

بهینه‌سازی تاخیر و مصرف انرژی در زمان‌بندی محاسبات اینترنت اشیا مبتنی بر معماری ابر-لبه با استفاده

از الگوریتم جستجوی گرانشی بهبودیافته مبتنی بر منطق فازی

امیرمحمد صانعی^۱، محمد فرشی^{۲*}، سپهر ابراهیمی مود^۳

^۱کارشناسی ارشد، گروه علوم کامپیوتر، دانشکده علوم ریاضی، دانشگاه یزد، یزد، ایران

^۲دانشیار، گروه علوم کامپیوتر، دانشکده علوم ریاضی، دانشگاه یزد، یزد، ایران

^۳استادیار، گروه علوم کامپیوتر، دانشکده علوم ریاضی، دانشگاه یزد، یزد، ایران

چکیده

مقاله پژوهشی

تاریخ دریافت:

۱۴۰۵/۰۲/۲۶

تاریخ پذیرش:

۱۴۰۵/۰۶/۳۱

کلیدواژه‌ها:

اینترنت اشیا، زمان‌بندی محاسبات،
محاسبات ابر-لبه، بهینه‌سازی،
الگوریتم جستجوی گرانشی،
سیستم استنتاج فازی

نویسنده مسئول:

mfarshi@yazd.ac.ir

در این مطالعه، به بررسی راهکارهای زمان‌بندی محاسبات در شبکه اینترنت اشیا دارای لایه‌های محاسباتی ابر و لبه با هدف کمینه‌سازی تاخیر و انرژی مصرفی پرداخته شده است. برای حل مسئله ذکر شده، از الگوریتم جستجوی گرانشی بهبودیافته (IGSA) استفاده شده است. از مهم‌ترین معایب الگوریتم جستجوی گرانشی (GSA)، نبود تعادل مناسب بین قابلیت‌های کاوش و بهره‌وری می‌باشد. در این مقاله به منظور رفع این چالش، از سیستم استنتاج فازی برای عبور از گرفتار شدن در بهینه‌های محلی استفاده شده است. از ۱۲ تابع آزمایشی چندوجهی برای ارزیابی الگوریتم پیشنهادی استفاده شده است. همچنین از این الگوریتم برای حل یک مسئله مهندسی مربوط به زمان‌بندی محاسبات در شبکه اینترنت اشیا استفاده شده است. این کار، با استفاده از یک شبیه‌ساز شبکه اینترنت اشیا دارای دو مدل محاسباتی ابر و لبه، با دو هدف کمینه‌سازی تاخیر و انرژی که به صورت وزن‌دار به یک مسئله تک‌هدفه تبدیل شده، انجام شده است. الگوریتم پیشنهادی در کنار الگوریتم‌های جستجوی گرانشی، گرگ خاکستری، سینوس کسینوس، بهینه‌سازی شامپانزه و بهینه‌سازی خرچنگ در این دو محیط آزمایشی، مورد ارزیابی قرار گرفته‌اند. نتایج اجرا در هر دو محیط آزمایشی نشان می‌دهند، الگوریتم پیشنهادی با بهبود روند جستجو فضای جواب‌های شدنی به صورت پویا، علاوه بر یافتن پاسخی با شایستگی (Fitness) بهتر نسبت به الگوریتم جستجو گرانشی پایه، توانسته در مقایسه با دیگر الگوریتم‌های فراابتکاری کارایی برتر را از منظر آزمون فریدمن را با مقادیر رتبه ۲/۴۰۹ در حل توابع آزمایشی استاندارد و ۲ در زمان‌بندی محاسبات به خود اختصاص دهد.

doi : 10.22034/ABMIR.2026.24574.1244

E-ISSN: [2821-2037](https://doi.org/10.22034/ABMIR.2026.24574.1244)

/The Author 2026. Published by Yazd University This is an open access article

under the CC BY 4.0 License <https://creativecommons.org/licenses/by/4.0/>.



۱- مقدمه

سرویس (PaaS) و یا نرم‌افزار به‌عنوان سرویس (SaaS) می‌باشد. مدل محاسباتی لبه، با نزدیک کردن منابع محاسباتی به مراکز تولید داده، توانایی پردازش اطلاعات حساس به تاخیر را به‌صورت محلی و در زمان کوتاه در اختیار شبکه قرار می‌دهد. محاسبات لبه با وجود توان پردازشی محدود، حریم خصوصی امن‌تر و کاهش مصرف پهنای باند را به همراه دارد [۴]. اگرچه محاسبات ابری توان پردازشی گسترده‌ای فراهم می‌کند، انتقال همه داده‌ها و محاسبات شبکه اینترنت اشیا به مراکز ابری باعث افزایش تاخیر ارتباطی و مصرف انرژی شبکه خواهد شد. از سوی دیگر، محدودیت منابع در لبه مانع از اجرای تمام وظایف در این بخش از شبکه اینترنت اشیا می‌گردد. بنابراین، ارائه یک راهکار زمان‌بندی هوشمند برای توزیع بهینه وظایف بین این دو لایه ضروری است. استفاده از منابع محاسباتی مختلف به‌صورت هم‌زمان برای انجام پردازش‌ها، نیاز به تقسیم‌بندی بهینه وظایف پردازشی بین منابع محاسباتی در دسترس دارد. به این تقسیم‌بندی وظایف که با اهداف کمینه‌سازی تاخیر زمانی در پاسخ‌گویی به نیازهای پردازشی و کمینه‌سازی انرژی مصرفی منابع پردازشی انجام می‌شود، زمان‌بندی محاسبات نیز گفته می‌شود. یکی از چالش‌های اصلی این حیطه، زمان‌بندی محاسبات در شبکه اینترنت اشیا است که یک مسئله پیچیده از نوع NP-hard می‌باشد و یافتن پاسخ دقیق برای آن، به روش‌های دقیق در زمان معقول غیرممکن است. الگوریتم‌های فراابتکاری، با قابلیت تصادفی بودن و جستجوی محلی، به کاوش و بهره‌وری در فضای جواب‌های شدنی مسئله می‌پردازند [۵] و یک راهکار مناسب برای حل مسائل زمان‌بندی محاسبات هستند. در این مقاله، از یک مدل سیستم ریاضی برای شبیه‌سازی این محیط استفاده شده است و با ذکر کلیات و محدودیت‌ها، به ارائه یک محیط برای آزمایش کارایی الگوریتم پیشنهادی در حل مسئله مذکور پرداخته شده است.

الگوریتم‌های فراابتکاری انواع مختلفی دارند که الگوریتم‌های مبتنی بر قوانین فیزیک، یکی از آنها می‌باشد. این الگوریتم‌ها، با الهام از قوانین فیزیک حاکم بر محیط، شکل گرفته‌اند و بر همین اساس در طول اجرای الگوریتم، نمونه جواب‌های جدیدی تولید می‌کنند.

دستگاه‌های حوزه اینترنت اشیا در همه جوانب زندگی، از خانه‌های هوشمند و مدیریت شهری تا صنعت و حوزه بهداشت و درمان حضور دارند، قابلیت نظارت، اتخاذ تصمیم و کنترل از راه دور را برای کاربر به همراه دارند [۱]. در مطالعه [۲] اشاره شده، با نصب حسگرهایی بر روی پمپ، توربین یا موتور و جمع‌آوری اطلاعاتی مثل دما، فشار، لرزش و جریان الکتریکی و همچنین تحلیل آنها، خرابی تجهیزات قبل از وقوع پیش‌بینی شده است. همچنین این پیش‌بینی به‌موقع توانسته زمان توقف خط تولید را کاهش و طول عمر تجهیزات را افزایش دهد. طبق گزارش‌ها در سال ۲۰۲۱ تعداد دستگاه‌های فعال در این شبکه، بیشتر از ۱۰ میلیارد دستگاه بوده است و پیش‌بینی می‌شود تا سال ۲۰۳۰ این تعداد از ۲۵٫۴ میلیارد دستگاه فراتر رود [۳]. با وجود رشد سریع سامانه‌های اینترنت اشیا، محدودیت توان پردازشی، ظرفیت باتری و نیاز روزافزون به پاسخ‌دهی بلادرنگ، اجرای مستقیم وظایف روی دستگاه‌های انتهایی با چالش‌هایی مواجه شده است. بنابراین، تخصیص هوشمند وظایف میان منابع محاسباتی ناهمگون ابر و لبه به یکی از مسائل اساسی در طراحی شبکه‌های اینترنت اشیا تبدیل شده است که موضوع اصلی این مطالعه نیز می‌باشد. با فراگیر شدن اینترنت اشیا در صنایع مختلف، حجم داده‌های تولید شده هر روز افزایش یافته و همچنین نیاز به پردازش و ذخیره‌سازی این داده‌ها رشد چشمگیری داشته است. پردازش داده‌های مذکور در حجم بالا، نیاز به پردازنده‌های قوی را به وجود آورده است که قابلیت استقرار در هر شرایط و مکانی را ندارند و باعث ایجاد مدل محاسباتی ابر شده‌اند. در کنار نیاز به پردازش‌های سنگین، این شبکه نیاز به پردازش داده‌ها در زمان کوتاه را نیز دارد که باعث تعریف مدل محاسباتی لبه شده است. منابع محاسباتی در شبکه اینترنت اشیا را می‌توان به دو لایه ابر و لبه تقسیم‌بندی کرد که هر کدام قابلیت‌های خاص خود را دارند. در مدل محاسباتی ابری، داده‌ها پس از جمع‌آوری به یک سرور دور انتقال یافته و در آنجا پردازش می‌شوند. این مدل محاسباتی، دارای قابلیت مقیاس‌پذیری و انعطاف‌پذیری زیاد در توان پردازشی است [۴] و ارائه سرویس در آن به‌صورت زیرساخت به‌عنوان سرویس (IaaS)، پلتفرم به‌عنوان

کلیدی الگوریتم، در طول اجرای الگوریتم به جستجو گسترده‌تر فضای جواب‌های شدنی می‌پردازد. در این راهکار تعداد مناسب نمونه جواب‌های غیر کارآمد به کمک منطق فازی برآورد شده و به همان تعداد نمونه جواب‌هایی در نقاط پرتراکم با نمونه جواب‌هایی در مناطق کم‌تراکم جایگزین می‌شود. منطق فازی یک چهارچوب ریاضی برای استدلال در شرایطی است که قطعیت وجود ندارد. این منطق که توسط پروفیسور لطفی زاده در سال ۱۹۶۵ معرفی شده، برای مدیریت اطلاعات مبهم، قوانین صفر و یک را گسترش داده است [۹]. در این منطق، برخلاف منطق بولی که در آن یک عضو به طور کامل در یک مجموعه عضو هست یا نیست، عضویت اعضا در مجموعه‌های مختلف با یک عددی موسوم به درجه عضویت که بین صفر و ۱ است، بیان می‌گردد. این منطق به ارائه مفاهیم کیفی که مبهم هستند، کمک بسزایی می‌کند. نمونه‌های متعددی از استفاده از منطق فازی در بهبود الگوریتم‌های فراابتکاری وجود دارد. در طی این کار تحقیقاتی، از منطق فازی برای تشخیص تعداد مناسب نمونه جواب‌های نه‌چندان کارآمد، استفاده شده است.

همچنین با پیاده‌سازی روش پیشنهادی و مقایسه آن با چند الگوریتم فراابتکاری دیگر، مشخص شده است که روش پیشنهادی توانسته است:

- از نقاط بهینه محلی و همگرایی زودهنگام دور بماند؛
- تنوع نمونه جواب‌ها را حفظ کند؛
- جستجو کامل‌تری در فضای جواب‌های شدنی انجام دهد؛
- در مقایسه با دیگر الگوریتم‌های اجرا شده، عملکرد بهتری در حل توابع چندوجهی داشته باشد؛
- و همچنین جواب بهتری برای مسئله زمان‌بندی محاسبات در محیط‌های مختلف به دست آورد.

در بخش بعدی از این مقاله به مرور پژوهش‌های مرتبط در این زمینه پرداخته شده است و در بخش سوم به ارائه راهکار پیشنهادی این مطالعه برای بهبود الگوریتم GSA پرداخته می‌شود و پس از آن مدل ریاضی شبیه‌سازی شده شبکه اینترنت اشیا، جهت استفاده از الگوریتم پیشنهادی در زمان‌بندی وظایف آن ارائه می‌شود. در

الگوریتم GSA، یک نمونه بارز از الگوریتم‌های فراابتکاری مبتنی بر قوانین فیزیک است که در سال ۲۰۰۹ توسط راشدی و همکارانش و با الهام از قانون گرانش و قوانین نیوتن در این زمینه، معرفی شده است. همان‌طور که در مقاله [۶] بدان اشاره شده، الگوریتم GSA فضای جواب‌های شدنی را به کیهان و نمونه جواب‌ها را به اجرام آسمانی در این کیهان تشبیه می‌کند. در این فضا، افزایش شایستگی هر نمونه جواب (از منظر تابع هدف) منجر به افزایش جرم آن نمونه جواب خواهد شد. از جهتی با افزایش جرم یک نمونه جواب، میزان نیروی گرانش وارد شده از طرف آن به سایر نمونه جواب‌ها افزایش می‌یابد که خود باعث تبادل اطلاعات بین نمونه جواب‌ها، مربوط به محیط پیرامون هر کدام، خواهد شد. به دلیل قابلیت ذاتی الگوریتم GSA در تبادل اطلاعات میان عوامل جستجوگر (نمونه جواب‌ها) و ساختار جمعیتی آن، برای بهینه‌سازی مسائل تخصیص منابع با فضای جواب‌های شدنی بزرگ مناسب است، هرچند ضعف آن در حفظ تنوع جمعیت انگیزه اصلی این پژوهش است.

قابلیت‌های کاوش و بهره‌وری را می‌توان دو قابلیت اصلی در الگوریتم‌های فراابتکاری دانست. قابلیت کاوش، امکان کشف راه‌حل‌های جدید و متنوع را فراهم می‌کند. از طرف دیگر بهره‌وری، راه‌حل‌های موجود را بهبود می‌بخشد و همگرایی را شتاب می‌دهد [۷]. با بررسی روند اجرای الگوریتم GSA و با استناد به مقاله [۸]، می‌توان پی برد که اصلی‌ترین عیب الگوریتم GSA، عدم وجود تعادل مناسب بین قابلیت‌های کاوش و بهره‌وری است که منجر به همگرایی زودرس نمونه جواب‌ها در یک بهینه محلی در مسائل پیچیده و چند مدی می‌شود.

با وجود ارائه روش‌های مختلف برای زمان‌بندی وظایف در شبکه اینترنت اشیا دارای معماری محاسباتی ابر-لبه، بسیاری از روش‌های موجود به دلیل فضای جستجوی بزرگ، ناهمگونی منابع و ماهیت پویای شبکه‌های اینترنت اشیا، با مشکل حفظ تعادل میان جستجوی گسترده و بهبود محلی مواجه هستند.

در این تحقیق، برای گذر از این مشکل، راهکاری به کمک منطق فازی برای بازمقداردهی نمونه جواب‌های الگوریتم، اندیشیده شده است. این راهکار، به‌صورت پویا و بدون تغییر در پارامترهای

در مقاله [۱۸]، پژوهشگران برای کاهش دو هدف تاخیر و انرژی مصرفی، با استفاده از مفهوم غلبه بهینه پرتو، از الگوریتم GWO چندهدفه (MOGWO) استفاده کردند. در این روش از مجموعه جواب‌های غیرمغلوب برای هدایت جمعیت و حفظ تنوع نمونه جواب‌ها استفاده می‌شود. نتایج شبیه‌سازی نشان می‌دهد، این الگوریتم پیشنهادی در مقایسه با روش‌های دیگر توانسته به طور هم‌زمان تاخیر اجرای وظایف و انرژی مصرفی را به طور چشمگیری کاهش و کیفیت جبهه پرتو را ارتقا دهد. با این حال، در این مطالعه اشاره شده است که استفاده از سازوکارهای چندهدفه مانند مرتب‌سازی غیرمغلوب و مدیریت آرشیو، پیچیدگی محاسباتی و زمان اجرای الگوریتم را افزایش می‌دهد. علاوه بر این، کیفیت جواب‌ها به نحوه تنظیم پارامترهای الگوریتم و اندازه آرشیو وابسته است و در مسائل با تعداد بسیار زیاد وظایف، هزینه محاسباتی افزایش می‌یابد.

در بسیاری مقالات، از الگوریتم ازدحام ذرات برای مسائل زمان‌بندی (PSO-based scheduling) استاندارد با تغییر در نمایش جواب و هماهنگی با تابع هدف برای مقایسه کارایی، استفاده شده است، لذا در این مطالعه نیز از این الگوریتم برای مقایسه کارایی الگوریتم پیشنهادی استفاده شده است. در مطالعه [۱۹] از دیگر پژوهش‌های حوزه زمان‌بندی، نسخه‌های متنوع این الگوریتم و روش‌های ترکیبی آن با سایر الگوریتم‌های فراابتکاری مورد بررسی قرار گرفته است. نتایج این بررسی نشان می‌دهد که PSO به دلیل سادگی ساختار، تعداد کم پارامترهای کنترلی و سرعت همگرایی مناسب، یکی از پرکاربردترین الگوریتم‌ها در مسئله زمان‌بندی محسوب می‌شود. با این حال، نویسندگان اشاره می‌کنند که PSO در مسائل پیچیده و با ابعاد بزرگ، مستعد همگرایی زودرس و گیر افتادن در بهینه‌های محلی است و کاهش تنوع جمعیت در مراحل پایانی جستجو منجر به کاهش کیفیت جواب‌ها می‌شود.

بررسی‌های بسیاری برای اصلاح و تغییر الگوریتم GSA برای کاربرد آن در زمینه‌های مختلف انجام شده است. از جمله این تغییرات و اصلاحات می‌توان به الگوریتم‌های ترکیبی اشاره کرد. الگوریتم ترکیبی ازدحام ذرات و جستجوی گرانشی (PSO-GSA)

بخش چهارم به مرور و تحلیل نتایج اجرا و در بخش پنجم به نتیجه‌گیری این پژوهش پرداخته خواهد شد.

۲- کارهای مرتبط

از مهم‌ترین چالش‌های معماری محاسباتی ابر که در مرجع [۱۰] بدان اشاره شده است، افزایش تاخیر، متعادل‌سازی بار و مصرف انرژی بالا در مراکز محاسباتی می‌باشد. از طرفی معماری محاسباتی لبه دارای مشکلاتی همچون، محدودیت توان پردازشی، محدودیت انرژی در دسترس منابع، ناهمگونی منابع، محیط پویا و پیچیدگی زمان‌بندی وظایف می‌باشد که در کار تحقیقاتی [۱۱] مورد بررسی قرار گرفته است.

همان‌طور که گفته شد، تقسیم‌بندی وظایف، یک مسئله پیچیده از نوع NP-hard است و روش‌های دقیق برای آن پاسخ‌گو نیست. در مطالعه [۱۲]، تحقیقات حوزه زمان‌بندی وظایف مورد بررسی قرار گرفته است و معیارهای اصلی این مسئله بیان شده است. این مطالعه نشان می‌دهد، با وجود الگوریتم‌های متنوع برای زمان‌بندی وظایف، اما چالش‌هایی نظیر ناهمگونی منابع، محیط پویا، مقیاس‌پذیری، تحرک کاربران و ایجاد تعادل میان اهداف، همچنان به‌صورت مسئله باقی مانده‌اند. همچنین روش‌های چندهدفه به عنوان یکی از مهم‌ترین مسیرهای تحقیقاتی آینده معرفی شده، که موضوع اصلی این مطالعه است. الگوریتم‌های فراابتکاری که یکی از بهترین الگوریتم‌های تقریبی برای حل مسائل بهینه‌سازی پیچیده هستند، این توانایی را دارند که بر پایه روش‌های تصادفی جواب بهینه و یا نزدیک به آن را بیابند [۱۳].

برای مقایسه کارایی الگوریتم پیشنهادی، از چند الگوریتم فراابتکاری قدیمی‌تر ولی مشهور در کنار الگوریتم‌های به‌روز و جدید استفاده شده است. الگوریتم گرگ خاکستری (GWO) با الهام از زندگی اجتماعی گرگ‌های خاکستری [۱۴]، الگوریتم سینوس کسینوس (SCA) با تکیه بر یک مدل ریاضی [۱۵]، الگوریتم بهینه‌سازی خرچنگ (COA) مشابه رفتار خرچنگ‌های دریایی در استراحتگاه تابستانی [۱۶] و الگوریتم بهینه‌ساز طوطی (PO) با الهام از رفتار طوطی‌های آموزش دیده [۱۷]، از الگوریتم‌های مقایسه شده می‌باشند.

با بررسی تغییرات و اصلاحات صورت گرفته بر الگوریتم GSA، مشخص می‌شود تمامی این تغییرات به صورت عمومی بر GSA اعمال شده‌اند و هیچ کدام به صورت ویژه برای کاربرد این الگوریتم در مسئله زمان بندی محاسبات اصلاح نشده است. از یک دیدگاه دیگر، هیچ کدام از نسخه‌های بهبود یافته این الگوریتم به صورت پویا بر وضعیت نمونه جواب‌ها را در طول اجرای الگوریتم نظارت نمی‌کنند و برای گرفتاری زود هنگام و سریع در یک بهینه محلی آمادگی ندارند که در این مقاله با ارائه یک نسخه بهبود یافته از این الگوریتم، سعی بر رفع این ایراد شده است. خلاصه‌ای از روش‌های ذکر شده، برای مقایسه و درک بهتر در جدول (۱) آورده شده است.

یکی از الگوریتم‌های ترکیبی هستند که در آن معادلات ریاضی مربوط به شتاب برگرفته از الگوریتم جستجوی گرانشی و معادلات سرعت بر اساس الگوریتم ازدحام ذرات است [۲۰]. همچنین در مقاله [۲۱] یک الگوریتم ترکیبی با استفاده از الگوریتم‌های ژنتیک و جستجوی گرانشی (GA-GSA) معرفی شده که در آن با استفاده از عملگرهای انتخاب و جهش سعی بر ارتقا عملکرد الگوریتم GSA و استفاده از آن در مسائلی با فضای جستجوی پیچیده شده است. ابراهیمی و همکاران در مقاله [۲۲]، با استفاده از توابع مقیاس بندی بولتزمن و سیگما، روش محاسبه جرم را در الگوریتم GSA تغییر داده‌اند. نتایج اجرای این الگوریتم نشان دهنده بهبود عملکرد آن نسبت به الگوریتم GSA و برتری آن نسبت به الگوریتم‌های ژنتیک و ازدحام ذرات و چند الگوریتم دیگر می‌باشد.

جدول (۱): خلاصه روش‌های ارائه شده

محدودیت‌ها	نقاط قوت	اهداف	رویکرد اصلی	الگوریتم	مرجع
پیچیدگی محاسباتی بالا و زمان اجرای زیاد به دلیل مرتب سازی غیر مغلوب و مدیریت آرشیو، وابستگی شدید کیفیت جواب‌ها به تنظیم پارامترها و اندازه آرشیو و افزایش هزینه محاسباتی در مسائل با تعداد وظایف بسیار زیاد	کاهش چشمگیر هر دو هدف به طور هم زمان. کیفیت بالای جبهه پارتو در مقایسه با روش‌های دیگر	کاهش هم زمان تاخیر و انرژی مصرفی. بهبود کیفیت جبهه پارتو	استفاده از مفهوم غلبه پارتو و مجموعه جواب‌های غیر مغلوب به همراه آرشیو برای هدایت جمعیت و حفظ تنوع	MOGWO	[۱۸]
مستعد همگرایی زودرس و گیر افتادن در بهینه‌های محلی، به ویژه در مسائل پیچیده و با ابعاد بزرگ. کاهش تنوع جمعیت در مراحل پایانی جستجو که منجر به کاهش کیفیت جواب‌های نهایی می‌شود	سادگی ساختار و پیاده سازی آسان، تعداد پارامترهای کنترلی کم و سرعت همگرایی مناسب در مسائل ساده. یکی از پرکاربردترین روش‌ها در حوزه زمان بندی	حل مسائل زمان بندی محاسبات	تغییر در نمایش جواب‌ها و هماهنگی با تابع هدف	PSO	[۱۹]
پیچیدگی محاسباتی بیشتر نسبت به الگوریتم پایه و عملکرد آن‌ها همچنان به مقدار اولیه ضرایب ترکیب وابسته است	بهره‌مندی هم زمان از قابلیت کاوش PSO و بهره‌وری GSA. تعادل بهتر بین کاوش و بهره‌وری نسبت به نسخه‌های پایه	ارتقا قابلیت جستجو و بهره‌مندی از قابلیت‌های هر دو الگوریتم	ترکیب معادلات دو الگوریتم PSO و GSA	PSO-GSA	[۲۰]
افزایش زمان همگرایی و نیازمند تنظیم دقیق نرخ جهش و انتخاب	استفاده از عملگرهای تکاملی (جهش و انتخاب) باعث افزایش تنوع جمعیت و بهبود توانایی فرار از بهینه‌های محلی می‌شود. کارایی بهتر در فضاهای جستجوی پیچیده نسبت به GSA پایه	حل مسائل با فضای جستجو پیچیده و ابعاد بالا	ترکیب عملگرهای انتخاب و جهش از الگوریتم ژنتیک با ساختار الگوریتم جستجوی گرانشی	GA-GSA	[۲۱]
همچنان یک الگوریتم تک هدفه محسوب می‌شود و برای مسائل با اهداف چندگانه نیاز به تبدیل به نسخه چندهدفه یا استفاده از مکانیزم‌های ذخیره سازی جبهه پارتو دارد	برتری قطعی نسبت به GSA پایه، GA و PSO و چندین الگوریتم دیگر. بهبود دقت و سرعت همگرایی با تنظیم هوشمندانه تر جرم‌ها	بهبود عملکرد کلی الگوریتم	تغییر روش محاسبه جرم اجسام	GSA اصلاح شده	[۲۲]

جدول (۳): ویژگی‌های منابع پردازشی [۲۴]

نماد	تعریف
P_i	سرعت پردازشی S_i
δ_{tri}	مدت زمان انتقال یک بسته درخواست، از شبکه اینترنت اشیا تا منبع پردازشی S_i (با صرف نظر از ترافیک شبکه)

در ادامه، برای تشریح مدل سیستم، به بررسی روش محاسبه تاخیر درخواست‌ها، انرژی مصرفی منابع و محدودیت‌های مدل سیستم پرداخته شده [۲۳، ۲۴، ۵].

۳-۲ محاسبه تاخیر درخواست‌های پردازشی

هر درخواست پردازشی از زمان تولید در مراحل مختلف سه نوع تاخیر زیر را تجربه می‌کند:

- تاخیر انتقال (δ_{tr})، مدتی زمانی که تمام بسته‌های یک درخواست به لینک مورد نظر ارسال می‌شوند؛
- تاخیر صف انتظار (δ_q)، به معنی مدت زمانی که درخواست در صف برای عبور از سوئیچ‌ها و روترها سپری می‌کند، که مقدار آن غیر قابل پیش‌بینی و تصادفی است؛
- تاخیر انتشار، به معنی مدت زمانی که یک سیگنال قبل از رسیدن به مقصد، در محیط انتقال منتشر می‌شود و به دلیل بسیار کوچک‌تر بودن از دیگر انواع تاخیر، از محاسبه آن صرف نظر شده است.

می‌توان پارامترهای مورد نیاز برای محاسبه تاخیر را به صورت فرمول‌های آورده شده در جدول (۴) محاسبه کرد.

رابطه (۱) برای محاسبه تاخیر کلی یک بسته از درخواست برای انتقال از لبه به منبع محاسباتی به کار می‌رود، که با نماد δ نمایش داده می‌شود. اگر δ_i برابر با مدت زمانی باشد که یک درخواست در مسیر لبه تا منبع محاسباتی S_i قرار دارد، TQT_{ij} بیانگر کل تاخیر تحمیل شده به درخواست r_j برای انتقال از لبه به منبع محاسباتی S_i می‌باشد که به واسطه رابطه (۲) قابل برآورد است. مدت زمان پردازش درخواست r_j با منبع محاسباتی S_i ، معادل پارامتر PT_{ij} می‌باشد که به کمک رابطه (۳) قابل محاسبه است. در نهایت، کل تاخیر در پاسخگویی به درخواست r_j اگر توسط منبع محاسباتی

۳-۱ روش پیشنهادی

در ابتدای این بخش، به تعریف یک محیط شبیه‌سازی (مدل سیستم) شبکه اینترنت اشیا برای حل مسئله زمان‌بندی محاسبات، در شبکه اینترنت اشیا دارای دو مدل محاسباتی ابر و لبه، پرداخته خواهد شد. پس از آن به ارائه روش پیشنهادی برای حل مسئله مطرح شده، به کمک الگوریتم جستجوی گرانشی بهبودیافته پرداخته می‌شود.

۳-۱-۱ مدل سیستم

درخواست‌های پردازشی در این مدل سیستم می‌توانند تجزیه و تحلیل داده‌ها، پردازش تصویر، محاسبات ریاضی و از این قبیل باشند. در این محیط شبیه‌سازی شده شبکه اینترنت اشیا، درخواست‌های پردازشی در مجموعه R و به صورت $\{r_j | j = 1, \dots, n\}$ نمایش داده می‌شوند و هر درخواست پردازشی دارای ویژگی‌های جدول (۲) است [۲۳].

جدول (۲): ویژگی‌های درخواست‌های پردازشی [۲۳]

نماد	تعریف
IT_j	زمان شروع درخواست r_j
D_j	مهلت انجام درخواست r_j
Q_j	اندازه درخواست r_j بر حسب تعداد بسته
W_j	ضریب وزنی اولویت درخواست r_j

که در این بین، مجموع W_j درخواست‌های مختلف برابر با ۱، یعنی $\sum_{j=1}^n W_j = 1$ می‌باشد. لازم به ذکر است، برای اثبات برتری کارایی الگوریتم پیشنهادی در حل مسئله زمان‌بندی محاسبات، الگوریتم در شش محیط مختلف، با تعداد درخواست‌های (n) متفاوت، مورد آزمون قرار گرفته است. تعداد درخواست‌های پردازشی در این محیط‌ها از ۱۰۰۰ تا ۶۰۰۰ با گام‌های ۱۰۰۰ تایی می‌باشد.

در این مدل سیستم، منابع پردازشی که در لایه ابر یا لایه لبه قرار دارند، به صورت $S = \{S_i | i = 1, \dots, m\}$ نشان داده خواهند شد که هر کدام دارای خصوصیات جدول (۳) می‌باشند.

▪ مصرف انرژی محاسبات، به معنی انرژی مصرف شده حین انجام محاسبات؛

▪ مصرف انرژی در زمان آماده‌به‌کار، به معنی انرژی مصرف شده منبع پردازشی در بازه‌های زمانی که در حال انجام هیچ‌کدام از درخواست‌های پردازشی نمی‌باشد و آماده دریافت یک وظیفه جدید است.

هرکدام از حالات مصرف انرژی، طبق رابطه (۷) محاسبه می‌شود.

$$EC_i = \begin{cases} \beta_i = 10^{-8} \times P_i^2, & \text{در حالت پردازش} \\ \alpha_i = 0.6 \times \beta_i, & \text{در حالت بیکاری} \end{cases} \quad (7)$$

اگر ET_i برابر با مدت زمان انجام تمام درخواست‌های پردازشی محول شده به منبع محاسباتی S_i باشد، آنگاه می‌توان آن را بر اساس رابطه (۸) محاسبه کرد.

$$ET_i = \sum_{j=1}^n X_{ij} \times ETC_{ij}, \forall r_j \in R, \forall S_i \in S, \quad (8)$$

و بر همین اساس بیشترین زمان مورد نیاز منابع پردازشی، برای پردازش درخواست‌هایشان، طبق رابطه (۹) به دست می‌آید.

$$M = \max(ET_i), \forall S_i \in S. \quad (9)$$

با در نظر گرفتن روابط (۵، ۸، ۹)، انرژی مصرفی هر یک از منابع محاسباتی را می‌توان طبق فرمول (۱۰) محاسبه کرد.

$$E(S_i) = ([ET_i \times \beta_i] + [(M - ET_i) \times \alpha_i])P_i. \quad (10)$$

بر اساس انرژی مصرفی همه منابع، دومین هدف این بهینه‌سازی یعنی کمینه‌سازی انرژی مصرفی منابع محاسباتی را می‌توان به شکل رابطه (۱۱) به یک فرمول ریاضی تبدیل کرد.

$$F_{energy} = \min\left(\sum_{i=1}^m E(S_i)\right). \quad (11)$$

از آنجایی که هر کدام از مقادیر محاسبه شده توسط روابط (۶، ۱۱) دارای محدوده متفاوتی هستند، برای جمع و تبدیل آنها به یک مسئله تک‌هدفه نیاز به وزن‌دهی دارند. به دلیل آزمایش الگوریتم پیشنهادی در شش محیط با تعداد درخواست‌های متفاوت، این وزن‌دهی نیز باید برای هر محیط متغیر باشد که وزن هر کدام از اهداف برای هر محیط با توجه به تجربیات به دست آمده و به‌صورت دستی تنظیم شده‌اند، در جدول (۶) آورده شده است.

S_i پردازش شود، در رابطه (۴) قابل محاسبه است که در آن، ST_{ij} زمان شروع پردازش r_j در منبع S_i می‌باشد.

جدول (۴): روابط مربوط به تاخیر

شماره رابطه	رابطه
۱	$\bar{\delta} = \delta_{tr} + \bar{\delta}_q$
۲	$TQT_{ij} = \sum_{i=1}^{Q_j} \bar{\delta}_i$
۳	$PT_{ij} = Q_j / P_i, \quad \forall r_j \in R, \forall S_i \in S$
۴	$LT_{ij} = ST_{ij} + PT_{ij} + TQT_{ij} - IT_j$

اگر در این مدل سیستم پارامتر X_{ij} را متغیر تصمیم پردازش در نظر بگیریم، به‌صورت رابطه (۵) تعریف می‌شود.

$$X_{ij} = \begin{cases} 1, & \text{اگر درخواست } r_j \text{ در منبع } S_i \text{ پردازش شود} \\ 0, & \text{در این غیر صورت} \end{cases} \quad (5)$$

آنگاه تابع هدف کمینه‌سازی تاخیر زمانی را می‌توان طبق رابطه (۶) تعریف کرد.

$$F_{latency} = \min\left(\sum_{i=1}^m \sum_{j=1}^n LT_{ij} \times W_j \times X_{ij}\right). \quad (6)$$

محاسبه انرژی مصرفی منابع پردازشی

بر طبق تعاریف گفته شده، می‌توان جدول زمان مورد انتظار پردازش (ETC) را همانند جدول (۵) تشکیل داد که در آن درایه ETC_{ij} برابر با زمان مورد نیاز پردازش درخواست r_j توسط منبع محاسباتی S_i و معادل با PT_{ij} در رابطه (۳) می‌باشد و طبق همان رابطه قابل محاسبه است.

جدول (۵): زمان پردازش درخواست r_j توسط منبع S_i

ماتریس ETC	S_1	S_2	...	S_i	...	S_m
r_1	ETC_{11}	ETC_{21}	...	ETC_{i1}	...	ETC_{m1}
r_2	ETC_{12}	ETC_{22}	...	ETC_{i2}	...	ETC_{m2}
...	...	...	...	...	...	...
r_j	ETC_{1j}	ETC_{2j}	...	ETC_{ij}	...	ETC_{mj}
...	...	...	...	...	...	...
r_n	ETC_{1n}	ETC_{2n}	...	ETC_{in}	...	ETC_{mn}

انرژی مصرف شده توسط یک منبع پردازشی حاصل جمع دو نوع مصرف انرژی است که در زیر به آنها اشاره شده است:

انجام محاسبات رعایت شود و در حالت معکوس این پارامتر برابر با صفر و پارامتر θ_{ikj} برابر با یک خواهد شد. با این تعریف این محدودیت را می‌توان همانند رابطه (۱۵) به فرم ریاضی تعریف کرد.

$$X_{ij} + X_{ik} = 2 \quad \text{اگر } \forall r_{j,k} \in R, \forall s_i \in S \quad (15)$$

$$\theta_{ijk} + \theta_{ikj} = 1 \quad \text{آنگاه}$$

پس باید رابطه (۱۶) را در نظر گرفت.

$$ST_{ik} \geq \theta_{ijk}(ST_{ij} + PT_{ij}). \quad (16)$$

- زمان پایان پردازش وظیفه r_j باید همانند رابطه (۱۷)، قبل از اتمام مهلت زمانی وظیفه (D_j) باشد.

$$\sum_{i=1}^m X_{ij} = 1, \forall r_j \in R. \quad (17)$$

- اندازه بسته‌های درخواست باید یکسان باشند.
- درخواست‌ها در هر لحظه از زمان به وجود می‌آیند.

۳-۳ الگوریتم جستجوی گرانشی بهبودیافته

به بهترین نمونه جواب یافته شده (از نظر شایستگی در تابع هدف)، تا هر تکرار از اجرای الگوریتم‌های فراابتکاری، بهترین جواب تاکنون (BSF) گفته می‌شود. علاوه بر مقاله [۸] و دلایل اشاره شده در بخش مقدمه مبنی بر همگرایی زود هنگام نمونه جواب‌های الگوریتم GSA به سمت یک بهینه محلی، با نظارت بر پارامتر BSF، در طول اجرای الگوریتم GSA، می‌توان به این مهم پی برد و همگرایی زود هنگام نمونه جواب‌ها در روند این الگوریتم را به طور عینی مشاهده کرد. همگرایی زود هنگام نمونه جواب‌ها، ناشی از عدم وجود تعادل مناسب بین دو قابلیت کاوش و بهره‌وری در این الگوریتم فراابتکاری است. در ادامه، به ارائه یک راهکار مناسب جهت گذر از این چالش پرداخته شده است.

بازمقداردهی نمونه جواب‌ها

یکی از دو راهکاری که برای حل این عیب از الگوریتم GSA در نظر گرفته شده، بازمقداردهی نمونه جواب‌های الگوریتم با استفاده از سیستم استنتاج فازی در زمان‌های خاصی از اجرای الگوریتم است. لازم به ذکر است نمونه جواب‌های اولیه همانند الگوریتم GSA، به صورت تصادفی با اختصاص هر وظیفه به یکی از منابع محاسباتی موجود در لبه یا ابر، با استفاده از توزیع یکنواخت

جدول (۶): وزن هر هدف در محیط‌های شبیه‌سازی شده مختلف

	تعداد درخواست‌ها					
	۱۰۰۰	۲۰۰۰	۳۰۰۰	۴۰۰۰	۵۰۰۰	۶۰۰۰
$W_{latency}$	10^{-3}	10^{-3}	10^{-3}	10^{-4}	10^{-4}	10^{-4}
W_{energy}	10^{-4}	10^{-5}	10^{-5}	10^{-5}	10^{-5}	10^{-5}

طبق روابط (۶، ۱۱) که تا اینجا گفته شد و همچنین با توجه به جدول (۶)، می‌توان زمان‌بندی محاسبات در شبکه اینترنت اشیا را یک مسئله بهینه‌سازی تک هدفه با تابع هدف ذکر شده در رابطه (۱۲) دانست.

$$Fitness = [F_{latency} \times W_{latency} + F_{energy} \times W_{energy}] \quad (12)$$

$$s.t \sum_{i=1}^m X_{ij} = 1, \forall r_j \in R.$$

در این مدل سیستم هر نمونه جواب، یک بردار می‌باشد. عضو λ_m از بردار نشان دهنده منبع محاسباتی است که در آن وظیفه محاسباتی λ_m پردازش می‌شود. به این صورت، هر نمونه جواب یک بردار به طول تعداد وظایف محاسباتی است که هر عضو آن یک عدد طبیعی با بیشینه مقدار برابر با مجموع تعداد منابع محاسباتی موجود در ابر و لبه است.

محدودیت‌های مدل سیستم

در محیط اینترنت اشیا، محدودیت‌هایی برای واگذاری وظایف محاسباتی به منابع در اختیار وجود دارد که در زیر به آنها اشاره شده است:

- هر درخواست پردازشی می‌تواند فقط در یک منبع محاسباتی پردازش شود (رابطه (۱۳)).

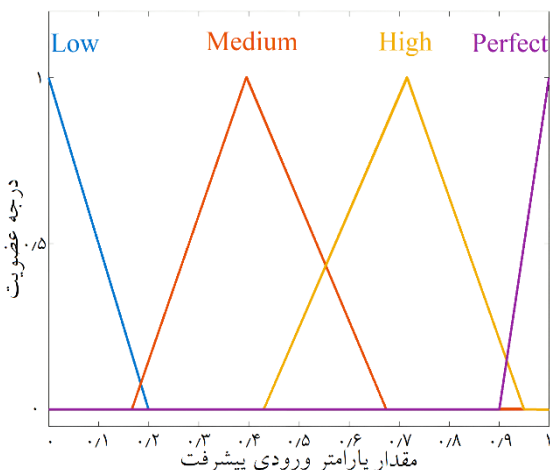
$$\sum_{i=1}^m X_{ij} = 1, \forall r_j \in R. \quad (13)$$

- همانند رابطه (۱۴)، درخواست نمی‌تواند قبل از رسیدن به منبع پردازش شود.

$$ST_{ij} \geq IT_j + TQT_{ij}, \forall r_j, r_k \in R, \forall s_i \in S. \quad (14)$$

- هر منبع در یک زمان واحد قادر به انجام یک درخواست پردازشی است. اگر پارامتر θ_{ijk} بیانگر تقدم درخواست r_j نسبت به درخواست r_k در پردازش توسط منبع s_i باشد، زمانی این پارامتر برابر با ۱ قرار می‌گیرد که این اولویت در

در نظر گرفت. با این حال بسته به شرایط مسئله، مقدار دقیق و بهینه آن متغیر خواهد بود.



شکل (۲): تابع عضویت پیشرفت در سیستم استنتاج فازی

سیستم استنتاج فازی تعریف شده با استفاده از قوانین جدول (۷) که تمامی حالات ورودی این سیستم را پوشش می‌دهد، به تولید خروجی فازی، که نسبت بازمقداردهی (*reinitial*) نام دارد، می‌پردازد.

جدول (۷): قوانین حاکم بر سیستم استنتاج فازی

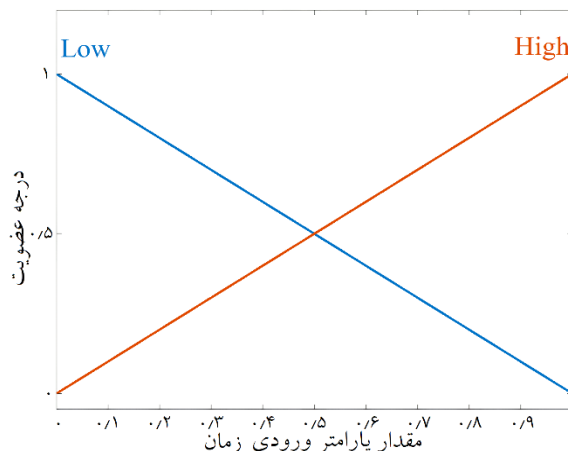
شماره	ورودی		خروجی	وزن
	time	improve	reinitial	
۱	Low	Low	High	۱
۲	Low	Medium	High	۱
۳	Low	High	Low	۱
۴	Low	Perfect	Zero	۱
۵	High	Low	High	۱
۶	High	Medium	Low	۱
۷	High	High	Zero	۱
۸	High	Perfect	Zero	۱

پس از آن، این سیستم با استفاده از مجموعه‌های فازی تعریف شده که در شکل (۳) نمودار آن را مشاهده می‌کنید، خروجی تولید شده را فازی‌زدایی و به یک مؤلفه بین صفر و ۱ تبدیل می‌کند. در بین مجموعه‌های فازی تعریف شده برای فازی‌زدایی خروجی تولید شده، مجموعه *Zero* وجود دارد تا بتوان مقادیر بسیار نزدیک به صفر را با عدد صفر جایگزین کرد.

گسسته تولید می‌شود. برای پیاده‌سازی این سیستم دو پارامتر زمان (*time*) و پیشرفت (*improve*) به عنوان ورودی‌های سیستم استنتاج فازی در نظر گرفته شده است. پارامتر ورودی زمان که طبق رابطه (۱۸) محاسبه می‌شود، به معنای میزان پیشروی تکرارهای اجرای الگوریتم می‌باشد.

$$time(t) = \frac{iteration(t)}{maximum\ iterations} \quad (18)$$

این پارامتر طبق نمودار تابع عضویت شکل (۱) که برای آن تعریف شده است، به عضویت مجموعه‌های فازی در می‌آید.



شکل (۱): تابع عضویت زمان در سیستم استنتاج فازی

پارامتر ورودی پیشرفت بیانگر میزان بهبود BSF می‌باشد. بهبود یک نمونه جواب از طریق رابطه (۱۹) محاسبه می‌گردد [۲۵].

$$IMP(t) = fit(t)/fit(t - gap), \quad (19)$$

و بر این اساس می‌توان رابطه (۲۰) را برای پیشرفت بهترین جواب تاکنون تعریف کرد.

$$IMP_{BSF}(t) = BSF(t)/BSF(t - gap), \quad (20)$$

برای متناظرسازی میزان پیشرفت با مقدار عددی پارامتر پیشرفت، می‌توان آن را طبق رابطه (۲۱) محاسبه کرد.

$$improve(t) = 1 - \frac{BSF(t)}{BSF(t - gap)}, \quad (21)$$

که حاصل آن به عنوان ورودی دوم سیستم استنتاج فازی به کار می‌رود. این سیستم مقدار این پارامتر را طبق نمودار تصویر (۲)، به عضویت در مجموعه‌های فازی تعریف شده، تبدیل می‌کند. یکی از مقادیر مناسب و متعادل پارامتر *gap* برای شرایط مختلف را می‌توان یک دهم از بیشترین تکرار (*maximum iterations*)

پس از محاسبه این پارامتر برای تمامی نمونه جواب‌ها، لیست مرتب شده آنها (SED)، تشکیل می‌شود.

برای محاسبه تعداد مناسب نمونه جواب‌های قابل حذف و تبدیل خروجی سیستم استنتاج فازی به تعداد قابل حذف، از فرمول (۲۳) استفاده می‌شود.

$$delete\ number = [reinitial \times N/3], \quad (23)$$

که در آن N بیانگر تعداد نمونه جواب‌هاست. در ادامه به‌ازای هر $delete\ number$ دو نمونه جواب از بالای لیست SED حذف و با مقداری نزدیک به نمونه جواب انتهایی لیست بازمقداردهی خواهند شد و در ماتریس اصلی نمونه جواب‌ها جایگزین می‌شوند. این بازمقداردهی به‌صورت رابطه (۲۴) انجام می‌شود.

$$\begin{aligned} \forall d = 0, 1, \dots, delete\ number, \\ SED_{\lfloor \frac{2N}{3} \rfloor - (2d)} = SED_{N-d} \times 1.01 \\ SED_{\lfloor \frac{2N}{3} \rfloor - (2d-1)} = SED_{N-d} \times 0.99 \end{aligned} \quad (24)$$

مقیاس‌بندی مجموع

همان‌طور که گفته شد، دو راهکار به‌صورت هم‌زمان و موازی برای بهبود الگوریتم GSA پیشنهاد شده است. راهکار اول بازمقداردهی نمونه جواب‌ها و راهکار دوم، استفاده از روش مقیاس‌بندی مجموع، برای اصلاح روش محاسبه جرم در این الگوریتم است. در این روش، از رابطه (۲۵) برای محاسبه جرم هر نمونه جواب در هر تکرار استفاده می‌شود [۲۶].

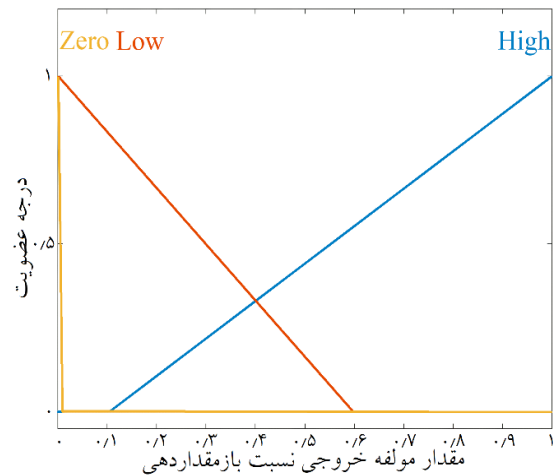
$$M_i(t) = \begin{cases} a + \frac{NFit_i(t) - \langle NFit \rangle_t}{b\sigma(t)}, & \sigma(t) \neq 0, \\ 1, & \sigma(t) = 0, \end{cases} \quad (25)$$

که ثابت‌های a و b به ترتیب برابر با ۱ و ۲ و تعریف متغیرها در آن به شرح جدول (۸) می‌باشد.

جدول (۸): تعاریف متغیرها در رابطه (۲۷)

نماد	تعریف
$M_i(t)$	جرم نمونه جواب i ام در تکرار t
$NFit_i$	جرم نرمال شده نمونه جواب i ام
$\langle NFit \rangle_t$	میانگین جرم تمام نمونه جواب‌ها در تکرار t
$\sigma(t)$	انحراف معیار جرم نمونه جواب‌ها در تکرار t

مراحل الگوریتم پیشنهادی در الگوریتم (۱) و دیاگرام بلوکی مربوطه در شکل (۴) قابل مشاهده است.



شکل (۳): تابع عضویت نسبت بازمقداردهی در سیستم استنتاج فازی

در این مطالعه سعی شده با حذف نمونه جواب‌هایی که در مناطق پرتراکم قرار دارند و بازمقداردهی آنها با مقداری نزدیک به نمونه جواب‌ها در مناطق کم‌تراکم، از گرفتاری الگوریتم در بهینه‌های محلی جلوگیری شود. تعداد نمونه جواب‌هایی که بازمقداردهی خواهند شد، با استفاده از سیستم استنتاج فازی شرح داده شده، تعیین می‌شود. از آنجایی که نمونه جواب‌ها در تکرارهای مختلف، اطلاعات مهمی از فضای جواب‌های شدنی در بر دارند، به روش آزمون و خطا مشخص شده و در این پژوهش توصیه می‌شود، سه برابر gap از تکرارهای ابتدایی و ۲۰ درصد از تکرارهای پایانی از اجرای الگوریتم بدون دخالت این روش انجام گیرد تا الگوریتم جستجوی اولیه و همگرایی پایانی را طبق روال عادی خود طی کند. همچنین بهتر است، این روش هر gap تکرار یکبار اجرا شود تا الگوریتم از روند عادی خود خیلی دور نشود.

برای تشخیص نمونه جواب‌های مناطق پرتراکم، از لیست فاصله اقلیدسی نمونه جواب‌ها تا دومین نمونه جواب نزدیک به آنها استفاده شده است. در این لیست فاصله‌ها که لیست مرتب شده بر مبنای فاصله اقلیدسی (SED) نام دارد، همه نمونه جواب‌ها به ترتیب از کمترین به بیشترین فاصله تا دومین نمونه جواب نزدیک، قرار دارند. این فاصله طبق رابطه (۲۲) محاسبه می‌شود.

$$d(x, y) = \sqrt{\sum_{i=1}^d (x_i - y_i)^2}, \quad (22)$$

پیچیدگی زمانی

پیچیدگی زمانی الگوریتم GSA برابر با $O(T \times N^2 \times D)$ می‌باشد، که در آن متغیر T برابر با تعداد بیشترین تکرار حلقه اصلی الگوریتم، N معادل تعداد نمونه جواب‌ها و D برابر با ابعاد مسئله است. بیشترین پیچیدگی در دو راهکار این بهبود، مربوط به محاسبه مقادیر لیست فاصله اقلیدسی نمونه‌جواب‌ها تا دومین نمونه جواب نزدیک به آنها می‌باشد. پیچیدگی زمانی بخش مربوطه، $O(N^2)$ است که در حلقه اصلی الگوریتم پیشنهادی قرار دارد. به دلیل برابری پیچیدگی زمانی حلقه اصلی الگوریتم با $O(N^2)$ ، پیچیدگی زمانی کل الگوریتم پیشنهادی نیز افزایش پیدا نمی‌کند و پیچیدگی الگوریتم IGSA نیز $O(T \times N^2 \times D)$ باقی می‌ماند. روش پیشنهادی بدون افزایش پیچیدگی زمانی، از همگرایی زود هنگام نمونه جواب‌ها در بهینه‌های محلی جلوگیری می‌کند.

الگوریتم (۱): الگوریتم جستجو گرانشی بهبودیافته

ورودی: تابع هدف.

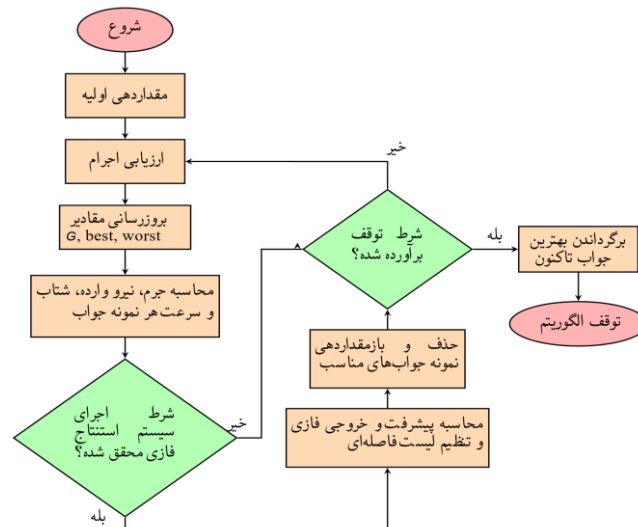
خروجی: بهترین جواب.

(۱) تعیین محیط سیستم و مقداردهی اولیه.

(۲) جایابی اولیه اجرام.

(۳) ارزیابی اجرام.

- (۴) به روز رسانی مقادیر G و $best, worst$.
- (۵) محاسبه جرم هر نمونه جواب به کمک روش مقیاس‌بندی مجموع.
- (۶) محاسبه نیرو وارد بر هر جرم.
- (۷) محاسبه شتاب و سرعت هر جرم.
- (۸) به روز رسانی موقعیت اجرام.
- (۹) اگر زمان اجرای سیستم استنتاج فازی فرا رسیده آنگاه
- (۱۰) محاسبه پیشرفت بهترین جواب تاکنون.
- (۱۱) محاسبه خروجی سیستم استنتاج فازی.
- (۱۲) تنظیم لیست SED .
- (۱۳) حذف و جایگزینی نمونه جواب‌های مناسب.
- (۱۴) پایان اگر
- (۱۵) اگر شرط توقف برآورده نشده آنگاه
- (۱۶) بازگشت به مرحله ۳.
- (۱۷) پایان اگر
- (۱۸) در غیر این صورت
- (۱۹) توقف الگوریتم و بهترین جواب تاکنون را برگردان.
- (۲۰) پایان اگر



شکل (۴): دیاگرام بلوکی الگوریتم IGSA پیشنهادی

خطوط ۹ تا ۱۳ از این الگوریتم، بیانگر مراحل اضافه شده به الگوریتم پایه، در جهت پیاده‌سازی روش بازمقداردهی نمونه جواب‌ها با استفاده از سیستم استنتاج فازی (راهکار اول) می‌باشد.

مرحله اصلاح شده از الگوریتم GSA، مربوط به محاسبه جرم نمونه جواب‌ها با استفاده از روش مقیاس‌بندی مجموع (راهکار دوم)، در خط ۵ از الگوریتم (۱) قرار دارد. همچنین شرط آورده شده در

صفر (برابری عملکرد الگوریتم‌ها) رد خواهد شد. این آزمون بدون توجه به تعداد بردهای الگوریتم، بر اساس میانگین رتبه در تمامی مسائل عمل می‌کند و به همین دلیل ممکن است الگوریتمی که در چند مسئله بهترین عملکرد را داشته است، در رتبه‌بندی کلی رتبه‌ای کمتر از الگوریتمی که در تمام مسائل عملکرد نسبتاً خوبی داشته، دریافت کند. این آزمون آماری، علاوه بر معرفی الگوریتم برتر از نظر عملکرد، پارامتری به عنوان مقدار p (p -value) محاسبه می‌کند که در صورتی که از سطح معناداری (که معمولاً برابر با ۰/۰۵ قرار می‌گیرد) کمتر باشد، فرض صفر رد خواهد شد که به معنی چشمگیر بودن اختلاف بین نتایج الگوریتم‌های مختلف و قابل اتکا بودن نتایج آزمایش است. ولی در صورتی که مقدار پارامتر p -value بیشتر از سطح معناداری باشد، فرض صفر قبول خواهد شد، اختلاف بین نتایج الگوریتم‌های مختلف جزئی بوده و نتایج رتبه‌بندی آزمون آماری قابل اتکا نخواهد بود. الگوریتم برتر با رتبه‌ای کمتر از دیگر الگوریتم‌ها مشخص خواهد شد. این رتبه، میانگین رتبه‌های الگوریتم در شرایط مختلفی است، که برای آزمون فریدمن ارائه شده است.

۴-۱-۴ توابع آزمایشی استاندارد

در بسیاری از مقالات معرفی الگوریتم‌های فراابتکاری، از توابع معرفی شده در پژوهش [۲۸] برای آزمایش عملکرد الگوریتم استفاده شده است. همچنین در مقاله [۲۵]، توابع هدف معرفی شده در پژوهش [۲۸] را دسته‌بندی کرده است. در این دسته‌بندی، بخشی از توابع در دسته‌بندی توابع چندوجهی و پیچیده قرار گرفته‌اند و این خصوصیت، این توابع هدف را شبیه به مسئله اصلی مطالعه حاضر، یعنی مسئله زمان‌بندی محاسبات، کرده است. به همین دلیل در مطالعه جاری، ابتدا از توابع چندوجهی آزمون آزمایشی استاندارد برای بررسی و مقایسه کارایی عملکرد الگوریتم IGSA استفاده شده است.

بخشی از مشخصات کلی توابع از جمله بازه جواب‌های امکان‌پذیر (Bounds)، ابعاد مسئله (Dim) و شایستگی بهترین جواب (F_{min})، در جدول (۱۰) ذکر شده است. هر کدام از الگوریتم‌های اجرا شده، ثابت‌هایی دارند که قابل تغییر و اصلاح است. برای پیاده‌سازی، این ثابت‌ها بر اساس مقاله اصلی آنها تعریف و

در دیاگرام بلوکی آورده شده در شکل (۴)، مراحل اصلاح شده یا اضافه شده به الگوریتم GSA برای بهبود و تبدیل آن به الگوریتم پیشنهادی، با حاشیه‌ای متفاوت مشخص شده است. این تفاوت، درک پیاده‌سازی انجام شده جهت بهبود و اختلاف بین دو الگوریتم GSA و IGSA را آسان‌تر خواهد کرد.

۴- نتایج اجرا و تحلیل

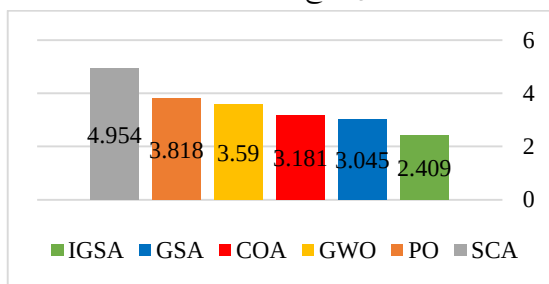
در این بخش، ابتدا عملکرد الگوریتم IGSA برای حل توابع آزمایشی استاندارد چندوجهی با عملکرد چند الگوریتم فراابتکاری دیگر و الگوریتم GSA مقایسه می‌شود. پس از مقایسه عملکرد در حل توابع آزمایشی استاندارد، این الگوریتم‌ها برای حل مسئله زمان‌بندی محاسبات در محیط شبیه‌ساز شبکه اینترنت اشیا ارائه شده در بخش قبل به کار گرفته می‌شوند و کارایی آنها با یکدیگر مقایسه خواهد گشت. برای مقایسه بهتر کارایی الگوریتم پیشنهادی، از نتایج اجرای الگوریتم‌های GSA، GWO، SCA، COA، PO و PSO استفاده شده است.

سخت‌افزار مجری این شبیه‌سازی که به زبان برنامه‌نویسی متلب (MATLAB) انجام شده است، دارای پردازنده AMD Ryzen 9 3900X 12-Core، حافظه رم ۱۲۸ گیگابایتی و گرافیک NVIDIA GeForce GT 610 می‌باشد. در این پیاده‌سازی، به‌منظور ارزیابی پایداری الگوریتم در شرایط مختلف، نمونه‌های مسئله در هر اجرای مستقل با استفاده از مولد اعداد تصادفی تولید شده‌اند. تمامی الگوریتم‌های اجرا شده در هر اجرا بر روی یک نمونه مسئله یکسان اجرا شده تا شرایط مقایسه به طور کامل منصفانه باشد. نتایج نهایی بر اساس میانگین و انحراف معیار حاصل از چند اجرای مستقل گزارش شده است.

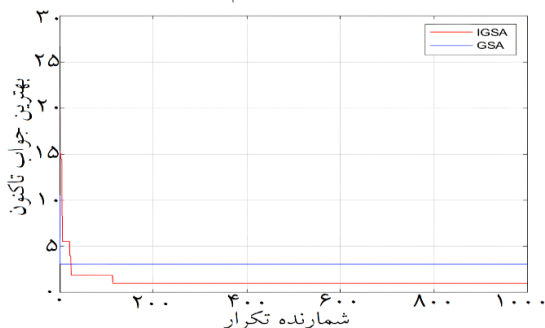
برای مقایسه و درک بهتر عملکرد الگوریتم‌های مختلف و کشف الگوریتم برتر، از آزمون آماری فریدمن بر نتایج حاصل از اجرا در هر دو مسئله استفاده شده است. این آزمون آماری ناپارامتری، برای مقایسه چندین الگوریتم روی چند مسئله، ابتدا در هر مسئله به هر الگوریتم یک رتبه اختصاص می‌دهد (بهترین رتبه ۱ و به ترتیب تا آخرین الگوریتم)، سپس میانگین رتبه هر الگوریتم در همه مسائل محاسبه شده و به رتبه کلی الگوریتم تعمیم داده می‌شود [۲۷] و اگر تفاوت میان میانگین رتبه‌ها از نظر آماری معنادار باشد، فرض

GSA+fuzzy و الگوریتم پیشنهادی بدون اصلاح روش محاسبه جرم (تحت عنوان GSA+mass) در کنار دیگر الگوریتم‌ها آورده شده است.

آزمون آماری فریدمن، بر نتایج حاصل از اجرای الگوریتم‌ها برای حل توابع آزمایشی استاندارد (جدول ۱۰)) انجام شده است و میانگین رتبه هر الگوریتم را می‌توان در ستون مربوطه در نمودار شکل (۵) مشاهده کرد. الگوریتم پیشنهادی این مطالعه با میانگین رتبه ۲/۴۰۹ کارایی برتر را به خود اختصاص داده و همچنین مقدار $p - value$ این آزمون آماری برابر با ۰/۰۳۰۸ شده است که کمتر از سطح معناداری (۰/۰۵) بوده و منجر به رد فرض صفر شده است. این آزمون آماری، تاییدی بر برتری الگوریتم IGSA نسبت به الگوریتم GSA و دیگر الگوریتم‌های اجرا شده (SCA, COA, GWO و PO) در حل توابع آزمایشی استاندارد می‌باشد.



شکل (۵): نمودار رتبه‌بندی فریدمن بر نتایج توابع آزمایشی استاندارد همچنین مقایسه نتایج حاصل از مطالعه قسمت به قسمت الگوریتم پیشنهادی (ablation) و الگوریتم پیشنهادی نشان می‌دهد که هر کدام از راهکارهای بهبود به صورت جداگانه GSA پایه را بهبود می‌دهد، اما بهترین عملکرد زمانی به طور مداوم و در تمام توابع میسر است که هر دو راهکار ادغام شده و به صورت مکمل یکدیگر به بهبود کاوش و بهره‌وری الگوریتم کمک کنند.



شکل (۶): نمودار روند بهبود BSF در طول اجرا بر روی تابع Shekel's Foxholes Function

مقداردهی شده است که مقادیر آنها در جدول (۹) قابل مشاهده است.

جدول (۹): مقادیر پارامترهای کلیدی الگوریتم‌های اجرا شده

[۶، ۱۳، ۱۴، ۱۵، ۱۶ و ۱۸]

الگوریتم	متغیر	مقدار	تعریف
IGSA	gap	maxits/10	فاصله بررسی پیشرفت
	α	24	ضریب کاهش همگرایی
	G_0	80	ثابت گرانشی اولیه
GSA	α	20	ضریب کاهش همگرایی
	G_0	100	ثابت گرانشی اولیه
GWO	α	[0, 2]	متغیر کنترل کاوش
SCA	α	2	متغیر کنترل گام
COA	temp	25	دما محیط
PSO	w	0.9	وزن اینرسی
	wdamp	0.99	ضریب کاهش وزن اینرسی
	c1	2	ضریب یادگیری فردی
	c2	2	ضریب یادگیری اجتماعی
	VelMax	$3/(up - low)$	ضریب یادگیری اجتماعی
	VelMin	$-3/(up - low)$	ضریب یادگیری اجتماعی
PO	alpha	Rand()/5	ضریب کنترل حرکت
	sita	Rand() * pi	رفتار ترس از غریبه
	st	Randi([1,4])	انتخاب رفتار

اجرای الگوریتم‌های مختلف، در محیطی با شرایطی یکسان انجام شده است. در این محیط، همه الگوریتم‌ها با تعداد نمونه جواب‌های ۱۰۰ عدد مجاز به انجام ۱۰۰۰ تکرار بوده‌اند و پس از آن متوقف و بهترین جواب یافته شده، برگردانده می‌شود. برای اطمینان از تصادفی نبودن جواب کشف شده توسط الگوریتم‌های مختلف، هر الگوریتم برای یافتن بهینه سراسری هر تابع هدف، ۳۰ بار اجرا شده است. میانگین شایستگی جواب نهایی الگوریتم‌ها (Ave) در ۳۰ بار اجرا و انحراف معیار آن (Std)، در جدول (۱۰) قابل مشاهده است. همچنین جهت مقایسه تاثیر هر راهکار به صورت تفکیکی (مطالعه ablation)، نتایج اجرای الگوریتم پیشنهادی بدون بازمقداردهی نتایج با استفاده از منطق فازی (تحت عنوان

توانسته به این مهم دست پیدا کند و در صورت همگرایی سریع، از آن جلوگیری کند. برای نمونه، BSF الگوریتم‌های مذکور برای تابع Shekel's Foxholes Function، در شکل (۶) آورده شده است.

همان‌طور که گفته شد، یکی از اهداف اصلی از این بهبود، کنترل دو قابلیت کاوش و بهره‌وری الگوریتم GSA در صورت همگرایی سریع می‌باشد. با بررسی BSF الگوریتم‌های GSA و IGSA در طول اجرای الگوریتم‌ها، مشخص می‌شود که الگوریتم پیشنهادی

جدول (۱۰): نتایج حل توابع آزمایشی استاندارد

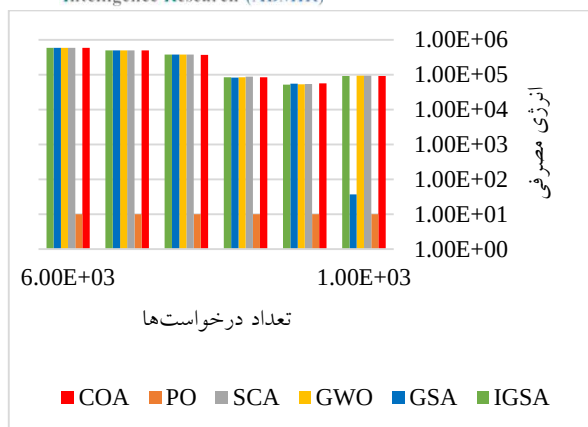
F-min	Dim	Bounds	GSA+fuzzy	GSA+mass	PO	COA	SCA	GWO	GSA	IGSA	پارامتر	نام تابع
0	10	[-50 50]	1.08 e-22	6.71 e-19	6.45 e-6	5.11 e-10	4.36 e-2	6.75 e-4	8.09 e-21	1.39 e-20	Ave	Generalized Penalized Function 1
			3.8 e-23	2.19 e-19	5.48 e-6	4.97 e-10	2.08 e-2	3.59 e-3	3.22 e-21	5.42 e-21	Std	
0	10	[-50 50]	3.31 e-11	2.01 e-9	5.5 e-5	1.41 e-2	1.95 e-1	4.9 e-7	3.9 e-20	7.91 e-20	Ave	Generalized Penalized Function 2
			5.61 e-12	3.43 e-10	1.16 e-4	3.46 e-2	9.17 e-2	1.84 e-7	1.28 e-20	2.77 e-20	Std	
1	2	[-65.53 65.53]	3.54 e-25	2.98 e-21	1.59	2.04	1.26	1.26	3.24	1.09	Ave	Shekel's Foxholes Function
			3.22 e-25	4.15 e-21	9.24 e-1	2.97	6.85 e-1	6.85 e-1	1.91	1.65 e-1	Std	
3.07 e-4	4	[-5 5]	2.66 e-12	1.97 e-10	4.21 e-4	4.47 e-4	8.76 e-4	2.31 e-3	1.7 e-3	1.69 e-3	Ave	Kowalik's Function
			7.35 e-13	4.24 e-11	2.75 e-4	2.72 e-4	4.23 e-4	6.11 e-3	6.11 e-4	6.11 e-3	Std	
-1.03	2	[-5 5]	8.38 e-3	5.38 e-3	-1.03	-1.03	-1.03	-1.03	-1.03	-1.03	Ave	Six-Hump Camel-Back Function
			7.36 e-3	5.35 e-3	4.96 e-9	5.68 e-16	7.23 e-6	1.23 e-9	6.51 e-16	2.7 e-12	Std	
3.98 e-1	2	[-5 0] [10 15]	0	0	3.97 e-1	3.97 e-1	3.97 e-1	3.97 e-1	3.97 e-1	3.97 e-1	Ave	Branin Function
			0	0	2.54 e-8	1.76 e-10	4.75 e-4	2.14 e-7	0	1.1 e-12	Std	
3	2	[-2 2]	2.87 e-4	3.87 e-4	3	3	3	3	3	3	Ave	Goldstein-Price Function
			2.33 e-4	4.31 e-4	2.55 e-7	6.76 e-15	7.25 e-6	1.1 e-6	1.97 e-15	1.51 e-10	Std	
-3.86	3	[0 1]	-2.5	-2.52	-3.86	-3.86	-3.85	-3.86	-3.86	-3.86	Ave	Hartman 3-D Function
			6.65 e-3	1.2 e-3	6.87 e-5	2.32 e-15	2.64 e-3	1.99 e-3	2.66 e-15	2.21 e-8	Std	
-3.32	6	[0 1]	1.92	1.89	-3.24	-3.26	-3.05	-3.25	-3.32	-3.32	Ave	Hartman 6-D Function
			9.83 e-1	9.62 e-1	7.52 e-2	6.37 e-2	7.18 e-2	7.1 e-2	1.34 e-15	1.62 e-10	Std	
-10	4	[0 10]	8.37 e-2	3.29 e-3	-6.58	-8.11	-4.54	-9.81	-5.95	-7.04	Ave	Shekel's Family Function (m=5)
			5.68 e-2	4.61 e-3	2.37	2.53	1.61	1.28	3.53	2.91	Std	
-10	4	[0 10]	2.46 e-2	2.46 e-2	-7.21	-8.63	-5.51	-1 e+1	-1.04 e+1	-1.04 e+1	Ave	Shekel's Family Function (m=7)
			3.69 e-10	0	2.64	2.54	1.76	1.34	0	4.49 e-10	Std	
-10	4	[0 10]	3.13	3.22	-6.93	-8.37	-5.42	-1.05 e+1	-1.05 e+1	-1.05 e+1	Ave	Shekel's Family Function (m=10)
			1.14 e-1	2.65 e-1	2.59	2.69	1.43	1.08 e-4	1.39 e-15	4.24 e-10	Std	

الگوریتم‌ها، برای حل مسئله زمان‌بندی محاسبات در مدل سیستم پرداخته شده است.

همان‌طور که پیش‌تر هم گفته شد، معماری این محیط شبیه‌سازی دارای دو لایه محاسباتی ابر و لبه است که هر کدام خصوصیات

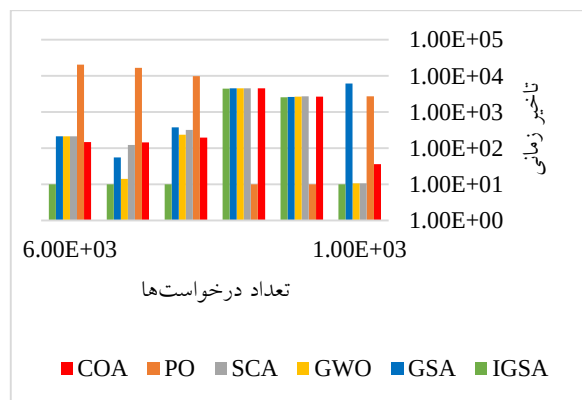
۴-۲ زمان‌بندی محاسبات در مدل سیستم

در این بخش ابتدا مشخصات و تنظیمات مدل سیستم اینترنت اشیا بیان خواهد شد. سپس به ارائه و بررسی نتایج حاصل از اجرای



شکل (۷): نمودار میله‌ای انرژی مصرفی انجام محاسبات

در شکل (۷)، نمودار میله‌ای لگاریتم تفاضل انرژی مصرفی جواب‌ها، از ده واحد کمتر از کمترین جواب و در شکل (۸)، نمودار میله‌ای لگاریتم تفاضل تأخیر زمانی جواب‌ها، از ده واحد کمتر از کمترین جواب، قرار گرفته است.



شکل (۸): نمودار میله‌ای تأخیر زمانی انجام محاسبات

همان‌طور که از نتایج شکل‌های (۷ و ۸) مشخص است، الگوریتم پیشنهادی توانسته در عین کنترل انرژی مصرفی منابع محاسباتی در مدل سیستم، تأخیر زمانی را در حالت کمینه نگه دارد. مشخصاً الگوریتم پیشنهادی جواب‌های بهتری نسبت به دیگر الگوریتم‌های مقایسه شده یافته است. مجدداً نتایج این اجرا نیز مورد آزمون آماری فریدمن قرار گرفته‌اند، تا بهبود عملکرد الگوریتم پیشنهادی نسبت به الگوریتم جستجو گرانشی پایه مشهودتر و تحلیل واضح‌تر باشد. نتیجه آزمون آماری فریدمن برای حل مسئله زمان‌بندی محاسبات، در نمودار شکل (۹) قرار گرفته است.

خاص خود را دارند. وظایف پردازشی در شبکه اینترنت اشیا تولید و برای تقسیم‌بندی بین منابع پردازشی به الگوریتم بهینه‌ساز ارسال می‌شوند. در ابتدا تمامی منابع پردازشی آزاد هستند و در ادامه برخی از منابع درگیر پردازش می‌شوند. الگوریتم بهینه‌ساز به دنبال تقسیم وظایف بین منابع پردازشی به نحوی است که انرژی مصرفی منابع و تأخیر زمانی تحمیل شده به وظایف در حالت کمینه قرار گیرند. در این مدل سیستم از ۵ منبع محاسباتی در لایه ابر و ۱۵ منبع محاسباتی در لایه لبه استفاده شده است که سرعت پردازش منابع موجود در هر لایه، به صورت اتفاقی و به ترتیب در بازه‌های ۹۰۰ تا ۱۰۰۰ و ۴۲ تا ۴۷ می‌باشد. تأخیر انتقال وظایف پردازشی به منابع محاسباتی نیز عددی تصادفی در بازه ۱ تا ۱۰ هزارم ثانیه می‌باشد. این آزمایش، در شش محیط مختلف با تعداد درخواست‌های پردازشی ۱۰۰۰، ۲۰۰۰، ۳۰۰۰، ۴۰۰۰، ۵۰۰۰ و ۶۰۰۰ انجام می‌شود، که این موضوع باعث اثبات کارایی الگوریتم برتر در محیط مختلف خواهد شد. اندازه درخواست‌های پردازشی در این مدل سیستم یک عدد تصادفی با توزیع یکنواخت بین ۵۰۰ تا ۵۵۰۰ می‌باشد. مهلت زمانی انجام هر درخواست نیز یک عدد است که از توزیع نرمال پیروی می‌کند و مجموعه آنها دارای میانگین ۴۰ و با انحراف معیار مجذور ۵۰ می‌باشد. وزن درخواست‌های پردازشی یک بردار به طول تعداد درخواست‌هاست که ابتدا هر عضو آن یک عدد صحیح با توزیع یکنواخت در بازه ۱ تا ۱۶ است و سپس هر کدام بر مجموع آنها تقسیم شده تا به وزن‌های تصادفی با مجموع ۱ تبدیل شوند. درخواست‌ها همگی در زمان واحد و در ابتدا تولید می‌شوند.

هر الگوریتم بهینه‌ساز با ۱۰۰ نمونه جواب و در ۱۰۰ تکرار و سعی بر یافتن بهینه سراسری مسئله داشته است که پس از آن متوقف و بهترین جواب تاکنون برگردانده شده است. هر الگوریتم بهینه‌ساز ۵ اجرای مستقل داشته و نتایج آزمون فریدمن بر میانگین نتایج ۵ اجرا اعمال شده است. ثابت‌های اختصاصی هر الگوریتم نیز، همانند زیر بخش پیشین و مطابق با جدول (۹) تنظیم شده‌اند. نتایج حاصل اجرای الگوریتم‌های بهینه‌ساز برای حل مسئله شبیه‌سازی شده، در شکل‌های (۷ و ۸) قابل مشاهده است.

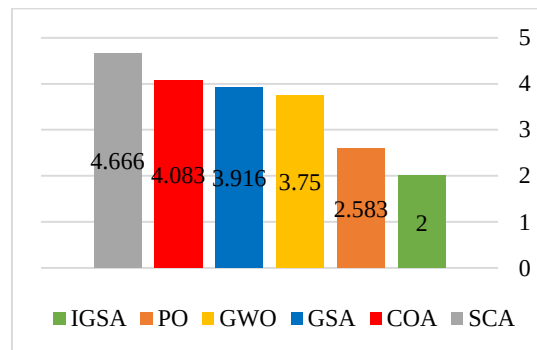
مقایسه نتایج الگوریتم‌های و سطح معناداری کمتر و اختلاف بین نتایج آنها چندان نیست، اما همچنان الگوریتم پیشنهادی در رتبه‌بندی آزمون فریدمن برای بهینه‌سازی مسائل توابع آزمایشی استاندارد و همچنین زمان‌بندی محاسبات، میانگین رتبه کمتری داشته است. آماره آزمون فریدمن برای نتایج بهینه‌سازی مدل سیستم با درجه آزادی $df_1 = 5$ و $df_2 = 55$ در جدول (۱۲) ذکر شده است.

جدول (۱۲): آماره آزمون فریدمن برای مسئله زمان‌بندی محاسبات

مقدار	آماره
۱۷/۲۲۳	χ^2
۴/۴۳۴	Iman-Davenport F

به جهت این که مسئله اصلی مورد مطالعه این مقاله زمان‌بندی محاسبات در محیط اینترنت اشیا است، نیاز به مقایسه الگوریتم پیشنهادی با یکی از روش‌های مطرح در این حوزه وجود دارد. از این رو در جدول (۱۳)، نتایج اجرای الگوریتم PSO-based scheduling (که در جدول (۱۳) به اختصار PSO نام گذاری شده) نیز در کنار روش پیشنهادی آورده شده تا تحلیل کامل‌تری از عملکرد روش پیشنهادی صورت گیرد. این اجرا در تمامی سناریوهای اشاره شده (تعداد درخواست ۱۰۰۰، ۳۰۰۰ و ۵۰۰۰) انجام شده است.

بررسی و تحلیل نتایج جدول (۱۳) نشان می‌دهد که الگوریتم پیشنهادی در پنج مورد از شش معیار، مقدار کمتری نسبت به الگوریتم PSO-based scheduling، که یک الگوریتم شناخته شده در این زمینه است، کسب کرده است و تنها از نظر معیار انرژی مصرفی منابع در محیطی با ۱۰۰۰ درخواست عملکردی به خوبی الگوریتم نداشته است که می‌تواند به دلیل کوچکتر بودن فضای جستجو ناشی از تاثیر بیشتر نوسانات متغیرهای تصادفی مسئله باشد. بیشترین بهبود مربوط به معیار تاخیر در محیطی با ۳۰۰۰ و ۵۰۰۰ درخواست پردازشی بوده که بیانگر توانایی الگوریتم پیشنهادی در کنترل تاخیر با افزایش بار سیستم است. با بررسی روند تمامی بهبودهای حاصل شده، قابلیت مقیاس پذیری الگوریتم IGSA برای حل مسائل بزرگتر مشخص خواهد شد.



شکل (۹): نمودار رتبه‌بندی فریدمن بر نتایج زمان‌بندی محاسبات

همان‌طور که از نتایج این آزمون مشهود است، الگوریتم IGSA با کسب کارایی برتر با میانگین رتبه ۲، نشان داده است عملکرد بهتری نسبت الگوریتم GSA دارد و برتری خود را نسبت به دیگر الگوریتم‌های فراابتکاری مقایسه شده، اثبات کرده است. همچنین پارامتر $p - value$ این آزمون آماری با مقدار ۰/۰۰۱۸۳ مجدداً منجر به رد فرض صفر شده که نتایج و تحلیل‌های حاصل را قابل اتکا می‌کند. میانگین رتبه هر الگوریتم در ستون مربوطه قابل مشاهده است. نتیجه مقایسه زوجی پس از آزمون (Post-hoc) با اصلاح هولم (Holm) نیز در جدول (۱۱) قابل مشاهده است.

جدول (۱۱): مقایسه زوجی پس از آزمون فریدمن برای مسئله زمان‌بندی محاسبات

مورد مقایسه	آماره تست پس از آزمون	$p - value$ تعدیل شده	نتیجه
IGSA-SCA	۳/۴۹۱	۰/۰۰۲	رد
IGSA-COA	۲/۷۲۷	۰/۰۲۵	رد
IGSA-GSA	۲/۵۰۹	۰/۰۳۶	رد
IGSA-GWO	۲/۲۹۱	۰/۰۴۳	رد
IGSA-PO	۰/۷۶۳	۰/۴۴۵	قبول

همان‌طور که با بررسی جدول (۱۱) مشخص می‌شود، الگوریتم پیشنهادی نسبت به تمام الگوریتم‌های COA، SCA و GWO و همچنین الگوریتم پایه (GSA) عملکرد بهتری با اختلاف معناداری دارد که مقدار $p - value$ تعدیل شده نشان‌دهنده آن است. با وجود این که الگوریتم PO از الگوریتم‌های شناخته‌شده، به‌روز و با کارایی خوب در حوزه الگوریتم‌های فراابتکاری است، الگوریتم پیشنهادی عملکرد ضعیف‌تری نسبت به آن نداشته و می‌تواند با این الگوریتم کارا، رقابت کند. با این که مقدار $p - value$ در

است. برای این نتایج ۵ اجرا مستقل در محیطی با ۱۰۰۰ درخواست پردازشی انجام شده است. در این جدول، سناریو ۲ طبق وزن‌های روش پیشنهادی (جدول (۶)) تنظیم شده است.

جدول (۱۴): نتایج اجرا پس از تغییر وزن اهداف

سناریو	هدف	وزن	میانگین شایستگی	انحراف معیار	شایستگی وزندار
۱	تاخیر	10^{-3}	۱۳۵۲۱/۷۴	۱۳۵/۲۹	۱۳/۶۹۳
	انرژی	10^{-3}	۱۷۲/۶۹	۲/۰۷	
۲	تاخیر	10^{-3}	۲۴۹۰/۶۹	۸۰/۵۱	۱۱/۶۱۷
	انرژی	10^{-4}	۹۱۲۷۲/۱۳	۲۷۷۱/۲۳	
۳	تاخیر	10^{-4}	۱۳۵۲۱/۷۴	۱۳۵/۲۹	۱/۳۶۹
	انرژی	10^{-4}	۱۷۲/۶۹	۲/۰۷	

همان‌طور که با بررسی جدول (۱۴) مشخص می‌شود، تعادل مناسب بین دو هدف تاخیر و انرژی مصرفی در محیطی با ۱۰۰۰ درخواست پردازشی، زمانی حاصل می‌شود که وزن هر معیار، مطابق جدول (۶) تعریف شده باشد.

۵- نتیجه‌گیری

در مطالعه حاضر، با بررسی الگوریتم GSA مشخص شد همگرایی زودهنگام یکی از معایب اصلی این الگوریتم می‌باشد و از طرف دیگر زمان‌بندی محاسبات یکی از چالش‌های مهم پیش رو در حوزه اینترنت اشیا می‌باشد. در مقاله حاضر، با ارائه راهکاری نوین برای بازمقداردهی نمونه جواب‌ها مبتنی بر منطق فازی و اصلاح معادلات محاسبه جرم به کمک روش مقیاس‌بندی مجموع، نقص معرفی شده از الگوریتم GSA برطرف شده است. برای حل مسئله زمان‌بندی محاسبات، یک محیط شبیه‌ساز شبکه اینترنت اشیا با دو هدف کاهش تاخیر و مصرف انرژی به طور کامل تعریف و طراحی شد و از این الگوریتم پیشنهادی برای یافتن بهترین حالت زمان‌بندی محاسبات استفاده شد.

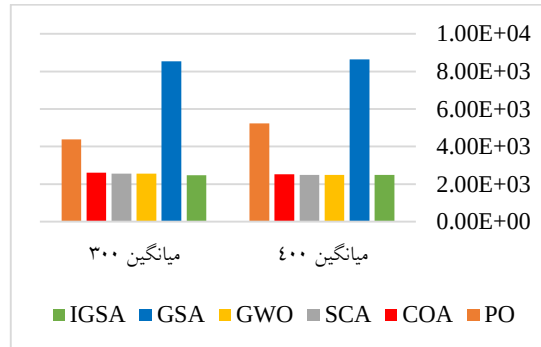
بررسی و تحلیل نتایج حاصل از اجرا، نشان دهنده عملکرد مناسب الگوریتم پیشنهادی و برتری آن در قیاس با الگوریتم GSA، GWO، SCA، COA و PO می‌باشد. الگوریتم پیشنهادی این پژوهش توانست برای حل توابع آزمایشی استاندارد در آزمون آماری فریدمن کارایی برتر را با مقدار ۲/۴۱۶ و برای حل مسئله زمان‌بندی محاسبات مجدد کارایی برتر را با مقدار ۱/۵ به دست

جدول (۱۳): نتایج عملکرد الگوریتم‌های PSO-based

scheduling و IGSA برای حل مسئله زمان‌بندی محاسبات

تعداد درخواست	پارامتر	IGSA	PSO	درصد بهبود
۱۰۰۰	تاخیر	۲۴۹۰/۶۹	۲۵۱۰/۴۷	۰/۷۹
	انرژی	۹۱۲۷۲/۱۳	۹۰۲۲۴/۰۳	-۱/۱۶
۳۰۰۰	تاخیر	۷۴۵۶/۲۱	۷۶۰۵/۰۵	۱/۹۶
	انرژی	۲۷۷۴۱۳/۰۱	۲۷۹۴۵۲/۵۶	۰/۷۳
۵۰۰۰	تاخیر	۱۲۴۶۴/۴۷	۱۲۷۰۸/۴۳	۱/۹۲
	انرژی	۴۹۰۵۷۱/۵	۴۹۱۲۱۱/۲۸	۰/۱۳

به‌منظور بررسی عملکرد در شرایطی که محدودیت زمانی درخواست‌های پردازشی سخت‌تر شده باشد، در ادامه به مرور نتایج در شرایط شبیه‌سازی شده آن پرداخته شده است.



شکل (۱۰): نمودار تغییر تاخیر در صورت تغییر مهلت زمانی

درخواست‌ها

همان‌طور که در نمودار شکل (۱۰) مشهود است، تغییر میانگین مهلت زمانی درخواست‌ها از ۴۰۰ به ۳۰۰ ثانیه و سخت‌تر شدن شرایط مهلت زمانی، تغییر زیادی بر عملکرد الگوریتم پیشنهادی از منظر تاخیر زمانی نداشته است و الگوریتم پیشنهادی بر خلاف الگوریتم‌های دیگری مثل PO همچنان بهترین پاسخ ممکن را در فضای جستجو، پیدا و برگردانده است.

وزن‌های به کار رفته برای تبدیل دو هدف تاخیر و انرژی مصرفی در جدول (۶) ذکر شد. برای بررسی بهتر عملکرد الگوریتم در صورت تغییر وزن‌ها، اجراهای تکمیلی انجام شده است. در این اجراها، وزن‌های متفاوتی برای این منظور در نظر گرفته شده است و نتایج حاصل از بهینه‌سازی با وزن‌های جدید توسط الگوریتم پیشنهادی در جدول (۱۴) قرار گرفته است. همچنین نتایج اجرای اصلی برای مقایسه تاثیر تغییر وزن‌ها نیز در این جدول قرار گرفته

پیشنهاد می‌شود در مطالعات آینده از توابعی برای نقض مهلت زمانی در کنار اهداف کاهش تاخیر و انرژی مصرفی استفاده شود. موضوع امنیت در شبکه اینترنت اشیا نیز حوزه مهمی است که می‌تواند پژوهش‌های گسترده‌ای را پوشش دهد. همچنین بررسی حساسیت الگوریتم پیشنهادی نسبت به پارامترهای داخلی آن (همچون gap و بازه‌های مجموعه‌های فازی تعریف شده) و پارامترهای از مسئله زمان‌بندی (مثل D_j , W_j , IT_j و تعداد منابع محاسباتی) می‌تواند به عنوان پژوهش‌هایی تاثیرگذار در این حوزه، در آینده مورد توجه قرار گیرد.

References

- [1] O. Aouedi, T.-H. Vu, A. Sacco, D. C. Nguyen, K. Piamrat, G. Marchetto, and Q.-V. Pham, "A Survey on Intelligent Internet of Things: Applications, Security, Privacy, and Future Directions," IEEE Communications Surveys & Tutorials, vol. 27, no. 2, pp. 1238–1292, 2025, doi: 10.1109/COMST.2024.3430368.
- [2] S. Latif, M. Driss, W. Boulilla, Z.-E. Huma, S. S. Jamal, Z. Idrees, and J. Ahmad, "Deep Learning for the Industrial Internet of Things (IIoT): A Comprehensive Survey of Techniques, Implementation Frameworks, Potential Applications, and Future Directions," Sensors, vol. 21, no. 22, p. 7518, 2021.
- [3] M. Majid, S. Habib, A. R. Javed, M. Rizwan, G. Srivastava, T. R. Gadekallu, and J. C.-W. Lin, "Applications of wireless sensor networks and internet of things frameworks in the industry revolution 4.0: A systematic literature review," Sensors, vol. 22, no. 6, p. 2087, 2022, doi: 10.3390/s22062087.
- [4] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," IEEE Internet of Things Journal, vol. 3, no. 5, pp. 637–646, 2016, doi: 10.1109/JIOT.2016.2579198.
- [5] R. Aburukba, M. AliKarrar, T. Landolsi, and K. El-Fakih, "Scheduling Internet of Things requests to minimize latency in hybrid Fog–Cloud computing," Future Generation Computer Systems, vol. 111, pp. 539–551, 2020.
- [6] E. Rashedi, H. Nezamabadi-Pour, and S. Saryazdi, "GSA: A gravitational search algorithm," Information Sciences, vol. 179, no. 13, pp. 2232–2248, 2009, doi: 10.1016/j.ins.2009.03.004.
- [7] L. J. Fogel, A. J. Owens, and M. J. Walsh, "Evolutionary programming made faster," IEEE Transactions on Evolutionary Computation, vol. 3, no. 2, pp. 82–102, 1999.
- [8] S. Mirjalili and S. Z. Mohd Hashim, "Training feedforward neural networks using hybrid particle swarm optimization and gravitational search algorithm," Applied Mathematics and Computation, vol. 218, no. 22, pp. 11125–11137, 2012.
- [9] S. D. Sağlam and G. K. Gözeri, "Some Properties of Boolean-like Laws in Fuzzy Logic," Symmetry, vol. 17, no. 4, p. 548, 2025.
- [10] Y. Sanjalawe, M. Al-Smadi, S. Al-Zoubi, and A. Al-Bataineh, "AI-driven job scheduling in cloud computing: a comprehensive review," Artificial Intelligence Review, vol. 58, no. 7, p. 197, 2025.
- [11] D. Alsadie, "Advancements in heuristic task scheduling for IoT applications in fog-cloud computing: challenges and prospects," PeerJ Computer Science, vol. 10, p. e2128, 2024.
- [12] Avan, A. Azim, and Q. H. Mahmoud, "A State-of-the-Art Review of Task Scheduling for Edge Computing: A Delay-Sensitive Application Perspective," Electronics, vol. 12, no. 12, p. 2599, 2023.
- [13] M. A. El-Shorbagy, A. Bouaouda, L. Abualigah, and F. A. Hashim, "Stochastic

بیاورد. بررسی پارامتر $p - value$ در هر دو آزمون فریدمن مشخص می‌کند اختلاف قابل توجه بین نتیجه الگوریتم پیشنهادی و دیگر الگوریتم‌ها وجود دارد و همچنین تاییدی بر قابل اتکا بودن نتایج رتبه‌بندی آزمون فریدمن می‌باشد. همچنین مشخص شد روش پیشنهادی در مقایسه با روش PSO-based scheduling (به عنوان یکی از روش‌های کلاسیک حوزه زمان‌بندی محاسبات)، قادر به ارائه راهکاری بهتر از منظر معیارهای تاخیر و انرژی، برای تقسیم وظایف محاسباتی به ویژه در مسائل بزرگ می‌باشد.

- Fractal Search: A Decade Comprehensive Review on Its Theory, Variants, and Applications,” *Computer Modeling in Engineering & Sciences*, vol. 142, no. 3, pp. 2339–2404, 2025.
- [14] S. Mirjalili, S. M. Mirjalili, and A. Lewis, “Grey wolf optimizer,” *Advances in Engineering Software*, vol. 69, pp. 46–61, 2014.
- [15] S. Mirjalili, “SCA: A sine cosine algorithm for solving optimization problems,” *Knowledge-Based Systems*, vol. 96, pp. 120–133, 2016.
- [16] M. A. Elaziz, A. A. Ewees, D. Oliva, and S. Mirjalili, “Crayfish optimization algorithm,” *Artificial Intelligence Review*, vol. 55, pp. 5565–5609, 2022.
- [17] J. Lian, G. Hui, L. Ma, T. Zhu, X. Wu, and A. A. Heidari, “Parrot optimizer: Algorithm and applications to medical problems,” *Computers in Biology and Medicine*, vol. 172, p. 108064, 2024.
- [18] F. A. Saif, R. Latip, Z. M. Hanapi, and K. Shafinah, “Multi-Objective Grey Wolf Optimizer Algorithm for Task Scheduling in Cloud-Fog Computing,” *IEEE Access*, vol. 11, pp. 20635–20646, 2023.
- [19] M. Masdari, F. Salehi, M. Jalali, and M. Bidaki, “A Survey of PSO-Based Scheduling Algorithms in Cloud Computing,” *Journal of Network and Systems Management*, vol. 25, no. 1, pp. 122–158, 2017.
- [20] S. Mirjalili and S. Z. Mohd Hashim, “A new hybrid PSOGSA algorithm for function optimization,” *Applied Soft Computing*, vol. 13, no. 5, pp. 2292–2304, 2013.
- [21] A. Heidari, S. Mirjalili, and H. Faris, “Hybrid GSA-GA algorithm for constrained optimization problems,” *Engineering Applications of Artificial Intelligence*, vol. 41, pp. 124–135, 2015.
- [22] E. Rashedi, H. Nezamabadi-Pour, and S. Saryazdi, “New mass calculation functions in gravitational search algorithm,” *Journal of Intelligent & Fuzzy Systems*, vol. 27, no. 2, pp. 567–576, 2014.
- [23] R. Mahmud, R. Kotagiri, and R. Buyya, “Scheduling IoT applications in fog computing,” *Future Generation Computer Systems*, vol. 82, pp. 416–426, 2018.
- [24] M. Aazam and E.-N. Huh, “Fog computing and smart gateway based communication for cloud of things,” *Future Generation Computer Systems*, vol. 59, pp. 1–8, 2016.
- [25] Faramarzi, M. Heidarinejad, S. Mirjalili, and A. H. Gandomi, “Equilibrium optimizer: A novel optimization algorithm,” *Knowledge-Based Systems*, vol. 191, p. 105190, 2020.
- [26] M. Mohammadi, F. BahraniPour, S. Ebrahimi Mood, and M. Farshi, “Security-aware resource allocation in fog computing using a meta-heuristic algorithm,” *Cluster Computing*, vol. 28, no. 2, p. 104, 2025.
- [27] O. Rainio, J. Teuho, and R. Klén, “Evaluation metrics and statistical tests for machine learning,” *Scientific Reports*, vol. 14, no. 1, p. 6086, 2024.
- [28] L. J. Fogel, A. J. Owens, and M. J. Walsh, *Artificial Intelligence through Simulated Evolution*. New York, NY, USA: Wiley, 1966.

Delay and Energy Optimization in Cloud-Edge Computing System Using Improved Gravitational Search Algorithm

Amir Mohammad Sanei¹, Mohammad Farshi^{2*}, Sepehr Ebrahimi Mood³

¹Master Of Science, Department of Computer Science, Yazd University, Yazd, Iran

²Associate Professor, Department of Computer Science, Yazd University, Yazd, Iran

³Assistant Professor, Department of Computer Science, Yazd University, Yazd, Iran

Article Information

Original Research Paper

Received:

2026 May 16

Accepted:

2026 September 22

Keywords:

Internet of Things, Task Scheduling, Cloud-Edge Computing, Optimization, Gravitational Search Algorithm, Fuzzy Inference System

Corresponding Author*:

mfarshi@yazd.ac.ir

Abstract

This article investigates task scheduling strategies in cloud-edge computing layers and proposes an appropriate solution for this problem. The scheduling is evaluated with the dual objectives of minimizing latency and energy consumption. To solve this scheduling problem, an improved gravitational search algorithm is employed as the algorithm proposed in this research. The enhancement implemented in this study on the gravitational search algorithm utilizes fuzzy logic to prevent entrapment in local optima. The improved gravitational search algorithm, alongside the basic gravitational search algorithm, Gray Wolf Optimizer, Sine Cosine Algorithm, Chimp Optimization Algorithm, and Crab Optimization Algorithm, was evaluated and tested in two different environments. Furthermore, Friedman's statistical test was used for the statistical analysis of the results. The proposed algorithm was first applied to optimize CEC benchmark functions, and the obtained results along with the statistical test indicate the improvement of the proposed algorithm over the basic gravitational search algorithm, as well as its superiority compared to the other algorithms. Subsequently, the improved gravitational search algorithm and the aforementioned other algorithms were used to optimize task scheduling in a cloud-edge environment. The execution results of the algorithms and the statistical analysis demonstrate the enhanced performance of the gravitational search algorithm after the improvement. Moreover, by examining the obtained results, the superior performance of the proposed algorithm compared to the other algorithms can be observed.



: 10.22034/ABMIR.2026.24574.1244

E-ISSN: [2821-2037](#)

/The Author 2026. Published by Yazd University This is an open access article under the CC BY 4.0 License <https://creativecommons.org/licenses/by/4.0/>.

