ارائهٔ الگوریتم فراابتکاری جدید براساس جست‌وجوی ممنوعه برای حلّ مسئلهٔ زمان‌بندی وظایف در سیستم مبتنی بر محاسبات ابری و مه

سیّده سمیرا محمّدی[۱]، محمود دی پیر[۲،*]

اطلاعات مقاله

دریافت مقاله: ۱۳۹۸/۰۴/۰۶
پذیرش مقاله: ۱۳۹۸/۰۸/۲۱

واژگان کلیدی:
رایانش مه،
رایانش ابری،
زمان‌بندی وظایف،
خانهٔ هوشمند،
سیستم مدیریت انرژی خانه،
جست‌وجوی ممنوعه،
پلتفرم ابر-مه.

چکیده

در جهان امروز با گسترش ارتباطات، حجم داده‌ها و نیاز به پردازش آن‌ها در زمان کم و با سرعت بالا افزایش یافته است. از طرفی، انجام این حجم از محاسبات نیازمند سیستم‌هایی با ظرفیت‌های پردازشی و ذخیره‌سازی بالا و در نتیجه، هزینهٔ بالاست. بنابراین، پیشنهاد یک زیرساخت مناسب و مقرون به صرفه می‌تواند بسیار قابل توجه باشد. در این مقاله، هدف، طرّاحی و ایجاد یک زیرساخت با هزینه و زمان پاسخ پایین با استفاده از رایانش ابری و مه است. علاوه بر این، یکی از مسائل مهم برای ایجاد چنین سیستم‌هایی با سرعت بالا و حدّاقل زمان، تخصیص مناسب منابع سیستم به درخواست‌های کاربران و در نتیجه تعادل بار در سیستم است. در میان متدهای فراابتکاری گوناگون، جست‌وجوی ممنوعه به دلیل گسترش زیاد آن در مسائل بهینه‌سازی مختلف و همچنین ویژگی حافظه‌دار بودن و سرعت بالا، آن را به یک روش رایج تبدیل کرده است. بنابراین، در این مقاله یک روش جدید فراابتکاری مبتنی بر جست‌وجوی ممنوعه پیشنهاد می‌گردد که با استفاده از روش‌های تقریب نزدیک‌ترین همسایگی و جست‌وجوی مگس میوه بهینه‌سازی می‌شود. برای ارزیابی روش پیشنهادی، یک مطالعه موردی روی خانه‌های هوشمند با زیرساخت پیشنهادی و با داده‌های واقعی شبیه‌سازی شده است. هر دو روش در این زیرساخت اجرا شده و کارایی آن‌ها براساس زمان اجرا و حافظهٔ مصرفی محاسبه شده که نتایج، نشان‌دهندهٔ توانایی و کارایی بالای روش پیشنهادی برای به‌کارگیری در مسائل گوناگون است.

۱- مقدّمه

در زندگی مدرن امروز، عامل هزینه، به یکی از مهم‌ترین اهداف سازمان‌ها و شرکت‌ها تبدیل شده است؛ زیرا انسان‌ها تمایل به مصرف یا تولید با هزینهٔ پایین و کیفیت بالا دارند. بنابراین برای مدیریت و کاهش هزینه، روش‌ها و استراتژی‌های مؤثّری در زمینه‌های گوناگون مورد نیاز است. از طرف دیگر، با پیشرفت روزافزون تکنولوژی، گسترش ارتباطات و همچنین گسترش شهرهای هوشمند و در نتیجه، افزایش تولید داده‌ها در جهان، اهمیت عامل زمان و سرعت انجام عملیات دوچندان شده است. برای رسیدن به این دو هدف و کاهش هم‌زمان آن‌ها، باید سیستم‌هایی با زیرساخت‌هایی ایجاد شود که بتواند این دو عامل را هم‌زمان پشتیبانی کند. همچنین برای ایجاد این زیرساخت‌ها باید پلتفرم‌هایی طرّاحی شوند که ارتباطات و تعاملات را شکل می‌دهند و قواعد و قوانین حاکم بر این تعامل‌ها را تعریف و تعیین می‌کنند. بنابراین، طرّاحی پلتفرم‌هایی که ما را به این دو هدف نزدیک کند، مورد نیاز است. در سال‌های اخیر، تکنولوژی رایج و محبوب برای ایجاد سیستم‌های هوشمند و بهبود ارتباطات و تعاملات، رایانش ابری بوده است که انقلاب بزرگی را در صنعت و تکنولوژی به وجود آورد. در

* پست الکترونیک نویسندهٔ مسئول: mdeypir@gmail.com
۱. گروه مهندسی کامپیوتر، واحد تهران جنوب،دانشگاه آزاد اسلامی، تهران، ایران
۲. گروه مهندسی کامپیوتر، واحد تهران جنوب،دانشگاه آزاد اسلامی، تهران، ایران

واقع رایانش ابری، بستری را با ظرفیت‌های محاسباتی و ذخیره‌سازی بالا برای این ارتباطات و تعاملات ایجاد کرد؛ امّا با افزایش تولید داده‌ها، زمان پردازش عملیات روی آن‌ها و تأخیر در شبکه افزایش یافت. به همین دلیل تکنولوژی رایانش مه در ترکیب با رایانش ابری استفاده شد که با نزدیک شدن به کاربر و دستگاه‌های هوشمند اینترنت اشیا[1] زمان انجام عملیات و تأخیر پردازش‌ها را کاهش داد. بنابراین تکنولوژی‌های رایج برای ایجاد چنین سیستم‌هایی، رایانش ابری و رایانش مه در ترکیب با دستگاه‌های هوشمند اینترنت اشیاست. در نتیجه، محقّقان به دنبال طرّاحی و پیاده‌سازی زیرساخت‌ها و پلتفرم‌ها با استفاده از این تکنولوژی‌ها هستند که بتوانند عوامل هدف را به‌طور هم‌زمان دنبال کنند. در ادامه، رایانش ابری، رایانش مه و ترکیب آن‌ها با دستگاه‌های هوشمند و همچنین هدف و خلاصهٔ کار این مقاله توصیف می‌شود.

همان طور که فنّاوری‌های دیجیتال قدرتمند و در دسترس در حال رشد هستند، سیستم عامل مبتنی بر ابر تقریباً به‌طور گسترده‌ای در حال گسترش است. ادغام محاسبات ابری با شبکه هوشمند عملکرد بالا، انتقال امن و توزیع قدرت، سیستم مدیریت مداوم، ویژگی‌های مقیاس‌پذیر، اقتصادی و انعطاف‌پذیر و توزیع بهتر منابع را فراهم می‌کند. محاسبات ابر به‌صورت پویا منابع محاسباتی، ارتباطاتی و منابع ذخیره‌سازی مجازی و کارآمد را به دستگاه‌های اینترنت اشیا اختصاص می‌دهد. محاسبات ابر خدماتی را براساس نیاز کاربر، صرف‌نظر از موقعیت جغرافیایی آن‌ها ارائه می‌دهد که به سه دستهٔ نرم‌افزار، سخت‌افزار و پلتفرم طبقه‌بندی شده است [۱]. ایدهٔ اصلی پشت این فنّاوری این است که منابع سخت‌افزاری و نرم‌افزاری را می‌توان با مجازی‌سازی و به‌اشتراک‌گذاری منابع از اینترنت فراهم کرد. در نتیجه، هزینه‌های خرید سخت‌افزار و نرم‌افزار و هزینه‌های تعمیر و نگهداری برای کاربران حذف می‌شوند. این یکی از دلایل اهمیت و محبوبیت این فنّاوری در سال‌های اخیر است. اگرچه محاسبات ابری زیربنای قابل توجهی برای این منظور بوده است و داده‌های بزرگ اینترنت اشیا تولیدشده توسط سیستم‌های خانهٔ هوشمند و سنسورهای هوشمند نیاز به پردازش و تجزیه و تحلیل در نزدیکی زمان واقعی دارند، معماری‌های مبتنی بر ابر حجم زیادی از داده‌ها را با تأخیر بالا و پیچیدگی زیاد پردازش می‌کنند. علاوه بر این، هسته ی ابر اغلب از لحاظ فیزیکی و یا منطقی دور از کاربران نهایی است، و اشاره می‌کند که ارتباطات و انتقال داده‌ها باید از بسیاری از هاب[2] ها عبور کند، که تأخیر زیادی را تولید می‌کند و پهنای باند شبکه را اشغال می‌کند [۲]. بنابراین، فن آوری ممکن دیگری مورد نیاز است که در آن سرورهای محاسباتی در نزدیکی کاربر نهایی فیزیکی در لبه شبکه به‌منظور ذخیرهٔ داده‌ها، برای محاسبه وظایف ارسال‌شده، انجام خدمات درخواست‌شده و انتقال داده‌های اینترنت اشیا به سرورهای ابر برای تجزیه و تحلیل عمده واقع شده‌اند.

گره‌های محاسباتی مه از لحاظ منابع کارآمد هستند؛ زیرا آن‌ها با فنّاوری‌های ماشین مجازی مجهّز شده‌اند که قادرند به‌طور مداوم جریان‌های داده‌های اینترنت اشیای تازه را پردازش کرده، داده‌های پردازش‌شده را برای پردازش بیشتر به ابر انتقال دهند [۳]. روش محاسباتی ابر، مزایای بسیار زیادی را پیشنهاد می‌دهد؛ مانند زیرساخت به‌عنوان سرویس[3]: دسترسی به فضای ذخیره‌سازی نامحدود، پلتفرم به‌عنوان سرویس[4]: دارای پتانسیل برای اجرای برنامه‌های کاربردی مؤثّر از لحاظ منابع، نرم‌افزار به‌عنوان سرویس[5]: تسهیلات دسترسی به نرم‌افزار و سرویس‌های سودمند: ذخیره کردن حجم زیادی از داده‌ها برای دسترسی از راه دور. سیستم محاسباتی مه نقش مهمی را در سیستم اینترنت اشیا ایفا می‌کند تا از پردازش داده‌های بزرگ برای پاسخ‌های نزدیک به زمان واقعی پشتیبانی کند. همچنین روش محاسباتی مه اساساً داده‌ها را در لبه سیستم ابر پردازش و ذخیره می‌کند [۴]. این معماری یکپارچه به ما اجازه می‌دهد مشکلات تأخیر مربوط به شبکه ارتباطی انتقال پایه سیستم ابر را که تأثیر ویژه‌ای بر برنامه‌های کاربردی حسّاس به زمان دارد، حلّ و فصل کنیم.

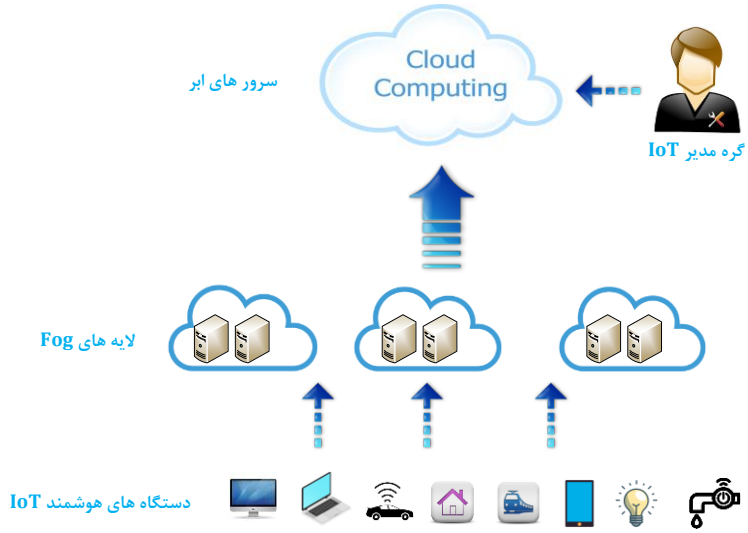
ادغام تجزیه و تحلیل داده‌های بزرگ اینترنت اشیا با معماری مه-ابر نیاز به پلتفرم به‌عنوان رابط کاربری بین مصرف‌کنندگان و تولیدکنندگان دارد [۵]. معماری سه‌بعدی معمول چنین پلتفرم‌هایی شامل دستگاه‌های هوشمند اینترنت اشیا، لایهٔ مه و لایهٔ ابر است که در آن با

[1]. IoT
[2]. Hub
[3]. Infrastructure as a service (IAAS)
[4]. Platform as a service (PAAS)
[5]. Software as a service (SAAS)

یکدیگر ارتباط برقرار کرده، تحت شبکه همکاری می‌کنند. چالش‌های متعدّدی همچنان در مفهوم سیستم‌های مه-ابر-اینترنت اشیا[1] باقی مانده است: مه و 5G، تخصیص منابع و بهینه‌سازی، محاسبات بدون سرور، مصرف انرژی، مدیریت داده‌ها و محلّی بودن سیستم، هماهنگی در گره‌های مه، پویایی، محاسبات شهری، اینترنت اشیا صنعتی و غیره [۶].

درون گره مه، تعداد زیادی از ماشین‌های مجازی روی سرورها و میزبان‌ها ایجاد می‌شوند. بنابراین، هنگامی که یک سرویس توسط کاربر نهایی درخواست می‌شود، آن را به مجموعه‌ای از وظایف شغلی تقسیم می‌کند تا بر روی ماشین‌های مجازی اجرا شود. تخصیص هزاران گره مه به ده هزار کار نیاز به یک الگوریتم زمان‌بندی مطلوب دارد. این تعریف مشکل زمان‌بندی شغلی است [۷].

در این مقاله با توجه به شکل (۱)، هدف، طرّاحی و پیاده‌سازی پلتفرمی با استفاده از تکنولوژی‌های رایانش ابر و مه و در ترکیب با دستگاه‌های هوشمند برای شهر هوشمند است که بتواند در عین انجام عملیات با سرعت پردازش بالا، هزینه و زمان را کاهش دهد و در زمینه‌ها و سیستم‌های گوناگون شهر هوشمند استفاده شود. برای پیاده‌سازی این پلتفرم، یک مطالعهٔ موردی از خانه‌ای هوشمند با داده‌های واقعی در تهران شبیه‌سازی شده است. این پلتفرم شامل چهار عنصر است: ۱. لوازم خانگی هوشمند؛ ۲. سیستم مدیریت سرویس‌های اینترنت اشیا؛ ۳. گره‌های مه؛ ۴. سیستم ابر. برای رسیدن به نتایج موردنظر، تعداد لایه‌های مه بین سیستم ابر و دستگاه‌های هوشمند با توجه به دو عامل هدف هزینه و زمان پاسخ، مورد بحث قرار می‌گیرد و مناسب‌ترین تعداد تعیین می‌شود، به‌طوری که هم هزینه و هم زمان انجام عملیات، کمترین میزان ممکن باشند. بنابراین حدّاقل هزینه و زمان پاسخ به دست می‌آید. مسئلهٔ دیگری که در این مقاله به آن پرداخته شده و امروزه به‌عنوان یکی از مسائل پرچالش در این گونه سیستم‌ها بیان می‌شود، مسئلهٔ زمان‌بندی وظایف میان گره‌های مه و در میان هزاران ماشین مجازی است که در پلتفرم پیشنهادی و در مطالعهٔ موردی شبیه‌سازی و اجرا می‌شود.



شکل ۱- عناصر پلتفرم پیشنهادی برای ایجاد زیرساخت موردنظر

در واقع، مهم‌ترین مزیت استفاده از الگوریتم زمان‌بندی وظایف، اختصاص تعداد زیادی از وظایف به کمترین تعداد گره‌های مه است که با استفاده از حافظه و زمان اجرای پردازشگر[2] کمتر است. علاوه بر این، اختصاص منابع مه به وظایف موردنظر کنترل می‌شود که در آن، بار سیستم متعادل می‌گردد، مصرف انرژی و زمان پاسخ به حدّاقل می‌رسد و سرعت پردازش وظایف افزایش می‌یابد. از آنجا که هیچ الگوریتم دقیقی برای حلّ مسائل زمان‌بندی وظایف وجود ندارد، آن را به‌عنوان یک مسئلهٔ بهینه‌سازی NP-سخت در نظر گرفته است.

پس اهداف مسئله به‌صورت خلاصه عبارت‌اند از:

۱. ارائه‌ی یک پلتفرم مبتنی بر Fog-Cloud با ارزیابی تعداد مختلف لایه‌های Fog برای به دست آوردن تعداد مناسب لایه‌ها با توجه به فاکتورهای هزینه و زمان پاسخ؛

۲. زمان‌بندی وظایف در میان ماشین‌های مجازی در گره‌های مه با استفاده از الگوریتم فراابتکاری بهبود یافته

[1]. Fog-Cloud-IoT

[2]. CPU

مبتنی بر جست‌وجوی ممنوعه.

در ادامهٔ این مقاله، ابتدا ادبیات موضوعی و مفاهیم مرتبط، از جمله پلتفرم مبتنی بر محاسبات ابر و مه و عناصر تشکیل‌دهندهٔ آن، زمان‌بندی وظایف، الگوریتم‌های زمان‌بندی ارائه شده در محاسبات ابر و مه و الگوریتم پیشنهادی بیان می‌شود، سپس زمان‌بندی کارهای مستقل بی‌درنگ براساس الگوریتم پیشنهادی را در پلتفرم محاسباتی پیشنهادی پیاده‌سازی کرده، درنهایت، مقایسه‌ای در شرایط یکسان بین الگوریتم پایه و الگوریتم پیشنهادی انجام می‌گیرد.

۲- ادبیات موضوعی

بسیاری از مطالعات، به مسائل مشابه با کار این مقاله پرداخته‌اند که تعداد محدودی از آن‌ها در این مقاله ذکر شده‌اند؛ مانند [۷]، [۸]، [۱۳]، [۱۴] و [۱۲]. یاسین و همکاران [۸] یک پلتفرم متحد ارائه داده‌اند که عملیاتی پیچیده را بر روی داده‌های دریافت شده از دستگاه‌های هوشمند اینترنت اشیا انجام می‌دهد. این مطالعه به چالش‌های مدیریت منابع برای طبقه‌بندی، پردازش و ذخیره‌سازی داده‌های اینترنت اشیا در حالت آنلاین و آفلاین پرداخته است. اجزای پلتفرم در کار آن‌ها عبارت‌اند از: صفحهٔ اصلی هوشمند، مدیریت اینترنت اشیا و ادغام خدمات و گره‌های محاسبات مه و سیستم ابر. بیتام و همکاران [۷] مشکل زمان‌بندی وظایف را در پارادایم محاسبات مه بررسی کرده‌اند که هدف آن، ارائه خدمات با کارایی بالا و مقرون به صرفه است. آن‌ها یک الگوریتم بهینه‌سازی زیستی جدید را پیشنهاد کردند که الگوریتم زندگی زنبورها[1] نامیده می‌شود تا مجموعه‌ای از وظایف شغلی را در میان گره‌های مه تخصیص دهد. علاوه بر این، آن‌ها مطالعهٔ خود را با بهینه‌سازی ازدحام ذرّات[2] و الگوریتم‌های ژنتیک مقایسه کردند. سینق و یاسین [۱۳] یک پلتفرم تجزیه و تحلیل داده‌های بزرگ چهارلایه‌ای ادغام‌شده با محاسبات مه را برای تجزیه و تحلیل داده‌های مصرف انرژی لوازم خانگی پیشنهاد داده‌اند. پلتفرم آن‌ها شامل کسب داده‌های اینترنت اشیا، مدیریت اینترنت اشیا و ادغام سرویس‌ها، گره محاسباتی مه و سیستم ابر است. این پلتفرم ارائه‌شده، از داده‌های بزرگ با عملیات پیچیده و برنامه‌های کاربردی همیشه در دسترس پشتیبانی می‌کند.

علاوه بر این، آن‌ها یک موتور تجزیه و تحلیل داده در پلتفرم خود برای تجزیه و تحلیل کوتاه‌مدّت در لبهٔ سیستم ابر ارائه داده‌اند. لی و همکاران [۱۴] بر روی مسئلهٔ تخصیص ماشین‌های مجازی در یک مدل ابر-مه تمرکز داشته‌اند. نویسندگان، دو روش مبتنی بر تصمیم‌گیری نیمه مارکوف[3] را پیشنهاد دادند تا هزینه‌های ارائهٔ سرویس‌ها توسط ابر از راه دور را کاهش دهند و به ظرفیت محدود محاسباتی گره‌های مه محلّی تعادل بخشند. سود و همکاران [۱۲] الگوریتم مبتنی بر قاعده را برای به حدّاقل رساندن زمان پاسخ پیشنهاد داده‌اند که در آن، اولویت‌ها را به وظایف و منابع را از مه و ابر اختصاص می‌دهد. همچنین، برای جلوگیری از وقوع بن‌بست، روش پیشنهادی آن‌ها منابع آزاد در دسترس را با یک مفهوم جدید فضای منابع آزاد[4] جمع‌آوری می‌کند. محمّدپور و پروین یک الگوریتم بهینه‌سازی پویای جدید مبتنی بر کلونی زنبور مصنوعی آشوب‌گونه با حافظه ارائه و آن را روی مسئلهٔ محک قلّه‌های متحرّک ارزیابی کرده‌اند [۱۶]. در موضوع مشابه، حسینی و حسنی، یک الگوریتم شاخه و کران را برای زمان‌بندی پردازش قطعات در ایستگاه اوّل و مونتاژ محصولات در ایستگاه دوم در سیستم تولید جریان کارگاهی توسعه داده‌اند، به‌طوری که زمان تکمیل حدّاقل شود [۱۷].

در زمینه‌ای دیگر، شریف‌زاده و امجدی، الگوریتم بهینه‌سازی دسته ذرّات را برای توزیع بهینه توان رآکتیو استفاده و با الگوریتم ژنتیک مقایسه کرده‌اند [۱۸]. **۳- روش تحقیق**

در این مقاله، با توجه به پلتفرم پیشنهادی برای ایجاد زیرساخت طبق شکل، لایه رایانش مه، متشکّل از گره‌های مه است که در قسمت‌های مختلف یک خانه (مانند اتاق خواب، آشپزخانه، اتاق نشیمن، اتاق ناهارخوری و حمّام) واقع شده‌اند و نزدیک به لوازم الکترونیکی هوشمند هستند. بنابراین، لوازم خانگی در هر بخش از خانه به گره مه محلّی خود متّصل می‌شوند. همچنین هر منطقهٔ یک شهر دارای گره یا گره‌های محلّی است که در لایهٔ دوم در این معماری قرار می گیرد و گره مه خانهٔ موردنظر به گره مه ناحیهٔ خود، بسته به موقعیت جغرافیایی در لایهٔ بالایی مرتبط است. درنهایت، این لایه به ابر سازمان انرژی متّصل می‌شود. با توجه به فاکتورهای هدف هزینه، زمان پاسخ، تعداد لایه‌های

[1].BLA
[2].Particle Swarm Optimization (PSO)
[3].SMDP
[4].FRF

مه در میان سیستم ابر و دستگاه‌های اینترنت اشیا تعیین می‌شود و فرض‌های مختلف با یکدیگر مقایسه می‌شوند. در این مدل پیشنهادی، گره مدیر اینترنت اشیا واقع در لایهٔ ابر، الگوریتم پیشنهادی زمان‌بندی وظایف را برای تخصیص تمام وظایف ارسال‌شده انجام می‌دهد.

برای پیدا کردن زمان‌بندی بهینه، یک الگوریتم فراشناختی

سلامت، خانه‌های هوشمند و غیره مورد نیاز است که در آن، باید زمان پاسخ برای پردازش، نزدیک به زمان واقعی پیش‌بینی شود. از طرف دیگر، محدودیت‌های زیادی در پردازش داده‌های اینترنت اشیا و فراخوانی سرویس‌های ابر وجود دارد. مجاورت منابع برای به حدّاقل رساندن تأخیر سیستم کمک می‌کند که مربوط به آماده‌سازی خدمات ابری

جدول ۱- مهمترین پژوهشهای اخیر، مشابه با کار مقاله

نویسندگان	ایده اصلی	اهداف	کمبودها
یاسین و همکاران [۸]	زمان‌بندی در پلتفرم چهارلایه‌ای مبتنی بر Fog-Cloud	• تجزیه و تحلیل مقیاس‌پذیر • مجازی‌سازی • توزیع منابع	تعداد مناسب لایه‌های Fog مشخص نشده است
بیتام و همکاران [۷]	مسئلهٔ زمان‌بندی وظایف در رایانش Fog، پیشنهاد الگوریتم BLA	• عملکرد بالا • خدمات مقرون به صرفه • مقایسه با دو الگوریتم PSO و GA	زمان اجرا، زمان پاسخ و حافظهٔ اختصاصی در نظر گرفته نشده است
سینگ و یاسین [۱۳]	پلتفرم چهارلایه‌ای مبتنی بر Fog-Cloud-IoT	• تجزیه و تحلیل محاسبات فشرده • دسترسی • تجزیه و تحلیل کوتاه‌مدّت	تأخیر، هزینه و زمان اجرا در نظر گرفته نشده است
لی و همکاران [۱۴]	تخصیص ماشین‌های مجازی در یک مدل Fog-Cloud	• کاهش هزینهٔ ارائهٔ سرویس‌ها • تعادل ظرفیت گره‌های Fog	زمان اجرا، زمان پاسخ و حافظهٔ اختصاص‌داده‌شده در نظر گرفته نشده است
سود و همکاران [۱۲]	زمان‌بندی وظایف در مدل Fog-Cloud با الگوریتم مبتنی بر قاعده	• به حدّاقل رساندن زمان پاسخ، مصرف انرژی و تأخیر • جلوگیری از وقوع بن‌بست	هزینه و حافظهٔ اختصاص داده‌شده در نظر گرفته نشده است
زنگ و همکاران [۱۵]	ترکیب الگوریتم‌های tabu search و variable neighborhood descent	• به حدّاقل رساندن زمان	با الگوریتم دیگری مقایسه نشده است

ارائه می‌گردد که توسط گره مدیر اینترنت اشیا اجرا می‌شود.

۳-۱- زمان‌بندی وظایف

دستگاه‌های هوشمند اینترنت اشیا و لوازم خانگی، حجم زیادی داده تولید می‌کنند که نیاز به روش‌های تحلیلی دارد که از نظر منابع و هزینه، مقرون به صرفه هستند. هر گره مه از مراکز داده تشکیل شده و هر مرکز داده دارای حدّاقل یک یا چند میزبان فیزیکی است. ماشین‌های مجازی نیز به‌طور جداگانه روی میزبان‌های فیزیکی ایجاد می‌شوند که منابع را به اشتراک می‌گذارند و این ماشین‌ها باید وظایفی را که توسّط یک کاربر درخواست شده، بر روی میزبان خود اجرا کنند و نتیجه را بازگردانند. بنابراین، ممکن است چند وظیفه را به‌طور هم‌زمان دریافت کند، در نتیجه، هر دستگاه مجازی باید مکانیسم زمان‌بندی داشته باشد. از این رو، برنامه‌ریزی و زمان‌بندی وظایف برای داده‌کاوی داده‌های اینترنت اشیا برای بسیاری از برنامه‌های کاربردی شهر هوشمند، مانند پاسخ خودکار تقاضا، برنامه‌های بهداشتی و

می‌شود. بنابراین، هماهنگی وظایف ارائه‌شده در بین گره‌های رایانش مه که نشان‌دهندهٔ سرویس‌های درخواست‌شده از سوی کاربران تلفن همراه است و در لبه شبکه واقع شده‌اند. همچنین اختصاص دادن این وظایف به منابع سیستم ابر به‌طور مناسب نیازمند بهینه‌سازی مکانیسم‌های زمان‌بندی وظایف است. مسئلهٔ زمان‌بندی وظایف با هدف به حدّاقل رساندن زمان اجرای پردازشگر و حافظهٔ استفاده شده توسّط وظایف ارائه‌شده استفاده می‌شود. در بخش بعد، مدل سیستم پیشنهادی برای پلتفرم ابر-مه ارائه می‌گردد و همچنین مسئلهٔ زمان‌بندی وظایف توضیح داده خواهد شد. بنابراین ، تکنولوژی رایانش مه، زمان پردازش و حافظهٔ اختصاص‌داده‌شده به سرویس‌های درخواست‌شده را نیز کاهش می‌دهد. عملیات مدل پیشنهادی با توضیح مراحل برای اجرای سرویس موردنیاز، به شرح زیر در شکل (۲) آمده است.

دیاگرام مدل سیستم	
شروع	یک کاربر تلفن همراه درخواست سرویس را به گره مدیر IOT ارسال می کند.
گام 1	گره مدیر درخواست را به n کار که هر کار شامل r وظیفه است تقسیم میکند.
گام 2	مدیر کارها را به گره fog محلی برای اجرا ارسال می کند.
گام 3	همچنین گره مدیر، الگوریتم زمانبندی را برای تخصیص وظایف ارسالی به ماشین های مجازی اجرا می کند.
گام 4	هر کار در گره محلی اجرا می شود وبه داده های مورد نیازش دسترسی پیدا میکند.
گام 5	هر گره محلی Fog نتایج خود را به گره Fog در لایه بالایی ارسال می کند و سپس پردازش بیشتر انجام می شود.
گام 6	هنگامی که پردازش وظایف در گره های Fog کامل شد ل‌اسرا ریدم مرگ‌ه‌بار دوخ‌چ‌یاتنا‌هنآ ، می کنند.
پایان	گره مدیر نتیجه نهایی را به عنوان پاسخ سرویس به برنامه کاربردی تلفن همراه ارسال می کند.

شکل ۲- مراحل انجام زمان‌بندی وطایف توسّط گره مدیر IoT

۳-۲- بیان ریاضی مسئله

در این بخش، مسئلهٔ زمان‌بندی وظایف در محیط محاسبات مه توضیح داده شده است. هنگامی که یک کاربر درخواست سرویس را به سیستم ارسال می‌کند، درخواست می‌تواند به‌عنوان مجموعه‌ای از مشاغل تعریف شود. سپس مجموعه‌ای از n کار برای پردازش و اجرا به سیستم ارسال می‌گردد که به‌صورت زیر تعریف می‌شود:

$$\boldsymbol{jobs} = \{\boldsymbol{J_1}.\boldsymbol{J_2}.\dots\dots\dots.\boldsymbol{J_i}.\dots\dots\dots\dots.\boldsymbol{J_n}\} \quad (۱)$$

هر کار i از مجموعه کارهای $(1\leq i\leq n)$ را می‌توان به مجموعه‌ای از r وظیفه $(ie1\leq k\leq r)$ تقسیم کرد و این کار به گره j فرستاده می‌شود.

$$\mathbf{job_i Tasks} = \{\mathbf{JTask^j_{i1}}.\mathbf{JTask^j_{i2}}.\dots\dots\dots.\mathbf{JTask^j_{ik}}.\dots\dots.\mathbf{JTask^j_{ir}}\} \quad (۲)$$

به‌عنوان مثال، وظایف کار i:

$$job_i Tasks = \{JTask^2_{i1}.JTask^2_{i2}.JTask^2_{i3}\} \quad (۳)$$

به شرح زیر انجام می‌شود:

وظیفهٔ اول $(JTask^2_{i1})$، وظیفهٔ دوم $(JTask^2_{i2})$ و وظیفهٔ سوم $(JTask^2_{i3})$ از کار i به گره مه ۲ ارسال می‌شود. بنابراین، هر گره محلّی مه کارهای خود را با مجموعه‌ای از وظایف آن‌ها دریافت می‌کند. FN_j شامل وظایف کار موردنظر به شرح زیر است:

$$FN_j Tasks = \{JTask^j_{ax}.JTask^j_{by}.\dots\dots\dots.JTask^j_{ik}.\dots\dots.JTask^j_{nr}\} \quad (۴)$$

علاوه بر این، هر گره FN_j دارای تعداد زیادی ماشین مجازی است،

که VM_m $(1 \leq the\ number\ of\ VMs \leq m)$ روی میزبان‌های آن ایجاد شده‌اند و وظایف برای اجرا به آن ارسال می‌شوند. مسئله زمان‌بندی وظایف این ماشین‌های مجازی در گره مه است.

$$FN_j Vms = \{VM_1.\dots\dots.VM_m\} \quad (۵)$$

پس از زمان‌بندی، وظایف ارسال‌شده به ماشین مجازی VM_m در گره مه FN_j تخصیص می‌یابند:

$$FN_j VM_m = \{JTask^{jm}_{13}.JTask^{jm}_{57}.JTask^{jm}_{84}\} \quad (۶)$$

پس VM_m وظیفهٔ سوم از کار ۱، وظیفهٔ هفتم از کار ۵ و وظیفهٔ چهارم از کار ۸ را در گره FN_j اجرا می‌کند.

علاوه بر این، زمان اجرای کل CPU برای اختصاص دادن وظایف (r وظیفه) به FN_j به شرح زیر است:

$$CPU_Execution_Time(FN_j Tasks) = \sum_{1 \leq k \leq r} (JTask^{jm}_{ik}.StartTime + JTask^{jm}_{ik}.ExeTime) \quad \boldsymbol{i} \in \boldsymbol{jobs\ of\ selected\ tasks} \quad (۷)$$

که $JTask^{jm}_{ik}.StartTime$ زمانی را که وظیفه k از یک کار i شروع به اجرا می‌کند، در FN_j و VM_m نشان می‌دهد. $JTask^j_{ik}.ExeTime$ CPU زمان اجرای وظیفه k در FN_j است. علاوه بر این، اندازهٔ حافظهٔ موردنیاز برای اختصاص وظیفهٔ k به FN_j به‌صورت زیر محاسبه می‌شود:

$$Memory(FN_j Tasks) = \max_{1 \leq k \leq r} (JTask^{jm}_{ik}.AllocatedMemory) \quad \boldsymbol{i} \in \boldsymbol{jobs\ of\ selected\ tasks} \quad (۸)$$

درنهایت، مسئلهٔ زمان‌بندی وظایف در گره رایانش مه را می‌توان به شرح زیر توصیف کرد:

$$FNTasks = \begin{Bmatrix} FN_1Tasks.FN_2Tasks.\ldots\ldots. \\ .FN_mTasks \end{Bmatrix} \quad (۹)$$

$$FN_jTasks = \begin{Bmatrix} JTask_{ax}^{j1}.JTask_{by}^{j2}.\ldots\ldots\ldots\ldots\ldots\ldots. \\ JTask_{nr}^{jm} \end{Bmatrix} \quad (۱۰)$$

۳-۳- تابع هزینه

تعریف یک تابع هزینه برای ارزیابی کیفیت راه‌حلّ پیشنهادشده لازم است. این تابع هزینه، یک تابع کمینه‌سازی برای محاسبهٔ مقدار مطلوب اهداف زیر است: زمان اجرای پردازشگر و اندازهٔ حافظهٔ اختصاص‌داده‌شده.

$$Cost_function(FNTasks) = Min\left[\sum_{m}^{j=1}\begin{pmatrix} Cost_{function} \\ (JTask_{ik}^{jm}.FN_j) \end{pmatrix}\right] \quad (۱۱)$$

$$Cost_function(JTask_{ik}^{jm}.FN_j) = w1.CPU_Execution_Time(FN_jTasks) + w2.Memory(FN_jTasks) \quad (۱۲)$$

که w1 و w2 ضرایب اهمیت هستند که تصمیم‌گیرندگان آن‌ها را براساس ترجیحات خود از این دو عامل هدف (یعنی زمان اجرای پردازشگر و حافظهٔ اختصاص‌داده‌شده) تعریف می‌کنند [۶].

۳-۴- ترکیب الگوریتم‌های فراابتکاری

در دهه‌های گذشته، الگوریتم‌های فراابتکاری در قالب روش‌های تک‌محور توسّط محقّقان مورد استفاده قرار نگرفته‌اند؛ امّا به روش‌های ترکیبی از سایر کلاس‌های الگوریتم‌های بهینه‌سازی، به‌عنوان یک موضوع مهم توجه شده است [۱۹]. این رویکردها به‌طور کلّی به‌عنوان ترکیب فراابتکارها[1] نامیده می‌شوند. ترکیب الگوریتم‌های مختلف با مفاهیم مختلف، سیستم‌هایی با کارایی بالا ایجاد می‌کند که به‌طور هم‌زمان مزایای الگوریتم‌های فردی را مورد بهره‌برداری قرار می‌دهد. کارایی این روش‌های ترکیبی به‌دلیل همکاری بین الگوریتم‌هاست. بنابراین، برای حلّ مشکلات بهینه‌سازی در شیوه‌ای مقرون به صرفه و با کارایی بالا، یک ترکیب مناسب از الگوریتم‌های فراابتکاری مورد نیاز است. در حقیقت، یک الگوریتم انحصاری تنظیم‌شده از ترکیب مناسب یک الگوریتم فراابتکاری و سایر تکنیک‌های الگوریتمی برای حلّ یک مشکل خاص مورد نیاز است.

مؤلّفه‌های کلیدی الگوریتم‌های بهینه‌سازی شامل دانش مفیدی از مشکل بهینه‌سازی، اجزای الگوریتمی مناسب و ترکیبی مناسب از آن‌ها در یک راه ممکن است. ترکیب فراابتکارها در رابطه با چهار معیار، یعنی انواع الگوریتم‌هایی که برای ترکیب انتخاب شده‌اند، سطح ترکیب‌سازی، ترتیب الگوریتم‌هایی که باید اجرا شوند و استراتژی کنترل، انجام می‌شود. همچنین الگوریتم‌های فراابتکاری می‌توانند با سایر الگوریتم‌های بهینه‌سازی، مانند جست‌وجو درخت، برنامه‌نویسی پویا، برنامه‌ریزی ریاضی، محدودیت‌های برنامه‌ریزی و زمینه‌های حلّ مسئله SMT ترکیب شوند.

۳-۵- الگوریتم جست‌وجوی ممنوعه

جست‌وجوی ممنوعه، یک رویکرد فراشناختی است که در سال ۱۹۸۶ برای اولین بار توسّط گلور و همکارانش ارائه شد [۲۰]. این روش همچنین به‌عنوان یک روش فراابتکاری پویا شناخته می‌شود، به دلیل حافظهٔ کوتاه‌مدّت یا بلندمدّت که استراتژی اصلی آن است. هنگامی که جست‌وجو در حدّاقل محلّی به دام می‌افتد، این روش می‌تواند راه‌حلّ دیگری برای از بین بردن این تله پیدا کند و پیشرفت‌های بیشتری به دست آورد؛ زیرا استفاده از یک مفهوم به نام لیست ممنوعه[2] است که در آن حرکات مربوط به راه‌حلّ فعلی با برچسب ممنوع‌شده[3] ذخیره می‌شوند، به‌طوری که آن‌ها دوباره به‌عنوان حرکت جدید در تکرارهای قانون ممنوعه[4] انتخاب نمی‌شوند و پس از آن هیچ حلقه‌ای ایجاد نمی‌گردد. علاوه بر این، جست‌وجوی ممنوعه اطلاعات دربارهٔ راه‌حل‌های اخیر، هزینهٔ کل برای هر راه‌حل و تعداد دفعاتی را که یک حرکت جست‌وجو شده است، در حافظهٔ خود نگه می‌دارد. این اطلاعات می‌تواند وضعیت فرایند جست‌وجو را تعیین کند؛ زیرا مقدار زیاد آن نشان می‌دهد که روند جست‌وجو در بهینه محلّی گیر کرده است و جست‌وجو باید پایان یابد. مدّت زمانی که یک حرکت در فهرست ممنوعه باقی می‌ماند، بستگی به طول لیست دارد و هر بار که یک حرکت به لیست اضافه می‌گردد، لیست به‌روز می‌شود. اگر حرکتی که به‌عنوان ممنوع‌شده علامت‌گذاری شده، بتواند به اندازهٔ کافی قوانین اشاعه[5] را برآورده کند، می‌تواند دوباره انتخاب و از لیست حذف شود. درنهایت، اگر جست‌وجوی ممنوعه نتواند نتایج بیشتری را از طریق تکرارهای طول مسیر جست‌وجو[6]

[1]. Hybrids of meta-heuristics
[2]. Tabu List
[3]. Tabued
[4]. Tabu Tenure
[5]. Aspiration Rule
[6].Serach Path Length

پیدا کند، جست‌وجو تمام شده و راه‌حلّ بهینه بازگردانده می‌شود. دلایل انتخاب الگوریتم جست‌وجوی ممنوعه از میان دیگر الگوریتم‌های فراابتکاری عبارت است از:

۱. دلیل محبوبیت آن، ساده بودن پروسهٔ زمان‌بندی و همچنین دارا بودن سه نوع حافظه که جست‌وجو را از قرار گرفتن در بهینه محلّی بازمی‌دارد. همچنین به‌دلیل داشتن مفهومی به نام لیست ممنوعه و معیار تنفّس، از حرکات و راه‌حل‌های تکراری جلوگیری می‌کند. بنابراین سرعت پیدا کردن بهترین جواب (بهینهٔ سراسری) افزایش می‌یابد [۲۱].

۲. جست‌وجوی ممنوعه از حرکات قطعی استفاده می‌کند که به‌دلیل راه‌حل‌های اولیه و سایر پارامترها، تنوّع را کاهش می‌دهد [۲۲].

۳. جست‌وجوی ممنوعه به‌جای گذراندن زمان بیشتر در مناطقی که راه‌حل‌های آن‌ها مطلوب نیست، تلاش بیشتری را برای کاوش مناطقی که راه‌حل‌های مناسب و مطلوب بیشتری دارند، اختصاص می‌دهد [۲۱].

شبه کد جست‌وجوی ممنوعه در الگوریتم ۱ آورده شده است.

۳-۶- راه‌حل اولیه

برای ایجاد راه‌حلّ اولیه، به‌جای روش تصادفی، در این مطالعه، الگوریتم جست‌وجوی تقریب نزدیک‌ترین همسایگی مورد استفاده قرار می‌گیرد.

۳-۶-۱- جست‌وجوی تقریب نزدیک‌ترین همسایه

برای به دست آوردن نتایج دقیق و جلوگیری از شروع جست‌وجو با راه‌حلّ اشتباه و همچنین برای به حدّاقل رساندن زمان برای پیدا کردن نزدیک‌ترین همسایه، هزینه‌های اجرایی و حافظهٔ مصرفی، این مطالعه، تقریب نزدیک‌ترین همسایه را برای ایجاد راه‌حلّ اولیه در الگوریتم جست‌وجوی ممنوعه به کار می‌برد.

مجموعه اشیای X را در نظر بگیرید، با توجه به دقّت پارامتر e و فاصلهٔ R و یک ساختار داده که در آن هر شیء پرس‌وجو q ، نزدیک‌ترین همسایهٔ خود را در X بازمی‌گرداند. بعضی از اطلاعات دیگر دربارهٔ مجموعهٔ X در یک ساختار داده ذخیره شده، برای پیدا کردن نزدیک‌ترین همسایه X بدون هیچ عملیات محاسباتی استفاده می‌شود. در قسمت بعد، مسئلهٔ تقریب نزدیک‌ترین همسایه[1] توصیف شده است.

مسئلهٔ پیدا کردن نزدیک‌ترین همسایه به شرح زیر است:

```
// construct initial solution randomly
S ← S0
sBest ← S
tabuList ← null
While (not stopping condition ())
    condidateList ← null
    For ( sCondidate in sNeighborhood)
        If
(not contains TabuElements (sCondidate, tabuList))
            condidateList ← condidateList +
sCondidate
        End
    End
    sCondidate ←
LocateBestCondidate (condidateList)
    If (fitness (sCondidate) > fitness (sBest))
        sBest ← sCondidate
        tabuList ←
featureDifferences (sCondidate, sBest)
        While (size (tabuList) >
maxTabuListSize)
            ExpireFeatures (tabuList)
        End
    End
End
Return (sBest)
```

الگوریتم ۱- شبه کد جست‌وجوی ممنوعه

مسئلهٔ تقریب نزدیک‌ترین همسایه عبارت است از:

- اگر شیء y در X وجود داشته باشد، به‌طوری که $d(q.y) \leq R$
- سپس z در X بازگردانده می‌شود: $d(q.z) \leq (1+e)R$
- اگر هیچ شیء y در X وجود نداشته باشد: $d(q.z) \geq (1+e)R$
- سپس No بازگردانده می‌شود.

تقریب نزدیک‌ترین همسایه می‌تواند به تقریب همسایهٔ نزدیک کاهش یابد:

- اجازه بده d و D کوچک‌ترین و بزرگ‌ترین فاصله باشند.

ساختارهای تقریب همسایهٔ نزدیک را بساز برای:

$$\boldsymbol{R = d.(1+e)d.(1+e)^2 d. \dots \dots . . D} \qquad (۱۳)$$

- برای پرس‌وجوی نقطه q، تمام ساختار تقریبی همسایهٔ نزدیک را جست‌وجو کن با:

$$R = d.(1+e)d.(1+e)^2 d. \dots \dots . . D$$

- نقطهٔ پیداشده در دایره غیرمولّد با کوچک‌ترین شعاع را بازگردان.
- جواب تقریب نزدیک‌ترین همسایه برای q است.

[1].Approximate Nearest Neighbor

۲-۶-۳- حل مسئلهٔ تقریب نزدیک‌ترین همسایگی با روش هش

برای حلّ مسئلهٔ تقریب همسایه نزدیک، محلّی‌سازی با حسّاسیت هش[1] مورد استفاده قرار گرفته است [۲۳].

به‌رغم پیشرفت‌های بسیاری و الگوریتم‌های مختلف جست‌وجوی اکتشافی، هنوز یافتن راه‌حلّ بهینه توسط هریک از جدیدترین الگوریتم‌ها تضمین نشده است. بنابراین، بهبود الگوریتم‌ها و پیشرفت‌های فنّاوری برای حلّ این مشکلات همچنان ادامه دارد. در این مطالعه، برای افزایش کارایی فرایند جست‌وجوی الگوریتم جست‌وجوی ممنوعه، الگوریتم بهینه‌سازی مگس‌های میوه به کار می‌رود و هنگامی اعمال می‌شود که الگوریتم جست‌وجوی ممنوعه نمی‌تواند بهبود بیشتری پیدا کند. بخش بعدی، الگوریتم بهینه‌سازی مگس‌های میوه را توصیف می‌کند.

۳-۷- الگوریتم بهینه‌سازی مگس میوه

الگوریتم بهینه‌سازی مگس‌های میوه ارائه‌شده توسط پن در سال ۲۰۱۲ [۲۴] بر مبنای رفتار جست‌وجوی مگس میوه برای غذاست که رفتار آن‌ها شامل دو مرحله است. مگس‌های میوه نسبت به سایر حشرات، توانایی بوییدن و بینایی بهتری دارند، به‌طوری که آن‌ها می‌توانند منابع غذایی را از ۴۰ کیلومتری تشخیص دهند. در مرحلهٔ اول، ابتدا مگس‌های میوه به‌طور تصادفی به‌سمت چند منبع غذایی که در اطراف گروه مگس‌های میوه هستند، از طریق Osphresis (حسّ بویایی) پرواز می‌کنند. این فاز فرایند کاوش کردن از طریق بو[2] نامیده می‌شود. در مرحلهٔ دوم که مرحلهٔ کاوش کردن از طریق بینایی[3] نامیده می‌شود، مگس میوه از طریق بینایی خود به‌سمت بهترین منبع غذا پرواز می‌کند و به منبع غذایی نزدیک‌تر می‌شود. این مرحله تا زمانی که شرط توقّف به دست آید، تکرار می‌شود. در نتیجه، در طول این فرایندها، مکان گروه مگس‌های میوه به‌روز می‌شود.

۳-۸- شرط توقّف

در الگوریتم اصلی جست‌وجوی ممنوعه، شرط توقّف تکرارهای فرایند می‌تواند طول لیست ممنوعه باشد، به‌طوری که،

$size\ (tabuList) > maxTabuListSize$ یا

$f(best) \geq f(initial\ solution)$

در این مطالعه، به‌منظور به حدّاقل رساندن زمان پاسخ فرایند، هزینه و مصرف انرژی، شرط توقّف در الگوریتم پیشنهادی جست‌وجوی ممنوعه با تقریب نزدیک‌ترین همسایه مبتنی بر الگوریتم مگس میوه[4]، حداکثر زمان پاسخ در نظر گرفته شده است، به‌طوری که:

While

$\boldsymbol{ResponseTime \geq maxResponseTime}$

سپس، روند متوقّف می‌شود.

۳-۹- الگوریتم پیشنهادی ATS-FOA

الگوریتم ATS-FOA پیشنهادی که وظایف ارائه‌شده را در میان ماشین‌های مجازی در گره‌های مه مدل ابر-مه زمان‌بندی می‌کند، جست‌وجوی ممنوعه را بهینه خواهد کرد. مطابق شکل (۳) این الگوریتم براساس الگوریتم جست‌وجوی ممنوعه است که در آن راه‌حلّ اولیه از طریق تقریب نزدیک‌ترین همسایگی و محلّی‌سازی با حسّاسیت هش ساخته شده است. شرط توقّف حدّاکثر زمان پاسخ در نظر گرفته شده، برای بهبود کارایی جست‌وجوی ممنوعه، بهینه‌سازی مگس میوه فراخوانده می‌شود، زمانی که هیچ راه‌حلّ بهتری توسّط جست‌وجوی ممنوعه یافت نشود. به عبارت دیگر، بهینه‌سازی مگس میوه با لیست بهترین راه‌حل‌های لیست (candidateList) الگوریتم جست‌وجوی ممنوعه شروع می‌کند و راه‌حل‌های قدیمی جست‌وجوی ممنوعه را از لیست حذف می‌کند و راه‌حل‌های جدید را تولید می‌کند. بنابراین فهرست نامزدها به‌روز می‌شود و بهبود می یابد. همچنین، آن دسته از راه‌حل‌های غیربهبودی که مایل به فرار از حدّاقل محلّی هستند، بهینه‌سازی مگس میوه آن‌ها را قبول می‌کند. الگوریتم پیشنهادشده در زیر و در الگوریتم ۲ شرح داده شده است:

مرحلهٔ ۱: راه‌حل اولیه S_0 را با تقریب نزدیک‌ترین همسایه و محلّی‌سازی با حسّاسیت هش ایجاد کن و $S = S_0$ و $sBest = S$ قرار بده.

مرحلهٔ ۲: یک زیرمجموعه $sNeighborhood$ از راه‌حل‌ها تولید کن.

مرحلهٔ ۳: بهترین $sCandidate$ را در $sNeighborhood$ انتخاب کن و اگر $tabuList$ حاوی

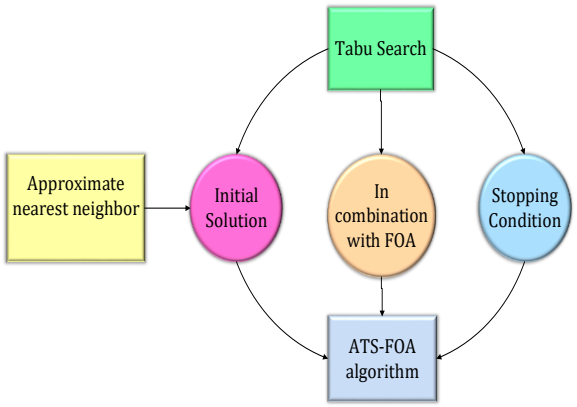
[1]. Locality Sensitive Hashing
[2]. Smell Foraging
[3]. Vision Foraging
[4]. Approximate Tabu search based on Fruit Fly Optimization Algorithm (ATS-FOA)

$sCandidate$ نیست، $sCandidate$ را به $candidateList$ اضافه کن.

مرحلهٔ ۴: در غیر این صورت، اگر بهترین جواب در لیست ممنوعه وجود داشت، جست‌وجوی ممنوعه را خاتمه بده و جست‌وجوی مگس میوه را با آخرین لیست ممنوعه اجرا کن.

مرحلهٔ ۵: بهترین راه‌حل‌های بهینه‌سازی مگس میوه را به $candidateList$ اضافه کن.

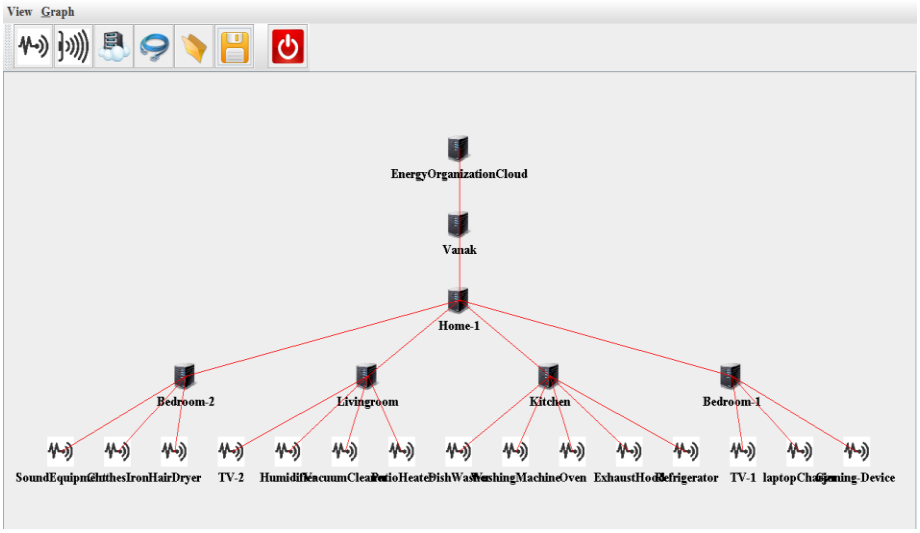
مرحلهٔ ۶: اگر $ResponseTime >$ $maxResponseTime$، فرایند را متوقّف کن، در غیر این صورت، به مرحلهٔ ۲ برو.



شکل ۳- دیاگرام روش پیشنهادی براساس جست‌وجوی ممنوعه

```
//پیدا کردن بهترین جواب اولیه با استفاده ازروش approximate nearest neighbor
S ← S0
sBest ← S
tabuList ← null                          لیست ممنوعه را در نظر بگیر:
While (not stopping condition ())        تا زمانی که شرط خاتمه برقرار نیست
  candidateList ← null                   لیست کاندید را در نظر بگیر
بهترین جواب را بین همسایگان انتخاب کن، اگر اگر در لیست ممنوعه وجود نداشت آن را به لیست کاندید اضافه کن.
  For ( sCandidate in sNeighborhood)
  If (not contains TabuElements (sCandidate,
   tabuList))
                candidateList ← candidateList + sCandidate
  End
  End
در غیر این صورت، اگر بهترین جواب در لیست ممنوعه وجود داشت، جست‌وجوی ممنوعه را خاتمه بده و جست‌وجوی مگس میوه را با آخرین لیست ممنوعه اجرا کن.
  Return (candidateList)
  Call FruitFlyOptimization (candidateList)
اگر ResponseTime > maxResponseTime، سپس فرایند را متوقف کن، در غیر این صورت، به مرحلهٔ ۲ برو.
  While (ResponseTime > maxResponseTim)
  End
  Return (sBest)
```

الگوریتم ۲- شبه کد الگوریتم پیشنهادی



شکل ۴. پیکربندی مدل پیشنهادی با IFogSim.

۴- شبیه‌سازی و ارزیابی روش پیشنهادی

۴-۱- تنظیمات شبیه‌سازی

در این مطالعه، یک خانه هوشمند در تهران برای اجرای مدل پیشنهادی و الگوریتم طراحی شده است. بخش های مختلف این خانه شامل آشپزخانه، اتاق خواب و اتاق نشیمن است که در آن دستگاه‌های الکترونیکی هوشمند که در این مدل در نظر گرفته شده اند در جدول.۲ نشان داده شده اند. این دستگاه‌های هوشمند که سنسورهایی برای اندازه گیری پارامترهای محاسبه مصرف انرژی خود دارند، در اولین لایه این توپولوژی قرار دارند. با توجه به شکل۴ ، دستگاه‌های هوشمند در هر بخش از خانه به گره مه از همان قسمت از خانه در لایه دوم توپولوژی متصل شده و داده‌های خود را به آنها ارسال می‌کنند. با توجه به تعداد لوازم هوشمند در هر بخش و گستردگی فضای خانه، بیش از یک گره مه در هر بخش می تواند قرار گیرد، اما به دلیل اهمیت عامل هزینه، در این مطالعه یک گره مه در هر قسمت از خانه در نظر گرفته شده است و پارامترهای گره‌های مه را برای انجام پردازش اولیه داده‌های اینترنت اشیا تنظیم می‌کند. پارامترهای مورد نظر برای گره‌های مه و سیستم ابر در جدول ۳ نشان داده شده است. همچنین مشخصات سخت‌افزار مورداستفاده در جدول ۴ آمده است.

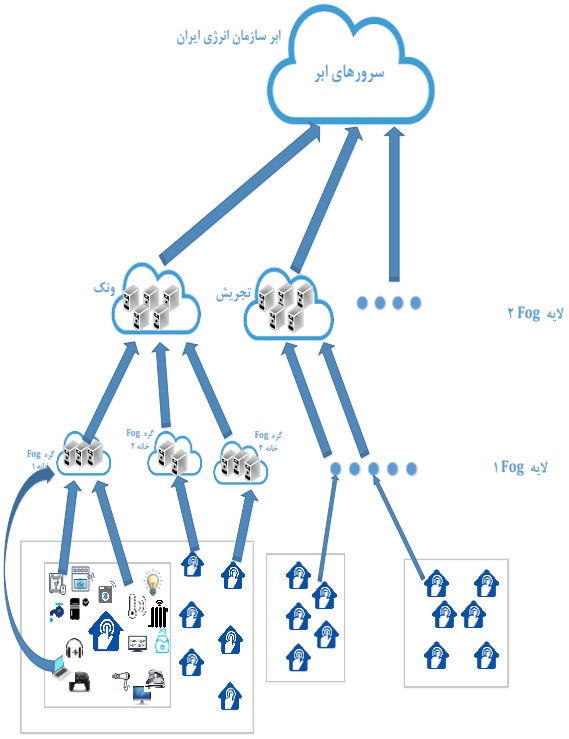
جدول ۲- لوازم الکترونیکی هوشمند

بخش‌های مختلف خانه	لوازم الکترونیکی
آشپزخانه	یخچال
	ماشین لباس‌شویی
	ماشین ظرف‌شویی
	هود خروجی
	فر برقی
اتاق خواب ۱	تلویزیون
	دستگاه بازی
	شارژرهای لپ‌تاپ و موبایل
اتاق خواب ۲	اتوی لباس
	تجهیزات صوتی
	سشوار
سالن نشیمن	تلویزیون
	جاروبرقی
	رطوبت‌ساز
	بخاری برقی

گره مه هر بخش به گره مه خانهٔ موردنظر در لایهٔ دوم متّصل می‌شود. علاوه بر این، ۲۲ منطقهٔ تهران دارای گره مه محلّی یا گره‌های مه هستند و خانهٔ هوشمند به گره مه منطقهٔ خود در لایهٔ سوم متّصل می‌شود. هر منطقه می‌تواند بیش از یک گره مه داشته باشد؛ امّا در این مدل، یک گره مه در لایهٔ سوم برای منطقهٔ ونک تهران در نظر گرفته شده که شامل بسیاری مراکز داده و میزبان است و بسیاری از ماشین‌های مجازی روی این میزبان‌ها برای دریافت و پردازش مقادیر زیادی از داده‌های تولیدشده توسط خانه‌های هوشمند ایجاد شده‌اند. درنهایت، گره‌های مه مناطق مختلف تهران به‌منظور انجام پردازش بیشتر و ذخیرهٔ داده‌ها به ابر سازمان انرژی متّصل می‌شوند. این توپولوژی در شکل (۴) و (۵) نشان داده شده است.

جدول ۳- تنظیمات گره‌های Fog

پارامترها	مقادیر
Level	۰-۳
Uplink BW	۱۰۰۰۰
Downlink BW	۲۷۰-۱۰۰۰۰
Mips	۲۰۰۰
Ram	۴۰۰۰
Rate/MIPS	۰٫۰۱



شکل ۵- زیرساخت موردنظر برای خانهٔ هوشمند با سه لایه مه

جدول ۴- مشخصات زیرساخت استفاده‌شده برای شبیه‌سازی

پارامترها	مقادیر
system architecture	x86
operating system	Linux
vmm[1]	Xen
costPerMemory	۰,۰۵
costPerStorage	۰,۰۰۱
costPerBw	۰,۰

یک مجموعه دادهٔ واقعی [۳۴] از یک خانهٔ هوشمند در این معماری شبیه‌سازی شده و برای ارزیابی الگوریتم پیشنهادی و مقایسهٔ نتایج به‌دست‌آمده با الگوریتم جست‌وجوی ممنوعه و همچنین بررسی فرضیه‌های مختلف برای تعیین تعداد لایه‌های مه بین ابر و دستگاه‌های اینترنت اشیا استفاده گردیده است. در اولین فرض، معماری این مدل سه لایه مه در میان سیستم ابر و لوازم هوشمند دارد. در این سه لایه تعداد گره‌های مه ۷ است و نتایج این فرض در جدول ۵ نشان داده شده است. سپس در سناریوی دوم فرض می‌شود لایهٔ دوم مه از این معماری حذف شود و گره‌های مه قسمت‌های مختلف خانه به گره منطقه جغرافیایی خود (به‌عنوان مثال منطقهٔ ونک در مدل فرض‌شده) متّصل گردد که تعداد گره‌های مه در این فرض ۶ است. از سوی دیگر، در سناریوی سوم، یک لایهٔ اضافی مه به این معماری اضافه می‌گردد که یک گره مه یا گره‌های مه برای مجموعه مجتمع مسکونی فرض می‌شود. بنابراین تعداد گره‌های مه را تا ۸ افزایش می‌دهد. سپس این فرض‌ها در مدل شبیه‌سازی‌شده که توسّط ابزار کلودسیم شبیه‌سازی گردیده است، برای زمان‌بندی وظایف بین گره‌های مه و ماشین‌های مجازی در یک گره مه با استفاده از الگوریتم ATS-FOA ارزیابی شده‌اند.

شبیه‌ساز مورد استفاده، کلود سیم[2] است. شبیه‌ساز کلود سیم، مدل‌سازی رفتار عناصر سیستم، مانند مراکز داده، ماشین‌های مجازی و سیاست‌های اختصاص منابع را پشیبانی می‌کند. بنابراین می‌توان از آن برای مدل کردن محیط ابرهای محاسباتی و تست کارایی الگوریتم‌های پیشنهادی استفاده کرد. این شبیه‌ساز مبتنی بر زبان جاوا و رویداد است؛ یعنی موجودیت‌هایی که در آن تعریف می‌شود، از طریق فرستادن رویداد با یکدیگر ارتباط برقرار می‌کنند.

در این شبیه‌سازی تعداد وظایف ارسال‌شده برای پردازش در سناریوهای مختلف با تعداد لایه‌های ۲و۳، ۴، ۴۰۰ در نظر گرفته شده است. هر وظیفه نیز 10^3 دستورالعمل دارد. همچنین در فرض موردمطالعه، ضرایب متغیّرهای زمان اجرا و حافظهٔ مصرفی در تابع هزینه توسط تصمیم‌گیرنده تعیین می‌شود:

$$0.9\ CPU_Execution_Time + 0.1\ Allocated_Memory \quad (۲۵)$$

تمام آزمایش‌ها با ۲۰ ثانیه پروفایل پروژه (زمان اجرای ماشین‌های مجازی ۲۰ ثانیه در نظر گرفته شده است) انجام شده و نتایج جدول ۵ به دست آمده است.

۲-۴- نتایج شبیه‌سازی

هدف از انجام این ارزیابی و شبیه‌سازی، اجرا و پیاده‌سازی دو الگوریتم جست‌وجوی ممنوعه و ATS-FOA در شرایط یکسان در زیرساخت پیشنهادی و به دست آوردن مقادیر زمان پاسخ ۴۰۰ کار اختصاص داده شده، زمان اجرا، حافظهٔ مصرف‌شده برای پردازش این کارها و هزینهٔ پیاده‌سازی و اجرای آن‌ها براساس زمان اجرا و حافظهٔ مصرفی است تا بتوان در شرایط یکسان مقایسه‌ای بین دو الگوریتم انجام داد و کارایی الگوریتم جدید را نشان داد.

همان طور که از شکل (۸) به دست می‌آید، زمان پاسخ کارهای درخواست‌شده (۴۰۰ کار) در زیرساخت پیشنهادی که دارای سه لایه مه است، با انجام الگوریتم زمان‌بندی جدید ۲۷۴۴ در ۲۰ ثانیه پروفایل پروژه، ۲۷۴۴,۰۷ ثانیه است (به این معنا که با ۷ گره مه زمان پاسخ ۴۰۰ وظیفه از تقسیم ۲۷۴۴,۰۷ بر ۲۰ به دست می‌آید و به همین ترتیب برای تمامی مقادیر) و با انجام الگوریتم جست‌وجوی ممنوعه ۲۹۵۴,۲۸ ثانیه است. دلیل کاهش زمان پاسخ در الگوریتم جدید این است که جست‌وجوی ممنوعه برای تخصیص وظایف به ماشین‌های مجازی در گره‌های مه با راه‌حل‌های تصادفی شروع به جست‌وجو می‌کند که در این میان ممکن است تخصیص وظایف به ماشین‌های مجازی مشغول، زمان انتظار کارها در صف و زمان پاسخ آن‌ها را افزایش دهد، در حالی که در روش جدید، به‌دلیل انتخاب مناسب‌ترین جواب به‌عنوان جواب اولیه و در واقع تخصیص مناسب وظایف به ماشین‌های مجازی که بار پردازشی کمتری دارند، سرعت انجام عملیات و در نتیجه، زمان پاسخ کمتر خواهد شد. همچنین در الگوریتم جدید، با توجه به الگوریتم ۲، در صورتی که بهترین جواب همسایه در فهرست ممنوعه وجود داشته باشد و معیار تنفّس آن از بهترین جواب یافت‌شده

[1].Virtual machine monitor

[2].cloudsim

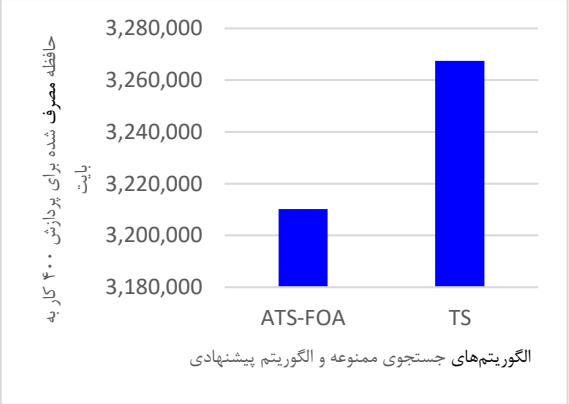
تاکنون نیز بهتر باشد، باز هم الگوریتم به آن حرکت نخواهد کرد و جست‌وجو خاتمه یافته، بلافاصله الگوریتم مگس میوه با لیست کاندید جست‌وجوی ممنوعه شروع به جست‌وجو می‌کند. در نتیجه، جست‌وجو در بهینه محلّی به دام نمی‌افتد و از طریق اجرای جست‌وجوی مگس میوه بهترین بهبود به دست می‌آید. با به دست آمدن بهترین جواب، وظایف به مناسب‌ترین ماشین‌های مجازی اختصاص یافته، در زمان کوتاه‌تری اجرا می‌شوند. در جست‌وجوی ممنوعه نیز با وجود لیست ممنوع، از افتادن به دام محلّی جلوگیری می‌شود، ولی به‌دلیل وجود شرط تنفّس، جواب در صورت وجود در لیست ممنوعه نیز امکان انتخاب دوباره را دارد؛ به همین دلیل ممکن است در شرایطی در بهینه محلّی به دام افتد. علاوه بر این، به‌دلیل اینکه شرط توقّف در الگوریتم جدید، حدّاکثر زمان پاسخی است که در نظر گرفته می‌شود، زمان پاسخ کلّ فرایند جست‌وجو از یک حدّی بیشتر نخواهد شد. شکل (۹) زمان اجرای پردازشگرها را برای پردازش ۴۰۰ کار را با پیاده‌سازی دو روش نشان می‌دهد. مشابه زمان پاسخ، زمان اجرا نیز برای الگوریتم جدید کمتر از جست‌وجوی ممنوعه خواهد بود؛ زیرا با اختصاص کارها به ماشین‌های مجازی که بار پردازشی کمتری دارند، سبب می‌شود کارها در زمان کمتری اجرا شوند. همچنین عدم محاسبهٔ معیار تنفّس برای جواب‌ها و عدم تکرار آن‌ها در فرایند جست‌وجو، زمان اجرای وظایف را کاهش می‌دهد. شکل (۶) حافظهٔ مصرف‌شده برای پردازش کارها توسط اجرای دو الگوریتم را نشان می‌دهد. همان طور که می‌بینیم میزان حافظهٔ مصرفی توسط الگوریتم جدید نسبت به جست‌وجوی ممنوعه به میزان بسیار کمی کاهش یافته است. دلیل این امر این است که بعد از انجام جست‌وجوی تقریب نزدیک‌ترین همسایه بـا روش هش برای یافتن جواب اولیه، حافظهٔ اختصاص‌داده‌شده به آن آزاد می‌شود.

جدول۵ - مقایسهٔ دو الگوریتم براساس مقادیر فاکتورهای هدف در ۳ سناریو در نظر گرفته شده

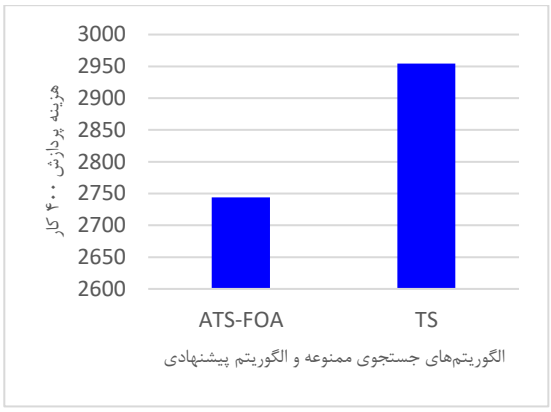
۴۰۰ وظیفه	زمان اجرا (S)			زمان پاسخ (S)			حافظه مصرفی (B)			هزینه		
سناریو	۱	۲	۳	۱	۲	۳	۱	۲	۳	۱	۲	۳
ATS-FAO	۲۶۷۷٫۳۸	۲۷۴۳٫۹۳	۲۹۶۲٫۲۹	۲۶۷۷٫۷۶	۲۷۴۴٫۰۷	۲۹۶۲٫۴۴	۳۰۰۱۱۰۴	۳۲۱۰۲۲۴	۳۳۵۰۴۸۰	۰٫۰۸۶۰۹۴	9.21E-02	9.62E-02
Tabu Search	۲۸۰۰٫۰۱	۲۹۵۴٫۱۳	۳۰۶۹٫۲۴	۲۸۰۰٫۱۷	۲۹۵۴٫۲۸	۳۰۶۹٫۳۸	۳۰۰۱۳۰۴	۳۲۶۷۴۴۸	۳۴۸۵۴۲۴	8.61E-02	9.38E-02	0.100101

علاوه بر این در الگوریتم جدید به‌دلیل عدم محاسبهٔ معیار تنفّس، حافظه‌ای به آن اختصاص نمی‌یابد. همچنین تنها جواب‌هایی که اخیراً ملاحظه شده‌اند، به‌عنوان حافظهٔ کوتاه‌مدّت نگهداری می‌شوند و اطلاعات دیگری همانند حرکاتی که انجام شده یا تعداد دفعاتی که یک حرکت یا مسیر ممنوع، جست‌وجو شده که در جست‌وجوی ممنوعه ذخیره می‌گردد و حافظه به آن‌ها اختصاص می‌یابد، نگهداری نمی‌شود. شکل (۷) هزینهٔ اجرا و پردازش کارها و در واقع کارایی را براساس زمان اجرا و حافظهٔ مصرفی توسط دو الگوریتم نشان می‌دهد که هزینه محاسبه شده برای الگوریتم جدید، کمتر از جست‌وجوی ممنوعه است. شکل (۱۰) نیز مقایسه بین نتایج دو الگوریتم برای سه سناریو را نشان می‌دهد. همان طور که پیداست، اگر تعداد لایه‌های مه کمتر باشد، مقادیر تمام فاکتورهای هدف نیز برای الگوریتم جدید بهتر است. این بدان معناست که با افزایش تعداد گره‌های مه و در نتیجه افزایش ظرفیت ذخیره‌سازی و پردازشی، نه‌تنها سرعت و کارایی را در این سیستم افزایش نمی‌دهد، بلکه زمان و هزینه را نیز بیشتر می‌کند. تعداد مناسب و مقرون به صرفه با هدف برآوردن هر دو فاکتور زمان و هزینه باید در هر زمینه و برای هر سیستمی بسته به نوع فعالیت آن تعیین شود.

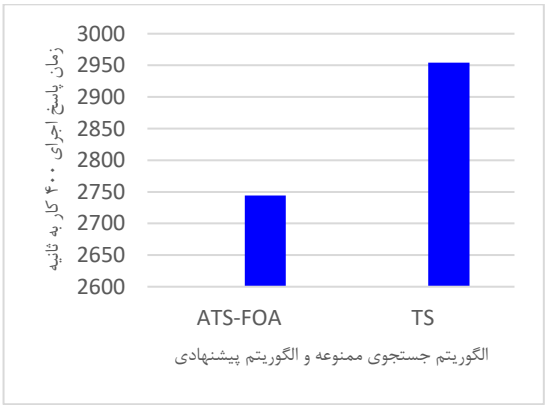
همگرایی یک الگوریتم، همان تعداد دورهایی است که الگوریتم برای رسیدن به جواب بهینه طی می‌کند. بنابراین، همگرایی الگوریتم پیشنهادی نیز در شبیه‌سازی برای رسیدن به بهترین زمان‌بندی به دست آمده است. برای مثال در سناریوی اول برای ۴۰۰ وظیفه ارسال‌شده به سیستم با دو لایه گره مه و تعداد شش گره مه، تعداد دورها برای به دست آمدن زمان‌بندی بهینهٔ گره‌های مه، ۲۹ دور است.



شکل ۶- حافظه مصرف‌شده توسط پردازش ۴۰۰ کار در زیرساخت خانهٔ هوشمند با سه لایه مه



شکل ۷- هزینه محاسبه‌شده برای پردازش ۴۰۰ کار در پلتفرم پیشنهادی با سه لایه مه



شکل ۸- زمان پاسخ اجرای ۴۰۰ کار در پلتفرم پیشنهادی با سه لایه مه برای خانهٔ هوشمند

الگوریتم برای زمان‌بندی بهینهٔ گره‌های مه از [۴, ۶, ۱, ۲, ۳, ۵] زمان‌بندی شروع به اجرا می‌کند و بعد از ۲۹ تکرار به [۲, ۳, ۵, ۶, ۱, ۴] ترتیب از گره‌های مه می‌رسد.

Best Selected Schedule Fog Nodes: [۲, ۳, ۵, ۶, ۱, ۴]

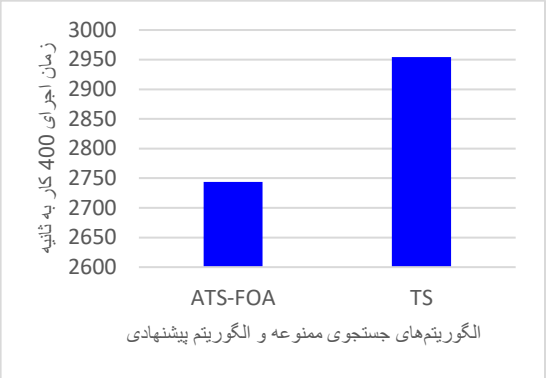
پیچیدگی زمانی هر دو الگوریتم نیز محاسبه شده که به شرح زیر است:

Time Complexity of Tabu Search:

$$T(n) = 1 + n^3 + n^2 \in O(n3)$$

Time Complexity of ATS-FOA:

$$T(n) = logn + n^3 + n \in O(n3)$$

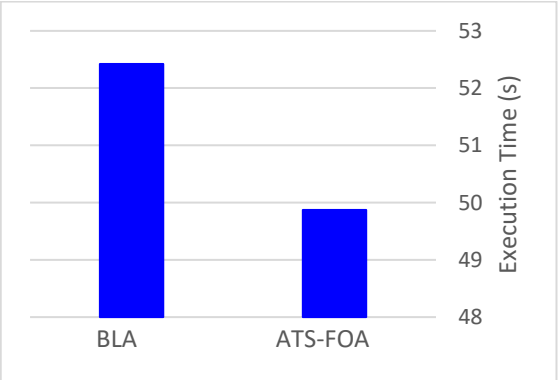


شکل ۹- زمان اجرای ۴۰۰ کار درخواست‌شده در پلتفرم پیشنهادی با سه لایه مه

بنابراین، پیچیدگی زمانی هر دو الگوریتم جست‌وجوی ممنوعه و الگوریتم پیشنهادی از مرتبهٔ n^3 است، ولی با توجه به نتایج آزمایش‌ها، زمان اجرای الگوریتم پیشنهادی، کمتر از جست‌وجوی ممنوعه است. از طرفی، تابع زمانی مربوط به الگوریتم پیشنهادی کوچک‌تر از تابع زمانی الگوریتم جست‌وجوی ممنوعه است:

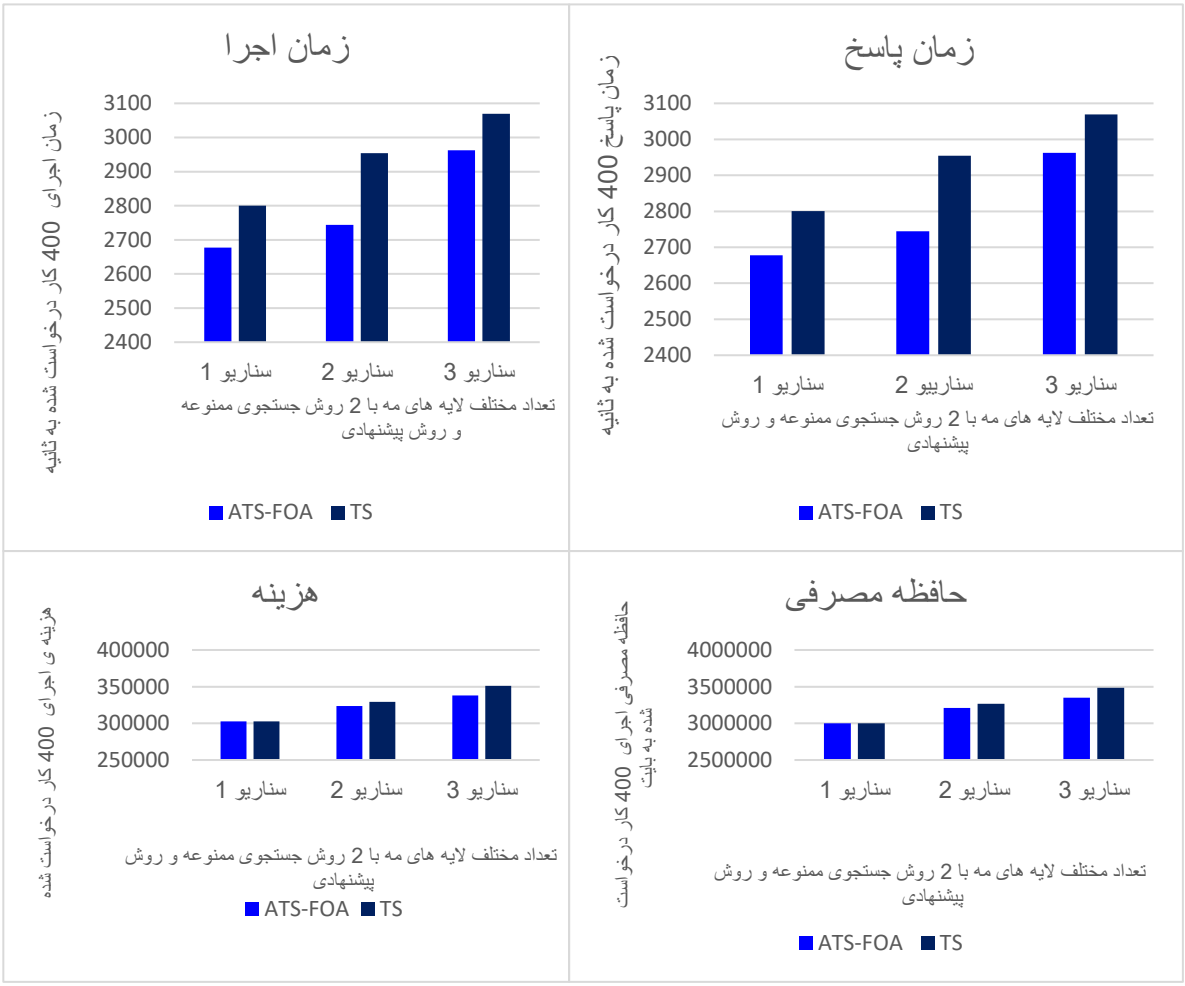
$$Logn + n3 + n \leq 1 + n^3 + n^2$$
$$n + logn \leq n^2$$

برای اعتبارسنجی به الگوریتم پیشنهادی در این مقاله، نتایج آن با نتایج الگوریتم پیشنهادی مقالهٔ [۷] تحت شرایطی مقایسه شده است. در مقالهٔ [۷]، پنج کار به سیستم ارسال شده و هر کار شامل پنج وظیفه و هر وظیفه از ۱۰۹ دستورالعمل تشکیل شده است. بنابراین، الگوریتم پیشنهادی این مقاله نیز با همین شرایط و با تعداد ۲۰ گره مه شبیه‌سازی و اجرا شده است و نتایج مقایسهٔ آن با الگوریتم BLA(Bees Life Algorithm) مقالهٔ مذکور در شکل (۱۱) نشان داده شده است.



شکل ۱۰۱- نتایج مقایسهٔ الگوریتم ATS-FOA و BLA.

همان طور که از شکل (۱۱) به دست می‌آید، زمان اجرای کل وظایف (۲۵ وظیفه) با تعداد ۱۰۹ دستورالعمل در زیرساختی با تعداد ۲۰ گره مه توسط الگوریتم ATS-FOA ۴۹,۸۷ ثانیه است، در حالی که اجرای این تعداد وظایف توسط الگوریتم BLA، ۵۲,۴۲ ثانیه است. بنابراین، الگوریتم پیشنهادی ATS-FOA عملکردی بهتر نسبت به الگوریتم BLA داشته است.



۵- نتیجه‌گیری

با توجه به اهمیت عوامل هزینه و زمان در دنیای امروز، با گسترش ارتباطات و تکنولوژی، روزبه‌روز حجم داده‌های تولیدشده در حال افزایش است که برای پردازش این حجم از داده، زمان و هزینه زیادی موردنیاز است. از طرفی، تکنولوژی‌ها در حال رشد به‌سمت سیستم‌هایی با سرعت بالا و تأخیر و هزینهٔ پایین هستند. در نتیجه، ایجاد سیستم‌هایی که بتوانند این حجم از داده را در حدّاقل زمان ممکن و با هزینهٔ پایین پردازش کنند، بسیار مورد نیاز است. برای ایجاد این سیستم‌ها، باید زیرساخت‌هایی ارائه شود که بتواند این اهداف را پشتیبانی کند. در این مقاله، زیرساختی با چهار عنصر مبتنی بر رایانش ابر و مه پیشنهاد شده و با استفاده از یک مطالعهٔ موردی روی خانه‌های هوشمند ارزیابی گردیده است. همچنین با پیاده‌سازی این پلتفرم برای خانهٔ هوشمند با داده‌های واقعی، مسئلهٔ زمان‌بندی وظایف درخواست‌شده در میان ماشین‌های مجازی و در میان گره‌های مه این پلتفرم نیز با استفاده از یک روش جدید پیشنهادی براساس جست‌وجوی ممنوعه انجام گردیده، به‌طوری که هزینه، زمان پاسخ، زمان اجرا و حافظهٔ مصرف‌شده با اجرای سناریوهای مختلف به دست آمده است. بنابر نتایج به‌دست‌آمده، با اضافه کردن گره‌ها و لایه‌های مه بیشتر و در نتیجه، ظرفیت پردازش و ذخیره‌سازی بیشتر به زیرساخت سیستم‌ها، هزینه و زمان افزایش می‌یابد. دلیل این افزایش آن است که با اضافه کردن لایه‌های بیشتر بین سرور ابر و دستگاه‌های هوشمند، تأخیر انتقال در شبکه بیشتر می‌شود و در نتیجه، زمان پاسخ افزایش می‌یابد. هزینه و حافظه مصرفی نیز با افزایش ظرفیت‌های ذخیره‌سازی بیشتر می‌شود. پس با توجه به نوع سیستم موردنظر و کاربرد

آن، زیرساخت مقرون به صرفه و با کارایی بالا باید طرّاحی شود. این روش پیشنهادی می‌تواند برای دیگر سیستم‌های شهر هوشمند مورد استفاده قرار گیرد. در کارهای آینده، می‌توان روش را با هدف میزان حافظهٔ مصرفی کمتر بهبود بخشید. همچینین می‌توان این زیرساخت و روش پیشنهادی را برای سیستم بانکی کشور و دیگر سازمان‌ها به کار برد.

مراجع

[1] B. Varghese and R. Buyya, "Next generation cloud computing: New trends and research directions", Future Generation Computer Systems, Vol. 79, 2018, pp. 849–861.

[2] M. Amadeo, A. Giordano, C. Mastroianni and A. Molinaro, "On the Integration of Information Centric Networking and Fog Computing for Smart Home Services", Springer, Cham, 2019, pp. 75–93.

[3] M. Marjani, F. Nasaruddin, A. Gani, A. Karim and I. Abaker, "Big IoT Data Analytics : Architecture , Opportunities , and Open Research Challenges", Vol. 3536, No. c, 2017, pp. 1–17.

[4] A. Mebrek, L. Merghem-boulahia and M. Esseghir, "Efficient Green Solution for a Balanced Energy Consumption and Delay in the IoT-Fog-Cloud Computing", 2017.

[5] S. Sarkar, S. Chatterjee and S. Misra, "Assessment of the Suitability of Fog Computing in the Context of Internet of Things", IEEE Transactions on Cloud Computing, Vol. 6, No. 1, 2018, pp. 46–59.

[6] A.N. Toosi, R. Mahmud, Q. Chi and R. Buyya, "Management and Orchestration of Network Slices in 5G, Fog, Edge, and Clouds", in Fog and Edge Computing, Hoboken, NJ, USA: John Wiley & Sons, Inc., 2019, pp. 79–101.

[7] S. Bitam, S. Zeadally and A. Mellouk, "Fog computing job scheduling optimization based on bees swarm", Enterprise Information Systems, Vol. 12, No. 4, 2018, pp. 373–397.

[8] A. Yassine, S. Singh, M.S. Hossain and G. Muhammad, "IoT big data analytics for smart homes with fog and cloud computing", Future Generation Computer Systems, Vol. 91, 2019, pp. 563–573.

[9] S.A.A. Naqvi, N. Javaid, H. Butt, M.B. Kamal, A. Hamza and M. Kashif, "Metaheuristic Optimization Technique for Load Balancing in Cloud-Fog Environment Integrated with Smart Grid", Springer, Cham, 2019, pp. 700–711.

[10] L. Peng, A.R. Dhaini and P.-H. Ho, "Toward integrated Cloud–Fog networks for efficient IoT provisioning: Key challenges and solutions", Future Generation Computer Systems, Vol. 88, 2018, pp. 606–613.

[11] P.G.V. Naranjo, Z. Pooranian, M. Shojafar, M. Conti and R. Buyya, "FOCAN: A Fog-supported smart city network architecture for management of applications in the Internet of Everything environments", Journal of Parallel and Distributed Computing, Jul. 2018.

[12] S.K. Sood and K.D. Singh, "SNA Based Resource Optimization in Optical Network using Fog and Cloud Computing", Optical Switching and Networking, Dec. 2017.

[13] S. Singh and A. Yassine, "IoT Big Data Analytics with Fog Computing for Household Energy Management in Smart Grids", Springer, Cham, 2019, pp. 13–22.

[14] Q. Li, L. Zhao, J. Gao, H. Liang, L. Zhao and X. Tang, "SMDP-Based Coordinated Virtual Machine Allocations in Cloud-Fog Computing Systems", IEEE Internet of Things Journal, Vol. 5, No. 3, 2018, pp. 1977–1988.

[15] Z. Zeng, X. Yu, K. He and Z. Fu, "Adaptive Tabu search and variable neighborhood descent for packing unequal circles into a square", Applied Soft Computing, Vol. 65, 2018, pp. 196–213.

[۱۶] مجید محمّدپور و حمید پروین، «الگوریتم کلونی زنبور مصنوعی آشوب‌گونه مبتنی بر حافظه برای حلّ مسائل بهینه‌سازی پویا»، مجلهٔ مدل‌سازی در مهندسی، دورهٔ ۱۵، شمارهٔ ۵۱، زمستان ۱۳۹۶، صفحهٔ ۱۱۳- ۱۳۲.

[۱۷] سید محمّدحسن حسینی و علی‌اکبر حسنی، «توسعهٔ یک الگوریتم شاخه و کران برای حلّ مسئلهٔ زمان‌بندی در سیستم تولید جریان کارگاهی مونتاژی»، مجلهٔ مدل‌سازی در مهندسی، دورهٔ ۱۵، شمارهٔ ۵۱، زمستان ۱۳۹۶، صفحهٔ ۸۵- ۹۸.

[۱۸] حسین شریف‌زاده و نیما امجدی، «توزیع بهینه توان رآکتیو با استفاده از الگوریتم بهینه‌سازی دسته ذرّت»، مجلهٔ مدل‌سازی در مهندسی، دورهٔ ۴، شمارهٔ ۱۸، زمستان ۱۳۸۸، صفحهٔ ۶۷- ۷۳.

[19] G.R. Raidl, J. Puchinger and C. Blum, Metaheuristic Hybrids.

[20] F. Glover, "Tabu Search—Part I", ORSA Journal on Computing, Vol. 1, No. 3, 1989, pp. 190–206.

[21] H. Pirim, E. Bayraktar and B. Eksioglu, "Tabu Search: A Comparative Study", Tabu Search, September 2008.

[22] F. Glover, "Tabu search fundamentals and uses", Vasa, 1995.

[23] T-61.5060 Algorithmic methods for data mining Slide set 5: locality-sensitive hashing.

[24] W.-T. Pan, "A new Fruit Fly Optimization Algorithm: Taking the financial distress model as an example", Knowledge-based Systems, Vol. 26, 2012, pp. 69–74.