EasyChair Preprint

№ 1859

Self Adaptive Improved Bat Algorithm

Jalil Alahloei Zare and Behroz Keshtegar

EasyChair preprints are intended for rapid dissemination of research results and are integrated with the rest of EasyChair.

November 7, 2019

الگوریتم بهبود یافته تطبیق پذیر خفاش

جلیل الهلوئی زارع[۱] ، بهروز کشته‌گر[۲]

[۱]دانشگاه زابل، دانشجوی کارشناسی ارشد مهندسی برق قدرت، jalilzare89@yahoo.com

[۲]دانشگاه زابل،استادیار دانشکده فنی ومهندسی،گروه مهندسی عمران، bkeshtegar@yahoo.com

چکیده:

هوش جمعی یکی از قویترین تکنیک‌های بهینه‌سازی می‌باشد، که بر مبنای رفتارهای گروهی موجودات عمل می‌کند. الگوریتم خفاش یک الگوریتم الهام گرفته از رفتار جمعی خفاش‌ها در محیط طبیعی است، که در سال ۲۰۱۰ توسط یانگ ارائه شد. در ادامه محققان سعی بر بهبود الگوریتم خفاش داشته‌اند. در این مقاله سعی بر این است تا با بهبود عملکرد الگوریتم خفاش به حل معادلات ساده و پیچیده ریاضی پرداخته شود. لذا برای بهبود عملکرد الگوریتم اقدام به اصلاح روابط سرعت و حرکت خفاش‌ها نموده به گونه‌ای که حرکت خفاش‌ها به عنوان راه‌حل‌های بهینه‌سازی به سمت هدف به صورت تطبیقی باشد، همچنین برای رهایی از بهینه‌های محلی، از عملگر جهش جهت بررسی تمام نقاط فضای جستجو استفاده شده است. نتایج بدست آمده حاکی از برتری الگوریتم اصلاحی نسبت به الگوریتم‌های دیگر می‌باشد.

کلمات کلیدی:

هوش جمعی، تکنیک‌های بهینه‌سازی، الگوریتم خفاش

۱- مقدمه:

طبیعت همیشه الهام‌بخش محققان بوده است. در گذشته، بسیاری از الگوریتم‌های الهام گرفته از طبیعت، برای حل مشکلات سخت در مسائل بهینه‌سازی توسعه یافته‌اند. بطور کلی دو نمونه از رفتارهای طبیعی برای توسعه الگوریتم‌ها مورد استفاده قرار گرفته شده است. الگوریتم‌های تکاملی والگوریتم‌های بر مبنای هوش جمعی، الگوریتم‌های تکاملی تکنیک‌های بهینه‌سازی[۱] هستند که بر مبنای اصل بقای داروین[۲] طراحی شده‌اند. این اصل بیان می‌دارد که در طبیعت گونه‌های وفق‌پذیر شانس بالاتری برای بقا دارند. الگوریتم‌های تکاملی شامل موارد زیر می‌باشد: الگوریتم ژنتیک، استراتژی تکاملی، برنامه‌نویسی ژنتیکی، برنامه‌نویسی تکاملی و تکامل دیفرانسیلی. همچنین همه این الگوریتم‌ها به طور مستقل توسعه یافته‌اند.

هوش جمعی یک رفتار نا متمرکز و خود سازمان یافته، طبیعی یا مصنوعی در بین موجودات می‌باشد، هوش جمعی در سال ۱۹۸۹ توسط بنی معرفی گردید. شناخته شده‌ترین الگوریتم‌های هوش جمعی عبارتند از: بهینه‌سازی ازدحام ذرات(PSO)، بهینه‌سازی کلونی مورچه، کلونی زنبور عسل مصنوعی، الگوریتم کرم شب‌تاب و جستجوی فاخته. الگوریتم خفاش یکی دیگر از الگوریتم‌های هوش جمعی برمبنای رفتار شکار ریزخفاش‌ها است که در سال ۲۰۱۰ توسط یانگ معرفی شد[۳]. BA با استفاده از نرخ انتشار فرکانس توسط ریز خفاش‌ها به دنبال یافتن پاسخ خواهد بود. در نتیجه، این نوع الگوریتم به دلیل شروع سریع، الگوریتم کارآمدی است، الگوریتم خفاش استاندارد بر اساس ویژگی جهت‌یابی ریز خفاش ها عمل می‌کند. بدیهی است که زمینه‌های زیادی برای بهبود وجود دارد.

به منظور بهبود عملکرد، محققین روشهای متنوعی را به کار بسته‌اند، که در زیر نمونه‌هایی از این بهبودها بیان می‌شود:

۱- الگوریتم خفاش با خوشه‌بندی K-means [۴]: کوموراسامی و واهی در سال ۲۰۱۲ ترکیبی از الگوریتم خفاش و الگوریتم K-means، (KMBA) را برای خوشه بندی بهتر ارائه داده‌اند.

۲- الگوریتم هرج و مرج خفاش (CBA)[۵]: لین و همکاران در سال ۲۰۱۲ یک الگوریتم خفاش هرج و مرج، برای تخمین پارامترها در سیستم‌های بیولوژیکی پویا ارائه داده‌اند.

۳- الگوریتم خفاش باینری (BBA) [۶]: ناکامورا و همکاران در سال ۲۰۱۲ الگوریتم خفاش گسسته را برای حل مسائل خوشه‌بندی و انتخاب ویژگی توسعه داده‌اند.

۴- الگوریتم خفاش بهبود یافته (IBA) [۷]: جمیل و همکاران در سال ۲۰۱۳ الگوریتم خفاش را با تغییرات در میزان بلندی و نرخ انتشار و ترکیب با پرواز لوی بهبود داده و با آزمایش بر روی توابع گوناگون کارایی الگوریتم را مشخص کردند.

۵- ژانگ و وانگ در سال ۲۰۱۲ از جهش برای تنوع فضای جستجوی راه‌حل‌ها استفاده کرده و برای پردازش تصویر از این الگوریتم استفاده کرده‌اند [۸].

۶- وانگ و ژئو در سال ۲۰۱۳ با ترکیب الگوریتم خفاش و جستجوی هارمونی به حل مسائل عددی و توابع محک پرداخته‌اند [۹].

۷- فاستر و همکاران در سال ۲۰۱۳ الگوریتم ترکیبی خفاش و تکامل دیفرانسیلی را ارائه داده‌اند [۱۰].

۸- فاستر و همکاران در سال ۲۰۱۳ الگوریتم خفاش چهارگانه (QBA) را برای محاسبات هندسی و مسائل بهینه سازی در مقیاس بزرگ ارائه داده‌اند [۱۱].

۲- الگوریتم خفاش[۳]

الگوریتم خفاش در سال ۲۰۱۰ توسط یانگ معرفی گردید. این الگوریتم از خاصیت پژواک‌یابی ریز خفاش‌ها استفاده می‌کند. خفاش‌ها با انتشار پالس‌های صوتی با سرعت و فرکانس معین و دریافت بازتاب پالس ارسالی، یک تصویر سه بعدی از فضای اطراف خود ایجاد می‌کنند، این کار به وسیله تأخیر زمانی دریافت پالس امکان پذیر است. هر خفاش در حدود ۱۰ الی ۲۰ پالس با فرکانس معین در هر ثانیه می‌تواند ارسال کند. با این روش خفاش‌ها محل دقیق طعمه خود را شناسایی می‌کنند.

این خفاش‌ها از طول موج‌های بین ۰/۷ تا ۱۷ میلیمتر و فرکانس متغییر بین ۲۵ تا ۱۵۰ کیلوهرتز استفاده می‌کنند. نرخ انتشار پالس توسط خفاش‌ها بین ۰ تا ۱ قابل تغییر است [۱۲،۱۳،۱۴].

با توجه به ویژگی پژواک‌یابی خفاش‌ها، آقای یانگ در سال ۲۰۱۰ الگوریتم خفاش را بر مبنای سه قانون اساسی زیر بنا نهاد:

- همه خفاش‌ها برای تشخیص محیط از جهت‌یابی استفاده می‌کنند و تفاوت بین طعمه و سایر موانع را با استفاده از ردیابی صوتی درک می‌کنند.
- خفاش‌ها به صورت تصادفی با سرعت v_i ، در موقعیت x_i و با فرکانس f_{min}، با طول موج λ و بلندی صدای A_0 در جستجوی طعمه هستند، همچنین آنها می‌توانند طول موج پالس‌های منتشر شده را به طور اتوماتیک تنظیم کرده و نرخ انتشار پالس خود را طبق نزدیکی طعمه‌ی خود، مطابقت دهند.
- اگرچه بلندی صدا می‌تواند به شیوه‌های متفاوتی تغییر کند، اما فرض می شود که بلندی صدا از مقدار A_0 تا A_{min} تغییر می‌کند.

۲-۱) حرکت خفاش‌ها

حرکت هر خفاش با سرعت v_i^t در موقعیت x_i^t ، با تکرار t در یک فضای جستجوی d بعدی، برای یافتن طعمه انجام می‌شود. در میان تمامی خفاش‌ها تنها یک راه حل بهینه x_* وجود دارد. سه قاعده فوق را می‌توان در روابط زیر بیان نمود.

$$f_i = f_{min} + (f_{max} - f_{\min})\beta \qquad (۱)$$

$$v_i^t = v_i^{t-1} + (x_i^{t-1} - x_*)f_i \qquad (۲)$$

$$x_i^t = x_i^{t-1} + v_i^t \qquad (۳)$$

در اینجا $\beta \in [0\sim1]$ یک بردار تصادفی است، که با توزیع یکنواخت بدست آمده است.

۲-۲) بلندی A_i و نرخ انتشار r_i صدا

بمنظور ارائه یک مکانیزم مؤثر، برای کنترل جستجو و تغییر در بهره برداری در الگوریتم، می‌توان بلندی صدا و نرخ انتشار پالس را در طول تکرارهای جستجو تغییر داد. از آنجاییکه که هر خفاش با یافتن طعمه خود، بلندی صدا را کاهش داده و نرخ انتشار پالس را افزایش می‌دهد، مقادیر بلندی صدا و نرخ انتشار پالس

برای خفاش‌ها در طی تکرار t به صورت معادلات زیر تغییر می‌کند.

$$A_i^{t+1} = \alpha A_i^t \qquad (۴)$$

$$r_i^{t+1} = r_i^0[1 - \exp(-\gamma t)] \qquad (۵)$$

$$A_i^t \to 0 \cdot \quad r_i^t \to r_i^0 \cdot \qquad as\ t \to \infty \qquad (۶)$$

در اینجا α و γ ضرایب ثابت بوده و $0 < \alpha < 1$ و $\gamma > 0$ می‌باشد، با این حال با توجه به تجربیات برای α و γ مقادیری بین ۰,۹ تا ۰,۹۸ بدست آمده است.

براساس قوانین گفته شده و روابط فوق مراحل عملکرد الگوریتم خفاش به صورت زیر بیان می‌گردد.

Bat Algorithm
Input: Bat population $x_i = (x_{i1}. \dots \dots . x_{iD})^T$ for $i = 1. \dots . N_p$
Output: the best solution x_{best} and its corresponding value $f_{min} = \min(f(x))$.
1: Initial_bat ()
2: eval = evaluate the new population
3: $f_{min} = find\ the\ best\ solution\ (x_{best})$.
4: while termination condition not meet do
5: for $i = 1$ **to** N_p do
6: y= generate new solution (x_i);
7: if rand(0,1)>r_i then
8: y= improve the best solution (x_{best})
9: end if
10: $f_{new} = evaluate\ new\ solution\ (y)$
11: eval=eval+1;
12: if $f_{new} \le f_i$ **and** $N(0.1) < A_i$ then
13: $x_i = y$; $f_i = f_{new}$;
14: end if
15: $f_{min} = find\ the\ best\ solution\ (x_{best})$;
16: end for
17: end while

عملکرد الگوریتم خفاش اصلی در فلوچارت فوق نشان داده شده است.

۳- الگوریتم بهبود یافته خفاش(ISABA)

در این تحقیق سعی بر این است تا با اصلاح روابط مربوط به موقعیت و سرعت هر خفاش به عنوان راه حل بهینه‌سازی باعث بهبود عملکرد الگوریتم شویم. چنانچه گفته شد الگوریتم خفاش سرعت بالایی در یافتن پاسخ دارد، در این تحقیق سعی بر این است تا با اصلاح رابطه شماره ۳، سرعت پاسخ‌گویی الگوریتم را افزایش دهیم. بدین منظور رابطه زیر، اصلاح شده رابطه تغییرات موقعیت برای هر خفاش به عنوان راه حل بهینه‌سازی می‌باشد.

$$x_i^t = x_* + v_i^t \qquad (۷)$$

این تغییر در شیوه حرکت خفاش ها باعث می‌شود تا هر خفاش بتواند به سمت موقعیت بهینه حرکت کند. این تغییر کمک زیادی به همگرایی الگوریتم می‌کند، ایراد این رابطه به دام افتادن الگوریتم در بهینه محلی می‌باشد، برای این منظور باید تدبیری اندیشیده شود تا الگوریتم تمام فضای جستجو را برای یافتن پاسخ اصلی بررسی نماید، برای رهایی از بهینه‌های محلی می‌توان از عملگر جهش استفاده کرد، در این تحقیق عملگر جهش به صورت اضافه کردن ثابت تصادفی در محدوده فضای جستجو به ماتریس موقعیت فعلی هر خفاش و موقعیت بهینه، می‌باشد. این عملگر باعث بررسی تمام فضای جستجو شده، که منتج به توانایی بالای الگوریتم برای پیدا کردن موقعیت بهینه اصلی برای مسائل می‌شود. عملگر جهش به صورت تصادفی به تعدادی از اعضای جمعیت و نسبت معینی از پارامترهای ماتریس موقعیت در هر راه‌حل اعمال می‌شود.

همانطور که ملاحظه می‌شود در رابطه شماره ۷ موقعیت هر خفاش به موقعیت بهینه و سرعت بستگی دارد، بردار سرعت مربوط به هر خفاش این احتمال را ایجاد می‌کند که موقعیت هر خفاش از موقعیت بهینه منحرف شود، با توجه به این موضوع و این واقعیت که سرعت هر خفاش بعد از شکار طعمه به صفر می‌رسد، از این ایده می‌توان برای عدم انحراف موقعیت هر راه‌حل از موقعیت بهینه استفاده کرد. برای این منظور میتوان از ضریب خود تطبیق w، برای صفر کردن بردار سرعت در زمان رسیدن به بهینه اصلی استفاده کرد. با توجه به مطالب گفته شده رابطه شماره ۲ مربوط به بردار سرعت هر خفاش به صورت زیر اصلاح می‌گردد.

$$v_i^t = w \cdot (v_i^{t-1} + (x_i^{t-1} - x_*) f_i) \qquad (۸)$$

ضریب w، برای عدم انحراف بردار موقعیت هنگام رسیدن به مکان بهینه می‌باشد، این ضریب تطبیق پذیر بوده و حین اجرای الگوریتم در هر مرحله متغییر می‌باشد، هنگام همگرایی الگوریتم به سمت بهینه این ضریب به سمت صفر میل خواهد کرد. رابطه w به صورت زیر می‌باشد.

$$w = x_{*_i} - x_{*_{i-1}} \qquad (۹)$$

در این رابطه x_{*_i} بردار موقعیت بهینه فعلی بوده و $x_{*_{i-1}}$ بردار موقعیت مربوط به بهینه در تکرار قبلی می‌باشد.

٤- نتایج آزمایش

برای ارزیابی عملکرد الگوریتم اصلاحی، این الگوریتم با الگوریتم‌های خفاش(BA)، دیفرانسیل تکاملی(DE)، الگوریتم ترکیبی خفاش و دیفرانسیل تکاملی (HDBA)، الگوریتم ازدحام ذرات (PSO) و الگوریتم ژنتیک، در توابع محک [۱۵] مورد مقایسه قرار گرفته است. در جداول ۱ تا ۳ توابع محک مورد استفاده در این تحقیق ارائه شده است.

جدول۱: توابع محک نمایی واحد

پاسخ	حدود	اندازه	تابع
۰	[−۱۰۰،۱۰۰]	۳۰	$f_1 = \sum_{i=1}^{n} x_i^2$
۰	[−۱۰،۱۰]	۳۰	$f_2 = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$
۰	[−۱۰۰،۱۰۰]	۳۰	$f_3 = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j \right)^2$
۰	[−۱۰۰،۱۰۰]	۳۰	$f_4 = max_i\{\lvert x_i \rvert.\ 1 \le i \le n\ \}$
۰	[−۳۰،۳۰]	۳۰	$f_5 = \sum_{i=1}^{n-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$
۰	[−۱۰۰،۱۰۰]	۳۰	$f_6 = \sum_{i=1}^{n} [x_i + 0.5]^2$
۰	[−۱/۲۸ ،۱/۲۸]	۳۰	$f_7 = \sum_{i=1}^{n} i x_i^4 + rand[0.1)$

برخلاف توابع نمایی واحد، توابع نمایی چندگانه دارای بهینه‌های محلی متعددی می‌باشند که الگوریتم ممکن است در آنها به دام افتد. این توابع برای محک زدن قدرت یک الگوریتم مناسب می‌باشند.

٤-۱) تجزیه و تحلیل همگرایی

در این بخش رفتار همگرایی الگوریتم اصلاحی بررسی شده است. طبق گفته برگ و همکاران [۱۶]، تغییرات ناگهانی در حرکت عوامل جستجو، باید در مراحل ابتدایی بهینه‌سازی ایجاد شود. این رویه به الگوریتم کمک می‌کند تا فضای جستجو را به طور گسترده بررسی نماید. سپس این تغییرات برای رسیدن به بهینه اصلی باید کاهش یابد.

برای بررسی عملکرد الگوریتم، نتایج میانگین، بهینه و انحراف معیار بدست آمده از ۳۰ بار اجرای هر الگوریتم برای توابع ذکر شده در جداول ۴تا ۶ بیان شده است.

جدول ۲: توابع محک نمایی چندگانه

پاسخ	حدود	اندازه	تابع
−۲۰۹۵	[−۵۰۰ ،۵۰۰]	۳۰	$f_8 = \sum_{i=1}^{n} x_i \sin\left(\sqrt{\lvert x_i \rvert}\right)$
۰	[−۵/۱۲ ،۵/۱۲]	۳۰	$f_9 = \sum_{i=1}^{n} [x_i^2 - 10\cos(2\pi x_i) + 10]$
۰	[−۳۲ ،۳۲]	۳۰	$f_{10} = -20exp\left(-0 \cdot 2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - exp\left(\frac{1}{n}\sum_{i=1}^{n} \cos(2\pi x_i)\right) + 20 + e$
۰	[−۶۰۰ ،۶۰۰]	۳۰	$f_{11} = \frac{1}{4000}\sum_{i=1}^{n} x_i^n - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$
۰	[−۵۰ ،۵۰]	۳۰	$f_{12} = \frac{\pi}{n}\left\{10\sin(\pi y_1) + \sum_{i=1}^{n-1}(y_i - 1)^2[1 + 10(\sin(\pi y_{i+1}))^2] + (y_n - 1)^2\right\} + \sum_{i=1}^{n} u(x_i. 10.100.4)$
			$u(x_i.a.k.m) = \begin{cases} k(x_i - a)^m & x_i > a \\ 0 & -a < x_i < a \\ k(-x_i - a)^m & x_i < -a \end{cases}$ $\left(y_i = 1 + \frac{x_i + 1}{4}\right)$
۰	[−۵۰ ،۵۰]	۳۰	$f_{13} = 0 \cdot 1\left\{(\sin(3\pi x_1))^2 + \sum_{i=1}^{n}(x_i - 1)^2[1 + (\sin(3\pi x_i + 1))^2] + (x_n - 1)^2[1 + (\sin(2\pi x_n))^2]\right\} + \sum_{i=1}^{n} u(x_i. 5.100.4)$
−۴/۶۸۷	[۰ π]	۳۰	$f_{14} = -\sum_{i=1}^{n} \sin(x_i) * \left(\sin\left(\frac{i \cdot x_i^2}{\pi}\right)\right)^{2m} . m = 10$
−۱	[−۲۰ ،۲۰]	۳۰	$f_{15} = \left[e^{-\sum_{i=1}^{n}\left(x_i/\beta\right)^{2m}} - 2e^{-\sum_{i=1}^{n} x_i^2}\right] * \prod_{i=1}^{n} (\cos x_i)^2 . m = 5$
−۱	[−۱۰ ،۱۰]	۳۰	$f_{16} = \left\{\left[\sum_{i=1}^{n} (\sin x_i)^2\right] - exp\left(-\sum_{i=1}^{n} x_i^2\right)\right\} * exp\left[-\sum_{i=1}^{n} \left(\sin\sqrt{\lvert x_i \rvert}\right)^2\right]$

جدول ۳: توابع محک نمایی چندگانه با اندازه ثابت

تابع	اندازه	حدود	پاسخ
$f_{17} = 4x_1^2 - 2 \cdot 1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$	۲	[−۵ ، ۵]	−۱/۰۳۱۶
$f_{18} = \left(x_2 - \frac{5 \cdot 1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos x_1 + 10$	۲	[−۵ ، ۵]	۰/۳۹۸
$f_{19} = [1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)] * [30 + (2x_1 - 3x_2)^2 * (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)]$	۲	[−۲ ، ۲]	۳

جدول ٤-۱: نتایج عملکرد الگوریتم ها برای توابع نمایی واحد

F	GA			PSO			ISABA		
	میانگین	انحراف معیار	بهینه	میانگین	انحراف معیار	بهینه	میانگین	انحراف معیار	بهینه
f_1	۰/۰۷۳	۰/۰۴۸	۰/۰۲۶	۰	۰	۰	۰	۰	۰
f_2	۰/۱۵	۰/۰۳۷	۰/۰۵۳	۹/۸e−۳	۴/۹e−۲	۱/۳e−۷	۰	۰	۰
f_3	۳۳۱/۵۱	۲۹۰/۹۶	۳۶/۵۶	۲۹۶۹۱	۱۳۶۰۶	۹۲۶۴	۰	۰	۰
f_4	۱/۷۱	۰/۳۱	۱/۲۱	۴/۴۱	۱/۰۱۱	۲/۰۸۸	۰	۰	۰
f_5	۹۶/۶۲	۴۴/۴۹	۱۹/۶۴	۷۶/۲۶	۳۴/۱۹	۷/۵۲۸	۴/۲e−۴	۱/۶۷e−۳	۳/۹۲e−۷
f_6	۰/۰۳۹	۰/۰۲۸	۰/۰۰۹۸	۶/۸۶e−۱۴	۲/۰۳e−۱۳	۲/۳e−۱۷	۱/۸۶e−۵	۹/۷۷e−۵	۸/۰۷e−۸
f_7	۰/۱۴	۰/۰۶۶	۰/۰۷	۰/۰۸۵	۰/۰۴۸	۰/۰۲۷	۱/۵۲e−۳	۳/۱۱e−۳	۲/۳۹e−۵

جدول٤-۲: نتایج عملکرد الگوریتم ها برای توابع نمایی واحد

F	BA			HDBA			DE		
	میانگین	انحراف معیار	بهینه	میانگین	انحراف معیار	بهینه	میانگین	انحراف معیار	بهینه
f_1	۵۷۳۸۵	۱۷۰۳۷	۳۵۱۱۸	۴۳۵	۱۴۱۷	۰/۰۰۰۲	۰	۰	۰
f_2	۱/۰۳e۱۳	۴/۷۱e۱۳	۹۶/۰۱	۳۲/۴۳	۲۷/۸۷	۳/۰۶	۵/۴۸e−۱۰	۱/۵۴e−۱۰	۳/۲۱e−۱۰
f_3	۱۰۲۷۷۷۲۱	۹۳۱۷۵۰۴	۱۴۴۸۳۲۶	۳۱۴۵۰۶۷	۲۹۹۴۱۲۶	۹۴۱۱۴۳	۶۴۶۷۰۳	۱۱۱۰۵۶	۳۹۷۵۴۵
f_4	۷۰/۹۷	۹/۴۶	۵۳/۱۴	۳۷/۶۷	۶/۹۹	۲۳/۱۴	۰/۹۴	۰/۲	۰/۶۱
f_5	####	####	####	۲۸۷۲۴۰۹	۶۱۹۴۳۳۰	۱۸۵۴	۸۱/۰۱	۳۵/۰۹۹	۴۴/۲۶
f_6	۶۲۰۳۵	۱۹۹۸۹	۳۱۵۶۳	۳۲۳۴	۴۵۸۶	۷۴	۱/۱۸e−۱۵	۵/۴۵e−۱۶	۴/۱۳e−۱۶
f_7	۲۴/۹	۱۸/۶	۳/۳۴	۴۹۰۰۰	۹۳۰۰۰	۱۰/۷	۰/۰۶۱۶	۰/۰۱۱۴	۰/۰۳۶

جدول ۵-۱: نتایج عملکرد الگوریتم ها برای توابع نمایی چندگانه

F	GA			PSO			ISABA		
	میانگین	انحراف معیار	بهینه	میانگین	انحراف معیار	بهینه	میانگین	انحراف معیار	بهینه
f_8	−۱/۰۶e۴	۲/۹۶e۲	−۱e۴	−۱/۰۳e۴	۱/۳۴e۳	−۷/۸۴e۳	−۱/۲۲e۴	۰/۵۳	−۱/۲۱e۴
f_9	۲/۶۷e−۲	۱/۲۶e−۲	۱/۰۱e−۲	۸/۱۵e۱	۲/۳e۱	۳/۶۸e۱	۳/۹۳e−۵	۲/۷۷e−۵	۱/۱۹e−۵
f_{10}	۴/۸e−۲	۱/۴۹e−۲	۲/۴۴e−۲	۲/۸۶e−۲	۸/۷۶e−۳	۶/۸۵e−۳	۳/۴e−۴	۸/۳۸e−۵	۲/۵۳e−۴
f_{11}	۸/۱۷e−۲	۴/۳۴e−۲	۲/۰۳e−۲	۳/۵۵e−۲	۵/۵۹e−۲	۷/۷۷e−۱۶	۱/۴۸e−۸	۱/۲۲e−۸	۵/۵۶e−۹
f_{12}	−۱/۰۲	۱/۷۵e−۳	−۱/۳	۰/۴۷	۱/۶۷	−۰/۶۱۳	−۸/۲۸e−۱	−۱/۸۶e−۱	−۳/۷۴e−۱
f_{13}	۲/۱۲e−۳	۳/۵۸e−۳	۱/۷۱e−۴	۳۹/۸	۴۴/۳	۲/۴۵	۴/۲e−۸	۱/۸۶e−۸	۱/۰۲e−۸
f_{14}	−۲۹/۵	۰/۱۴۹	−۲۹/۶	−۴۳/۷	۱/۷۴	−۴۶/۵	−۱۲/۷	۲/۶۶	−۸/۶۹
f_{15}	۰	۰	۰	۰	۰	۰	−۱	۰	−۱
f_{16}	۰	۰	۰	۰	۰	۰	−۹/۹۸e−۱	۹/۲۶e−۴	−۱

جدول ۵-۲: نتایج عملکرد الگوریتم ها برای توابع نمایی چندگانه

F	BA			HDBA			DE		
	میانگین	انحراف معیار	بهینه	میانگین	انحراف معیار	بهینه	میانگین	انحراف معیار	بهینه
f_8	−۳/۳۲e۱۷۷	####	−۵/۸۳e۷۸	−۸/۶۵e۳	۱/۰۴e۳	−۴/۴۸e۳	−۱/۹۴e۴	۹/۵۲e۲	−۱/۶۸e۴
f_9	۲/۲۹e۲	۸/۸e۱	۹/۷۹e۱	۳/۸۴e۲	۶/۵۴e۱	۲/۹۲e۲	۱/۶۲e۲	۸	۱/۴۶e۲
f_{10}	۱۷/۷	۰/۸۹	۱۵/۷	۱۹/۴	۰/۴۲	۱۸/۳	۱/۰۷e−۸	۲/۷۷e−۹	۵/۷۷e−۹
f_{11}	۵۷۴	۱۸۵	۲۶۹	۳۸/۲	۶۱/۷	۱/۰۷	۱/۰۹e−۱۴	۹/۳۹e−۱۵	۱/۱۱e−۱۵
f_{12}	۶/۳۹e۷	۶/۵۴e۷	۱/۰۵e۵	۶۸/۲	۱۵۵	−۰/۶۰۵	−۰/۶۱۳	۳/۸۲e−۱۴	−۰/۶۱۳
f_{13}	۱/۵۴e۸	۱/۵۸e۸	۱/۸۱e۷	۲۹۶	۶۶۶	۸/۲۱e−۱۰	۱/۴۱e−۱۴	۶/۹۷e−۱۵	۲/۹e−۱۵
f_{14}	−۱۶/۲	۱/۵	−۱۸/۸	−۲۰/۶	۲/۳۷	−۲۵/۶	−۳۲/۳	۰/۹۱۲	−۳۳/۹
f_{15}	۰	۰	۰	۰	۰	۰	۰	۰	۰
f_{16}	۰	۰	۰	۰	۰	۰	۰	۰	۰

جدول ٦-۱: نتایج عملکرد الگوریتم ها برای توابع نمایی چندگانه با اندازه ثابت

F	GA			PSO			ISABA		
	میانگین	انحراف معیار	بهینه	میانگین	انحراف معیار	بهینه	میانگین	انحراف معیار	بهینه
f_{17}	−۱/۰۳۱۶	۰	−۱/۰۳۱۶	−۱/۰۳۱۶	۰	−۱/۰۳۱۶	−۱/۰۳۱۶	۰	−۱/۰۳۱۶
f_{18}	۰/۳۹۸	۰	۰/۳۹۸	۰/۳۹۸	۰	۰/۳۹۸	۰/۳۹۸	۰	۰/۳۹۸
f_{19}	۳	۶/۵e−۵	۳	۳	۹/۱۲e−۵	۳	۳	۰	۳

جدول ٦-٢: نتایج عملکرد الگوریتم ها برای توابع نمایی چندگانه با اندازه ثابت

F	BA			HDBA			DE		
	میانگین	انحراف معیار	بهینه	میانگین	انحراف معیار	بهینه	میانگین	انحراف معیار	بهینه
f_{17}	−۱/۰۳۱۶	۰	−۱/۰۳۱۶	−۱/۰۳۱۶	۰	−۱/۰۳۱۶	−۱/۰۳۱۶	۰	−۱/۰۳۱۶
f_{18}	۰/۳۹۸	۰	۰/۳۹۸	۰/۳۹۸	۰	۰/۳۹۸	۰/۳۹۸	۰	۰/۳۹۸
f_{19}	۹	۲۰/۵۴	۳	۱e۱۳	۵/۳۸e۱۳	۳	۳	۰/۰۰۰۱	۳

در جداول ۴ تا ۶ ، نتایج عملکرد الگوریتم بهبود یافته خفاش و تعداد دیگری از الگوریتم‌های بهینه‌سازی در توابع محک [۱۵]، نشان داده شده است. مقادیر ذکر شده در نتایج فوق از ۳۰ بار اجرای هر الگوریتم در ۲۰۰۰ تکرار بدست آمده است، مقایسه میانگین، انحراف معیار و مقدار بهینه بدست آمده از اجرای الگوریتم‌ها نشان دهنده عملکرد مطلوب الگوریتم پیشنهادی می‌باشد. توانایی الگوریتم (ISABA) در یافتن بهینه اصلی هر تابع در نتایج بدست آمده مشهود می‌باشد، نتایج نشان دهنده برتری الگوریتم پیشنهادی نسبت به الگوریتم‌های دیگر می‌باشد.

مراجع

1- A.E. Eiben and J.E. Smith. *Introduction to Evolutionary Computing*. Springer-Verlag, Berlin,2003.

2- C. Darwin. *The origin of species*. John Murray, London, UK, 1859.

3- Yang, X.S. *A new Metaheuristic Bat-Inspired Algorithm, in: Nature Inspired Cooperative Strategies for Optimization, studies in Computational Intelligence Vol*. 284, Springer Berlin,2010, pp. 65-74.

4- Komarasamy, G., and Wahi, A.,(2012). *An optimized K-means clustering techniqueusing bat algorithm, European J*. Scientific Research, Vol. 84, No. 2, pp.263-273.

5- Lin, J. H., Chou, C. W., Yang, C. H.,Tsai, H. L., (2012). *A chaotic Levy flight bat algorithm for parameter estimation in nonlinear dynamic biological systems, J*. Computer and Information Technology, Vol. 2, No. 2, pp. 56–63.

6- Nakamura, R. Y. M., Pereira, L. A. M., Costa, K. A., Rodrigues, D., Papa, J. P., Yang, X. S., *BBA: A binary bat algorithm for feature selection*, in: 25th SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI), 22-25 Aug. 2012, IEEE Publication, pp. 291-297.

7- Jamil, M., Zepernic, H.-J., and Yang, X. S., (2013). *Improved bat algorithm for global optimization, Applied Soft Computing*, (2013, submitted).

8- Zhang, J. W., and Wang, G. G., (2012). *Image matching using a bat algorithm with mutation, Applied Mechanics and Materials (Editted by Z. Y. Du and Bin Liu), Vol*. 203, No. 1, pp. 88–93.

9- Wang, Gaige, and Guo, Lihong, (2013*). A novel hybrid bat algorithm with harmony search for global numerical optimization*. Journal of Applied Mathematics, (in press).

10- Fister Jr., I., Fister, D., and Yang, X. S., (2013). *A hybrid bat algorithm* .Elekrotehni˘ski Vestnik (English Edition), (2013, submitted).

11- Fister, I., Fister Jr., I., Yang, X. S., and Brest, J., (2013). *On the representation of individual s using quaternions in swarm intelligence and evolutionary computation, IEEE Trans*. Evol. Computation, (2013, submitted).

12- A.H. Gandomi, X.S. Yang, A.H. Alavi, and S. Talatahari. *Bat algorithm for constrained optimization tasks*. Neural Computing & Applications, pages 1-17, 2012.

13- P.W. Tsai, J.S. Pan, B.Y. Liao, M.J. Tsai, and V. Istanda. *Bat algorithm inspired algorithm for solving numerical optimization problems.* Applied Mechanics and Materials, 148:134-137, 2012.

14- X.S. Yang. *Review of meta-heuristics and generalised evolutionary walk algorithm*. International Journal of Bio-Inspired Computation, 3(2):77-84, 2011.

15- S.A. Mirjalili, S.M. Mirjalili, A. Lewis. *Grey Wolf Optimizer*. Advances in Engineering Software 69 (2014) 46–61.

16- van den Bergh F, Engelbrecht A. *A study of particle swarm optimization particle trajectories*. Inf Sci 2006;176:937–7