بهینه‌سازی چندمدی با الگوریتم جستجوی گرانشی به همراه تکنیک جایگاه‌یابی k-means و نخبه‌گرایی loop in Loop

محمد نورمحمدی زرده‌سوار، شهرام گلزاری، امین موسوی

چکیده

بشر برای حل مسائل خود، همواره به دنبال راه‌حلی بوده که هزینه کمتری داشته باشد. از این‌رو مسائل بهینه‌سازی، توجه محققان را به خود جلب نموده‌اند. از مهم‌ترین روش‌های برخورد با این مسائل، الگوریتم‌های تکاملی هستند که بیشتر آن‌ها از طبیعت الهام گرفته شده‌اند. الگوریتم جستجوی گرانشی یکی از الگوریتم‌های تکاملی می‌باشد که در برخورد با مسائل تک‌مدی کارایی خود را نشان داده است. برای موفقیت این الگویتم در مسائل چندمدی، آن را با یکی از تکنیک‌های جایگاه‌یابی به نام K-means و تکنیک نخبه‌گرایی جدید loop in Loop ترکیب نموده‌ایم. کارایی این الگوریتم ترکیبی در بخش نتایج آزمایشگاهی مشاهده می‌شود.

کلمات کلیدی

جستجوی گرانشی، چند مدی، جایگاه‌یابی، K-means

۱- مقدمه

بهینه‌سازی یک مسأله، یافتن بهترین جواب یا جواب‌ها در بین تمامی راه‌حل‌های ممکن برای آن مسأله می‌باشد. مسائلی که به دنبال چندین راه‌حل بهینه هستند، مسائل چندمدی[1] نامیده شده و در مقابل آن مسائل تک‌مدی[2] که فقط به دنبال یافتن تک جواب هستند وجود دارند. در دهه‌های اخیر، توجه بیشتری به الگوریتم‌هایی که از پدیده‌های طبیعی استفاده می‌نمایند شده است. الگوریتم‌های تکاملی برخلاف عملکرد مناسب در برخورد با مسائل تک‌مدی، در مسائل چندمدی چندان موفق نبوده‌اند. از این‌رو تکنیک‌های جایگاه‌یابی[3] جهت کمک به الگوریتم‌های تکاملی در مسائل چندمدی ارائه گردیدند. یکی از الگوریتم‌های تکاملی کارا جهت یافتن بهینه‌ها، الگوریتم جستجوی گرانشی[4] [1,2] است. در این پژوهش، از الگوریتم جستجوی گرانشی به همراه تکنیک جایگاه‌یابی K-means و تکنیک جدید نخبه‌گرایی[5] به نام loop in Loop، برای حل مسائل چندمدی استفاده گردیده است. از دلایل انتخاب الگوریتم جستجوی گرانشی، یافتن بهینه‌های موجود با تکرار[6] کمتر و فرار از بهینه‌های محلی و همچنین وابستگی کمتر به پارامترها می‌باشد. از میان تکنیک‌های خوشه‌بندی[7]، K-means به دلیل سادگی، موثر بودن و پیچیدگی زمانی کم برای تقسیم جمعیت اصلی به زیر جمعیت‌ها برگزیده شده است؛ به این ترتیب که اعضاء هر زیرجمعیت، همدیگر را در یافتن بهینه‌ها یاری می‌رسانند.

در طول تاریخ سعی بر این بوده است که نسل‌های جدید، نسبت به نسل‌های گذشته قوی‌تر باشند. در گیاهان و حیوانات هم با اصلاح نژاد سعی در ایجاد نسل‌های بعدی قوی‌تر شده است. این ایده به درون الگوریتم‌های مکاشفه‌ای تحت عنوان نخبه‌گرایی رخنه نموده و باعث تسریع همگرایی این‌گونه الگوریتم‌ها گردیده است. NichePSO [3]، CBN [4]، SPSO [5]، MGPSO [6]، r2PSO [7]، r3PSO [7]، r2PSO-lhc [7]، r3PSO-lhc [7]، FER-PSO [8]، Deterministic Crowding (DC) [9] و Sequential Niche (SN) [10]، NGSA [11] برخی از الگوریتم‌های مطرح برای حل مسائل چند مدی می‌باشند.. اکثر این الگوریتم‌ها، به پارامترها بسیار وابسته می‌باشند. در این پژوهش سعی گردیده با حذف این پارامترها و بالا بردن دقت در ارزیابی‌های کمتر، الگوریتمی موفق ارائه شود. در ادامه مفاهیم پایه به همراه الگوریتم پیشنهادی تشریح و نتایج قابل قبولی گزارش گردیده است.

۲- مفاهیم پایه

۲-۱- الگوریتم جستجوی گرانشی

الگوریتم جستجوی گرانشی توسط راشدی و همکارانش بر اساس قانون گرانش نیوتن، ارائه گردید. در الگوریتم جستجوی گرانشی، عامل‌ها به صورت اجسامی هستند که یکدیگر را با نیروی گرانشی جذب می‌نمایند و این نیرو

موجب حرکت همه اجسام به سمت اجسام برآزنده‌تر می‌شود. سیستمی با N عامل را درنظر بگیرید. کل سیستم را با فرمول (۱) نمایش می‌دهیم. همچنین موقعیت و مکان ذره iام به صورت فرمول (۲) تعریف می‌شود.

$$X = (X_1, X_2, \ldots, X_i, \ldots, X_N) \quad (۱)$$

$$X_i = (x_i^1, x_i^2, \ldots, x_i^d, \ldots, x_i^n) \quad (۲)$$

x_i^d موقعیت عامل iام در بعد dام برای عامل X_j است و n تعداد بعد فضای جستجو است. همچنین N همان تعداد عامل‌ها است. در زمان معین t، نیروی اعمالی بر جرم i از سمت جرم j از فرمول (۳) محاسبه می شود.

$$F_{ij}^d = G(t)\frac{M_{pi}(t) \times M_{aj}(t)}{R_{ij}(t) + \varepsilon}(x_j^d(t) - x_i^d(t)) \quad (۳)$$

$$R_{ij}(t) = \|X_i(t), X_j(t)\| \quad (۴)$$

جایی که M_{aj} جرم گرانشی فعال جرم jام، M_{pi} جرم گرانشی غیر فعال جرم iام، $G(t)$ ثابت گرانشی در زمان t، ε ثابت کوچک و R_{ij} فاصله اقلیدسی بین دو عامل i و j می‌باشد که با فرمول (۴) نشان داده شده است.

کل نیروی وارد شده به عامل i در بعد d از سوی K تا از بهترین عامل‌ها از فرمول (۵) محاسبه می‌شود.

$$F_i^d(t) = \sum_{j \epsilon Kbest, j \neq i} rand_j F_{ij}^d \quad (۵)$$

جایی که $rand_j$ عددی تصادفی در بازه [0 1] می‌باشد. Kbest مجموعه‌ای از K عامل با بهترین مقدار برآزندگی و تابعی از زمان است که مقدار اولیه‌اش در شروع برابر K_0 و با گذشت زمان کاهش می‌یابد.

شتاب عامل iام در زمان t در بعد d، از فرمول (۶) محاسبه می‌شود.

$$a_i^d(t) = \frac{F_i^d(t)}{M_{ii}(t)} \quad (۶)$$

M_{ii} جرم اینرسی عامل i می‌باشد. همچنین سرعت و موقعیت عامل در زمان t+1 از فرمول‌های (۷و۸) محاسبه می‌شود.

$$v_i^d(t+1) = rand_i \times v_i^d(t) + a_i^d(t) \quad (۷)$$

$$x_i^d(t+1) = x_i^d(t) + v_i^d(t) \quad (۸)$$

جایی که $rand_i$ عددی تصادفی در بازه [0 1] است.

ثابت گرانشی G تابعی از زمان است که با مقدار اولیه G_0 شروع و برای کنترل دقت جستجو با گذشت زمان کاهش می‌یابد. از این رو مقدار G از فرمول‌های (۹و۱۰) محاسبه می‌شود.

$$G(t) = (G_0, t) \quad (۹)$$

$$G(t) = G_0 e^{\frac{-\alpha t}{T}} \quad (۱۰)$$

جایی که α و G_0 مقداری ثابت هستند و T نشان دهنده کل تکرارها می‌باشد. همچنین جرم‌های اینرسی و گرانشی با استفاده از فرمول‌های (۱۱-۱۳) به روز می‌شوند.

$$M_{ai} = M_{pi} = M_{ii} = M_i \qquad i = 1, 2, \ldots, N \quad (۱۱)$$

$$m_i(t) = \frac{fit_i(t) - worst(t)}{best(t) - worst(t)} \quad (۱۲)$$

$$M_i(t) = \frac{m_i(t)}{\sum_{j=1}^{N} m_j(t)} \quad (۱۳)$$

در جایی که $fit_i(t)$ مقدار برآزندگی عامل iام را در زمان t نشان می‌دهد. worst(t) و best(t) برای مسأله ماکزیمم‌سازی از فرمول‌های (۱۴و۱۵) محاسبه می‌شود.

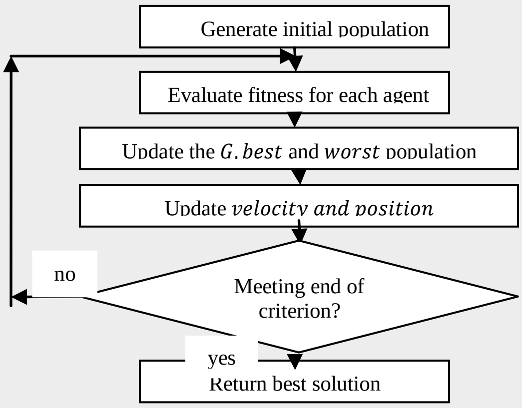
$$best(t) = \max_{j \in \{1,\ldots,N\}} fit_j(t) \quad (۱۴)$$

$$worst(t) = \min_{j \in \{1,\ldots,N\}} fit_j(t) \quad (۱۵)$$

فلوچارت الگوریتم جستجوی گرانشی در شکل (۱) نشان داده شده است.

۲-۲- جایگاه‌یابی

در طبیعت همه جانداران به دنبال یافتن منابع غذایی می‌باشند. هر جاندار در نزدیک منابع غذایی خود زندگی می‌کند. از این رو طبیعت تقسیم‌بندی می‌شود. معمولا در هر قسمت جانورانی از یک گونه وجود دارند که برای یافتن منابع غذایی با یکدیگر رقابت می‌کنند. از این فعل و انفعال طبیعت، مفهوم جایگاه‌یابی به ذهن محققان خطور کرد و از آن به عنوان راهی برای غلبه بر مسائل چند-مدی استفاده گردید. یکی از روش‌های مطرح جایگاه‌یابی، خوشه‌بندی داده‌ها با الگوریتم K-means می‌باشد. در این روش به تعداد خوشه‌هایی که می‌خواهیم تشکیل دهیم، دانه تصادفی تعیین می‌کنیم. هر عضو از جمعیت به نزدیکترین دانه اختصاص می‌یابد. حال وقتی اعضاء هر خوشه مشخص شد دانه‌های هر خوشه به مرکز خوشه منتقل می‌شوند و این کار تا هنگامی که دانه‌ها تغییر حرکتی نداشته باشند تکرار می‌شود.



شکل (۱) : الگوریتم جستجوی گرانشی [2]

۳- الگوریتم پیشنهادی (KGSA[8])

۳-۱- تولید خوشه¬های مناسب از جمعیت

در این پژوهش، از دو روش تصادفی و Partition [11] برای مقداردهی اولیه جمعیت استفاده می‌شود. خوشه نامناسب در فاز مقداردهی اولیه، خوشه‌ای است که یا تک عضوی یا تهی و یا در سایر مراحل الگوریتم تهی باشد. از این‌رو اگر خوشه‌ای نامناسب باشد اگر در فاز مقداردهی اولیه باشیم مدام مقداردهی اولیه را رد نموده و دوباره مقداردهی نموده و این اعضای جدید را توسط K-means خوشه‌بندی می‌نماییم تا بالاخره به هر خوشه تعداد اعضای مناسبی اختصاص یابد. ولی اگر در سایر مراحل الگوریتم باشیم، فقط خوشه‌بندی را تکرار می‌نماییم تا اینکه هیچ کدام از خوشه‌ها تهی نباشند.

۳-۲- محاسبه جرم، نیرو، سرعت و ایجاد نسل بعدی

برای محاسبه جرم هر عضو از جمعیت، ابتدا همسایه‌های هر فرد از جمعیت مشخص می‌گردند. همسایه‌های هر فرد، افرادی از جمعیت هستند که در خوشه آن فرد قرار گرفته‌اند. در ادامه با استفاده از فرمول‌های (۱۱-۱۳) مقدار جرم برای هر عضو از جمعیت مشخص می‌شود و سپس طبق فرمول (۵)، مجموع برآیند نیروهای وارده به جرم i مشخص می‌گردد. نیروهای وارده به

یک عضو، فقط از سوی همسایگانش می‌باشد. البته برای تسریع بخشیدن به همگرایی، فقط نیروی ۷۰ درصد نفرات هر خوشه با اولویت برازندگی به هر عضو آن خوشه وارد می‌–شود تا اعضاء حرکت نمایند. می‌توان سرعت حرکت اعضاء را از فرمول (۷) محاسبه نمود. در اثر حرکت اعضاء، جمعیتی جدید ایجاد شده که با ترکیب با نسل قبلی و انتخاب برازنده‌ترها نسل بعدی را تشکیل می‌دهند. نحوه انتخاب اعضای نسل آینده بر اساس [12] می‌باشد. اصلاح جمعیت در هر نسل طبق [12] برای توابع مقید[9] استفاده شده تا از ایجاد راه‌حل‌های ناممکن اجتناب شود.

۳-۳- loop in Loop و کشف بهینه‌ها

در روش loop in Loop، جمعیت به تعداد T_l تکرار، تکامل می‌یابد. بعد از پایان هر T_l، جمعیت جدیدی را جایگزین می‌نماییم که مقداردهی اولیه می‌گردد. این نسل نیز T_l بار تکامل می‌یابد و نسل جدیدی دوباره ایجاد می‌گردد. این روال تکراری، آن‌قدر ادامه می‌یابد که تعداد کل تکرارها به پایان برسد و یا به تعداد ماکزیمم ارزیابی مجاز رسیده باشیم. از این پس به هر کدام از این قسمت‌ها یک loop و به جایگاه‌هایی که بعد از پایان هر loop پیدا می‌شوند جایگاه‌های بالقوه می‌گوییم. اگر هدف پیدا کردن فقط بهینه‌های سراسری باشد فقط آن اعضایی برای روال انتخاب جایگاه‌های بالقوه، باقی می‌مانند که از مقدار ماکزیمم حداکثر فاصله مجاز را رعایت نموده باشند. البته این در صورتی است که تعداد جایگاه‌های باقیمانده از تعداد بهینه‌هایی که باید کشف شوند بیشتر باشد. چون اعضاء جمعیت بعد از هر loop به نزدیکی جایگاه‌های خود رسیده‌اند، نزدیک‌ترین فاصله بین دو عضو از جمعیت را پیدا نموده و هر کدام برازندگی بهتری دارند باقی مانده و دیگری حذف می‌گردد. این روال تکرار می‌شود تا به تعداد جایگاه‌های برسیم که باید یافت شوند. اعضاء باقیمانده، همان جایگاه‌های بالقوه می‌باشند. در پایان هر T_l، جایگاه‌های بالقوه به انبار جایگاه‌های بالقوه اضافه می‌گردند. و همچنین بهترین برازنده هر loop به همراه آن‌هایی که تا ۸۰ درصد برازندگی بهترین فرد را دارند، به انبار برازنده‌ها اضافه می‌شوند تا به همراه انبار جایگاه‌های بالقوه به عنوان کاندیدا برای ورود به loop بعدی آماده شوند. بعد از پایان هر loop جمعیت جدیدی مقداردهی اولیه شده و کاندیداها با نزدیک‌ترین عضو از جمعیت جدید به رقابت پرداخته و اگر دارای برازندگی بهتری باشند جایگزین آن می‌شوند. به این صورت جمعیت اولیه برای loop بعدی شکل می‌گیرد. این روال تکراری با انباشته شدن کاندیداها برای loopهای بعدی ادامه می‌یابد. لازم به ذکر است در عبور از یک نسل به نسل بعدی نیز جایگاه‌های هر نسل با مقایسه برازندگی آن‌ها با نزدیکترین عضو از نسل بعد به نسل بعدی اضافه می‌گردند. جایگاه‌های بین هر نسل به انبار جایگاه‌های موقت اضافه گردیده و به عنوان کاندیداهایی برای ورود به نسل‌های بعدی به همراه انبار جایگاه‌ها آماده می‌شوند. در پایان هر T_l، سرعت اعضاء صفر می‌شود و همچنین انبار موقت بین نسلی هم خالی می‌شود، چرا که در آن ممکن است جایگاه‌های مناسبی وجود نداشته باشند. در پایان برای کشف جایگاه‌های اصلی، بعد از اینکه loopها به پایان رسیدند از میان جایگاه‌های بالقوه و لیست برازنده‌ها به همان روشی که جایگاه‌های بالقوه یافت شدند جایگاه‌های اصلی را پیدا می‌نماییم.

در الگوریتم پیشنهاد شده، تعداد نسل‌های تولید شده ثابت و به تعداد T می‌باشد. اما برای الگوریتم K-means، شرط توقف موقعی است که مرکز خوشه‌ها دیگر تغییر نمی‌کند.

٤- نتایج آزمایشگاهی

٤-۱- توابع ارزیابی مقید و نامقید

در توابع نامقید[10] اگر دامنه این مسائل را قید ننامیم، راه‌حل‌ها از قانون خاصی تبعیت نمی‌کنند ولی در توابع مقید راه‌حل‌ها در چهارچوب خاصی می‌باشند. جدول (۱) توابع نامقید و جدول (۲) توابع مقید مورد استفاده در این پژوهش را نشان می‌دهد. دامنه این توابع مطابق [11] می‌باشد.

٤-۲- معیارهای ارزیابی الگوریتم‌های خوشه‌بندی

برای ارزیابی الگوریتم پیشنهادی از سه معیار نرخ موفقیت، تعداد ارزیابی مورد نیاز برای کشف بهینه‌ها و نرخ خطا استفاده گردیده است

در هر اجرا، اگر تمامی بهینه‌ها یافت شدند آن اجرا موفق، در غیر این‌صورت آن اجرا ناموفق است. پیک وقتی یافت شده است که اختلاف مقدار برازندگیش از برازندگی پیک واقعی از خطای قابل قبول بیشتر نباشد. نرخ موفقیت، حاصل تقسیم تعداد اجراهای موفق به تعداد کل اجراها می‌باشد.

در هر نسلی اگر الگوریتم جایگاه‌ها را پیدا نموده باشد، از لحظه شروع الگوریتم تا آن لحظه که جایگاه‌ها پیدا شده‌اند، تعداد دفعاتی که تابع ارزیابی فراخوانی شده یعنی تعداد اعضایی که برازندگی‌شان محاسبه شده را برای هر اجرا محاسبه نموده و از آن در نهایت میانگین‌گیری می‌نماییم.

اختلاف بهینه‌های اصلی مساله از بهینه‌های کشف شده، خطا نامیده می‌شود و از فرمول (۱۶) محاسبه می‌گردد.

$$\zeta = \frac{1}{m_i}\sum_{i=1}^{m}\left(\sum_{j=1}^{n}\left(s_i^j - \varphi_i^j\right)^2\right)^{\frac{1}{2}} \qquad (۱۶)$$

جایی که m تعداد بهینه‌ها و n تعداد ابعاد مساله است. همچنین s موقعیت بهینه پیدا شده توسط الگوریتم و φ موقعیت بهینه اصلی مساله است.

٤-۳- پارامترها و نتایج الگوریتم‌های دیگر

نتایج و پارامترهای سایر الگوریتم‌ها از جمله مقدار خطای قابل قبول ε از [7] و [11] قابل استخراج است. G_0 برابر ۰.۱ دامنه و α برابر ۸ است.

٤-٤- کشف بهینه‌ها

برای یافتن بهینه‌های عمومی در توابع نامقید، الگوریتم بر روی توابع F_1 تا F_{12} اجرا شده تا نتایج در جدول‌های (۱۱و۱۲) گزارش شود. در جدول (۷) پارامترهای مورد نیاز ارائه گردیده است. در این جدول تعداد حداکثر ارزیابی مجاز الگوریتم‌ها به جز الگوریتم پیشنهادی برای توابع F_1 تا F_{12} برابر ۱۰۰۰۰۰ می‌باشد.

برای کشف بهینه‌های عمومی و محلی در توابع نامقید، از جمعیت‌های با اندازه متفاوت استفاده شده و نتایج با دو نوع مقداردهی اولیه در جدول‌های (۳و۴) با الگوریتم NGSA مقایسه گردیده‌اند. همچنین در جدول (۵)، خطاهای الگوریتم پیشنهادی و الگوریتم NGSA برای نتایج به دست آمده در جدول‌های (۳و۴) و N=20 مقایسه گردیده‌اند. لازم به ذکر است در اینجا T_l برای تمامی حالت‌ها برابر 15 در نظر گرفته شده است. برای بررسی بیشتر، الگوریتم پیشنهادی برای کشف بهینه‌های عمومی و محلی در توابع نامقید بر

روی توابع F_1 تا F_5 با پارامترهای جدول (۱۰) اجرا گردیده تا این نتایج در جدول‌های (۶و۱۴) نشان داده شوند. همچنین توابع F_6 تا F_{10} با پارامترهای جدول (۸) برای الگوریتم پیشنهادی و برای الگوریتم‌های دیگر با پارامترهای جدول (۷) و حداکثر ۱۰۰۰۰۰ ارزیابی اجرا شده و نتایج در جدول (۱۳) با الگوریتم NGSA مقایسه گردیده است.

الگوریتم پیشنهادی برای کشف بهینه‌های سراسری در توابع مقید، با پارامترهای جدول (۹) اجرا شده تا نتایج جدول (۱۵) شکل گیرد.

در جدول‌ها حداکثر ارزیابی مجاز با ME و توابع با F نماد شده‌اند.

۵- نتیجه‌گیری

الگوریتم KGSA با حذف پارامترهای k_i و k_f از الگوریتم NGSA و همچنین حذف پارامتر شعاع جایگاه از سایر الگوریتم‌ها به نتایج بهتری از آن‌ها دست پیدا نموده است. اهمیت حذف پارامترها در NGSA به حدی است که با کمترین اختلافی در تنظیم پارامترها، برخی جایگاه‌ها پیدا نشده و یا جایگاه-های اضافی دیگری یافت می‌شوند. علاوه بر حذف پارامترهای مهم، تاثیر نوع مقدار دهی اولیه نیز کاهش یافته است و از همه مهم‌تر، کارا بودن ۱۰۰ درصدی آن برای کشف جایگاه‌ها و همچنین خطای کمتر و ارزیابی کمتر با جمعیت کمتر در برخورد با مسائل بهینه‌سازی چندمدی است. با توجه به ظهور تازه الگوریتم جستجوی گرانشی زمینه دارویی، نظامی و اقتصادی، از جمله زمینه‌هایی است که از آن می‌توان به عنوان کارهای آینده نام برد.

جدول (۱) : توابع نامقید

F
$F_1(x)=\sin^6(5\pi x)$
$F_2(x)=e^{-2\log(2)\times\left(\frac{x-0.1}{0.8}\right)^2}\sin^6(5\pi x)$
$F_3(x)=\sin^6\left(5\pi\left(x^{\frac{3}{4}}-0.05\right)\right)$
$F_4(x)=e^{-2\log(2)\times\left(\frac{x-0.08}{0.854}\right)^2}\sin^6\left(5\pi\left(x^{\frac{3}{4}}-0.05\right)\right)$
$F_5(x_1,x_2)=200-(x_1^2+x_2-11)^2-(x_1+x_2^2-7)^2$
$F_6(x)=\begin{cases}\frac{160}{15}(15-x) & \text{for } 0\le x\le 15\\ \frac{200}{5}(x-15) & \text{for } 15\le x\le 20\end{cases}$
$F_7(x)=\begin{cases}\frac{160}{10}x & \text{for } 0\le x\le 10\\ \frac{160}{5}(15-x) & \text{for } 10\le x\le 15\\ \frac{200}{5}(x-15) & \text{for } 15\le x\le 20\end{cases}$
$F_8(x)=\begin{cases}80(2.5-x) & \text{for } 0.0\le x\le 2.5\\ 64(x-2.5) & \text{for } 2.5\le x\le 5.0\\ 64(7.5-x) & \text{for } 5.0\le x\le 7.5\\ 28(x-7.5) & \text{for } 7.5\le x\le 12.5\\ 28(17.5-x) & \text{for } 12.5\le x\le 17.5\\ 32(x-17.5) & \text{for } 17.5\le x\le 22.5\\ 32(27.5-x) & \text{for } 22.5\le x\le 27.5\\ 80(x-27.5) & \text{for } 27.5\le x\le 30\end{cases}$
$F_9(x_1,x_2)=-4\left[\left(4-2.1x_1^2+\frac{x_1^4}{3}\right)x_1^2+x_1x_2+(-4+4x_2^2)x_2^2\right]$
$F_{10}(x_1,x_2)=500-\frac{1}{0.002+\sum_{i=0}^{24}\frac{1}{1+i+(x_1-a(i))^6+(x_2-b(i))^6}}$
where $a(i)=16((i\bmod 5)-2)$, and $b(i)=16(\lfloor(i/5)\rfloor-2)$
$F_{11}(x)=-\prod_{i=1}^{n}\sum_{j=1}^{5} j\times\cos[(j+1)x_i+j]$
$F_{12}(x)=\frac{1}{n}\sum_{i=1}^{n}\sin(10\times\log(x_i))$

جدول (۲) : توابع مقید

F
$F_{13}(x)=10n+\sum_{i=1}^{n}[x_i^2-10\cos(2\pi x_i)]$
$\text{s.t.}\ h(x)=\sum_{i=1}^{n}x_{i=0}$
$F_{14}(x)=\frac{1}{4000}\sum_{i=1}^{n}x_i^2-\prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$
$\text{s.t.}\ h(x)=\frac{1}{512(n-1)}\sum_{i=2}^{n}x_i^2-\lvert x_1\rvert=0$
$F_{15}(x)=(x_1+a_1)^2+(x_2+a_2)^2+\ \dots+(x_n+a_n)^2$ $\text{s.t.}\ [(x_1+b_1)^2+\ \dots\ (x_n+b_n)^2]\ge n^2,$ $-(n+1)\le x_i\le n+1$

جدول (۳) : نرخ موفقیت کشف بهینه‌های سراسری و محلی با ۱۲۰ بار تکرار و مقداردهی اولیه به روش تصادفی در ۳۰ بار اجرای مستقل

F	N=۷۵		N=۵۰		N=۳۵		N=۲۰	
	KGSA	NGSA	KGSA	NGSA	KGSA	NGSA	KGSA	NGSA
F_1	۱۰۰	۱۰۰	۱۰۰	۱۰۰	۱۰۰	۹۳	۱۰۰	۸۰
F_2	۱۰۰	۱۰۰	۱۰۰	۱۰۰	۱۰۰	۹۶	۱۰۰	۷۳
F_3	۱۰۰	۱۰۰	۱۰۰	۱۰۰	۱۰۰	۸۶	۱۰۰	۷۳
F_4	۱۰۰	۱۰۰	۱۰۰	۹۶	۱۰۰	۹۰	۱۰۰	۶۶
F_5	۱۰۰	۱۰۰	۱۰۰	۱۰۰	۱۰۰	۹۶	۱۰۰	۷۶

جدول (٤) : نرخ موفقیت کشف بهینه‌های سراسری و محلی با ۱۲۰ بار تکرار و مقدار دهی اولیه به روش partition در ۳۰ بار اجرای مستقل

F	KGSA	NGSA	KGSA	NGSA	KGSA	NGSA	KGSA	NGSA
F_1	۱۰۰	۱۰۰	۱۰۰	۱۰۰	۱۰۰	۱۰۰	۱۰۰	۱۰۰
F_2	۱۰۰	۱۰۰	۱۰۰	۱۰۰	۱۰۰	۱۰۰	۱۰۰	۱۰۰
F_3	۱۰۰	۱۰۰	۱۰۰	۱۰۰	۱۰۰	۱۰۰	۱۰۰	۱۰۰
F_4	۱۰۰	۱۰۰	۱۰۰	۱۰۰	۱۰۰	۱۰۰	۱۰۰	۱۰۰
F_5	۱۰۰	۱۰۰	۱۰۰	۱۰۰	۱۰۰	۱۰۰	۱۰۰	۱۰۰

جدول (٥) : نرخ خطا برای نتایج جدول‌های (۳و٤) با N=20

F	KGSA ζ random	KGSA ζ partition	NGSA ζ partition
F_1	**۱.۷٥e - ٦ ± ۱.۰۳e - ٦**	**۱.۷۸e - ٦ ± ۹.٤۱e - ۷**	۱.۶۲e - ۵ ± ۳.۴۷e - ۵
F_2	**٤.۹٦e - ۷ ± ۱.٥٤e - ٦**	**۲.۷٥e - ۷ ± ۲.۹٤e - ۷**	۰.۰۰۱± ۱.۶۶e - ۴
F_3	**۲.٤۱e - ٦ ± ۲.۱۲e - ٦**	**۲.۳٥e - ٦ ± ۱.۲۹e - ٦**	۰.۰۰۱۲ ± ۵.۰۴e - ۴
F_4	**٦.۸۷e - ۷ ± ۱.۹۸e - ٦**	**٥.۳٤e - ۷ ± ٦.۸۱e - ۷**	۲.۳۷e - ۴ ± ۳.۳۸e - ۴
F_5	**۳.٥۹e - ۳ ± ۲.۱۳e - ۳**	**٤.۲۹e - ۳ ± ٤.۲۲e - ۳**	۰.۰۵۷۰ ± ۰.۰۲۶۲

جدول (٦) : نرخ موفقیت در ۳۰ اجرای مستقل

F	KGSA	DC	SN	Niche PSO	NGSA
F_1	۱۰۰	۱۰۰	۱۰۰	۱۰۰	۱۰۰
F_2	۱۰۰	۹۳	۸۳	۹۳	۱۰۰
F_3	۱۰۰	۹۰	۱۰۰	۱۰۰	۱۰۰
F_4	۱۰۰	۹۰	۹۳	۹۳	۱۰۰
F_5	۱۰۰	۹۰	۸۶	۱۰۰	۱۰۰

جدول (۷) : پارامترهای مورد نیاز برای جدول‌های (۱۱و۱۲)

F	F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8	F_9	F_{10}	F_{11}	F_{12}
N KGSA	۱۰	۱۰	۲۰	۱۰	۲۰	۱۰	۸	۳۰	۱۵	٤۰	۱۰۰	٥۰۰
ME KGSA	۸۰۰	٦۰۰	۸۰۰	۷٥۰	۲٤۰۰	۱۸۰۰	۱۲۸۰	۳٦۰۰	۱۳٥۰	۱۰۰۰	٦۰۰۰۰	۹۰۰۰۰
T_l	۲۰	۱۵	۱۰	۱۵	۲۰	۹۰	۸۰	۶۰	۳۰	۵۰	۶۰	۴۵
N NGSA	۵۰	۵۰	۵۰	۵۰	۵۰	۱۰۰	۱۰۰	۱۰۰	۱۰۰	۵۰۰	۲۵۰	۱۰۰

جدول (۸) : پارامترهای مورد نیاز KGSA و جدول (۱۳)

F	F_6	F_7	F_8	F_9	F_{10}
N	**۱٥**	**۸**	**۳۰**	**۱٥**	**۸۰**
ME	**۲۷۰۰**	**٥٦۰۰**	**۳٦۰۰**	**٥۲٥۰**	**۲۰۰۰۰**
T_l	۹۰	۷۰	۶۰	۵۰	۵۰

جدول (۹) : پارامترهای مورد نیاز برای جدول (١٤)

	KGSA			NGSA		
F	F_{13}	F_{14}	F_{15}	F_{13}	F_{14}	F_{15}
N	**۱۰**	**٤۰**	**۲٥**	۳۰۰	۲۰۰	۲۰۰
T	**۸۰**	**۱۰۰**	**٤٥**	۲۰۰	۱۰۰	۱۰۰
T_l	۴۰	۵۰	۴۵	-	–	–

جدول (۱۰) : پارامترهای مورد نیاز جدول‌های (٦و١٥)

F	F_1	F_2	F_3	F_4	F_5
N KGSA	۱۰	۱۰	۲۰	۱۰	۲۰
T KGSA	**۸۰**	**٦۰**	**٤۰**	**۷٥**	**۱۲۰**
T_l	۲۰	۱۵	۱۰	۱۵	۲۰
N NGSA	۲۰	۲۰	۲۰	۲۰	۲۰
T NGSA	۲۰۰۰	۲۰۰۰	۲۰۰۰	۲۰۰۰	۲۰۰۰

جدول (۱۱) : نرخ موفقیت برای یافتن بهینه‌های سراسری در ۵۰ بار اجرای مستقل

F	KGSA	NGSA	r2 PSO	r3 PSO	r2 PSO -lhc	r2 PSO -lhc	FER-PSO	SPSO
F_1	**۱۰۰**	۱۰۰	۱۰۰	۱۰۰	۱۰۰	۱۰۰	۱۰۰	۱۰۰
F_2	**۱۰۰**	۱۰۰	۹۸	۱۰۰	۱۰۰	۱۰۰	۱۰۰	۱۰۰
F_3	**۱۰۰**	۱۰۰	۹۸	۹۸	۱۰۰	۱۰۰	۱۰۰	۱۰۰
F_4	**۱۰۰**	۱۰۰	۱۰۰	۱۰۰	۱۰۰	۱۰۰	۱۰۰	۱۰۰
F_5	**۱۰۰**	۱۰۰	۹۲	۷۴	۱۰۰	۹۸	۹۸	۱۰۰
F_6	**۱۰۰**	۱۰۰	۹۸	۱۰۰	۹۴	۷۸	۸۸	۲۴
F_7	**۱۰۰**	۱۰۰	۱۰۰	۹۶	۹۸	۸۸	۱۰۰	۲۲
F_8	**۱۰۰**	۹۴	۱۰۰	۹۶	۹۶	۹۶	۹۸	۴۰
F_9	**۱۰۰**	۱۰۰	۱۰۰	۱۰۰	۱۰۰	۱۰۰	۱۰۰	۱۰۰
F_{10}	**۱۰۰**	۱۰۰	۱۰۰	۱۰۰	۷۲	۷۸	۱۰۰	۵۰
F_{11}(2D)	**۱۰۰**	۱۰۰	۹۰	۹۸	۹۸	۱۰۰	۵۶	۴۹
F_{12}(1D)	**۱۰۰**	۹۲	۹۴	۸۶	۹۲	۹۰	۸۸	۸۴

جدول (۱۲) : تعداد ارزیابی مورد نیاز الگوریتم‌های مختلف برای نتایج ارائه شده در جدول(۱۱)

F	KGSA	NGSA	r2 PSO	r3 PSO	r2 PSO -lhc	r2 PSO -lhc	FER-PSO	SPSO
F_1	**۲۱٤±۱۳۳**	۲۶۳±۸۹	۳۷۶±۳۰	۴۴۳±۵۱	۳۹۶±۵۱	۴۴۷±۵۲	۳۸۴±۲۹	۳۵۵±۳۰
F_2	**۱۰٥±۱۰۸**	۳۰۰±۱۰۷	۲۱۲۰±۱۹۹۹	۱۴۱±۱۱	۱۴۳±۱۴	۱۴۴±۱۳	۱۷۰±۱۲	۱۲۷±۹
F_3	**۲٦۳±۱٤۰**	۳۳۴±۸۱	۲۴۳۰±۱۹۹۴	۲۴۴۰±۱۹۹۴	۴۵۶±۳۳	۶۲۳±۲۷۳	۳۱۷±۳۱	۳۴۳±۲۳
F_4	**۱۳۰±۱۲٤**	۳۱۶±۷۶	۱۷۵±۱۷	۱۶۰±۲۰	۱۷۸±۱۸	۱۶۲±۱۶	۱۸۹±۲۰	۱۴۴±۱۳
F_5	**۸٦٤±۳۸۲**	۱۶۳۲±۳۳۰	۷۸۷۰±۲۸۹۱	۲۱۴۰۰±۵۴۶۷	۱۴۹۰±۱۳۸	۷۳۸۰±۳۳۴۷	۵۰۷۰±۱۹۴۵	۱۲۵۰±۴۵
F_6	**۳۰٤±۱۱۰**	۴۷۷±۴۰۹	۳۴۶۰±۱۹۷	۲۶۲۰±۸۷۴	۷۳۹۰±۳۳۴۰	۲۳۲۰۰±۵۸۳۴	۱۴۴۰۰±۴۵۳۵	۷۷۲۰۰±۵۸۵۹
F_7	۲۹۰±۱۳۰	۲۳۴±۱۰۸	۲۹۶۰±۱۵۲۰	۵۳۴۰±۲۷۶۴	۴۳۴۰±۲۲۲۹	۱۳۱۰۰±۴۵۸۸	۲۱۱۰±۲۲۷	۷۸۳۰۰±۵۸۵۶
F_8	**٥٥۷±۳٤٦**	۶۹۴±۸۵۳	۹۷۸±۱۸۶	۴۶۵۰±۲۷۸۴	۴۷۱۰±۲۷۸۳	۶۷۳۰±۳۰۸۸	۲۶۶۰±۱۹۹۲	۶۳۳۰۰±۶۷۷۳
F_9	**۲۳۰±۱۲۱**	۱۰۳۲±۸۹۲	۶۱۹±۲۴	۶۸۴±۳۰	۶۱۸±۳۰	۶۵۰±۲۵	۹۶۵±۵۳	۶۵۳±۳۲
F_{10}	**۳۱۱۹±۲۱۷۲**	۴۱۶۴±۱۷۶۸	۴۳۶۰±۵۵۹	۵۳۱۰±۴۵۳	۲۹۷۰۰±۶۲۷۷	۲۴۸۰۰±۵۷۳۸	۳۴۷۰±۳۳۶	۴۲۸۰۰±۶۹۶۸
F_{11}(2D)	۳۳۳۴۴±۸۰۹۹	۵۳۶۹±۱۹۳۰	۵۵۹۰۰±۲۶۷۶	۳۹۱۰۰±۱۶۴۸	۳۷۸۰۰±۱۴۸۰	۳۲۴۰۰±۵۸۱	۹۴۹۰۰±۱۲۶۱	۶۱۶۰۰±۴۴۶۳
F_{12}(1D)	۱۲۴۸۰±۱۸۱۷۵	۲۱۳۴±۴۳۰	۸۳۱۰±۳۳۷۱	۱۵۴۰۰±۴۹۰۶	۹۶۰۰±۳۸۲۴	۱۴۷۰۰±۴۳۴۴	۱۳۰۰۰±۴۶۰۱	۱۷۰۰۰±۵۱۶۲

جدول (۱۳) : مقایسه NGSA و KGSA برای کشف همه بهینه‌های محلی و عمومی در ۵۰ بار اجرای مستقل

	KGSA			NGSA		
Function	تعداد ارزیابی	ζ	نرخ موفقیت	تعداد ارزیابی	ζ	نرخ موفقیت
F_6	**٤۱۳ ± ۱۱۳**	**۱.۰۲e – ٤ ± ۱.۳۸e – ٤**	**۱۰۰**	۵۴۲ ± ۳۸۶	۴.۲۷e – ۴ ± ۴.۱۸e – ۴	۱۰۰
F_7	**۲٦۷ ± ۱۲۸**	**٥.۱۲e – ٦ ± ٥.٦٦e – ٦**	**۱۰۰**	۳۲۱ ± ۱۱۸	۷.۴۴e – ۶ ± ۶.۰۱e – ۶	۱۰۰
F_8	**۸۷۳ ± ۷۲۱**	**٦.٥۱e – ٥ ± ۲.۰۸e – ٤**	**۱۰۰**	۹۶۶ ± ۷۶۵	۲.۱۹e – ۳ ± ۱.۳۷e – ۲	۹۴
F_9	**۱۰۹۷ ± ۸٥٦**	**۷.۲۹e – ٥ ± ۱.۸۸e – ٤**	**۱۰۰**	۱۱۲۲ ± ۳۵۳	۱.۵۰e – ۳ ± ۳.۳۳e – ۲	۹۸
F_{10}	۱۱۳۱۸ ± ۲۸۶۱	۴.۵۱e – ۲ ± ۶.۷۳e – ۳	۱۰۰	۶۱۸۶ ± ۲۱۳۳	۹.۶۶e – ۴ ± ۱.۱۸e – ۳	۱۰۰

جدول (١٤) : تعداد ارزیابی لازم برای نتایج به دست آمده در جدول (٦)

F	KGSA	DC	SN	Niche PSO	NGSA
F_1	**۲۰۸ ± ۱۱٥**	۱۴۶۴۷ ± ۴۶۱۲	۴۱۰۲ ± ۵۷۷	۲۳۷۲ ± ۱۰۹	۱۷۸۶ ± ۲۰۴
F_2	**۲۱۱ ± ۱۰۳**	۱۳۰۵۲ ± ۲۵۰۷	۳۵۰۵ ± ۴۶۳	۲۹۳۴ ± ۴۷۵	۱۸۹۲ ± ۵۶۱
F_3	**۲۸٦ ± ۱۳٦**	۱۳۹۳۰ ± ۳۲۸۴	۴۱۴۱ ± ۵۵۴	۲۴۰۴ ± ۱۹۵	۱۷۵۲ ± ۲۷۳
F_4	**۲۷۲ ± ۱٤۹**	۱۳۹۲۹ ± ۲۹۹۶	۳۴۶۴ ± ۲۸۷	۲۸۲۰ ± ۵۱۷	۱۸۰۶ ± ۳۰۷
F_5	**۸۹۷ ± ٤٤٦**	۱۴۲۹۶ ± ۳۴۰۸	۳۴۲۳ ± ۴۰۲	۲۱۵۱ ± ۲۰۰	۲۰۳۳ ± ۲۰۱

جدول (۱۵) : نرخ موفقیت، نرخ خطا و تعداد ارزیابی مورد نیاز برای یافتن بهینه‌های سراسری در ۵۰ بار اجرای مستقل

F	ζ	تعداد ارزیابی	نرخ موفقیت(%)	روش
F_{13}	۳.۱۵۸۶e - ۰۴ ± ۰.۰۰۸۰	۵۱۰ ± ۴۳۸.۱۰۸۲	۱۰۰	r2PSO
	۶.۳۴۲۵e - ۰۴ ± ۰.۰۱۰۵	۵۴۶ ± ۳۹۶.۰۰۵۶	۱۰۰	r3PSO
	۵.۵۲۵۶e - ۰۴ ± ۰.۰۰۶۷	۴۷۴ ± ۳۸۳.۷۵۶۹	۱۰۰	r2PSO-lhc
	۱.۲۶۱۹e - ۰۳ ± ۰.۰۰۸۳	۴۵۰ ± ۲۹۲.۲۴۹۸	۱۰۰	r3PSO-lhc
	۴.۳۷۱۷e - ۰۴ ± ۰.۰۱۰۳	۱۰۹۸ ± ۶۶۲.۸۶۰۲	۱۰۰	Deterministic crowding
	۳.۱۲۵۵e - ۰۴ ± ۰.۰۱۰۷	۷۸ ± ۱۴۶.۰۹۱۹	۱۰۰	NGSA
	٦.٠٤e - ٠٥ ± ٥.٦٧ e - ٠٥	۹۲ ± ۴۳	**١٠٠**	KGSA
F_{14}	۲.۳۳۱۳e - ۰۲ ± ۲.۲۵۲۲	۲۳۹۶ ± ۰.۱۴۶۹	۱۰۰	r2PSO
	۷.۱۵۳۱e - ۰۳ ± ۰.۹۸۳۵	۲۰۹۲ ± ۰.۵۰۵۰	۱۰۰	r3PSO
	۸.۹۰۴۷e - ۰۳ ± ۱.۸۳۱۴	۲۴۷۶ ± ۰.۵۱۲۳	۱۰۰	r2PSO-lhc
	۱.۳۱۱۶e - ۰۲ ± ۲.۱۰۶۵	۲۲۳۲ ± ۰.۵۵۳۹	۱۰۰	r3PSO-lhc
	۱.۶۹۰۲e - ۰۲ ± ۰.۴۹۵۶	۲۱۵۵۲ ± ۱.۰۰۵۶	۱۰۰	Deterministic crowding
	۱.۹۶۷۲e - ۰۲ ± ۰.۳۷۹۰	۱۹۴۴ ± ۱.۲۹۴۴e + ۰۳	۱۰۰	NGSA
	٥.٧٩e - ٠٣ ± ٠.٠٤٠٩٨	**١٣٩٥ ± ٥٨٧**	**١٠٠**	KGSA
F_{15}	۰.۳۹۶۶ ± ۰.۲۳۵۲	۷۸۸ ± ۰.۰۸۴۹	۱۰۰	r2PSO
	۰.۳۸۱ ± ۰.۱۹۹۷	۷۹۲ ± ۰.۰۸۴۹	۱۰۰	r3PSO
	۴.۹۸۳۲e - ۰۳ ± ۰.۲۰۸۱	۸۱۲ ± ۰.۰۳۹۶	۱۰۰	r2PSO-lhc
	۰.۳۲۳۷ ± ۰.۱۸۷۳	۷۹۶ ± ۰.۰۴۸۰	۱۰۰	r3PSO-lhc
	۶.۲۵۷۱e - ۰۴ ± ۰.۰۵۵۲	۶۶۷۲ ± ۰.۰۲۸۳	۱۰۰	Deterministic crowding
	۵.۰۲۶۲e - ۰۴ ± ۰.۰۲۰۳	۳۱۱۶ ± ۱.۰۱۳۵	۱۰۰	NGSA
	٣.٣٧e - ٠٤ ± ٦.٣٧ e - ٠٤	**٤٨٠ ± ١٠٣**	**١٠٠**	KGSA

مراجع

[1] Rashedi, E., Nezamabadi-Pour, H., Saryazdi, S., "BGSA: binary gravitational search algorithm", Natural Computing, Vol. 9, No. 3, pp. 727-745, 2010.

[2] Rashedi, E., Nezamabadi-Pour, H., Saryazdi ,S., "*GSA: a gravitational search algorithm*", Information sciences, Vol 179,No 13, pp 2232-2248, 2009.

[3] *Brits, R., Niching strategies for particle swarm optimization, PhD diss., University of Pretoria, 2002.*

[4] *Streichert, F., Stein, G., Ulmer, H., Zell, A., "A clustering based niching method for evolutionary algorithms", In Genetic and Evolutionary Computation—GECCO, pp. 644-645, 2003.*

[5] *Li, X., "Adaptively choosing neighbourhood bests using species in a particle swarm optimizer for multimodal function optimization", In Genetic and Evolutionary Computation–GECCO, pp. 105-116, 2004.*

[6] *Seo, J. H., Im, C. H., Heo, C. G., Kim, J. K., Jung, H. K., Lee, C.-G., "Multimodal function optimization based on particle swarm optimization", Magnetics, IEEE Transactions, Vol. 42, No. 4, pp. 1095-1098, 2006.*

[7] *Li, X., "Niching without niching parameters: particle swarm optimization using a ring topology," Evolutionary Computation, IEEE Transactions, Vol. 14, No. 1, pp. 150-169, 2010.*

[8] *Li, X., "Multimodal function optimization based on fitness-euclidean distance ratio", In Proc. Genet. Evol. Comput. Conf, pp. 78-85, 2007.*

[9] *Das, S., Maity, S., Qu, B., Ponnuthurai, N. S., "Real-parameter evolutionary multimodal optimization—A survey of the state-of-the-art", Swarm and Evolutionary Computation, Vol. 1, No. 2, pp. 71-88, 2011*

[10] *Beasley, D., Bull, D. R., Martin, R. R., "A sequential niche technique for multimodal function optimization. Evolutionary computation",* Vol *1, No 2, pp 101-125, 1993.*

[11] *Yazdani, S., Nezamabadi-pour, H., Kamyab, S, "A Gravitational Search Algorithm for Multimodal Optimization", Swarm and Evolutionary Computation, pp. 1–14, 2013.*

[12] Kimura, S., Matsumura, K., "Constrained multimodal function optimization using a simple evolutionary algorithm", In *Evolutionary Computation (CEC), 2011 IEEE Congress on*, pp. 447-454, 2011.

زیرنویس‌ها

[1] Multimodal
[2] Unimodal
[3] Niching
[4] Gravitational search algorithm
[5] Elitism
[6] Iteration
[7] Clustering
[8] K-means GSA
[9] Constraint
[10] Unconstraint