

# یک روش بهینه‌سازی ترکیبی (EO + PSO)

فرشاد جعفریه

محمد رضا میبیدی

زهرا گلشنی

دانشکده مهندسی برق و فناوری اطلاعات    دانشکده مهندسی برق و فناوری اطلاعات    دانشکده مهندسی برق و فناوری اطلاعات

دانشگاه آزاد اسلامی قزوین

دانشگاه صنعتی امیرکبیر

دانشگاه آزاد اسلامی قزوین

[jafarieh\\_farshad@yahoo.com](mailto:jafarieh_farshad@yahoo.com)

[mmeybodi@aut.ac.ir](mailto:mmeybodi@aut.ac.ir)

[golshani\\_zahra@yahoo.com](mailto:golshani_zahra@yahoo.com)

**چکیده:** الگوریتم بهینه‌سازی حدی با پارامتر  $\tau$  برای مسائلی که بیش از یک نقطه بهینه دارند به خوبی عمل می‌کند ولی از آنجایی که این الگوریتم یک الگوریتم جستجوی محلی است، نسبت به مقدار دهی اولیه حساس است. با این حال، این الگوریتم این ویژگی مهم را دارد که در صورت قرار گرفتن در اطراف نقطه بهینه، جواب را با دقت بالایی محاسبه می‌کند. از طرف دیگر مطالعات بر روی الگوریتم PSO نشان می‌دهند که این الگوریتم با وجود جستجوی مناسب دامنه مساله، جواب‌هایی با دقت بالا تولید نمی‌کند و با احتمال بالایی در نقاط بهینه محلی گرفتار می‌شود. در این بررسی روشهای پیشنهادی مختلفی از الگوریتم بهینه‌سازی ترکیبی که آنها را EPSO، IEPSO، و با پارامتر  $\tau$  می‌نامیم معرفی شده اند که از ترکیب الگوریتم بهینه‌سازی حدی، الگوریتم PSO و PSO با وزن جبری پویا حاصل می‌شوند پیشنهاد می‌گردد. آزمایش‌ها نشان می‌دهد که الگوریتمهای پیشنهادی ویژگی‌های مطلوب هر دو الگوریتم پایه خود را دارا می‌باشند. در ادامه نتایج حاصل از پیاده‌سازی این چهار الگوریتم بر روی شش تابع معیار مختلف مورد بررسی قرار گرفته است. نتایج بدست آمده کارایی الگوریتمهای پیشنهادی را تایید می‌کند.

**کلمات کلیدی:** بهینه‌سازی حدی، PSO، بهینه‌سازی

## ۱. مقدمه

بهینه‌سازی حدی و الگوریتم‌های مشتق شده از آن، دسته جدیدی از الگوریتم‌ها هستند که برای حل مسائل بهینه‌سازی مورد استفاده قرار می‌گیرند.

الگوریتم‌های مبتنی بر بهینه‌سازی حدی<sup>۱</sup> بر خلاف دیگر الگوریتم‌های بهینه‌سازی نظیر الگوریتم ژنتیک که همواره از مجموعه‌ای از جواب‌ها (جمعیت) تشکیل شده‌اند، در هر لحظه

تنها یک جواب در اختیار دارند و در هر مرحله این جواب را بهبود می‌بخشند؛ بهبود جواب با انتخاب یکی از اجزای جواب و جایگزین کردن مقدار آن با مقداری جدید انجام می‌شود. برای انتخاب جزء تعویضی، الگوریتم از شایستگی محلی اجزا استفاده می‌کند؛ بنابراین این الگوریتم‌ها علاوه بر تابع شایستگی کلی جواب، به تابعی برای ارزیابی شایستگی اجزای جواب (شایستگی محلی) نیز احتیاج دارند. واضح است که این تابع وابسته به نوع مساله بوده و برای هر مساله باید به صورت جداگانه تعریف و پیاده‌سازی شود.

در این مقاله برای حل این مشکل پیشنهاد می‌شود که الگوریتم بهینه‌سازی حدی در کنار الگوریتم بهینه‌سازی دیگری به نام PSO قرار گیرد تا در شرایطی که به تنهایی از کارایی بالایی برخوردار نیست بتواند با کمک گرفتن از PSO عملکرد خود را بهبود بخشد. علت این پیشنهاد این است که بنا بر آزمایش‌های انجام شده بر روی الگوریتم PSO، مشاهده شده است که در این الگوریتم بر خلاف الگوریتم بهینه‌سازی حدی امکان جستجوی دامنه مساله به شکل مطلوبی محیاست، اما احتمال گیر افتادن آن در مینیمم محلی بالاست. در واقع نقاط قوت هر کدام از این دو الگوریتم منطبق بر نقطه ضعف‌های الگوریتم دیگر است. بنابراین انتظار می‌رود که در صورت ترکیب این دو الگوریتم بتوان ضعف‌های هر دو الگوریتم را مرتفع نمود.

## ۲. بهینه‌سازی حدی

بهینه‌سازی حدی [۳] الگوریتمی اکتشافی است که از مدل یک-اسپن الهام گرفته شده است.

مشخص‌ترین تفاوت الگوریتم بهینه‌سازی حدی و دیگر الگوریتم‌های اکتشافی نظیر الگوریتم ژنتیک، نیاز الگوریتم به دانستن شایستگی سلول‌ها (شایستگی محلی) علاوه بر شایستگی کلی جواب است. در واقع این ویژگی الگوریتم بهینه‌سازی حدی،

<sup>1</sup> Extremal Optimization

هسته مرکزی و عامل کار آن است. الگوریتم بهینه‌سازی حدی به صورت کلی شکل زیر را دارد:

۱. جوابی به عنوان جواب اولیه تولید شده و به عنوان جواب جاری و همچنین به عنوان بهترین جواب در نظر گرفته می‌شود.
۲. بدترین سلول جواب جاری انتخاب شده و مقدار آن با مقدار دیگری جایگزین می‌شود.
۳. جواب ایجاد شده به عنوان جواب جاری در نظر گرفته می‌شود.
۴. در صورتی که جواب جاری بهتر از بهترین جواب باشد به عنوان بهترین جواب نیز انتخاب می‌شود.
۵. مرحله ۲ تا رسیدن به جواب مورد نیاز تکرار می‌شود.
۶. بهترین جواب برگردانده می‌شود.

الگوریتم بهینه‌سازی حدی برای حل مسائلی نظیر تقسیم گراف به دو زیر گراف، رنگ آمیزی گراف با سه رنگ و فروشنده دوره‌گرد مورد استفاده قرار گرفته و جواب‌های مطلوبی ارائه داده است.

## ۲-۱. بهینه‌سازی حدی با پارامتر $\tau$

هرچند الگوریتم بهینه‌سازی حدی جواب‌های مطلوبی برای مسائل به کار گرفته شده ارائه می‌دهد، با این حال احتمال گرفتار شدن آن در نقاط بهینه محلی و متوقف شدن آن در مکانی به جز جواب مساله زیاد است. برای حل این مشکل پارامتر جدیدی به نام  $\tau$  به الگوریتم بهینه‌سازی حدی اضافه شده و الگوریتم جدیدی به نام بهینه‌سازی حدی با پارامتر  $\tau$  [۴] معرفی شده است؛ تفاوت این الگوریتم جدید و الگوریتم بهینه‌سازی حدی به مرحله انتخاب سلول تعویضی مربوط می‌شود. در این الگوریتم برای پیدا کردن سلول تعویضی، ابتدا سلول‌ها بر اساس شایستگی و به صورت صعودی مرتب می‌شوند. سپس به هر سلول مقداری برابر  $P_k$  که از رابطه ۲ قابل محاسبه است اختصاص داده می‌شود؛ در این رابطه  $k$  برابر ترتیب سلول در فهرست مرتب شده است:

$$P_k \propto k^{-\tau} \quad (2)$$

پس از نسبت دادن مقادیر  $P_k$  به سلول‌ها، سلول تعویضی به این‌صورت انتخاب می‌شود که:

۱. در حلقه‌ای به طور اتفاقی یکی از سلول‌ها انتخاب می‌شود.

۲. عددی به صورت اتفاقی بین ۰ و ۱ انتخاب می‌شود.

۳. در صورتی که عدد تولید شده کوچکتر از  $P_k$  سلول باشد سلول به عنوان سلول تعویضی انتخاب می‌شود و حلقه پایان می‌پذیرد، در غیر این‌صورت حلقه تکرار می‌شود.

مراحل کلی الگوریتم بهینه‌سازی حدی با پارامتر  $\tau$  به شرح زیر است:

۱. جواب اولیه‌ای تولید شده به عنوان جواب جاری و همچنین به عنوان بهترین جواب در نظر گرفته می‌شود.
  ۲. سلول‌ها با توجه به شایستگی آنها به صورت صعودی مرتب می‌شوند.
  ۳. با توجه به رابطه ۲ یک سلول به عنوان سلول تعویضی انتخاب و مقداردهی مجدد می‌شود.
  ۴. جواب تولید شده به عنوان جواب جاری در نظر گرفته می‌شود.
  ۵. در صورتی که جواب جاری بهتر از بهترین جواب باشد به عنوان بهترین جواب نیز انتخاب می‌شود.
  ۶. مرحله ۲ تا رسیدن به جواب مورد نیاز تکرار می‌شود.
  ۷. بهترین جواب برگردانده می‌شود.
- لازم به ذکر است که الگوریتم ارائه شده و نحوه انتخاب سلول تعویضی ساده‌ترین شکل پیاده‌سازی بهینه‌سازی حدی با پارامتر  $\tau$  بوده و بسته به نوع مساله قابل تغییر خواهد بود.

## ۳. الگوریتم PSO

PSO یک روش بهینه‌سازی مبتنی بر جمعیت است. PSO مفهوم ساده‌ای دارد و داری پارامترهای کمی است. PSO به راحتی قابل پیاده‌سازی است و برای توابع غیرخطی پیوسته، شبکه‌های عصبی، مسائل بهینه‌سازی غیرخطی، کنترل سیستم فازی و ... با موفقیت به کار برده شده است [1,2].

### ۳-۱ الگوریتم PSO استاندارد

در PSO هر راه حل تنها یک "پرنده" در فضای جستجو است و "عضو" نامیده می‌شود. تمام پرندگان یک مقدار شایستگی دارند که توسط تابع شایستگی که باید بهینه شود ارزیابی می‌گردد. علاوه بر این هر پرنده  $i$  دارای یک موقعیت در فضای  $D$  بعدی مسئله است که در تکرار  $t$ ام، با یک بردار به صورت زیر نمایش داده می‌شود:

$$X_i^t = (x_{i1}^t, x_{i2}^t, \dots, x_{iD}^t)$$

همچنین این پرنده سرعتی دارد که پروازش را هدایت می‌کند و در تکرار  $t$ ام با بردار زیر نشان داده می‌شود:

$$V_i^t = (v_{i1}^t, v_{i2}^t, \dots, v_{iD}^t)$$

و این پرنده نیز در هر تکرار یک حافظه از بهترین موقعیت قبلی خودش را دارد که با بردار  $P$  نشان داده می‌شود:

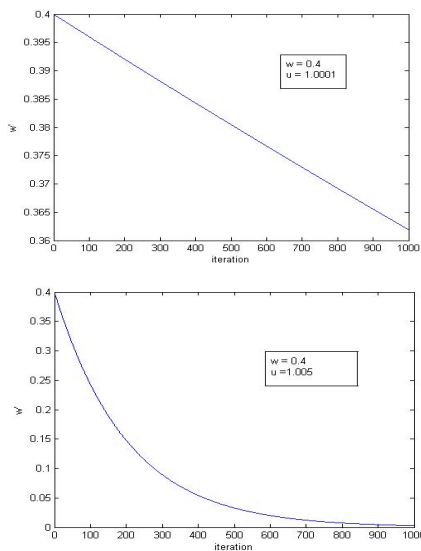
$$P_i^t = (p_{i1}^t, p_{i2}^t, \dots, p_{iD}^t)$$

در هر تکرار جستجو، هر عضو با در نظر داشتن دو مقدار "بهترین" به روز رسانی می‌شود. اولی مربوط به بهترین راه حلی است که پرنده تا کنون آن را تجربه کرده است. (مقدار شایستگی این بهترین راه حل نیز ذخیره می‌گردد). این مقدار را "بهترین

یک فضای جستجوی  $D$  بعدی و  $m$  عضو را در نظر بگیرید. بردارهای موقعیت، سرعت،  $pbest$  و  $gbest$  در این الگوریتم همانند الگوریتم استاندارد PSO است. تفاوت PSO با IPSO در فرمول های به روزرسانی سرعت و موقعیت است که در IPSO به صورت زیر می باشد.

$$w \in [0,1] \quad u \in [1.00001, 1.005]$$

و مابقی ضرایب همانند PSO استاندارد است. برای درک بهتر نحوه تغییرات وزن جبری پویا در IPSO، نمودار تغییرات  $w'$  برای ۱۰۰۰ نسل و به ازای  $w = 0.4$ ،  $u = 1.0001$  و  $u = 1.005$  رسم شده است. شکل ۱ این نمودار را نشان می دهد.



شکل (۱): نحوه تغییرات وزن جبری پویا در الگوریتم IPSO

به ازای  $w = 0.4$ ،  $u = 1.0001$  و  $u = 1.005$

#### ۴. الگوریتمهای پیشنهادی

همانطور که قبلاً اشاره شد، بهینه سازی حدی دامنه مساله را به شکل خوبی جستجو نکرده و از آنجایی که این الگوریتم یک الگوریتم جستجوی محلی است باعث شده که استفاده از این الگوریتم تنها به مجموعه خاصی از مسائل محدود شود. ولی این الگوریتم این ویژگی مهم را دارد که در صورت قرار گرفتن در اطراف نقطه بهینه، جواب را با دقت بالایی پیدا می کند. از طرف دیگر نتایج گزارش شده برای الگوریتم PSO نشان می دهد که این الگوریتم با وجود جستجوی مناسب دامنه مساله، امکان گیرافتادن آن در مینیمم محلی زیاد است.

#### ۴-۱ الگوریتم پیشنهادی اول (EPSO) :

"p" یا اصطلاحاً " $pbest$ " می نامند. دومین بهترین که توسط PSO دنبال می شود بهترین موقعیتی است که تا کنون در جمعیت به دست آمده است. این مقدار بهینه عمومی است و اصطلاحاً " $gbest$ " نامیده می شود. زمانیکه یک عضو، بخشی از جمعیت را به عنوان توپولوژی همسایگانش در نظر می گیرد، بهترین مقدار یک بهترین محلی است و " $lbest$ " نامیده می شود. بعد از اینکه دو بهترین مقدار پیدا شدند موقعیت و سرعت هر عضو توسط دو فرمول زیر به روزرسانی می شوند:

$$V_i(t+1) = wV_i(t) + c_1r_{1,i}(t)(P_i(t) - X_i(t)) + c_2r_{2,i}(t)(P_g(t) - X_i(t))$$

$$X_i(t+1) = X_i(t) + V_i(t+1)$$

$P_i$  بهترین موقعیت قبلی عضو  $i$ ام است ( $pbest$ ). مطابق تعاریف مختلف  $P_g$ ، دو نسخه متفاوت از PSO وجود دارد. اگر  $P_g$ ، بهترین موقعیت در میان تمام اعضای جمعیت باشد، چنین نسخه ای را نسخه عمومی می نامند. اگر  $P_g$  از میان تعداد محدودی از اعضای جمعیت ( $lbest$ ) گرفته شود، چنین نسخه ای را نسخه محلی می نامند.  $P_i$  و  $P_g$  به ترتیب با دنبال کردن معادلات زیر محاسبه می شوند:

$$P_i(t) = \begin{cases} P_i(t) & \text{if } f(X_i(t)) \geq f(P_i(t)) \\ X_i(t) & \text{if } f(X_i(t)) < f(P_i(t)) \end{cases}$$

$P_g \in \{P_1, P_2, \dots, P_m\} \mid f(P_g) = \min(f(P_1), f(P_2), \dots, f(P_m))$  در فرمول های فوق  $f$  تابع شایستگی،  $t$  بیانگر شماره تکرار، و متغیرهای  $c_1, c_2$  فاکتورهای یادگیری هستند. اغلب  $c_1 = c_2 = 2$  است که میزان جابجایی یک پرده را در یکبار تکرار کنترل می کند.  $r_1, r_2$  دو عدد تصادفی یکنواخت در رنج  $[0,1]$  هستند.  $w$  یک وزن جبری است که به صورت نوعی در رنج  $[0,1]$  مقداردهی اولیه می گردد. یک وزن جبری بزرگتر یک استکشاف عمومی و وزن جبری کوچکتر استکشاف محلی را تسهیل می نماید.

#### ۳-۲ الگوریتم PSO بهبود یافته (IPSO)

در الگوریتم استاندارد PSO، اطلاعات بهترین موقعیت هر فرد و بهترین عمومی، با نسل بعد اعضا به اشتراک گذاشته می شود. در این بررسی یک الگوریتم بهبود یافته PSO ارائه شده است که از یک وزن درونی پویا استفاده می کند که با افزایش تکرار نسلها، کاهش می یابد.

$$V_i(t+1) = wV_i(t) + c_1r_{1i}(t)(P_i(t) - X_i(t)) + c_2r_{2i}(t)(P_g(t) - X_i(t))$$

$$X_i(t+1) = X_i(t) + V_i(t+1)$$

همان طور که مشاهده می شود تفاوت این الگوریتم با الگوریتم PSO استاندارد محاسبه مقدار شایستگی محلی هر بعد و تغییر آن بجای تغییر همه ابعاد یک عضو می باشد، که این همان ایده اصلی الگوریتم بهینه سازی حدی می باشد.

#### ۴-۲ الگوریتم پیشنهادی دوم (EPSTO)

هرچند الگوریتم EPSTO جواب های مطلوبی برای مسائل به کار گرفته شده ارائه می دهد، با این حال احتمال گرفتار شدن آن در نقاط بهینه محلی و متوقف شدن آن در مکانی به جز جواب مساله زیاد است. برای حل این مشکل پارامتر جدیدی به نام  $\tau$  به این الگوریتم اضافه شده و الگوریتم جدیدی به نام EPSTO با پارامتر  $\tau$  معرفی شده است؛ تفاوت این الگوریتم جدید و الگوریتم EPSTO به مرحله انتخاب سلول تعویضی مربوط می شود. در این الگوریتم برای پیدا کردن سلول تعویضی، ابتدا سلول ها بر اساس شایستگی و به صورت صعودی مرتب می شوند. سپس به هر سلول مقداری برابر  $P_k$  که از رابطه ۲ قابل محاسبه است اختصاص داده می شود؛ در این رابطه  $k$  برابر ترتیب سلول در فهرست مرتب شده است.

در این الگوریتم با توجه به این موضوع که در بحث انتخاب، بجای انتخاب بر اساس شایسته سالاری، انتخاب متناسب با شایستگی صورت می گیرد، بنابراین با اضافه شدن پارامتر  $\tau$ ، الگوریتم این امکان را می یابد تا بدترین بعد را حذف نکند، بلکه طوری رفتار کند که احتمال حذف بدترین بعد بیشتر باشد. با این کار امکان فرار از مینیمم های محلی فراهم خواهد شد. در مورد این الگوریتم می توان گفت این الگوریتم در مواجهه با هر عضو از مجموعه، شایستگی محلی هر بعد یک عضو را محاسبه کرده و پس از نسبت دادن مقادیر  $P_k$  به سلول ها، سلول تعویضی به اینصورت انتخاب می شود که:

۱. در حلقه ای به طور اتفاقی یکی از سلول ها انتخاب می شود.
۲. عددی به صورت اتفاقی بین ۰ و ۱ انتخاب می شود.
۳. در صورتی که عدد تولید شده کوچکتر از  $P_k$  سلول باشد سلول به عنوان سلول تعویضی انتخاب می شود و حلقه پایان می پذیرد، در غیر اینصورت حلقه تکرار می شود.
- مراحل کلی الگوریتم EPSTO با پارامتر  $\tau$  به شرح زیر است:

۱. شایستگی محلی هر بعد محاسبه می شود.
۲. سلول ها با توجه به شایستگی آنها به صورت صعودی مرتب می شوند.
۳. با توجه به رابطه ۲ یک سلول به عنوان سلول تعویضی انتخاب و بعد سرعت مربوط به آن مقداردهی مجدد می شود.

در این بخش یک الگوریتم ترکیبی به نام EPSTO که دارای مزایای هر دو الگوریتم بهینه سازی حدی و PSO می باشد پیشنهاد می گردد. در الگوریتم PSO که در این مقاله استفاده شده است، همسایه یک عضو شامل تمامی اعضای جمعیت می شود، بنابراین از نسخه عمومی این الگوریتم استفاده شده است در الگوریتم پیشنهادی ابتدا جمعیت و سرعت با مقادیر تصادفی مقدار دهی اولیه می شوند. سپس در هر بار تکرار الگوریتم که باید مقادیر سرعت و موقعیت توسط فرمولهای اشاره شده در بخش ۳-۱ بروز شوند از ایده اصلی الگوریتم بهینه سازی حدی استفاده می شود، به این صورت که تمام ابعاد یک عضو تغییر نمی کنند بلکه با توجه به شایستگی محلی هر بعد، بعدی از سرعت و موقعیت بروز می شود که بدترین تاثیر را در موقعیت فعلی داشته باشد. به بیان دیگر در این الگوریتم به تعداد اعضا از بهینه سازی حدی استفاده شده است، و این در حالی است که این بهره جستن از بهینه سازی حدی تعداد تکرارها را افزایش نمی دهد و به واسطه این عمل سر باری نخواهد داشت.

در الگوریتم پیشنهادی، وظیفه جستجوی دامنه مساله به عهده PSO و وظیفه دقیقتر شدن در جواب ها به عهده الگوریتم بهینه سازی حدی است. به بیان ساده تر، در الگوریتم پیشنهادی ابتدا فضای مساله توسط PSO جستجو می شود و هر جا احتمال حضور جواب وجود داشته باشد بهینه سازی حدی وارد عمل شده و محل مورد نظر را برای یافتن جواب بهتر با دقت بیشتری جستجو می کند. سپس، الگوریتم PSO مجدداً کنترل را بر عهده گرفته و جستجو ادامه پیدا می کند. لازم به ذکر است که جستجوی دامنه توسط PSO قابل جایگزینی با جستجوی اتفاقی نیست، زیرا PSO به شکلی هدفمند فضا را جستجو کرده و الگوریتم بهینه سازی حدی را به شکل بهتری به سمت نقاط بهینه هدایت می کند. مراحل کلی الگوریتم EPSTO به شرح زیر است.

#### الگوریتم EPSTO

۱. مقداردهی اولیه جمعیت با موقعیتها و سرعتهای تصادفی

۲. محاسبه مقدار شایستگی تمام اعضا

۳. ست کردن  $pbest$  و مقدار شایستگی اش برای هر عضو

$$P_i(t+1) = \begin{cases} P_i(t) & \text{if } f(\vec{X}_i(t+1)) \geq f(\vec{P}_i(t)) \\ X_i(t+1) & \text{if } f(\vec{X}_i(t+1)) < f(\vec{P}_i(t)) \end{cases}$$

۴. ست کردن  $gbest$  و مقدار شایستگی اش

$$P_g \in \{P_1, P_2, \dots, P_m\} | f(P_g) = \min(f(P_1), f(P_2), \dots, f(P_m))$$

۵. محاسبه مقدار شایستگی محلی برای هر بعد

۶. به روز رسانی موقعیت و سرعت بدترین بعد هر عضو

۷. در صورت برقراری شرط توقف دادن  $gbest$  و مقدار

شایستگی اش به عنوان خروجی و در غیر این صورت

برو به مرحله ۲

جدول (۲): نحوه عملکرد الگوریتم های پیشنهادی برای روی تابع schwefel

Algorithm	Best Result	Average
PSO	0.009	0.1449
EPSO	2.2121e-10	3.821415e-10
EPSO + $\tau$	1.03e-14	1.39e-13
IPSO*	39.02	282.981
IEPSO	2.231233e-16	4.1466e-16
IEPSO + $\tau$	6.49e-14	1.49e-12

جدول (۳): نحوه عملکرد الگوریتم های پیشنهادی برای روی تابع Ackley

Algorithm	Best Result	Average
PSO	0.089	7.44
EPSO	1.7e-14	2.46e-13
EPSO + $\tau$	2.36e-9	3.17e-9
IPSO	0.02	3.20
IEPSO	2.33e-16	2.31e-15
IEPSO + $\tau$	1.47e-14	1.11e-13

جدول (۴): نحوه عملکرد الگوریتم های پیشنهادی برای روی تابع Griewank

Algorithm	Best Result	Average
PSO	1.14	7.77
EPSO	2.86e-14	3.12e-14
EPSO + $\tau$	1.14e-14	2.83e-13
IPSO	2.15	14.36
IEPSO	1.2e-15	6.15e-15
IEPSO + $\tau$	*	9.7e-19
IEPSO + $\tau$ *	0	2.81e-22

## ۶. نتیجه گیری

در این مقاله یک الگوریتم بهینه سازی ابر اکتشافی ترکیبی که از ترکیب الگوریتم های بهینه سازی حدی و PSO حاصل شده است معرفی گردید. این الگوریتم ترکیبی مزایای هر دو الگوریتم بهینه سازی حدی و PSO را دارا می باشد. نتایج آزمایش ها بر روی مسایل مختلف نشان داد که الگوریتم های پیشنهادی هم از دقت الگوریتم بهینه سازی حدی و هم از قابلیت جستجوی بالای الگوریتم PSO برخوردار است.

## مراجع

- [1] B. Jiao, Z. Lian, X. Gu, "A dynamic inertia weight particle swarm optimization algorithm," ELSEVIER Chaos, Solitons and Fractals (2006)
- [2] R. C. Eberhart and Y. Shi, "Guest Editorial Special Issue on Particle Swarm Optimization," IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, VOL. 8, NO. 3, JUNE 2004.
- [3] Wolfram, S., "Universality and complexity in cellular automata", Physica D, no. 10, pp. 1-35, January 1984.
- [4] Narendra K. S. and Thathachar M.A.L., "Learning Automata: An Introduction", Prentice Hall, 1989.
- [5] Beigy, H. and Meybodi, M. R., "A Mathematical Framework for Cellular Learning Automata", Advances on Complex Systems, Vol. 7, Nos. 3-4, pp. 295-320, September/December 2004

۴. جواب تولید شده به عنوان جواب جاری در نظر گرفته می شود.  
۵. در صورتی که جواب جاری بهتر از  $P_i$  عضو باشد به عنوان  $P_i$  جواب نیز انتخاب می شود.

## ۳-۴ الگوریتم پیشنهادی سوم (IEPSO)

این الگوریتم از ترکیب الگوریتم IPSO و EO ایجاد شده است. چنانچه در هر مرحله از الگوریتم EPSO وزن ها را بصورت پویا تغییر دهیم بدین ترتیب که

$$w' = w * u^{-k} \quad w \in [0,1]$$

شوند، خواهیم توانست دقت الگوریتم را در رسیدن به جواب در تعداد تکرار برابر نسبت به الگوریتم EPSO بالاتر ببریم.  
نحوه بروز رسانی سرعت و موقعیت بدترین عضو با توجه به الگوریتم IPSO خواهد بود

## ۴-۴ الگوریتم پیشنهادی چهارم (IEPSO+ $\tau$ )

تفاوت این متد با روش قبلی در نحوه انتخاب بعد ذره جهت بروزرسانی می باشد که در این حالت با در نظر گرفتن ایده متناسب با شایستگی می توان از گیر افتادن در مینیمم های محلی اجتناب کرد.

## ۵. نتایج آزمایش ها

در این بخش ابتدا به بررسی رفتار الگوریتم های PSO و IPSO و مقایسه آن با رفتارهای الگوریتم های پیشنهادی می پردازیم. سپس نتایج اجرای این الگوریتم ها بر روی چند مساله بهینه سازی ارائه می شود.

در تابعی همچون schwefel با توجه به نحوه کارکرد الگوریتم های PSO و IPSO در مواجهه با مینیمم های محلی بر می آید، که در این چنین توابعی استفاده از این الگوریتم ها باعث همگرایی زود رس شده و سریعاً در مینیمم محلی قرار خواهد گرفت. در صورتیکه استفاده از الگوریتم های پیشنهادی علاوه بر اینکه کار کرد عضو بد را اصلاح میکند و از خاصیت حرکت جمعی برخوردار است میتواند با تکیه بر خصوصیت فرار از مینیمم محلی که در روش های دوم و چهارم تعبیه شده است نتایجی با دقت بسیار بالایی تولید نمود.

در زیر به نتایج بدست آمده اشاره شده است:

جدول (۱): نحوه عملکرد الگوریتم های پیشنهادی برای روی تابع sphere

Algorithm	Best Result	Average
PSO	5.98	145.444
EPSO	1.170389e-10	2.18820055e-9
EPSO + $\tau$	3.33e-9	8.41e-9
IPSO*	32.17	478.135
IEPSO	1.025803e-16	3.9826027e-16
IEPSO + $\tau$	1.36e-15	3.19e-14