاده شده‌است. در این راه‌کار ارائه‌شده در این‌روش، چندین دسته ردیاب فعال وجود دارد. پس از تغییر محیط، هر دسته ردیاب فعال به‌سمت قله‌ای که با انجام یک جستجوی محلی در آن قرار داده شده‌است حرکت می‌کند. در واقع در این‌روش مقدار خطای جاری با توجه به نتایج دسته‌ای که مقدار شایستگی بهتری برای بهینه سراسری دارد، تعیین می‌شود.

در این الگوریتم، یک سازوکار جدید براساس یک ذره بهره‌بردار به‌منظور افزایش قابلیت جستجوی محلی در اطراف بهترین موقعیت‌های پیداشده توسط دسته ردیاب فعال به‌کار گرفته شده‌است. از معایب این‌روش نسبت به روش پیشنهادشده در این مقاله این‌است‌که، در این‌روش با افزایش تعداد قله‌ها (افزایش پیچیدگی فضای مسئله) کارآیی این الگوریتم کاهش می‌یابد. این درحالی‌است‌که روش ارائه‌شده در این مقاله با افزایش تعداد قله‌ها با کاهش خطا مواجهه است.

۳- روش پیشنهادی

در این بخش، نمونه جدیدی از الگوریتم ازدحام ذرات ارائه شده‌است. در این‌روش پیشنهادی همانند روش مشابه mQSO از ذرات کوانتوم، ذرات خنثی، عملگر دفع، عملگر ضدهم‌گرایی[20] و شعاع کوانتوم استفاده شده‌است. راهکار ذرات کوانتوم در روش پیشنهادی با روش mQSO متفاوت بوده و هم‌چنین در این‌روش از عملگرهای مختلف دیگری نیز استفاده شده‌است. در ادامه روش پیشنهادی به‌همراه عملگرها و راهکارهای مختلف به‌طور مفصل تشریح شده‌است.

۳-۱- تولید جمعیت اولیه ذرات با استفاده از تئوری آشوب

تئوری آشوب[21] شاخه‌ای از ریاضیات است که به مطالعه رفتار پویای سیستم‌هایی که به انتخاب‌های اولیه بسیار حساس هستند می‌پردازد. تغییرات کوچک در انتخاب‌های اولیه (مثل خطای گردکردن در

محاسبات عددی) باعث واگرایی نتایج سیستم‌های پویا می‌گردد و در حالت‌کلی پیش‌بینی‌های بلندمدت را غیرممکن می‌سازد [۲۱]. از ویژگی‌های تئوری آشوب می‌توان به خودسازماندهی (وفق دادن خود با شرایط محیطی) در محیط‌های پویا و خودمانایی (هر جزئی از سیستم دارای ویژگی کل بوده و مشابه آن‌است) و حساسیت به شرایط اولیه اشاره کرد[۲۱]. این ویژگی‌ها در سیستم آشوب‌گونه منجرشده تا از نظریه آشوب برای تولید جمعیت اولیه ذرات در روش پیشنهادی استفاده شود. در سیستم آشوب‌گونه پیش‌بینی می‌تواند فقط برای مدت‌زمان محدود و با یک خطای پیش‌بینی مجاز صورت پذیرد. در یک سیستم تصادفی، برخلاف سیستم آشوب‌گونه، وضعیت فعلی سیستم مستقل از وضعیت پیشین آن می‌باشد. بنابراین در مدل تصادفی، حتی برای یک مدت زمان کوتاه هم نمی‌توان پیش‌بینی دقیق از خروجی مدل داشت. پدیده‌های تصادفی[۲۲] که تاکنون با عجز و ناتوانی، دلیلی برای آن‌ها یافت نمی‌شد، به‌کمک نظریه آشوب، توجیه می‌شوند. در این‌روش برخلاف الگوریتم ازدحام ذرات استاندارد که جمعیت اولیه به‌صورت تصادفی ایجاد می‌شود، از تئوری آشوب برای ایجاد جمعیت اولیه استفاده شده‌است. یک سیستم آشوب‌گونه پیش‌بینی دقیق‌تری نسبت به یک سیستم تصادفی از آینده دارد. تولید اعداد تصادفی یکنواخت برای شبیه‌سازی پدیده‌های پیچیده همانند بهینه‌سازی در محیط‌های پویا بسیار مهم می‌باشد. یک تولیدکننده اعداد تصادفی وسیله‌ای فیزیکی و یا روشی محاسباتی است که برای تولید دنباله‌ای از اعداد که الگوی خاصی ندارند (یعنی به‌طور تصادفی ظاهر شده‌اند) به‌کار می‌رود. سیستم‌های کامپیوتری به‌طور وسیعی برای تولید اعداد تصادفی مورد استفاده قرار می‌گیرند، درحالی‌که تولیدکننده‌های خوبی نیستند و الگوهای آن‌ها به‌راحتی قابل تشخیص نیست

. آشوب پدیده‌ای است که در سیستم‌های غیرخطی تعریف‌پذیر رخ می دهد که حساسیت زیاد به شرایط اولیه و رفتار شبه‌تصادفی از خود نشان می‌دهند. چنین سیستم‌هایی درحالتی‌که شرایط معادلات نمایی لیاپانوف را برآورده سازند در مد آشوب به‌حالت پایدار باقی خواهند ماند[۲۱]. ویژگی مهمی چون تعریف‌پذیری سیستم در عین رفتار شبه‌تصادفی آن باعث می‌گردد خروجی سیستم تصادفی به‌نظر برسد، درحالی‌که از دید داخلی سیستمی تعریف‌پذیر بوده و لذا قابل بازیابی است. روابط آشوبی که در این‌روش مورد بررسی قرار گرفته در ادامه آورده شده‌اند. در روش پیشنهادی اعداد آشوب‌گونه تولیدشده توسط توابع آشوب را جایگزین اعداد تصادفی کنیم.

$$x_{n+1} = f(\mu, x_n) = \mu x_n (1 - x_n), n = 0,1,2,... \quad (۳)$$

$$f(x) = 1 - 2 \times |(u - \frac{1}{2})| \quad (۴)$$

$$x_{n+1} = \begin{cases} 2x_n, 0 < x_n < \frac{1}{2} \\ 2(1-x_n), \frac{1}{2} < x_n < 1 \end{cases} \quad (۵)$$

در رابطه ۳ که به تابع نگاشت لجستیکی معروف است، x_n یک عدد حقیقی در بازه $[0,1]$ است و پارامتر μ که به ضریب لجستیک معروف است، موجب ایجاد ویژگی‌های منحصربه‌فردی در این تابع می‌شود. در رابطه ۴، u یک عدد حقیقی در بازه $[0,1]$ می‌باشد. در رابطه ۵، x_n یک عدد حقیقی در بازه $[0,1]$ است. در واقع این سه تابع، سه تا از مهم‌ترین توابع در نظریه آشوب محسوب می‌شوند.برای آشوب‌سازی جمعیت ذرات در روش پیشنهادی کافی است که به‌جای مقدار تصادفی *rand* از توابع آشوب‌گونه بالا استفاده شود. بنابراین جمعیت اولیه ذرات براساس تئوری آشوب ایجاد می‌شود.

۳-۲- روش چندجمعیتی

در یک محیط پویا چندین قله وجود دارد که پس از تغییر در محیط هر یک از این قله‌ها ممکن است به بهینه تبدیل شوند. بنابراین الگوریتم باید بتواند تا جای ممکن این قله‌ها را پوشش دهد تا هنگامی که یکی از این قله‌ها به بهینه تبدیل شد تعدادی از ذرات در اطراف آن قرار گیرند. درصورتی‌که تعدادی از این قله‌ها بدون پوشش باشند، ممکن است بعد از تغییر محیط به بهینه تبدیل شوند که این امر باعث پایین‌آمدن دقت و کارایی الگوریتم می‌شود. یکی از راه‌حل‌هایی که برای این مشکل وجود دارد، استفاده از روش چندجمعیتی می‌باشد. هدف چندجمعیتی این است که هر زیرجمعیت یک قله را پوشش داده و آن‌را تحت‌نظر گیرد. باید در نظر داشت که افزایش بیش از حد تعداد ذرات در جمعیت نیز می‌تواند کارآیی را برای الگوریتم پایین آورده و سرعت اجرای الگوریتم را کاهش دهد. در این‌روش پیشنهادی جمعیت به چند زیرجمعیت تقسیم می‌شود. هر زیرجمعیت خود شامل ذرات خنثی و ذرات کوانتوم می‌باشد. در این‌روش ابتدا تعداد ازپیش‌تعیین‌شده‌ای زیرجمعیت وجود دارد. یکی از زیرجمعیت‌ها به عنوان زیرجمعیت پیش‌آهنگ در نظر گرفته می‌شود. این زیرجمعیت پیش‌آهنگ ابتدا در فضای مسئله غیرفعال می‌باشد. روش چندجمعیتی می‌تواند ذرات را به زیرنواحی مختلف اختصاص دهد. چندجمعیتی به‌صورت تطبیقی تعدادی از زیرگروه‌های موردنیاز را تنظیم نموده و به‌صورت خودکار نواحی را برای هر زیرگروه جستجو می‌نماید. درحقیقت در این‌روش برخلاف روش مشابه mQSO که تعداد زیرجمعیت‌ها ثابت می‌باشند (تعداد زیرجمعیت‌ها از ابتدا مشخص شده و ثابت هستند)، زیرجمعیت‌ها طبق یک راه‌کار تطبیقی تولید می‌شوند. ازآن‌جایی‌که تعداد قله‌ها از قبل معلوم نمی‌باشد، تعداد زیرجمعیت‌ها (M) نمی‌تواند به‌راحتی قابل‌تعیین باشد. به‌این‌دلیل یک راهکار تطبیقی برای تعداد زیرجمعیت‌ها ارائه شده‌است. این راهکار کاملاً ساده

می‌باشد. ابتدا جمعیت اولیه به‌صورت آشوب‌گونه ایجاد می‌شود. سپس این جمعیت به تعدادی زیرجمعیت با تعداد مشخصی تقسیم می‌شود. حال اگر تمام زیرجمعیت‌های فعلی به قله موردنظر خود هم‌گرا شده باشند و درصورتی‌که بعضی از قله‌ها هنوز بدون پوشش باقی مانده باشند، دراین‌صورت دسته پیش‌آهنگ در فضای مسئله فعال خواهدشد. اگر فقط یک قله بدون‌پوشش وجود داشته باشد دسته پیش‌آهنگ به این قله هم‌گرا می‌شود و مشکل حل‌شده و نیاز به جستجوی بیشتری نمی‌باشد. اما اگر بیش از یک قله بدون پوشش وجود داشته باشد، دراین‌صورت دسته پیش‌آهنگ باید به‌عنوان یک دسته یابنده عمل نموده و بتواند مکان این قله‌ها را کشف نموده و به دسته‌های ایجادشده جدید اطلاع‌رسانی نماید. باید خاطرنشان کرد که دسته‌های جدید براساس مقدار کارآیی (شایستگی) دسته پیش‌آهنگ درمی‌یابند که این دسته یک قله را پیدا نموده یا خیر. برای مقدار شایستگی یک مقدار آستانه $theshold$ را برای دسته پیش‌آهنگ در نظر گرفته می‌شود. اگر میانگین شایستگی اعضای دسته پیش‌آهنگ بزرگتر یا مساوی این مقدار آستانه گردد بدین‌معنی است که باید یک دسته جدید ایجاد گردد (دسته پیش‌آهنگ یک قله را پیدا نموده‌است). این موضوع به صورت شبه‌کد زیر بیان می‌شود:

$$if\ \ mean\ FEs\ for\ scout\ swarm \geq theshold\ then\ create\ new\ swarm$$

پیدا نمودن یک آستانه مناسب برای مقدار شایستگی می‌تواند به بهبود کارآیی الگوریتم کمک شایانی نماید. راهکار موردنظر برای دسته پیش‌آهنگ بدین‌صورت است که این دسته طبق شعاع دفع [۱۱] دیگر نمی‌تواند در محدوده دسته‌های دیگر قرار گیرد بنابراین این دسته دیگر فضاهای مسئله را که خارج از شعاع دفع دیگر دسته‌ها می‌باشد را ردیابی می‌کند. این موضوع می‌تواند به افزایش سرعت ردیابی دسته پیش‌آهنگ کمک شایانی نماید. درصورتی‌که این دسته یک قله را پیدا کند، اعضای دسته جدید را به‌سمت آن سوق داده و هنگامی‌که این دسته همگرا شدند، دسته پیش‌آهنگ مجدداً به جستجوی خود در خارج از محدوده دیگر دسته‌ها ادامه می‌دهد. زیرجمعیت پیش‌آهنگ زمانی‌که یک قله را ردیابی نمود و زیرجمعیت جدید را به‌سمت آن سوق داد، به‌صورت تصادفی (خارج از شعاع دفع دیگر دسته‌ها) در فضای مسئله به‌صورت تصادفی قرار می‌گیرد. این کار تا زمانی ادامه می یابد که تمام محدوده فضای مسئله مورد کاوش این دسته قرار گرفته‌باشد. این راه‌کار می‌تواند جانشین مناسبی برای عملگر ضدهم‌گرایی باشد که در مرجع [۱۴] آورده شده‌است. در عملگر ضدهم‌گرایی کل فضای مسئله باید توسط بدترین دسته به‌صورت کاملاً تصادفی مجدداً جستجو شود که این موضوع زمان جستجو و کارآیی را تا حدودی تحت تأثیر خود قرار می‌دهد. اما در راهکار تطبیقی پیشنهادی فقط فضاهایی از مسئله جستجو می‌شوند که تابه‌حال مورد جستجو قرار نگرفته‌اند و درنتیجه زمان کم‌تری برای جستجو موردنیاز است. اگر در روش پیشنهادی دو یا بیشتر از زیرجمعیت‌ها بر روی یک قله قرار گیرند یک چالش اساسی رخ می‌دهد که برای حل این چالش از عملگر دفع [۱۴] استفاده می‌شود. ارتباط محلی بین زیرجمعیت‌ها از طریق یک شعاع دفع به‌نام r_{excl} صورت می‌گیرد[۱۴]. هدف اصلی روش پیشنهادی ارائه یک روش کارا و مناسب برای حل مسائل بهینه سازی پویاست به‌طوری‌که بتواند به یک موازنه در جستجو برسد و تنوع برای افزایش کارایی حفظ شود. موقعیت هر ذره در هر زیرجمعیت را با x_{nk} (موقعیت ذره k-ام از زیرجمعیت n-ام) نشان می‌دهیم. سرعت هر ذره از هر زیرجمعیت را با v_{nk} (سرعت ذره k-ام از زیرجمعیت n-ام) نشان‌داده می‌شود. فرض کنید یک ذره خنثی k در زیرجمعیت n قرار دارد، حال این ذره موقعیت و سرعت خود را طبق روابط استاندارد الگوریتم ازدحام ذرات به‌روزرسانی می‌نماید. اگر این ذره بعد از چندین بار به‌روزرسانی موقعیت خود، نتواند بهینه مطلوب (x_{gbest}) را بیابد آن ناحیه را متروکه اعلام نموده و به بخش دیگری از زیرجمعیتی که در آن قرار دارد حرکت تصادفی می‌نماید.

در این‌روش چندجمعیتی به هر بُعد از فضای مسئله یک جمعیت از ذرات اختصاص می‌یابد. هر زیرجمعیت وظیفهٔ بهینه‌سازی یک بُعد را برعهده دارد. به‌ازای هر بُعد از فضای مسئله یک زیرجمعیت از ذرات تک‌بُعدی تخصیص می‌یابد، بنابراین D دسته جداگانه از ذرات یک‌بُعدی وجود دارد که هر زیرجمعیت خود دارای n ذره تک‌بُعدی می‌باشد. در واقع بردار D بُعدی که قرار است بهینه‌سازی شود به D بردار یک‌بُعدی تقسیم می‌شود،که هر قسمت توسط یک دسته بهینه‌سازی خواهدشد. هدف هر زیرجمعیت، بهینه‌سازی مؤلفه‌ای از بردار بهینه‌سازی است که مربوط به آن گروه باشد.

۳-۳- ذرات خنثی و ذرات کوانتوم

در این‌روش پیشنهادی ذرات خنثی وظیفه هم‌گرایی سریع به بهینه موردنظر را برعهده دارند. درحالی‌که ذرات کوانتوم وظیفه ایجاد تنوع را بر عهده دارد. برای ذرات خنثی رابطه سرعت و موقعیت ذرات به‌صورت روابط ۱ و ۲ می‌باشد. در روش پیشنهادی ابتدا ذرات خنثی موجود در هر زیرجمعیت موقعیت خود را به‌روز می‌کنند و سپس موقعیت $gbest$ هر زیرجمعیت با استفاده از ذرات کوانتوم در چندین مرحله بهبود داده‌می‌شود. هر ذره کوانتوم به تعداد Ω بار می‌تواند موقعیت $gbest$ را بهبود دهد. فرض شود موقعیت هر ذره کوانتوم در اطراف $gbest$ از زیرجمعیت i با $x_{i,j}$ نشان داده‌می‌شود. حال موقعیت هر ذره کوانتوم از رابطه ۴ به‌دست می‌آید.

$$x_{i,j} = G(t)_i + (R_j \times r_{cloud}) \qquad (۴)$$

در رابطه ۴ $x_{i,j}$ موقعیت ذره کوانتوم j از زیرجمعیت i می‌باشد. R_j یک عدد تصادفی از توزیع یکنواخت در بازه $[0,1]$ می‌باشد بنابراین موقعیت هر ذره کوانتوم در هر یک از ابعاد فضای جستجو می تواند در شعاع کوانتوم از اطراف موقعیت $gbest$ باشد. بعد از محاسبه موقعیت ذرات کوانتوم مقدار شایستگی برای هر ذره محاسبه می‌شود. حال اگر مقدار شایستگی ذره کوانتومی از مقدار بهینه سراسری $gbest$ بیشتر شود، دراین‌صورت موقعیت ذره کوانتوم به‌عنوان بهینه

سراسری در نظر گرفته می‌شود و در غیراین‌صورت موقعیت تغییری نمی‌کند. رابطه ۴ به اندازه Ω بار تکرار می‌شود تا بهینه سراسری بهبود یابد. شبه‌کد مربوطه به‌صورت شکل ۱ است.

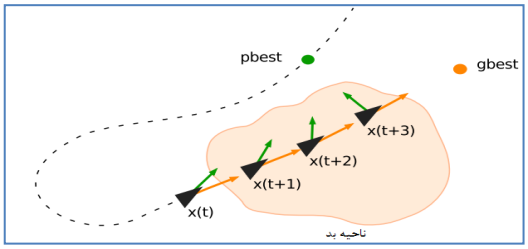
Algorithm2: Improve gbest

```
//update G(t) with Quantum particles
for each swarm i
  For c=1 to Ω
    Update Qi,j based on equation 1
    If f(Q_{i,j}) > f(G(t)_i) THEN
      G(t)_i = Q_{i,j}
  End for
End for
```

شکل ۱: شبه‌کد مربوط به بهبود بهینه سراسری توسط ذرات کوانتوم

۳-۴- عمل‌گر کنترل ذرات خنثی:

در این‌روش پیشنهادی از یک عمل‌گر برای کنترل‌نمودن ذرات خنثی در روش چندگروهی ارائه شده جهت افزایش کارآیی استفاده شده‌است. عمل‌گر کنترل با عنوان کنترل ذرات خنثی[۲۳] نامیده می‌شود. عمل‌گر کنترل ذرات خنثی قادر است بدترین ذرات خنثی در هر زیرجمعیت را که کارآیی چندانی ندارند شناسایی نماید. در این‌روش ذرات همانند الگوریتم ازدحام ذرات استاندارد موقعیت و سرعت خود را در فضای جستجو به‌روزرسانی می‌نمایند. ایده موردنظر برای عمل‌گر کنترل ذرات خنثی بدین‌صورت می‌باشد که هر ذره در هر حرکت خود به‌سمت موقعیت جدید، درصورت عدم یافتن پاسخ بهینه یک فاکتور عدم‌موفقیت به‌نام فاکتور شکست را تولید نماید. در این‌روش از یک حد آستانه برای توالی شکست‌ها استفاده شده‌است. بدین‌صورت که اگر میانگین تعداد شکست‌ها از این حد آستانه بزرگتر شود آن ناحیه از فضای جستجو به‌عنوان ناحیه متروکه شناسایی می‌شود. شکل ۲ یک توالی جابه‌جایی ذرات در فضا که با شکست مواجه شده‌اند را نشان می‌دهد.



شکل ۲: توالی حرکت ذرات با عدم موفقیت (شکست) در فضای جستجو

در شکل ۲ همان‌گونه که مشاهده می‌شود یک ذره از $x(t)$ شروع به حرکت نموده و تا رسیدن به $x(t+3)$ نتوانسته پاسخ بهینه را بیابد و این ناحیه به‌عنوان یک ناحیه بد شناسایی شده و ذرات در این ناحیه ذرات شکست خورده می‌باشند. در این شکل کارآیی ذره‌ای که در موقعیت $x(t)$ قرار دارد همان $f(x(t))$ می‌باشد. در این شکل فرض بر این‌است که کارآیی موقعیت $x(t+1)$ از $x(t)$ بدتر می‌باشد و در تمام این توالی نیز کارآیی از یک موقعیت به موقعیت جدید، بدتر می‌شود. عمل‌گر کنترل ذرات خنثی در این‌روش پیشنهادی یک توالی از شکست‌ها را برای یک ذره نگهداری نموده تا بتواند بدترین ذراتی که در نواحی بدی از فضای جستجو قرار گرفته‌اند را شناسایی نماید. برای افزایش کارآیی عمل‌گر کنترل ذرات خنثی باید رفتارهای ذرات به‌خوبی در محیط شناسایی شده و یک حد آستانه مناسب برای توالی شکست‌ها در نظر گرفته‌شود. در این‌روش درصورتی‌که ذرات در یک ناحیه بدی قرار گیرند این ذرات یا به‌صورت تصادفی مجدد در فضای مسئله قرار می‌گیرند یا این‌که در صورت نبود زمان کافی این گروه متوقف می‌شوند (مطابق شکل ۳).

Algorith3: *Change Rule*

IF *the fitness of swarm is bad*
THEN
relocate the swarm randomly or pause it if there is not enough time

شکل ۳: قانون برای فعال یا غیرفعال بودن ذرات

۳-۵- بهبود جستجوی محلی در هر زیرجمعیت

یکی از بزرگترین چالش‌های الگوریتم‌های بهینه‌سازی، برقراری تعادل میان توانایی جستجوی سراسری و جستجوی محلی است. در PSO وزن اینرسی برای برقراری توازن میان جستجوی سراسری و جستجوی محلی ارائه شده‌است. بنابراین الگوریتم ازدحام ذرات یکی از الگوریتم های مناسب در حل مسائل بهینه‌سازی به‌شمار می‌رود. توانایی الگوریتم پیشنهادی در جستجوی سراسری در بخش گذشته به اثبات رسید. همچنان‌که بیان‌شد در روش پیشنهادی برای بهبود تنوع و افزایش جستجوی سراسری به‌جای یک جمعیت از چند جمعیت استفاده شده‌است. حال برای بهبود دادن نسبی جستجوی محلی از سازوکاری استفاده شده‌است که در ادامه تشریح می‌گردد.

***Algorithm4:* Hill climbing** *algorithm*

1. Procedure hill climbing;
2. Create a solution (s')
3. Best = s';
4. Loop
5. s = best;
6. s' = **neighbors** (**s**)
7. Best = select Best (s');
8. **IF** there is no change in Best solution **THEN**
9. Jump to new state in state space;
10. **Until** stop criterion satisfied;
11. End

شکل ۴:شبه‌کد تپه‌نوردی با جهش

در هر زیرجمعیت برای بهبود جستجوی محلی از روش تپه‌نوردی بهبودیافته استفاده می‌شود. زمانی‌که یک هم‌گرایی ایجاد شود برای حفظ تنوع از یک روش تپه‌نوردی استفاده می‌شود. ایده موردنظر در این‌روش جستجوی محلی بدین‌صورت است که اگر ذرات خنثی به یک قله هم‌گرا شدند درآن‌صورت برای اجتناب از گرفتارشدن در یک بهینه محلی از یک جهش برای ذرات خنثی استفاده می‌شود. جهش در روش

تپه‌نوردی باعث ایجاد تنوع در میان ذرات شده و درنتیجه از گرفتارشدن ذرات به بهینه محلی تا حدودی اجتناب می‌شود. الگوریتم تپه‌نوردی فقط از یک حلقه تشکیل شده‌است که مدام در جهت افزایش مقدار حرکت می‌کند (به سمت بالای تپه). این الگوریتم زمانی خاتمه می‌یابد که به قله‌ای برسد که در آن‌جا هیچ همسایه‌ای مقدار بیشتری نداشته باشد. شبه‌کد الگوریتم تپه‌نوردی به‌صورت شکل ۴ است.

۳-۶- کشف تغییر در محیط

یکی از چالش‌های محیط‌های پویا این است که تغییر محیط غیرقابل‌پیش‌بینی و ازپیش‌تعریف‌نشده است. تغییر در محیط دو چالش اساسی را به‌وجود می‌آورد، که شامل کشف تغییر محیط و ازدست‌رفتن اعتبار شایستگی ذرات می‌باشد. هنگامی‌که محیط تغییر می‌کند، مقدار شایستگی ذرات نیز تغییر می‌کند. بنابراین پس از تغییر محیط، مقادیر شایستگی ذرات نامعتبر خواهدبود و ذرات نمی‌توانند به‌درستی رفتارهای مناسبی را در این محیط پویا از خود نشان دهند. ازآن‌جایی که شرط اجرای تمام رفتارها براساس مقدار شایستگی ذرات است، درنتیجه باید مجدداً پس از تغییر محیط مقدار شایستگی تمام ذرات محاسبه و به‌روز شود تا بتوانند به‌درستی رفتارها را پس از تغییر محیط انجام دهند. در واقع برای آزمایش کردن این‌که تغییر محیط رخ داده یا خیر، می‌توان شایستگی یکی از ذرات را در هر زیرجمعیت به‌صورت تصادفی مجدداً ارزیابی کرد. درصورتی‌که مقدار شایستگی به‌دست‌آمده برابر مقدار شایستگی ثبت‌شده برای ذره موردنظر باشد، یعنی محیط تغییر نکرده است، درغیراین‌صورت محیط تغییر کرده است. ایده موردنظر بدین‌صورت است که، در روش پیشنهادی سازوکارهایی که وجود دارند، مشخص می‌نمایند که هر زیرجمعیت باید صرفاً روی یک قله قرار گیرند. بنابراین اگر در هر زیرجمعیت شایستگی یکی از ذارت به‌صورت تصادفی ارزیابی مجدد شود و در این ارزیابی مجدد، شایستگی حتی یک ذره در یکی از زیرجمعیت‌ها تغییر نموده باشد، به معنی این‌است که محیط تغییر کرده است. شبه‌کد مربوط به کشف تغییر در محیط به‌صورت شکل ۵ آمده‌است. شبه‌کد روش پیشنهادی به‌صورت شکل ۶ آورده شده‌است.

Algorithm5: Detect Change

Evaluate $f(p_{ng})$ for each swarm.
IF new value is different from last iteration THEN
Re-evaluate each particle
$ChangFlag$;
Evaluate a particle from each swarm
if new value is different from last iteration then
$ChangFlag = 1$; Else $ChangFlag = 0$;

شکل ۵: شبه‌کد کشف تغییر در محیط

Algorithm6: Proposed Algorithm

// Initialization
FOR EACH particle *ni*
Initialize v_{ni} , $x_{ni} = p_{ni}$ the particle in the search space;
(initialize particles with chaos theory)
Evaluated $f(p_{ni})$
FOR EACH swarm *n*
$p_{ng} = argmax\{f(p_{ni})\}$
//update G(t) with Quantum particles
REAPET
// local search for each swarm
Procedure hill climbing.
// Operator instead of Anti-Convergence operator
$if\ mean$ Evaluated $f(p_{ni})\ for\ scout\ swarm \geq theshold1$ THEN
create new swarm.
FOR EACH swarm *n*
// Test for Change
Evaluate $f(p_{ng})$
IF new value is different from last iteration THEN
Re-evaluate each particle from each swarm
$ChangFlag$;
Evaluate a particle from each swarm
if new value is different from last iteration then
$ChangFlag = 1$;
Else
$ChangFlag = 0$;
FOR EACH particle *i* of swarm *n*
Apply the **Change Rule**();

FOR EACH particle *i* of swarm *n*
// Update particle
Apply equation (1) and (2) for each particle
// Exclution
FOR EACH swarm $m \neq n$
IF swarm p_{ng} is within r_{excl} of p_{mg} THEN
IF $f(p_{ng}) \leq f(p_{mg})$ THEN
Re-initialize swarm *n*
ELSE
Re-initialize swarm m
FOR EACH particle in re-initialize swarm
Re-evaluate function value.
Update swarm particles with equation (1) and (2).
// Control nutral particles operator
FOR EACH nutral particle *j* of swarm *n*
Insert failure factor *j* of swarm *n*.
Threshold2 considered for consecutive failure.
if (average(failure)>threshold2) then
Region is obsolete;
Evaluate each nutral particle position;
// stop algorithm
UNTIL number of function evaluations performed>max

شکل ۶: شبه‌کد روش پیشنهادی

تفاوت‌های اساسی روش پیشنهادی با روش‌های مشابه به‌خصوص روش مشابه mQSO:

۱- ایجاد جمعیت اولیه در روش پیشنهادی براساس نظریه آشوب صورت می‌گیرد، درحالی‌که در روش مشابه تولید جمعیت اولیه به‌صورت تصادفی است.

۲- نحوه ایجاد چندجمعیتی در روش پیشنهادی با روش مشابه متفاوت است.

۳- در روش پیشنهادی از عملگری متفاوت به‌جای ضدهمگرایی استفاده نموده‌است.

۴- در روش پیشنهادی از عملگر کنترل ذرات خنثی استفاده شده‌است.

۵- در این‌روش برای بهبود جستجوی محلی در هر زیرجمعیت از یک روش تپه‌نوردی با جهش استفاده شده‌است.
۶- در این‌روش از یک راهکار مناسب برای بهبود بهینه سراسری استفاده شده‌است.

۳-۷- پیچیدگی محاسباتی روش پیشنهادی

ازجمله مولفه‌های اصلی در روش پیشنهادی می‌توان، ارزیابی مجدد جمعیت بعد از هر تغییر و به‌روزرسانی ذرات بعد از هر تغییر را نام برد. هر دو مؤلفه پس از هر تغییر محیطی انجام می‌شوند، بنابراین تعداد کل اجراهای روش پیشنهادی، $\frac{Ite}{CF}$ بار می‌باشد که Ite نشان‌دهنده تعداد تکرار و CF نشان‌دهنده فرکانس تغییرات است. با توجه به این موضوع، پیچیدگی محاسباتی روش پیشنهادی از مرتبه‌ای است که در ادامه آورده شده‌است:

$$O\left(\frac{Ite}{CF}\times\left(N\times P\times m+N\times\log N\right)+Ite\times T\right)$$

که N، P، m و T به‌ترتیب اندازه جمعیت ذرات، تعداد چرخه‌های الگوریتم بعد از هر تغییر در محیط، تعداد زیرجمعیت‌ها در الگوریتم و زمان اجرای تابع کارایی است. باتوجه به آن‌که اصل زمان مصرفی در مرتب‌سازی کارایی ذرات صرف می‌شود، مرتبه الگوریتم $O(\frac{Ite}{CF}\times N\times\log N)$ خواهدبود که با حالت ساده آن یک‌سان است. پس به‌طورکلی مرتبه الگوریتم $O\left(N^{2}\times\log N\right)$ می‌باشد.

۴- نتایج تجربی

در این بخش الگوریتم پیشنهادی با الگوریتم‌های استانداردی هم‌چون mQSO10 (5+5q) [۱۴]، FMSO [۱۶]، CellularPSO [۱۷]، Multi-Swarm PSO [۱۱]، AmQSO [۱۵]، FTMPSO [۲۰]، CDEPSO [۲۳]، mQSOE [۱۹]، AMSO [۱۸] و DPSABC [۲۴] برروی محیط‌هایی با فرکانس تغییرات و تعداد قله‌های مختلف مورد مقایسه قرار می‌گیرد. به‌طورکلی هر یک از نتایج (خطای برون‌خطی) حاصل ۵۰ بار آزمایش با هسته‌های[۲۴] تصادفی مختلف هم برای تابع محک و هم برای الگوریتم بوده و هر آزمایش شاهد ۱۰۰ تغییر محیطی است. تغییرات محیطی براساس میزان فرکانس تغییرات ایجاد می‌شود. هم‌چنین خطای برون‌خطی نهایی برابراست با میانگین خطای برون‌خطی به‌دست‌آمده در هر بار آزمایش. برانک در [۲۲] یک تابع محک پویا برای آزمایش الگوریتم‌های بهینه‌سازی در محیط‌های پویا پیشنهاد داده است که تابع محک متحرک[۲۵] نامیده می‌شود. محک قله‌های متحرک رفتاری شبیه به مسائل پویا در دنیای واقعی را شبیه‌سازی می‌کند و به‌عنوان معیاری برای سنجش الگوریتم‌های بهینه‌سازی در محیط‌های پویا مورداستفاده قرار می‌گیرد. این تابع شامل یک محیط چندبعدی است که دارای چندین قله می‌باشد. هنگامی‌که یک تغییر در محیط رخ دهد، ارتفاع، عرض و محل قله‌ها تغییر می‌نماید. از این تابع پویا برای تست کارایی الگوریتم پیشنهادی استفاده شده است. در این تابع این قابلیت وجود دارد که ابعاد محیط، تعداد قله‌ها، شکل قله‌ها، میزان شدت تغییرات و فرکانس تغییرات تنظیم شود. تنظیمات پیش‌فرض برای محک قله‌های متحرک [۲۲] در جدول ۱ آمده‌است.

جدول ۱: تنظیمات استاندارد برای تابع قله‌های متحرک[۲۲]

پارامتر	مقدار
تعداد قله‌ها (M)	۱۰
فرکانس تغییرات (f)	در هر ۵۰۰۰
شدت تغییرات ارتفاع قله‌ها	۷
شدت تغییرات عرض قله‌ها	۱
شکل قله‌ها	مخروطی
میزان حرکت مکان قله‌ها	۱
تعداد ابعاد (D)	۵
محدوده ارتفاع قله‌ها	[۷۰،۳۰]
محدوده عرض قله‌ها	[۱۲،۱]
ارتفاع استاندارد قله‌ها	۵۰
محدوده فضای جستجو	[۱۰۰،۰]
شدت تغییرات (S)	۱

۴-۱- تنظیمات اولیه برای آزمایشات

برای الگوریتم پیشنهادی ضرایب مولفه‌های اجتماعی و شناختی c_1 و c_2 در سرعت ذره به ترتیب برابر ۲/۸ و ۱/۳ و وزن اینرسی w میانگین c_1 و c_2 فرض شده‌است (۲/۰۵). تعداد کل ذرات برابر با ۱۰۰ در نظر گرفته شده‌است. برای روش پیشنهادی تعداد ۱۰ زیرجمعیت اولیه ایجادشده و برای هر زیرجمعیت ۵ ذره کوانتوم و ۵ ذره خنثی ایجاد می‌شود. برای این الگوریتم شعاع کوانتوم برابر ۰/۵ و شعاع دفع و شعاع هم‌گرایی برابر ۳۱/۵ تعیین شده‌است. حد آستانه برای توالی شکست‌ها برابر ۱۰ در نظر گرفته شده‌است.
در این‌جا برای mQSO از پیکربندی 10(5+5q) استفاده شده‌است که در آن ۱۰ گروه ایجاد می‌شود و در آن هر یک از گروه‌ها دارای ۵ ذره خنثی و دارای ۵ ذره کوانتوم می‌باشند. هم‌چنین برای این الگوریتم شعاع کوانتوم برابر ۰/۵ و شعاع دفع و شعاع هم‌گرایی برابر ۳۱/۵ تعیین شده‌است. برای الگوریتم FMSO ماکزیمم تعداد گروه‌های فرزند برابر ۱۰، شعاع دفع مابین گروه‌های فرزند برابر ۲۵، تعداد ذرات در گروه والد و گروه‌های فرزند به ترتیب برابر ۱۰۰ و ۱۰ در نظر گرفته شده است. برای CellularPSO یک آتاماتای سلولی ۵ بعدی با ۱۰۵سلول و همسایگی مور ۲۶ با شعاع ۲ سلول در فضای جستجو به کار رفته است. بیشینه سرعت ذرات برابر شعاع همسایگی و ماکزیمم تعداد ذرات برای هر سلول برابر ۱۰و شعاع جستجوی محلی برابر ۰/۵ تعیین شده است. هم‌چنین جستجوی محلی را همه ذرات پس از مشاهده تغییر در محیط برای یک مرحله اجرا می‌کنند. برای روش Multi-Swarm، تعداد ذرات برای گروه والد و تعداد ذرات برای گروه‌های فرزندان به‌ترتیب برابر ۵ و ۱۰ تعیین شده‌است. شعاع دفع

مابین گروه‌های فرزندان و شعاع ذرات کوانتوم به‌ترتیب برابر ۳۰ و ۰/۵ تعیین شده‌است. جدول ۲ تنظیمات پیش‌فرض را برای روش پیشنهادی نشان می‌دهد.

جدول ۲: تنظیمات پیش‌فرض برای روش پیشنهادی

پارامتر	مقدار
c_1	۲/۸
c_2	۱/۳
W	میانگین c_2 و c_1
تعداد کل ذرات	۱۰۰
تعداد اولیه زیرجمعیت‌ها	۱۰
تعداد ذرات کوانتوم در هر زیرجمعیت	۵
تعداد ذرات خنثی در هر زیرجمعیت	۵
آستانه شکست	۱۰
شعاع کوانتوم	۰/۵
شعاع دفع	۳۱/۵
شعاع هم‌گرایی	۳۱/۵
آستانه شایستگی	۶۰
آستانه شکست	۱۰
ضریب آشوب	۴

در این بخش به‌شرح آزمایش‌های انجام‌شده بر روی مدل پیشنهادی در فرکانس‌های ۵۰۰ تا ۱۰۰۰۰ و با تعداد قله‌های ۱ الی ۲۰۰ پرداخته می‌شود. تنظیمات پیش‌فرض این بخش برای تابع قله‌های متحرک در جدول ۳ آمده‌است. نتایج آزمایش‌ها برای همه الگوریتم‌ها، میانگین خطای برون‌خطی با فاصله اطمینان ۹۵ درصد در ۱۰۰ مرتبه اجرا می‌باشد. خطای برون‌خطی و خطای استاندارد به‌دست‌آمده از آزمایش‌ها برای محیط‌های با پویایی‌های متفاوت در جداول ۳ تا ۶ نشان داده شده‌است. نتایج بهتر به‌صورت پررنگ می‌باشد. همان‌طورکه مشاهده می‌شود، اختلاف خطای برون‌خطی الگوریتم پیشنهادی نسبت به الگوریتم‌های دیگر با افزایش فرکانس محیط و هم‌چنین با افزایش پیچیدگی فضا (افزایش تعداد قله‌ها) افزایش پیدا می‌کند. علت امر این است که الگوریتم پیشنهادی سریع‌تر می‌تواند راه‌حل‌های بهتر را پس از مشاهده تغییر در محیط به‌دست بیاورد. در روش پیشنهادی به‌دلیل وجود تنوع زیاد، تقریباً اکثر قله‌ها مورد پوشش ذرات قرار می‌گیرند. همین موضوع باعث می‌شود که در روش پیشنهادی برخلاف اکثر روش‌های مورد مقایسه، با افزایش تعداد قله‌ها، خطای برون‌خطی برای روش پیشنهادی کاهش یابد. نتایج به‌دست‌آمده در جداول ۳ تا ۶ نشان داده شده‌اند. با افزایش تعداد قله‌ها در فرکانس‌های ۵۰۰، ۵۰۰۰، ۱۰۰۰۰ و ۱۰۰۰۰ کارایی روش پیشنهادی افزایش پیدا می‌کند. در فرکانس‌های تغییر پایین بنا به این که، بعد از تعداد ارزیابی‌های کم توسط الگوریتم، محیط دچار تغییر می شود، بنابراین الگوریتم زمان کمتری برای یافتن بهینه دارد. نتایج جداول در همه روش‌ها بیان‌گر همین موضوع هستند که در فرکانس های پایین، خطای روش‌ها بیشتر است. در جداول ۳ تا ۶ همان‌گونه که مشاهده می‌شود، ابتدا در تعداد قله‌های کم‌تر، برخی روش‌ها، ازجمله روش FTMPSO از روش پیشنهادی بهتر شده‌است ولی با افزایش تعداد قله‌ها، کارآیی روش پیشنهادی بهتر گردیده‌است.

جدول ۳: مقایسه خطای برون‌خطی و خطای استاندارد روش پیشنهادی با سایر روش‌ها برای f=500

تعداد قله‌ها	mQSO10 (5+5q)	FMSO	Cellular PSO	Multi Swarm PSO	FTMPSO	DPSABC	AmQSO	Proposed
۱	۳۳/۶۷+۳/۴	۷/۵۸±۰/۹	۱۳/۴±۰/۷۴	۵/۴۶±۰/۳۰	**۱/۷۶±۰/۰۹**	۲/۷۷±۰/۰۰	۳/۰۲±۰/۳۲	۴/۰۱±۰/۲۱
۵	۱۱/۹۱±۰/۷	۹/۴۵±۰/۴	۹/۶۳±۰/۴۹	۵/۴۸±۰/۱۹	**۲/۹۳±۰/۱۸**	–	۵/۷۷±۰/۵۶	۳/۷۸±۰/۲۴
۱۰	۹/۶۲±۰/۳۴	۱۸/۲۶±۰/۳	۹/۴۲±۰/۲۱	۵/۹۵±۰/۰۹	۳/۹۱±۰/۱۹	**۳/۴۲±۰/۰۰**	۵/۳۷±۰/۴۲	۳/۵۴±۰/۱۲
۲۰	۹/۰۷±۰/۲۵	۱۷/۳۴±۰/۳	۸/۸۴±۰/۲۸	۶/۴۵±۰/۱۶	۴/۸۳±۰/۱۹	**۳/۱۲±۰/۰۰**	۶/۸۲±۰/۳۴	۳/۲۷±۰/۱۷
۳۰	۸/۸۰±۰/۲۱	۱۶/۳۹±۰/۴	۸/۸۱±۰/۲۴	۶/۶۰±۰/۱۴	۵/۰۵±۰/۲۱	۳/۶۹±۰/۰۰	۷/۱۰±۰/۳۹	**۳/۱۹±۰/۱۱**
۴۰	۸/۵۵±۰/۲۱	۱۵/۳۴±۰/۴	۸/۹۴±۰/۲۴	۶/۸۵±۰/۱۳	–	–	۷/۵۷±۰/۳۲	**۳/۱۴±۰/۱۴**
۵۰	۸/۷۲±۰/۲۰	۵/۵۴±۰/۲	۸/۶۲±۰/۲۳	۷/۰۴±۰/۱۰	۴/۹۸±۰/۱۵	۳/۲۲±۰/۰۰	۷/۵۵±۰/۳۲	**۳/۱۰±۶/۱۶**
۱۰۰	۸/۵۴±۰/۱۶	۲/۸۷±۰/۶	۸/۵۴±۰/۲۱	۷/۳۹±۰/۱۳	۵/۳۱±۰/۱۱	**۳/۰۱±۰/۰۰**	۷/۳۴±۰/۳۱	۳/۰۶±۰/۱۲
۲۰۰	۸/۱۹±۰/۱۷	۱۱/۵۲±۰/۶	۸/۲۸±۰/۱۸	۷/۵۲±۰/۱۲	۵/۵۲±۰/۲۱	۳/۱۶±۰/۰۰	۷/۴۸±۰/۱۹	**۳/۰۱±۰/۱۱**

جدول ۴: مقایسه خطای برون‌خطی و خطای استاندارد روش پیشنهادی با سایر روش‌ها برای f=1000

تعداد قله‌ها	mQSO10 (5+5q)	FMSO	Cellular PSO	Multi Swarm PSO	FTMPSO	DPSABC	AmQSO	Proposed
۱	۱۸/۶±۱/۶	۱۴/۴۲±۰/۴	۶/۷۷±۰/۳۸	۲/۹۰±۰/۱۸	**۰/۸۹±۰/۰۵**	۱/۶۸±۰/۰۰	۲/۳۳±۰/۳۱	۲/۷۲±۰/۲۲
۵	۶/۵۶±۰/۳۸	۱۰/۵۹±۰/۲	۵/۳۰±۰/۳۲	۳/۳۵±۰/۱۸	**۱/۷۰±۰/۱۰**	–	۲/۹۰±۰/۳۲	۲/۲۶±۰/۲۰
۱۰	۵/۷۱±۰/۲۲	۱۰/۴۰±۰/۱	۵/۱۵±۰/۱۳	۳/۹۴±۰/۰۸	۲/۳۶±۰/۰۹	۳/۲۳±۰/۰۰	۴/۵۶±۰/۴۰	**۲/۱۸±۰/۱۶**
۲۰	۵/۸۵±۰/۱۵	۱۰/۳۳±۰/۱	۵/۲۳±۰/۱۸	۴/۳۳±۰/۱۲	۳/۰۱±۰/۱۲	۳/۴۰±۰/۰۰	۵/۳۶±۰/۴۷	**۱/۲۶±۰/۱۶**
۳۰	۵/۸۱±۰/۱۵	۱۰/۰۶±۰/۱	۵/۳۳±۰/۱۶	۴/۴۱±۰/۱۱	۳/۰۶±۰/۱۰	۳/۲۸±۰/۰۰	۵/۲۰±۰/۳۸	**۱/۳۰±۰/۱۶**
۴۰	۵/۷۰±۰/۱۴	۹/۸۵±۰/۱۱	۵/۶۱±۰/۱۶	۴/۵۲±۰/۰۹	-	-	-	**۱/۲۵±۰/۱۱**
۵۰	۵/۸۷±۰/۱۳	۹/۵۴±۰/۱۱	۵/۵۵±۰/۱۴	۴/۵۷±۰/۰۸	۳/۲۹±۰/۱۰	۲/۶۷±۰/۰۰	۶/۰۶±۰/۱۴	**۱/۱۹±۰/۱۵**
۱۰۰	۵/۸۳±۰/۱۳	۸/۷۷±۰/۰۹	۵/۵۷±۰/۱۲	۴/۷۷±۰/۰۸	۳/۶۳±۰/۰۹	۳/۰۸±۰/۰۰	۴/۷۷±۰/۴۵	**۱/۱۱±۰/۰۹**
۲۰۰	۵/۵۴±۰/۱۱	۸/۰۶±۰/۰۷	۵/۵۰±۰/۱۲	۴/۷۶±۰/۰۷	۳/۷۴±۰/۰۹	۳/۰۱±۰/۰۰	۵/۷۵±۰/۲۶	**۱/۰۳±۰/۱۰**

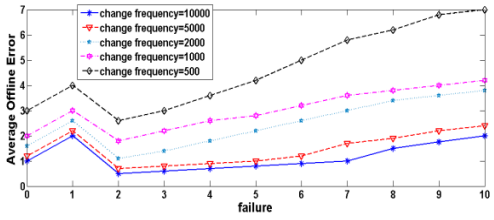
جدول ۵: مقایسه خطای برون‌خطی و خطای استاندارد روش پیشنهادی با سایر روش‌ها برای f=5000

تعداد قله‌ها	mQSO10 (5+5q)	FMSO	Cellular PSO	Multi Swarm PSO	FTMPSO	DPSABC	AmQSO	CDEPSO	Proposed
۱	۳/۸۲±۰/۳۵	۳/۴۴±۰/۱۱	۲/۵۵±۰/۱۲	۰/۵۶±۰/۰۴	**۰/۱۸±۰/۰۱**	۲/۲۵±۰/۰۰	۲/۶۲±۰/۱۰	۰/۴۱±۰/۰۰	۱/۰۰±۰/۱۹
۵	۱/۹۰±۰/۰۸	۲/۹۴±۰/۰۷	۱/۶۸±۰/۱۱	۱/۰۶±۰/۰۶	**۰/۴۷±۰/۰۵**	–	۱/۰۱±۰/۰۹	۰/۹۷±۰/۰۱	۰/۸۹±۰/۱۱
۱۰	۱/۹۱±۰/۰۸	۳/۱۱±۰/۰۶	۱/۷۸±۰/۰۵	۱/۵۱±۰/۰۴	**۰/۶۷±۰/۰۴**	۲/۱۳±۰/۰۰	۱/۵۱±۰/۱۰	۱/۲۲±۰/۰۱	۰/۷۴±۰/۱۰
۲۰	۲/۵۶±۰/۱۰	۳/۳۶±۰/۰۶	۲/۶۱±۰/۰۷	۱/۸۹±۰/۰۴	۰/۹۳±۰/۰۴	۲/۰۷±۰/۰۰	۲/۰۰±۰/۱۵	۱/۵۴±۰/۰۱	**۰/۶۵±۰/۱۱**
۳۰	۲/۶۸±۰/۱۰	۳/۲۸±۰/۰۵	۲/۹۳±۰/۰۸	۲/۰۳±۰/۰۶	۱/۱۴±۰/۰۴	۱/۸۸±۰/۰۰	۲/۱۹±۰/۱۷	۲/۶۲±۰/۰۱	**۰/۵۷±۰/۰۹**
۴۰	۲/۶۵±۰/۰۸	۳/۱۴±۰/۰۸	۳/۲۶±۰/۰۴	۲/۰۴±۰/۰۶	-	-	-	-	**۰/۵۵±۰/۰۸**
۵۰	۲/۶۳±۰/۰۸	۳/۲۲±۰/۰۵	۳/۲۶±۰/۰۸	۲/۰۸±۰/۰۲	۱/۳۲±۰/۰۴	۱/۹۱±۰/۰۰	۲/۴۳±۰/۱۳	۲/۲۰±۰/۰۱	**۰/۴۹±۰/۰۸**
۱۰۰	۲/۵۲±۰/۰۶	۳/۰۶±۰/۰۴	۳/۴۱±۰/۰۷	۲/۱۴±۰/۰۲	۱/۶۱±۰/۰۳	۱/۸۹±۰/۰۰	۲/۶۸±۰/۱۲	۱/۵۴±۰/۰۱	**۰/۳۷±۰/۰۴**
۲۰۰	۲/۳۶±۰/۰۵	۲/۸۴±۰/۰۳	۳/۴۰±۰/۰۶	۲/۱۱±۰/۰۳	۱/۶۷±۰/۰۳	۱/۸۷±۰/۰۰	۲/۶۲±۰/۱۰	۲/۱۱±۰/۰۱	**۰/۳۵±۰/۰۳**

جدول ۶: مقایسه خطای برون‌خطی و خطای استاندارد روش پیشنهادی با سایر روش‌ها برای f=10000

تعداد قله‌ها	mQSO10(5+5q)	FMSO	CellularPSO	MultiSwarmPSO	FTMPSO	DPSABC	AmQSO	Proposed
۱	۱/۹۰±۰/۱۸	۱/۹۰±۰/۰۶	۱/۵۳±۰/۱۲	۰/۲۷±۰/۰۲	**۰/۰۹±۰/۰۰**	۲/۶۷±۰/۰۰	۰/۱۹±۰/۰۲	۰/۱۵±۰/۱۰
۵	۱/۰۳±۰/۰۶	۱/۷۵±۰/۰۶	۰/۹۲±۰/۱۰	۰/۷۰±۰/۱۰	۰/۳۱±۰/۰۴	–	۰/۴۵±۰/۰۴	**۰/۱۳±۰/۰۸**
۱۰	۱/۱۰±۰/۰۷	۱/۹۱±۰/۰۴	۱/۱۹±۰/۰۷	۰/۹۷±۰/۰۴	۰/۴۳±۰/۰۳	۹/۰۱±۰/۰۱	۰/۷۶±۰/۰۶	**۰/۱۰±۰/۰۶**
۲۰	۱/۸۴±۰/۰۸	۲/۱۶±۰/۰۴	۲/۲۰±۰/۱۰	۱/۳۴±۰/۰۸	۰/۵۶±۰/۰۱	۶/۶۰±۰/۰۱	۱/۲۸±۰/۱۲	**۰/۰۸±۰/۰۷**
۳۰	۲/۰۰±۰/۰۹	۲/۱۸±۰/۰۴	۲/۶۰±۰/۱۳	۱/۴۳±۰/۰۵	۰/۶۹±۰/۰۹	۷/۷۰±۰/۰۱	۱/۷۸±۰/۰۹	**۰/۰۶±۰/۰۵**
۴۰	۱/۹۹±۰/۰۷	۲/۲۱±۰/۰۳	۲/۷۳±۰/۱۱	۱/۴۷±۰/۰۶	-	-	-	**۰/۰۴±۰/۰۴**
۵۰	۱/۹۹±۰/۰۷	۲/۶۰±۰/۰۸	۲/۸۴±۰/۱۲	۱/۴۷±۰/۰۴	۰/۸۶±۰/۰۲	۸/۱۰±۰/۰۱	۱/۵۵±۰/۰۸	**۰/۰۸±۰/۰۳**
۱۰۰	۱/۸۵±۰/۰۵	۲/۲۰±۰/۰۳	۲/۹۳±۰/۰۹	۱/۵۰±۰/۰۳	۱/۰۸±۰/۰۳	۸/۳۴±۰/۰۱	۱/۸۹±۰/۱۴	**۰/۰۹±۰/۰۳**

نمودار ترسیم‌شده در شکل ۷ متوسط خطای برون‌خطی را برای روش پیشنهادی در فرکانس‌های تغییر مختلف و با تعداد شکست‌های مختلف برای ذرات را نشان می‌دهد.



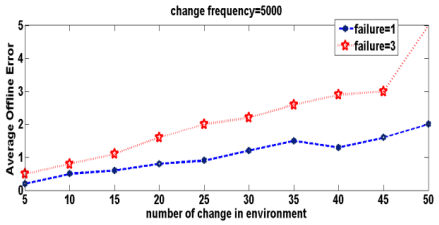
شکل ۷: متوسط خطای برون‌خطی روش پیشنهادی با تعداد شکست های مختلف و فرکانس‌های تغییر متفاوت

در شکل ۷ واضح است که هر چه تعداد شکست‌ها برای ذرات بیشتر می‌شود، کارآیی برای روش پیشنهادی نیز کاهش می‌یابد.

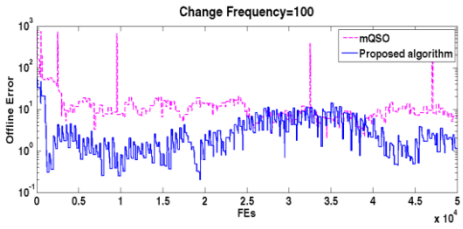
نمودار ترسیم‌شده در شکل ۸ روش پیشنهادی را با تعداد تغییرات مختلف در محیط و همچنین تعداد ۱ و ۳ شکست برای ذرات نشان می‌دهد. واضح است که هر چه تعداد تغییرات در محیط و همچنین تعداد شکست‌ها بیشتر می‌شود، کارآیی برای روش پیشنهادی نیز پایین می‌آید.

شکل ۹، ۱۰ و ۱۱ روش پیشنهادی را در فرکانس‌های تغییر مختلف با روش مشابه mQSO مقایسه می‌نماید. هر چه فرکانس تغییر در محیط پایین‌تر باشد، بدین معنی است که محیط در زمان بسیار

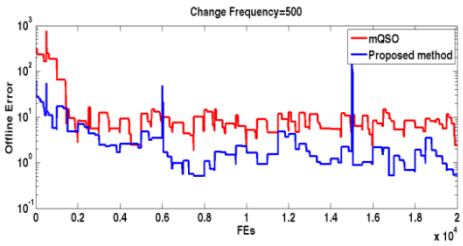
اندکی دچار تغییر می‌شود و این کار الگوریتم بهینه‌سازی را برای ردیابی بهینه در حال تغییر بسیار مشکل می‌نماید.



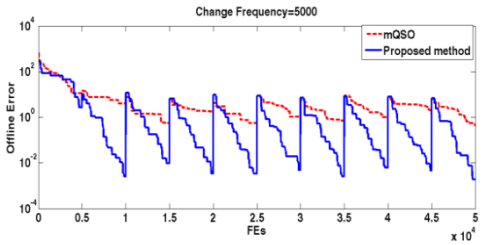
شکل ۸: متوسط خطای برون‌خطی روش پیشنهادی با تعداد تغییرات مختلف و با تعداد ۱ و ۳ شکست در فرکانس تغییرات ۵۰۰۰ و با تعداد ۱۰ قله



شکل ۹: روند تغییرات خطای آفلاین در فرکانس ۵۰۰



شکل ۱۰: روند تغییرات خطای برون‌خطی در فرکانس ۱۰۰۰



شکل ۱۱: روند تغییرات خطای برون‌خطی در فرکانس ۵۰۰۰

شدت تغییرات در قله‌ها یکی از مهم‌ترین پارامترهای محک قله‌های متحرک به‌شمار می‌رود. منطقی به‌نظر می‌رسد که، با افزایش شدت تغییرات در ارتفاع و پهنای قله‌ها، کارآیی نیز برای الگوریتم کاهش یابد. هر چه شدت تغییرات در قله‌ها بیشتر شود، به‌همان اندازه ذرات در یافتن بهینه دچار مشکل می‌شوند. در این بخش تأثیر پارامتر شدت تغییرات بر روی روش پیشنهادی مورد آزمایش قرار گرفته‌است. جدول ۷ بیان‌گر این موضوع می‌باشد. پیچیدگی محاسباتی یک الگوریتم می تواند یک معیار مناسب برای مقایسه الگوریتم با سایر الگوریتم‌های مشابه باشد.

جدول ۷: مقدار خطای برون‌خطی روش پیشنهادی با شدت تغییرات مختلف

شدت تغییرات	روش پیشنهادی
۱	۰/۷۴±۰/۱۰
۲	۱/۰۹±۰/۱۹
۳	۱/۴۵±۰/۲۲
۴	۲/۱۰±۰/۳۵
۵	۲/۹۲±۰/۴۵

در این قسمت در جدول ۸ روش پیشنهادی از نظر پیچیدگی محاسباتی با روش‌های مشابه mQSO، CPSOR [۲۵]، CPSO [۲۵] و روش محمدپور و پروین [۲۶] مورد مقایسه قرار گرفته‌است.

جدول ۸: مقایسه روش پیشنهادی با دیگر روش‌ها از نظر پیچیدگی محاسباتی

الگوریتم	پیچیدگی محاسباتی
روش پیشنهادی	$O\left(M^{2}\log M\right)$
CPSO	$O\left(M^{3}\right)$
CPSOR	$O\left(M^{3}\right)$
mQSO	$O\left(M^{2}\right)$
محمدپور و پروین	$O\left(M\log M\right)$

مشاهده می‌شود روش پیشنهادی از نظر پیچیدگی محاسباتی در میان روش‌های مشابه رتبه سوم را داراست. در روش مشابه mQSO مرتبه زمانی نسبت به روش پیشنهادی کمتر می‌باشد. ولی پیچیدگی محاسباتی روش‌های مشابه CPSO و CPSOR که روش‌های نسبتاً جدیدی نیز هستند، نسبت به روش پیشنهادی بیشتر گردیده‌است.

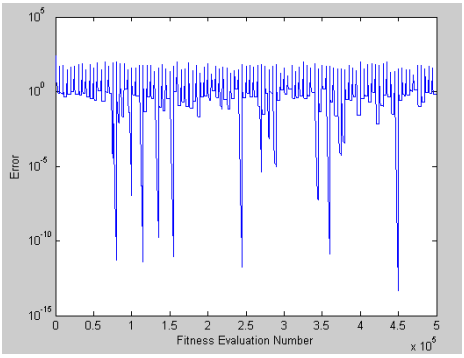
در این‌جا، کارآیی الگوریتم پیشنهادی بر روی MPB مورد بررسی قرار می‌گیرد. با این تفاوت که این بار آزمایش‌ها بر روی فرکانس تغییرات ۵۰۰۰، تعداد ۱۰ قله، طول گام حرکتی ۱ و با ابعاد مختلف ۲، ۳، ۴ و ۵ می‌باشد. درنهایت نتایج با چند الگوریتم دیگر ازجمله الگوریتم‌های استاندارد mQSO و AmQSO مقایسه گردیده‌است. همانند گذشته معیار سنجش، مجدداً متوسط خطای برون‌خطی در ۵۰ بار آزمایش می‌باشد.

در جدول ۹ ملاحظه می‌شود که کارآیی الگوریتم پیشنهادی در تمامی موارد بهتر از سایر الگوریتم‌ها است. نکته قابل توجه این است که کارآیی الگوریتم‌ها با افزایش ابعاد مسئله کاهش می‌یابد که این امر یک مسئله کاملاً عادی در میان مسائل بهینه‌سازی به‌شمار می‌آید. درواقع با افزایش تعداد پارامترهای بهینه‌سازی در تابع شایستگی، جستجو برای یافتن موقعیت‌های بهتر در فضای مسئله برای الگوریتم‌ها سخت‌تر می‌شود.

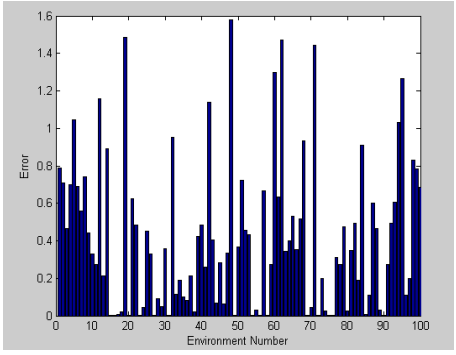
جدول ۹: مقایسه خطای برون‌خطی (خطای استاندارد) الگوریتم‌ها بر روی MPB با f=5000، M=10، S=1 و ابعاد مختلف

Algorithm	ابعاد			
	۲	۳	۴	۵
mQSO [23]	۱/۰۱±۰/۰۴	۱/۴۹±۰/۰۹	۱/۴۷±۰/۰۸	۱/۸۵±۰/۰۸
AmQSO [26]	۰/۷۱±۰/۰۵	۱/۱۶±۰/۱۰	۱/۳۳±۰/۰۸	۱/۵۱±۰/۱۰
MultiSwarmPSO [11]	۱/۲۴±۰/۰۷	۱/۴۲±۰/۱۰	۱/۳۵±۰/۰۹	۱/۶۱±۰/۱۲
Proposed	**/۷۴±۰/۱۰**	**۱/۱۰±۰/۱۱**	**۱/۲۸±۰/۱۳**	**۱/۴۸±۰/۱۰**

در ادامه نمودار خطای برون‌خطی و خطای جاری برای الگوریتم پیشنهادی با ابعاد ۵ در شکل‌های ۱۲ و ۱۳ آورده شده‌است. شکل ۱۲، نمودار هم‌گرایی الگوریتم پیشنهادی را براساس خطای جاری و در ۱۰۰ تغییر محیطی با فرکانس تغییرات ۵۰۰۰، تعداد ۱۰ قله، طول گام حرکتی ۱ و تعداد ابعاد ۵ به نمایش می‌گذارد. در کنار آن شکل شده‌است. درنهایت میانگین کل همه خطاهای برون‌خطی موجود در شکل با متغیر Mean در بالای شکل قابل مشاهده است. که مقدار عددی این متغیر برابر با همان مقداری است که در جدول ۵ درج گردیده‌است. همچنین انحراف معیار این ۵۰ بار آزمایش نیز در بالای نمودار دیده می‌شود.



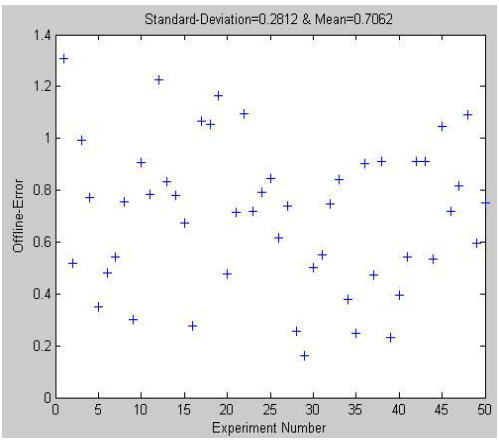
شکل ۱۲: گراف خطای برون خطی الگوریتم با تعداد ابعاد ۵ بر اساس تعداد ارزیابی‌ها



شکل ۱۳: نمودار میله‌ای خطای برون خطی الگوریتم با تعداد ابعاد ۵ براساس تعداد ارزیابی‌ها

۱۳ قرار دارد که نمودار هم‌گرایی همین الگوریتم را براساس خطای جاری و تعداد ارزیابی‌های صورت‌گرفته که ۵۰۰۰۰۰ (۵۰۰۰ × ۱۰۰) می‌باشد، بر روی ۱۰ قله، طول گام حرکتی ۱ و تعداد ابعاد ۵ نشان می دهد.

نمودار پایداری الگوریتم پیشنهادی طبق سناریوی دوم تابع محک قله‌های متحرک (جدول ۱) به نمایش گذاشته می‌شود. همان‌طورکه در شکل ۱۴ نیز دیده می‌شود. این نمودار براساس خطای برون‌خطی و در ۵۰ بار آزمایش با هسته‌های تصادفی هم از نظر تابع محک و هم از نظر خود الگوریتم ترسیم شده‌است. در هر بار اجرا ۱۰۰ تغییر محیطی اتفاق می‌افتد که طبق سناریوی دوم هر تغییر محیطی دراین‌جا برابر با ۵۰۰۰ ارزیابی می‌باشد. همچنین میانگین خطای برون‌خطی در هر ۱۰۰ تغییر محیطی با علامت "+" در شکل نشان داده

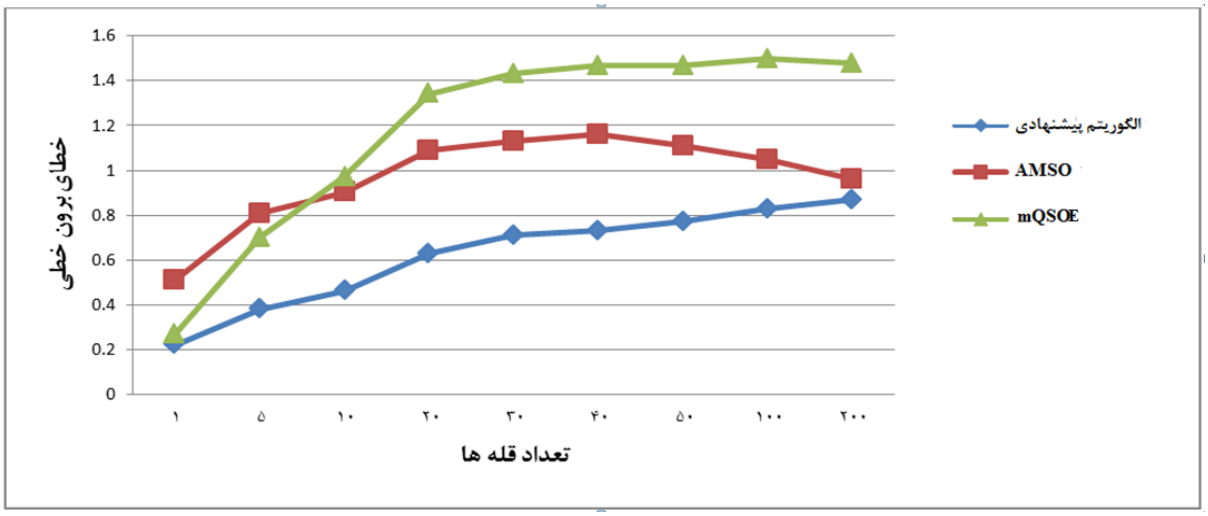


شکل ۱۴: نمودار پایداری الگوریتم پیشنهادی در سناریوی دوم تابع محک قله‌های متحرک

باید دقت شود در تمام نتایج به‌دست‌آمده در بالا از تابع نگاشت لجستیکی برای اعمال تئوری آشوب در روش پیشنهادی استفاده شده‌است. حال در این بخش می‌خواهیم سه تابع آشوب را که در روابط ۳، ۴ و ۵ بیان شده‌است، در روش پیشنهادی از نظر میزان کارآیی مورد بررسی و مقایسه قرار دهیم. در جدول ۱۰ سه تابع آشوب‌گونه (روابط ۳، ۴ و ۵) در روش پیشنهادی اعمال گردیده و از نظر خطای برون‌خطی مورد مقایسه قرار گرفته‌اند.

جدول ۱۰: مقایسه خطای برون‌خطی (خطای استاندارد) الگوریتم‌ها بر روی MPB با f=5000، M=10، S=1 و تعداد ابعاد مختلف

Chaotic function	تعداد ابعاد			
	۲	۳	۴	۵
Function 3	**/۷۴±۰/۱۰**	**۱/۱۰±۰/۱۱**	**۱/۲۸±۰/۱۳**	۱/۴۸±۰/۱۰
Function 4	۰/۷۸±۰/۱۲	۱/۱۲±۰/۱۰	۱/۳۰±۰/۱۰	**۱/۴۲±۰/۱۰**
Function 5	۰/۸۱±۰/۱۰	۱/۱۵±۰/۰۹	**۱/۲۸±۰/۱۲**	۱/۴۷±۰/۰۸



شکل ۱۵: مقایسه الگوریتم پیشنهادی با سایر الگوریتم‌ها در فرکانس ۵۰۰۰ و به‌ازای تعداد قله‌های متفاوت

همان‌گونه‌که از جدول ۱۰ مشاهده می‌شود برای تابع نگاشت لجستکی که در رابطه ۳ بیان شده‌است نتایج بهتری نسبت به دو تابع دیگر حاصل شده‌است.

در این بخش روش پیشنهادی با دو روش مشابه AMSO و mQSOE که در بخش کارهای گذشته تشریح شده‌اند، از نظر خطای برون‌خطی مورد مقایسه قرار گرفته‌است. مقادیر به‌دست‌آمده برای دو روش مورد مقایسه، از مراجع آن‌ها به دست آمده است. شکل ۱۵ مقایسه روش پیشنهادی با دو روش دیگر را نشان می‌دهد. در این شکل محور افقی تعداد قله‌ها را نشان داده و محور عمودی مقدار خطای برون‌خطی را نشان می‌دهد.

نتایج حاصل‌شده از شکل ۱۵ حاکی از برتری روش پیشنهادی نسبت به دو روش مشابه دیگر است. در روش mQSOE همان‌گونه‌که مشاهده می‌شود، با افزایش تعداد قله‌ها، خطای برون‌خطی به‌طور فزاینده‌ای رو به افزایش است. در روش AMSO با افزایش تعداد قله‌ها تا حدود ۵۰ قله، خطا رو به افزایش است ولی بعد از ۵۰ قله خطا نسبتاً کاهش یافته است. در روش پیشنهادی خطا با افزایش تعداد قله‌ها، نسبت به دو روش دیگر با شیب خیلی کمی افزایش یافته است.

ضمناً برای مقایسه آماری الگوریتم‌ها از آزمون t استفاده شده‌است. در این آزمون ما از ۹۸ درجه آزادی با سطح معنی‌دار برابر با ۰،۵ استفاده کرده‌ایم. نوع آزمون را paired و one-tailed قرار داده‌ایم. بعد از انجام آزمایش نتایج بدین صورت نمایش داده می‌شوند که اگر:

الگوریتم اول از الگوریتم دوم به‌صورت معناداری بهتر باشد از علامت + استفاده شده‌است. اگر الگوریتم اول از الگوریتم دوم به‌صورت معناداری بدتر باشد از علامت – استفاده شده‌است. اگر شواهد لازم برای معناداری هیچ کدام از الگوریتم‌ها پیدا نشود از علامت ~ استفاده شده‌است. جدول ۱۱ نتایج آماری را برای روش پیشنهادی و سایر روش‌ها نشان می‌دهد. در این آزمایش از پارامترهای استاندارد سناریوی دوم محک قله‌های متحرک در فرکانس تغییرات مختلف استفاده شده‌است.

۵- نتیجه‌گیری و پیشنهادات آتی

قابلیت‌های الگوریتم بهینه‌سازی ازدحام ذرات در محیط ایستا این انگیزه را فراهم کرد که در این پژوهش به ارائه ایده‌هایی جهت پویاسازی این الگوریتم پرداخته شود. ازطرفی به‌دلیل این‌که محیط‌های پویا و به‌دنبال آن مسائل بهینه‌سازی پویا در حیطه وسیعی از علوم مختلف و حتی در زندگی روزمره کاربرد دارند، بررسی و یافتن راه‌کارهایی جهت حل آن‌ها به‌عنوان یک ضرورت به‌شمار می‌آیند. در این مقاله از الگوریتمی ارائه گردید که نسخه بهبودیافته‌ای از الگوریتم بهینه‌سازی mQSO بوده، به‌عنوان الگوریتم پایه برای پویاسازی در محیط پویا استفاده شد. چنین محیط‌هایی نیازمند جستجوی هر چه دقیق‌تر بهینه(ها) و کاهش تعداد ارزیابی‌ها است. به‌همین دلیل در الگوریتم پیشنهادی از یک راه‌کار خود-تطبیقی در برای بهبود کارآیی بهره گرفته شد تا بتواند باعث افزایش قدرت جستجوی بهینه(ها) در فضای مسئله گردد. نکته مهم دیگری که در محیط‌های پویا وجود دارد، ردیابی نقطه (نقاط) بهینه جدید پس از تغییرات محیطی و نیز یافتن هر چه سریع‌تر آن(ها) تا قبل از تغییرات بعدی محیطی می‌باشد. ازاین‌رو در الگوریتم پیشنهادی از راهکار چند-دسته‌ای، راهکار ایجاد یک دسته آزاد موقع هم‌گراشدن دسته‌ها، راهکار انحصار و نیز روش افزایش تنوع بهره گرفته شد. درنهایت یک راهکار کنترل ذرات خنثی نیز برای یافتن فضاهای نامناسب مسئله به‌کار گرفته شد. الگوریتم پیشنهادی برروی معروف‌ترین تابع محک در محیط‌های پویا به‌نام MPB مختلف آزمایش گردید. در ۹۵٪ موارد، نسبت به الگوریتم‌های مورد مقایسه ازجمله الگوریتم‌های استانداردی چون mQSO، AmQSO، CellularPSO وMultiSwarmPSO نتایج قابل‌ملاحظه‌ای را از خود نشان داد.

با توجه به پیش‌فرض‌ها و نیز محاسن و معایبی که در الگوریتم پیشنهادی این مقاله مشاهده می‌شود، می‌توان به راه‌کارهای آتی جهت بهبود الگوریتم و ارائه نسخه‌های جدید اشاره کرد. این راه‌کار در قالب ۱ پیشنهاد ارائه‌شده و به‌شرح زیر می‌باشد:

الگوریتم‌های مورداستفاده در محیط‌های پویا بایستی توانایی سرعت هم‌گرایی بالا در رسیدن به نقطه (نقاط) بهینه و نیز توانایی جستجوی محلی بالایی را داشته باشند. برای تحقق این امر در الگوریتم پیشنهادی سعی شده تا از یک راه‌کار خود-تطبیقی در الگوریتم استفاده گردد. با این‌حال می‌توان با به‌کارگیری حافظه برای ذخیره موقعیت‌های بهتر و استفاده از آن‌ها در عملگر مهاجرت سرعت هم‌گرایی را بهبود بخشید.

جدول ۱۱: نتایج آزمون *t*

t-test results	Moving Peaks Bnechmark														
فرکانس تغییرات	۵۰۰	۱۰۰۰	۲۵۰۰	۵۰۰۰	۱۰۰۰۰	۵۰۰	۱۰۰۰	۲۵۰۰	۵۰۰۰	۱۰۰۰۰	۵۰۰	۱۰۰۰	۲۵۰۰	۵۰۰۰	۱۰۰۰۰
mQSO	-	+	-	-	+	+	-	+	+	+	+	+	+	+	+
AmQSO	-	+	-	-	+	+	-	+	+	+	+	+	+	+	+
MultiSwarmPSO	-	+	~	+	+	+	+	+	+	+	+	+	+	+	+
FTMPSO	+	+	+	-	+	+	-	+	+	+	+	~	+	+	+
CellularPSO	-	+	~	+	-	-	-	+	+	+	-	+	-	+	-
DPSABC	+	+	+	~	+	+	+	~	+	-	+	+	-	+	+
AMSO	+	+	+	+	-	+	+	-	+	+	-	+	+	+	-
mQSOE	+	+	+	+	-	+	+	-	+	+	+	+	~	+	-

سپاس‌گزاری

از دانشگاه آزاد به‌خاطر حمایت‌های خود از این مقاله، کمال تشکر را داریم.

مراجع

[1] R.I. Lung, D. Dumitrescu, "A Collaborative Model for Tracking Optima in Dynamic Environments". In: IEEE Congress on Evolutionary Computation, pp.564–567, 2007.

[2] F.B. Ozsoydan and A. Baykasoglu, "A multi-population firefly algorithm for dynamic optimization problems", Evolving and Adaptive Intelligent Systems (EAIS), IEEE International Conference. Pp 1-7. DOI: 10.1109/EAIS.2015.7368777, 2015.

[3] X. Hu, R.C. Eberhart, "Adaptive particle swarm optimization: detection and response to dynamic systems". In: IEEE Congress on Evolutionary Computation, Honolulu, HI, USA, vol. 2, pp. 1666–1670, 2002.

[4] S. Sadeghi, H. Parvin and F. Rad, "Particle Swarm Optimization for Dynamic Environments", Springer International Publishing, 14th Mexican International Conference on Artificial intelligence, MICAI 2015, PP 260-269, October 2015.

[5] H. Wang, D. Wang, and S. Yang, , "A Memetic Algorithm With Adaptive Hill Climbing Strategy For Dynamic Optimization Problems," Soft Computing - A Fusion of Foundations, Methodologies and Applications - Special Issue on Emerging Trends in Soft Computing - Memetic Algorithms, Vol. 13, pp. 763-780, 2009.

[6] A. Rezvanian, M. R. Meybodi, , "Tracking Extrema In Dynamic Environments Using A Learning Automata-Based Immune Algorithm," *Grid and Distributed Computing, Control and Automation*, Vol. 121, pp. 216-225, 2010.

[7] T.T. Nguyen, "Solving dynamic optimization problems by combining Evolutionary Algorithms with KD-Tree",Soft Computing and Pttern Recogonition (SoCPaR), International Conference, pp. 247-252, 2013.

[8] Y. Xin, T. Ke, and Y. Xin, , "Immigrant Schemes For Evolutionary Algorithms In Dynamic Environments: Adapting The Replacement Rate," *Science in China Series F - Information Sciences*, Vol. II, pp. 543-552, 2011.

[۹] هوشمند، محکمی، خدابخشیان، "روشی جدید در جایابی بهینه خازنها و ژنراتورهای توزیع شده در شبکه‌های توزیع با استفاده از الگوریتم جستجوی باکتریای جهت داده شده با pso". مجله مهندسی برق دانشگاه تبریز. جلد ۳۹. شماره ۲. ۱۳۸۹.

[۱۰] محمدپور و پروین، "الگوریتم ژنتیک آشوب‌گونه مبتنی بر حافظه و خوشه‌بندی برای حل مسائل بهینه‌سازی پویا". مجله مهندسی برق دانشگاه تبریز، جلد 64 ، شماره3 ، پائیز۱۳۹۵.

[11] M. Kamosi, A.B. Hashemi, M.R. Meybodi, "A New Particle Swarm Optimization Algorithm for Dynamic Environments". SEMCCO. pp. 129-138, 2010.

[12] I. Rezazadeh, M. R. Meybodi, and A. Naebi, "Adaptive Particle Swarm Optimization Algorithm For Dynamic Environments," *ICSI'11 Proceedings of the Second international conference on Advances in swarm intelligence*, vol. I, pp. 120-129, 2011.

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

[14] T. Blackwell and J. Branke, "Multi-Swarms, Exclusion, and Anti-Convergence in Dynamic Environments," IEEE Transactions on Evolutionary Computation 10, 459–472, 2006.

[15] T. Blackwell, J. Branke and X. Li, "Particle swarms for dynamic optimization problems," Swarm Intelligence. Springer Berlin Heidelberg, 193-217, 2008.

[16] S. Yang and C. Li, "Fast Multi-Swarm Optimization for Dynamic Optimization Problems," Proc, Int'l Conf. Natural Computation, vol. 7, no. 3, pp. 624-628, 2008.
[17] B. Hashemi and M. R. Meybodi, "Cellular PSO: A PSO for Dynamic Environments", in Advances in Computation and ntelligence, Lecture Notes in Computer Science, vol. 5821, pp. 422-433, 2009.
[18] C. Li, S. Yang and M. Yang, "An Adaptive Multi-Swarm Optimizer for Dynamic Optimization Problems", in Evolutionary Computation, vol. 22 no. 4, pp. 559–594, 2014.
[19] P. Novoa-Hernández, C. Cruz Corona, D. A. Pelta, "Efficient multi-swarm PSO algorithms for dynamic environments", Memetic Comp, vol. 3, pp.163–174, 2011.
[20] D.Yazdani, B. Nasiri, A.Sepas-Moghaddam, Meybodi, M.R., "A Novel Multi-Swarm Algorithm For Optimization In Dynamic Environments Based On Particle Swarm Optimization," *Applied Soft Computing*, Vol.13, pp. 2144- 2158, 2013.
[21] N . Krasnogor and j. Smith . "*A Tutorial for Competent Memetic Algorithms: Model, Taxonomy, and Design Issues,"* IEEE Trans. Evolutionary Computation, vol. 9, no. 5, pp. 474-488, 2005.
[22] J. Branke, "Evolutionary Optimization in Dynamic Environment". Kluwer, 2002.
[23] J. K. Kordestani, A. Rezvanian, and M. R. Meybodi, "CDEPSO: a bi-population hybrid approach for dynamic optimization problems," *Applied intelligence*, vol. 40, pp. 682-694, 2014.
[24] N. Baktash and M. R. Meybodi, "A New Hybrid Model of PSO and ABC Algorithms for Optimization in Dynamic Environment," *Int'l Journal of Computing Theory Engineering*, vol. 4, pp. 362-364, 2012.
[25] S. Yang and C. Li, "A clustering particle swarm optimizer for locating and tracking multiple optima in dynamic environments," IEEE Trans.Vol 16, no. 4, Aug 2012.

زیرنویس‌ها

[1] Diversity
[2] Lung
[3] Dumitrescu
[4] CROWDING BASED DIFFERENTIAL EVOLUTION
[5] Hu and Eberhart
[6] Adaptive Dual Mapping (ADM)
[7] triggered random immigrants
[8] action
[9] action probability vector
[10] state
[11] replacement rate
[12] Adaptive Particle Swarm Optimization
[13] Cooprative Particle Swarm Optimization
[12] Personal Best
[15] Global Best
[16] Inertia Wight
[17] Acceleration Coefficients
[18] Quantum particles
[19] awakening–sleeping
[20] Anti-Convergence
[21] chaos
[22] Random
[23] Control Particle Trajectory
[24] seed
[25] Moving Peaks Benchmark
[26] Moore