ارزیابی و مقایسه عملکرد الگوریتم گله اسب با برخی الگوریتم‌های فراابتکاری

جلال ایزی[۱]، علی اکبر نقابی[۲,*]

چکیده

الگوریتم‌های فراابتکاری به دلیل توانایی غلبه بر بسیاری از مشکلات در بهینه‌سازی سنتی، به ابزاری محبوب برای حل مسائل متعدد در کاربردهای دنیای واقعی تبدیل‌شده‌اند. عملکرد این الگوریتم‌ها در مسائل مختلف متفاوت است لذا لازم است که ارزیابی دقیقی از عملکرد آن‌ها صورت گیرد. یکی از الگوریتم‌های فراابتکاری که اخیراً توجه زیادی را به خود جلب کرده است، الگوریتم گله اسب است که از رفتار اسب‌ها در سنین مختلف الهام گرفته شده است. هدف از این پژوهش، مقایسه و ارزیابی عملکرد الگوریتم گله اسب با برخی از الگوریتم‌های فراابتکاری دیگر برای حل مسائل پیچیده است. در مطالعه حاضر، عملکرد الگوریتم گله اسب با ۹ الگوریتم فراابتکاری دیگر شامل الگوریتم‌های کلونی مورچگان، چهل دزد و علی‌بابا، شیر مورچه، خفاش، جستجوی کلاغ، کرم شب‌تاب، ژنتیک، ازدحام ذرات و الگوریتم نهنگ مقایسه شده است. در این ارزیابی از ۱۰ تابع تست استاندارد استفاده شده و مقایسه عملکرد الگوریتم ها بر اساس سه معیار بهترین جواب، انحراف معیار و زمان اجرا در ابعاد ۵۰۰، ۱۰۰۰ و ۲۰۰۰ انجام‌شده است. نتایج شبیه‌سازی نشان می‌دهد که با توجه به تعداد پارامترهای زیادی که الگوریتم گله اسب دارد، یکی از چالش‌های این الگوریتم تنظیم کردن پارامترهای آن می‌باشد. همچنین، در ابعاد بالا، الگوریتم گله اسب عملکرد خوبی نسبت به سایر الگوریتم‌های فراابتکاری مقایسه شده ندارد.

کلمات کلیدی: الگوریتم‌های فراابتکاری، بهینه‌سازی، الگوریتم گله اسب، الگوریتم نهنگ، الگوریتم چهل دزد و علی‌بابا.

تاریخ ارسال : ۱۴۰۳/۰۵/۱۹

تاریخ پذیرش: ۱۴۰۳/۰۶/۲۷

۱-مقدمه

حل بسیاری از مسائل دنیای واقعی مبتنی بر جستجو و بهینه‌سازی است و به دلیل پیچیدگی‌هایی مانند عدم تحدب، غیرخطی بودن، ماهیت مختلط متغیرها و ابعاد بزرگ مسائل، الگوریتم‌های کلاسیک کارایی لازم را برای حل چنین مسائلی ندارند [۱]. از طرفی تاکنون الگوریتم ریاضیاتی شناخته‌شده‌ای برای یافتن راه‌حل بهینه برای همه این مسائل در یک‌زمان محاسباتی محدود پیدا نشده است[۲, ۳]. بنابراین برای حل چنین مسائلی الگوریتم‌های فراابتکاری استفاده می‌شوند. این الگوریتم‌ها اگرچه فاقد پایه‌های ریاضی قوی می‌باشند اما دررسیدن به یک‌راه‌حل تقریبی در مدت‌زمان معقول مناسب هستند. این الگوریتم‌ها یافتن راه‌حل بهینه دقیق را تضمین نمی‌کنند، اما می‌توانند به یک‌راه‌حل تقریباً بهینه با روشی کارآمد منجر شوند [۴]. با توجه به این موارد و نیز سهولت پیاده‌سازی، استفاده از این الگوریتم‌ها در حوزه‌های مختلف محبوبیت پیدا کرده است. بیشتر روش‌های فراابتکاری ماهیت تصادفی دارند و از یک اصل طبیعی، فیزیکی یا بیولوژیکی شبیه به جستجو یا فرآیند بهینه‌سازی تقلید می‌کنند [۵]. درواقع الگوریتم‌های فراابتکاری می‌توانند با ارائه‌ی یک‌راه حل عمومی مسئله را با سرعت و دقتِ معقولی حل کنند [۶]. مطالعات متعددی در مورد توسعه این الگوریتم‌ها انجام‌شده است و طیف گسترده‌ای از این الگوریتم‌ها در دو دهه اخیر

* Aa_neghabi@iaus.ac.ir [۲]

۱. دانشجوی دکتری، گروه مهندسی کامپیوتر و فناوری اطلاعات، دانشگاه آزاد اسلامی واحد سبزوار، سبزوار، ایران

۲. استادیار، گروه مهندسی کامپیوتر و فناوری اطلاعات، دانشگاه آزاد اسلامی واحد سبزوار، سبزوار، ایران

ظهور کرده‌اند که بسیاری از آنان به‌طور فزآینده‌ای محبوب شده‌اند [۷]. یکی از الگوریتم‌های فراابتکاری که اخیراً توجه زیادی را به خود جلب کرده است، الگوریتم گله اسب است. الگوریتم فراابتکاری گله اسب[1] [۸] در سال ۲۰۲۱ ارائه شده و از رفتار اسب‌ها در سنین مختلف الهام گرفته است. این الگوریتم از شش رفتار مهم اسب‌ها پیروی می‌کند که عبارت‌اند از: رفتن به چراگاه، زندگی سلسله‌مراتبی، قابلیت اجتماعی، تقلید کردن، مکانیسم دفاعی و پرسه زدن. از مزایای این الگوریتم می‌توان به قابلیت حل مسائل پیچیده، قابلیت یافتن راه‌حل‌های بهینه و سرعت‌بالا در مقایسه با برخی از الگوریتم‌های فراابتکاری دیگر مانند الگوریتم ژنتیک اشاره کرد.

با توجه به اهمیت الگوریتم‌های فراابتکاری و کاربرد وسیع آن‌ها در علوم مختلف بهتر است تا عملکرد این الگوریتم‌ها مورد ارزیابی قرار گیرد. الگوریتم گله اسب اولین بار در پژوهش میر نعیمی و همکاران [۸] معرفی‌شده است. در آن پژوهش عملکرد الگوریتم گله اسب با الگوریتم‌های ملخ[2]، چند نظمی[3]، شعله پروانه[4]، تکاملی تفاضلی[5] و الگوریتم گرگ خاکستری[6] مورد مقایسه قرارگرفته است. هدف از پژوهش حاضر، مقایسه عملکرد الگوریتم گله اسب با ۹ الگوریتم دیگر شامل الگوریتم های کلونی مورچگان[7] [۹]، چهل دزد و علی‌بابا[8] [۱۰]، شیر مورچه[9] [۱۱]، خفاش[10] [۱۲]، جستجوی کلاغ[11] [۱۳]، کرم شب‌تاب[12] [۱۴]، الگوریتم ژنتیک[13] [۱۵]، الگوریتم ازدحام ذرات[14] [۱۶] و الگوریتم نهنگ[15] [۱۷] می باشد. برای بررسی عملکرد الگوریتم‌های انتخابی ۱۰ تابع استاندارد در نظر گرفته‌شده است. همچنین، از سه معیار بهترین پاسخ[16]، انحراف معیار[17] و زمان اجرا[18] استفاده می‌شود. با مقایسه عملکرد الگوریتم گله اسب با این الگوریتم‌ها، نقاط قوت و ضعف هر یک شناسایی‌شده و به بهبود کاربرد آن‌ها در حوزه‌های مختلف کمک خواهد شد.

۲-مرور ادبیات تحقیق

۲-۱-الگوریتم‌های فراابتکاری

الگوریتم‌های تقریبی به دو دسته الگوریتم‌های ابتکاری و فراابتکاری تقسیم می‌شوند. الگوریتم‌های ابتکاری با دو چالش اصلی مواجه هستند، چالش اول، گیر افتادن در نقاط بهینه محلی و چالش دوم، همگرایی زودرس به این نقاط می‌باشد. الگوریتم‌های فراابتکاری به‌منظور پیشگیری از این مشکلات و بهبود عملکرد ارائه‌شده‌اند. درواقع الگوریتم‌های فراابتکاری یکی از انواع الگوریتم‌های بهینه‌سازی تقریبی می‌باشند که دارای راهکارهای برون‌رفت از نقاط بهینه محلی بوده و قابلیت کاربرد در طیف گسترده‌ای از مسائل را دارند [۱۸]. در دهه‌های اخیر انواع مختلفی از این نوع الگوریتم‌ها توسعه‌یافته است. واژه «فراابتکاری» را اولین بار «گلوور[19] » [۱۹] در سال ۱۹۸۶ به کاربرد که از ترکیب دو واژه یونانی «متا» و «هیوریستیک» ساخته‌شده است. پیشوند «متا» به معنای «فراتر» یا «در سطحی بالاتر» است و کلمه «هیوریستیک» به معنای «یافتن» است. اخیراً بسیاری از الگوریتم‌های فراابتکاری با موفقیت برای حل بسیاری از مسائل استفاده‌شده است. جذابیت استفاده از این الگوریتم‌ها برای حل مسائل پیچیده و بسیار سخت است که بهترین راه‌حل‌ها را حتی برای مسئله‌هایی با مقیاس بسیار بزرگ در مدت‌زمان کم به دست می‌آورند. مسائل بهینه‌سازی، از جنبه‌های مختلفی همچون تک هدفه یا چندهدفه بودن تابع هدف، پیوسته یا گسسته بودن متغیرهای تصمیم، مقید یا نامقید بودن متغیرهای تصمیم و غیره می‌توان دسته‌بندی کرد. به دلیل پیچیدگی این مسائل، استفاده از الگوریتم‌های دقیق برای حل آن‌ها به‌خصوص در مقیاس‌های بزرگ و ارائه جواب دقیق‌تر در زمان موردقبول با چالش همراه است [۲۰, ۲۱].

[1] Horse herd Optimization Algorithm (HOA)
[2] Grass Hopper Optimization Algorithm (GOA)
[3] Multi-Verse Optimizer (MVO)
[4] Moth-flame Optimization Algorithm (MFO)
[5] Differential Evolution Algorithm (DEA)
[6] Grey Wolf Optimizer (GWO)
[7] Ant Colony Optimization (ACO)
[8] Ali Baba and the Forty Thieves (AFT)
[9] Ant Lion Optimizer (ALO)
[10] Bat Algorithm (BA)
[11] Crow Search Algorithm (CSA)
[12] Firefly Algorithm (FA)
[13] Genetic Algorithm (GA)
[14] Particle Swarm Optimization (PSO)
[15] Whale Optimization Algorithm (WOA)
[16] Best Cost
[17] Standard Deviation (SD)
[18] Run Time
[19] Glover

الگوریتم‌های فراابتکاری یک‌راه‌حل عملی و ظریف برای حل بسیاری از این مسائل ارائه می‌دهند و برای دستیابی به راه‌حل‌های تقریباً بهینه در زمان‌های اجرای عملی برای مسائل بهینه‌سازی سخت طراحی‌شده‌اند [۲۲]. الگوریتم‌هایی همچون ژنتیک، بهینه‌سازی ازدحام ذرات، بهینه‌سازی کلونی مورچگان ازجمله الگوریتم‌های فراابتکاری مشهور می‌باشند. در ادامه الگوریتم‌های فراابتکاری موردبحث در این پژوهش معرفی می‌شوند.

۲-۲- مروری بر الگوریتم‌های استفاده‌شده

در این بخش توضیح مختصری در خصوص نحوه‌ی عملکرد الگوریتم‌هایی که در این پژوهش استفاده‌شده است، ارائه خواهد شد. این الگوریتم‌ها شامل الگوریتم‌های گله اسب، کلونی مورچگان، چهل دزد و علی‌بابا، شیرمورچه، خفاش، جستجوی کلاغ، کرم شب‌تاب، ژنتیک، ازدحام ذرات و نهنگ می‌باشند.

۲-۲-۱- الگوریتم گله اسب

الگوریتم گله اسب از رفتار طبیعی اسب‌ها در طبیعت الهام گرفته و برای حل مسائل بهینه‌سازی با بُعدهای بالا پیشنهادشده است. این الگوریتم عملکردهای اجتماعی اسب‌ها در سنین مختلف را با استفاده از شش ویژگی مهم تقلید می‌کند که عبارت‌اند از: رفتن به چراگاه[۱]، زندگی سلسله‌مراتبی[۲]، قابلیت اجتماعی[۳]، تقلید کردن[۴]، مکانیسم دفاعی[۵] و پرسه زدن[۶]. رابطه (۱) حرکت اسب‌ها در هر تکرار را نشان می‌دهد.

$$x^{iter,AGE} = \vec{v}_m^{iter,AGE} + x_m^{(iter-1),AGE}, \quad AGE = \alpha, \beta, \gamma, \delta \qquad (۱)$$

که در آن، $x^{iter,AGE}$ موقعیت اسب m ام، AGE محدوده سنی اسب در نظر گرفته‌شده، $iter$ تکرار فعلی و $\vec{v}_m^{iter,AGE}$ بردار سرعت این اسب را نشان می‌دهد. اسب‌ها رفتارهای مختلفی را در سنین مختلف از خود نشان می‌دهند. حداکثر طول عمر یک اسب در حدود ۲۵ تا ۳۰ سال است. در این رابطه، اسب‌های دلتا[۷] در محدوده سنی ۵ - ۰ سال، اسب‌های گاما[۸] در محدوده سنی ۱۰ – ۵ سال، اسب‌های بتا[۹] در محدوده سنی ۱۵ - ۱۰ سال و سن اسب‌های آلفا[۱۰] بیشتر از ۱۵ سال است. یک ماتریس جامع از راه‌حل‌ها باید در هر تکرار ایجاد می‌شود تا اسب‌ها انتخاب گردند. در این راستا، ابتدا ماتریس بر اساس بهترین راه‌حل‌ها مرتب می‌شود. سپس از بالای ماتریس مرتب‌شده ۱۰ درصد اولیه از کل اسب‌ها به‌عنوان اسب‌های α انتخاب می‌شوند. ۲۰ درصد بعدی در گروه اسب‌های β قرار می‌گیرند و اسب‌های γ و δ به ترتیب ۳۰ و ۴۰ درصد از مابقی ماتریس راه‌حل‌ها را تشکیل می‌دهند. رابطه (۲) بردار حرکت اسب‌ها در سنین مختلف در طول هر چرخه‌ی الگوریتم را نشان می‌دهد.

$$\begin{aligned}
\vec{v}_m^{iter,\alpha} &= \mathrm{G}_m^{iter,\alpha} + D_m^{iter,\alpha} \\
\vec{v}_m^{iter,\beta} &= G_m^{iter,\beta} + H_m^{iter,\beta} + S_m^{iter,\beta} + D_m^{iter,\beta} \\
\vec{\mathrm{v}}_m^{iter,\gamma} &= G_m^{iter,\gamma} + H_m^{iter,\gamma} + \mathrm{S}_m^{iter,\gamma} + I_m^{iter,\gamma} + D_m^{iter,\gamma} + R_m^{iter,\gamma} \\
\vec{v}_m^{iter,\delta} &= G_m^{iter,\delta} + I_m^{iter,\delta} + R_m^{iter,\delta}
\end{aligned} \qquad (۲)$$

رابطه (۲) نشان می‌دهد که اسب‌ها در چهار گروه سنی تقسیم‌بندی می‌شوند. $\vec{v}_m^{iter,\alpha}$ بردار حرکت اسب‌ها در گروه آلفا را نشان می‌دهد به‌طوری‌که اسب‌های گروه آلفا دارای ویژگی چریدن ($G_m^{iter,\alpha}$) و دفاعی ($D_m^{iter,\alpha}$) می‌باشند. اسب‌های گروه بتا دارای ویژگی‌های چریدن ($G_m^{iter,\beta}$)، زندگیسلسله مراتبی ($H_m^{iter,\beta}$)، قابلیت اجتماعی ($S_m^{iter,\beta}$) و ویژگی دفاعی ($D_m^{iter,\beta}$) می‌باشند، $\vec{v}_m^{iter,\beta}$ بردار حرکت اسب‌ها در گروه بتا را نشان می‌دهد. $\vec{v}_m^{iter,\gamma}$ بردار حرکت اسب‌ها در گروه گاما را نشان می‌دهد به‌طوری‌که اسب‌های گروه بتا دارای ویژگی‌های چریدن ($G_m^{iter,\gamma}$)، زندگیسلسله مراتبی ($H_m^{iter,\gamma}$)، قابلیت اجتماعی ($S_m^{iter,\gamma}$)، تقلید کردن($I_m^{iter,\gamma}$)، مکانیسم دفاعی ($D_m^{iter,\gamma}$) و ویژگی پرسه زدن ($R_m^{iter,\gamma}$) می‌باشند. اسب‌های گروه دلتا دارای ویژگی‌های چریدن ($G_m^{iter,\delta}$)، تقلید کردن ($I_m^{iter,\delta}$) و پرسه زدن ($R_m^{iter,\delta}$) می‌باشند، $\vec{v}_m^{iter,\delta}$ بردار حرکت اسب‌ها در گروه دلتا را نشان می‌دهد.

[1] Grazing(G)
[2] Hierarchy(H)
[3] Sociability(S)
[4] Imitation(I)
[5] Defense mechanism(D)
[6] Roam(R)

[7] δ (Delta)
[8] γ (Gama)
[9] β (Beta)
[10] α (Alpha)

۲-۲-۱-۱- مراحل الگوریتم گله اسب

۲-۲-۱-۱-۱- رفتن به چراگاه

اسب‌ها حیوانات چراگاهی هستند که از گیاهان، چمن‌زارها، علف‌زارها و مواد تغذیه‌ای دیگر تغذیه می‌کنند. آن‌ها ۱۶ تا ۲۰ ساعت در روز در چراگاه در حال چریدن می‌باشند و زمان استراحت کوتاهی دارند. الگوریتم گله اسب مدل چراگاه اطراف هر اسب را با ضریب g مدل‌سازی می‌کند به‌گونه‌ای که هر اسب در مناطق خاصی در چراگاه مشغول چرا می‌باشد. به‌طوری‌که رابطه (۳) نشان می‌دهد که هر اسب در مناطق خاصی چرا می‌کند.

$$\mathrm{G}_m^{iter,AGE} = g_{iter}(u + p \times l)\left[x_m^{(iter-1)}\right], \quad \text{(۳-الف)}$$

$$AGE = \alpha, \beta, \gamma, \delta$$

$$g_m^{iter,AGE} = \mathrm{g}_m^{(iter-1),AGE} \times \omega_g \quad \text{(۳-ب)}$$

که در آن، $G_m^{iter,AGE}$ حرکت اسب m ام در چراگاه را نشان می‌دهد. $x_m^{(iter-1),AGE}$حرکت اسب m ام در مرحله قبل می‌باشد. $g_m^{(iter-1),AGE}$ چریدن اسب‌ها در مرحله قبل را نشان می‌دهد. در مرحله چریدن اسب‌ها، هر اسب دارای یک وزن ثابت است که با ω_g نشان داده می‌شود. u و l فضای بالا و پایین جستجو را نشان می‌دهد. متغیر p یک عدد تصادفی بین ۰ و ۱ می‌باشد.

۲-۲-۱-۱-۲-زندگی سلسله‌مراتبی

اسب‌ها زندگی خود را زیر نظر یک رهبر می‌گذرانند. ممکن است یک اسب نر بالغ یا یک اسب ماده مسئول رهبری در گله‌های اسب باشد که در قانون سلسله‌مراتبی اتفاق می‌افتد. در این حالت، ضریب H در الگوریتم گله اسب به‌عنوان باتجربه‌ترین و قوی‌ترین اسب در نظر گرفته می‌شود. اسب‌هایی که از قانون سلسله‌مراتبی پیروی می‌کنند معمولاً اسب‌هایی هستند که در سنین میانی گروه‌های بتا و گاما (بین ۵ تا ۱۵ سال) می‌باشند. رابطه (۴) قانون سلسله‌مراتبی را نشان می‌دهد.

$$\mathrm{H}_m^{iter,AGE} = h_m^{iter,AGE}\left[X_*^{(iter-1)} - \mathrm{X}_m^{(iter-1)}\right], \quad \text{(۴-الف)}$$

$$AGE = \alpha, \beta \; and \; \gamma \quad \text{(۴-ب)}$$

که در آن $H_m^{iter,AGE}$ موقعیت اسب رهبر می‌باشد. $h_m^{(iter-1),AGE}$ موقعیت اسب رهبر در مرحله قبل را نشان می‌دهد. $X_m^{(iter-1)}$ موقعیت اسب موردنظر در مرحله قبل می‌باشد. $X_*^{(iter-1)}$ بهترین موقعیتی که کل اسب‌ها در مرحله قبل داشته‌اند را نشان می‌دهد. AGE محدوده سنی هر اسب می‌باشد. هر اسب رهبر دارای یک وزن ثابت است که با ω_h نشان داده می‌شود.

۲-۲-۱-۱-۳-قابلیت اجتماعی

اسب‌ها به زندگی اجتماعی نیاز دارند و با یکدیگر زندگی می‌کنند. زندگی اجتماعی اسب‌ها امنیت آن‌ها را تضمین کرده است، زیرا آن‌ها توسط شکارچیان شکار می‌شوند و زندگی اجتماعی احتمال شکار شدن هرکدام از اسب‌ها را کاهش داده و شانس زنده ماندن آن‌ها را افزایش می‌دهد. اسب‌ها به‌ندرت احساس تنهایی را دوست دارند. این رفتار به‌عنوان یک حرکت به سمت موقعیت میانگین دیگر اسب‌ها در نظر گرفته می‌شود و با فاکتور s نشان داده می‌شود، رابطه (۵) نشان می‌دهد که اسب‌ها در سنین ۵ تا ۱۵ سال به گله علاقه دارند.

$$\mathrm{S}_m^{iter,AGE} = s_m^{iter,AGE}\left[\left(\frac{1}{N}\sum_{j=1}^{n} x_j^{(iter-1)}\right) - X_m^{(iter-1)}\right], \quad \text{(۵-الف)}$$

$$AGE = \beta, \gamma \quad \text{(۵-ب)}$$

$$s_m^{iter,AGE} = \mathrm{s}_m^{(iter-1),AGE} \times \omega_s$$

$S_m^{iter,AGE}$بردار حرکت اجتماعی i امین اسب و جهت‌گیری اسب مربوطه به سمت گله در تکرار را نشان می دهد. $s_m^{(iter-1),AGE}$ قابلیت اجتماعی اسب در مرحله قبل را نشان می‌دهد. N همچنین تعداد کل اسب‌ها را نشان می‌دهد و AGE محدوده سنی هر اسب می‌باشد. ضریب s برای اسب‌ها در تحلیل حساسیت پارامترها محاسبه می‌شوند. اسب‌ها در قابلیت اجتماعی دارای یک وزن ثابت می‌باشند که با ω_s نشان داده می‌شود.

۲-۲-۱-۱-۴- تقلید کردن

اسب‌ها از یکدیگر تقلید می‌کنند و عادات خوب و بد یکدیگر را یاد می‌گیرند، همانند پیدا کردن محل مناسب چراگاه، رفتار تقلید اسب‌ها نیز به‌عنوان فاکتور i در الگوریتم در نظر گرفته می‌شود. اسب‌های جوان سعی می‌کنند از دیگران تقلید کنند و این ویژگی در تمام مدت زندگی آن‌ها محو نشده و (۶-الف) چگونگی تقلید کردن را نشان می‌دهد.

$$I_m^{iter,AGE} = i_m^{iter,AGE}\left[\left[\left(\frac{1}{pN}\sum_{j=1}^{pN} x\right]\right]_j^{(iter-1)}\right) - X^{(iter-1)}\right], \quad \text{(۶-ب)}$$

$$AGE = \gamma$$

به‌طوری‌که $I_m^{iter,AGE}$ بردار حرکت i امین اسب به سمت میانگین بهترین اسب‌ها با موقعیت‌های X است. pN تعداد اسب‌هایی با بهترین مکان را نشان می‌دهد. علاوه بر این،

همان‌طور که قبلاً نشان داده‌شده است، i ضریب کاهش در هر چرخه برای هر $iter$ است. $X^{(iter-1)}$ موقعیت اسب در مرحله قبل را نشان می‌دهد. $i_m^{(iter-1),AGE}$ تقلید کردن اسب m ام در مرحله قبل را نشان می‌دهد. اسب‌ها در مرحله تقلید کردن دارای یک وزن ثابت می‌باشند که با ω_i نشان داده می‌شود.

۲-۲-۱-۱-۵- مکانیسم دفاعی

واکنش اسب‌ها بازتابی از این واقعیت است که آن‌ها قربانی شکارچیان شده‌اند. آن‌ها با نشان دادن واکنش مبارزه یا گریز از خود دفاع می‌کنند. اولین واکنش آن‌ها فرار است و از محیط‌های خطرناک که در آن دشمنانی مانند گرگ‌ها به‌طور غریزی وجود دارند، اجتناب کنند. سیستم دفاعی اسب‌ها در الگوریتم گله اسب بدترین پاسخ‌ها را نشان می‌دهد که از بهینه‌ترین حالت بسیار دور هستند. همان‌طور که در بالا ذکر شد، اسب‌ها باید فرار کنند یا با دشمنانشان بجنگند. چنین مکانیسم دفاعی در تمام طول عمر یک اسب جوان یا بالغ، هر زمان که ممکن باشد، وجود دارد. مکانیسم دفاعی اسب‌ها با ضریب منفی می‌باشد که در رابطه (۷) نشان داده‌شده است.

$$D_m^{iter,AGE} = d_m^{iter,AGE}\left[\left(\frac{1}{qN}\sum_{j=1}^{qN} x_j^{(iter-1)}\right) - X^{(iter-1)}\right], \quad \text{(۷-الف)}$$

$$AGE = \alpha, \beta \; and \; \gamma \quad \text{(۷-ب)}$$

$$d_m^{iter,AGE} = d^{(iter-1),AGE} \times \omega_d$$

که در آن $D_m^{iter,AGE}$ بردار فرار i امین اسب از میانگین اسب‌ها با بدترین مکان‌ها را نشان می‌دهد، که توسط بردار X نشان داده‌شده است. همچنین qN تعداد اسب‌های دارای بدترین مکان را نشان می‌دهد. معمولاً q برابر با ۲۰ درصد از کل اسب‌ها در نظر گرفته می‌شود. $X^{(iter-1)}$ موقعیت اسب در مرحله قبل را نشان می‌دهد. $d_m^{(iter-1),AGE}$ مکانیسم دفاعی اسب m امین در مرحله قبل را نشان می‌دهد. اسب‌ها در مرحله دفاع کردن دارای یک وزن ثابت می‌باشند که با ω_d نشان داده می‌شود.

۲-۲-۱-۱-۶- پرسه زدن

اسب‌ها در طبیعت در جستجوی غذا می‌چرند. بیشتر اسب‌ها در اسطبل نگهداری می‌شوند، اگرچه ویژگی ذکرشده را حفظ می‌کنند. ممکن است یک اسب ناگهان به‌جایی دیگر برود. اسب‌ها شدیداً کنجکاو هستند و اغلب به همه‌جا سر می‌زنند تا چراگاه‌های جدید کشف کنند و همسایگانشان را بشناسند. دیواره‌های جانبی یک اسطبل طوری طراحی می‌شود که اسب‌ها بتوانند یکدیگر را ببینند و کنجکاوی آن‌ها در یک اسطبل برآورده شود. رابطه (۸) نشان‌دهنده پرسه زدن اسب‌ها است.

$$\mathrm{R}_m^{iter,AGE} = r_m^{iter,AGE}\, pX^{(iter-1)},\ AGE = \gamma, \delta \quad \text{(۸-الف)}$$

$$r_m^{iter,AGE} = r_m^{(iter-1),AGE} \times \omega_r \quad \text{(۸-ب)}$$

این رفتار به‌عنوان یک حرکت تصادفی شبیه‌سازی‌شده و با یک فاکتور r نشان داده می‌شود. ویژگی پرسه زدن در اسب‌ها تقریباً در سنین جوانی مشاهده می‌شود و با رسیدن به بلوغ به‌تدریج ناپدید می‌شود که در آن، $R_m^{iter,AGE}$ نشان‌دهنده بردار سرعت تصادفی i امین اسب برای یک جستجوی محلی و یک فرار از کمینه‌های محلی است و r ضریب کاهش $R_m^{iter,AGE}$ در هر چرخه را نشان می‌دهد. $r_m^{(iter-1),AGE}$پرسه زدن اسب mام در مرحله قبل را نشان می‌دهد. اسب‌ها در مرحله پرسه زدن دارای یک وزن ثابت می‌باشند که با ω_r نشان داده می‌شود.

۲-۲-۲- الگوریتم کلونی مورچگان

الگوریتم کلونی مورچگان از مطالعات و مشاهدات روی کلونی مورچگان در طبیعت الهام گرفته‌شده و اولین بار در سال (۱۹۹۲) توسط مارکو دوریگو[1] ارائه‌شده است. این الگوریتم زیرمجموعه‌ای از الگوریتم‌های هوش ازدحامی (هوش گروهی) می‌باشد و به مدل‌سازی رفتار مورچه‌های واقعی می‌پردازد. مورچه‌ها حشراتی هستند که می‌توانند گروه‌ها (کلونی‌ها) را شکل دهند. چنین رویکرد جمعیت محوری این امکان را برای الگوریتم کلونی مورچگان ایجاد می‌کند تا به حل مسائل بهینه‌سازی پویا به‌طور کاملاً کارآمد بپردازد. مورچه‌ها به‌عنوان مخلوقات خودسازمان‌ده می‌باشند. چنین ویژگی به‌عنوان هسته مسئله است زیرا این ویژگی دقیقاً همان چیزی است که باعث می‌شود حشرات به‌سرعت با شرایط متغیر محیطی‌شان به‌منظور دستیابی به اهداف از طریق تعامل سطح پایین، وفق یابند. مورچگان چشم ندارند و تعاملات آن‌ها از طریق ماده شیمیایی به نام فرومون شکل می‌گیرد که از آن برای نشان‌گذاری مسیر استفاده می‌کنند. هرچه فرومون های بیشتری در مسیر قرار گیرد مابقی مورچگان از این مسیر بیشتر استفاده می‌کنند؛ بنابراین، چنین کمیتی نشان می‌دهد که این مسیر به‌عنوان یکی از بهینه‌ترین و کوتاه‌ترین راه‌حل‌ها است [۱۸]. در این

[1] Marco Dorigo

الگوریتم هدف پیدا کردن بهترین مسیر از نقطه آغازین N (آشیانه) به نقطه مقصد F (منبع غذا) می‌باشد. الگوریتم کلونی مورچگان بر پایه رفتار طبیعی کلونی‌های مورچگان و مورچگان کارگر در آن‌ها بنانهاده شده است. فرآیند یافتن منابع غذایی در کلونی مورچگان بسیار بهینه است. زمانی که مورچه‌ها عملیات کاوش برای یافتن منابع غذایی را آغاز می‌کنند، به‌طور طبیعی یک مسیر «منطقی» و «بهینه» از آشیانه خود به منابع غذایی پیدا می‌کنند. به‌عبارت‌دیگر، جمعیت مورچگان به نحوی همیشه قادر هستند تا یک مسیر بهینه را برای تأمین منابع غذایی موردنیاز بیابند. شبیه‌سازی چنین رفتار بهینه‌ای، پایه و اساس الگوریتم بهینه‌سازی کلونی مورچگان را شکل می‌دهد [۱۹].

۲-۲-۳- الگوریتم چهل دزد و علی‌بابا

الگوریتم چهل دزد و علی‌بابا توسط مالک بریک، محمد هاشم ریالت و حسین الذوبی در سال (۲۰۲۱) پیشنهادشده است. این الگوریتم یک الگوریتم فراابتکاری الهام گرفته از داستان علی‌بابا و گنج سارقان است. شیوه پیگیری سارقان برای پیدا کردن علی‌بابا و تلاش‌های آن‌ها تبدیل به یک الگوریتم بهینه‌سازی شده است. هدف کلی این کار ارائه یک روش بهینه‌سازی جدید با تقلید از داستان علی‌بابا و چهل دزد به‌عنوان الگوی هماهنگ رفتار اجتماعی اعمال انسان است. اصول زیر که از این داستان به‌دست‌آمده است، مفروضات اساسی این الگوریتم را تشکیل می دهند:

- چهل دزد به‌صورت گروهی باهم همکاری می‌کنند و از یک نفر از سارقان راهنمایی می‌گیرند تا خانه علی‌بابا را پیدا کنند. این اطلاعات ممکن است درست باشد یا نباشد.
- چهل دزد مسافتی را از مسافت اولیه طی می‌کنند تا بتوانند خانه علی‌بابا را پیدا کنند.
- مرجانه می‌تواند بارها با روش‌های زیرکانه سارقان را فریب دهد تا به نحوی از علی‌بابا در مقابل دزدان محافظت کند. رفتارهای سارقان و مرجانه را می‌توان به‌گونه‌ای ترسیم کرد که بتوان آن‌ها را به یک تابع هدف مرتبط نمود تا بهینه شود. این امر امکان تکامل یک الگوریتم فراابتکاری را امکان‌پذیر می‌کند.

در الگوریتم چهل دزد و علی‌بابا، سه مورد اساسی که در بالا بیان شد ممکن است هنگام جستجوی سارقان برای پیدا کردن علی‌بابا رخ دهد. در هر مورد، فرض بر این است که سارقان به‌طور مؤثر در محیط اطراف علی‌بابا جستجو می‌کنند، درحالی‌که سارقان به دلیل هوش و ذکاوت مرجانه مجبور می‌شوند مکان‌های تصادفی را برای پیدا کردن علی‌بابا به جستجو بپردازند. مدل ریاضی رفتار جستجوی سارقان به این شرح می‌باشد، در فاز اول سارقان ممکن است علی‌بابا را با کمک اطلاعاتی که از شخصی به دست آورده‌اند ردیابی کنند. در این صورت مکان‌های جدید سارقان را می‌توان از رابطه (۹) به دست آورد:

$$x_{t+1}^{i} = gbest - [Td_t(best_t^i - y_t^i)r1 + Td_t\left(y_t^i - m_t^{a(i)}\right)r2]\, sgn(rand \quad (۹)$$

که در آن x_{t+1}^{i} موقعیت i امین دزد در لحظه جدید می‌باشد. y_t^i موقعیت علی‌بابا نسبت به دزد در هر تکرار است. $gbest$ نشان‌دهنده بهترین موقعیت سراسری است که تاکنون توسط هر دزدی در هر تکرار به‌دست‌آمده است، $m_t^{a(i)}$ نشان‌دهنده سطح هوش و ذکاوت مرجانه است که برای گمراه کردن سارقان می‌باشد. Td_t فاصله ردیابی سارقان در زمان در آن لحظه می‌باشد. $r2$ و $r1$ اعداد تصادفی هستند که با توزیع یکنواخت بین ۰ و ۱ می‌باشند. sgn برای تغییر جهت فرآیند جستجو است.

در فاز دوم، ممکن است سارقان متوجه شوند که فریب‌خورده‌اند، بنابراین به‌طور تصادفی فضای جستجوی علی‌بابا را کشف می‌کنند. در این صورت مکان‌های جدید سارقان را می‌توان با استفاده از رابطه (۱۰) به دست آورد:

$$x_{t+1}^{i} = Td_t\,[(u_i - l_j)rnad + l_j] \quad (۱۰)$$

به‌طوری‌که x_{t+1}^{i} مکان جدید سارقان را نشان می‌دهد، Td_t فاصله ردیابی سارقان در آن لحظه می‌باشد. l_j و u_i کران بالا و پایین جستجو را نشان می‌دهد. $rnad$ یک عدد تصادفی بین ۰ و ۱ می‌باشد.

در فاز سوم، به‌منظور بهبود ویژگی‌های اکتشاف و بهره‌برداری می‌توان از رابطه (۱۱) استفاده کرد. در این صورت می‌توان مکان‌های جدید سارقان را به دست آورد:

$$x_{t+1}^{i} = gbest + [Td_t(best_t^i - y_t^i)r1 + Td_t\left(y_t^i - m_t^{a(i)}\right)r2]\, sgn(rand - 0.5) \quad (۱۱)$$

این رابطه همانند رابطه (۹) می‌باشد با این تفاوت که علامت آن به منفی تغییر پیدا کرده است.

۲-۲-۴- الگوریتم شیرمورچه

الگوریتم شیرمورچه یا مورچه‌گیر اولین بار توسط سید علی میر جلیلی در سال (۲۰۱۵) بر اساس رفتار شیر مورچه‌ها در طبیعت، ارائه‌شده است. شیرمورچه رفتار شکار

منحصربه‌فردی دارد. شیرمورچه یک چاله مخروطی در شن ایجاد می‌کند، سپس زیر پایه مخروطی شکل پنهان می‌شود و به‌عنوان یک شکارچی نشسته و انتظار می‌کشد تا حشرات (معمولاً مورچه‌ها) در دام بیافتند. لبه مخروط به‌اندازه‌ای تیز است که حشرات به‌راحتی به کف دام فرود می‌آیند. زمانی که شیر مورچه متوجه می‌شود که یک طعمه در دام گرفتارشده است، سعی می‌کند تا آن را به دهان گرفته، به زیرخاک کشیده و شکار کند. پس از شکار طعمه، شیرمورچه باقیمانده‌ی طعمه را به بیرون از دام پرتاب می‌کند تا دام برای شکار بعدی آماده شود. الگوریتم شیرمورچه تعامل بین شیرمورچه و مورچه‌ها در دام را شبیه‌سازی می‌کند. برای مدل کردن چنین تعاملاتی، نیاز است که مورچه‌ها بر روی فضای جستجو حرکت کنند و شیرمورچه اجازه شکار با استفاده از دام را داشته باشد. ازآنجاکه مورچه‌ها در طبیعت به‌صورت تصادفی حرکت می‌کنند هنگام جستجو برای غذا، یک قدم زدن تصادفی در رابطه (۱۲) برای مدل کردن حرکت مورچه‌ها انتخاب می‌شود:

$$x(t) = [0, cumsum(2r(t_1) - 1), cumsum(2r(t2) - 1), \dots, cumsum(2r(tn - 1)) \quad (۱۲)$$

در این رابطه، $Cumsum$ مجموع تجمعی را محاسبه می‌کند؛ nحداکثر تعداد تکرار خواهد بود و همچنین در این حالت tمرحله پیاده‌روی تصادفی مورچه‌ها را نشان داده و $r(t)$یک تابع تصادفی است که از رابطه (۱۳) به دست می‌آید. در این رابطه، tنشان‌دهنده مرحله پیاده‌روی تصادفی مورچه‌ها است. همچنین، $rand$ یک عدد تصادفی است که با توزیع یکنواخت در بازه $[۰،۱]$ تولید می‌شود.

$$r(t) = \begin{cases} 1 & if\ rand > 0.5 \\ 0 & if\ rand \le 0.5 \end{cases} \quad (۱۳)$$

بعد از حرکت تصادفی مورچه‌ها، شیر مورچه‌ها با استفاده از حفره‌هایی که ایجاد کرده‌اند مورچه‌ها را درون حفره به دام می‌اندازند. همان‌طور که اشاره شد، پیاده‌روی‌های تصادفی مورچه‌ها تحت تأثیر دام‌های شیر مورچه‌ها قرار می‌گیرد. رابطه (۱۴ و ۱۵) مدل سازی ریاضی این فرضیه را نشان می‌دهد.

$$C_i^t = Antlion_j^t + C^t \quad (۱۴)$$
$$d_i^t = Antlion_i^t + d^t \quad (۱۵)$$

در این دو رابطه، C^t یک عبارت یا عامل تصحیح اضافی است. C_i^t موقعیت مورچه i ام در تکرار t ام را نشان می‌دهد. موقعیت شیر مورچه j ام انتخاب‌شده در تکرار t ام توسط $Antlion_j^t$ نشان داده می‌شود. d^t بیانگر بیشینه در تمام متغیرها در زمان t است. d_i^t موقعیت جدید مورچه i ام در تکرار t ام است که برابر است با موقعیت شیرمورچه jام در تکرار t ام به‌علاوه‌ی بردار d^t است.

۲-۲-۵- الگوریتم خفاش

الگوریتم خفاش، یک نوع الگوریتم هوش جمعی است که از رفتار خفاش‌های کوچک در زمان جهت‌یابی برای شکار، الهام گرفته‌شده است. این الگوریتم اولین بار توسط آقای یانگ در سال (۲۰۱۰) معرفی شده و بر اساس ویژگی‌های پژواک مکان‌یابی خفاش‌ها طراحی‌شده است. همچنین، الگوریتم خفاش از تکنیک تنظیم فرکانس برای افزایش تنوع راه‌حل‌ها در جمعیت استفاده می‌کند. حدود ۱۰۰۰ گونه مختلف خفاش وجود دارد که اندازه‌های آن‌ها می‌تواند بسیار متفاوت باشد. اکثر خفاش‌ها تا حدی از پژواک یابی استفاده می‌کنند. در میان همه گونه‌ها، خفاش‌های بسیار کوچک به‌طور گسترده‌تری از پژواک استفاده می‌کنند، درحالی‌که خفاش‌های بسیار بزرگ این کار را نمی‌کنند. خفاش‌های کوچک معمولاً از نوعی پژواک برای تشخیص طعمه، اجتناب از برخورد با موانع و تعیین موقعیت شکاف‌های خود در تاریکی استفاده می‌کنند. آن‌ها می‌توانند یک پالس صدای بسیار بلند منتشر کرده و به پژواکی که از اجسام اطراف بازمی‌گردد گوش دهند. پالس‌های آن‌ها در خواص متفاوت است و بسته به گونه می‌تواند با استراتژی‌های شکار آن‌ها مرتبط باشد. در الگوریتم خفاش عمدتاً از ویژگی‌های مکان‌یابی صوتی خفاش استفاده می‌شود تا بتوانند برخی از آن‌ها را با تابع هدف یک مسئله بهینه‌سازی مرتبط کنند و امکان فرمول‌بندی الگوریتم خفاش هوشمند ممکن شود. با توجه به توضیحات و ویژگی‌های مکان‌یابی خفاش، آقای یانگ الگوریتم خفاش را با سه قانون زیر توسعه داده اند:

- همه خفاش‌ها از پژواک برای حس کردن فاصله استفاده می‌کنند و همچنین تفاوت بین مواد غذایی یا شکار و موانع پس‌زمینه را به روشی جادویی می‌دانند.
- خفاش‌ها به‌طور تصادفی با سرعت v_i در موقعیت x_i با فرکانس f_{min}، طول‌موج متغیر λو بلندی صدای A_0 برای جستجوی طعمه پرواز می‌کنند. آن‌ها می‌توانند به‌طور خودکار

طول‌موج (فرکانس) پالس‌های ساطع‌شده خود را تنظیم کنند و نرخ ساطع شدن پالس ($r\in [0,1]$) را بسته به نزدیکی هدفشان تنظیم کنند.

- اگرچه بلندی صدا می‌تواند از بسیاری جهات متفاوت باشد، فرض شده است که بلندی صدا از یک مقدار A_0 بزرگ (مثبت) تا یک مقدار ثابت حداقل A_{min} متغیر باشد.

$$x_i^t = x_i^{t-1} + v_i^t \quad (۱۶)$$

رابطه (۱۶)، موقعیت هر خفاش را به‌روزرسانی می‌کند. x_i^{t-1} حرکت خفاش در مرحله قبل می‌باشد، v_i^t سرعت خفاش i ام در زمان t می‌باشد.

با استفاده از رابطه (۱۷) بهترین موقعیت خفاش به دست می‌آید، x_* موقعیت بهترین خفاشی می‌باشد که تاکنون یافته شده است و f_i یک پارامتر فرکانس است که نشان‌دهنده مقداری است که خفاش باید حرکت کند.

$$v_i^t = v_i^{t-1} + (x_i^{t-1} - x_*)f_i \quad (۱۷)$$

۲-۲-۶- الگوریتم جستجوی کلاغ

الگوریتم جستجوی کلاغ اولین بار در سال (۲۰۱۶) توسط علیرضا عسکرزاده ارائه‌شده است. این الگوریتم از رفتار گروهی کلاغ‌ها در جست‌وجوی غذا الهام گرفته است که در آن کلاغ‌ها غذای اضافی خود را در مکان‌های مخفی ذخیره کرده و در صورت نیاز به غذا آن را بازیابی می‌کنند. کلاغ‌ها یکی از باهوش‌ترین پرندگان می‌باشند. آن‌ها بزرگ‌ترین مغز را نسبت به‌اندازه بدن خود دارند. بر اساس شاخص نسبت مغز به بدن، مغز آن‌ها کمی پایین‌تر از مغز انسان است. آن‌ها خودآگاهی را در آزمایش‌هایی موسوم به آزمایش آینه نشان داده‌اند و از توانایی ابزارسازی برخوردار هستند. کلاغ‌ها می‌توانند چهره‌ها را به خاطر بسپارند و وقتی شخصی غیردوستانه نزدیک می‌شود به یکدیگر هشدار می‌دهند. علاوه بر این، آن‌ها می‌توانند به روش‌های پیچیده‌ای ارتباط برقرار کرده و مخفی گاه غذای خود را تا چندین ماه بعد به یادآورند [۲۰]. در تشریح تئوری الگوریتم فرض بر این است که یک محیط d بعدی شامل تعدادی کلاغ وجود دارد. تعداد کلاغ‌ها (اندازه گروه) N است و موقعیت کلاغ i در هر تکرار در فضای جستجو توسط یک بردار $x^{i,iter}$ مشخص می‌شود که در آن $i = 1,2,3,\dots Ni$ و $iter = 1,2,3,\dots,Maxiter$ تعریف می‌شود. $x^{i,iter}$ در فضای d بعدی با استفاده از رابطه (۱۸) محاسبه می‌شود.

$$x^{i,iter} = x_1^{i,iter}, x_2^{i,iter}, \dots, x_d^{i,iter} \quad (۱۸)$$

هر کلاغ حافظه‌ای دارد که در آن موقعیت مخفیگاه آن حفظ می‌شود. در هر تکرار $(iter)$، موقعیت مخفیگاه کلاغ i توسط $m^{i,iter}$ نشان داده می‌شود. این بهترین موقعیتی است که کلاغ i تاکنون به دست آورده است. درواقع هر کلاغ موقعیت بهترین تجربه خود را حفظ می‌کند. کلاغ‌ها در محیط حرکت کرده و بدنبال منابع غذایی بهتر (مخفیگاه‌ها) می‌گردند. فرض کنید که در تکرار $iter$، کلاغ j می‌خواهد از مخفیگاه خود یعنی $m^{j,iter}$ دیدن کند. در این تکرار، کلاغ i تصمیم می‌گیرد کلاغ j را دنبال کند تا به مخفیگاه کلاغ j نزدیک شود. در این حالت، دو حالت ممکن است رخ دهد، در حالت اول کلاغ j نمی‌داند کلاغ i او را دنبال می‌کند، درنتیجه کلاغ i به مخفیگاه کلاغ j نزدیک می‌شود. موقعیت جدید کلاغ i با استفاده از رابطه (۱۹) به دست می‌آید:

$$m^{i,iter+1} = \mathrm{x}^{i,iter} + r_i \times fl^{i,iter} \times (m^{j,iter} - x^{i,iter}) \quad (۱۹)$$

در رابطه فوق r_i یک عدد تصادفی با توزیع یکنواخت بین ۰ و ۱ است و $fl^{i,iter}$ طول پرواز کلاغ i در تکرار $iter$ ام است. مقادیر کوچک fl منجر به جستجوی محلی شده و مقادیر بزرگ‌تر منجر به جستجوی سراسری می‌شود. در حالت دوم کلاغ j می‌داند که کلاغ i او را دنبال می‌کند. درنتیجه، کلاغ j برای محافظت از مخفیگاه غذای خود (حافظه پنهان خود) در برابر سرقت، با رفتن به موقعیت دیگری از فضای جستجو، کلاغ i را فریب می‌دهد. رابطه (۲۰) این حالت را بیان می‌کند.

$$x^{i,iter+1} = \begin{cases} x^{i,iter} + r_i \times fl^{i,iter} \times \left(m^{j,iter} - x^{i,iter}\right) & r_j \geq AP^{j,iter} \\ a\ Random\ position, & \end{cases} \quad (۲۰)$$

که در آن r_j یک عدد تصادفی با توزیع یکنواخت بین ۰ و ۱ است و $AP^{j,iter}$ بیانگر احتمال آگاهی کلاغ j در هر تکرار می‌باشد. الگوریتم‌های فراابتکاری باید تعادل خوبی بین تنوع[1] و تشدید[2] ایجاد کنند. در الگوریتم جستجوی کلاغ،

[1] Diversification

[2] Intensification

شدت و تنوع عمدتاً توسط پارامتر احتمال آگاهی کنترل می‌شود. با کاهش مقدار احتمال آگاهی، الگوریتم تمایل به جستجو در محلی را دارد که یک‌راه حل خوب در حال حاضر در این محل پیداشده است. درنتیجه، استفاده از مقادیر کم AP، شدت را افزایش می‌دهد. از طرف دیگر، با افزایش مقدار احتمال آگاهی، احتمال جستجو در مجاورت راه‌حل‌های خوب فعلی کاهش‌یافته و الگوریتم جستجوی کلاغ تمایل دارد فضای جستجو را در مقیاس سراسری جستجو کند. درنتیجه، استفاده از مقادیر زیاد پارامتر احتمال آگاهی تنوع را افزایش می‌دهد.

۲-۲-۷- الگوریتم کرم شب‌تاب

الگوریتم کرم شب‌تاب در اواخر سال (۲۰۰۷) توسط ژین شی یانگ معرفی‌شده است [۱۴]. این الگوریتم یکی از نمونه‌های هوش جمعی است که در آن از همکاری اعضای ساده و کم‌هوش، مرتبه بالاتری از هوشمندی ایجاد می‌شود که قطعاً توسط هیچ‌یک از اجزا قابل حصول نیست. کرم‌های شب‌تاب نوری از خود تولید می‌کنند که کاملاً شبیه سایر نورها است. به‌جز اینکه این نور حرارتی ندارد. در کرم شب‌تاب این نور توسط ماده‌ای به نام لوسی فرین تولید می‌شود که این ماده با اکسیژن ترکیب‌شده و تولید نور می‌نماید. الگوریتم کرم شب‌تاب از ارتباط نوری بین کرم‌های شب‌تاب الهام گرفته است. برای ساده‌سازی فرمول‌بندی این الگوریتم فرض شده است که همه کرم‌های شب‌تاب، تک‌جنسی[1]هستند. به‌عبارت‌دیگر، کرم‌های شب‌تاب، بدون توجه به جنسیت آن‌ها، می‌توانند به دیگر کرم‌های شب‌تاب موجود در فضای مسئله جذب شوند.

جذابیت[2] یک کرم شب‌تاب متناسب با روشنایی آن کرم است. بنابراین، به ازای وجود دو کرم شب‌تاب چشمک‌زن، کرمی که نور کمتری دارد به سمت کرمی که نور بیشتری دارد جذب خواهد شد. وقتی‌که فاصله بین دو کرم افزایش پیدا می‌کند، مقدار جذابیت و روشنایی آن‌ها کاهش پیدا می‌کند. به‌عبارت‌دیگر، وقتی‌که فاصله دو کرم شب‌تاب از یکدیگر بیشتر می‌شود، علاوه بر اینکه جذابیت آن‌ها برای یکدیگر کاهش پیدا می‌کند، روشنایی آن‌ها (برای یکدیگر) نیز کاهش پیدا می‌کند. درصورتی‌که روشنایی یک کرم شب‌تاب خاص از دیگر کرم‌ها بیشتر باشد، به‌طور تصادفی در محیط حرکت خواهد کرد (به سمت هیچ‌یک از کرم‌های دیگر جذب نخواهد شد). الگوریتم کرم شب‌تاب با مدل‌سازی رفتار مجموعه‌ای از کرم‌های شب‌تاب و تخصیص مقداری مرتبط با برازندگی مکان هر کرم شب‌تاب به‌عنوان مدلی برای میزان روشنایی کرم شب‌تاب و نیز به‌روز کردن مکان کرم‌ها در تکرارهای متوالی الگوریتم به جستجوی جواب بهینه مسئله می‌پردازد. درواقع الگوریتم در هر تکرار دارای دو مرحله اصلی است: مرحله به‌روزرسانی میزان روشنایی و مرحله حرکت. کرم‌های شب‌تاب به سمت کرم‌های شب‌تاب دیگر با روشنایی بیشتر که در همسایگی آن‌ها باشند حرکت می‌کنند. به‌این‌ترتیب طی تکرارهای متوالی، مجموعه به سمت جواب بهتر متمایل می‌گردد. الگوریتم با قرار دادن جمعیت n عضوی از کرم‌های شب‌تاب در نقاط مختلف فضای جستجو آغاز می‌شود. در مرحله اول، همه‌ی کرم‌ها مقدار یکسانی از لوسی فرین (ماده تولیدکننده نور) دارند و در تکرارهای بعدی، با استفاده از رابطه (۲۱) میزان روشنایی کرم شب‌تاب به‌روزرسانی می‌شود.

$$\beta ij=\beta 0 e^{-\gamma r^2 ij} \qquad (۲۱)$$

βij ضریب تأثیر فاصله بین کرم شب‌تاب i و j می‌باشد، $\beta 0$ e ضریب اولیه‌یا پایه می‌باشد، γ ضریب جذب نور می‌باشد. $\gamma r^2 ij$ فاصله اقلیدسی بین کرم‌های شب‌تاب i و j است.

برای شبیه‌سازی حرکت کرم شب‌تابi به سمت کرم شب‌تاب jاز رابطه زیر استفاده می‌شود.

$$xi(t+1) = xi(t) + \beta ij(t) \cdot (xj(t) - xi(t)) + \alpha \cdot \epsilon(t) \qquad (۲۲)$$

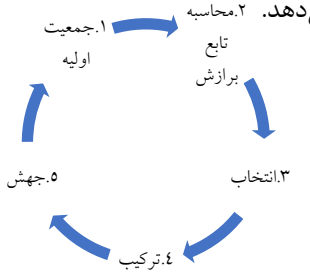
به‌طوری‌که $xi(t+1)$ نشان‌دهنده مکان کرم شب‌تاب در مرحله بعد می‌باشد، $xi(t)$ مکان کرم شب‌تاب در مرحله جاری را نشان می‌دهد، α پارامتر مقیاس است که میزان گام را نشان می‌دهد، $\epsilon(t)$ یک بردار تصادفی است که از یک توزیع گوسی با میانگین ۰ و واریانس ۱ نمونه‌برداری می‌شود. سپس حرکت کرم شب‌تاب i به سمت کرم شب‌تاب jاست که توسط جذابیت βij با یک عدد تصادفی برای جستجو مقداردهی می‌شود.

۲-۲-۸- الگوریتم ژنتیک

الگوریتم ژنتیک یک الگوریتم تکاملی است که توسط جوهان‌هالند در سال (۱۹۹۲) پیشنهادشده است. این الگوریتم یک روش بهینه‌سازی مبتنی بر جستجو و بر اساس

[1] Unisex
[2] Attractiveness

اصول ژنتیک و انتخاب طبیعی است. ساختار اصلی الگوریتم ژنتیک با انتخاب یک جمعیت اولیه شروع می‌شود (که ممکن است به‌طور تصادفی یا توسط سایر روش‌های اکتشافی ایجاد شود). سپس والدینی برای جفت‌گیری انتخاب می‌شوند تا برای نسل بعدی فرزندانی ایجاد کنند. انتخاب والدین برای نرخ همگرایی الگوریتم ژنتیک بسیار مهم است زیرا انتخاب والدین خوب الگوریتم را به سمت ایجاد راه‌حل‌های بهتر و مناسب‌تر سوق می‌دهد [۲۱]. پس از انتخاب والدین، عملگرهای ترکیب[۱] و جهش[۲] روی والدین اعمال می‌شوند تا فرزندان جدید تولید شوند. عمل ترکیب می‌تواند به‌صورت تک‌نقطه‌ای، چندنقطه‌ای و یا ترکیب‌های دیگری صورت گیرد. عمل جهش به‌عنوان یک تغییر تصادفی کوچک در کروموزوم برای به دست آوردن یک راه حل جدید تعریف می‌شود. جهش برای حفظ و معرفی تنوع در جمعیت ژنتیکی اهمیت دارد. درنهایت فرزندان جدید جایگزین اعضای موجود در جمعیت قبلی می‌شوند و این روند تکرار می‌شود. شکل (۱) ساختار اصلی الگوریتم ژنتیک را نشان می‌دهد.



شکل ۱- ساختار اصلی الگوریتم ژنتیک

۲-۲-۹- الگوریتم ازدحام ذرات

الگوریتم ازدحام ذرات در سال (۱۹۹۵) توسط ابرهارت و کندی معرفی شده است. این الگوریتم الهام گرفته از حرکت دسته‌جمعی پرندگان می‌باشد. گروهی از پرندگان در فضا به‌صورت تصادفی دنبال غذا می‌گردند. در این الگوریتم، یک گروه از ذرات در فضای جستجو حرکت می‌کنند و با توجه به موقعیت و سرعت خود، به دنبال بهبود موقعیت خود در فضای جستجو هستند. هر ذره در هرلحظه، با توجه به موقعیت خود و موقعیت بهترین ذره در گروه، سرعت خود را تغییر داده و به سمت بهترین موقعیت حرکت می‌کند.

این الگوریتم با تکرار این فرآیند، به بهبود موقعیت و پیدا کردن راه‌حل بهینه برای مسئله کمک می‌کند [۲۲]. هر ذره دارای یک موقعیت و سرعت در فضای جستجو است که با هدف پیدا کردن نقطه بهینه تغییر می‌کند. با توجه به وضعیت هر ذره و موقعیت فعلی آن، مقدار مناسبی به‌عنوان تابع هدف[۳] برای آن محاسبه می‌شود. سپس با توجه به مقدار تابع هدف، ذرات حرکت کرده و سعی در بهبود موقعیت خود دارند. همچنین، ذرات با هم اطلاعات را به اشتراک می‌گذارند تا به‌صورت گروهی بهینه‌سازی انجام شود. الگوریتم ازدحام ذرات با گروهی از ذرات تصادفی (راه‌حل‌ها) شروع می‌شود و سپس با به‌روزرسانی نسل‌ها جستجو ادامه پیدا می‌کند. ذرات بهترین موقعیت شخصی خود[۴] و بهترین موقعیت جهانی[۵] را که توسط هر ذره در ازدحام یافت می‌شود حفظ می‌کنند. با در نظر گرفتن بهترین موقعیت شخصی و جهانی، ذرات سرعت خود را تنظیم می‌کنند و در فضای جستجو حرکت می‌کنند تا به سمت راه‌حل‌های بهتر همگرا شوند. روابط (۲۳ و ۲۴) در این الگوریتم برای به‌روزرسانی موقعیت و سرعت ذرات استفاده می‌شوند.

$$V_{i,t+1} = W.V_{i,t-1} + C_1.r_1.(Pbest_i - P_{i,t}) + C_2.r_2.(Gbest_i - P_{i,t}) \quad (۲۳)$$

$$P_{t+1} = P_t + V_t \quad (۲۴)$$

به‌طوری‌که $V_{i,t+1}$ سرعت ذره i ام در تکرار $t+1$ می‌باشد. $P_{i,t}$ موقعیت ذره i ام در تکرار t و $Pbesti$ و $Gbesti$ به ترتیب نشان‌دهنده‌ی بهترین موقعیت ذره i ام در طول حرکت خود و بهترین موقعیت گروه یا بهترین جمعیت فعلی هستند. r_1 و r_2 اعداد تصادفی با توزیع یکنواخت بین ۰ و ۱ هستند. W، $C1$ و $C2$ پارامترهای یادگیری می‌باشند و با توجه به نوع و صورت‌مسئله به‌منظور بهبود کارایی و کنترل رفتار الگوریتم تنظیم می‌شوند. P_{t+1} موقعیت ذره در تکرار $t+1$ را مشخص می‌کند و P_t موقعیت ذره در تکرار t ام را نشان می‌دهد. V_t و $V_{i,t-1}$ به ترتیب سرعت ذره در تکرار t ام و سرعت ذره i ام در تکرار $t-1$ هستند.

[1] Crossover
[2] Mutation
[3] Fitness function
[4] Pbest
[5] Gbest

۲-۲-۱۰-الگوریتم نهنگ

الگوریتم نهنگ ازجمله الگوریتم‌های هوش جمعی است که در سال (۲۰۱۶) توسط سید علی میر جلیلی پیشنهادشده است. این الگوریتم از روش شکار تور حبابی که توسط نهنگ‌های گوژپشت انجام می‌شود، الهام گرفته است. رفتار شکار نهنگ‌های گوژپشت تغذیه حبابی نامیده می‌شود که در آن نهنگ‌های گوژپشت ترجیح می‌دهند گله‌ای از ماهی‌های نزدیک به سطح آب را شکار کنند. جستجوی غذا با ایجاد حباب‌های متمایز در امتداد یک مسیر دایره‌ای شکل انجام می‌شود. الگوی مرتبط با تغذیه حبابی شامل مراحل «مارپیچ رو به بالا» و «حلقه‌های دوتایی» می‌باشند. در مرحله «مارپیچ رو به بالا» نهنگ‌های گوژپشت حدود ۱۲ متر به پایین شیرجه زده و سپس شروع به ایجاد حباب‌هایی به شکل مارپیچی در اطراف طعمه می‌کنند و به سمت سطح شنا می‌کنند. مرحله «حلقه‌های دوتایی» از سه بخش مختلف تشکیل می‌شود که عبارت‌اند از: شکار محاصره‌ای، حمله به حباب تور (فاز بهره‌برداری) و جستجوی شکار (فاز اکتشاف)، در مرحله‌ی شکار محاصره‌ای نهنگ‌ها می‌توانند مکان شکار را شناسایی کرده و آن‌ها را محاصره کنند. الگوریتم فرض می‌کند که شکار هدف بهترین راه‌حل کاندید حال حاضر بوده و یا نزدیک به حالت مطلوب است. بعدازاینکه بهترین عامل جستجو شناسایی شد، عوامل دیگر جستجو سعی می‌کنند تا مکان خود را نسبت به بهترین عامل جستجو، به‌روزرسانی کنند. این رفتار از طریق روابط (۲۵) و (۲۶) بیان‌شده است:

$$\vec{D} = \left|\vec{C}.\ \vec{X^*}(t) - \vec{X}(t)\right| \qquad (۲۵)$$

$$\vec{X}(t+1) = \vec{X}(t) - \vec{A}.\vec{D} \qquad (۲۶)$$

که در آن t تکرار جاری را نشان می‌دهد، $\vec{A}$ و $\vec{C}$ بردارهای ضرایب، $\vec{X^*}$ بردار مکان بهترین راه‌حل به‌دست‌آمده در حال حاضر و X بردار مکان است. لازم به ذکر است که در صورت وجود راه‌حل بهتر، $\vec{X^*}$ در هر تکرار باید به‌روزرسانی شود. بردارهای $\vec{A}$ و C با استفاده از رابطه‌های (۲۷) و (۲۸) محاسبه می‌گردند:

$$\vec{A} = 2\vec{a}\ .\vec{r} - \vec{a} \qquad (۲۷)$$

$$\vec{C} = 2\ .\vec{r} \qquad (۲۸)$$

که a به‌صورت خطی از مقدار ۲ تا ۰ و در طی تکرارها کاهش می‌یابد. r یک بردار تصادفی در فاصله ۰ تا ۱ است. جهت مدل‌سازی ریاضی رفتار حمله به حباب تور نهنگ‌ها، دو روش پیشنهادشده است: مکانیزم محاصره‌ی انقباضی: این رفتار از طریق افزایش مقدار a در رابطه (۲۷) حاصل می‌شود. محدوده نوسان A به‌وسیله a کاهش می‌یابد. به‌عبارت‌دیگر، A مقداری تصادفی در فاصله a تا a− است و a در طول تکرارها، از مقدار ۲ تا ۰ کاهش می‌یابد. با انتخاب مقادیر تصادفی برای A در فاصله‌ی ۱ تا ۱- می‌توان مکان جدید عامل جستجو را در هرکجا بین مکان اصلی عامل و مکان بهترین عامل کنونی، تعیین کرد.

مکان در حال به‌روزرسانی مارپیچی: این روش در ابتدا فاصله بین وال قرارگرفته در مختصات X^* و Y^* طعمه موجود در X^* و Y^* را محاسبه می‌کند. معادله‌ای مارپیچی بین موقعیت نهنگ و طعمه ایجاد می‌شود تا حرکت حلزونی شکل نهنگ گوژپشت را تقلید کند که در رابطه (۲۹) نشا ن داده‌شده است.

$$\vec{X}(t+1) = \vec{D'}.\ e^{bl}.cos(2\pi l) + \vec{X^*}(t) \qquad (۲۹)$$

به‌طوری‌که D به فاصله ۱ امین نهنگ تا طعمه اشاره دارد (بهترین راه‌حل به‌دست‌آمده تاکنون)، b عددی تصادفی بین ۱ تا ۱- می‌باشد و ثابتی برای تعریف شکل مارپیچ لگاریتمی است. باید به این نکته توجه داشت که نهنگ، حول طعمه در امتداد یک دایره‌ی انقباضی و هم‌زمان در مسیر مارپیچی شکلی به شنا درمی‌آید. جهت مدل‌سازی این رفتار هم‌زمان، فرض می‌شود که نهنگ با احتمال ۵۰ درصد از بین مکانیزم محاصره‌ی انقباضی و یا مدل مارپیچی یکی را انتخاب می‌کند تا موقعیت نهنگ‌ها در طول بهینه‌سازی به‌روزرسانی شوند. رابطه (۳۰) مدل ریاضی این مسئله را نشان می‌دهد.

$$\vec{X}(t+1) = \begin{cases} \vec{X^*}(t) - \vec{A}.\vec{D} & if\ p < 0.5 \\ \vec{D'}.e^{bl}.cos(2\pi l) + \vec{X^*}(t) & if\ p \geq 0.5 \end{cases} \qquad (۳۰)$$

در این رابطه P عددی تصادفی بین ۰ تا ۱ است. در مرحله‌ی جستجوی شکار، از روشی مشابه بر مبنای واریاسیون بردار A جهت جستجوی شکار به‌کاربرده می‌شود. در حقیقت نهنگ‌ها بر طبق مکان یکدیگر، به‌صورت تصادفی به جستجو می‌پردازند. بنابراین، بردار A را با مقادیر تصادفی بزرگ‌تر از ۱ و یا کمتر از ۱- به کار گرفته می‌شود تا عامل جستجو را مجبور به دور شدن از

نهنگ مرجع نماید. برخلاف فاز استخراج، جهت به‌روزرسانی موقعیت عامل جستجو در فاز شکار (اکتشاف) به‌جای استفاده از داده‌های بهترین عامل جستجو، از انتخاب تصادفی عامل بهره برده می‌شود. این مکانیزم به همراه شرط $A>1$ بر اکتشاف تأکیددارند و به الگوریتم نهنگ اجازه می‌دهد تا جستجوی سراسری را انجام دهد. برای مدل‌سازی این فاز از رابطه (۳۱) استفاده می‌شود

$$\vec{D} = |\vec{C}.\ \overrightarrow{X_{rand}} - \vec{X}|$$
$$\vec{X}(t+1) = \overrightarrow{X_{rand}} - \vec{A}.\vec{D} \qquad (۳۱)$$

در رابطه فوق، X_{rand} بردار موقعیت تصادفی انتخاب‌شده (نهنگ تصادفی) از جمعیت جاری است. الگوریتم نهنگ با مجموعه‌ای از راه‌حل‌های تصادفی آغاز به کار می‌کند. در هر تکرار، عوامل جستجو موقعیت خود را با در نظر گرفتن عامل جستجویی که تصادفی انتخاب‌شده و بهترین راه‌حل به‌دست‌آمده‌ی جاری، به‌روزرسانی می‌کنند. پارامتر a جهت فراهم آوردن اکتشاف و استخراج، به ترتیب از مقدار ۲ تا ۰ کاهش می‌یابد. یک عامل جستجوی تصادفی در حالت $|A|>1$ انتخاب می‌شود، این در حالی است که بهترین راه‌حل زمانی انتخاب می‌شود که جهت به‌روزرسانی موقعیت عوامل جستجو، $|A|<1$ باشد. بسته به مقدار P، الگوریتم نهنگ قادر است بین حرکت دایره‌ای و یا مارپیچی یکی را انتخاب کند. الگوریتم نهنگ درنهایت با ارضای شرایط خاتمه، پایان می‌پذیرد.

۳. پیشینه تحقیق

در این بخش تحقیقات پیشین مرتبط با موضوع مورد تحقیق مورد بررسی قرار می گیرد.

محمدعلی واقدی و همکاران [۲۳] در سال (۲۰۲۳)، عملکرد الگوریتم‌های فراابتکاری توسعه‌یافته را بر روی توابع معیار *CEC 2021* توسعه‌یافته مورد ارزیابی قرار داده‌اند. توابع هدف با گنجاندن عملگرهایی مانند بایاس[۱]، شیفت[۲] و چرخش[۳] تغییر داده‌شده‌اند. ترکیبات مختلفی از عملگرهای باینری بر روی توابع هدف اعمال‌شده است تا توابع هدف منجر به توابع معیار *CEC 2021* شوند. در این پژوهش، الگوریتم‌های فراابتکاری به دو دسته‌ی توسعه‌یافته و پایه تقسیم شده‌اند. با استفاده از هر یک از این الگوریتم‌ها توابع هدف با ابعاد[۴] مختلفی حل شده‌اند. درنهایت، عملکرد الگوریتم‌های انتخابی به‌طور تجربی بررسی‌شده است. نتایج به‌دست‌آمده حاکی از آن است که در میان دسته الگوریتم‌های پایه، الگوریتم بهینه‌سازی مبتنی بر یادگیری آموزش[۵] و از دسته الگوریتم‌های توسعه‌یافته الگوریتم تفاضل تکاملی توسعه‌یافته‌ی چند عملگره[۶] عملکرد بهتری از سایر الگوریتم‌ها داشته‌اند. اما، در این پژوهش الگوریتم گله اسب مورد ارزیابی قرار نگرفته است.

سادهو و همکاران [۲۴] در سال (۲۰۲۳)، پژوهشی با عنوان «مقایسه الگوریتم‌های فراابتکاری بر اساس عملکرد آن‌ها در مسائل پیچیده» ارائه کرده‌اند. هدف از این مطالعه، آزمایش قابلیت کاربردها و مقایسه عملکرد الگوریتم‌های تبرید شبیه‌سازی‌شده[۷]، ژنتیک، بهینه‌سازی ازدحام ذرات، تکاملی تفاضلی و نسل جدیدی از الگوریتم کرم شب‌تاب، گله کریل[۸]، گرگ خاکستری و جستجوی ارگانیسم‌های همزیستی[۹] می‌باشد. در این پژوهش برای مقایسه‌ی عملکرد الگوریتم‌های انتخابی از توابع هدف بسیار پیچیده استفاده‌شده است. نتیجه شبیه‌سازی‌ها نشان داده است که در بین تمام الگوریتم‌های انتخاب‌شده، الگوریتم جستجوی ارگانیسم‌های همزیستی و الگوریتم گله کریل با موفقیت بیشتری توابع هدف بسیار پیچیده را حل کرده و بهترین راه‌حل را برای اکثر آن‌ها به دست آورده‌اند. از طرف دیگر الگوریتم‌های تکاملی تفاضلی و ازدحام ذرات به‌طور مؤثر به راه‌حل بهینه‌ای دست‌یافته‌اند که بسیار نزدیک به بهترین جواب بوده است. بااین‌حال، در این پژوهش عملکرد الگوریتم گله اسب مورد ارزیابی قرار نگرفته است.

حجت‌الله رجبی مشتاقی و همکاران[۲۵] در سال (۲۰۲۲)، مقاله‌ای با عنوان «مقایسه و رتبه‌بندی الگوریتم‌های فراابتکاری با استفاده از روش‌های تصمیم‌گیری گروهی» ارائه کرده‌اند. هدف از انجام این تحقیق رتبه‌بندی الگوریتم‌های فراابتکاری با استفاده از روش‌های تصمیم‌گیری گروهی بوده است. در این راستا، پنج الگوریتم شامل: الگوریتم های ژنتیک، ازدحام ذرات، جهش

---

[1] Bias
[2] Shift
[3] Rotation
[4] Dimention
[5] Teaching-learning-based optimization (TLBO)
[6] Improved multi-operator differential evolution algorithm
[7].Simulated Annealing (SA)
[8].Krill herd (KH)
[9] Symbiotic Organisms Search (SOS)

قورباغه[1]، کلونی زنبورعسل مصنوعی[2] و الگوریتم رقابت استعماری[3] انتخاب و عملکرد آنها با بهره‌گیری از ۱۵ تابع تست استاندارد مقایسه شده‌اند. برای مقایسه عملکرد الگوریتم ها از دو معیار «میانگین تابع هدف» و «میانگین زمان محاسباتی» استفاده‌شده است. نتایج شبیه‌سازی الگوریتم‌ها نشان داده است که الگوریتم‌های ازدحام ذرات، رقابت استعماری، ژنتیک، کلونی زنبورعسل مصنوعی و جهش قورباغه به ترتیب دارای بهترین میانگین تابع هدف و میانگین زمان محاسباتی هستند. اما، در این پژوهش اغلب الگوریتم‌های قدیمی ارزیابی‌شده‌اند و عملکرد الگوریتم گله اسب نیز موردبررسی قرار نگرفته است.

در سال (۲۰۲۱)، پژوهشی با عنوان «اندازه‌گیری عملکرد الگوریتم‌های بهینه‌سازی فراابتکاری» توسط فرانسیوس اسکات و همکاران [۲۶] ارائه‌شده است. در این تحقیق استراتژی‌های جدیدی برای ارزیابی دقیق عملکرد الگوریتم‌های فراابتکاری ارائه‌شده است که ازجمله آن‌ها یک معیار جدید برای امتیازدهی به الگوریتم‌ها می‌باشد که عملکرد الگوریتم‌ها را با در نظر گرفتن مقدار نهایی بهینه‌سازی و سرعت همگرایی ارزیابی می‌کند. این اندازه‌گیری بر اساس مجموعه‌ای از نتایج آماری از مسائل مختلف بهینه‌سازی انجام‌شده و مشکلاتی ازجمله بُعد، توابع هدف و محدودیت‌های ارزیابی در نظر گرفته‌شده است. در این پژوهش، از روش‌های تجمیع برای ارتباط دادن وزن مسئله با امتیاز محاسبه‌شده استفاده‌شده است. با استفاده از این استراتژی عملکرد الگوریتم‌های ازدحام ذرات، منطق فازی، ژنتیک و تبرید شبیه‌سازی‌شده بررسی‌شده است. نتایج شبیه‌سازی‌ها نشان داده است که الگوریتم ازدحام ذرات بهترین عملکرد را در میان الگوریتم‌های دیگر داشته است. بااین‌حال، در این پژوهش الگوریتم‌های جدید مورد ارزیابی قرار نگرفته‌اند و همچنین عملکرد الگوریتم گله اسب بررسی نشده است.

سید جلال‌الدین موسوی راد و همکاران [۲۷] در سال (۲۰۲۰)، پژوهشی با عنوان «بررسی برخی از الگوریتم‌های فراابتکاری مبتنی بر جمعیت برای آموزش شبکه عصبی چندلایه» ارائه کرده‌اند. در این پژوهش، رویکردهای مبتنی بر کاهش گرادیان برای جلوگیری از افتادن در بهینه‌های محلی در کاربردهای صنعتی مورد بررسی قرارگرفته است. برای این منظور از ۱۵ الگوریتم فراابتکاری مبتنی بر جمعیت که از جمله جدیدترین و پرکاربردترین الگوریتم‌ها می‌باشند برای آموزش شبکه‌های عصبی استفاده‌شده است. این الگوریتم‌ها شامل الگوریتم‌های ازدحام ذرات، تکاملی تفاضلی، کلونی زنبورهای مصنوعی، رقابت استعماری[4]، جستجوی فاخته[5]، جستجوی گرانشی[6]، خفاش[7] و... می‌باشند. نتایج شبیه سازی مشخص کرده است که الگوریتم ازدحام ذرات عملکرد بهتری نسبت به سایر الگوریتم‌ها داشته است. اما در این پژوهش، عملکرد الگوریتم گله اسب مورد بررسی قرار نگرفته است.

ایزوگو و همکاران [۲۸] در سال (۲۰۱۹)، پژوهشی با عنوان «مطالعه مقایسه‌ای الگوریتم‌های بهینه‌سازی فراابتکاری برای مسئله کوله‌پشتی» ارائه کرده‌اند. هدف از این پژوهش، مقایسه الگوریتم‌های فراابتکاری استفاده‌شده برای حل مسئله کوله‌پشتی است. الگوریتم‌هایی همچون، ژنتیک، تبرید شبیه‌سازی‌شده، کلونی مورچگان، کرم شب‌تاب، جستجوی حریصانه و یک الگوریتم ترکیبی شامل الگوریتم ژنتیک و الگوریتم تبرید شبیه‌سازی‌شده که در حل مسئله کوله‌پشتی استفاده‌شده‌اند، موردبررسی قرارگرفته‌اند. در این پژوهش، مشکلات مرتبط با مسئله کوله‌پشتی در هر الگوریتم به‌گونه‌ای طراحی‌شده است که راه‌حل‌های غیرقابل اجرا را جریمه و راه‌حل‌های امکان‌پذیر را بهینه می‌کند. آزمایش‌ها بر روی مسائل کوله‌پشتی با ابعاد مختلف انجام‌شده است و نتایج عددی نشان داده است که الگوریتم ترکیبی بهترین پاسخ را ارائه داده است. با این حال در این پژوهش الگوریتم‌های جدیدتر مورد ارزیابی قرار نگرفته و همچنین، عملکرد الگوریتم گله اسب مورد بررسی قرار نگرفته است.

۴. روش پژوهش

در این پژوهش، برای مقایسه و ارزیابی عملکرد الگوریتم‌های انتخابی از ۱۰ تابع تست استاندارد که مجموعه‌ای از توابع تک‌وجهی[8] و چندوجهی[9] را شامل می‌شوند، استفاده‌شده است [۸]. این توابع به همراه

[1] Shuffled Frog Leaping Algorithm (SFLA)
[2] Artificial bee colony algorithm (ABC)
[3] Imperialist Competitive Algorithm (ICA)
[4] Imperialist Competitive Algorithm (ICA)
[5] Cuckoo search
[6] Gravitational Search Algorithm (GSA)
[7] Bat algorithm
[8] Uni-modal benchmark functions
[9] Multi-modal benchmark functions

مشخصات آن‌ها از قبیل نام تابع، دامنه، تک‌وجهی یا چندوجهی بودن تابع و فرمول هر یک از آنان در جدول (۱) نشان داده‌شده‌اند. توابع تک‌وجهی، ابزاری مناسب برای نشان دادن قدرت الگوریتم در طول مرحله اکتشاف هستند. توابع $F1$ تا $F4$ و $F8$، $F9$ جزء توابع تک‌وجهی هستند. توابع چندوجهی، یک معیار مؤثر برای به چالش کشیدن توانایی بهره‌برداری یک الگوریتم هستند. در این تحقیق، از چهار تابع چندوجهی مختلف شامل $F5$ تا $F7$ و $F10$ استفاده می‌شود. این توابع جزء توابع استانداردی هستند که معمولاً در ارزیابی الگوریتم‌های فراابتکاری مورداستفاده قرار می‌گیرند.

جدول ۱- مجموعه توابع تست [۸]

تابع	دامنه	ویژگی	کمینه تابع	فرمول تابع
F1 (Sphere function)	[-100,100]	تک‌وجهی	$f_{min} = 0$	$f(x) = \sum_{i=1}^{dim} x_i^2$
F2 (Rotated Hyper-Ellipsoid function)	[-100,100]	تک‌وجهی	$f_{min} = 0$	$f(x) = \sum_{i=1}^{dim} (\Sigma_j^i = 1x_J)^2$
F3 (Schwefel.1 function)	[-100,100]	تک‌وجهی	$f_{min} = 0$	$f(x) = \max \{\|x\|, 1 \le i \le Dim\}$
F4 (Schwefe.2 function)	[-2.5,2.5]	تک‌وجهی	$f_{min} = 0$	$f(x) = \sum_{i=1}^{dim} \|x_i\| + \Pi_{1=1}^{\dim \|x_i\|}$
F5 (Rastrigin function)	[-100,100]	چندوجهی	$f_{min} = 0$	$f(x) = \Sigma_{i=1}^{\dim (x_i^2 - 10\cos(2\Pi x_i) + 10)}$
F6 (Ackley function)	[-100,100]	چندوجهی	$f_{min} = 0$	$f(x) = -20\exp\left(-0.2\sqrt{\frac{\sum_{i=1}^{dim} x_i^2}{2}}\right) - \exp\left(\frac{\Sigma^{dim}\cos(2\Pi x_i)}{2}\right) + 2 + e$
F7 (Drop-Waves function)	[-100,100]	چندوجهی	$f_{min} = -1$	$f(x) = -\frac{1 + \cosh\sqrt[12]{\sum_{i=1}^{dim} x_i^2}}{2 + 0.5\sum_{i=1}^{dim}(x_i^2)}$
F8 Uni-modal	[-30,30]	تک‌وجهی	$f_{min} = 0$	$f(x) = \sum_{i=1}^{dim} 100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2$
F9 Uni-modal	[-1.28,1.28]	تک‌وجهی	$f_{min} = 0$	$f(x) = \sum_{i=1}^{dim} (ix_i)^4 + rand[0,1)$
F10 Multi-modal	[-600,600]	چندوجهی	$f_{min} = 0$	$f(x) = \sum_{i=1}^{dim} \frac{1}{4000} - \prod_{i=1}^{dim} \cos(\frac{x_i}{\sqrt{i}}) + 1$

برای ارزیابی عملکرد مقیاس‌پذیری الگوریتم‌های انتخابی باید توابع تست را در ابعاد مختلف به کار برد. بنابراین، الگوریتم‌های انتخاب‌شده از تابع $F1$ تا $F7$ با بعدهای ۵۰۰، ۱۰۰۰ و ۲۰۰۰ و همچنین از تابع $F8$ تا $F10$ با بُعد ۱۰۰۰ پیاده‌سازی شده‌اند.

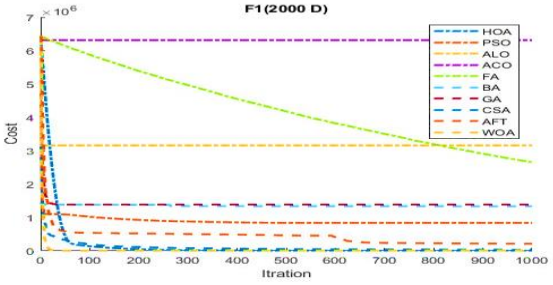
برای پیاده‌سازی هر الگوریتم لازم است پارامترهای اولیه‌ی آن الگوریتم مقداردهی شوند. با توجه به تعداد پارامترها و توابع تست، مقادیر در نظر گرفته‌شده برای پارامترهای مهم هر الگوریتم در جدول (۲) نشان داده‌شده است. شرط خاتمه هر یک از الگوریتم‌ها، تعداد ۱۰۰۰ بار اجرا در نظر گرفته‌شده است. درنهایت نتایج به‌دست‌آمده بر اساس سه شاخص بهترین پاسخ به‌دست‌آمده، انحراف معیار و زمان اجرا در ۱۰۰۰ بار اجرای هر الگوریتم می‌باشد که در جداول (۳)، (۴)، (۵) و (۶) نشان داده‌شده است. در این پژوهش، شبیه‌سازی با استفاده از نرم‌افزار متلب نسخه ۲۰۲۲b انجام‌شده و سیستم کامپیوتری استفاده‌شده لپ‌تاپ HP نسل ۲ و ۵ هسته‌ای با رم ۸ گیگابایت بوده است.

جدول ۲- برخی پارامترهای تنظیم شده برای الگوریتم‌ها

نام الگوریتم	پارامتر ها	مقادیر پارامتر ها
HOA	Hierarchy factor for β, γ Sociability factor for β, γ Imitation factor for γ, δ Defense factor for β, γ, α Roam factor for γ, δ ωg=ωs=ωh=ωr=ωd=ωi	**0.9 , 0.5** **0.2, 0.1** **0.3, 0.3** **0.2, 0.1, 0.5** **0.05, 0.1** **0.999**
ACO	Selection Pressure,Deviation-Distance Ratio	**0.5 , 1**
AFT	Marjaneh	**Rand[0,1]**
ALO	antlion by rolette wheel, best antlion so far	**Rand[0,1] ,** **Rand[0,1]**
BA	Loudness, Pulse rate	**0.9 , 0.6**
CSA	Awareness probability, Flight length (fl)	**0.1 , 2**
FA	Light Absorption Coefficient,Attraction Coefficient Base Valu, MutationCoefficient, Mutation Coefficient Damping Ratio	**1, 2, 0.2, 0.99**
GA	Crossover Rate, Mutation Rate	**0.7 , 0.3**
PSO	Inertia Weight,Personal Learning Coefficient,Global Learning Coefficient	**0.9 , 1, 1**
WOA	Convergence_curve	**zeros(1,Max_iter)**

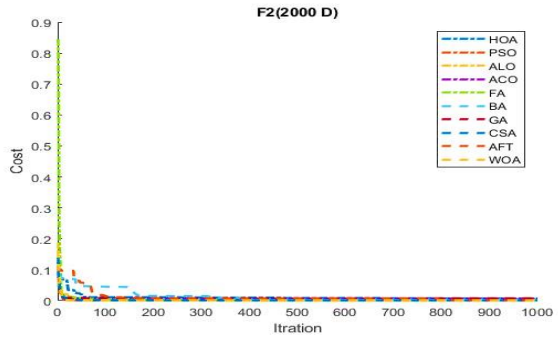
۲,۴. نتایج پژوهش

در این پژوهش با توجه به پارامترهای اولیه در نظر گرفته‌شده برای هر الگوریتم در جدول (۲) و با استفاده از مجموعه توابع تست، الگوریتم‌های موردنظر شبیه‌سازی‌شده‌اند. در شکل‌های (۲) تا (۷) منحنی همگرایی الگوریتم‌ها برای توابع *F1* تا *F7* در بُعد ۲۰۰۰و تکرار ۱۰۰۰ نشان داده‌شده است. همچنین، نتایج به‌دست‌آمده برای معیارهای بهتر بد، پاسخ، انحراف معیار و



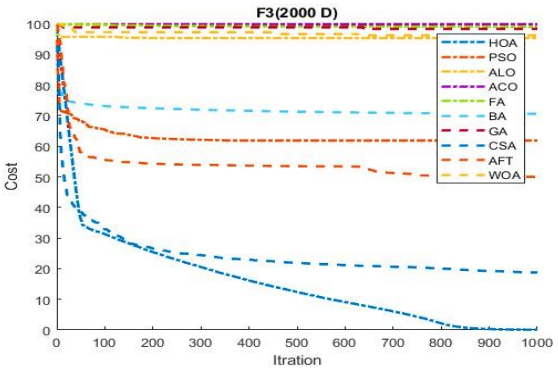
شکل ۲ - منحنی همگرایی تابع *F1* در بُعد ۲۰۰۰

شکل(۲) منحنی همگرایی الگوریتم‌ها برای حل تابع*F1* در بُعد ۲۰۰۰ و با تعداد تکرار ۱۰۰۰ را نشان می‌دهد. این شکل نشان می‌دهد که یافتن پاسخ بهینه در تکرارهای اولیه مزیت اصلی الگوریتم نهنگ است. اما در نهایت الگوریتم هایی مانند کلونی مورچگان و شیرمورچه به پاسخ بهینه همگرا نشده اند. الگوریتم کرم شب تاب در ابتدا سرعت همگرایی بالایی دارد ولی در نهایت به پاسخ بهینه همگرا نشده است.



شکل ۳ - منحنی همگرایی تابع *F2* در بُعد ۲۰۰۰

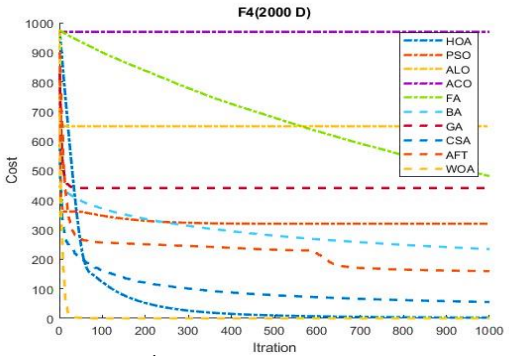
منحنی همگرایی الگوریتم‌ها برای حل تابع F2 در بُعد ۲۰۰۰ و با تعداد تکرار ۱۰۰۰ در شکل (۳) نشان داده‌شده است. این شکل نشان می‌دهد که در تابع F2، اکثر الگوریتم‌ها همگرایی خوبی دارند و نزدیک به پاسخ بهینه می‌باشند.



شکل ۴ - منحنی همگرایی تابع *F3* در بُعد ۲۰۰۰

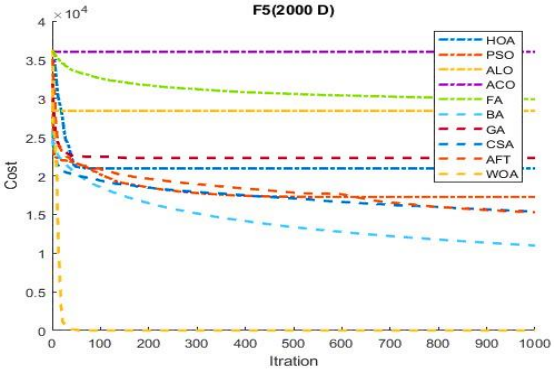
در شکل(۴) منحنی همگرایی الگوریتم‌ها برای حل تابع*F3* نشان داده‌شده است. در شکل مشخص است که سرعت همگرایی الگوریتم گله اسب نسبت به سایر الگوریتم‌ها برای

حل تابع *F3* بیشتر بوده است. اما، الگوریتم هایی مانند کرم شب تاب، ژنتیک، شیرمورچه، و نهنگ به پاسخ بهینه این تابع همگرا نشده و نتوانسته اند پاسخ مناسبی برای این تابع بدست آورند.



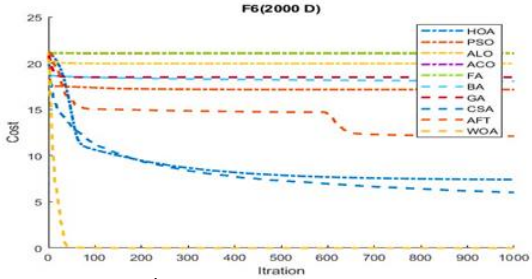
شکل۵ - منحنی همگرایی تابع *F4* در بُعد ۲۰۰۰

منحنی همگرایی الگوریتم‌ها برای حل تابع F4 در بُعد ۲۰۰۰ و تعداد تکرار ۱۰۰۰ در شکل(۵) نمایش داده‌شده است. مشخص است که الگوریتم نهنگ در تکرارهای اولیه به سطح قابل قبولی از پاسخ بهینه رسیده است. از طرف دیگر، الگوریتم گله اسب نیز در رتبه بعدی قرار داشته و سرعت همگرایی بالایی داشته است.



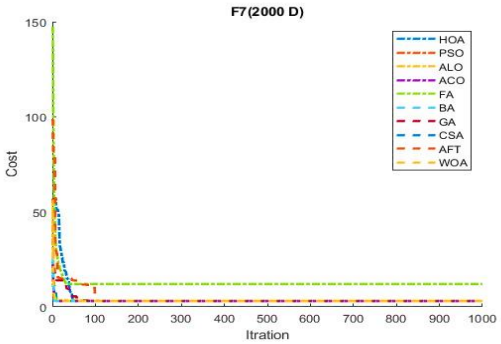
شکل ۶ - منحنی همگرایی تابع *F5* در بُعد ۲۰۰۰

شکل(۶) منحنی همگرایی الگوریتم‌ها را نشان می‌دهد که برای حل تابع F5 در بُعد ۲۰۰۰ با ۱۰۰۰ تکرار به‌کاررفته است. نتایج بسیار متفاوت هستند و عملکرد بسیار کارآمد الگوریتم نهنگ با منحنی همگرایی آن در مقایسه با روش‌های دیگر در شکل مشخص است. الگوریتم گله اسب در این تابع نتوانسته است به پاسخ بهینه همگرا شود.



شکل ۷ - منحنی همگرایی تابع *F6* در بُعد ۲۰۰۰

است. عملکرد عالی الگوریتم نهنگ ازنظر سرعت و دقت به‌خوبی قابل‌نمایش است. الگوریتم های گله اسب و کلاغ در ابتدا سرعت همگرایی بالایی داشته اند ولی در نهایت نتوانسته اند به پاسخ بهینه همگرا شوند.



شکل ۸ - منحنی همگرایی تابع *F7* در بعد ۲۰۰۰

شکل(۸) منحنی همگرایی الگوریتم‌ها را نشان می‌دهد که برای حل تابع *F7* در بُعد ۲۰۰۰ با تعداد تکرار ۱۰۰۰ به‌کاررفته است. مشخص است که نتیجه به‌دست‌آمده توسط اکثر الگوریتم‌ها تقریباً مشابه است. اگرچه رفتارهای اکثر الگوریتم‌ها در شکل (۸) یکسان به نظر می‌رسد ولی همگرایی الگوریتم‌های نهنگ و گله اسب به‌مراتب بهتر از سایر الگوریتم‌ها می‌باشد.

عملکرد الگوریتم گله اسب با سایر الگوریتم‌ها با استفاده از توابع F1 تا F7 و با بُعد ۵۰۰ در سه معیار بهترین پاسخ، انحراف معیار و زمان اجرا موردبررسی و ارزیابی قرارگرفته است، نتایج این ارزیابی در جدول (۳) نشان داده‌شده است. نتایج نشان می‌دهد که الگوریتم نهنگ در توابع F1، F4، F5 وF6 نسبت به الگوریتم‌های دیگر عملکرد بهتری در معیارهای بهترین پاسخ و انحراف معیار را دارد. همچنین، در توابع F2 وF3 به ترتیب الگوریتم‌های ازدحام ذرات، خفاش و کلونی مورچگان بهترین پاسخ و کمترین انحراف معیار را به خود اختصاص داده‌اند. در تابع F7 الگوریتم‌های ازدحام ذرات، الگوریتم خفاش، الگوریتم ژنتیک، الگوریتم جستجوی کلاغ، الگوریتم کلونی مورچگان، الگوریتم چهل دزد و علی‌بابا و الگوریتم نهنگ دارای بهترین پاسخ می‌باشند، از طرف دیگر مشخص‌شده است که در معیار زمان اجرا الگوریتم خفاش کمترین زمان اجرا در بین تمام الگوریتم‌ها را دارا است. در این مقایسه الگوریتم گله اسب بر اساس معیار بهترین پاسخ، در تابع F1 و F4 با مقدار۱,۸۲۴۵ و ۶,۷۶۶۶ رتبه سوم، در تابع F2 وF5 با

جدول ۳- مقایسه عملکرد الگوریتم‌ها با توابع تک‌وجهی و چندوجهی با بُعد ۵۰۰ در تکرار۱۰۰۰

F		HOA	PSO	FA	BA	GA	ALO	ACO	CSA	AFT	WOA
F1	Best Cost	1.8246	1.0677e+05	84432	3.6298e+05	3.3474e+05	7.5785e+05	1.5556e+06	4138.1	26382	**2.5963e-238**
	SD	3.8594	6520.5	4056.5	62746	28190	20731	17403	282.62	2311.7	**0**
	Run Times (s)	3.4828	1.9939	18.233	0.70476	2.1663	1073.2	53.067	0.92212	1.463	**4.0549**
F2	Best Cost	0.010163	0.00030749	0.0044778	0.015988	0.010304	0.0065515	0.00092985	0.00049062	0.0010031	**0.00064823**
	SD	0.0050331	4.0656e-14	0.0088807	0.0082719	0.0095264	0.0068959	0.00042573	0.00040951	0.00054426	**0.0004401**
	Run Times (s)	3.111	1.7822	26.448	0.65989	2.1389	1059.3	50.719	0.65917	0.97335	**3.8351**
F3	Best Cost	5.4121	49.133	87.247	68.099	81.604	91.194	99.226	15.778	45.086	**85.198**
	SD	12.098	2.1761	2.1085	4.5161	5.8997	2.3901	0.19638	0.77102	1.5966	**12.567**
	Run Times (s)	3.3876	1.8849	26.245	0.67041	2.1351	1066.7	52.205	0.70862	1.2959	**4.1888**
F4	Best Cost	6.7666	889.24	8.1579e+233	2.1891e+123	1089.8	3.749e+246	3.8058e+266	120.12	327.38	**6.4942e-136**
	SD	2.4328	255.49	62.876	4.8949e+123	51.863	58.134	41.248	5.0364	28.33	**1.4087e-135**
	Run Times (s)	3.3053	1.7988	30.565	0.64582	2.0238	1002.3	48.655	0.71923	1.3246	**3.8201**
F5	Best Cost	5137.6	2875.5	5723.1	1122	4658.1	6860.7	8781.7	2676.6	2344.5	**0**
	SD	166.98	130.79	108.85	161.82	84.039	159.08	130.63	82.908	79.361	**0**
	Run Times (s)	3.4006	2.2851	36.406	1.2358	2.6089	1005	47.94	1.2375	1.7382	**4.0252**
F6	Best Cost	11.49	15.703	20.977	18.318	17.711	19.918	20.957	6.1343	13.465	**1.1546e-15**
	SD	0.12391	0.56586	0.018538	1.5524	0.34642	0.16976	0.0046018	0.39501	0.7909	**1.5888e-15**
	Run Times (s)	3.3704	2.3336	50.756	1.2768	2.5905	1009.6	48.239	1.2815	1.8231	**4.0193**
F7	Best Cost	3.0039	3	24.6	3	19.283	3.6708	3	3	3	**3**
	SD	0.0076013	9.1551e-16	35.204	1.5208e-06	36.396	0.58368	0	9.4206e-16	2.9036e-15	**2.9676e-05**
	Run Times (s)	2.9448	1.6704	29.791	0.60106	2.0141	1044	50.956	0.61327	0.93831	**3.8308**

مقدار ۰,۰۱۰۱۶۳ و ۵۱۳۷,۶ رتبه هشتم، در تابع F3 با مقدار ۵,۴۱۲۱ رتبه دوم، در تابع F6 رتبه چهارم با مقدار ۱۱,۴۹ و درنهایت در تابع F7 رتبه دوم را با مقدار ۳,۰۰۹ به‌دست‌آمده است. همچنین در تابع F1 الگوریتم شیرمورچه، در تابعF2 الگوریتم خفاش، در توابع F3، F4، F5، F7 الگوریتم کلونی مورچگان، در F6 الگوریتم کرم شب‌تاب بدترین پاسخ‌ها را داشته‌اند. همچنین در ادامه

.

جدول ۴- مقایسه عملکرد الگوریتم‌ها با برخی توابع تک‌وجهی و چندوجهی با بُعد ۱۰۰۰در تکرار ۱۰۰۰

F		HOA	PSO	FA	BA	GA	ALO	ACO	CSA	AFT	WOA
F1	Best Cost	20.067	3.228e+05	6.9646e+05	7.974e+05	8.6416e+05	1.5597e+06	3.1531e+06	10665	89340	**1.249e-232**
	SD	22.99	38820	15006	1.0773e+05	1.8045e+05	30047	34229	632.82	8321.9	**0**
	Run Times (s)	4.8744	2.7009	26.246	**0.9803**	2.5511	2138.1	104.73	1.0732	1.919	**7.5936**
F2	Best Cost	0.0057487	0.00030749	0.00046202	0.017478	0.005442 6	0.00895 74	0.00466 01	**0.00030 749**	0.0046019	**0.00071 754**
	SD	0.0047339	3.9014e-10	0.0001922 3	0.02698 4	0.008405	0.00576 7	0.00877 84	**4.7644e -14**	0.0088218	**0.00047 508**
	Run Times (s)	4.5049	2.5214	31.77	1.0054	2.4295	2099.1	95.689	**0.81333**	1.2945	**7.0738**
F3	Best Cost	**0.037259**	55.95	95.144	73.391	88.528	94.157	99.615	17.944	44.89	**88.148**
	SD	**0.080777**	1.8252	1.2419	4.4523	7.6086	0.76573	0.15141	1.1871	1.8188	**5.5593**
	Run Times (s)	4.6722	2.6292	37.446	0.94331	2.3178	2027.9	93.427	**0.823**	1.5898	**6.9794**
F4	Best Cost	5.8264	690.49	4.9907e+178	858.55	1060.4	2.0504e+225	1.0798e+243	134.71	374.66	**3.6721e -135**
	SD	1.2097	27.217	12.435	136.29	59.56	6.3421	3.1232	2.5518	16.319	**7.111e-135**
	Run Times (s)	5.6923	3.193	47.333	1.2035	2.8658	2238.3	109.7	**1.1407**	2.0525	**8.3353**
F5	Best Cost	10257	7564.5	13597	3907.8	10380	13950	17723	6715.6	6230.7	**0**
	SD	124.36	320.37	98.398	525.87	232.57	87.009	145.89	86.384	146.99	**0**
	Run Times (s)	5.1787	3.3553	51.339	1.8936	3.3371	2052.2	96.232	**1.6517**	2.4204	**8.1604**
F6	Best Cost	10.962	16.548	21.07	17.329	18.175	19.903	21.009	5.9946	12.475	**4.7073e -15**
	SD	0.19443	0.64325	0.014026	1.7831	0.35387	0.02677	0.00919 84	0.11736	0.6643	**1.5888e -15**
	Run Times (s)	5.2132	3.5379	70.57	1.8529	3.3272	2015	95.407	**1.7167**	2.4622	**7.6744**
F7	Best Cost	3.003	**3**	8.4	**3**	**3**	3.0681	**3**	**3**	**3**	**3**
	SD	0.0029578	8.3081e-16	12.075	1.331e-06	3.9306e-05	0.14572	**0**	1.5384e-15	9.6787e-16	**1.1106e -05**
	Run Times (s)	4.452	2.424	39.627	0.877	2.2556	2019.7	93.779	**0.71735**	1.2701	**7.0589**

مقایسه‌های انجام شده ، عملکرد الگوریتم گله اسب را نسبت به توابع تک وجهی و چند وجهی را مورد ارزیابی قرارداده‌ایم، که نتایج نشان می‌دهد از بین مجموع توابع تک‌وجهی تابع *F2* و از بین مجموع توابع چندوجهی تابع *F7* دارای بهترین عملکرد می‌باشند.

در جدول (۴) نتایج مقایسه عملکرد الگوریتم گله اسب با سایر الگوریتم‌ها و با استفاده از توابع *F1* تا *F7* و با بُعد ۱۰۰۰ قابل‌مشاهده است. ارزیابی‌های صورت گرفته بر اساس معیارهای بهترین پاسخ، انحراف معیار و زمان اجرا انجام‌شده و نشان می‌دهند که در معیار بهترین پاسخ، در توابع *F1*، *F4*، *F5* و *F6* الگوریتم نهنگ، در تابع *F2* الگوریتم‌های ازدحام ذرات و جستجوی کلاغ، در تابع *F3* الگوریتم گله اسب و درنهایت در تابع *F7* الگوریتم‌های ازدحام ذرات، الگوریتم خفاش، الگوریتم ژنتیک، الگوریتم جستجوی کلاغ، الگوریتم کلونی مورچگان، الگوریتم چهل دزد و علی‌بابا و الگوریتم نهنگ بهترین عملکرد را دارند. همچنین، در توابع *F1*، *F4*، *F5* و *F6* الگوریتم نهنگ، در تابع F4 و درنهایت در معیار زمان اجرا در توابع *F1* و *F4* الگوریتم خفاش و در سایر توابع الگوریتم جستجوی کلاغ کمترین زمان اجرا را به خود اختصاص داده است. در مقایسه الگوریتم گله اسب بر اساس معیار بهترین پاسخ، در تابع *F1* و *F4* با مقدار ۲۰,۰۶۸ و ۵,۸۲۶۴ رتبه سوم را کسب کرده است. در توابع *F2* و *F5* با مقدار ۰,۰۰۵۷۴۸۷ و ۱۰۲۵۷ رتبه هشتم، در تابع *F3* رتبه اول با مقدار ۰,۰۳۷۲۵۹، در تابع *F6* رتبه چهارم با مقدار ۱۰,۶۹۲ و درنهایت در تابع *F7* رتبه دوم با مقدار ۳,۰۰۳ را به خود اختصاص داده است. همچنین در توابع *F1*، *F3*، *F4*،*F6*،*F7* الگوریتم کلونی مورچگان، در تابع *F2* الگوریتم خفاش و سپس در تابع *F5* الگوریتم شیرمورچه دارای بدترین پاسخ ها می‌باشند. بر اساس ارزیابی‌های انجام‌شده، عملکرد الگوریتم گله اسب روی توابع تک‌وجهی و چندوجهی مختلف بررسی شده است. نتایج نشان می‌دهد که در میان توابع تک‌وجهی، تابع *F2* و در میان توابع چندوجهی، تابع *F7* عملکرد بهتری داشته‌اند.

در جدول (۵)، عملکرد الگوریتم گله اسب با سایر الگوریتم‌ها و با استفاده از توابع *F1* تا *F7* و نیز با بعد ۲۰۰۰ مورد مقایسه قرارگرفته است. مقایسه‌های صورت گرفته در سه معیار بهترین پاسخ، انحراف معیار و زمان اجرا انجام‌شده است. نتایج نشان می‌دهند که در توابع *F1*، *F4*، *F5* و *F6* الگوریتم نهنگ، در تابع *F2* الگوریتم جستجوی کلاغ، در تابع *F3* الگوریتم گله اسب و در تابع *F7* معیار توابع *F1*، *F4*، *F5* و *F6* الگوریتم نهنگ، در تابع *F2* الگوریتم جستجوی کلاغ، در تابع *F3* الگوریتم گله اسب و در تابع *F7* الگوریتم کلونی مورچگان دارای کمترین انحراف معیار می‌باشند. در معیار زمان اجرا در توابع *F1*،*F2* و *F3* الگوریتم خفاش و در سایر توابع الگوریتم جستجوی کلاغ کمترین زمان اجرا را به خود اختصاص داده‌اند. در مقایسه الگوریتم گله اسب بر اساس معیار بهترین پاسخ، در توابع *F1*، *F4* و *F7* با مقدار،۵۷,۳۸۷، ۲,۳۰۷۱، ۳,۰۰۴۹ رتبه دوم، در تابع *F2* با مقدار ۰,۰۰۵۱۹۶۲ رتبه نهم، در تابع *F3* با مقدار ۳,۰۴۷۵ رتبه اول، در تابع *F5* با مقدار ۲۰۹۵۰ رتبه پنجم را به خود اختصاص داده است. سپس در توابع *F1*، *F4*،*F5*،*F7* الگوریتم کلونی مورچگان، در توابع *F2*، الگوریتم ژنتیک و درنهایت در توابع *F3*، *F6* الگوریتم خفاش دارای بدترین جواب می‌باشند. طبق ارزیابی‌های صورت‌گرفته، الگوریتم گله اسب در برخورد با توابع تک‌وجهی و چندوجهی نشان داد که، تابع *F2* از میان توابع تک‌وجهی و تابع *F7* از میان توابع چندوجهی بهترین عملکرد را نشان داده‌اند.

در جدول (۶)، عملکرد الگوریتم گله اسب با سایر الگوریتم‌ها و با کمک توابع *F8* تا *F10* که جزء توابع تک‌وجهی چندوجهی هستند و نیز در بعد ۱۰۰۰ ارزیابی‌شده و نتایج آن در سه معیار بهترین پاسخ، انحراف معیار و زمان اجرا در جدول (۶) نشان داده‌شده است. نتایج نشان می‌دهند که در توابع *F8*، *F9* و *F10*الگوریتم نهنگ بهترین پاسخ را دارد. همچنین، در معیار انحراف معیار در توابع *F8*، *F9* و *F10* الگوریتم نهنگ دارای کمترین انحراف معیار می‌باشد. در معیار زمان اجرا الگوریتم جستجوی کلاغ کمترین زمان اجرا را در تمام توابع دارد. در مقایسه با الگوریتم گله اسب بر اساس معیار بهترین پاسخ، در تابع *F8* با مقدار ۱۰۰۶,۱ رتبه دوم، در تابع *F9* با مقدار ۱,۸۰۴۳ رتبه سوم و در تابع *F10* با مقدار ۱,۹۶۶۱ رتبه دوم را به خود اختصاص داده است، همچنین بدترین پاسخ در توابع *F8*، *F9*، *F10* متعلق به الگوریتم کلونی مورچگان است. بر اساس نتایج ارزیابی‌ها، الگوریتم گله اسب در تحلیل توابع مختلف عملکردهای متنوعی داشته است. به طور خاص، تابع *F9* در میان توابع تک‌وجهی و تابع *F10* در میان توابع چندوجهی بالاترین کارایی را از خود نشان داده‌اند.

جدول ۵ - مقایسه عملکرد الگوریتم‌ها با برخی توابع تک‌وجهی و چندوجهی با بُعد ۲۰۰۰ در تکرار ۱۰۰۰

F		HOA	PSO	FA	BA	GA	ALO	ACO	CSA	AFT	WOA
F1	Best Cost	57.387	8.3273e+05	2.6507e+06	1.3315e+06	1.3826e+06	3.1558e+06	6.3072e+06	25150	2.0677e+05	**2.1762e-235**
	SD	60.81	16853	47655	1.9778e+05	4207	9599.1	9713.8	1215.3	19772	**0**
	Run Times (s)	8.4484	3.9138	35.516	1.6533	2.9977	4015.2	200.31	**1.2751**	2.5139	**15.439**
F2	Best Cost	0.0051962	0.00065211	**0.00047993**	0.0013739	0.0076571	0.0031096	0.0072736	0.00061272	0.00094691	**0.00087025**
	SD	0.0026987	0.00059569	**0.00020036**	0.00011945	0.011189	0.0013846	0.011336	0.00052867	0.00055406	**0.00031501**
	Run Times (s)	8.0667	4.0602	51.77	1.5574	2.9012	4031.4	199.03	**1.0897**	2.0514	**15.558**
F3	Best Cost	**0.059933**	61.828	98.92	70.567	98.302	95.306	99.837	18.748	49.941	**96.179**
	SD	**0.0041856**	2.9139	1.3791	4.7464	1.3509	0.58438	0.022863	0.26131	1.3228	**3.9476**
	Run Times (s)	8.2273	3.8506	62.52	1.5523	2.9404	4011.1	198.75	**1.1665**	2.3986	**15.705**
F4	Best Cost	2.3071	319.99	481.49	234.26	440.46	649.96	968.96	55.202	159.53	**1.7118e-134**
	SD	0.51487	9.4514	8.3351	42.212	6.3697	10.658	10.252	1.2217	5.0811	**2.9649e-134**
	Run Times (s)	8.3345	4.0094	37.05	1.8233	3.2992	4017.8	201.1	**1.3346**	2.6261	**15.884**
F5	Best Cost	20950	17260	29879	10982	22282	28380	36025	15369	15273	**0**
	SD	120.76	783.42	135.31	932.8	679.24	151.99	328.21	405.97	365.42	**0**
	Run Times (s)	9.124	5.3549	75.688	2.9437	4.3495	4029.6	199.11	**2.4486**	3.6741	**16.087**
F6	Best Cost	7.3987	17.134	21.109	18.05	18.468	19.953	21.054	6.0076	12.096	**3.9968e-15**
	SD	6.0386	0.77625	0.0035291	1.6775	0.41664	0.020791	0.0060662	0.091559	0.66297	**3.5527e-15**
	Run Times (s)	9.1055	5.4939	103.86	3.011	4.6512	4006	200.05	**2.4594**	3.8557	**16.465**
F7	Best Cost	3.0049	**3**	12	**3**	3.0005	3.174	**3**	**3**	**3**	**3**
	SD	0.0061515	3.1402e-16	15.588	1.118e-06	0.00065789	0.21428	**0**	2.0592e-15	1.1749e-15	**4.1234e-06**
	Run Times (s)	7.8731	3.6856	58.893	1.4599	2.8544	4009.5	198.91	**1.0578**	1.9528	**15.452**

جدول ۶ - مقایسه عملکرد الگوریتم‌ها با برخی توابع تک‌وجهی و چندوجهی با بُعد ۱۰۰۰ در تکرار ۱۰۰۰

F		HOA	PSO	FA	BA	GA	ALO	ACO	CSA	AFT	WOA
F8	Best Cost	1006.1	3.0403e+08	1.1275e+09	1.1343e+09	1.8682e+09	4.4992e+09	1.5169e+10	3.5532e+05	1.9238e+07	**992.51**
	SD	13.957	7.0467e+07	7.9825e+07	3.4842e+08	5.0263e+08	1.4613e+08	1.8097e+08	22004	1.2187e+06	**1.2299**
	Run Times (s)	5.366	2.9626	28.183	1.1843	2.5864	2100.9	98.205	1.1073	1.9814	**7.6197**
F9	Best Cost	1.8043	3429.1	1650	804.36	19802	76658	2.4358e+05	7.2117	236.1	**0.0026415**
	SD	0.51449	759.54	138.89	299.15	6293.7	4285.3	6851.2	0.35537	28.345	**0.0024009**
	Run Times (s)	7.8737	5.6256	70.563	4.1476	5.5203	2042.4	99.212	3.9224	4.6663	**10.81**
F10	Best Cost	1.9661	3080.8	6307.2	6627.6	8307.5	13898	28380	94.868	779.67	**0**
	SD	1.2292	343.01	371.5	745.66	824.08	484.64	366.59	2.8769	70.369	**0**
	Run Times (s)	8.5686	4.3867	50.503	2.8421	4.3573	2372.4	113.83	2.3216	3.2216	**10.304**

5. نتیجه‌گیری و پیشنهادهای آینده

الگوریتم‌های بهینه‌سازی فراابتکاری قادر هستند تا طیف گسترده‌ای از مسائل بهینه‌سازی غیرخطی را به‌طور بهینه حل کنند [۲۹] که این به دلیل ویژگی منحصربه‌فرد آن‌ها در تنوع و قابلیت بهره‌برداری است. بااین‌حال، هر الگوریتم دارای فلسفه و پس‌زمینه متفاوتی است که با استراتژی جستجوی پویا ترکیب‌شده و منجر به تفاوت در همگرایی، بهترین پاسخ و زمان محاسبه می‌شود. پژوهش حاضر، عملکرد الگوریتم گله اسب را با ۹ الگوریتم فراابتکاری دیگر شامل الگوریتم‌های کلونی مورچگان، چهل دزد و علی‌بابا، شیر مورچه، خفاش، جستجوی کلاغ، کرم شب‌تاب، ژنتیک، ازدحام ذرات و الگوریتم نهنگ مقایسه نموده است. برای انجام مقایسه از ۱۰ تابع تست استاندارد استفاده‌شده است. همچنین، مقایسه‌ها بر اساس سه شاخص بهترین جواب، انحراف معیار و زمان اجرا در بُعدهای ۵۰۰، ۱۰۰۰ و ۲۰۰۰ انجام‌شده است. با توجه به نتایج شبیه‌سازی به‌دست‌آمده از شبیه‌ساز متلب، مشخص گردید که الگوریتم نهنگ به‌طور قابل‌توجهی در مقایسه با سایر الگوریتم‌ها در اکثر توابع بهتر عمل می‌کند. الگوریتم نهنگ به دلیل استفاده از مکانیسم‌های محاصره طعمه و چرخش اطراف طعمه، توانایی زیادی برای یافتن جواب بهینه در فضای جستجوی مسئله را دارد. در پژوهش میر نعیمی و همکاران، الگوریتم گله اسب با الگوریتم‌های چند نظمی، شعله پروانه، تکاملی تفاضلی و گرگ خاکستری مقایسه شده‌اند. در آن پژوهش، الگوریتم گله اسب عملکرد بهتری را در معیارهای بهترین پاسخ و انحراف معیار ارائه نمود. همچنین، در معیار زمان اجرا الگوریتم شعله و پروانه کمترین زمان اجرا را نشان داد. علاوه بر این، توابع تک‌وجهی و چندوجهی را مورد ارزیابی و بررسی قرار دادیم که، استفاده از توابع تک‌وجهی و چندوجهی در الگوریتم‌های تقریبی بستگی به نوع مسئله و پیچیدگی آن دارد. در حالی که توابع تک وجهی برای مسائل ساده‌تر و یک‌بعدی مناسب هستند، توابع چندوجهی در مواردی که نیاز به تحلیل و بهینه‌سازی در فضای چندبعدی باشد، کاربرد دارند. انتخاب نوع تابع مناسب می‌تواند به کاهش پیچیدگی محاسباتی و بهبود کارایی الگوریتم‌های تقریبی کمک کند، نتایج نشان داد که در جداول (۳)، (۴)، (۵) مجموعه توابع تست تک‌وجهی تابع *F2* و در چندوجهی *F7* و همچنین در جدول(۶) تابع تک‌وجهی*F9* و در تابع چندوجهی تابع *F10* دارای برترین عملکرد می‌باشد .بنابراین، می‌توان گفت بااینکه الگوریتم گله اسب در بین الگوریتم‌های مقایسه شده عملکرد قابل قبولی داشته است اما بر اساس نتایج تحقیق حاضر مشخص‌شده است که الگوریتم گله اسب همیشه دارای

عملکرد مناسبی نیست و لااقل نسبت به برخی از الگوریتم‌های مقایسه شده در این تحقیق از عملکرد پایین‌تری برخوردار است. لذا، برای انتخاب الگوریتم فراابتکاری گله اسب باید اندازه مسئله را در نظر داشت چراکه تعداد پارامترهای اولیه این الگوریتم زیاد بوده و نیز در بُعدهای بزرگ عملکرد مناسبی ندارد. یافته‌های این مقاله ضرورت انجام مطالعات مقایسه‌ای بیشتر در زمینه‌های مختلف به‌ویژه در مسائل دنیای واقعی نشان می‌دهد. نتایج به‌دست‌آمده می‌توانند به‌عنوان الگو برای پژوهش‌های آینده در زمینه بهبود عملکرد الگوریتم‌های فراابتکاری مورداستفاده بیشتر در زمنیه‌های مختلف به‌ویژه در مسائل دنیای واقعی را نشان دهد. نتایج به‌دست‌آمده می‌توانند به‌عنوان الگو برای پژوهش‌های آینده در زمینه بهبود عملکرد الگوریتم‌های فراابتکاری مورداستفاده قرار بگیرند. یکی از پژوهش‌هایی که به‌عنوان کارهای آتی می‌توان در نظر داشت فرآیند توسعه الگوریتم گله اسب برای حل مسائل بهینه‌سازی چندهدفه است. همچنین مقایسه عملکرد این الگوریتم با الگوریتم‌های جدیدتری مانند الگوریتم جستجوی مواد غذایی مارماهی[1] *Sandpiper* [۳۰]، الگوریتم بهینه‌سازی ماهی *Pufferfish* [2] [۳۱]، الگوریتم بهینه‌سازی بوتاکس[3] *[32]*، الگوریتم بهینه‌سازی خرچنگ[4] [۳۳] و .... برای مطالعات آینده پیشنهاد می‌شود.

مراجع:

[1]. Bandaru, S. and K. Deb, *Metaheuristic techniques*. Decision sciences, 2016: p. 693-750.

[2]. Gavish, B. and H. Pirkul, *Efficient algorithms for solving multiconstraint zero-one knapsack problems to optimality*. Mathematical programming, 1985. **31**: p. 78-105.

[3]. Neghabi, A.A., et al., *Energy-aware dynamic-link load balancing method for a software-defined network using a multi-objective artificial bee colony algorithm and genetic operators.* IET communications, 2020. **14**(18): p. 3284-3293.

[4]. Khodadadi, R., G. Ardeshir, and H. Grailu, *Compression of face images using meta-heuristic algorithms based on curvelet transform with variable bit allocation.* Multimedia Systems, 2023. **29**(6): p. 3721-3744.

[5]. Talbi, E.-G., *Metaheuristics: from design to implementation*. 2009: John Wiley & Sons.

[6]. Gandomi, A.H., et al., *Metaheuristic algorithms in modeling and optimization.* Metaheuristic applications in structures and infrastructures, 2013. **1**: p. 1-24.

[7]. Tilahun, S.L. and H.C. Ong, *Prey-predator algorithm: a new metaheuristic algorithm for optimization problems*. International Journal of Information Technology & Decision Making, 2015. **14**(06): p. 1331-1352.

[8]. MiarNaeimi, F., G. Azizyan, and M. Rashki, *Horse herd optimization algorithm: A nature-inspired algorithm for high-dimensional optimization problems.* Knowledge-Based Systems, 2021. **213**: p. 106711.

[9]. Den Besten, M., T. Stützle, and M. Dorigo. *Ant colony optimization for the total weighted tardiness problem*. in *Parallel Problem Solving from Nature PPSN VI: 6th International Conference Paris, France, September 18–20, 2000 Proceedings 6*. 2000. Springer.

[10]. Alphonse, A.S., S. Abinaya, and S. Abirami, *Alibaba and forty thieves algorithm and novel Prioritized Prewitt Pattern (PPP)-based convolutional neural network (CNN) using hyperspherically compressed weights for facial emotion recognition.* Journal of Visual Communication and Image Representation, 2023. **97**: p. 103948.

[11]. Saleem, F., et al., *Ant lion optimizer based clustering algorithm for wireless body area networks in livestock industry.* IEEE Access, 2021. **9**: p. 114495-114513.

[12]. Bangyal, W.H., J. Ahmad, and H.T. Rauf, *Optimization of neural network using improved bat algorithm for data classification.* Journal of Medical Imaging and Health Informatics, 2019. **9**(4): p. 670-681.

[13]. Hussien, A.G., et al., *Crow search algorithm: theory, recent advances, and applications.* IEEE Access, 2020. **8**: p. 173548-173565.

[1] Sandpiper food search algorithm

[2] Pufferfish optimization algorithm

[3] Botox optimization algorithm

[4] Crayfish optimization algorithm

[14]. Mangharam, R., A. Rowe, and R. Rajkumar, *FireFly: a cross-layer platform for real-time embedded wireless networks.* Real-Time Systems, 2007. **37**: p. 183-231.
[15]. Gen, M. and L. Lin, *Genetic algorithms and their applications*, in *Springer handbook of engineering statistics*. 2023, Springer. p. 635-674.
[16]. Kennedy, J. and R. Eberhart. *Particle swarm optimization (PSO).* in *Proc. IEEE international conference on neural networks, Perth, Australia*. 1995.
[17]. Mirjalili, S. and A. Lewis, *The whale optimization algorithm.* Advances in engineering software, 2016. **95**: p. 51-67.
[18]. Saleh, B. and A. Neghabi, *Optimal Routing-Clustering Aware of Energy Consumption in Wireless Sensor Networks based on Deep Tree Learning.* Transactions on Machine Intelligence, 2023. **6**(4): p. 236-247.
[19]. Fidanova, S. and S. Fidanova, *Ant colony optimization.* Ant Colony Optimization and Applications, 2021: p. 3-8.
[20]. Huang, K.-W. and Z.-X. Wu, *CPO: a crow particle optimization algorithm.* International Journal of Computational Intelligence Systems, 2018. **12**(1): p. 426-435.
[21]. Abdolmanafi, S., et al., *The Impact of Content Produced on Instagram Social Network on Successful Economic Services of Isfahan in Corona Crisis Using a Combination of Genetic Algorithm and Forbidden Search Algorithm.* International Journal of Digital Content Management, 2023.
[22]. Gad, A.G., *Particle swarm optimization algorithm and its applications: a systematic review.* Archives of computational methods in engineering, 2022. **29**(5): p. 2531-2561.
[23]. Mohamed, A.W., et al., *Evaluating the performance of meta-heuristic algorithms on CEC 2021 benchmark problems.* Neural Computing and Applications, 2023. **35**(2): p. 1493-1517.
[24]. Sadhu, T., et al., *A comparative study of metaheuristics algorithms based on their performance of complex benchmark problems.* Decision Making: Applications in Management and Engineering, 2023. **6**(1): p. 341-364.
[25]. Rajabi Moshtaghi, H., A. Toloie Eshlaghy, and M.R. Motadel, *Comparing and Ranking of Meta-Heuristic Algorithms Using Group Decision Making Methods.* ۲۰۲۲. مدیریت صنعتی, ۵(58): p. 65.
[26]. Schott, F., et al., *Performance measure and tool for benchmarking metaheuristic optimization algorithms.* Journal of Applied and Computational Mechanics, 2021.
[27]. Mousavirad, S.J., et al. *A benchmark of recent population-based metaheuristic algorithms for multi-layer neural network training*. in *Proceedings of the 2020 genetic and evolutionary computation conference companion*. 2020.
[28]. Ezugwu, A.E., et al., *A comparative study of meta-heuristic optimization algorithms for 0–1 knapsack problem: Some initial results.* IEEE Access, 2019. **7**: p. 43979-44001.
[29]. Ismail, I. and A.H. Halim, *Comparative study of meta-heuristics optimization algorithm using benchmark function.* International Journal of Electrical and Computer Engineering (IJECE), 2017. **7**(3): p. 1643-1650.
[30]. Wira, J.C., *Sandpiper Food Search Algorithm: A New Optimization Approach for Agents with Limited Knowledge.* 2024.
[31]. Al-Baik, O., et al., *Pufferfish Optimization Algorithm: A New Bio-Inspired Metaheuristic Algorithm for Solving Optimization Problems.* Biomimetics, 2024. **9**(2): p. 65.
[32]. Hubálovská, M., Š. Hubálovský, and P. Trojovský, *Botox Optimization Algorithm: A New Human-Based Metaheuristic Algorithm for Solving Optimization Problems.* Biomimetics, 2024. **9**(3): p. 137.
[33]. Jia, H., et al., *Crayfish optimization algorithm.* Artificial Intelligence Review, 2023. **56**(Suppl 2): p. 1919-1979.

Evaluation and comparison of the performance of the horse herd algorithm with some meta-heuristic algorithms

Jalal Iziy[1], Ali Akbar Neghabi[2]

1. PhD student, Department of Computer Engineering and Information Technology, Islamic Azad **University, Sabzevar Branch, Sabzevar, Iran**
2. Assistant Professor, Department of Computer Engineering and Information Technology, Islamic Azad **University, Sabzevar Branch, Sabzevar, Iran**

*****Corresponding Author: Ali Akbar Neghabi

ABSTRACT

Meta-heuristic algorithms have become a popular tool for solving many problems in real-world applications due to their ability to overcome many problems in traditional optimization. The performance of these algorithms is different in different problems, so it is necessary to make a careful evaluation of their performance. One of the meta-heuristic algorithms that has recently attracted a lot of attention is the horse herd algorithm, which is inspired by the behavior of horses at different ages. The purpose of this research is to compare and evaluate the performance of the horse herd algorithm with some other meta-heuristic algorithms for solving complex problems. In this study, the performance of horse herd algorithm is compared with 9 other meta-heuristic algorithms including ant colony, forty thieves and Alibaba, ant lion, bat, crow search, firefly, genetics, particle swarm and whale algorithm. In this evaluation, 10 standard test functions were used and the performance of the algorithms was compared based on three criteria: best answer, standard deviation, and execution time in dimensions of 500, 1000, and 2000. The simulation results show that due to the large number of parameters that the horse herd algorithm has, one of the challenges of this algorithm is to adjust its parameters. Also, in high dimensions, the horse herd algorithm does not perform well compared to other compared meta-heuristic algorithms.

Keywords: meta-heuristic algorithms, optimization, horse herd algorithm, whale algorithm, forty thieves algorithm and Alibaba.