علوم رایانش و فناوری اطلاعات
نشریه علمی انجمن کامپیوتر ایران
مجلد ۲۳، شماره ۱، ۱۴۰۴
صفحات ۸- ۲۱
مقاله پژوهشی

جایابی سرویس‌های اینترنت اشیا در محیط‌های مه–ابرسلسه‌مراتبی با استفاده از الگوریتم ژنتیک مبتنی بر نخبه‌گرایی

فرهاد طاوسی[۱]، سعدون عزیزی[۲*]، عبدالباقی قادرزاده[۳]

*نویسنده مسئول، دریافت: ۱۴۰۱/۱۲/۰۸، بازنگری: ۱۴۰۲/۰۳/۱۱، پذیرش: ۱۴۰۲/۰۴/۰۶

[۱] استاد مدعو گروه مهندسی کامپیوتر و فناوری اطلاعات، دانشگاه فنی و حرفه‌ای، تهران، ایران

[۲] دانشیار گروه مهندسی کامپیوتر و فناوری اطلاعات، دانشگاه کردستان، سنندج، ایران

[۳] استادیار گروه مهندسی کامپیوتر و فناوری اطلاعات، دانشگاه آزاد اسلامی، سنندج، ایران

چکیده

رایانش مه- ابر سلسله‌مراتبی، یک الگوی محاسباتی امیدبخش برای پردازش و ذخیره‌سازی داده‌های حجیم تولیدشده توسط دستگاه‌های اینترنت اشیا است. با توجه به اینکه گره‌های محاسباتی مه توزیع‌شده، ناهمگن و دارای محدودیت منابع هستند، نمی‌توان همه سرویس‌های اینترنت اشیا را داخل محیط مه استقرار داد. بنابراین، لازم است یک الگوریتم کارآمد برای انتخاب مجموعه سرویس‌هایی که باید روی محیط مه قرار داده شوند طراحی کرد. وقتی که نیازمندی‌های متنوع سرویس‌ها شامل زمان پاسخ و هزینه درنظر گرفته شود مسئله پیچیده‌تر می‌شود. با این انگیزه، در این مقاله ابتدا یک مدل بهینه‌سازی برنامه‌ریزی عدد صحیح مختلط برای مسئله جایابی سرویس‌های اینترنت اشیا در محیط مه-ابر سلسله‌مراتبی با هدف کمینه‌سازی زمان پاسخ، هزینه و هدررفت منابع ارائه می‌شود. سپس به‌منظور حل مدل به‌صورت کارآمد، یک الگوریتم ژنتیک مبتنی بر نخبه‌گرایی پیشنهاد می‌گردد که در آن سعی می‌شود تعادل خوبی بین اکتشاف و بهره‌برداری برقرار شود. آزمایش‌های گسترده و متنوعی برای ارزیابی عملکرد روش پیشنهادی انجام شده است. نتایج حاصل نشان می‌دهد که روش ارائه‌شده از نظر معیارهای کیفیت سرویس، هزینه و بهره‌وری منابع، به‌مراتب بهتر از سایر الگوریتم‌ها عمل می‌کند. به‌طور ویژه، روش پیشنهادی مقدار تابع هدف را به‌طور متوسط بین ۴۲.۳ تا ۴۹.۸ درصد بهبود می‌دهد.

کلمات کلیدی: اینترنت اشیا، رایانش مه-ابر سلسه‌مراتبی، جایابی سرویس، الگوریتم ژنتیک مبتنی بر نخبه‌گرایی، کیفیت سرویس، هزینه، بهره‌وری منابع

۱- مقدمه

با توجه به تغییرات سبک زندگی، که از حالت سنتی به مدرن در حال وقوع است، سیستم‌های کامپیوتری نیز باعث تسریع در این تغییرات شده‌اند. با ظهور اینترنت اشیا (IoT) تمامی تجهیزات از طریق اینترنت، قابل دسترسی هستند [۱]. این دستگاه‌های هوشمند متصل به هم شامل حسگرها، دوربین-ها، محرک‌ها و کنتورهای هوشمند، حجم عظیمی از داده‌ها را تولید می کنند که باید ذخیره‌سازی، پردازش و تجزیه‌وتحلیل شوند تا اطلاعات ارزشمندی به‌دست آید. با این حال، دستگاه‌های اینترنت اشیا حتی قوی‌ترین دستگاه‌های هوشمند، قادر نخواهد بود به‌تنهایی این نیازمندی‌ها را برآورده کنند [۲، ۳]. با توجه به اینکه دستگاه‌های اینترنت اشیا دارای محدودیت منابع از قبیل توان محاسباتی، ظرفیت ذخیره‌سازی، طول عمر باطری و پهنای‌باند هستند، در مواجهه با حجم زیاد داده‌های تولیدشده دچار مشکل می‌شوند. برای مدیریت حجم بسیار زیاد داده‌های تولیدشده توسط این دستگاه‌ها، الگوهای محاسباتی متعددی از قبیل رایانش ابری، رایانش مه و

رایانش لبه ارائه شده است [۴]. در روش‌های مبتنی بر ابر، داده‌هایی که نیازمند پردازش سنگین هستند به مراکز داده ابری ارسال شوند [۵، ۶، ۷]. انتقال حجم بالای داده‌های تولیدشده به ابر، منجر به مصرف بالای پهنای باند شبکه و مسائل امنیتی می‌شود. همچنین، این مدل محاسباتی، جوابگوی زمان پاسخ برنامه‌های کاربردی حساس به تأخیر نیست.

در سال ۲۰۱۲ سیسکو مدل رایانش مه را پیشنهاد داد [۸]. الگوی رایانش مه، امکان انجام بسیاری از کارهای محاسباتی و ذخیره‌سازی در دستگاه‌های لبه شبکه، تحت عنوان *گره‌های مه* را فراهم می‌سازد. اگرچه قراردادن منابع در لبه شبکه سبب کاهش تأخیر در انتقال داده‌ها و در نتیجه کاهش زمان تحویل سرویس‌ها می‌شود اما گره‌های مه دارای محدودیت منابع هستند [۹]. بنابراین، رایانش مه به‌تنهایی قادر به پاسخگویی به نیازهای برنامه‌های اینترنت اشیاء نیست. در واقع، این الگو یک مکمل برای رایانش ابری محسوب می‌شود. ترکیب مه و ابر باعث شکل‌گیری یک سیستم محاسباتی جدید به نام رایانش مه–ابر شده است. با کمک این سیستم می‌توان از برنامه‌های IoT با نیازهای مختلف پشتیبانی کرد. این باعث می‌شود کیفیت خدمات به‌طور چشمگیری بهبود پیدا کند [۱۰، ۱۱، ۱۲].

سیستم رایانشی مه-ابر هنوز در مراحل ابتدایی خود است. اگرچه این الگوی رایانشی برای سرویس‌های اینترنت اشیا دارای مزیت‌های بسیاری شامل به‌حداقل‌رساندن تأخیر، کاهش ترافیک شبکه و کاهش هزینه می‌باشد اما معضلات متعددی دارد که نیاز به مطالعه و تحقیق بیشتری دارد. تخصیص منابع، یکی از مهم‌ترین معضلات چنین سیستم‌هایی به‌شمار می‌رود. مسئله جایابی سرویس‌های اینترنت اشیا در محیط‌های سلسله‌مراتبی مه–ابر از دو جنبه تأثیرگذار است: جنبه اول شامل کیفیت و هزینه خدمات است؛ جنبه دوم استفاده بهینه از منابع است [۱۳]. ما در [۱۴] برای حل مسئله جایابی سرویس‌ها با هدف دستیابی به کیفیت خدمات و استفاده بهینه از منابع، یک الگوریتم ابتکاری ارائه دادیم. در این پژوهش، مدل جامع‌تری برای این مسئله ارائه می‌شود که در آن، علاوه بر کیفیت خدمات و بهره‌وری منابع، هزینه ارائه خدمات نیز درنظر گرفته شده است. علاوه بر این، به‌منظور جلوگیری از بهینه محلی، یک الگوریتم فراابتکاری کارآمد برای حل مدل پیشنهادی ارائه می‌گردد.

در این مقاله ما ابتدا یک مدل برنامه‌ریزی خطی صحیح مختلط[1] با هدف افزایش کیفیت خدمات، کاهش هزینه جایابی و کمینه‌سازی هدررفت منابع محاسباتی برای مسئله جایابی سرویس‌های اینترنت اشیا در محیط‌های مه-ابرسلسله‌مراتبی ارائه می‌دهیم. مدل پیشنهادی به‌صورت حاصل‌جمع وزنی اهداف بهینه‌سازی بیان می‌شود. سپس به‌منظور حل کارآمد مدل ارائه‌شده، الگوریتم ژنتیک مبتنی بر نخبه‌گرایی پیشنهاد می‌شود. در نهایت، کارایی روش پیشنهادی توسط آزمایش‌های تجربی متعددی ارزیابی می‌گردد. نتایج حاصل از آزمایش‌های انجام‌شده نشان می‌دهد که روش پیشنهادی در مقایسه با رقبای خود، از عملکرد بسیار مطلوبی برخوردار است. دلیل استفاده از الگوریتم ژنتیک مبتنی بر نخبه‌گرایی برای حل مدل ارائه‌شده این است که این رویکرد تضمین می‌کند که بهترین جواب‌ها از هر نسل برای نسل بعدی حفظ شوند. به عبارت دیگر، اطلاعات ارزشمند هر نسل به‌طور مستقیم به نسل بعدی منتقل می‌گردد. این استراتژی به الگوریتم کمک می‌کند تا سریع‌تر و کارآمدتر به یک راه‌حل بهینه یا نزدیک به بهینه همگرا شود.

ساماندهی مقاله در ادامه به این صورت است: در بخش ۲ به بررسی تحقیقات پیشین پرداخته می‌شود. معماری سیستم درنظر گرفته‌شده و فرموله‌سازی مسئله در بخش ۳ ارائه می‌گردد. روش پیشنهادی برای حل مدل در بخش ۴ توصیف می‌شود. بخش ۵ و ۶ به‌ترتیب به ارزیابی کارایی اختصاص داده شده است. در نهایت، بحث مختصر، نتیجه‌گیری و پیشنهادهایی برای آینده در بخش ۶ و ۷ آورده شده است.

۲- تحقیقات مرتبط

در [۱۵]، نویسندگان یک رویکرد یادگیری تقویتی عمیق برای قراردادن خدمات پیشگیرانه در محیط مه ارائه داده‌اند. این روش با استفاده از یک فرایند تصمیم‌گیری مارکوف مقیاس‌پذیر، توسعه یافته است. هدف اصلی این مطالعه، بهبود کیفیت خدمات تجربه‌شده توسط کاربران اینترنت اشیا است.

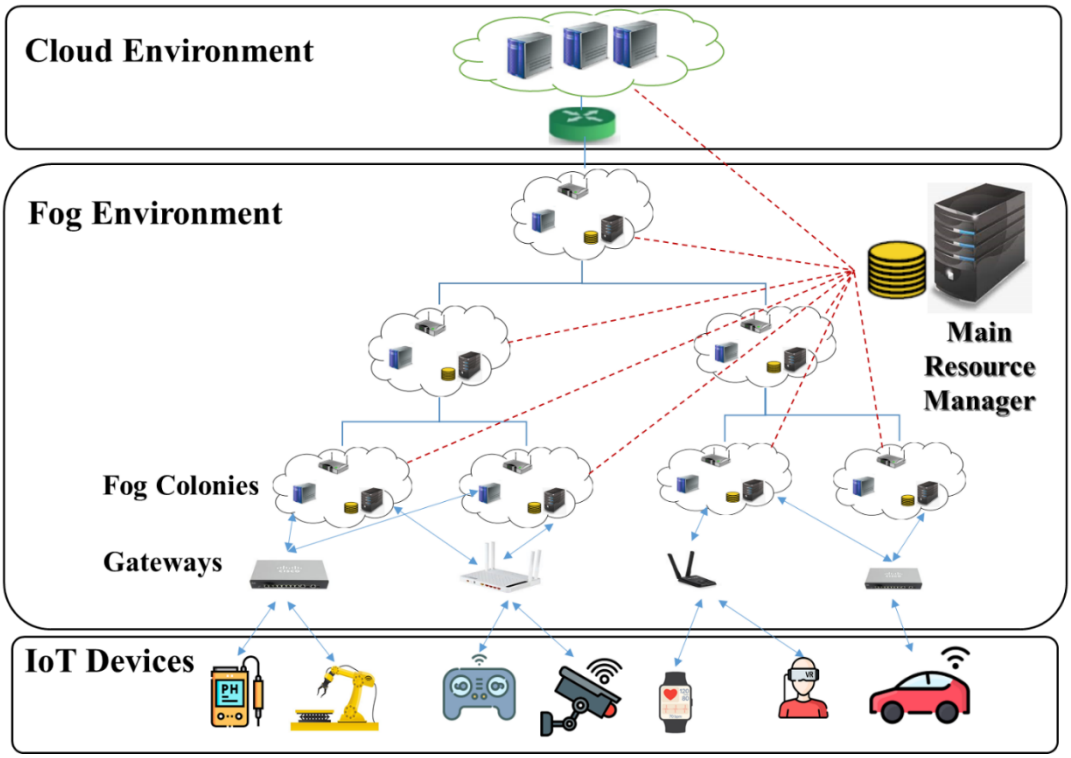
نویسندگان در [۱۶] سرویس‌های اینترنت اشیا را به دو دسته حیاتی و عادی تقسیم کرده‌اند. سپس دو استراتژی ابتکاری با هدف کاهش زمان پاسخ کاربران و کم‌کردن مصرف انرژی سیستم پیشنهاد داده‌اند. در [۱۷] روشی برای مدیریت برنامه‌ها و تخصیص منابع به آنها در محیط مه-ابر پیشنهاد شده است .در این پژوهش با توجه به پارامترهایی همچون مهلت زمانی تعیین‌شده، میزان داده ورودی و تعداد دفعات درخواست به ازای هر برنامه، یک دسته‌بندی برای برنامه‌ها انجام شده است. سپس براساس اولویت آنها، مجموعه‌ای از برنامه‌ها در لایه مه مستقر می‌شوند و برنامه‌هایی که تحمل تأخیر را دارند به مراکز داده ابری ارسال می‌شوند.

برای حل مسئله جایابی سرویس‌های اینترنت اشیا در [۱۸] یک روش ابتکاری کارآمد ارائه گردیده است. اهداف این کار، کاهش تاخیر و هزینه جایابی سرویس‌ها و افزایش بهره‌وری منابع مه-ابر می‌باشد. به‌منظور کاهش مصرف انرژی و هزینه محاسبات و همچنین افزایش کیفیت سرویس کاربران در محیط مه-ابر، در [۱۹] یک الگوریتم ژنتیک آگاه از اولویت ارائه شده است. نتایج شبیه‌سازی نشان می‌دهد که الگوریتم پیشنهادی از زمان همگرایی بسیار خوبی برخوردار است. نویسندگان در [۲۰] برای قراردادن سرویس‌های پرکاربرد در نزدیکی کاربران از پارامتر نرخ درخواست هر سرویس برای تعیین اولویت سرویس‌ها استفاده کرده‌اند. در این کار، پارامترهای تأخیر سرویس و مقدار استفاده از شبکه ارزیابی شده است. به‌منظور جایابی سرویس به‌صورت پویا در محیط مه، یک چارچوب کارآمد در [۲۱] ارائه شده است که در آن یک سرویس تنها در صورتی در نزدیکی کاربران مستقر می‌شود که برای آن تقاضا وجود داشته باشد. در غیر این صورت سرویس موردنظر از آن مکان برداشته می‌شود و منابع آن آزاد می‌گردد. در این کار، نویسندگان، دو الگوریتم ابتکاری برای مسئله استقرار سرویس‌ها ارائه داده‌اند. تمرکز الگوریتم اول روی کاهش زمان نقض مهلت است در حالی که تمرکز الگوریتم دوم روی کمینه‌سازی هزینه محاسبات و پهنای‌باند می‌باشد.

1. Mixed Integer Linear Programming (MILP)

جدول ۱. خلاصه کارهای مرتبط و مقایسه آنها

مرجع	محیط	اهداف بهینه‌سازی	ساختار سلسله‌مراتبی	خوشه بندی گره‌های مه	الگوریتم
[۲]	مه – ابر	تأخیر کار، هزینه	×	×	الگوریتم ژنتیک
[۲۱]	مه	تأخیر	✓	×	الگوریتم‌های حریصانه
[۲۰]	مه – ابر	تعداد گام، تأخیر	✓	×	الگوریتم‌های ابتکاری
[۳۲]	مه - ابر	زمان پاسخ، انرژی، هزینه	✓	✓	الگوریتم‌های ابتکاری
[۱۷]	مه - ابر	زمان پاسخ، مصرف منابع	×	✓	الگوریتم‌های ابتکاری
[۲۷]	مه	زمان تکمیل، مصرف انرژی	×	×	الگوریتم ژنتیک مبتنی بر پارتو
[۱۶]	مه-ابر	انرژی مصرفی، زمان پاسخ	×	✓	الگوریتم ابتکاری
[۱۹]	مه-ابر	زمان پاسخ، مصرف انرژی، هزینه محاسبات	×	✓	الگوریتم ژنتیک آگاه از اولویت
[۲۷]	لبه-ابر	زمان پاسخ، هزینه عملیات، - دسترس‌پذیری	✓	×	الگوریتم ژنتیک
[۲۸]	مه - ابر	توان عملیاتی و مصرف انرژی	✓	✓	الگوریتم بهینه‌سازی نهنگ
[۲۹]	مه	تأخیر خدمات	×	✓	الگوریتم ابتکاری
[۱۴]	مه-ابر	زمان‌پاسخ، هدررفت منابع	✓	✓	منطق فازی، ابتکاری
این کار	مه- ابر	زمان‌پاسخ، هدررفت منابع، هزینه سرویس	✓	✓	الگوریتم ژنتیک مبتنی بر نخبه‌گرایی



شکل ۱. معماری سلسه‌مراتبی IoT-Fog-Cloud

در [۲۲]، مسئله قراردادن برنامه‌های کاربردی اینترنت اشیا در یک معماری چندلایه آگاه از کیفیت خدمات، بررسی شده است. سپس با استفاده از الگوریتم ژنتیک، محل استقرار برنامه‌ها روی گره‌های مه مشخص شده است. به‌منظور پیداکردن محل استقرار سرویس‌ها در محیط مه-ابر با هدف کاهش هزینه سرویس، مصرف انرژی، زمان پاسخ و افزایش بهره‌وری منابع مه، در [۲۳] از الگوریتم مدل توسعه متن-باز استفاده شده است.

در مقاله [۲۴]، یک روش مبتنی بر پارتو برای قراردادن برنامه‌های کاربردی اینترنت اشیا روی مه ارائه شده است. این مدل برنامه‌های کاربردی را طبق ماشین متناهی، مدل می‌کند و با استفاده از سه هدف متضاد بهینه‌سازی زمان تکمیل، مصرف انرژی و هزینه‌های اقتصادی، اقدام به قراردادن برنامه‌ها روی محیط مه می‌کند. نویسندگان مقاله [۲۵]، روشی برای مسئله قراردادن کارها در یک محیط مه-ابر ارائه داده‌اند. در این مطالعه، فرض بر این است که سرویس‌ها شامل حس‌کردن، محرک، محاسبات و ذخیره‌سازی هستند و به‌صورت ایستا روی گره‌های مه یا مراکز داده ابری قرار می‌گیرند. هر درخواست سرویس دستگاه‌های اینترنت اشیا توسط گره کنترل‌کننده کلونی دریافت می‌شود و آن گره تصمیم می‌گیرد که با توجه نیازمندی‌های سرویس درخواست‌شده، کار به یکی از گره‌های کلونی تخصیص داده شود یا اینکه به نزدیک‌ترین کلونی همسایه ارسال شود یا به مراکز داده روی ابر ارسال گردد. در [۲۶]، نویسندگان با استفاده از الگوریتم بهینه‌سازی ازدحام ذرات و قوانین منطق فازی، یک روش زمانبندی برای اجرای کارهای تقاضاشده توسط دستگاه‌های اینترنت اشیا در محیط مه ارائه کرده‌اند. در این کار، برنامه‌های کاربردی به دو دسته حساس به تأخیر و مدارا با تأخیر تقسیم شدند و اهداف آن، کاهش استفاده از شبکه و تأخیر انتشار می‌باشد. در [۲۷]، برای حل مسئله جایابی سرویس‌های اینترنت اشیا، یک روش چندهدفه با استفاده از الگوریتم ژنتیک ارائه گردیده است. در این کار برای دستیابی به اهداف، ابتدا برنامه‌های کاربردی اولویت‌بندی شدند سپس با به‌کارگیری یک الگوریتم ژنتیک مبتنی بر نخبه‌گرایی سعی می‌کند کروموزمی را تولید کند که بیانگر ترتیب جایابی برنامه‌های کاربردی در گره‌های رایانشی محیط لبه، هسته شبکه یا مراکز داده ابری باشد. اهداف این کار، کاهش نقض مهلت زمانی درخواست‌ها و هزینه عملیات و افزایش دسترس‌پذیری سرویس‌ها می‌باشد.

۳- معماری سیستم و فرموله‌سازی مسئله

در این بخش، ابتدا معماری سیستم پیشنهادی توصیف می‌شود. سپس فرموله‌سازی مسئله در قالب برنامه‌ریزی خطی عدد صحیح مختلط (MILP) ارائه می‌گردد.

۳-۱- معماری سیستم

معماری سیستم درنظر گرفته‌شده، در شکل (۱) نشان داده شده است. در ادامه، عناصر تشکیل‌دهنده آن و چگونگی تعامل آنها با یکدیگر بحث می‌شود.

دستگاه‌های اینترنت اشیا: در معماری درنظر گرفته‌شده دستگاه‌های اینترنت اشیا در پایین‌ترین لایه قرار دارند که از لحاظ جغرافیایی در نقاط مختلف توزیع شده و غیرهمگن می‌باشند. در این لایه داده‌ها توسط دستگاه-های اینترنت اشیا از قبیل حسگرهای بی‌سیم، وسایل نقلیه، دوربین‌ها، ساعت‌های هوشمند و غیره تولید شده و به دروازه‌ها ارسال می‌گردد. گره‌های دروازه درخواست‌های ایجادشده توسط لایه اینترنت اشیا را دریافت و آنها را به گره مدیر منابع در نزدیک‌ترین کلونی مه ارسال می‌کنند. شایان ذکر است که برخی از درخواست‌ها ممکن است توسط خود دستگاه‌های اینترنت اشیا پاسخ داده شوند که در این صورت نیازی به برون‌سپاری ندارند [۲۹].

محیط مه: این بخش به‌عنوان لایه میانی بین دستگاه‌های اینترنت اشیا و ابر عمل می‌کند. در این محیط، برخی از دستگاه‌ها که دارای قابلیت مجازی-سازی هستند، مانند کامپیوترهای شخصی، مسیریاب‌ها، سوئیچ‌ها، سرورهای محلی و نقاط دسترسی قرار دارند. به این دستگاه‌ها گره‌های مه گفته می‌شود. این گره‌ها به‌صورت خودمختار به یک شبکه پویا متصل و خوشه‌بندی می‌شوند. هریک از خوشه‌های ایجادشده را یک کلونی مه می‌نامند. در این کار فرض شده است که معماری کلونی‌ها به‌صورت سلسله‌مراتبی است و ظرفیت منابع گره‌های موجود در کلونی‌های سطوح پایین‌تر، کمتر از گره‌های موجود در کلونی‌های سطوح بالاتر می‌باشد. در هر کلونی، یک گره مدیر منابع مه وجود دارد که درخواست‌های ارسال‌شده به آن کلونی را جمع‌آوری می‌کند [۳۰]. سپس درخواست‌ها به مدیر منابع اصلی، ارجاع داده می‌شود تا بر اساس الگوریتم تعبیه‌شده در آن، مکان هر درخواست یا سرویس مشخص شود.

محیط ابر: در محیط ابر، یک مرکز داده شامل تعدادی زیادی سرور است. سرویس‌های اینترنت اشیا که محیط مه قادر به تأمین نیازمندی‌های آنها نیست یا حساس به تأخیر نیستند در سرورهای واقع در مرکز داده ابر جای داده و اجرا می‌شوند.

۳-۲- فرموله‌سازی مسئله

در این بخش، مسئله جایابی سرویس‌های اینترنت اشیا در یک سیستم مه-ابرسلسله‌مراتبی فرموله‌سازی می‌شود. برای سهولت و درک بهتر فرمول‌ها، نمادهای استفاده‌شده و توصیف آنها به‌طور یک‌جا در جدول (۲) آمده است.

جدول ۲. لیست نمادها و توصیف آنها

نماد	توصیف
N	مجموعه گره‌های رایانشی در محیط‌های مه و ابر
S	مجموعه سرویس‌ها که باید در محیط‌های مه-ابر جایابی شوند
N_j^c	تعداد CPU در گره N_j
N_j^r	ظرفیت حافظه گره N_j
N_j^{cc}	کل ظرفیت CPU گره N_j
N_j^h	فاصله بین گره N_j و گره مدیر منابع در کلونی
N_{ij}^{cc}	مقدار ظرفیت CPU تخصیص‌یافته به سرویس S_i توسط گره N_j
U_j^{cpu}	نرمال‌شده مقدار ظرفیت CPU استفاده‌شده در گره N_j
U_j^{ram}	نرمال‌شده مقدار حافظه مصرف‌شده در گره N_j
W_j^{cpu}	نرمال‌شده مقدار ظرفیت CPU هدررفته در گره N_j

W_j^{ram}	نرمال‌شده مقدار حافظه هدررفته در گره N_j
$\mathrm{S}_i^{\mathrm{c}}$	تعداد CPU موردنیاز سرویس S_i
$\mathrm{S}_i^{\mathrm{r}}$	مقدار حافظه موردنیاز سرویس S_i
$\mathrm{S}_i^{\mathrm{s}}$	تعداد دستورالعمل‌های سرویس S_i
S_i^{d}	مهلت‌زمانی تعیین‌شده توسط کاربر برای سرویس S_i
S_i^{resp}	زمان پاسخ به هریک از درخواست‌ها توسط سرویس S_i
t_i^{in}	مدت‌زمانی که صرف می‌شود تا گره‌ای که سرویس S_i روی آن قرار دارد تقاضای سرویس را دریافت کند
t_i^{exe}	زمان لازم برای اجرای دستورالعمل‌های سرویس S_i
t_i^{out}	زمان لازم برای برگشت پاسخ به دستگاه اینترنت اشیای متقاضی
S_i^{viol}	مدت‌زمان تخطی از مهلت زمانی هر درخواست اینترنت اشیا برای سرویس S_i
$d_i^{D\to G}$	تأخیر از دستگاه اینترنت اشیا تا گره دروازه
$d_i^{G\to FRM}$	تأخیر از گره دروازه تا گره مدیر منابع در کلونی که سرویس S_i روی یکی از گره‌های رایانشی آن جایابی شده است
$d_i^{FRM\to N_j}$	تأخیر از گره مدیر منابع تا گره رایانشی میزبان سرویس S_i
φ_c^{cpu}	هزینه هر واحدمحاسباتی مجازی در هر ساعت در محیط مه
φ_c^{RAM}	هزینه هر گیگابایت حافظه استفاده‌شده در هر ساعت در محیط مه
$\acute{\varphi}_c^{cpu}$	هزینه هر واحد محاسباتی مجازی (VCPU) در هر ساعت در محیط ابر
$\acute{\varphi}_c^{RAM}$	هزینه هر گیگابایت حافظه استفاده‌شده در هر ساعت در محیط ابر
δ_i^k	زمان تأخیر انتقال در گام kام برای سرویس S_i
γ_i^k	زمان تأخیر انتشار در گام kام برای سرویس S_i
x_{ij}	چنانچه سرویس $S_i \in S$ روی گره $N_j \in N$ قرار گرفته باشد برابر ۱ در غیر این صورت، برابر صفر است

سرویس: فرض بر این است که تعداد n سرویس وجود دارد که توسط مجموعه $S = \{S_1, S_2, \dots, S_n\}$ نشان داده می‌شود. هریک از سرویس‌ها دارای ویژگی‌های زیر هستند. $\mathrm{S}_i^{\mathrm{c}}$ بیانگر مقدار CPU موردنیاز سرویس i است. $\mathrm{S}_i^{\mathrm{r}}$ نشان‌دهنده مقدار حافظه موردنیاز سرویس i است. S_i^{λ} بیانگر نرخ درخواست برای سرویس i است که واحد آن تعداد درخواست به ازای یک ثانیه است. $\mathrm{S}_i^{\mathrm{s}}$ نشان‌دهنده اندازه یا تعداد دستورالعمل‌های i (واحد: میلیون دستورالعمل) می‌باشد. همچنین $\mathrm{S}_i^{\mathrm{d}}$ بیانگر مهلت موردانتظار (واحد: ثانیه) به ازای هر درخواست است.

در این کار می‌خواهیم سرویس‌های موردنظر دستگاه‌های اینترنت اشیا را روی منابع محاسباتی در دسترس به گونه‌ای قرار دهیم که:

الف) مقدار زمان نقض مهلت هر درخواست کاهش یابد.

ب) میزان هدررفت منابع در گره‌های فعال حداقل باشد.

ج) هزینه خدمات ارائه‌شده به درخواست‌های دستگاه‌های اینترنت اشیا حداقل باشد.

۳-۲-۱ نقض مهلت

فرض کنید $X_{n\times m}$ یک ماتریس دودویی برای نمایش نگاشت سرویس‌ها به گره‌های محاسباتی باشد که در آن، مقدار x_{ij} برابر ۱ است اگر سرویس S_i روی گره N_j قرار بگیرد؛ در غیر این صورت، مقدار آن صفر خواهد بود. برای محاسبه مقدار جریمه ناشی از نقض مهلت ابتدا باید زمان پاسخ هر سرویس محاسبه گردد. مقدار زمان پاسخ سرویس S_i را می‌توان توسط رابطه ۱ به‌دست آورد.

$$S_i^{resp} = t_i^{in} + t_i^{exe} + t_i^{out} \qquad (۱)$$

که در آن مقدار پارامتر t_i^{in} از مجموع سه تأخیر ارتباطی زیر قابل محاسبه است:

- تأخیر از دستگاه‌های اینترنت اشیا تا گره دروازه ($d_i^{D\to G}$).
- تأخیر از گره دروازه تا گره مدیر منابع در کلونی که سرویس S_i در آن جایابی شده است ($d_i^{G\to FRM}$).
- تأخیر از گره مدیر منابع تا گره رایانشی درون کلونی که میزبان سرویس است ($d_i^{FRM\to N_j}$).

به عبارت دیگر، مقدار پارامتر t_i^{in} طبق رابطه ۲ به‌دست می‌آید.

$$t_i^{in} = d_i^{D\to G} + d_i^{G\to FRM} + d_i^{FRM\to N_j} \qquad (۲)$$

مقدار $d_i^{D\to G}$ معمولاً بسیار ناچیز و قابل چشم‌پوشی است. مقدار $d_i^{G\to FRM}$ از مجموع تأخیر انتشار و تأخیر انتقال از گره دروازه تا گره مدیر منابع کلونی که سرویس S_i روی آن جایابی‌شده محاسبه می‌گردد. هرچه کلونی میزبان سرویس به دروازه دستگاه اینترنت اشیا نزدیک‌تر باشد مدت‌زمان تأخیر، کمتر خواهد بود. شایان ذکر است چنانچه سرویسی روی سرورهای مرکز داده ابری جایابی شده باشد در رابطه (۲) پارامتر $d_i^{G\to FRM}$ با پارامتر $d_i^{G\to C}$ جایگزین می‌گردد. پارامتر $d_i^{G\to C}$ نشان‌دهنده مجموع تأخیر انتشار و تأخیر انتقال از گره دروازه تا مرکز داده ابر است. مقدار پارامتر سوم، یعنی $d_i^{FRM\to N_j}$، با توجه به اینکه سرویس S_i روی کدام گره رایانشی در کلونی قرار دارد و فاصله آن گره تا گره مدیر منابع چند گام است توسط رابطه (۳) محاسبه می‌شود. شایان ذکر است که تبادل اطلاعات بین گره‌ها در یک کلونی به‌صورت چند گامه می‌باشد.

$$d_i^{FRM\to N_j} = \sum_{k=1}^{N_j^h \times x_{ij}} (\delta_i^k + \gamma_i^k) \qquad (۳)$$

زمان لازم برای اجرای یک درخواست از نوع سرویس S_i، یعنی t_i^{exe}، از رابطه ۴ به‌دست می‌آید.

$$t_i^{exe} = \sum_{j=1}^{m} \frac{S_i^s}{N_{ij}^{cc}} \times x_{ij} \quad (۴)$$

که در آن N_{ij}^{cc} بیانگر ظرفیت محاسباتی تخصیص‌یافته توسط گره N_j به سرویس S_i می‌باشد که توسط رابطه ۵ محاسبه می‌شود.

$$N_{ij}^{cc} = S_i^c \times \frac{N_j^{cc}}{N_j^c} \quad (۵)$$

در رابطه (۱)، مدت‌زمان لازم برای برگشت پاسخ به دستگاه اینترنت اشیا متقاضی، t_i^{out}، دقیقاً همانند پارامتر t_i^{in} محاسبه می‌گردد. شایان ذکر است به دلیل اینکه اندازه فایل خروجی معمولاً از فایل ورودی کوچک‌تر است، تأخیر انتقال در t_i^{out} کمتر از t_i^{in} می‌باشد.

میزان تخطی از مهلت زمانی تعیین‌شده توسط دستگاه‌های اینترنت اشیا برای هر درخواست سرویس مانند S_i به‌صورت رابطه (۶) محاسبه می‌شود.

$$S_i^{viol} = \begin{cases} S_i^{resp} - S_i^d & \text{if } S_i^{resp} > S_i^d \\ 0 & o.w. \end{cases} \quad (۶)$$

یکی از اهداف اصلی این کار، قراردادن سرویس‌ها روی محیط سلسله‌مراتبی مه-ابر به گونه‌ای است که مقدار زمان نقض مهلت سرویس‌ها کاهش یابد. رابطه ۷ مقدار این پارامتر را به ما می‌دهد.

$$\textbf{Min} \sum_{i=1}^{n} S_i^{viol} \quad (۷)$$

۳-۲-۲ هدررفت منابع

هدررفت منابع، نشان‌دهنده مقدار منابع ازدست‌رفته در یک گره فعال است که به هیچ سرویسی تخصیص پیدا نکرده است. در این کار، ظرفیت CPU و حافظه اصلی (RAM) به‌عنوان دو منبع مهم برای هر گره رایانشی درنظر گرفته شده است. برای محاسبه میزان هدررفت منابع، ابتدا مقدار CPU و حافظه استفاده‌شده برای هر گره را نرمال‌سازی می‌کنیم. در اینجا منظور از نرمال‌سازی مقدار CPU استفاده‌شده برای گره N_j عبارت است از مجموع CPU تخصیص‌یافته به سرویس‌های جایابی‌شده روی گره N_j تقسیم بر ظرفیت CPU آن گره که توسط رابطه ۸ محاسبه می‌شود.

$$U_j^{cpu} = \frac{1}{N_j^{cc}} \sum_{i=1}^{n} N_{ij}^{cc} \times x_{ij} \quad (۸)$$

به همین ترتیب می‌توان توسط رابطه ۹ به مقدار نرمال‌سازی‌شده حافظه اصلی دست پیدا کرد.

$$U_j^{ram} = \frac{1}{N_j^{r}} \sum_{i=1}^{n} \mathrm{S}_i^{\mathrm{r}} \times x_{ij} \quad (۹)$$

پس از محاسبه میزان منابع CPU و حافظه استفاده‌شده در هر گره رایانشی و نرمال‌سازی آنها می‌توان میزان هدررفت منابع برای هر گره فعال N_j را توسط روابط (۱۰) و (۱۱) محاسبه کرد.

$$W_j^{cpu} = 1 - U_j^{cpu} \quad (۱۰)$$

$$W_j^{ram} = 1 - U_j^{ram} \quad (۱۱)$$

یکی دیگر از اهداف اصلی این تحقیق، کمینه‌سازی مقدار هدررفت منابع محاسباتی گره‌های رایانشی فعال است که در رابطه ۱۲ آمده است.

$$\textbf{Min} \sum_{j=1}^{m} (k_1 \times W_j^{cpu} + k_2 \times W_j^{ram}) \quad (۱۲)$$

در رابطه (۱۲)، پارامترهای k_1 و k_2 اعداد طبیعی و مثبت هستند که مجموع آنها برابر ۱ است و از آنها برای تعیین میزان اهمیت منابع استفاده می‌شود. چانچه مقدار k_1 بزرگ‌تر از k_2 باشد بدان معنی است که هدررفت CPU نسبت به هدررفت حافظه دارای اهمیت بیشتری است و در صورتی که مقدار k_1 کوچک‌تر از k_2 باشد یعنی هدررفت CPU نسبت به حافظه دارای اهمیت کمتری است. در این کار، مقادیر این دو پارامتر برابر ۰.۵ درنظر گرفته شده است.

۳-۲-۳ هزینه سرویس

برای محاسبه هزینه سرویس S_i، هزینه استفاده از دو منبع مهم شامل CPU و حافظه درنظر گرفته شده است. همانند کار [۴۳]، از رابطه ۱۳ برای محاسبه هزینه سرویس استفاده می‌شود.

$$C_i = C_i^{cpu} + C_i^{ram} \quad (۱۳)$$

که در آن C_i^{cpu} بیانگر هزینه پردازش درخواست‌های سرویس i روی گره-های رایانشی است و توسط رابطه ۱۴ قابل محاسبه است.

$$C_{CPU} = \sum_{i=1}^{m} \sum_{j=1}^{n} (\varphi_c^{cpu} \times S_i^c \times x_{ij}) \quad (۱۴)$$

جایی که φ_c^{cpu} بیانگر هزینه پردازش یا اجاره هر واحد CPU برحسب دلار در ساعت است.

هزینه حافظه مصرفی C_i^{ram} بستگی به مقدار حافظه موردنیاز سرویس i دارد و توسط رابطه ۱۵ تعریف می‌گردد.

$$C_{RAM} = \sum_{i=1}^{m} \sum_{j=1}^{n} (\varphi_c^{ram} \times S_i^r \times x_{ij}) \quad (۱۵)$$

که در آن φ_c^{ram} بیانگر هزینه اجاره یک گیگابایت حافظه برحسب دلار در ساعت است.

هدف سوم این تحقیق، کاهش هزینه خدمات است که از رابطه ۱۶ به‌دست می‌آید.

$$\textbf{Min} \sum_{i=1}^{n} C_i \qquad (۱۶)$$

مدل بهینه‌سازی ارائه‌شده شامل چهار محدودیت زیر است.

$$\sum_{i=1}^{n} S_i^c \times x_{ij} \le N_j^c \quad \forall N_j \in N \qquad (۱۷)$$

$$\sum_{i=1}^{n} S_i^r \times x_{ij} \le N_j^r \quad \forall N_j \in N \qquad (۱۸)$$

$$\sum_{j=1}^{m} x_{ij} = 1 \quad \forall N_j \in N \qquad (۱۹)$$

$$x_{ij} \in \{0,1\} \qquad (۲۰)$$

محدودیت‌های (۱۷) و (۱۸) بیان می‌کنند که تعداد CPUها و مقدار حافظه تخصیص‌یافته به سرویس‌های جایابی‌شده روی هر گره نباید از ظرفیت آن گره تجاوز کند. محدویت (۱۹) تضمین می‌دهد که هر سرویس تنها روی یک گره رایانشی جای داده می‌شود و محدودیت (۲۰) دامنه مقادیر متغیر تصمیم‌گیری x_{ij} را تعیین می‌کند.

۴- روش پیشنهادی

در این تحقیق، برای حل مسئله جایابی سرویس‌های اینترنت اشیا در محیط سلسله‌مراتب مه–ابر از الگوریتم ژنتیک مبتنی بر نخبه‌گرایی استفاده می‌شود که خروجی کروموزمی (راه‌حلی) است که بیانگر نگاشتی از سرویس‌ها روی گره‌های محاسباتی در محیط مه-ابر سلسله‌مراتبی درنظر گرفته شده می‌باشد.

۴-۱- الگوریتم ژنتیک

عملیات اساسی در الگوریتم ژنتیک مبتنی بر نخبه‌گرایی عبارتند از: کدینگ، عملگر انتخاب، عملگر ترکیب، عملگر جهش و محاسبه تابع برازندگی که در ادامه هریک با جزئیات توصیف می‌گردد.

۴-۱-۱- کدینگ

در این کار، طول هر کروموزوم برابر تعداد سرویس‌ها است که در آن هر ژن بیانگر مکان هر سرویس روی یک کلونی مشخص می‌باشد. شایان ذکر است که مرکز داده ابر نیز به عنوان یک کلونی مجزا درنظر گرفته می‌شود. شکل (۲) یک کروموزم با ۸ سرویس را نشان می‌دهد که در آن، سرویس S_1 به کلونی اول، S_2 به کلونی سوم و به همین ترتیب الی آخر انتساب شده است.

S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8
۱	۳	۲	۵	۳	٤	٦	۲

شکل ۲. ساختار کروموزوم

۴-۱-۲ ایجاد جمعیت اولیه

برای ایجاد جمعیت اولیه، ابتدا برای هریک از سرویس‌ها کلونی‌های موجود در محیط مه بررسی شده و کلونی‌هایی که دارای حداقل یک گره محاسباتی باشند که بتواند نیازمندی‌های آن سرویس را تأمین کند به‌عنوان کلونی کاندید انتخاب می‌شود. در نتیجه، برای هر سرویس، یک مجموعه از کلونی‌ها تشکیل می‌گردد که می‌توانند از آن سرویس میزبانی کنند. سپس برای هر سرویس یک کلونی از مجموعه کاندید به‌صورت تصادفی انتخاب می‌شود. داخل هر کلونی، هر سرویس روی گرهی قرار می‌گیرد که حداقل هدررفت منابع را داشته باشد. در صورتی که کلونی انتخاب‌شده در محیط ابر باشد سرویس‌ها با استفاده از سیاست First Fit [۳۱] جایابی می‌شوند.

۴-۱-۳- عملگر انتخاب

در این کار ابتدا با استفاده از روش انتخاب نخبه‌گرایی تعدادی از راه‌حل‌های برتر (نخبه) انتخاب و به نسل بعدی منتقل می‌شوند. بدین منظور جمعیت بر اساس تابع برازندگی مرتب می‌شود. سپس ده درصد برتر کل جمعیت به‌عنوان کروموزوم‌های نخبه، انتخاب و به نسل بعدی منتقل می‌شوند. سپس برای انتخاب هر زوج والد، یکی به‌صورت تصادفی و دیگری با روش چرخ رولت انتخاب می‌گردد.

۴-۱-۴- عملگر ترکیب

در این کار از تکنیک ترکیب یکنواخت استفاده شده است. در ترکیب یکنواخت، ابتدا یک بردار دودویی با طول برابر با طول کروموزوم به‌عنوان بردار ماسک درنظر گرفته می‌شود و مقادیر درایه‌های آن به‌صورت تصادفی تعیین می‌گردد. همان‌طور که در شکل (۳) مشاهده می‌گردد، برای تولید فرزند اول اگر درایه بردار ماسک دارای مقدار ۱ باشد، ژن متناظر از والد ۱ و در غیر این صورت ژن متناظر از والد ۲ انتخاب می‌شود. برای تولید فرزند دوم، اگر مقدار درایه بردار ماسک برابر ۰ باشد، ژن متناظر از والد ۱ و در غیر این صورت ژن متناظر از والد ۲ انتخاب می‌گردد.

والد۱	۱	۳	۲	۵	۳	٤	٦	۲
والد۲	۲	٤	٦	۲	۵	۳	۱	٤
ماسک	۱	۰	۰	۱	۱	۰	۱	۰
فرزند ۱	۱	٤	٦	۵	۳	۳	٦	٤
فرزند ۲	۲	۳	۲	۲	۵	٤	۱	۲

شکل ۳. عملگر ترکیب

۴-۱-۵- عملگر جهش

عملگر جهش فرایندی تصادفی برای به‌هم ریختن و ایجاد اختلال در اطلاعات ژنتیکی محسوب می‌شود. برخلاف عملگر ترکیب، عملگر جهش در سطح ژن کار می‌کند؛ یعنی زمانی که ژن‌ها از رشته یا کروموزوم فعلی در رشته یا

کروموزوم جدید کپی می‌شوند، این احتمال وجود دارد که هرکدام از این ژن‌ها جهش پیدا کنند. این احتمال معمولاً مقدار بسیار کوچکی است که به آن احتمال جهش گفته می‌شود. عملگر جهش بر اساس سطح اعمال، به دو طریق قابل انجام است: جهش تک‌نقطه‌ای و جهش مقیاس وسیع. در روش اول فقط یک ژن تغییر می‌کند اما در روش دوم چندین ژن یا حتی همه ژن‌های کروموزوم، تغییر پیدا می‌کنند. در این کار از روش جهش تک‌نقطه‌ای با احتمال جهش ۵ درصد استفاده می‌شود.

۴-۱-۶- تابع برازندگی

الگوریتم ژنتیک از اصل بقای برازنده‌ترین‌ها در طبیعت برای ایجاد فرایند جستجو و متعاقباً جستجو در فضای جواب مسئله استفاده می‌کند. برای ارزیابی هریک از راه‌حل‌های کاندید و تعیین کروموزوم‌های برازنده، از تابع برازندگی استفاده می‌شود. برای تعیین راه‌حل (کروموزوم) برازنده، باید کروموزومی انتخاب شود که سه هدف اصلی این کار را کمینه کند؛ یعنی مقدار نقض مهلت، مقدار هدررفت و هزینه خدمات. تابع برازندگی برای هر راه‌حل (کروموزوم) طبق رابطه ۲۱ محاسبه می‌شود.

$$\text{Fitness Function}=\alpha \times viol_k+\beta \times cost_k +\gamma \times waste_k \quad (۲۱)$$

که در آن پارامتر $viol_k$ بیانگر مجموع نقض مهلت نرمال‌سازی‌شده برای اجرای سرویس‌ها در کروموزوم k است و به‌صورت زیر محاسبه می‌شود.

$$viol_k = \sum_{i=1}^{n} norm_{viol_i} \quad (۲۲)$$

در این رابطه، $norm_{viol_i}$ بیانگر مقدار نرمال‌سازی‌شده مقدار تخطی از مهلت زمانی برای اجرای سرویس S_i در کروموزوم k می‌باشد که مقدار آن از رابطه ۲۳ به‌دست می‌آید.

$$norm_{viol_i} = \frac{viol_i - \min\left(viol_{\forall j \epsilon n}\right)}{\max\left(viol_{\forall j \epsilon n}\right) - \min\left(viol_{\forall j \epsilon n}\right)} \quad (۲۳)$$

جایی که پارامتر $viol_i$ نشان‌دهنده میزان تخطی از مهلت زمانی تعیین‌شده ناشی از اجرای سرویس S_i است و طبق رابطه (۶) محاسبه می‌گردد.

پارامترهای $\min\left(viol_{\forall j \epsilon napps}\right)$ و $\max\left(viol_{\forall j \epsilon napps}\right)$ به‌ترتیب بیانگر کمترین و بیشترین مقدار تخطی از مهلت زمانی در بین جمعیت‌های آن نسل است. در رابطه (۲۱)، پارامتر $cost_k$ بیانگر مجموع هزینه سرویس نرمال‌سازی‌شده برای اجرای سرویس‌ها در کروموزوم K است که توسط رابطه ۲۴ قابل محاسبه است.

$$cost_k = \sum_{i=1}^{n} norm_{cost_i} \quad (۲۴)$$

که در آن $norm_{cost_i}$ مقدار هزینه نرمال‌سازی‌شده است که به کمک رابطه ۲۵ محاسبه می‌گردد.

$$norm_{cost_i} = \frac{cost_i - \min\left(cost_{\forall j \epsilon n}\right)}{\max\left(cost_{\forall j \epsilon n}\right) - \min\left(cost_{\forall j \epsilon n}\right)} \quad (۲۵)$$

در نهایت، مقدار $waste_k$ در رابطه (۲۱) بیانگر مجموع هدر رفت منابع در گره‌های رایانشی فعال بر اساس کروموزوم k است که مقدار آن بر اساس رابطه ۲۶ به‌دست می‌آید.

$$waste_k = \sum_{j=1}^{m} (k_1 \times W_j^{cpu} + k_2 \times W_j^{ram}) \quad (۲۶)$$

در رابطه (۲۱)، پارامترهای α، β و γ اعداد حقیقی هستند که مجموع آنها برابر ۱ است و برای تحلیل حساسیت الگوریتم پیشنهادی مورد استفاده قرار می‌گیرند. مقدار هریک از پارامترهای α، β و γ به‌ترتیب تعیین‌کننده میزان اهمیت پارامترهای میزان تخطی از مهلت زمانی اجرای سرویس، میزان هزینه اجرای سرویس و میزان هدررفت منابع در گره‌های فعال است. برای تعیین مقادیر آنها، الگوریتم پیشنهادی روی مجموعه سرویس‌ها و گره‌ها با سناریوهای مختلف اجرا و مقدار تابع هدف محاسبه می‌شود و میانگین آنها محاسبه می‌گردد. نتایج حاصل در جدول (۳) نمایش داده شده است. همان‌طور که داده‌ها نشان می‌دهد بهترین حالت در سناریوی اول رخ داده است؛ جایی که مقدار α=۰.۵ و β=۰.۲۵ و γ=۰.۲۵ درنظر گرفته شده است.

جدول ۳. تحلیل حساسیت پارامترهای الگوریتم پیشنهادی

میانگین اجراها				
$\alpha = 0.34$ $\beta = 0.33$ $\gamma = 0.33$	$\alpha = 0.25$ $\beta = 0.25$ $\gamma = 0.5$	$\alpha = 0.25$ $\beta = 0.5$ $\gamma = 0.25$	$\alpha = 0.5$ $\beta = 0.25$ $\gamma = 0.25$	
۳.۷۴	۴	۶.۱۹	۱.۶۶	نقض مهلت
۱۰.۶۸	۱۲.۲۸	۸.۰۸	۱۱.۹۵	هزینه
۳.۲۹	۲.۴۸	۳.۷۳	۳.۸	هدر رفت منابع
۵.۸۸	۵.۶۶	۶.۵۲	**۴.۷۳**	تابع برازندگی

۴-۲- الگوریتم پیشنهادی

شبه کد الگوریتم پیشنهادی برای حل مسئله جایابی سرویس‌های اینترنت اشیا توسط **الگوریتم ۱** نمایش داده شده است. الگوریتم لیست سرویس‌ها S و لیست کلونی‌ها C را به‌عنوان ورودی دریافت می‌کند و در خروجی نگاشت سرویس‌ها به کلونی‌ها را به ما می‌دهد. ابتدا جمعیت اولیه برابر ۱۰۰، تعداد تکرار برابر ۲۰۰، نرخ نخبه‌گرایی ۱۰ درصد و احتمال جهش برابر ۵ درصد درنظر گرفته شده است. سپس جمعیت اولیه به‌صورت تصادفی تولید می‌شود (خط ۲). در ادامه، مقدار تابع برازندگی برای هریک از کروموزوم‌ها بر اساس رابطه (۲۱) محاسبه می‌گردد (خط ۳). در خط ۴، جمعیت اولیه براساس مقدار تابع برازندگی به‌صورت صعودی مرتب می‌شوند. سپس ۱۰ درصد اول انتخاب و در لیست نخبه‌ها قرار می‌گیرند (خط ۵). در خط ۶ بهترین کروموزم انتخاب و در AL قرار می‌گیرد. در ادامه، ابتدا کروموزم‌های نخبه به جمعیت جدید اضافه می‌شوند (خط ۷). برای تکمیل بقیه کروموزم-های نسل جدید، هر بار یک والد به‌صورت تصادفی و والد دیگر با استفاده از روش چرخ رولت انتخاب می‌شود و توسط عملگر ترکیب دو فرزند تولید می‌گردد و به لیست جدید اضافه می‌شوند (خطوط ۱۰-۱۴). برای هریک از فرزندان موجود در لیست با احتمال ۵ درصد عمل جهش نقطه‌ای انجام می-شود (خطوط ۱۵-۱۷). پس از این مرحله، جمعیت جدید و جمعیت پیشین با هم ادغام می‌شوند، خط ۱۸، و برای هریک از افراد جمعیت، حاصل مقدار تابع برازندگی، محاسبه و در بردار برازندگی قرار می‌گیرد (خط ۱۹).

الگوریتم ۱. جایابی سرویس‌های اینترنت اشیا با استفاده از الگوریتم ژنتیک مبتنی بر نخبه‌گرایی

Input: S, C
Output: AL← []

initialize: num_population ← 100, num_generation ←200, Elitism_rate←0.1 Mutation_rate (P_m)← 0.05, new_population← [], cross_list←[], mut_list ←[], Elite_list ← []
population ← Generate Population randomly
fitness ← Calculate Fitness (Population) using eq. (21)
sort (population, fitness)
Elite_list ← select (Population, Elitism_rate)
AL ← the best chromosome
new_Population ← new_Population U Elite_list
remined_Population ← Population- Elite_list
while num_generation **do**
    for all chromosome ∈ population **do**
        parent1 ← select population randomly (population)
        parent2 ← select population roulette wheel(population)
        child1, child2 ← crossover operation (parent1, parent2)
        cross_list ← cross_list U (child1, child2)
    end for
    for all chromosome ∈ cross_list **do**
        newchild ← mutation operation (cross_list, P_m)
        new_population ← new_population U newchild
    end for
    new_ population ← new_population U remined_population
    fitness ← Calculate Fitness(new_Population) using Eq. (21)
    sort (new_population, fitness)
    population← select num_population (new_population)
    Elite_list ← select (Population, Elitism_rate)
    new_population ← [], cross_list←[]
    new_Population ← new_Population U Elite_list
    AL ← the best chromosome
    remined_Population ← Population- Elite_list
end while
return AL

سپس کل جمعیت براساس مقدار برازندگی، مرتب (خط ۲۰) و به تعداد جمعیت (به‌جز ۱۰ درصد نخبه) به‌ترتیب برازندگی انتخاب می‌شوند (خط ۲۱). در ادامه، ۱۰ درصد جمعیت جدید در لیست نخبه‌ها قرار می‌گیرد (۲۲) و برازنده‌ترین کروموزوم در AL ذخیره می‌شود (خط ۲۵). فرایند به تعداد نسل‌ها تکرار می‌شود و در نهایت لیست نگاشت سرویس‌های تخصیص‌یافته به کلونی‌ها، AL، برگشت داده می‌شود.

۵- ارزیابی کارایی

در این بخش ابتدا سیاست‌های مقایسه‌ای توصیف می‌گردد. سپس تنظیمات پارامترهای شبیه‌سازی ارائه می‌شود. پس از آن، معیارهای ارزیابی بیان خواهد شد. در نهایت نتایج حاصل از شبیه‌سازی ارائه می‌شود.

۵-۱- سیاست‌های مقایسه‌ای

به‌منظور ارزیابی روش پیشنهادی، آن را با استراتژی‌های زیر مقایسه کرده‌ایم:

استراتژی First Fit: در این روش [۳۹]، برای هریک از سرویس‌های ارسال‌شده به کلونی، گره مدیر منابع آن کلونی به‌ترتیب گره‌های درون کلونی، بررسی می‌شود و اولین گره‌ای که نیازمندی‌های آن سرویس را تأمین کند انتخاب می‌گردد. چنانچه درون کلونی، گره‌ای پیدا نشود که نیازمندی‌های سرویس را تأمین کند سرویس به کلونی سطح بالاتر ارسال می‌گردد.

استراتژی Edge Affinity: در این کار [۱۷]، ابتدا ویژگی‌های سرویس‌ها از قبیل مهلت‌زمانی تعیین‌شده توسط کاربر، مقدار داده‌هایی که باید پردازش شوند و نرخ درخواست دستگاه‌های اینترنت اشیا برای سرویس نرمال‌سازی می‌گردد و سپس سرویس‌ها براساس یک روش مرتب‌سازی غیرغالب، طبقه‌بندی می‌شوند، سپس گره مدیر منابع درون کلونی برای هریک از سرویس‌های دریافتی بر اساس اولویت آنها گره‌های موجود در کلونی را بررسی می‌کند. چنانچه گره رایانشی پیدا شود که نیازمندی‌های سرویس را تأمین کند عمل جایابی سرویس اجرا می‌شود و منابع گره به‌روزرسانی می‌گردد. در غیر این صورت سرویس به کلونی دیگر ارسال می‌گردد.

استراتژی Fuzzy Approach: در این روش [۱۴]، ابتدا سرویس‌ها بر اساس سیستم فازی اولویت‌بندی می‌شوند به‌گونه‌ای که سرویس‌های حیاتی با تعداد دستورالعمل پایین و اندازه فایل ورودی بالا بیشترین اهمیت را خواهند داشت. سپس داخل هر کلونی سرویس‌ها بر اساس یک الگوریتم ابتکاری با هدف بهینه‌سازی بهره‌وری منابع و بهبود کیفیت سرویس جایابی می‌شوند.

۵-۲- تنظیمات شبیه‌سازی

محیط شبیه‌سازی با زبان C++ ایجاد شده است. تمام آزمایش‌ها روی یک کامپیوتر شخصی با پردازنده intel Core i7 3.6 GHz و حافظه 8 GB انجام شده است. مقادیر پارامترهای شبیه‌سازی در جدول (۴) آورده شده است [۱۴، ۱۷]. با توجه به معماری پیشنهادی، در آزمایش‌های انجام‌شده، تعداد گره‌های رایانشی در هر کلونی، یک عدد تصادفی بین ۴ تا ۷ فرض شده است. به‌منظور ارائه نتایج با اطمینان بالا، هر آزمایش ۱۰ بار تکرار شده و در نهایت میانگین، بیشترین و کمترین مقدار گزارش گردیده است.

جدول ۴. پارامترهای شبیه‌سازی

پارامتر	مقدار
گره‌های سطح اول محیط مه	
ظرفیت CPU هر گره	[۸۰۰۰ – ۳۰۰۰] MIPS
تعداد CPU هر گره	[۲-۵] CPUs
مقدار حافظه هر گره	[۲-۸] GB
گره‌های سطح دوم محیط مه	
ظرفیت CPU هر گره	[۱۰۰۰۰ – ۴۰۰۰] MIPS
تعداد CPU هر گره	[۳-۷] CPUs
مقدار حافظه هر گره	[۴-۱۲] GB
گره‌های سطح سوم محیط مه	
ظرفیت CPU هر گره	[۱۲۰۰۰– ۶۰۰۰] MIPS
تعداد CPU هر گره	[۵-۹] CPUs
مقدار حافظه هر گره	[۸-۱۶] GB
ماشین‌های مجازی در مرکز داده ابر	
ظرفیت CPU هر ماشین مجازی	[۴۰۰۰ – ۲۰۰۰] MIPS
تعداد CPU هر ماشین مجازی	[۸-۳۲] CPUs
مقدار حافظه هر ماشین مجازی	[۱۶-۶۴] GB
سرویس	
تعداد CPU مورد نیاز	[۲-۵] CPUs
مقدار حافظه مورد نیاز	[۲-۸] GB
تعداد دستورالعمل‌ها	[۳۰۰–۳۰۰۰] MI
نرخ درخواست	[۱-۸] request/second
مهلت زمانی سرویس	[۰.۳–۱.۲] second
اندازه داده‌های ورودی	[۰.۳-۱.۵] MB
اندازه داده‌های خروجی	[۰.۱-۱] MB

۵-۳- معیارهای کارایی

برای ارزیابی سیاست پیشنهادی در مقایسه با سایر استراتژی‌ها از معیارهای زیر استفاده شده است:

۵-۳-۱ درصد سرویس‌های قرار داده‌شده در محیط مه

این معیار برای سنجش تعداد سرویس‌های جایابی‌شده روی گره‌های رایانشی در محیط مه به‌کار می‌رود. بالابودن مقدار این معیار، بیانگر این است که بیشتر سرویس‌ها روی گره‌های رایانشی محیط مه قرار گرفته شده است. مقدار این معیار به صورت رابطه ۲۷ به‌دست می‌آید.

$$PSF = \frac{1}{n}\sum_{j=1}^{m}\sum_{i=1}^{n} x_{ij} \times 100\% \qquad (۲۷)$$

۵-۳-۲ درصد هدررفت منابع در محیط مه

یکی دیگر از معیارهایی که برای سنجش کارایی سیاست پیشنهادی به‌کار گرفته شده است درصد هدررفت منابع CPU و حافظه در گره‌های فعال محیط مه است. پایین‌بودن مقدار این معیار، نشان‌دهنده این است که از منابع موجود به‌صورت کارآمدتری استفاده شده است. درصد هدررفت CPU و حافظه، به‌ترتیب توسط رابطه‌های (۲۸) و (۲۹) قابل محاسبه است.

$$PCWF = \frac{\sum_{j=1}^{m} a_j \; W_j^{cpu}}{\sum_{j=1}^{m} a_j \; N_j^{cc}} \times 100 \qquad (۲۸)$$

$$PRWF = \frac{\sum_{j=1}^{m} a_j \; W_j^{ram}}{\sum_{j=1}^{m} a_j \; N_j^{r}} \times 100 \qquad (۲۹)$$

در رابطه‌های بالا، پارامتر a_j یک متغیر دودویی است. چنانچه گره N_j یک گره فعال باشد مقدار پارامتر a_j برابر ۱ است در غیر این صورت مقدار آن برابر صفر خواهد بود.

۵-۳-۳- میانگین جریمه ناشی از نقض مهلت

یکی از معیارهای کیفیت خدمات، مقدار میانگین جریمه ناشی از نقض مهلت زمانی سرویس‌ها است. هر اندازه این مقدار پایین‌تر باشد کیفیت خدمات بهتر خواهد بود. رابطه ۳۰ مقدار این معیار را به ما می‌دهد.

$$AP = \frac{\sum_{i=1}^{n}\sum_{k=1}^{A_i^{\lambda}} A_i^{viol}}{\sum_{i=1}^{n} A_i^{\lambda}} \qquad (۳۰)$$

۵-۳-۴- میانگین هزینه اجرای سرویس‌ها

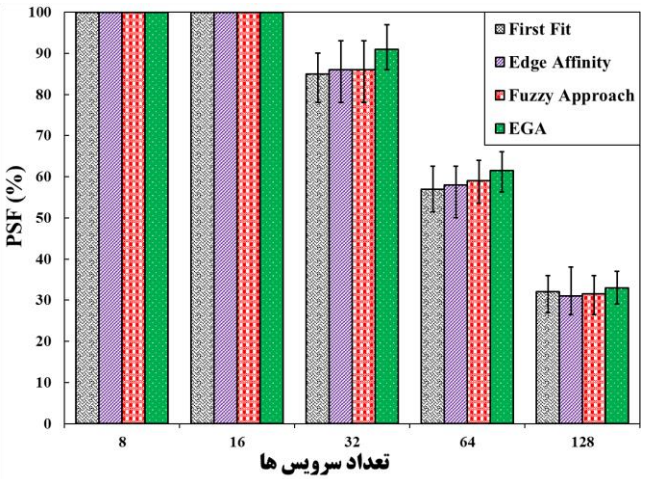
کاهش میانگین هزینه کل اجرای سرویس‌ها یکی دیگر از معیارهای کلیدی برای ارزیابی یک استراتژی جایابی سرویس است که مقدار آن توسط رابطه ۳۱ بدست می‌آید.

$$AC = \frac{\sum_{i=1}^{n} cost_i}{n} \qquad (۳۱)$$

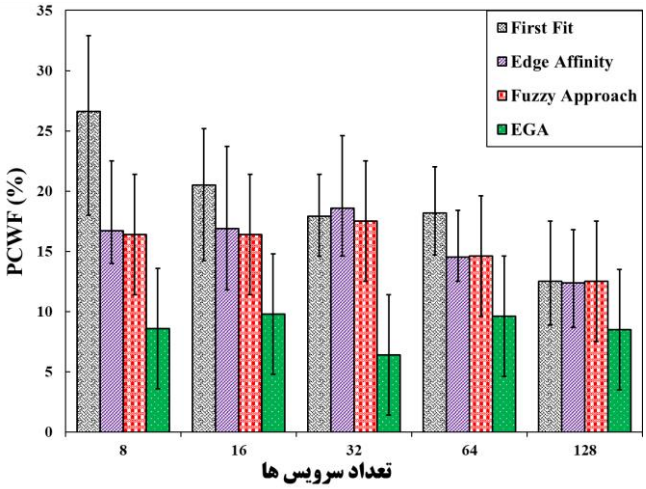
۵-۴- تحلیل نتایج

در این بخش به ارائه نتایج حاصل از آزمایش‌های انجام‌شده می‌پردازیم. در نمودارها الگوریتم ژنتیک مبتنی بر نخبه‌گرایی پیشنهادی را با EGA نشان داده‌ایم. شکل (۴) نتایج مربوط به معیار PSF را نشان می‌دهد. همان‌طور که می‌توان مشاهده کرد وقتی تعداد سرویس‌ها کم باشد همه روش‌ها قادر خواهند بود که تمام سرویس‌ها را در محیط مه جای دهند. اما با افزایش تعداد سرویس‌ها، روش پیشنهادی عملکرد بهتری از خود نشان می‌دهد. دلیل این امر، استفاده مؤثرتر از منابع مه است که روش پیشنهادی به‌خوبی توانسته است به آن دست یابد. اهداف کاهش هدررفت منابع و کمینه‌سازی

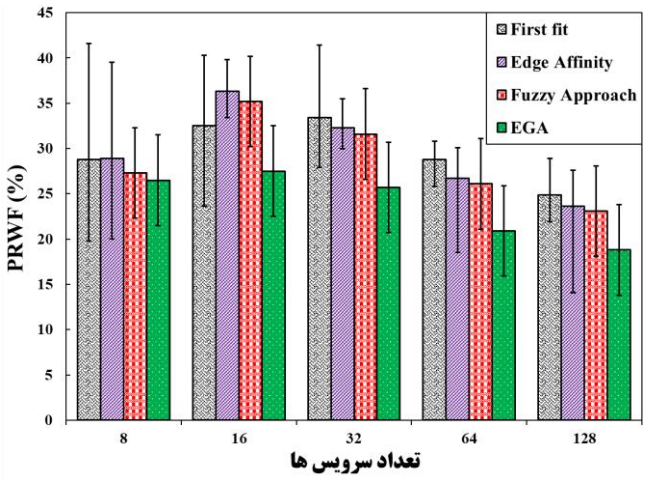
جریمه ناشی از نقض مهلت سرویس‌ها به‌صورت توأم به بهبود این معیار کمک کرده است.



شکل ۴. درصد سرویس‌های جای‌داده‌شده در محیط مه



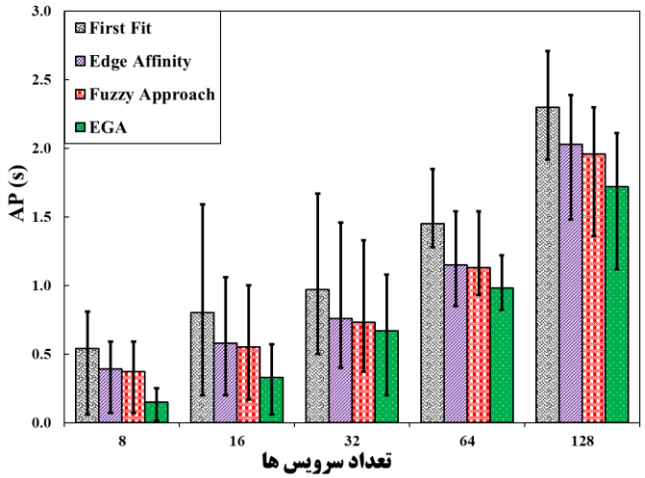
شکل ۵. میانگین هدررفت CPU در محیط مه



شکل ۶. میانگین هدررفت حافظه در محیط مه

شکل‌های (۵) و (۶) به‌ترتیب نتایج مربوط به درصد هدررفت CPU و حافظه برای گره‌های فعال در محیط مه را نشان می‌دهند. نمودارها بیانگر این هستند که روش پیشنهادی در مقایسه با سایر روش‌ها، میزان هدررفت منابع را به‌طور چشمگیری کاهش می‌دهد. به‌عنوان یکی از دلایل اصلی این بهبود می‌توان به عملکرد الگوریتم در یافتن جواب‌های نزدیک به بهینه اشاره کرد که در ذات الگوریتم ژنتیک مبتنی بر نخبه‌گرایی نهفته است. همچنین، تابع برازش پیشنهادی به‌گونه‌ای طراحی شده است که هر اندازه یک راه‌حل بتواند از منابع مه بهتر استفاده کند امتیاز بیشتری به‌دست خواهد آورد؛ چون هم جریمه ناشی از نقض مهلت و هم هدررفت منابع کاهش می‌یابد.

نتایج مربوط میانگین جریمه، ناشی از نقض مهلت سرویس‌ها در شکل (۷) به نمایش درآمده است. دوباره نتایج حاکی از عملکرد بهتر روش پیشنهادی در مقایسه با رقیبانش است. با توجه به اینکه در تابع برازش اهمیت بیشتری به این معیار داده شده است، الگوریتم به‌کار گرفته‌شده به سمت راه‌حل‌هایی هدایت خواهد شد که زمان پاسخ سرویس‌ها را تا حد امکان کاهش دهد و از نقض مهلت تعیین‌شده خودداری کند. از نظر این معیار، Fuzzy Approach و Edge Affinity عملکرد تقریباً مشابهی از خود نشان داده‌اند.

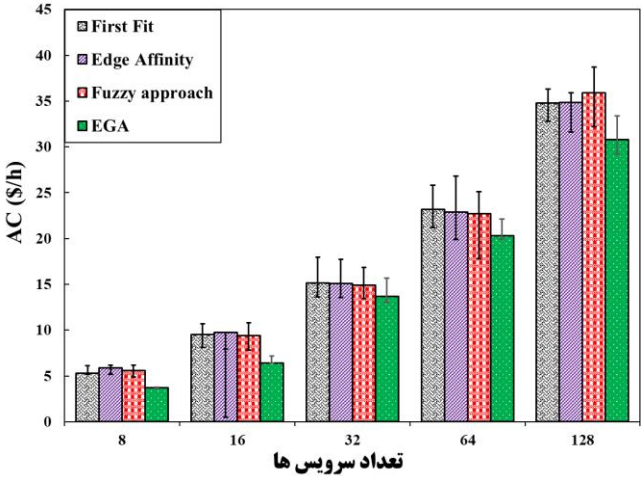


شکل ۷. میانگین نقض مهلت زمانی اجرای درخواست

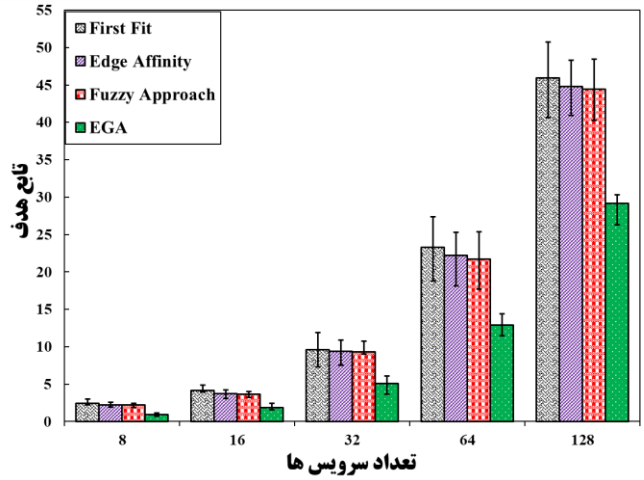
شکل (۸) نتایج مربوط به معیار میانگین هزینه را به تصویر کشیده است. همان‌طور که از نتایج این نمودار قابل مشاهده است در مقایسه با سایر روش‌ها، روش پیشنهادی هزینه، جایگذاری کمتری ارائه می‌دهد. دلیل اصلی این برتری، به سیاست جایابی الگوریتم برمی‌گردد که در آن سعی می‌شود سرویس‌هایی که منابع بیشتری نیاز دارند در محیط ابر جای داده شوند تا هزینه استقرار آنها کاهش یابد. شایان ذکر است که سایر روش‌ها بدون توجه به این معیار، سرویس‌ها را جایابی می‌کنند.

نتایج آخرین آزمایش در شکل (۹) نشان داده شده است جایی که مقدار تابع هدف روش‌های موردمقایسه در کنار هم آمده است. با افزایش تعداد

سرویس‌ها، مقدار تابع هدف برای همه روش‌ها افزایش پیدا کرده است. در اینجا نیز برتری عملکرد روش پیشنهادی، کاملاً محسوس و چشمگیر است. بر اساس این معیار، مقدار متوسط بهبود نسبت به سایر روش‌ها بین ۴۲.۳ تا ۴۹.۸ درصد است. دلیل اصلی این برتری، تنظیم مناسب پارامترهای مدل و عملکرد مطلوب الگوریتم پیشنهادی در جستجوی فضای راه‌حل‌ها است. در حالی که سایر الگوریتم‌ها در بهینه محلی قرار می‌گیرند که در نهایت منجر به راه‌حل‌های غیربهینه می‌شود، الگوریتم ژنتیک مبتنی بر نخبه‌گرایی می‌تواند با حفظ بهترین راه‌حل‌ها از نسلی به نسل دیگر و ارتقای تنوع در جمعیت، به جلوگیری از این مشکل کمک کند، که در نهایت به عملکرد کلی بهتر و احتمال بیشتر برای یافتن بهینه سراسری منتهی می‌شود.



شکل ۸. میانگین هزینه اجرای سرویس‌ها



شکل ۹. تابع هدف

۶- بحث

در این مقاله ما از مدل جمع وزنی به‌جای الگوریتم‌های چندهدفه استفاده کردیم. دلایل اصلی ما برای این انتخاب شامل موارد زیر است. مزیت اصلی مدل جمع وزنی نسبت به الگوریتم‌های چندهدفه، سادگی و سهولت در پیاده‌سازی آن است. مدل جمع وزنی، یک رویکرد ساده است که فقط به تعیین وزن‌های اختصاص داده‌شده به هر هدف نیاز دارد. در مقابل، الگوریتم‌های چندهدفه می‌توانند پیچیده‌تر باشند و به منابع محاسباتی بیشتری برای حل مسئله بهینه‌سازی نیاز دارند. مزیت دیگر مدل جمع وزنی این است که می‌تواند یک راه‌حل بهینه واحد ارائه دهد در حالی که الگوریتم‌های چندهدفه معمولاً مجموعه‌ای از راه‌حل‌های غیرغالب را تولید می‌کنند. در برخی موارد، یک راه‌حل بهینه واحد ترجیح داده می‌شود، به‌خصوص زمانی که اهداف، متناقض نباشند.

شایان ذکر است که اهداف درنظر گرفته‌شده در این مقاله، لزوماً متناقض نیستند. برای مثال، اهداف نقض مهلت و هدررفت منابع می‌توانند در یک راستا باشند.

۷- نتیجه‌گیری و کار آینده

در این مقاله، مسئله جایابی سرویس‌های اینترنت اشیا در محیط مه-ابر سلسه‌مراتبی، مطالعه و بررسی شدند. ابتدا، یک مدل بهینه‌سازی برنامه‌ریزی خطی عدد صحیح مختلط برای این مسئله ارائه شد که در آن، هدف کمینه‌سازی توأم جریمه ناشی از نقض مهلت، میزان هدررفت منابع محاسباتی و هزینه استقرار سرویس‌ها است. سپس از الگوریتم ژنتیک مبتنی بر نخبه‌گرایی به‌منظور پیداکردن یک راه‌حل کارآمد برای مدل ارائه‌شده استفاده گردید. به‌منظور ارزیابی عملکرد روش پیشنهادی، آن را با روش‌های First Fit، Edge Affinity و Fuzzy Approach مقایسه کردیم. نتایج حاصل از آزمایش‌های انجام‌شده تأیید کرد که روش پیشنهادی از نظر معیارهایی همچون بهره‌وری منابع مه، جریمه ناشی از نقض مهلت و هزینه استقرار سرویس‌ها عملکرد به مراتب بهتری نسبت به رقیبان از خود نشان می‌دهد. در آینده از جهات مختلف می‌توان این کار را بهبود داد. اول اینکه می‌توان مدل را به گونه‌ای توسعه داد که مصرف انرژی نیز در آن دیده شود. با توجه به اینکه ارائه‌دهندگان خدمات ابری و مهی، در حال استفاده از منابع انرژی تجدیدپذیر هستند این مقوله می‌تواند از جایگاه ویژه‌ای در کارهای آینده برخوردار باشد. دوم اینکه روش پیشنهادی را می‌توان به گونه‌ای توسعه داد که بتوان از آن در محیط‌های رایانش بدون سرور نیز بهره گرفت. این کار نیازمند توجه ویژه‌ای به معماری رایانش بدون سرور و استفاده از مفهوم تابع به‌جای سرویس است.

مراجع

[1] M. F. Mehmandar, S. Jabbehdari, and H. H. S. Javadi, A dynamic fog services provisioning approach for IoT applications. *International Journal of Communication Systems*, *33*(14), e4541, 2020.

[2] H. T. T. Binh, T. T. Anh, D. B. Son, P. A. Duc, and B. M. Nguyen, An evolutionary algorithm for solving task scheduling problem in cloud-fog computing environment. In *Proceedings of the 9th International Symposium on Information and Communication Technology*, pp. 397-404, 2018.

[3] B. Zhang, N. Mor, J. Kolb, D. S. Chan, K. Lutz, E. Allman, J. Wawrzynek, E. Lee, and J. Kubiatowicz, The Cloud is not enough: saving iot from the cloud. In *7th USENIX Workshop on Hot Topics in Cloud Computing (HotCloud 15)*, 2015.

[4] A. Yousefpour, C. Fung, T. Nguyen, K. Kadiyala, F. Jalali, A. Niakanlahiji, J. Kong, and J. P. Jue, All one needs to know about fog computing and related edge computing paradigms: A complete survey. *Journal of Systems Architecture*, *98*, 289-330, 2019.

[5] Z. Yu, Y. Gong, S. Gong, and Y. Guo, Joint task offloading and resource allocation in UAV-enabled mobile edge computing. *IEEE Internet of Things Journal*, *7*(4), 3147-3159, 2020.

[6] B. Ravandi, and I. Papapanagiotou, A self-learning scheduling in cloud software defined block storage. In *2017 IEEE 10th International Conference on Cloud Computing (CLOUD)*, pp. 415-422, 2017.

[7] S. Omer, S. Azizi, M. Shojafar, and R. Tafazolli, "A priority, power and traffic-aware virtual machine placement of IoT applications in cloud data centers," *Journal of Systems Architecture*, vol. 115, p. 101996, 2021.

[8] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, Fog computing and its role in the internet of things. In *Proceedings of the first edition of the MCC workshop on Mobile cloud computing*, pp. 13-16, 2012.

[9] R. Mahmud, R. Kotagiri, and R. Buyya, Fog computing: A taxonomy, survey and future directions. *Internet of Everything: Algorithms, Methodologies, Technologies and Perspectives*, pp. 103-130, 2018.

[10] S. Azizi, F. Khosroabadi, and M. Shojafar, A priority-based service placement policy for fog-cloud computing systems. *Computational Methods for Differential Equations*, *7*(4), 521-534, 2019.

[11] M. Taneja, and A. Davy, Resource aware placement of IoT application modules in Fog-Cloud Computing Paradigm. In *2017 IFIP/IEEE Symposium on Integrated Network and Service Management*, pp. 1222-1228, 2017.

[12] R. Mahmud, K. Ramamohanarao, R. Buyya, Application management in fog computing environments: A taxonomy, review and future directions. *ACM Computing Surveys (CSUR)*, *53*(4), 1-43, 2020.

[13] P. Farzin, S. Azizi, M. Shojafar, O. Rana, and M. Singhal, FLEX: a platform for scalable service placement in multi-fog and multi-cloud environments. In *Proceedings of the 2022 Australasian Computer Science Week*, pp. 106-114, 2022.

[14] F. Tavousi, S. Azizi, and A. Ghaderzadeh, A fuzzy approach for optimal placement of IoT applications in fog-cloud computing. *Cluster Computing*, 25, 303-320, 2022.

[15] H. Sami, A. Mourad, H. Otrok, and J. Bentahar, Demand-driven deep reinforcement learning for scalable fog and service placement. *IEEE Transactions on Services Computing*, *15*(5), 2671-2684, 2021.

[16] H. O. Hassan, S. Azizi, and M. Shojafar, Priority, network and energy-aware placement of IoT-based application services in fog-cloud environments. *IET communications*, 14(13), 2117-2129, 2020.

[17] R. Mahmud, K. Ramamohanarao, and R. Buyya, Edge affinity-based management of applications in fog computing environments. In *Proceedings of the 12th IEEE/ACM International Conference on Utility and Cloud Computing*, pp. 61-70, 2019.

[18] S. Azizi, M. Shojafar, P. Farzin, and J. Dogani, DCSP: A delay and cost-aware service placement and load distribution algorithm for IoT-based fog networks. *Computer Communications*, 215, 9-20, 2024.

[19] F. Hoseiny, S. Azizi, M. Shojafar, F. Ahmadiazar, and R. Tafazolli, PGA: a priority-aware genetic algorithm for task scheduling in heterogeneous fog-cloud computing. In *IEEE INFOCOM 2021-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pp. 1-6, 2021.

[20] C. Guerrero, I. Lera, and C. Juiz, A lightweight decentralized service placement policy for performance optimization in fog computing. *Journal of Ambient Intelligence and Humanized Computing*, *10*, 2435-2452, 2019.

[21] A. Yousefpour, A. Patil, G. Ishigaki, I. Kim, X. Wang, H. C. Cankaya, Q. Zhang, W. Xie, and J. P. Jue, Fogplan: A lightweight qos-aware dynamic fog service provisioning framework. *IEEE Internet of Things Journal*, *6*(3), 5080-5096, 2019.

[22] M. Sriraghavendra, P. Chawla, H. Wu, S. S. Gill, and R. Buyya, DoSP: A deadline-aware dynamic service placement algorithm for workflow-oriented IoT applications in fog-cloud computing environments. *Energy Conservation Solutions for Fog-Edge Computing Paradigms*, 21-47, 2022.

[23] D. Zhao, Q. Zou, and M. Boshkani Zadeh, A QoS-Aware IoT Service Placement Mechanism in Fog Computing Based on Open-Source Development Model. *Journal of Grid Computing*, *20*(2), 1-29, 2022.

[24] N. Mehran, D. Kimovski, and R. Prodan, MAPO: a multi-objective model for IoT application placement in a fog environment. In *Proceedings of the 9th International Conference on the Internet of Things*, pp. 1-8, 2019.

[25] M. Q. Tran, D. T. Nguyen, V. A. Le, D. H. Nguyen, and T. V. Pham, Task placement on fog computing made efficient for IoT application provision. *Wireless Communications and Mobile Computing*, 1-17, 2019.

[26] S. Javanmardi, M. Shojafar, V. Persico, and A. Pescapè, FPFTS: A joint fuzzy particle swarm optimization mobility-aware approach to fog task scheduling algorithm for Internet of Things devices. *Software: practice and experience*, *51*(12), 2519-2539, 2021.

[27] A. H. Maia, Y. Ghamri-Doudane, D. Vieira, and M. F. de Castro, An improved multi-bjective genetic algorithm with heuristic initialization for service placement and load

distribution in edge computing. *Computer Networks, 194*, 108146, 2021.

[28] M. Ghobaei-Arani, and A. Shahidinejad, A cost-efficient IoT service placement approach using whale optimization algorithm in fog computing environment. *Expert Systems with Applications*, *200*, 117012, 2022.

[29] A. Davies, Cisco pushes iot analytics to the extreme edge with mist computing. *Blog, Rethink Research*, accessed on 19 December 2014.

[30] M. Ghobaei-Arani, A. Souri, A., A. A. Rahmanian, Resource management approaches in fog computing: a comprehensive review. *Journal of Grid Computing*, *18*(1), 1-42, 2020.

[31] G. Keller, M. Tighe, H. Lutfiyya, and M. Bauer, An analysis of first fit heuristics for the virtual machine relocation problem. In *2012 8th international conference on network and service management (cnsm) and 2012 workshop on systems virtualiztion management (svm)*, pp. 406-413, 2012.

فرهاد طاوسی مدرس گروه مهندسی کامپیوتر و فناوری اطلاعات دانشگاه فنی و حرفه‌ای کرمانشاه است. ایشان کارشناسی ارشد خود را در رشته مهندسی فناوری اطلاعات گرایش شبکه‌های کامپیوتری در سال ۱۳۸۸ از دانشگاه آزاد اسلامی قزوین و دکتری مهندسی نرم افزار در سال ۱۴۰۱ از دانشگاه آزاد اسلامی سنندج دریافت نمود. زمینه‌های تحقیقاتی ایشان شامل اینترنت اشیاء، رایانش ابری و رایانش مه است.

آدرس پست الکترونیکی ایشان عبارت است از:

mail_f_t@yahoo.com

سعدون عزیزی دانشیار گروه مهندسی کامپیوتر و فناوری اطلاعات دانشگاه کردستان است. ایشان دکترای خود را سال ۱۳۹۵ در رشته علوم کامپیوتر از دانشگاه صنعتی امیرکبیر تهران دریافت کرده اند. زمینه‌های تحقیقاتی اصلی ایشان شامل اینترنت اشیاء، رایانش ابری، رایانش لبه/مه، رایانش بدون سرور و استفاده از تکنیک‌های هوش مصنوعی و یادگیری ماشین برای حل مسائل بهینه‌سازی در حوزه‌های مختلف علوم کامپیوتر است. هم اکنون ایشان سرپرست آزمایشگاه پژوهشی سیستم‌های رایانشی توزیع شده (DCS Lab) و مدیر مرکز محاسبات سریع دانشگاه کردستان هستند. برای کسب اطلاعات بیشتر می‌توانید به آدرس زیر مراجعه نمایید:

https://research.uok.ac.ir/~sazizi

آدرس پست الکترونیکی ایشان عبارت است از:

s.azizi@uok.ac.ir

عبدالباقی قادرزاده کارشناسی خود را در رشته علوم کامپیوتر از دانشگاه تبریز در سال ۱۳۸۳، کارشناسی ارشد فناوری اطلاعات را از دانشگاه علم و صنعت ایران در سال ۱۳۸۵ و دکتری مهندسی نرم افزار را در سال ۱۳۹۶ از دانشگاه آزاد اسلامی واحد علوم و تحقیقات دریافت کرد. تحقیقات ایشان بر طراحی، تجزیه و تحلیل و کنترل شبکه‌های توزیع شده، یادگیری توزیع شده در سیستم های P2P، رایانش ابری و اینترنت اشیا متمرکز است.

آدرس پست الکترونیکی ایشان عبارت است از:

b.ghaderzadeh@gmail.com