بهینه‌سازی در محیط‌های غیرقطعی و پیچیدهٔ پویا با روش‌های تکاملی

سید مسعود اجابتی* **و سید حمید ظهیری**

دانشکده مهندسی برق و کامپیوتر، دانشگاه بیرجند، بیرجند، ایران

چکیده

در دنیای واقعی بسیاری از مسائل بهینه‌سازی، پویا، غیرقطعی و پیچیده هستند که در آن تابع هدف یا محدودیت‌ها می‌توانند در طول زمان تغییر یابند و درنتیجه، بهینهٔ این مسائل نیز می‌تواند تغییر کند؛ ازاین‌رو الگوریتم‌های بهینه‌سازی نه‌تنها باید مقدار بهینه سراسری را در فضای جستجو پیدا، بلکه باید مسیر تغییرات بهینه را در محیط پویا دنبال کنند. در این مقاله برای دست‌یابی به این توانایی الگوریتم جدیدی بر مبنای الگوریتم بهینه‌سازی ذرات به نام الگوریتم بهینه‌سازی ذرات افزایشی کاهشی، پیشنهاد شده است. این الگوریتم همواره در روند بهینه‌سازی به‌طور انطباقی با کاهش یا افزایش تعداد ذرات الگوریتم، توانایی یافتن و دنبال‌کردن تعداد بهینه متغیر با زمان را در محیط‌هایی که تغییرات آن قابل آشکارسازی نیست، دارد؛ علاوه‌براین تعریف جدیدی به نام ناحیه جستجو متمرکز با هدف برجسته‌کردن فضاهای امیدبخش برای سرعت بخشیدن به فرآیند جستجوی محلی و جلوگیری از همگرایی زودرس تعریف شده است. نتایج حاصل از الگوریتم پیشنهادی بر روی معیار قله‌های متحرک ارزیابی و با نتایج چندین الگوریتم معتبر مقایسه شده است. نتایج نشان‌دهنده تأثیر مثبت سازوکار کاهش/افزایش ذرات بر زمان یافتن و دنبال‌کردن چندین بهینه در مقایسه با سایر الگوریتم‌های بهینه‌سازی مبتنی بر چند جمعیتی است.

واژگان کلیدی: افزایش و کاهش ذرات، مسائل بهینه‌سازی پویا (DOPs)، روش چندجمعیتی، الگوریتم بهینه‌سازی ذرات

Optimization in Uncertain and Complex Dynamic Environments with Evolutionary Methods

Seyyed Masoud Ejabati* & Seyed Hamid Zahiri

Electrical and Computer Faculty, University of Birjand, Birjand, Iran.

Abstract

In the real world, many of the optimization issues are dynamic, uncertain, and complex in which the objective function or constraints can be changed over time. Consequently, the optimum of these issues is changed nonlinearly. Therefore, the optimization algorithms not only should search the global optimum value in the space but also should follow the path of optimal change in dynamic environment. Accordingly, several researchers believe in the effectiveness of following a series of optimums compared to a global optimum. Therefore, when an environment is changed, following a global optimum in a series of best optimums is more efficient.

Evolutionary algorithms (EA) were inspired by biological and natural evolution. Because of changing characteristic of nature, it can be a good option for dynamic optimization. In recent years, different methods have been proposed to improve EA of static environments. One of the most common methods is multi-population method. In this method, the whole space is divided into sub-spaces. Each sub-space covers some local optimums and represents a sub-population. The algorithm updates the particles of each sub-space and

* Corresponding author

*نویسندهٔ عهده‌دار مکاتبات

● تاریخ ارسال: ۹۶/۰۸/۲۶ ● تاریخ پذیرش: ۹۷/۱۲/۲۲ ● تاریخ انتشار: ۲۸ ۹۸/۱۲

[Downloaded from jsdp.rcisp.ac.ir on 2026-11-04]

[DOI: 10.29252/jsdp.16.4.27]

searches the best optimum. The most challenging issue of multi-population method is to create the desired number of sub-population and people to cover different sub-spaces in the search space.
In the present study, in order to deal with the challenges, a new algorithm based on particle optimization algorithm, which is called decrement and increment particle optimization algorithm, was proposed. The algorithm is able to follow and find the number of time-varied optimum in an environment with invisible changes by increasing or decreasing the number of particles adaptively.

Another challenging issue in dynamic optimization is the detection of environmental changes, due to the impossibility of this issue and failure of detection-based algorithms. In the proposed method, there is no need to detect the environmental changes and it always adapts itself to the environment.
Furthermore, the terms of focused search area were defined to emphasize on promising spaces to accelerate the local search process and prevent early convergence. The results of the proposed algorithm were evaluated on moving peaks and compared with several valid algorithms. The results showed the positive effect of decrement/increment mechanism of particles on finding and following time of many optimums compared to other multi-population based optimization algorithm.

Keywords: Increase decrease particle, Dynamic optimization problems (DOPs), Multi-population approach, Particle swarm optimization

۱- مقدمه

در سال‌های اخیر استفاده از هوش جمعی و الگوریتم‌های تکاملی برای بهینه‌سازی در محیط‌های پویا[1] و غیرقطعی[2] بسیار مورد توجه قرار گرفته است [1, 2].

در دنیای واقعی بسیاری از مسائل در محیط‌های پویا که یکی از عمومی‌ترین انواع آن مسائل غیرقطعی است، اتفاق می‌افتند. محیط‌های پویا بر خلاف محیط‌های ایستا[3] همواره در حال تغییر و درنتیجه مکان و مقدار نقاط بهینه نیز تغییر می‌کنند؛ ازاین‌رو الگوریتم‌های بهینه‌سازی نه‌تنها باید پاسخ بهینه سراسری را در فضای جستجوی خاصی پیدا کنند، بلکه نیاز دارند که به‌طور پیوسته تغییرات بهینه سراسری و گاه چندین بهینه نزدیک به آن را در محیط‌های مختلف دنبال کنند؛ ازاین‌رو الگوریتم‌ها باید توانایی تطابق با تغییرات محیطی را داشته باشند. بنابراین اهداف، چالش‌ها و سنجش عملکرد و توابع محک در بهینه‌سازی محیط‌های پویا با محیط‌های ایستا به‌طورکامل متفاوت است.

هم‌گرایی زودرس، گیرافتادن در بهینهٔ محلی و مصالحه بین اکتشاف و استخراج ازجمله چالشی‌هایی است که در محیط‌های ایستا با آن برخورد می‌کنیم، اما در محیط‌های پویا علاوه‌بر چالش‌های قبلی آشکارسازی تغییر در محیط، تبدیل بهینه محلی[4] به بهینه سراسری[5] و بالعکس، ازدست‌دادن تنوع[6] بعد از یک تغییر، حافظه منسوخ‌شده بعد از یک تغییر، احاطه بیش از یک بهینه توسط یک زیرجمعیت[7] و بی‌اطلاعی از میزان تغییرات در محیط، تغییر قسمتی از محیط و عدم اطلاع از زمان تغییر در محیط را نیز می‌توان نام برد.

به‌دلیل آن‌که الگوریتم‌های تکاملی برگرفته از تکامل طبیعی یا بیولوژیکی هستند و طبیعت نیز همواره در حال تغییر است، ازاین‌رو توانمندی خوبی برای استفاده در بهینه‌سازی پویا دارند. در سال‌های اخیر روش‌های مختلفی برای ارتقای الگوریتم‌های تکاملی سنتی که در محیط‌های ایستا کاربرد داشتند، پیشنهاد شده‌اند. این روش‌ها را می‌توان به پنج گروه دسته‌بندی کرد:

۱- افزایش تنوع بعد از تغییر در محیط [3-5]
۲- حفظ تنوع در طول اجرا [6-8]
۳- طرح‌های حافظه [9, 10]
۴- روش‌های چندجمعیتی[8] [11-17]
۵- هیبریداسیون [18, 19]

بسیاری از پژوهش‌گران بر این عقیده‌اند که برای حل مسایل بهینه‌سازی پویا، یافتن و دنبال‌کردن مجموعه‌ای از بهینه‌ها نسبت به یک بهینه سراسری مؤثرتر است [20-22]. بدین خاطر که دنبال‌کردن بهینه سراسری در مجموعه‌ای از بهترین بهینه‌ها زمانی‌که محیط تغییر می‌کند، کارآمدتر است.

در همین راستا به‌نظر می‌رسد استفاده از روش‌های چندجمعیتی مانند تعداد زیادی از پژوهش‌ها [11-17]، برای یافتن و دنبال‌کردن چندین بهینه محلی نزدیک به بهینه سراسری در بهینه‌سازی مسائل پویا، نسبت به سایر روش‌ها کارآیی بهتری داشته باشد. اصول کار بدین صورت است که کل فضای جستجو به زیرفضاهایی[9] تقسیم می‌شود؛ هر زیرفضا یک یا تعداد کمی بهینه محلی را پوشش داده و معرف یک زیرجمعیت است که الگوریتم به‌طور جداگانه ذرات هر

[1] Dynamic Environments
[2] Uncertain
[3] Static Environments
[4] Local Optimum
[5] Global optimum
[6] Diversity
[7] Sub-population
[8] Multi-population
[9] Sub-spaces

[Downloaded from jsdp.rcisp.ac.ir on 2026-11-04]
[DOI: 10.29252/jsdp.16.4.27]

زیرجمعیت را به‌روزرسانی کرده و به جستجوی بهینهٔ بهتر می‌پردازد. موضوع چالش برانگیز در روش‌های چندجمعیتی چگونگی ایجاد تعداد مناسب زیرجمعیت و تعداد مناسب افراد برای پوشش‌دادن زیرفضاهای مختلف در فضای جستجو است. به‌عنوان نمونه یانگ و لی روش خوشه‌بندی سلسله‌مراتبی را برای تقسیم خودکار فضای جستجو به زیرجمعیت‌ها به‌کار گرفته‌اند [22, 23].

بسیاری از الگوریتم‌های بهینه‌سازی پویا بر مبنای آشکارسازی تغییر[1] در محیط شکل گرفته‌اند و در آن‌ها روش‌هایی برای آشکارسازی تغییر در محیط استفاده شده است. آشکارسازی تغییرات در محیط کاری سخت و حتی در مواقعی می‌تواند غیرممکن باشد. فرض کنید که فقط قسمتی از کل فضای جستجو دست‌خوش تغییر شود، در این حالت پیش‌بینی آن زیرفضا و یا آشکارسازی تغییر بسیار دشوار می‌شود؛ به این دلیل الگوریتم‌هایی که نیازی به آشکارسازی تغییرات در محیط نداشته باشند، در محیط‌های گوناگون عملکرد بهتری از خود نشان می‌دهند.

در این مقاله جهت استفاده مؤثر از روش چندجمعیتی در محیط‌های غیرقطعی و غیرقابل آشکارسازی، الگوریتم idPSO[2] پیشنهاد شده است. الگوریتم معرفی‌شده بر مبنای الگوریتم PSO[3] برای محیط‌های پویا طراحی شده و برای سرعت‌بخشیدن به روند بهینه‌سازی از سازوکار ساده و در عین حال قدرتمندی برخوردار است. در روش پیشنهادی، برای مسائلی چون چگونگی ایجاد زیرجمعیت‌ها، جلوگیری از همگرایی زودرس و اجتماع بیش از حد افراد و چگونگی تطبیق رفتار افراد با شرایط محیطی راه کارهای نوینی ارائه شده است. همچنین تعاریف جدیدی به نام ناحیه جستجوی متمرکز و شعاع جستجو معرفی شده است.

الگوریتم پیشنهادی بر روی تابع محک قله‌های متحرک[4] [24, 25] که یکی از معروف‌ترین مسائل بهینه‌سازی در محیط‌های پویا برای ارزیابی الگوریتم‌های بهینه‌سازی پویا به‌شمار می‌رود [26] به‌کار گرفته شده است. برای ارزیابی، عملکرد idPSO با چندین الگوریتم که از روش‌های چندجمعیتی برای حل مسائل بهینه‌سازی پویا استفاده کرده‌اند، مقایسه و همچنین برای بررسی حساسیت الگوریتم نسبت به پارامترهای ساختاری خود آزمایش‌هایی انجام شده است.

سازماندهی این مقاله به این صورت است که در بخش دو به مرور روش‌های چند جمعیتی که برای حل مسائل پویا توسعه یافته‌اند، می‌پردازیم. ساختار الگوریتم پیشنهادی idPSO را در بخش سه بیان می‌کنیم. بخش چهار شامل نتایج به‌دست‌آمده از عملکرد الگوریتم در مواجهه با تابع محک قله‌های متحرک با تنظیم‌های مختلف است و درنهایت مقاله با نتیجه‌گیری و بحث در بخش پنج به پایان می‌رسد.

۲- روش‌های چندجمعیتی در محیط‌های پویا

در بهینه‌سازی محیط‌های پویا حفظ گوناگونی جمعیت یکی از اساسی‌ترین مسائل به شمار می‌رود. در همین راستا بسیاری از پژوهش‌ها در مواجهه با مسائل چندهدفه[5] در محیط‌های پویا نشان داده‌اند که برای حفظ گوناگونی جمعیت، رویکرد چندجمعیتی بسیار مؤثر است.

در [11-13] الگوریتم ACO چندکلونی[6] به‌کار گرفته شده است که در آن برای به بیشینه‌رساندن اکتشاف[7] در کل فضای جستجو، هر کلونی از جدول فرمون جداگانه‌ای استفاده می‌کند. اگرچه روش صریح و روشنی برای حفظ کلونی‌ها در کل فضای جستجو ارائه نشده اما نتایج حاکی از عملکرد بهتر نسبت به الگوریتم تک کلونی بود. در [14, 15] نیز از الگوریتم ACO چند کلونی استفاده شده است با این تفاوت که جدول فرمون برای همه کلونی‌ها که قبیله نامیده شده‌اند، مشابه است و تفاوت در تنظیم پارامترهای آن‌ها مسبب رفتارهای مختلف قبیله‌ها برای پوشش‌دادن نواحی بیشتری از فضای جستجو می‌شود. مشابه چنین ایده‌ای در PSO نیز به‌کار گرفته شده است [16].

در [27] الگوریتم PSO چندجمعیتی بر اساس مدل جزیره‌ای معرفی شده است؛ به‌طوری‌که ذرات به‌طور منظم بین جمعیت‌های مختلف مهاجرت می‌کنند. در [17] الگوریتم متفاوت عمل کرده و ارتباط بین جمعیت‌ها فقط زمانی رخ می‌دهد که تغییر در محیط مشاهده شود.

در پژوهش [28] کل جمعیت به دو دسته جمعیت والدین که موظف به جستجوی فضای جستجو و جمعیت فرزندان که موظف به دنبال‌کردن بهینه‌ها هستند، تقسیم شده‌اند. جمعیت والدین همواره در حال بررسی شرایط برای

[1] Detect change
[2] Increasing/Decreasing particle swarm optimization
[3] Particle swarm optimization
[4] Moving peaks Benchmark (MPB)
[5] Multi Modal
[6] Multi-colony ACO
[7] Exploration

[Downloaded from jsdp.rcisp.ac.ir on 2026-11-04]
[DOI: 10.29252/jsdp.16.4.27]

ایجاد جمعیت کودکان هستند. گفتنی است که تعداد کل افراد همواره ثابت است. الگوریتم بهینه‌سازی چندجمعیتی سریع[1] [29] با این تفاوت که جمعیت والدین به‌عنوان جمعیت پایه بعد از تغییر در محیط به‌دنبال مناطق محتمل‌تر می‌شود و جمعیت کودکان برای جستجوی محلی در این مناطق محتمل استفاده می‌شوند، ارائه شد. ایده‌ای مشابه توسط کاموسی و همکاران معرفی شده است که در آن جمعیت کودکان در صورت مؤثرنبودن تا زمانی‌که تغییر در محیط احساس نشود به‌اصطلاح در خواب زمستانی می‌روند [30].

در [31, 32] از خواص اتم‌ها و اصل دافعه ذرات با بار هم‌نام برای حفظ تنوع در جمعیت‌ها استفاده شده است. بدین صورت که جمعیت‌های باردار برای پوشش‌دادن بهینه‌ها به‌صورت مجزا معرفی شده‌اند. در نسخه بهبودیافته دو قانون ابتکاری برای افزایش تنوع اضافه شده است. به موجب یکی از قوانین در زمان ایجاد تغییر در محیط تعداد ذرات کوانتوم افزایش و تعداد مسیرهای ذرات کاهش می‌یابد. در قانون دوم ذرات با عملکرد نامناسب مقداردهی اولیه و یا متوقف می‌شوند [33].

الگوریتم بهینه‌سازی ذرات خوشه‌ای[2] توسط لی و یانگ ارائه شده که در آن از یک روش خوشه‌بندی سلسله‌مراتبی برای تقسیم جمعیت اولیه به زیرجمعیت‌هایی برای پوشش‌دادن به مناطق مختلف محلی استفاده شده است [22, 23].

یزدانی و همکاران یک الگوریتم چندجمعیتی به نام finder-tracker multi-swarm PSO پیشنهاد کرد. FTMPSO با استفاده از چندین سازوکار برای مقابله با چالش‌های موجود در محیط‌های پویا از جمله: طرح شناسایی بهینه، طرح ردیابی بهینه، سازوکار بیداری و خواب برای زیرجمعیت‌ها، یک روش تشخیص تغییر در محیط و یک روش جستجوی محلی است، شکل گرفته است [34].

در پژوهش [35] روشی چندمنطقه‌ای[3] مبتنی بر WDO[4] اصلاح‌شده به‌کار گرفته شده است. در این روش که الهام‌گرفته از هواشناسی است با توجه به مناطق غیر محتمل به‌منظور شناسایی مناطق امیدوارکننده در طی فرآیند جستجو راه‌کارهایی ارائه شده است. منظور از مناطق امیدوارکننده، منطقه‌ای است که ممکن است، شامل بهینه‌هایی باشد که هر یک از آن‌ها به‌احتمال در محیط بعدی، به بهینه سراسری مبدل شوند.

در پژوهشی دیگر، یک روش تطبیقی مبتنی بر الگوریتم جستجوی فاخته[5] برای حل مسائل بهینه‌سازی پویای پیوسته پیشنهاد شده است [36]. در این ارتقا این الگوریتم برای محیط‌های پویا، به‌جای استفاده از یک مقدار برای پرواز لووی[6] از دو مقدار یکی برای جهش‌های[7] کوتاه و دیگری برای جهش‌های طولانی استفاده شده است. این سازوکار سبب بهبود کنترل الگوریتم بر اکتشاف و استخراج[8] نقاط بهینه در طی روند جستجو می‌شود.

فولادگر و لطفی [37] ایده‌هایی دیگر به الگوریتم جستجوی فاخته برای بهینه‌سازی پویا اضافه کرده‌اند. رویکرد آن‌ها استفاده از یک الگوریتم جستجوی فاختهٔ اصلاح‌شده برای افزایش میزان هم‌گرایی جمعیت برای یافتن قله‌ها و استفاده از محرومیت[9] برای جلوگیری از هم‌گراشدن جمعیت به مناطق مشابه در فضای جستجو است.

۱-۲- محیط‌های پویا با تغییرات غیرقابل تشخیص

در بسیاری از پژوهش‌هایی [38] که تاکنون برای حل مسائل بهینه‌سازی پویا[10] (DOPs) بر مبنای الگوریتم‌های تکاملی صورت گرفته است، بر مبنای تشخیص تغییر در محیط [22, 23, 32, 39, 40] و یا پیش‌بینی تغییرات با فرض این‌که تغییرات از یک الگوی خواص تبعیت می‌کنند [41]، توسعه یافته‌اند. هنگامی که تغییرات تشخیص داده شد و یا پیش‌بینی شد آن گاه از استراتژی‌های گوناگونی برای افزایش تنوع بهره می‌گیرند.

با توجه به اهمیت حفظ تنوع در حل مسائل بهینه‌سازی پویا و وابستگی آن به تشخیص تغییرات محیط، حائز اهمیت است که عملکرد الگوریتم‌هایی که برای محیط‌های پویا پیشنهاد می‌شود وابسته به شناسایی تغییرات در محیط نباشند؛ زیرا آشکارسازی تغییرات محیط در برخی موارد بسیار دشوار و غیرممکن می‌شود. به‌طور مثال تصور کنید که فقط برخی نواحی از محیط جستجو دست‌خوش تغییر شوند و این قضیه روند الگوریتم را برای تشخیص تغییر در محیط بسیار سخت می‌کند و درصورتی‌که موفق به کشف تغییر نشود، تمامی استراتژی‌های متداول به هم ریخته و الگوریتم از کار خواهد افتاد؛ همچنین در محیط‌های نوفه‌ای

[1] Fast multi-swarm optimization (FMSO)
[2] Clustering PSO (CPSO)
[3] Multi-region
[4] Wind Driven Optimization
[5] Cuckoo search
[6] Levy flight
[7] Mutation
[8] Exploitation
[9] Exclusion
[10] Dynamic optimization problems

[Downloaded from jsdp.rcisp.ac.ir on 2026-11-04]
[DOI: 10.29252/jsdp.16.4.27]

تشخیص تغییر برای این دست از الگوریتم‌ها، بسیار دشوار است. برای مقابله با چنین نقطه ضعفی، الگوریتم پیشنهادی نیازی به کشف تغییر در محیط ندارد و همواره به‌صورت تطبیقی خود را با شرایط محیطی وقف می‌دهد.

در ادامه برای بهبود در حل مسائل بهینه‌سازی پویا و غلبه بر کاستی‌های مرورشده در این بخش، به توصیف روش پیشنهادی پرداخته می‌شود.

۳- الگوریتم بهینه سازی ذرات افزایشی کاهشی

در این بخش الگوریتم جدیدی به نام Inceasing/Decreasing PSO برای بهینه‌سازی در محیط‌های پویا معرفی شده است. برای همگام‌سازی PSO مرسوم برای محیط‌های پویا، ذرات به دو گروه ذرات آزاد و ذرات متمرکز تقسیم می‌شوند.

وظیفه ذرات آزاد جستجوی همیشگی در کل فضای جستجو است. زمانی‌که این ذرات به سمت هم‌گرایی پیش بروند تعدادی از آن‌ها که در محدوده‌ای به نام ناحیه جستجوی متمرکز[1] FSZ قرار دارند، به‌عنوان ذرات متمرکز معرفی می‌شوند. محدوده ناحیه جستجوی متمرکز حول بهترین ذره متمرکز با شعاعی با نام شعاع جستجو متمرکز[2] FSR مشخص می‌شود. با به‌روزرسانی ذرات متمرکز در صورت بهبود بهترین ذره، مرکز FSZ نیز به مکان بهترین ذره متمرکز تغییر پیدا می‌کند و سایر ذرات متمرکز این ناحیهٔ جستجو، ملزم به قرارگرفتن و جستجو در این زیرفضا می‌شوند. هم‌زمان تمامی ذرات آزاد برای یافتن نقاط بهینه دیگر در فضای جستجو پراکنده می‌شوند. درواقع وظیفه ذرات آزاد جستجوی ابتدایی و یافتن زیرفضاهای محتمل و وظیفهٔ ذرات متمرکز یافتن و بهبود بهینه و دنبال‌کردن آن در حین تغییرات محیط است؛ بدین صورت تعداد ذرات با توجه به ایجاد نواحی جستجوی متمرکز و به تبع آن تولید ذرات متمرکز جدید افزایش می‌یابد. ساختار الگوریتم به‌گونه‌ای است که هر زمان بسته به شرایط محیط بهینه مربوط به هر ناحیه جستجوی متمرکز از بین برود، کلیه ذرات متمرکز عضو آن ناحیه حذف می‌شوند. بدین صورت همواره تعداد ذرات به‌طور تطبیقی در حال تغییر است. در الگوریتم (۱) چهارچوب کلی الگوریتم نشان داده شده است.

[1] Focused Search Zone
[2] Focused Search Radius

(الگوریتم-۱): چهارچوب کلی الگوریتم پیشنهادی
(Algorithm-1): The general framework of the porposed algorithm

Algorithm 1 *idPSO)*

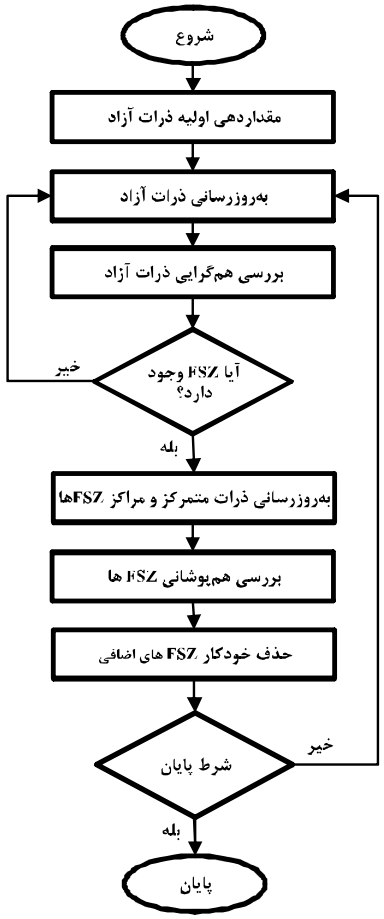
```
1   Initialize the free particles
2   while stop criteria is not satisfied do
3      for free particles do
4         PSO();
5      end for
6      convergenceChecking (free particles)
7      if FSZ exist then
8         for each FSZ[i] do
9            PSO(FSZ[i])
10           Update FSZ centers
11           Check velocity of focus particles
12        end for
13     end if
14     FSZoverlapChecking (FSZ,FSR)
15     RemoveUselessFSZ
16  end while
```

حلقه اصلی الگوریتم پیشنهادی با مقداردهی اولیه برای ذرات آزاد شروع می‌شود و در مرحله بعدی مکان‌های این ذرات توسط PSO مرسوم[3] به‌روز می‌شود. در ادامه هم‌گرایی ذرات آزاد مورد بررسی قرار می‌گیرد؛ درصورت تشخیص امکان هم‌گرایی، ذراتی که در ناحیه جستجوی متمرکز (FSZ) یعنی در شعاع جستجوی متمرکز (FSR) حول بهترین ذره باشند، به‌عنوان ذرات متمرکز، شناخته شده و یک ناحیهٔ جستجوی متمرکز جدید ایجاد می‌شود. حال مکان ذرات متمرکز هر ناحیهٔ جستجوی به‌طور جداگانه توسط PSO به‌روزرسانی می‌شود و درصورت بهبود بهینهٔ هر FSZ، مرکز به بهینهٔ مناسب‌تر یافت‌شده انتقال می‌یابد. در مرحلهٔ بعدی نواحی جستجوی متمرکز از جهت روی هم‌افتادگی بررسی و درنهایت FSZ هایی که بهینه خود را در اثر تغییرات شدید شرایط محیطی از دست داده‌اند، حذف می‌شوند؛ چون فرض بر این است که تغییر مکانی بهینه پس از تغییر در محیط در محدوده FSZ باشد. روند روش پیشنهادی در روندنمای شکل (۱) نمایش داده شده است.

در الگوریتم پیشنهادی ذرات به دو گروه ذرات آزاد و ذرات متمرکز تقسیم می‌شوند. ذرات آزاد در قسمت نخست الگوریتم به تعداد از پیش تعیین‌شده، مقداردهی اولیه است. یکی از ویژگی‌های الگوریتم پیشنهادی متغیربودن تعداد ذرات متمرکز در طول زمان بسته به شرایط محیط است. بدین‌سان تعداد کل ذرات در طول اجرای الگوریتم همواره تابع شرایط محیطی است و این ویژگی الگوریتم را از قید به‌روزرسانی همیشگی تعداد بالای ذرات جدا می‌سازد.

[3] Traditional

[Downloaded from jsdp.rcisp.ac.ir on 2026-11-04]
[DOI: 10.29252/jsdp.16.4.27]



(شکل-۱): فلوچارت الگوریتم پیشنهادی

(Figure -1): Flowchart Proposed Algorithm

در گام بعدی موقعیت ذرات آزاد توسط الگوریتم PSO که نخستین‌بار توسط کندی و ابرهارت پیشنهاد شد [42, 43]، به‌روزرسانی می‌شود. روند الگوریتم PSO در الگوریتم (۲) نشان داده شده است. هر ذره i با یک بردار سرعت $\vec{v}_i$ و بردار موقعیت $\vec{x}_i$ نمایش داده و توسط نسخه [44] با وزن انرسی[1] به‌صورت زیر به‌روزرسانی می‌شود:

$$v_i'^d = \omega v_i^d + c_1 r_1 \left(x^d_{pbest_i} - x_i^d\right) + c_2 r_2 \left(x^d_{gbest} - x_i^d\right) \quad (۱)$$

$$x_i'^d = x_i^d + v_i'^d \quad (۲)$$

در رابطهٔ بالا $x_i'^d$ موقعیت کنونی، x_i^d موقعیت قبلی، $v_i'^d$ سرعت کنونی و v_i^d سرعت قبلی ذره iام در بُعد dام است. $\vec{x}^d_{pbest_i}$ بهترین موقعیت ذره iام تاکنون و $\vec{x}^d_{gbest}$ بهترین موقعیت در بین کل ذرات است. $\omega \in (0.1)$ وزن انرسی است و میزان تأثیر سرعت قبلی در سرعت کنونی ذره را تعیین می‌کند. r_1 و r_2 اعدادی تصادفی بین صفر و یک هستند. c_1 و c_2 با نام ثابت شتاب شناخته می‌شوند و به‌ترتیب میزان تبعیت ذره از بهترین خود (جزء شناختی) و بهترین جمعی (جزء اجتماعی) را نشان می‌دهد.

[1] Inertia weight

(الگوریتم-۲): الگوریتم بهینه‌سازی ذرات

(Algorithm-2): The process of PSO algorithm

Algorithm 2 *PSO*

```
Evaluate the fitness of each particle
for each particle i do
    Update particle i according to (1) and (2);
    if f(x_i) < f(x_pbest_i) then
        x_pbest_i := x_i;
        if f(x_i) < f(x_gbest) then
            x_gbest := x_i;
        end if
    end if
end for
```

در گام بعدی حلقه اصلی الگوریتم پیشنهادی، روند هم‌گرایی ذرات آزاد مورد بررسی قرار می‌گیرد. شرط هم‌گرایی اولیه، بدین صورت است که میزان تغییرات موقعیت x_{gbest} ذرات آزاد و برازندگی آن نسبت به مرحله قبل سنجیده می‌شود. روند بررسی هم‌گرایی ذرات آزاد در الگوریتم (۳) نمایش داده شده است. درصورتی‌که تغییرات x_{gbest} کمتر از $r_{conv}/5$ و تغییرات برازندگی آن کمتر از r_{conv} باشد، الگوریتم امکان هم‌گرایی ذرات آزاد را تشخیص می‌دهد. در این مرحله به بررسی موقعیت بهترین نقطه یافت‌شده پرداخته می‌شود و درصورتی‌که در هر یک از FSZها واقع شده باشد و مقدار برازندگی آن بهینه‌تر از برازندگی مرکز FSZ واقع در آن باشد، FSZ یادشده و به تبع آن ذرات متمرکز عضو حذف و FSZ جدید با تعداد ذرات آزادی که در محدوده FSR به مرکزیت x_{gbest} باشند، تشکیل و تمامی ذرات آزاد برای جستجوی نواحی محتمل دیگر در کل فضای جستجو پراکنده می‌شود. از جهت دیگر درصورتی‌که مقدار برازندگی x_{gbest} بهینه‌تر از برازندگی مرکز FSZ واقع در آن نباشد، تمامی ذرات آزاد پراکنده می‌شوند.

پس از ارضای شرط اولیه هم‌گرایی، اگر موقعیت بهترین نقطه ذرات آزاد در هیچ یک از FSZها واقع نشده باشد، آنگاه به بررسی مجموع فواصل دو ذره آزاد که کمترین فاصله را نسبت به x_{gbest} دارند، پرداخته می‌شود؛ درصورتی‌که این مجموع نیز مقداری کمتر از r_{conv} داشته باشد، نشانهٔ آن است که ذرات آزاد رو به هم‌گرایی هستند و x_{gbest} به‌عنوان مرکز FSZ در نظر گرفته شده و تعداد ذرات آزاد تا بیشینهٔ چهار

[Downloaded from jsdp.rcisp.ac.ir on 2026-11-04]

[DOI: 10.29252/jsdp.16.4.27]

ذره که در محدوده FSR نسبت به این مرکز باشند به‌عنوان ذرات متمرکز آن FSZ معرفی و تمامی ذرات آزاد برای جستجوی مناطق محتمل دیگر در کل فضای جستجو پراکنده می‌شوند. درصورتی‌که تعداد ذرات آزاد در محدوده FSZ جدید کمتر از دو ذره باشد، دو ذره نزدیک‌تر به x_{gbest} به‌عنوان ذرات متمرکز برای FSZ یادشده پذیرفته می‌شوند.

(الگوریتم-۳): روند بررسی همگرایی ذرات آزاد

(Algorithm-3): The process of investigating free particle convergence

```
Algorithm 3 convergenceChecking ()
if (f(x_gbest)^(t-1)-f(x_gbest)^t < r_conv) and (||(x_gbest)^(t-1),
(x_gbest)^t | < r_conv/5) then
    if x_gbest was within FSZ[i] then
        if f(x_gbest) better than f(center_FSZ[i]) then
            Replace selected free particles with
             focus... particles in FSZ[i]
            re-initialize the free particles
        else
            re-initialize the free particles
        end if
    else
        create a new FSZ with selected free …
         particles as the focus particles
        re-initialize the free particles
    end if
end if
```

در گام بعدی موقعیت ذرات متمرکز هر FSZ به‌طور مجزا توسط الگوریتم PSO به‌روزرسانی می‌شوند و درصورت بهبود بهترین ذره هر FSZ، مرکز آن FSZ به موقعیت بهترین ذره تغییر مکان می‌دهد. بدین صورت موقعیت مکانی FSZ ها نیز در فضای جستجو به‌روزرسانی می‌شود. الگوریتم همچنین سرعت ذرات متمرکز را رسد می‌کند تا هر زمان نزدیک به صفر شد، با پراکنده‌کردن ذرات متمرکز یادشده در محدوده FSZ خودشان، از گیرافتادن در بهینه محلی جلوگیری کند.

در الگوریتم پیشنهادی سازوکاری برای جلوگیری از هم‌پوشانی FSZ ها و موازی‌کاری ذرات متمرکز به‌کار گرفته شده است. در این روش فواصل بین مراکز FSZ ها محاسبه شده و دو FSZ همسایه که فاصله مراکزشان کمتر از دو برابر FSR از هم باشند، در نظر گرفته می‌شوند؛ حال برای اجتناب از تحت تأثیر قرارگرفتن الگوریتم از بهینه‌های محلی، میانگین برازندگی بهینه‌های یافت‌شده به‌وسیلهٔ سایر FSZاها محاسبه می‌شود. درصورتی‌که برازندگی مراکز دو FSZ مذکور بهتر از ضریبی (P_{mean}) از میانگین برازندگی مراکز سایر FSZ ها بود، شعاع جستجو به نصف فاصله مراکز این دو FSZ کاهش می‌یابد تا هر دو بتوانند بهینه خود را دنبال کنند؛ حال اگر برازندگی یکی یا هر دو FSZ بدتر از ضریبی (P_{mean}) از میانگین برازندگی مراکز سایر FSZ ها بود، FSZ ای که برازندگی بهینه یافت‌شدهٔ آن توسط ذرات متمرکز مقدار بدتری نسبت به دیگری داشت به‌عنوان بهینه محلی تلقی شده و FSZ یادشده و ذرات متمرکز مربوط به آن حذف می‌شوند. بدین‌صورت هم از جستجوی چند FSZ در یک ناحیه از فضای جستجو جلوگیری شده و هم از گیرافتادن در بهینه‌های محلی اجتناب می‌شود که نتیجه آن افزایش سرعت الگوریتم در یافتن نقاط بهینه بیشتر در کل فضای جستجو در زمان محدودتری است. در الگوریتم (۴) روند بررسی هم‌پوشانی FSZاها نمایش داده شده است.

ایدهٔ دیگر به‌کار گرفته‌شده در الگوریتم پیشنهادی که در الگوریتم (۵) نمایش داده شده، حذف خودکار FSZ هایی است که پس از تغییرات شدید در محیط دیگر شاهد نقطه بهینه در منطقه خود نیستند و یا حذف ذرات متمرکزی که در بهینه محلی گیر افتاده‌اند. در اینجا نیز ضریبی (P_{mean}) از میانگین برازندگی نقاط بهینهٔ یافت‌شده به‌عنوان معیار سنجش معرفی می‌شود و در هر مرحله با مقایسه بدترین بهینه در FSZ ها با این معیار درصورتی‌که مقداری نامناسب‌تر داشته باشند، FSZ مربوطه همراه با ذرات متمرکز آن حذف می‌شوند. در بخش بعدی روش پیشنهادی را مورد آزمایش‌های مختلف قرار می‌دهیم تا عملکرد آن را نسبت به سایر روش‌های مطرح بسنجیم.

(الگوریتم-۴): روند بررسی هم‌پوشانی FSZاها

(Algorithm-4): The FSZ overlap checking process

```
Algorithm 4 FSZoverlapChecking (FSZ,FSR)
if distance between FSZ[i] center and
FSZ[j]… center  < 2*FSR then
    f_mean(center_FSZ) – Calculation of mean
    fitness … other centers
    if f(center_FSZ[i]) and f(center_FSZ[j]) beter
    than … P_mean* f_mean(center_FSZ) and FSR>1
    then
        FSR = (distance between FSZ[i] center
        and… FSZ[j] center)  2
    else
        delete FSZ with worst fitness between
        … FSZ[i] and FSZ[j]
    end if
end if
```

(الگوریتم-۵): حذف خودکار FSZاها

(Algorithm-5): The automatic removal of FSZs

```
Algorithm 5 RemoveUselessFSZ
f_mean(center_FSZ) = Calculation of mean all
fitness… centers
if worst f(center_FSZ[i]) > f_mean(center_FSZ) then
    delete FSZ with worst fitness FSZ[i]
end if
```


[Downloaded from jsdp.rcisp.ac.ir on 2026-11-04]

[DOI: 10.29252/jsdp.16.4.27]

۴- نتایج شبیه‌سازی

در این بخش الگوریتم بهینه‌سازی ذرات افزایشی کاهشی برای حل مسأله MPB به‌کار گرفته شده است [24, 25]. آزمایش‌های انجام‌شده را می‌توان به دو قسمت تقسیم کرد.

هدف از قسمت نخست بررسی سازوکار idPSO و تجزیه تحلیل پارامترهای کلیدی در روند بهینه‌سازی الگوریتم در مواجهه با مسأله MPB است. در قسمت دوم از آزمایش‌ها، عملکرد idPSO با تعدادی از الگوریتم‌هایی که در این زمینه معرفی شده‌اند، مقایسه می‌شود. نتایج تمام الگوریتم‌هایی که در این مقاله نشان داده شده‌اند، برگرفته از مقالات پیشنهادشان است.

در الگوریتم PSO ثابت‌های شتاب c_1 و c_2 به‌ترتیب میزان تبعیت ذره از بهترین خود (جزء شناختی) و بهترین جمعی (جزء اجتماعی) را نشان می‌دهد و وزن انرسی سرعت هم‌گرایی را تغییر می‌دهد. با توجه به وظایف مختلفی که برای ذرات آزاد و ذرات متمرکز در idPSO در نظر گرفته شده است، مقادیر ثابت‌های شتاب و وزن انرسی برای به‌روزرسانی ذرات آزاد که وظیفه یافتن نواحی محتمل در کل فضای جستجو و ذرات متمرکز که موظف به یافتن و دنبال‌کردن بهینه در FSZ خودشان هستند، متفاوت در نظر گرفته شده است. مقادیر c_1 ، c_2 و ω ذرات آزاد برای جستجوی کلی به‌ترتیب ۲، ۲ و ۰/۴ و برای ذرات متمرکز با جستجوی دقیق ۳، ۱ و ۰/۵ در نظر گرفته شده است.

۴-۱- نتایج تجربی

۱. *Moving Peaks Benchmark (MPB) Problem:*

تابع محک قله‌های متحرک یکی از معروفترین مسائل بهینه‌سازی در محیط‌های پویا است و به‌طور گسترده‌ای برای ارزیابی الگوریتم‌های بهینه‌سازی پویا به‌کار گرفته شده است [24, 25]. در مسأله MPB، بهینه می‌تواند با سه ویژگی موقعیت، ارتفاع و عرض قله‌ها متفاوت باشد. این مسأله در D بعد به‌صورت زیر تعریف می‌شود:

$$F(\vec{x}.t) = \max_{i=1,\dots,p} \frac{H_i(t)}{1 + W_i(t)\sum_{j=1}^{D}\left(x_j(t) - X_{ij}(t)\right)^2} \qquad (۳)$$

در رابطهٔ (۳) $H_i(t)$ و $W_i(t)$ به‌ترتیب ارتفاع و پهنای قله i در زمان t و $X_{ij}(t)$ امین عنصر از مکان قله i در زمان t است. پارامتر p به‌طور مستقل برای قله‌های مشخص توسط تابع max مخلوط می‌شود. مکان قله در جهتی تصادفی توسط بردار $\vec{v}_i$ به اندازه یک فاصله s[1] که معرف حساسیت پویایی مسأله است، انتقال می‌یابد. حرکت یک قله تنها به‌صورت رابطه زیر تعریف می‌شود:

$$\vec{v}_i(t) = \frac{s}{|\vec{r} + \vec{v}_i(t-1)|}\left((1-\lambda)\vec{r} + \lambda\vec{v}_i(t-1)\right) \qquad (۴)$$

بردار انتقال $\vec{v}_i(t)$ یک ترکیب خطی از بردار تصادفی $\vec{r}$ و بردار انتقال قبلی $\vec{v}_i(t-1)$ است و نسبت به طول تغییر s نرمالیزه شده است. مقدار پارامتر همبستگی λ صفر در نظر گرفته شده است که نشان‌دهندهٔ ناهمبستگی حرکات قله است. روابط تغییرات یک قله به‌صورت زیر بیان می‌شود:

$$H_i(t) = H_i(t-1) + height_severity \times \sigma \qquad (۵)$$

$$W_i(t) = W_i(t-1) + width_severity \times \sigma \qquad (۶)$$

$$\vec{X}_i(t) = \vec{X}_i(t)(t-1) + \vec{v}_i(t) \qquad (۷)$$

که در آن σ یک عدد تصادفی با توزیع نرمال و میانگین صفر و واریانس یک است.

۲. *تنظیم‌های آزمایش:*

در جدول (۱) تنظیم‌های پیش‌فرض تابع محک قله‌های متحرک آورده شده است. اصطلاح تغییر فرکانس U به معنای تغییر محیط بعد از U بار ارزیابی تابع برازندگی است. مقدار Peak location range نشان‌دهندهٔ بازه تغییرات مکان قله در هر بعد است. ارتفاع قله به‌صورت تصادفی در بازه [30,70] و پهنای آن در بازه [1,12] تغییر می‌کند.

(جدول-۱): تنظیم‌های پیش‌فرض برای مسأله MPB

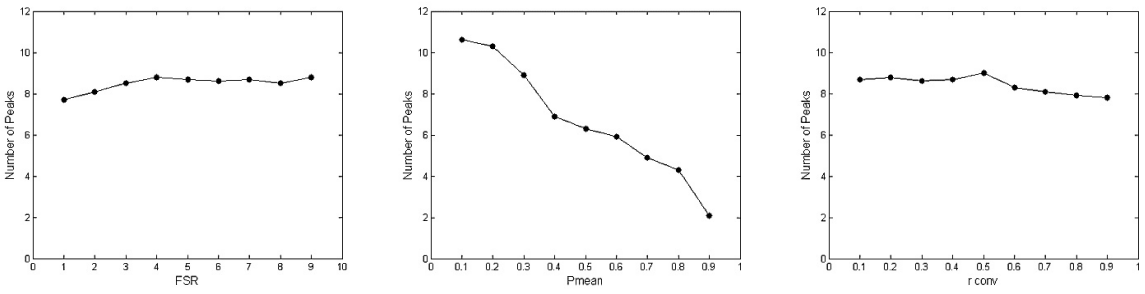
(Table-1): Default setting for MPB

Parameter	Value
Number of peaks, p	[1,200]
Change frequency, U	5000
Height severity	7.0
Width severity	1.0
Peak shape	Cone
Basic function	No
Shift length, s	1.0
Number of dimensions, D	5
Correlation coefficient, λ	0
Peaks location range	[0,100]
Peak height, H	[30,70]
Peak width, W	[1,12]
Initial value of peaks	50.0

[1] Shift length

[Downloaded from jsdp.rcisp.ac.ir on 2026-11-04]

[DOI: 10.29252/jsdp.16.4.27]



(الف) (ب) (ج)

(شکل-۲): الف) تأثیر تغییر r_{conv} بر تعداد قله‌های یافت‌شده ب) تأثیر تغییر P_{mean} بر تعداد قله‌های یافت‌شده ج) تأثیر تغییر FSR بر تعداد قله‌های یافت‌شده

(Figure-2): (a) The effect of r_{conv} change on the number of peaks found (b) The effect of p_{mean} change on the number of peaks found (c) The effect of the FSR change on the number of peaks found

۳ *اندازه‌گیری عملکرد:*

برای اندازه‌گیری عملکرد الگوریتم‌ها در محیط‌های پویا چندین معیار سنجش معرفی شده است [45]. به‌منظور ایجاد نتایج قابل قیاس با سایر روش‌های مطرح در این زمینه، از معیار خطای برون‌خط[1] که به‌صورت میانگین اختلاف مقدار بهینهٔ یافت‌شده توسط الگوریتم با مقدار بهینه سراسری در هر محیط تعریف می‌شود، استفاده شده است.

$$OE = \frac{1}{K}\sum_{k=1}^{K}(h_k - f_k) \qquad (۸)$$

در رابطه بالا f_k بهترین جواب یافت‌شده توسط الگوریتم تا قبل از تغییر kام در محیط و h_k مقدار بهینه محیط kام است. OE میانگین اختلاف f_k و h_k در کل K تغییر در محیط است [45]. همه نتایج گزارش‌شده برای بیش از پنجاه بار اجرای برنامه برای یکصد تغییر در محیط است.

۲-۴- حساسیت الگوریتم نسبت به پارامترهای ساختاری

به‌منظور بررسی حساسیت الگوریتم نسبت به پارامترهای r_{conv} (شعاع همگرایی)، P_{mean} (ضریب میانگین برازندگی) و FSR (شعاع جستجوی متمرکز) آزمایش‌هایی روی MPB با مقادیر پیش‌فرض (p=10, U=5000, s=1) صورت گرفته است.

به‌دلیل آن که مقدار خطای برون‌خط به‌ازای تغییرات پارامتر در فرکانس تغییرات پنج‌هزار، به‌طورتقریبی برابر و همواره به کمترین مقدار می‌رسد، برای مشاهدهٔ تأثیر تغییرات هر پارامتر، به‌جای بررسی مقدار خطای برون‌خط تعداد متوسط قله‌های یافت‌شده به‌وسیلهٔ الگوریتم، در نظر گرفته شده است. در شکل (۲) تاثیر تغییرات پارامترها بر روی تعداد قله‌های یافت شده، آمده است.

در شکل (۲) الف، تغییرات r_{conv} بر روی تعداد قله‌های یافت‌شده، با توجه به ماهیت تصادفی‌بودن مسأله MPB در هر تکرار، نشان از آن دارد که الگوریتم حساسیت چندانی در قبال تغییر r_{conv} نشان نمی‌دهد؛ زیرا افزایش و کاهش r_{conv} درواقع زمان انتقال وظیفه جستجوی نواحی محتمل و نقطه بهینه از ذرات آزاد به ذرات متمرکز را به‌ترتیب کم و زیاد می‌کند. هنگامی که r_{conv} افزایش می‌یابد، ذرات آزاد زودتر به سمت هم‌گرایی تشخیص داده می‌شوند و درنتیجه FSZ جدید تشکیل شده و شروع به یافتن بهینه می‌کند و ذرات آزاد دوباره به وظیفهٔ اصلی خود یعنی یافتن نواحی محتمل بر می‌شوند. در این حالت امکان ایجاد بهینه‌های محلی افزایش می‌یابد که این ضعف سریعا توسط الگوریتم‌های کنترلی با حذف FSZ یا ارتقا ذرات متمرکز برطرف می‌شود. در مقابل، با کاهش r_{conv} در واقع ذرات آزاد علاوه‌بر جستجوی نواحی محتمل، تا حد زیادی محکوم به یافتن مقدار بهینه که جزو وظایف ذرات متمرکز است، می‌شوند. این قضیه زمانی اهمیت می‌یابد که سرعت تغییرات محیط افزایش یابد (فرکانس تغییرات U کاهش یابد). در اینجا به‌دلیل توانایی الگوریتم تغییرات r_{conv} تأثیر چندانی بر مقدار خطای برون‌خط و تعداد قله‌های یافت‌شده ندارد.

در شکل (۲) ب نیز مشابه آنچه در تغییرات r_{conv}اتفاق می‌افتد، با افزایش مقدار P_{mean} ایجاد نواحی محتمل با صرف زمان بیشتری صورت می‌گیرد و تا یافتن قله به تأخیر می‌افتد. افزایش بیش از ۰/۷ P_{mean} الگوریتم را برای یافتن همه نواحی

[1] Offline error (OE)

[Downloaded from jsdp.rcisp.ac.ir on 2026-11-04]

[DOI: 10.29252/jsdp.16.4.27]

محتمل و درواقع همه قله‌ها مختل می‌کند؛ زیرا در صورتی که قله‌های یافت‌شده اولیه دارای مقدار برازندگی زیاد باشند، در نتیجه از میانگین بالاتری نیز برخورد بوده و الگوریتم‌های کنترلی دیگر اجازه یافتن قله‌های با کمینه مقدار برازندگی را نمی‌دهند. گفتنی است در این صورت، دوباره الگوریتم بهترین نقطه بهینه را که در محاسبه خطای برون‌خط به‌کار می‌رود، پیدا می‌کند؛ اما درصورت تغییرات شدید در محیط، امکان دنبال‌کردن آن توسط ذرات متمرکز از دست می‌رود که با صرف زمان دوباره توسط ذرات آزاد در مکان جدید قابل یافت‌شدن است. در مقابل، با کاهش P_{mean} به مقادیر کمتر از ۰/۳ الگوریتم دچار سردرگمی در تشخیص نواحی محتمل واقعی از محلی می‌شود و به‌طورمعمول تعداد FSZ ها از تعداد قله‌ها یعنی ده عدد فراتر می‌رود و این قضیه دوباره در مقدار بهینه سراسری یافت شده و به تبع آن خطای برون‌خط تأثیر نداشته و فقط تعداد ذرات و در نتیجه میزان محاسبات الگوریتم را بی مورد افزایش می‌دهد.

طبق تعریف r_{sr}FSR (شعاع جستجوی متمرکز) محدوده ناحیه جستجوی متمرکز حول بهترین ذره متمرکز آن ناحیه در کل روند بهینه‌سازی، را مشخص می‌کند. دو وظیفه برای این ناحیه متصور است. وظیفهٔ نخست یافتن بهینه در منطقه محتمل و بهبود هر چه بیشتر بهینه یافت‌شده تا زمانی که تغییر در محیط صورت نگیرد. وظیفه دوم زمانی است که در محیط تغییر ایجاد شود. تغییر مکان بهینه در این محیط دنبال می‌شود. درنتیجه اهمیت تنظیم مقدار FSR برای این است که پس از تغییر در محیط، همچنان بهینه در همین ناحیه جستجوی متمرکز باشد. همچنین درنظرگرفتن مقادیر زیاد FSR برای اطمینان تغییر مکان بهینه در همین ناحیه پس از تغییر، سبب می‌شود که بهبود بهینه (وظیفه نخست) به‌دلیل مواجه‌بودن با ناحیهٔ بزرگ‌تر، سخت‌تر شود. بدین‌سان باید مصالحه ای بین بهبود بهینه و وجود بهینه پس از تغییر در ناحیه برقرار شود.

در شکل (۲) ج اثر تغییر مقدار r_{sr}FSR بر تعداد قله‌های یافت‌شده، نشان داده شده است. مقادیر کم FSR (کمتر از ۲) سبب بهبود هر چه بیشتر بهینه می‌شود، اما با تغییر در محیط بیش‌تر FSZ ها بهینه خود را از دست داده و توسط سازوکار *RemoveUselessFSZ* حذف می‌شوند. درنهایت الگوریتم پیشنهادی به‌دلیل سازوکار مختصر، زمان زیادی برای بهینه‌سازی با فرکانس تغییر محیط (U) ۵۰۰۰ دارد و مقدار بهینه سراسری را با دقت بالایی می‌یابد، اما نکته حائز اهمیت آن است که ذرات آزاد مجبور به یافتن نواحی محتمل جدید برای ایجاد FSZ ها و بهبود بهینه‌ها توسط ذرات متمرکز جدید می‌شوند و این به معنی بار محاسباتی اضافه و صرف زمان بیشتر برای بهینه‌سازی است.

همان‌گونه که درقبل ذکر شد، درنظرگرفتن مقادیر بالا (بین ۶ تا ۱۰) موجب می‌شود، بهبود بهینه با سرعت کمتری صورت گیرد؛ درصورتی‌که بعد از تغییر در محیط احتمال حضور بهینه در همین FSZ بالا می‌رود. درعمل مقادیر بالا برای FSR به‌وسیله سازوکار *FSZoverlapChecking* تعدیل شده و به مقادیر متوسط کاهش می‌یابد.

با توجه به توصیفات اخیر، مقادیر بهینه پارامترهای idPSO برای حل مسأله MPB با مقادیر پیش‌فرض (p–10, U–5000, s–1) در جدول (۲) آورده شده است.

(جدول-۲): تنظیم‌های idPSO برای حل مسأله MPB پیش‌فرض

(Table-2): idPSO settings for solving the default MPB problem

Parameter	Value
Number of Free Particles	20
r_{conv}	0.5
P_{mean}	0.3
FSR	4

شکل (۳) به‌منظور تجسم رفتار ذرات در مراحل مختلف روند بهینه‌سازی برای یک محیط دوبعدی آورده شده است. در این شکل ذرات آزاد با علامت به‌علاوهٔ آبی، مکان قله‌ها با مربع مشکی، ذرات متمرکز با نقاط آبی، ناحیه جستجوی متمرکز با دوایر آبی و مراکز آن با ستارهٔ قرمزرنگ مشخص شده‌اند.

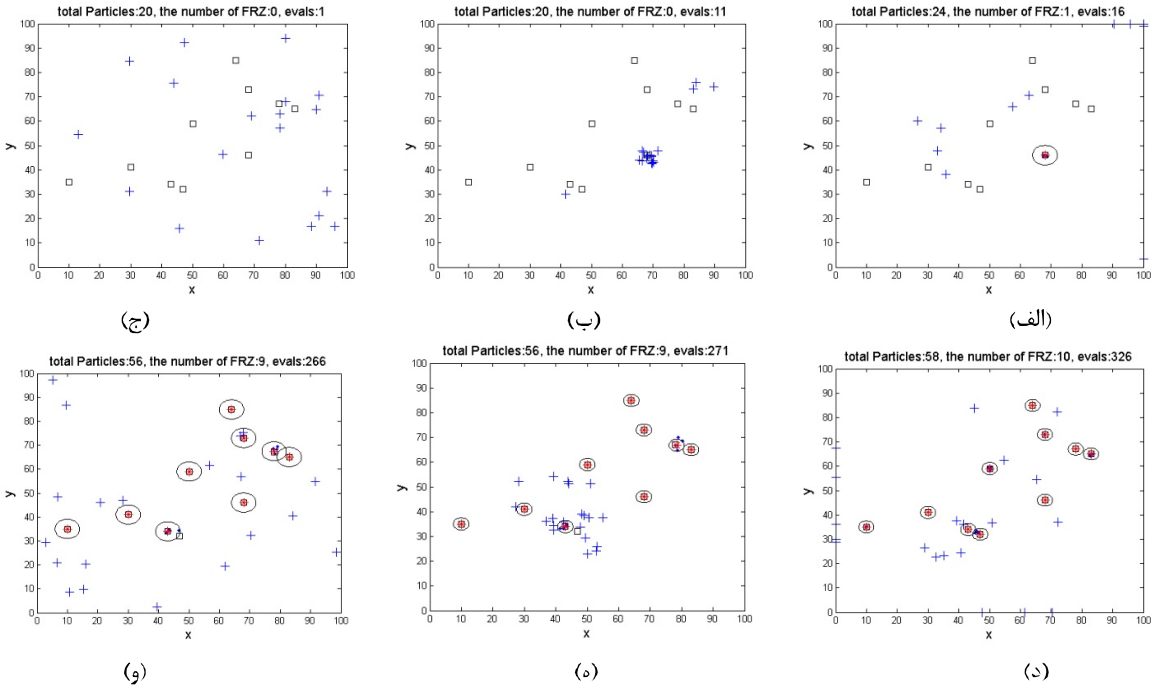
روند بهینه‌سازی از (الف) تا (و) نشان می‌دهد که الگوریتم با تعداد ذرات آزاد ۲۰ شروع به کار می‌کند (شکل (۳) الف) و با بررسی هم‌گرایی (شکل (۳) ب)، با یافت‌شدن نخستین منطقه محتمل در تکرار یازدهم ارزیابی برازندگی و ایجاد نخستین FSZ، ۴ ذره متمرکز به کل ذرات اضافه می‌شود (شکل (۳) ج) و پس از آن کلیه ۲۰ ذره آزاد پراکنده می‌شوند. همین روند برای یافتن تمامی مناطق محتمل و قله‌ها ادامه می‌یابد. در شکل (۳) حرکت مراکز FSZ در سیر تکامل الگوریتم به سمت قله‌ها نشان داده شده است. در تکرار ۳۲۵ تمامی ده قله توسط ذرات متمرکز احاطه شده و مرکز هر FSZ بر روی قله مورد نظر خود قرار گرفته است. همان‌گونه که در شکل (۳-د) مشخص است، دو FSZ که مجاور هم هستند، دچار هم‌پوشانی شده و سازوکار *FSZoverlap-Checking*، مقدار FSR را از ۴ به ۲/۴۵ کاهش می‌دهد. از این به بعد همان‌گونه که پیش‌تر بیان شد، سازوکارهای *convergenceChecking* و *RemoveUselessFSZ*، اجازه

[Downloaded from jsdp.rcisp.ac.ir on 2026-11-04]

[DOI: 10.29252/jsdp.16.4.27]

ایجاد FSZهای جدید را تا تغییر در شرایط محیطی و یا اضافه‌شدن قله نمی‌دهند.

با توجه مرحله اولیه تا مرحله نهایی در شکل (۳)، مشخص می‌شود که idPSO به‌صورت خودکار با کاهش و افزایش تعداد ذرات، سیر تکامل را طی کرده و کلیه مقادیر بهینه را پیدا می‌کند. در ادامه به مقایسه میزان کارآیی روش پیشنهادی با سایر روش‌ها پرداخته می‌شود.



(شکل-۳): روند یافتن مناطق محتمل توسط ذرات آزاد و قله‌ها توسط ذرات متمرکز در یک محیط دوبعدی

(Figure-3): The process of locating probable areas by free particles and peaks by focus particles in a two-dimensional environment

۳-۴- مقایسه idPSO با الگوریتم‌های دیگر

در این بخش از آزمایش‌ها، الگوریتم پیشنهادی با سایر الگوریتم‌های مطرح در این مبحث از جمله CPSO [22,23]، mCPSO [32]، mQSO [32]، CESO [45]، rSPSO [46]، SPSO [21]، AmQSO [47] ، mPSO [48]، APSO [49]، FTMPSO [34]، SFA [50]، PSO-AQ [51]، CDEPSO [52]، CbDE-wCA [53]، WD2O [35]، BfCS-wVN [36] و cGA [54] در حل مسأله MPB با تنظیم‌های مختلف مقایسه شده است. در ادامه نتایج پژوهش تغییر تعداد قله‌ها، شدت تغییرات و فرکانس تغییرات محیط U بر روی خطای برون‌خط الگوریتم idPSO آورده شده است.

۱. تغییر تعداد قله‌ها:

در مجموعه آزمایش‌هایی که در جدول (۳) جمع‌آوری شده است، چگونگی برخورد الگوریتم idPSO با تعداد قله‌های متفاوت در بازه یک تا دویست برای مسأله MPB نمایش داده شده است. مقایسه بر مبنای خطای برون‌خط و انحراف معیار با هفده الگوریتم دیگر که مقادیر برگرفته از مقالاتشان و با تنظیم‌های بهینه است، صورت گرفته است.

با توجه به جدول (۳)، نتایج برای الگوریتم پیشنهادی برای بیش‌تر تعداد قله‌ها بهتر از سایر الگوریتم‌ها در زمینه بهینه‌سازی پویا است. تنظیم‌های الگوریتم برای تمامی تعداد قله‌ها همان مقادیر تنظیم‌های پیش‌فرض بیان‌شده، است.

با دقت در مقادیر خطای برون‌خط idPSO و سایر الگوریتم‌ها در جدول (۳) متوجه می‌شویم که با افزایش تعداد قله‌ها مقدار خطای برون‌خط کاهش می‌یابد. درست است که کار الگوریتم در یافتن و دنبال‌کردن تعداد بالای قله‌ها مشکل‌تر می‌شود، اما منطقی است که با افزایش بهینه‌های محلی یافت‌شده احتمال نزدیک‌ترشدن مقدار برازندگی این بهینه‌ها به برازندگی بهینه سراسری افزایش می‌یابد. همچنین مناطق بیش‌تری در فضای جستجو توسط ذرات متمرکز تحت نظر و جستجو هستند که درنتیجه پس از تغییر در محیط امکان حضور بهینه سراسری در این مناطق بیش از پیش افزایش می‌یابد.

[Downloaded from jsdp.rcisp.ac.ir on 2026-11-04]
[DOI: 10.29252/jsdp.16.4.27]

۲. *تغییر حساسیت پویایی محیط (شدت تغییرات):*[1]

در جدول (۴) مقادیر خطای برون‌خط هشت الگوریتم در مقایسه با الگوریتم idPSO برای چهار شدت تغییر مختلف آورده شده است. تنظیم‌های محیط همان فرکانس تغییرات ۵۰۰۰ برای ده قله و شدت تغییر۱، ۲، ۳ و ۵ است. همان‌طورکه می‌دانیم با افزایش مقدار shift severity شدت تغییرات در محیط افزایش می‌یابد؛ به‌عبارت دیگر هر چه مقدار shift severity بیشتر افزایش یابد، بعد از تغییر در محیط مکان بهینه به فاصله دورتر جابه‌جا می‌شود. این قضیه کار الگوریتم را برای دنبال‌کردن بهینه بسیار سخت‌تر می‌کند؛ ازاین‌رو با توجه به جدول (۴) مقادیر خطای برون‌خط برای الگوریتم‌ها با افزایش مقدار shift severity افزایش می‌یابد. افزایش مقدار خطای برون‌خط برای الگوریتم پیشنهادی با تنظیم‌های بهینه الگوریتم که درقبل بیان شد، قابل چشم‌پوشی است و این نکته نشان از پایداری بالای الگوریتم در یافتن و دنبال‌کردن بهینه در هر مقدار شدت تغییر است.

(جدول-۳): مقایسه خطای آفلاین الگوریتم‌ها برای تعداد قله‌های متفاوت در مسأله MPB در فرکانس تغییرات U = 5000

(Table-3): Comparison of offline error algorithms for different peak numbers in the MPB problem at the change frequency of U=5000

Algorithm	Number of peaks, p							
	1	5	10	20	30	50	100	200
CPSO	0.14(0.11)	0.72(0.30)	1.06(0.24)	1.59(0.22)	1.58(0.17)	1.54(0.12)	1.41(0.08)	1.24(0.06)
mCPSO	4.93(0.17)	2.07(0.08)	2.08(0.07)	2.64(0.07)	2.63(0.08)	2.65(0.06)	2.49(0.04)	2.44(0.04)
mQSO(5,5q)	2.24(0.05)	1.82(0.08)	1.85(0.08)	2.48(0.09)	2.51(0.10)	2.53(0.08)	2.35(0.06)	2.24(0.05)
CESO	1.04(0.00)	-	1.38(0.02)	1.72(0.02)	1.24(0.01)	1.45(0.01)	1.28(0.02)	-
rSPSO	1.42(0.06)	1.04(0.03)	1.50(0.08)	2.20(0.07)	2.62(0.07)	2.72(0.08)	2.93(0.06)	2.79(0.05)
SPSO	2.64(0.10)	2.15(0.07)	2.51(0.09)	3.21(0.07)	3.64(0.07)	3.86(0.08)	4.01(0.07)	3.82(0.05)
AmQSO	2.62(0.10)	1.01(0.09)	1.51(0.10)	2.00(0.15)	2.19(0.17)	2.43(0.13)	2.68(0.12)	2.62(0.10)
mPSO	2.42(0.05)	1.82(0.08)	1.85(0.08)	2.48(0.09)	2.51(0.10)	2.53(0.08)	2.35(0.06)	2.24(0.05)
APSO	0.53(0.01)	1.05(0.06)	1.31(0.03)	1.69(0.05)	1.78(0.02)	1.95(0.02)	1.95(0.01)	1.90(0.01)
FTMPSO	0.18(0.01)	0.47(0.05)	0.67(0.04)	0.93(0.04)	1.14(0.04)	1.32(0.04)	1.61(0.03)	1.67(0.03)
SFA	0.42(0.03)	0.89(0.07)	1.05(0.04)	1.48(0.05)	1.56(0.06)	1.87(0.05)	2.01(0.04)	1.99(0.06)
PSO-AQ	0.34(0.02)	0.80(0.12)	0.89(0.03)	1.45(0.06)	1.52(0.04)	1.77(0.05)	1.95(0.05)	1.96(0.04)
CDEPSO	0.41(0.00)	0.97(0.01)	1.22(0.01)	1.54(0.01)	2.62(0.01)	2.20(0.01)	1.54(0.01)	2.11(0.01)
CbDE-wCA	0.14(0.03)	**0.30(0.02)**	0.86(0.08)	0.98(0.05)	1.34(0.04)	1.31(0.04)	1.35(0.03)	1.29(0.02)
WD2O	1.21(0.03)	0.76(0.003)	1.25(0.02)	1.22(0.01)	1.75(0.01)	1.87(0.01)	2.10(0.01)	1.95(0.01)
cGA	0.92(0.09)	1.06(0.07)	1.15(0.10)	1.18(0.06)	1.35(0.51)	1.65(0.07)	1.80(0.06)	1.71(0.05)
BfCS-wVN	0.30(0.06)	0.38(0.21)	**0.51(0.11)**	**0.74(0.12)**	1.00(0.11)	0.84(0.06)	1.11(0.06)	1.18(0.08)
idPSO	0.12(0.02)e-7	0.58(0.07)	0.62(0.05)	0.75(0.03)	0.67(0.04)	0.64(0.01)	0.25(0.02)	0.26(0.04)

(جدول-۴): مقایسه خطای برون‌خط الگوریتم‌ها برای مقادیر مختلف shift severity در مسأله MPB

(Table-4): Comparison of the offline error of algorithms for various shift Severity in MPB problem

Algorithm	Shift severity, s			
	1	2	3	5
CPSO	1.06(0.24)	1.17(0.22)	1.36(0.28)	1.58(0.32)
mCPSO	2.05(0.07)	2.80(0.07)	3.57(0.08)	4.89(0.11)
mQSO(5,5q)	1.85(0.08)	2.40(0.06)	3.00(0.06)	4.24(0.10)
CESO	1.38(0.02)	1.78(0.02)	2.03(0.03)	2.52(0.06)
rSPSO	1.50(0.08)	1.87(0.05)	2.40(0.08)	3.25(0.09)
SPSO	2.51(0.09)	3.78(0.09)	4.96(0.12)	6.76(0.15)
cGA	0.92(0.09)	2.19(0.15)	3.31(0.25)	6.45(0.45)
BfCS-wVN	**0.51(0.11)**	0.89(0.16)	1.26(0.13)	2.39(0.20)
idPSO	0.62(0.05)	0.67(0.07)	0.76(0.06)	0.91(0.09)

۳. *تغییر فرکانس تغییرات محیط:*

فرکانس تغییر محیط U درواقع زمان الگوریتم برای یافتن بهینه در هر محیط قبل از تغییر را مشخص می‌کند. واضح است که مقدار فرکانس کمتر یعنی زمان به‌مراتب کمتر برای وفق یافتن با محیط جدید و در طرف مقابل فرصت مناسب برای یافتن و بهبود بهترین بهینه متناظر با مقدار فرکانس بالا است. در جدول (۵) و جدول (۶) نتایج حاصل از حل مسأله MPB با تعداد قله‌های مختلف و فرکانس تغییر محیط هزار و ده‌هزار آمده است. به‌عنوان نمونه برای ده قله در فرکانس تغییر هزار خطای برون‌خط ۱/۶۱ به‌دست آمد؛ درصورتی‌که در فرکانس ده‌هزار الگوریتم زمان بیشتری داشته و خطای برون‌خط تا ۰/۳۳ کاهش می‌یابد.

۴-۴- به‌کارگیری روش پیشنهادی در ردیابی هدف متحرک

یکی از کاربردهای روش پیشنهادی در ردیابی هدف در فیلم است. تابع برازندگی با استفاده از معیار شباهت بین دو قاب تنظیم شده است که میزان شباهت هدف را با نامزد احتمالی

[1] shift severity

[Downloaded from jsdp.rcisp.ac.ir on 2026-11-04]

[DOI: 10.29252/jsdp.16.4.27]

اندازه گیری می‌کند و بیشترین شباهت را به‌عنوان محل هدف انتخاب می‌کند [55]. شباهت بین دو تصویر a و b به‌صورت زیر تعریف می‌شود:

$$\mathrm{SIM}(a,b) = \left(\frac{2\mu_a\mu_a}{\mu_a^2+\mu_b^2}\right)^{\alpha}\left(\frac{2\sigma_a\sigma_b}{\sigma_a^2+\sigma_b^2}\right)^{\beta}\left(\frac{\sigma_{ab}}{\sigma_a\sigma_b}\right)^{\gamma} \quad (۹)$$

که در آن μ میانگین:

$$\mu_a = \frac{1}{N}\sum_{i=1}^{N} a_i \quad (۱۰)$$

انحراف معیار:

$$\sigma_a = \left(\frac{1}{N-1}\sum_{i=1}^{N}(a_i-\mu_a)^2\right)^{\frac{1}{2}} \quad (۱۱)$$

σ_{ab} کواریانس بین دو تصویر:

$$\sigma_{ab} = \frac{1}{N-1}\sum_{i\ 1}^{N}(a_i-\mu_a)(b_i-\mu_b) \quad (۱۲)$$

سه مؤلفه شباهت از چپ، معیار نزدیکی روشنایی، وضوح و شباهت ساختاری بین دو تصویر است. توان‌های α، β و γ برای تنظیم اثر هر معیار روی مقدار پایانی شباهت استفاده شده است.

$$\alpha,\beta,\gamma \geq 0 \quad (۱۳)$$

معیار شباهت تعریف‌شده در معادله (۹) یک مرز بالایی یکتایی دارد: SIM(a,b)≤1 و اگر دو تصویر یکی باشند: SIM(a,b)=1. در این روابط پارامترهای α=1 =β=γ در نظر گرفته شده است.

همان‌طور که در شکل (۴) دیده می‌شود، الگوریتم پیشنهادی قابلیت پیداکردن هدف را در قاب‌های مختلف دارد. در این ویدئو فاصله بین قاب‌ها پنج قاب در نظر گرفته شده است تا حرکت هدف در تصویر محسوس باشد.



(شکل-۴): ردیابی هدف توسط روش پیشنهادی

(Figure-4): Target tracking by the proposed method

(جدول-۵): مقایسه خطای آفلاین الگوریتم‌ها برای تعداد قله‌های متفاوت در مسأله MPB در فرکانس تغییرات U = 1000

(Table-5): Comparison of offline error algorithms for different peak numbers in the MPB problem at the change frequency of U=1000

Algorithm	Number of peaks, p							
	1	5	10	20	30	50	100	200
mQSO(5,5q)	18.60(1.63)	6.56(0.38)	5.71(0.22)	5.85(0.15)	5.81(0.15)	5.87(0.13)	5.83(0.13)	5.54(0.11)
AmQSO	2.33(0.31)	2.90(0.32)	4.56(0.40)	5.36(0.47)	5.20(0.38)	6.06(0.14)	4.77(0.45)	5.75(0.26)
mPSO	4.44(0.02)	3.93(0.16)	4.57(0.18)	4.97(0.13)	5.15(0.12)	5.33(0.10)	5.60(0.09)	5.78(0.09)
APSO	2.72(0.04)	2.99(0.09)	3.87(0.08)	4.13(0.06)	4.12(0.04)	4.11(0.03)	4.26(0.04)	4.21(0.02)
FTMPSO	0.89(0.05)	1.70(0.10)	2.36(0.09)	3.01(0.12)	3.06(0.10)	3.29(0.10)	3.63(0.09)	3.74(0.09)
SFA	2.45(0.12)	2.71(0.06)	3.64(0.04)	4.01(0.07)	4.02(0.08)	4.12(0.07)	4.40(0.07)	4.43(0.07)
cGA	1.10(0.10)	**1.12(0.11)**	**1.28(0.13)**	1.76(0.09)	2.01(0.14)	2.56(0.10)	2.42(0.14)	2.20(0.11)
idPSO	0.21(0.02)	1.33(0.05)	1.61(0.03)	1.73(0.06)	1.70(0.04)	1.41(0.01)	0.92(0.03)	0.96(0.03)

(جدول-۶): مقایسه خطای آفلاین الگوریتم‌ها برای تعداد قله‌های متفاوت در مسأله MPB در فرکانس تغییرات U = 10000

(Table-6): Comparison of offline error algorithms for different peak numbers in the MPB problem at the change frequency of U=10000

Algorithm	Number of peaks, p							
	1	5	10	20	30	50	100	200
mQSO(5,5q)	1.90(0.18)	1.03(0.06)	1.10(0.07)	1.84(0.09)	2.00(0.09)	1.99(0.07)	1.85(0.05)	1.71(0.04)
AmQSO	0.19(0.02)	0.45(0.04)	0.76(0.06)	1.28(0.12)	1.78(0.09)	1.55(0.08)	1.89(0.14)	2.52(0.10)
mPSO	0.27(0.02)	0.70(0.10)	0.97(0.04)	1.34(0.08)	1.43(0.05)	1.47(0.04)	1.50(0.03)	1.48(0.02)
APSO	0.25(0.01)	0.57(0.03)	0.82(0.02)	1.23(0.02)	1.39(0.02)	1.46(0.01)	1.38(0.01)	1.36(0.01)
FTMPSO	0.09(0.00)	0.31(0.04)	0.43(0.03)	0.56(0.01)	0.69(0.09)	0.86(0.02)	1.08(0.03)	1.13(0.04)
SFA	0.26(0.03)	0.53(0.04)	0.72(0.02)	0.91(0.03)	0.99(0.04)	1.19(0.04)	1.44(0.04)	1.52(0.03)
BfCS-wVN	0.18(0.04)	0.20(0.02)	**0.30(0.06)**	0.84(0.05)	0.66(0.08)	0.54(0.70)	0.85(0.05)	0.85(0.05)
idPSO	0.3(0.04)e-12	0.12(0.03)	0.33(0.04)	0.44(0.07)	0.53(0.10)	0.43(0.05)	0.19(0.00)	0.19(0.00)

۴-۵- پیچیدگی محاسباتی و زمان اجرای الگوریتم

پیچیدگی محاسباتی الگوریتم idPSO به‌صورت زیر محاسبه می‌شود:

- زمان محاسبه برازندگی برای ذرات آزاد ($FreePartc$) و متمرکز ($FocusPartc$):

$$T_1 = O(FreePartc + FocusPartc)$$

- زمان محاسبه بررسی هم‌گرایی که قسمت عمده آن صرف مرتب‌سازی ذرات آزاد می‌شود:

$$T_2 = O(\log(FreePartc))$$

- زمان محاسبه بررسی هم‌پوشانی مراکز FSZها:

$$T_3 = O(\log(FocusPartc))$$

[Downloaded from jsdp.rcisp.ac.ir on 2026-11-04]

[DOI: 10.29252/jsdp.16.4.27]

- زمان مورد نیاز برای تشخیص و حذف FSZهای اضافی که صرف مرتب‌سازی برازندگی مراکز FSZها و مقایسه با P_{mean} می‌شود: $T_4 = O(\log(FocusPartc))$

پیچیدگی محاسباتی الگوریتم پیشنهادی به‌صورت $T_{total} = T_1 + T_2 + T_3 + T_4$ محاسبه می‌شود. درصورتی که M را مجموع کل ذرات در نظر بگیریم مرتبه پیچیدگی الگوریتم پیشنهادی به‌صورت $T_{total} = O(M)$ ساده می‌شود.

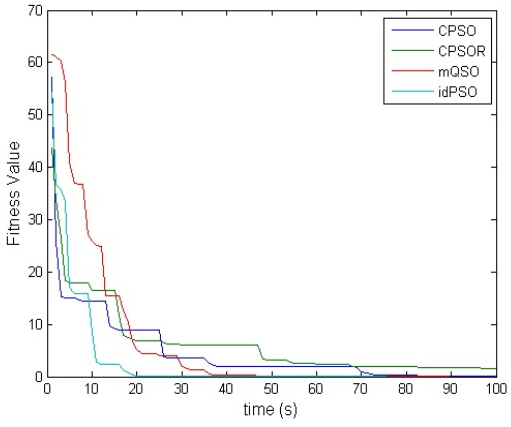
در جدول (۷) الگوریتم idPSO با سه الگوریتم CPSO، CPSOR و mQSO از نظر پیچیدگی محاسباتی مقایسه شده است (مقدار M اندازه جمعیت در الگوریتم‌ها است).

(جدول-۷): مقایسه پیچیدگی محاسباتی

(Table-7): Comparison of Computation Complexity

Algorithm	Computation Complexity
CPSO	$O(M^3)$
CPSOR	$O(M^3)$
mQSO(5,5q)	$O(M^3)$
idPSO	$O(M)$

در شکل (۵) زمان اجرای الگوریتم پیشنهادی آورده شده است. نتایج الگوریتم idPSO و سه الگوریتم دیگر برای یک‌بار تغییر در محیط پنج‌بعدی و تعداد قله ده در فرکانس تغییر هزار گردآوری شده است. نتایج شکل (۵) در کنار جدول (۵) نشان از سرعت و دقت الگوریتم در رسیدن به پاسخ بهینه‌تر است. الگوریتم با صرف زمان کمتر به مقدار خطای برون‌خط بهتری نسبت سایر الگوریتم‌ها دست یافته است.



(شکل-۵): مقایسه زمان اجرای الگوریتم‌ها برای مسأله MPB

(Figure-5): Comparison of run time algorithms for the MPB problem

۵- نتیجه‌گیری و بحث

در این مقاله الگوریتم نوینی برای بهینه‌سازی در محیط‌های پویا، غیرقطعی و پیچیده پیشنهاد شده است. این الگوریتم به دلیل سازوکار کوتاه و مختصر خود سرعت مناسبی برای بهینه‌سازی پویا دارد. ایده نوین اصلی الگوریتم پیشنهادی، انطباق همیشگی تعداد ذرات با شرایط محیطی برای یافتن و دنبال‌کردن بهینه است. در این الگوریتم ذرات به دو گروه ذرات آزاد و متمرکز تقسیم شده‌اند و توسط سازوکارهای نوینی روند ایجاد و حذف ذرات متمرکز کنترل می‌شود. همچنین الگوریتم idPSO به‌گونه‌ای طراحی شده است که نیاز به آشکارسازی تغییر در محیط را ندارد و همواره خود را با شرایط محیط وقف می‌دهد.

در بهینه‌سازی پویا کاستن از زمان بهینه‌سازی به‌عبارت دیگر هم‌گرایی سریع‌تر به بهینه در عین حفظ تنوع در کل فضای جستجو بسیار مهم است. بیش‌تر الگوریتم‌های پویا با تعداد بالای ذرات شروع به کار می‌کنند و با پیشرفت روند بهینه‌سازی از تعداد ذرات کاسته می‌شود. این قضیه به معنای صرف زمان بیهوده برای محاسبه برازندگی تعداد ذرات بالا در مراحل اولیه بهینه‌سازی است. برای جلوگیری از این دست محاسبات بیهوده الگوریتم idPSO روند نوینی را پیش گرفته است و با ذرات کم شروع به کار می‌کند و همواره افزایش تعداد ذرات تابع شرایط محیط ازجمله افزایش تعداد نقاط بهینه، است. در مقابل، با توجه به شرایط محیط الگوریتم اقدام به کاهش تعداد ذرات کرده که خود سبب کاهش بار محاسباتی و افزایش سرعت بهینه‌سازی می‌شود.

کارایی الگوریتم idPSO برای حل مسأله قله‌های متحرک (MPB) که یکی از معروف‌ترین توابع محک در محیط‌های پویا است، مورد بررسی قرار گرفته است. گستردگی آزمایش‌ها برای تنظیم‌های مختلف مسأله MPB اعم از تعداد قله‌های متفاوت، تغییر حساسیت پویایی محیط و فرکانس تغییر محیط متفاوت، نشان از عملکرد مناسب الگوریتم پیشنهادی در مقایسه با سایر الگوریتم‌های مطرح در زمینه بهینه‌سازی پویا دارد.

از جمله پیشنهادهای پژوهش‌های آینده می‌توان به به‌کارگیری الگوریتم برای بهینه‌سازی مسائل واقعی دنیا، اشاره کرد. همچنین به‌کارگیری الگوریتم در خوشه‌یابی[1] پویا مانند داده‌های web مد نظر است. به‌کارگیری سازوکارهای انطباقی[2] و خودانطباقی[3] برای پارامترهای ساختاری الگوریتم می‌تواند توانایی روش را برای انطباق سریع‌تر با شرایط محیطی در پی داشته باشد.

[1] Clustering
[2] Adaptive
[3] Self adaptive

[Downloaded from jsdp.rcisp.ac.ir on 2026-11-04]
[DOI: 10.29252/jsdp.16.4.27]

6- References ۶- مراجع

[1] M. Mavrovouniotis, Ch. Li and S. Yang, "A survey of swarm intelligence for dynamic optimization: Algorithm and application", *Swarm and Evolutionary Computation*, Vol. 33, pp. 1-17, 2017.

[2] T. T. Nguyen, S. Yang, and J. Branke, "Evolutionary dynamic optimization: A survey of the state of the art", *Swarm and Evolutionary Computation*, Vol. 6, pp. 1-24, 2012.

[3] J. Euchi, A. Yassine, H. Chabchoub, "The dynamic vehicle routing problem: Solution with hybrid metaheuristic approach", *Swarm and Evol. Comput*, Vol. 21, pp. 41–53, 2015.

[4] Y. E. Demirtas¸, E. ¨Ozdemir, U. Demirtas¸, "A particle swarm optimization for the dynamic vehicle routing problem", in Modeling, Simulation, and Applied Optimization (ICMSAO), 6th International Conference on, 2015, pp. 1–5.

[5] M. Mavrovouniotis, S. Yang, "Applying ant colony optimization to dynamic binary-encoded problem", in A. Mora, G. Squillero (Eds.), Evo-Applications: Applications of Evolutionary Computation, Vol. 9028 of Lecture Notes in Computer Science, Springer International Publishing, 2015, pp. 845–856.

[6] U. Boryczka, Ł. Stra¸k, "Diversification and entropy improvement on the DPSO algorithm for DTSP", in T. N. Nguyen, B. Trawi´nski, R. Kosala (Eds.), Intelligent Information and Database Systems: 7th Asian Conference, ACIIDS 2015, Bali, Indonesia, Proceedings, Part I, Springer International Publishing, Cham, 2015, pp. 337–347.

[7] M.Mavrovouniotis, S. Yang, "Ant colony optimization with self-adaptive evaporation rate in dynamic environments", *in IEEE Symposium on Computational Intelligence in Dynamic and Uncertain Environments (CIDUE)*, pp. 47–54, 2014.

[8] S. Gao, Y. Wang, J. Cheng, Y. Inazumi, Z. Tang, "Ant colony optimization with clustering for solving the dynamic location routing problem", Appl. Math. Comput., Vol. 285, pp. 149–173, 2016.

[9] M. Mavrovouniotis, S. Yang, "Ant colony optimization with memorybased immigrants for the dynamic vehicle routing problem", *in 2012 IEEE Congress on Evolutionary Computation (CEC)*, pp. 2645– 2652, 2012.

[10] M. Mavrovouniotis, S. Yang, "Ant colony optimization with immigrants schemes for the dynamic travelling salesman problem with traffic factors", *Appl. Soft Comput.*, Vol. 13, pp. 4023–4037, 2013.

[11] M. Mavrovouniotis, S. Yang, X. Yao, "Multi-colony ant algorithms for the dynamic travelling salesman problem", *in IEEE Symposium on Computational Intelligence in Dynamic and Uncertain Environments (CIDUE)*, pp. 9–16, 2014.

[12] B. van Veen, M. Emmerich, Z. Yang, T. B¨ack, J. Kok, "Ant colony algorithms for the dynamic vehicle routing problem with time windows", in J. M. Ferr´andez Vicente, J. R. ´Alvarez S´anchez, F. de la Paz L´opez, F. J. Toledo Moreo (Eds.), Natural and Artificial Computation in Engineering and Medical Applications: 5th International Work-Conference on the Interplay Between Natural and Artificial Computation, IWINAC 2013, Mallorca, Spain, Proceedings, Part II, Springer Berlin Heidelberg, Berlin, Heidelberg, 2013, pp. 1–10.

[13] Z. Yang, M. Emmerich, T. B¨ack, "Ant based solver for dynamic vehicle routing problem with time windows and multiple priorities", in 2015 IEEE Congress on Evolutionary Computation (CEC), 2015, pp. 2813– 2819.

[14] L. Melo, F. Pereira, E. Costa, "Multi-caste ant colony algorithm for the dynamic traveling salesperson problem", in M. Tomassini, A. Antonioni, F. Daolio, P. Buesser (Eds.), Adaptive and Natural Computing Algorithms, Vol. 7824 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2013, pp. 179–188.

[15] L. Melo, F. Pereira, E. Costa, "Extended experiments with ant colony optimization with heterogeneous ants for large dynamic traveling salesperson problems", in 14th International Conference on Computational Science and Its Applications (ICCSA), 2014, pp. 171–175.

[16] U. Boryczka, Ł. Stra¸k, "Heterogeneous DPSO algorithm for DTSP", in M. N´u˜nez, T. N. Nguyen, D. Camacho, B. Trawi´nski (Eds.), Computational Collective Intelligence: 7th International Conference, ICCCI 2015, Madrid, Spain, Proceedings, Part II, Springer International Publishing, Cham, 2015, pp. 119–128.

[17] M. Okulewicz, J. Ma´ndziuk, "Two-phase multi-swarm PSO and the dynamic vehicle routing problem", *in IEEE Symposium on Computational Intelligence for Human-like Intelligence (CIHLI)*, pp. 1–8, 2014.

[18] M. Mavrovouniotis, S. Yang, "Dynamic vehicle routing: A memetic ant colony optimization approach", in A. Uyar, E. Ozcan, N. Urquhart (Eds.), Automated Scheduling and Planning, Vol. 505 of Studies in Computational Intelligence, Springer Berlin Heidelberg, 2013, pp. 283– 301.

[19] M. Mavrovouniotis, F. M. M¨uller, S. Yang, "Ant colony optimization with local search for the dynamic travelling salesman problems", *IEEE Trans. Cybern*, Vol. 99, pp. 1–14, 2016.

[20] L. Liu and S. R. Ranjithan, "An adaptive optimization technique for dynamic envir-

[Downloaded from jsdp.rcisp.ac.ir on 2026-11-04]

[DOI: 10.29252/jsdp.16.4.27]

onments", Eng. Appl. Artif. Intell., Vol. 23, No. 5, pp. 772– 779, 2010.

[21] D. Parrott and X. Li, "Locating and tracking multiple dynamic optima by a particle swarm model using speciation", *IEEE Trans. Evol. Comput.,* Vol. 10, No. 4, pp. 440–458, 2006.

[22] S. Yang and C. Li, "A clustering particle swarm optimizer for locating and tracking multiple optima in dynamic environments", *IEEE Trans. Evol. Comput.*, Vol. 14, No. 6, pp. 959–974, 2010.

[23] C. Li and S. Yang, "A clustering particle swarm optimizer for dynamic optimization", *in Congr. Evol. Comput.*, pp. 439–446, 2009.

[24] J. Branke, The Moving Peaks Benchmark, 1999.Available:http://web.archive.org/web/20130906140931/http:/people.aifb.kit.edu/jbr/MovPeaks/

[25] J. Branke, "Memory enhanced evolutionary algorithms for changing optimization problems", *in Congr. Evol. Comput.*, Vol. 3, 1999, pp. 1875–1882.

[26] J. Yaochu and J. Branke, "Evolutionary optimization in uncertain environments-a survey", *IEEE Transactions on Evolutionary Computation*, Vol. 9, pp. 303-317, 2005.

[27] M. Khouadjia, E. Alba, L. Jourdan, E.-G. Talbi, "Multi-swarm optimization for dynamic combinatorial problems: A case study on dynamic vehicle routing problem", in M. Dorigo, M. Birattari, G. Di Caro, R. Doursat, A. Engelbrecht, D. Floreano, L. Gambardella, R. Groß, E. S¸ahin, H. Sayama, T. St¨utzle (Eds.), Swarm Intelligence, Vol. 6234 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2010, pp. 227– 238.

[28] J. Branke, T. Kaußler, C. Schmidth, and H. Schmeck, "A multipopulation approach to dynamic optimization problem", in 4th Int. Conf. Adaptive Comput. Des. Manuf., 2000 pp. 299–308.

[29] C. Li and S. Yang, "Fast multi-swarm optimization for dynamic optimization problems", in 4th Int. Conf. Natural Comput., Vol. 7, 2008, pp. 624–628.

[30] M. Kamosi, A. B. Hashemi, and M. R. Meybodi, "A hibernating multiswarm optimization algorithm for dynamic environments", in World Congr. on Nature and Biologically Inspired Computing, NaBIC-2010, 2010, pp. 363–369.

[31] T. M. Blackwell and J. Branke, "Multi-swarm optimization in dynamic environments", in EvoWorkshops 2004: Appl. Evol. Comput., LNCS 3005, 2004, pp. 489–500.

[32] T. M. Blackwell and J. Branke, "Multi-swarms, exclusion, and anticonvergence in dynamic environments", IEEE Trans. Evol. Comput., Vol. 10, No. 4, pp. 459–472, 2006.

[33] I. del Amo, D. Pelta, Gonz´aez, and J. lez, "Using heuristic rules to enhance a multi-swarm pso for dynamic environments", in 2010 Congr. Evol. Comput., 2010, pp. 1–8.

[34] D. Yazdani, B. Nasiri, A. Sepas-Moghaddam, and M. R. Meybodi, "A novel multi-swarm algorithm for optimization in dynamic environments based on particle swarm optimization", *Applied Soft Computing,* Vol. 13, pp. 2144–2158, 2013.

[35] A. Boulesnane, S. Meshoul, "WD2O: a novel wind driven dynamic optimization approach with effective change detection," Applied Intelligence, Vol. 47, pp. 488–504, September 2017.

[36] J. Kazemi Kordestani, H. Abedi Firouzjaee, M. R. Meybodi, "An adaptive bi-flight cuckoo search with variable nests for continuous dynamic optimization problems," *Applied Intelligence*, Vol. 48, pp. 97–117, January 2018.

[37] N. Fouladgar, S. Lotfi, "A novel approach for optimization in dynamic environments based on modified cuckoo search algorithm," Soft Comput, pp 1–15, 2015.

[38] C. Li, S. Yang, "A general framework of multipopulation methods with clustering in undetectable dynamic environments", IEEE Trans. Evol. Comput, Vol. 16, No. 4, pp. 556–577, 2012.

[39] R. I. Lung and D. Dumitrescu, "Evolutionary swarm cooperative optimization in dynamic environments", Natural Comput., Nol. 9, No. 1, pp. 83–94, 2010.

[40] H. Richter, "Detecting change in dynamic fitness landscapes", in 2009 Congr. Evol. Comput., 2009, pp. 1613–1620.

[41] A. Simoes and E. Costa, "Evolutionary algorithms for dynamic environments: prediction using linear regression and markov chains", *in Parallel Problem Solving from Nature*, pp. 306–315, 2008.

[42] R. C. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory", in Proc. 6th Int. Symp. Micro Mach. Human Sci., 1995, pp. 39–43.

[43] J. Kennedy and R. C. Eberhart, "Particle swarm optimization", in Proc. IEEE Int. Conf. Neural Netw., 1995, pp. 1942–1948.

[Downloaded from jsdp.rcisp.ac.ir on 2026-11-04]

[DOI: 10.29252/jsdp.16.4.27]

[44] Y. Shi, R. Eberhart, "A modified particle swarm optimizer", in Proc. IEEE Conf. Evol. Comput., 1998, pp. 69–73.

[45] R. I. Lung and D. Dumitrescu, "A collaborative model for tracking optima in dynamic environments", in Proc. Congr. Evol. Comput., 2007, pp. 564–567.

[46] S. Bird and X. Li, "Using regression to improve local convergence", in Proc. IEEE Congr. Evol. Comput., 2007, pp. 592–599.

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

[48] M. Kamosi, A. Hashemi, and M. Meybodi, "A New Particle Swarm Optimization Algorithm for Dynamic Environments", *Swarm Evolutionary and Memetic Com-puting*, pp. 129-138, 2010.

[49] I. Rezazadeh, M. Meybodi, and A. Naebi, "Adaptive particle swarm optimization algorithm for dynamic environments", *Advances in Swarm Intelligence*, pp. 120-129, 2011.

[50] B. Nasiri and M. Meybodi, "Speciation based firefly algorithm for optimization in dynamic environments", *International Journal of Artificial Intelligence*, Vol. 8, pp. 118-132, 2012.

[51] D. Yazdani, B. Nasiri, R. Azizi, A. Sepas-Moghaddam, and M. R. Meybodi, "Optimization in Dynamic Environments Utilizing a Novel Method Based on Particle Swarm Optimization", *International Journal of Artificial Intelligence*, Vol. 11, pp. 170-192, 2013.

[52] 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.

[53] R. Mukherjee, G. R. Patra, R. Kundu, and S. Das, "Cluster-based differential evolution with Crowding Archive for niching in dynamic environments", Information Sciences, Vol. 267, pp. 58- 82, 2014.

[54] M. Mohammadpour, H. Parvin, M. Sina, "Chaotic Genetic Algorithm based on Explicit Memory with a new Strategy for Updating and Retrieval of Memory in Dynamic Environments," AI and Data Mining, Vol. 6, pp. 191-205, Winter and Spring 2018.

[55] E. Wang, Z. Bovik, A. Sheik, H. Simoncelli, "Image quality assessment: from error visibility to structural similarity", *IEEE Transactions on Image processing*, 2004.



سیدمسعود اجابتی دوره کارشناسی و کارشناسی ارشد خود را در رشته مهندسی برق-الکترونیک به‌ترتیب در سال ۱۳۸۹ از دانشگاه گیلان و سال ۱۳۹۲ از دانشگاه بیرجند اخذ کرده و در حال حاضر نیز دانشجوی دکترای مهندسی برق-الکترونیک دانشگاه بیرجند است. از جمله زمینه‌های پژوهشی و علاقه‌مندی ایشان به بازشناسی الگو، بهینه‌سازی پویا، الگوریتم‌های تکاملی و پردازش سیگنال می‌توان اشاره کرد.

نشانی رایانامه ایشان عبارت است از:

ejabati_masoud@birjand.ac.ir



سیدحمید ظهیری ممقانی مدارک کارشناسی و کارشناسی ارشد خود را به‌ترتیب در رشته مهندسی برق-الکترونیک از دانشگاه صنعتی شریف و تربیت مدرس دریافت کردند. ایشان در سال ۱۳۸۴ مدرک دکترای خود را در رشته مهندسی-برق الکترونیک در دانشگاه فردوسی مشهد به اتمام رساندند. وی در حال حاضر استاد تمام گروه برق-الکترونیک دانشکده مهندسی برق و کامپیوتر دانشگاه بیرجند هستند. زمینه‌های مورد علاقه ایشان، روش‌های بهینه‌سازی تکاملی، روش‌های بهینه‌سازی هوش جمعی، محاسبات نرم، تشخیص الگو، پردازش تصویر و پردازش سیگنال است.

نشانی رایانامه ایشان عبارت است از:

hzahiri@birjand.ac.ir

[Downloaded from jsdp.rcisp.ac.ir on 2026-11-04]

[DOI: 10.29252/jsdp.16.4.27]

[DOI: 10.29252/jsdp.16.4.27] [Downloaded from jsdp.rcisp.ac.ir on 2026-11-04]