علوم رایانش و فناوری اطلاعات
نشریه علمی انجمن کامپیوتر ایران
مجلد ۲۴، شماره ۱، ۱۴۰۵
صفحات ۱-۱۱
مقاله پژوهشی

بارسپاری وظایف آگاه از تحرک در محیط‌های پردازش مه سه لایه

سهیل مهدی‌زاده[۱]، محسن انصاری*[۲]

*نویسنده مسئول، دریافت: ۱۴۰۴/۰۳/۰۳، بازنگری: ۱۴۰۴/۰۷/۱۱، پذیرش: ۱۴۰۴/۰۷/۲۱

[۱] دانشجوی کارشناسی ارشد، دانشکده مهندسی کامپیوتر، دانشگاه صنعتی شریف
[۲] استادیار، دانشکده مهندسی کامپیوتر، دانشگاه صنعتی شریف

چکیده

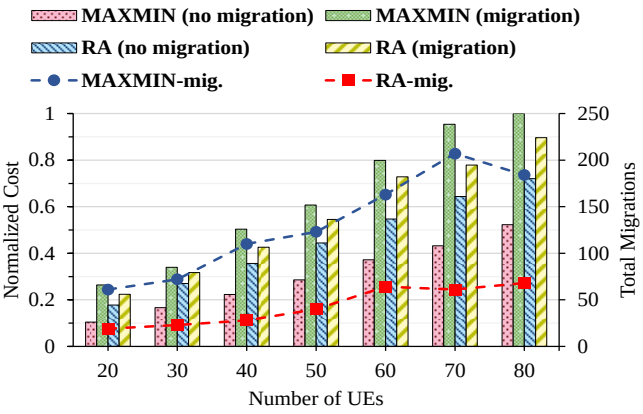
پردازش مه سه‌لایه با در بر داشتن لایه‌های ابر، مه و کاربر، تبدیل به یکی از کلیدی‌ترین بخش‌های معماری شبکه‌های اینترنت اشیاء شده است. ساختارهای قبلی، مانند پردازش ابر، بدلیل فاصله زیاد خدمت‌گزارها از دستگاه‌های کاربری باعث افزایش زمان اجرای وظایف می‌شدند و با اضافه لایه میانی مه می‌توان در اجرای وظایف تسریع ایجاد کرد. در این مقاله، قصد حل مسئله بارسپاری وظایف دستگاه‌های سیار کاربری را داریم که بدلیل تحرکشان ممکن است پدیده مهاجرت را رقم بزنند. پدیده مهاجرت، هزینه نهایی را بدلیل سربار ارسال وظیفه به لایه ابر، افزایش می‌دهد. مسئله بارسپاری و تخصیص منابع به یک مسئله MINLP مدل شده است با استفاده از روش پیشنهادی مبتنی بر الگوریتمی ابتکاری و ژنتیک حل شده است. مدل تحرک دستگاه‌ها با استفاده از شبیه‌سازی SUMO بازسازی شده است و الگوریتم به طور متوسط ۱۶٪ در بهبود هزینه نسبت به روش‌های پیشین مؤثر بوده است. به طور خلاصه پارامتر هزینه شامل انرژی و تأخیر کل سیستم است.

کلمات کلیدی: اینترنت اشیاء، پردازش مه، بارسپاری وظایف، تحرک، مهاجرت، تأخیر، انرژی، تخصیص منابع

۱- مقدمه

با پیشرفت سریع ارتباطات دستگاه‌های سیار، سرویس‌های مانند اینترنت اشیاء و به دنبال آن اینترنت وسایل نقلیه ظهور پیدا کردند. هر دوی این فناوری‌های، تقاضای بیشتر منابعی مانند پهنای باند و ظرفیت پردازشی را از زیرساخت بی‌سیم شبکه خواستارند [۱]. به همین دلیل نسل پنجم سیستم‌های بی‌سیم توسعه یافت [۲]. در ابتدا، وظایف بر روی دستگاه‌های کاربری به صورت محلی اجرا می‌شدند. بدلیل محدودیت قدرت محاسباتی و همچنین توان محدود و مبتنی بر باتری، روشی تحت عنوان پردازش ابر معرفی شد [۱]. در این معماری، وظیفه‌ها با استفاده از شبکه Cloud radio access یا C-RAN به مراکز داده ارسال می‌شدند و انرژی مصرفی محلی کاهش پیدا کرد و منبع پردازشی زیادی در اختیار دستگاه‌های کاربری قرار گرفت. اما فاصله زیاد مراکز داده ابر و دستگاه‌های کاربری باعث شد که تأخیر ارسال وظایف بالا رود و اجرای وظایف حساس به تأخیر به درستی پیش نرود [۲].

برای رفع مشکلات تأخیر در معماری دو لایه پردازش ابر، شبکه Fog radio access یا F-RAN معرفی شد [۲]. پردازش مه، مانند لایه‌ای میانی بین ابر و دستگاه‌های کاربری عمل می‌کند و منابع پردازشی را نزدیکتر به کاربران ارائه می‌دهد. در نتیجه وظایف حساس به زمان اجرا، در این معماری می‌توانند به موقع اجرا شوند [۳]. مثال‌هایی از این وظایف شامل سیستم مدیریت ترافیک، شبکه خودروهای خودران و سنسورهای تشخیص مخاطره‌های محیطی است. در بهداشت و درمان هوشمند نیز، دستگاه‌های مانیتور بیمار برای اجرای وظایف به موقع و حساس به زمان اجرا به لایه مه اتکا می‌کنند [۴]. هم‌چنین بدلیل مصرف بهتر توان در واحدهای لایه مه، انرژی کل سیستم نیز با بارسپاری به این لایه می‌تواند بهبود پیدا کند.



شکل ۱- هزینه کل روش‌های پیشین حریصانه و تصادفی با در نظر گرفتن یا نگرفتن مهاجرت

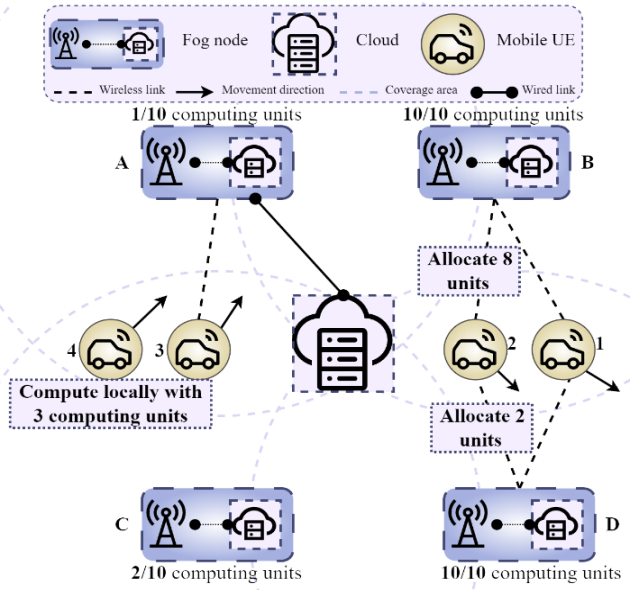
هدف اصلی معماری سه لایه پردازش مه، بارسپاری بهینه وظایف دستگاه‌های کاربری به لایه مه است که انتخاب واحد مه مقصد به عنوان خدمت‌گزار وظیفه تأثیر زیادی در تأخیر کل سیستم و توازن بار کاری دارد [۴]-[۷]. مقالات و مطالعات پیشین زیادی در این حوزه ارائه شده‌اند [۱۳]-[۱۹]. هم‌چنین کارهای پیشینی بوده‌اند که مسئله تصمیم‌گیری بارسپاری را توأم با تخصیص منابع برطرف کرده‌اند. اما با وجود این مطالعات، دستگاه‌های کاربری متحرک، مسئله جدید و پیچیده‌تری را ایجاد می‌کنند. یکی از پیچیدگی‌ها پدیده مهاجرت است.

بدلیل تحرک دستگاه‌ها، یک کاربر ممکن است شعاع پوشش یک واحد مه را ترک کنند که در نتیجه آن، نتیجه پردازش وظیفه مجبور از مسیر متفاوتی را برای برگشت به دستگاه طی کند و انرژی و تأخیر سیستم را افزایش می‌دهد [۸]، [۹]. این پدیده، ممکن است اتفاقات غیرقابل پیشبینی مانند شکست مهاجرت که در اثر مهاجرت دیر یا زود به واحد مه جایگزین رخ می‌دهد و اشغال حافظه واحدهای مه را نیز رقم بزند که باعث افزایش تأخیر و انرژی مصرفی سیستم می‌شوند.

۱-۱- مشاهده

برای نشان دادن نقش بحرانی پدیده مهاجرت در پردازش‌های سه لایه مه دستگاه‌های کاربری سیار، سناریوی واقعی با ابزار SUMO را شبیه‌سازی کرده‌ایم [۱۰] و نتایج آن در شکل ۱ رسم شده‌اند. توضیح جزئی و دقیق مدل سیستم و محیط شبیه‌سازی در بخش‌های بعدی آورده شده است، ولی فرض کنید که الگوریتم‌های پیشین RA و MAXMIN پدیده مهاجرت را در نظر نمی‌گیرند. الگوریتم RA به طور تصادفی جوابی برای مقصد بارسپاری در نظر می‌گیرد و MAXMIN نیز به طور حریصانه واحدهای مه را اولویت قرار می‌دهد. ابتدا یکبار این الگوریتم‌ها را بدون در نظر گرفتن هزینه اضافی مهاجرت شبیه‌سازی می‌کنیم و بار دیگر این پدیده را در نظر می‌گیریم. همانطور که از نمودار پیداست، با افزایش تعداد دستگاه‌های کاربری، تعداد کل مهاجرت‌های رخ داده برای وظایف افزایش پیدا می‌کند و هزینه زیادی به سیستم اضافه می‌شود. پس نمی‌توان از این پدیده صرف‌نظر کرد و باید الگوریتم ارائه شده، به خصوص در شرایطی که ازدحام شبکه بالاست، این پدیده را در نظر داشته باشد.

برای طراحی یک الگوریتم بارسپاری مؤثر، تصمیم‌گیرنده باید موارد و فاکتورهای اضافی مانند منبع باقی مانده واحدهای پردازشی و مدت زمان وجود یک دستگاه در شعاع پوششی یک واحد مه را در نظر بگیرد. برای نشان دادن تأثیر این پارامترها، در بخش بعدی، مثال انگیزشی تنظیم گردیده است.

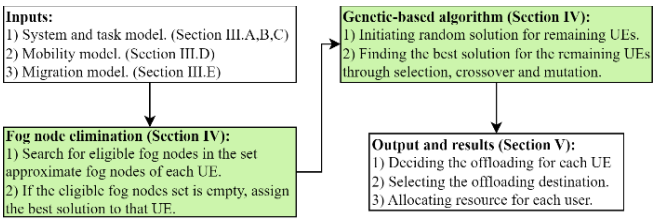


شکل ۲- چهار سناریو مختلف در محیط پردازش مه سه لایه دارای تحرک

۱-۲- مثال انگیزشی

در این بخش، چهار سناریو مختلف ارائه می‌شود که در معماری سه لایه ممکن است اتفاق بیفتد و تأکید بر روی متحرک بودن دستگاه‌های کاربری دارد. خواهیم دید که چگونه تحرک دستگاه‌های کاربری بر روی مسئله تصمیم‌گیری بارسپاری و تخصیص منابع تأثیر می‌گذارد. شکل ۲ این سناریوها را نشان می‌دهد.

- سناریو ۱: در این سناریو، اگر مبحث تخصیص منابع را در نظر نگیریم، خودرو متحرک ۱ تحت پوشش دو واحد مه B و D است و در حال حرکت به سمت محدوده واحد مه D است. در این حالت، تصمیم منطقی، بارسپاری وظیفه به واحد مه D است چراکه در غیر این صورت ممکن است خودرو از محدوده واحد مه B خارج شود و مهاجرت رخ دهد. اما اگر بارسپاری بر روی واحد مه D صورت گیرد، با احتمال بیشتری هنگامی که اجرای وظیفه تمام می‌شود، خودروی ۱ در محدوده مقصد بارسپاری خود قرار دارد.
- سناریو ۲: مشابه سناریو قبل، دستگاه کاربری متحرک ۲ در شعاع پوششی دو واحد مه B و D قرار دارد. اما این بار، سیاست تخصیص منابع را نیز وارد مسئله می‌کنیم. فرض کنید که پس از بارسپاری بر روی B، ۸ واحد پردازشی برای این وظیفه تخصیص دهیم و پس از بارسپاری بر روی D، ۲ واحد پردازشی برای آن تخصیص دهیم. در این صورت زمان اجرای وظیفه در این دو روش متفاوت است و بارسپاری بر روی D احتمال مهاجرت بیشتری دارد چراکه زمان اجرای آن بیشتر است و ممکن است، با اینکه در حال حرکت به سمت آن هستیم، خودرو از محدوده آن خارج شود. اما، با اینکه دستگاه کاربری در حال فاصله گرفتن از واحد مه B است، بارسپاری روی B زمان اجرای بسیار کمتری دارد و احتمال اینکه مهاجرت رخ دهد پایین‌تر است.
- سناریو ۳: در این سناریو، دستگاه کاربری در محدوده پوشش دو واحد مه A و C قرار دارد. اما بدلیل تخصیص منابعی که وظایف دیگر در گذشته داشته‌اند، منبع باقی‌مانده روی این دو واحد مه محدود است. بارسپاری بر روی هر یک از این واحدهای مه ایده‌آل نیست چراکه در آن صورت، خودرو ممکن است از محدوده پوشش هر دو گره مه خارج شود. در این حالت، بارسپاری به لایه ابر می‌تواند سودمند باشد چون هم انرژی مصرفی را از بین می‌برد و احتمال مهاجرت را به صفر می‌رساند، اما زمان اجرا را افزایش می‌دهد.



شکل ۳- ساختاربندی کلی مقاله

• سناریو ۴: مشابه سناریو ۳، دستگاه کاربری تحت پوشش A و C است که منابع پردازشی محدودی دارند. راه‌حل دیگر این است که بارسپاری را انجام ندهیم و دستگاه کاربری به صورت محلی وظیفه را اجرا کند. در این صورت، احتمال مهاجرت صفر است و تأخیر ارسال هم نداریم. هزینه اضافی این روش در مصرف انرژی است که سیستم مبتنی بر باتری تحمل خواهد کرد. اما مشابه بارسپاری به ابر، اجرای محلی نیز می‌تواند سودمند باشد چون مهاجرت را از معادله حذف می‌کند.

در این چهار سناریو، چالش‌ها و پیچیدگی‌های مسئله را بررسی کردیم و بر روی نقش اساسی تحرک دستگاه‌های کاربری نیز تأکید کردیم. هم‌چنین فایده و ضرر یا مبادله‌ای که به هنگام بارسپاری به دو لایه یا اجرای محلی داریم را نیز توضیح دادیم.

۱-۳- دستاوردها

در حالی که مطالعات پیشین، به مبحث بارسپاری وظایف پرداخته‌اند و تأخیر و انرژی را بهبود داداند [۴]-[۷]، [۱۱]-[۱۹]، بیشتر آن‌ها تحرک دستگاه‌ها را به خوبی مدل نکرده‌اند. مقاله ما، در مدل اجرا و شبیه‌سازی خود، سناریو واقعی و فیزیکی را از محیط یک شهر مدل می‌کند و مکان جغرافیایی و سرعت و جهت دستگاه‌های کاربری را در نظر می‌گیرد. هم‌چنین پدیده مهاجرت نیز مدل شده است [۸]، [۹] و هزینه‌ای که به سیستم وارد می‌کند را نیز در نظر می‌گیریم. کارهای پیشینی موجود است که تمرکز بر روی خود پدیده مهاجرت داشته‌اند و هزینه این پدیده را کاهش داده‌اند، اما تأکید کار ما در کاهش تعداد وقوع این پدیده است چراکه هزینه مهاجرت حتی اگر کم نیز باشد، اگر الگوریتم بارسپاری آن را در نظر نگیرد، بدلیل تحرک زیاد خودروها، هزینه کل تمام شده در آخر غیرقابل صرف نظر است. هم‌چنین مدل ما تصمیم بارسپاری را بر اساس محیط فیزیکی در یک لحظه انجام می‌دهد و مدل‌های احتمالی را وارد شبیه‌سازی نمی‌کند. نتایج نیز نشان می‌دهد که الگوریتم راه‌حلی نزدیک بهینه در محیطی کاملا پویا ارائه می‌دهد. به طور خلاصه، نوآوری‌های مقاله عبارتند از:

۱. در بخش ۳، مدل سیستم سه لایه شامل دستگاه کاربری، مه و ابر طراحی می‌شود و هم‌چنین تحرک دستگاه‌های کاربری مدل می‌شود. در این مدل سازی، گراف شبکه و واحدهای مه در اختیار هر دستگاه کاربری به صورت پویا تغییر می‌کنند. هم‌چنین سیستم قادر به شبیه‌سازی اجرای وظایف برای بازه زمانی مشخصی می‌باشد.

۲. در بخش ۴، الگوریتمی مبتنی بر روش‌های ابتکاری و فراابتکاری ارائه می‌شود که با استفاده از آن مسئله تصمیم بارسپاری، مقصد بارسپاری و تخصیص منابع حل می‌شود. این مسئله ابتدا به صورت MINLP مدل می‌شود که مسئله‌ای NP-hard است و سپس با

۳. الگوریتم ذکر شده جواب نزدیک بهینه با هدف حداقل کردن هزینه نهایی که ترکیبی از انرژی مصرفی سیستم و تأخیر است محاسبه می‌شود. در این الگوریتم، هزینه اضافی پدید مهاجرت نیز لحاظ می‌شود.

۴. در آخر، در بخش ۵، روش پیشنهادی شبیه‌سازی می‌شود و مدل تحرک دستگاه‌ها نیز از یکی از سناریوهای واقعی شبیه‌ساز معروف SUMO گرفته می‌شود و میزان اثرگذاری روش در یک بازه زمانی طولانه در محیطی با ترافیک بالا سنجیده می‌شود.

ساختار کلی مقاله در شکل ۳ آمده است. به طور کلی، در بخش ۲ کارهای پیشین و تفاوت و شباهت‌های آن‌ها با کار این مقاله توضیح داده می‌شوند. در بخش ۳ مدل سیستم پیشنهادی و بخش‌های آن توضیح داده می‌شوند. بخش ۴ الگوریتم و روش پیشنهادی برای حل مسئله را با جزئیات توضیح می‌دهد. در بخش ۵، شبیه‌سازی تحت شرایط محیطی و شبکه‌ای متفاوت انجام می‌شود و کارایی روش پیشنهادی با مقایسه با روش‌های قبلی سنجیده می‌شود.

۲- کارهای پیشین

در این بخش کارهای پیشین به ۳ مجموعه دسته‌بندی شده‌اند و تمام جنبه‌های مختلف مسئله بررسی شده است.

۲-۱- بارسپاری وظایف و تخصیص منابع در شبکه‌های پردازش مه

تمرکز اصلی تمام کارهای پیشین حول محور بارسپاری و تخصیص منابع است. در مطالعاتی مانند [۱۱]، مسئله بارسپاری با روشی مبتنی بر نظریه بازی حل شده است. در [۱۲]، نویسندگان مسئله بارسپاری جزئی را به هدف بهبود انرژی و تأخیر حل کرده‌اند. کار ارائه شده در [۱۳] تمرکزش بر روی کاهش هزینه دستگاه‌های کاربری است و قصد داشته است که بیشترین هزینه دستگاه‌های کاربری را کمینه کند. در [۱۴] الگوریتم FJORA مسئله بارسپاری و تخصیص منابع و تخصیص توان را حل کرد. در [۱۵] نویسندگان با پیدا کردن تعادل نش، سعی در بهبود کیفیت خدمات دستگاه‌های کاربری بودند. درحالی که تمام کارهای ذکر شده روی مسئله بارسپاری تمرکز کرده بودند، اما هیچ یک محیط متحرکی شبیه‌سازی نکرده بود.

۲-۲- آگاهی از تحرک در شبکه‌های پردازش مه

در این بخش به مسئله اصلی پردازش مه در محیط‌های متحرک پرداخته شده است. بعضی از کارهای مدل‌سازی محدود و بعضی دیگر تحرک واقعی دستگاه‌های کاربری را در نظر می‌گیرند.

در [۱۶]، الگوریتمی برای استریم کردن ویدئو ارائه شده است که در آن دستگاه‌های کاربری با بارسپاری مبتنی بر همکاری، هزینه خود را بدون استفاده کردن از مداخله فرد سومی، کاهش می‌دهند. نوآوری اصلی مقاله [۱۷] شامل ارائه الگوریتم تخصیص منابع با استفاده از یادگیری تقویتی عمیق با اعمال محدودیت‌های تأخیر بر روی لینک‌های V2V و هم‌چنین تعریف فضای حالت، عمل و تابع پاداش در سناریوهای متحرک می‌باشد. در [۱۸] الگوریتمی پویا با سربار زمانی کم ارائه شد که منابع پردازشی را با در نظر داشته وابستگی وظایف تخصیص می‌داد. نویسندگان مقاله [۱۹] الگوریتمی تحت عنوان JTORA ارائه می‌دهند که مسئله بارسپاری، توان لینک و تخصیص منابع را در محیط پردازش لبه سیار حل می‌کرد.

درحالی که کارهای ذکر شده در محیطی متحرک مطرح می‌شوند، هیچ‌یک از مقالات مدلی تحت عنوان تحرک برای دستگاه‌های کاربری در نظر نگرفته‌اند. اما در مقاله [۲۰]، دو نوآوری وجود دارد. اولی معرفی الگوریتمی برای پیشبینی زمان دسترسی کاربران است که با استفاده از نقش قبلی دسترسی‌ها محاسبه می‌شود و در هنگام تصمیم‌گیری بارسپاری از آن استفاده می‌شود. دومی، ارائه الگوریتمی مبتنی بر ژنتیک است که مسئله مقصد بارسپاری و هم‌چنین تخصیص منابع را با در نظرگرفتن انرژی مصرفی دستگاه‌های کاربری حل می‌کند. کار مطرح شده در [۲۱]، محیط با فضای حالت بزرگ این مسئله با استفاده از یادگیری تقویتی و

الگوریتمی ابتکاری حل شده است که سرعت یادگیری مدل را افزایش می‌داد. مقاله [۲۲] دستگاه‌ها کاربری مبتنی بر وسایل نقلیه، به صورت هوشمند با تبادل اطلاعات بین همدیگر و اجرا شدن وظایف بر روی وسایل نقلیه دیگر یا زیرساخت‌ها، مسئله را حل کرده بود. مدل تحرکی که در این مقاله در نظر گرفته شده است، پیشبینی مکان است و از الگوریتم EKF برای پیشبینی مکان آن‌ها استفاده می‌کند.

یکی از مدل‌های مرسوم تحرک، در نظر گرفتن دستگاه سیار بر روی یک جاده بدون پیچ و خم است. در این مدل، شتاب را نیز برای کاربر یکنواخت مدل می‌کنند. کارهای [۲۳]-[۲۵] در این مدل استفاده می‌کنند. نویسندگان در [۲۳] نوآوری در محاسبه منبع پردازشی هر خودرو همسایه با استفاده از مسیر اتصال و دفعات ارسال است. در کار مقاله [۲۴] مدل ارسال وظیفه به صورت چند پرشی است و چالش اصلی در پیدا کردن بهترین خودروی کاندید برای اجرای وظایف است. نویسندگان در [۲۵] مدل مبتنی بر نظریه بازی ارائه کرده‌اند که همکاری و رقابت را بین سرورهای لبه و ابر و دستگاه‌های کاربری متحرک در نظر می‌گیرد و مسئله بارسپاری وظایف به مسئله بیشینه کردن سود یا همان پاداش سیستم تبدیل می‌شود. کار مقاله [۲۶] مدل پردازش مبتنی بر لبه و خودرو ارائه می‌دهد و الگوریتم آن قصد دارد که انرژی مصرفی و تأخیر سیستم را با در نظر گرفتن وابستگی وظایف کمینه کند.

۲-۳- مهاجرت در شبکه‌های پردازش مه

روش Folo که در مقاله [۲۷] معرفی می‌شود پدیده مهاجرت را در نظر گرفته است و مسئله بارسپاری را به شکل بهینه‌سازی توام تأخیر و کیفیت خدمات مدل می‌کند. راه حلی که ارائه می‌دهد بر پایه الگوریتم LBO و BPSO است. نویسندگان در [۲۸]، که نزدیک‌ترن کار پیشین به مدل پیشنهادی ماست، در محیط دولایه پردازش مه، مدلی آگاه از تحرک و مهاجرت ارائه می‌دهد. در مدل تحرک این مقاله، زمان ماندگاری هر دستگاه در محدوده واحدهای مه پیشبینی می‌شود. الگوریتم مقاله نیز به دو مرحله تصمیم بارسپاری و تخصیص منابع شکسته شده است. مرحله بارسپاری با استفاده از الگوریتمی مبتنی بر Gini Coefficient حل می‌شود و مرحله تخصیص منابع با استفاده از الگوریتم فراابتکاری ژنتیک حل می‌گردد. کمبود این مدل، در نظر گیری لایه ابر است که می‌تواند برای وظایفی که حساسیت زمانی ندارند مفید باشد و هم‌چنین به کاهش انرژی کل سیستم کمک کند. به علاوه، زمان اقامت که مدل تحرک دستگاه‌های کاربری است، ممکن است در شرایط غیر قابل پیشبینی محیطی، به خوبی تحرک را مدل نکند و دقت پایینی داشته باشد. خلاصه مقالات بررسی شده در جدول ۱ آورده شده است و نوآوری‌های اصلی هر مقاله ارائه شده‌اند.

مدل سه لایه‌ای که مقاله ما در نظر می‌گیرد، انعطاف‌پذیری زیادی ارائه می‌کند. به علاوه در نظر گرفتن مکان و سرعت هر خودرو کاربری امکان تصمیم‌گیری با دقت بالایی را در محیط‌های پرترافیک واقعی فراهم می‌کند. پدیده مهاجرت را نیز در نظر می‌گیریم تا هزینه اضافی آن را نیز به حداقل برسانیم.

۳- مدل سیستم

به طور کلی مدل سیستم در نظر گرفته شده سه لایه است. لایه دستگاه‌های کاربری شامل کاربران متحرکی هستند که در یک محیط جغرافیایی در حال حرکت‌اند. لایه مه شامل واحدهای مه ثابت است که در نقاط مختلفی از این محیط به طور ثابت قرار دارند و شعاعی حول خود را پوشش می‌دهند. لایه ابر نیز قدرت پردازشی بدون محدودیت را در اختیار تمام کاربران قرار می‌دهد ولی تأخیر ارسال آن شامل ۲ پرش است. اولین پرش به واحد مه نزدیک و دیگری از واحد مه به ابر که زمان زیادی می‌برد. هر دستگاه کاربری می‌تواند از طریق اتصال بی‌سیم به یکی از واحدهای مه وظیفه خود را بارسپاری کند و یا اینکه به صورت محلی پردازش را انجام دهد. تأخیر نهایی هر وظیفه شامل تأخیر ارسال و پردازش در واحد مه است. انرژی مصرفی نیز شامل انرژی دستگاه‌های کاربری در صورت اجرای محلی یا ارسال وظیفه به لایه مه است.

مجموعه دستگاه‌های کاربری $\mathcal{N}^u = \{N_1^u, N_2^u, \dots, N_U^u\}$ است و هم‌چنین مجموعه واحدها مه نیز $\mathcal{N}^f = \{N_1^f, N_2^f, \dots, N_F^f\}$ است. در بخش‌های بعدی، به ارتباطات و جزئیات این مدل می‌پردازیم.

۱-۳- مفروضات مدل

با پیشرفت فناوری‌های حوزه ارتباطات، یکی از مفروضات مهم مدل در نظر گرفته شده، وجود شبکه نرم‌افزار محور است. تصمیم‌گیری‌ها در یک کنترل‌کننده مرکزی مبتنی بر SDN و خارج از مسیر داده انجام می‌شود، بنابراین سربار محاسباتی مستقیماً بر تأخیر وظایف اثر ندارد. بر این اساس تمرکز پژوهش بر دقت تصمیمات و هزینه بوده و پیچیدگی زمانی الگوریتم نیز در مقاله تحلیل شده است.

در این پژوهش هزینه انتقال نتیجه (فاز دانلود) در نظر گرفته نشده است. دلیل اصلی آن تمرکز بر فرآیند اجرای وظیفه و مراحل ارسال و اجراست که بیشترین وابستگی را به منابع پردازشی و مقصد بارسپاری دارند، در حالی که فاز دریافت نتیجه تأثیر مستقیمی بر تصمیم‌گیری بارسپاری ندارد. این ساده‌سازی در بسیاری از مقالات پیشین مانند [۲۸] نیز پذیرفته شده و با هدف تمرکز بر حل مسئله بارسپاری در یک سیستم سه‌لایه متحرک منطقی است. این فرض به‌ویژه در کاربردهایی مانند پردازش بلادرنگ حسگرهای محیطی یا سیستم‌های هوشمند حمل‌ونقل که حجم داده‌ی خروجی ناچیز و زمان‌بری انتقال نتیجه بسیار کم است، معتبرتر خواهد بود. یکی دیگر از دلایل این فرض وضعیت بهتر دانلینک نسبت به آپلینک است.

۲-۳- مدل پردازش محلی

هر دستگاه کاربری وظیفه T_i را دارد که با سه تایی $\{\tau_i, f_i, D_i\}, \forall N_i^u \in \mathcal{N}^u$ نشان داده می‌شود و در آن τ_i دوره وظیفه است، f_i تعداد سایکل‌های مورد نیاز برای اجرای وظیفه است و D_i نیز اندازه وظیفه به بیت است. با توجه به کارهای پیشین متعدد مانند [۱۵] و [۲۸]، f_i می‌تواند به صورت $f_i = \epsilon_i D_i$ محاسبه شود که ϵ_i مشخص کننده بارکاری پردازشی وظیفه است و تعداد سایکل به ازای هر بیت را مشخص می‌کند. بر اساس این مدل برای اجرای محلی فرمول زیر را خواهیم داشت:

$$T_i^{local} = \frac{f_i}{c_i^u}, \forall N_i^u \in \mathcal{N}^u \qquad (۱)$$

که در آن c_i^u قدرت پردازش کل کاربر N_i^u است. یک روش برای محاسبه مصرف انرژی، استفاده از فرمول $E = \kappa c^2$ است [۲۹]-[۳۱]. که در آن κ خازن موثر سوئیچ است و به معماری تراشه وابسته است. هم‌چنین c فرکانسی است که به آن وظیفه اختصاص داده می‌شود. طبق این فرمول که انرژی مصرفی در هر سایکل را می‌دهد، انرژی مصرفی محلی به صورت زیر قابل محاسبه است:

$$E_i^{local} = \kappa (c_i^u)^2 f_i, \forall N_i^u \in \mathcal{N}^u \qquad (۲)$$

با توجه به معادلات (۱) و (۲)، و اینکه نیازی به انتقال در پردازش محلی نیست، هزینه اجرای وظیفه به صورت زیر قابل تعریف است:

$$\phi_i^{local} = \lambda^T T_i^{local} + (1 - \lambda^T) E_i^{local}, \forall N_i^u \in \mathcal{N}^u \qquad (۳)$$

که λ^T ثابت وزن است و حساسیت وظیفه به تأخیر زمان اجرا را مشخص می‌کند و داریم $\lambda^T \in [0,1]$ اگر $\lambda^T > 0.5$، وظیفه به زمان اجرا حساسیت زیادی دارد، مانند وظایف مربوط به استریم ویدئو یا ارتباطات. اگر $\lambda^T < 0.5$، وظیفه حساسیتش نسبت به انرژی بالاتر می‌رود. در حالت‌های شدید، ممکن است

λ^T صفر یا ۱ هم باشد که یعنی هزینه وظیفه به ترتیب از تأخیر یا انرژی مصرفی صرف نظر می‌کند.

۳-۳- مدل پردازش مه

واحدهای مه وظایف دستگاه‌های کاربری را بعد از بارسپاری قبول می‌کنند و مقداری از منبع پردازشی را به آن‌ها تخصیص می‌دهند. این فرایند شامل انتخاب شدن واحد مه توسط دستگاه کاربری، فرستادن داده وظیفه به آن از طریق ارتباط بی‌سیم و اجرای وظیفه است. در این مدل، همانند کارهای پیشینی مثل [۲۸]، انتقال نتیجه به دستگاه کاربری مشابه کارهای پیشین متعدد در نظر گرفته نشده است زیرا اندازه داده کم است و هم‌چنین وضعیت کانال دانلینک از آپلینک بهتر است. در این مدل ما فرض می‌کنیم که هر واحد مه با چندین دستگاه کاربری تا وقتی که در محدوده پوشش آن هستند، می‌تواند ارتباط برقرار کند. این تکنولوژی تحت عنوان OFDMA معرفی می‌شود. اما دستگاه‌های کاربری تنها می‌توانند وظیفه را به یک مقصد ارسال کنند. برای محاسبه انرژی و تأخیر ناشی از ارسال بی‌سیم، ما SINR یا همان signal-to-interference-plus-noise-ration را در نظر می‌گیریم که برای خودرو i و واحد مه j برابر است با $SINR_{ij} = \frac{P_i^u}{\sigma^2 + I_{ij}}$ [۳۴]، که P_i^u توان انتقال کل دستگاه کاربری است، σ^2 نویز توان، I_{ij} نیز ناشی از تداخلی است که توسط دستگاه‌های کاربری که از همان کانال N_i^u استفاده می‌کنند. به این ترتیب نرخ ارسال از N_i^u به N_j^f مطابق زیر تعریف می‌شود:

$$r_{ij} = W \log_2\left(1 + SINR_{ij}\right), \forall N_i^u \in \mathcal{N}^u, \forall N_j^f \in \mathcal{N}^f \quad (۴)$$

که در آن W پهنای باند کانال بی‌سیم است. حال که نرخ ارسال از N_i^u به N_j^f تعیین شد، تأخیر مطابق زیر محاسبه می‌شود:

$$T_{ij}^{trans} = \frac{D_i}{r_{ij}}, \forall N_i^u \in \mathcal{N}^u, \forall N_j^f \in \mathcal{N}^f \quad (۵)$$

حال با استفاده از زمان انتقال و توان تخصیص داده شده به انتقال، به صورت زیر انرژی انتقال را حساب می‌کنیم:

$$E_{ij}^{trans} = P_i^u T_{ij}^{trans}, \forall N_i^u \in \mathcal{N}^u, \forall N_j^f \in \mathcal{N}^f \quad (۶)$$

هر وظیفه می‌تواند به واحد مه یا به لایه ابر بارسپاری شود. این تصمیم که تحت عنوان تصمیم بارسپاری مطرح می‌شود برای دستگاه کاربری i با $s_i \in \{0,1\}$ نشان داده می‌شود. هم‌چنین مجموعه تمام بارسپاری‌ها به صورت زیر تعریف می‌شود:

$$S = \{s_i | N_i^u \in \mathcal{N}^u\} \quad (۷)$$

برای مقصد بارسپاری، دو مقصد لایه مه یا ابر را داریم. برای بارسپاری به یک واحد مه، فرایند انتخاب توسط $a_{ij} \in \{0,1\}$ تعیین می‌شود که نشان می‌دهد واحد مه j توسط دستگاه کاربری i انتخاب شده است یا خیر. مجموعه تمام انتخاب‌های لایه مه به صورت زیر است:

$$A^f = \{a_{ij} | N_i^u \in \mathcal{N}^u, N_j^f \in \mathcal{N}^f\} \quad (۸)$$

پس از انتخاب مقصد، باید فرکانس برای وظیفه تخصیص یابد. مجموعه تمام تخصیص‌ها برای واحد مه j به صورت زیر است:

$$C_j = \{c_{ij}^f | N_i^u \in \mathcal{N}^u\} \quad (۹)$$

هر واحد مه یک بیشینه فرکانس دارد که با c_j^f نشان می‌دهیم و برای واحدهای مه متفاوت است. تخصیص منابع باید این مقدار را مد نظر داشته باشد و از آن عبور نکند. این محدودیت به شکل زیر قابل توصیف است:

$$\sum_{i=1}^{U} c_{ij}^f \le c_j^f \quad (۱۰)$$

مجموعه تمام استراتژی‌های تخصیص به شکل زیر است:

$$C = \bigcup_{j=1}^{F} C_j \quad (۱۱)$$

مشابه اجرای محلی، زمان اجرا مطابق زیر تعریف می‌شود:

$$T_i^j = \frac{f_i}{c_{ij}^f} \quad (۱۲)$$

در آخر، هزینه در این حالت پردازش، مطابق زیر خواهد بود:

$$\phi_i^j = \lambda^T \left(T_i^j + T_{ij}^{trans}\right) + (1 - \lambda^T) E_{ij}^{trans}, \forall N_i^u \in \mathcal{N}^u \quad (۱۳)$$

۴-۳- مدل پردازش ابر

پردازش مه برای تکمیل ساختار و معماری دو لایه قبلی اضافه شده است به همین دلیل مدل این مقاله، پردازش ابر را نیز در نظر دارد. بارسپاری به ابر در صورت مشغول بودن واحدهای مه می‌تواند سودمند باشد. انتخاب ابر به عنوان مقصد بارسپاری با $a_{ic} \in \{0,1\}$ مشخص می‌شود که در آن a_{ic} نشان‌دهنده انتخاب شدن ابر توسط کاربر i است. به این ترتیب مجموعه تمام بارسپاری‌های به ابر به صورت زیر قابل تعریف است:

$$A^c = \{a_{ic} | N_i^u \in \mathcal{N}^u\} \quad (۱۴)$$

فرایند بارسپاری به ابر به این ترتیب است که ابتدا دستگاه کاربری از طریق ارتباط بی‌سیم به یکی از واحدهای مه وظیفه را ارسال می‌کند و سپس از طریق ارتباط فیزیکی و باسیم به ابر می‌رسد که تأخیر زیادی را به تأخیر کل اجرای وظیفه اضافه می‌کند. دلیل این موضوع نیز مقدار زیاد تأخیر انتشار در backbone شبکه است. از آنجایی که مقدار و محدودیتی روی منبع پردازشی لایه ابر نداریم، وظیفه به هنگام رسیدن به لایه ابر تمام شده در نظر گرفته می‌شود و انرژی مصرفی بارسپاری به لایه ابر نیز تنها ناشی از ارسال وظیفه به صورت بی‌سیم است. اگر نرخ ارسال داده از لایه مه به ابر را با r_{fc} نشان دهیم، در آن صورت برای تأخیر این حالت خواهیم داشت:

$$T_{ic}^{trans} = T_{ij}^{trans} + \frac{D_i}{r_{fc}}, \forall N_i^u \in \mathcal{N}^u, \forall N_j^f \in \mathcal{N}^f \quad (۱۵)$$

پس در آخر هزینه کل بارسپاری و اجرای وظیفه روی لایه ابر به صورت زیر قابل محاسبه است:

$$\phi_i^{cloud} = \phi_i^c = \lambda^T T_{ic}^{trans} + (1 - \lambda^T) E_{ij}^{trans}, \forall N_i^u \in \mathcal{N}^u \quad (۱۶)$$

۵-۳- مدل تحرک

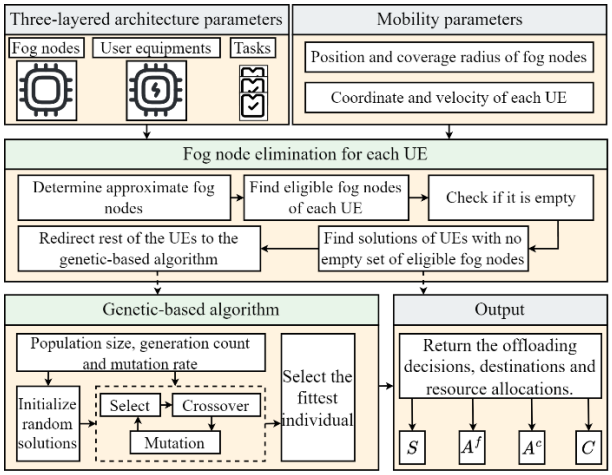
محیط تحرک دستگاه‌های کاربری یک منطقه مربعی‌ست که مسیرها و خیابان‌های مشخص شده واقعی در آن قرار دارد. به همین دلیل در لحظه t مدل تحرک دستگاه کاربری N_i^u با چهارتایی مرتب $(x_i^u(t), y_i^u(t), v_i^u(t), \alpha_i^u(t))$ قابل نمایش است. که در آن x_i^u و y_i^u مختصات دستگاه کاربری، v_i^u سرعت و α_i^u زاویه حرکت است. واحدهای مه با سه تایی (x_i^f, y_i^f, ρ_i^f) نشان داده می‌شود که x_i^f و y_i^f مختصات واحد مه و ρ_i^f شعاع پوششی آن است.

در این پژوهش از مدل تحرک خطی با سرعت ثابت استفاده شده است، زیرا هدف اصلی ارزیابی کارایی الگوریتم تصمیم‌گیری بارسپاری در چارچوب سه لایه بوده و نه مدل‌سازی دقیق تحرک که خود حوزه‌ای مجزا محسوب می‌شود. این مدل ساده برای پیش‌بینی موقعیت در بازه‌های کوتاه کافی است و همان‌طور که نتایج نشان می‌دهد، تعداد مهاجرت با همین مدل‌سازی، مقدار قابل قبولی کاهش یافته است. بنابراین، دقت مورد نیاز الگوریتم تأمین شده و انتخاب مدل حرکتی ساده قابل توجیه است. در آینده می‌توان از روش‌های پیشرفته‌تر مانند Markov Models یا LSTM برای افزایش دقت پیش‌بینی بهره گرفت.

۶-۳- مدل مهاجرت

با داشتن مکان دقیق هر دستگاه در هر لحظه مدل کردن مهاجرت شامل دو حالت است. باید توجه داشته باشیم که برای دستگاه کاربری i اگر $s_i = 1$ و $a_{ij} = 1, \exists N_j^f \in \mathcal{N}^f$ باشد احتمال مهاجرت مطرح می‌شود که یعنی وظیفه به لایه مه واگذار شده است.

- **حالت ۱:** اگر دستگاه کاربری در محدوده پوششی واحد مهی که وظیفه به آن بارسپاری شده است باشد و اجرا تمام شود، نتیجه می‌تواند بدون هزینه اضافی به دستگاه کاربری برگردد. در این سناریو مهاجرت رخ



شکل ۴- ساختار کلی روش پیشنهادی

این هزینه قابل محاسبه است که δ هزینه مهاجرت به ازای هر بیت است [۲۸]، [۳۵]. حالت ۲ به صورت زیر قابل تشخصی است: δD_i

$$\sqrt{\left(x_i^u(t_2)-x_j^f\right)^2+\left(y_i^u(t_2)-y_j^f\right)^2}>\rho_j^f \quad (۱۹)$$

که در این حالت هزینه کل مطابق زیر قابل محاسبه است:

$$\phi_i^{j,2}=\phi_i^j+\phi_i^{mig} \quad (۲۰)$$

برای تعیین متوسط هزینه نهایی در حالت بارسپاری به لایه مه، نیاز داریم که متغیر تصادفی $M_{ij}(t_1,t_2)$ را تعریف کنیم. به همین دلیل، پیشبینی از مکان بعدی دستگاه کاربری به صورت زیر خواهیم داشت. فرض کنید وقتی حالت ۱ رخ می‌دهد این متغیر تصادفی صفر و در حالت ۲ مقدار ۱ دارد. در این صورت هزینه بارسپاری به مه به صورت زیر است:

$$\phi_i^j=\begin{cases}\phi_i^{j,1}, & if M_{ij}(t_1,t_2)=0,\\ \phi_i^{j,2}, & if M_{ij}(t_1,t_2)=1.\end{cases} \quad (۲۱)$$

برای اینکه احتمال این متغیر تصادفی را محاسبه کنیم، باید پیشبینی بر اساس مدل حرکتی تعیین کنیم. به این صورت اگر لحظه t_2 اتمام اجرای وظیفه و لحظه t_1 لحظه ایجاد وظیفه باشد، خواهیم داشت:

$$x_i'^u(t_1,t_2)=x_i^u(t_1)+\cos\left(\alpha_i^u(t_1)\right)v_i^u(t_1)(t_2-t_1) \quad (۲۲)$$

$$y_i'^u(t_1,t_2)=y_i^u(t_1)+\sin\left(\alpha_i^u(t_1)\right)v_i^u(t_1)(t_2-t_1) \quad (۲۳)$$

حال می‌توانیم فاصله تخمینی از واحد مه که اجرای وظیفه بر روی آن صورت گرفته است را به شکل زیر تعریف کنیم:

$$d_{ij}(t_1,t_2)=\sqrt{\left(x_i'^u(t_1,t_2)-x_j^f\right)^2+\left(y_i'^u(t_1,t_2)-y_j^f\right)^2} \quad (۲۴)$$

که یعنی احتمال مهاجرت به صورت زیر قابل تعریف است:

$$P\left(M_{ij}(t_1,t_2)=1\right)=min\left(1,\frac{d_{ij}(t_1,t_2)}{\rho_j^f}\right) \quad (۲۵)$$

در نهایت با توجه به معادلات (۲۱) و (۲۵)، امید ریاضی هزینه Φ_i^j وقتی بارسپاری به لایه مه داریم به صورت زیر است:

$$\overline{\phi_i^j}=\phi_i^{j,1}\cdot P\left(M_{ij}(t_1,t_2)=0\right)+\phi_i^{j,2}\cdot P\left(M_{ij}(t_1,t_2)=1\right) \quad (۲۶)$$

که بعد از ساده‌سازی به شکل زیر خواهد شد:

$$\overline{\phi_i^j}=\phi_i^j+\phi_i^{mig}\cdot P\left(M_{ij}(t_1,t_2)=1\right) \quad (۲۷)$$

۴- روش پیشنهادی

در این بخش، مسئله به زبان ریاضی بیان می‌شود و سپس با الگوریتم پیشنهادی حل می‌شود. این موارد در زیربخش‌های آتی توضیح داده می‌شوند.

Algorithm 1 The MOFCO algorithm

Input: $\mathcal{N}^u, \mathcal{N}^f, p, g, P_m$
Output: S, A^f, A^c, C

Step 1 Fog node elimination:
Initialize: $F=\emptyset$; // Set of unassigned UEs
for $N_i^u \in \mathcal{N}^u$ **do**
  Initialize: $D_i = FindProximateFogNodes(N_i^u)$;
  Initialize: $E_i=\emptyset$; // Set of eligible fog nodes
  for $N_j^f \in D_i$ **do**
    if $\phi_i^{best,j} \le \min(\phi_i^c, \phi_i^{local})$ **then**
      $E_i.add(N_j^f)$;
    end if
  end for
  if $E_i == \emptyset$ **then**
    if $\phi_i^{local} > \phi_i^{cloud}$ **then**
      $s_i = 1$;
      $a_i^c = 1$;
    else
      $s_i = 0$;
    end if
  else
    $F.add(N_i^u)$;
  end if
end for
Step 2 Using genetic to find solution for remaining UEs:
Initialize: $P = InitializePopulation(F, p)$;
for $c = 1 : g$ **do**
  $P = Select(P)$; // Selecting fittest individuals based on eq. (28)
  $P = Crossover(P)$; // Two-point crossover
  $P = Mutate(P, P_m)$; // Mutation based on algorithm (2)
end for
$FindFittest(P)$;

الگوریتم ۱- روش پیشنهادی MOFCO

Algorithm 2 Mutation Operation

Input: P, P_m
Output: P^{new}

Step 1: Initialize Mutation Parameters
for each $person$ in P **do**
  if $PerformMutation(P_m)$ **then**: // Randomly decide whether to mutate
    $mutationType = SelectMutationType()$; // Select mutation type (decision, destination, resource)
    if $mutationType == decision$ **then**:
      $FlipOffloadingDecision(person)$; // Flip offloading decision s_i
    else if $mutationType == destination$ **then**:
      $ChangeDestination(person)$; // Adjust destination a_i^j or set $a_i^c = 1$
    else if $mutationType == resource$ **then**:
      $AdjustResourceAllocation(person)$; // Adjust resource allocation c_{ij}^f
    end if
  end if
end for

الگوریتم ۲- فرایند جهش الگوریتم

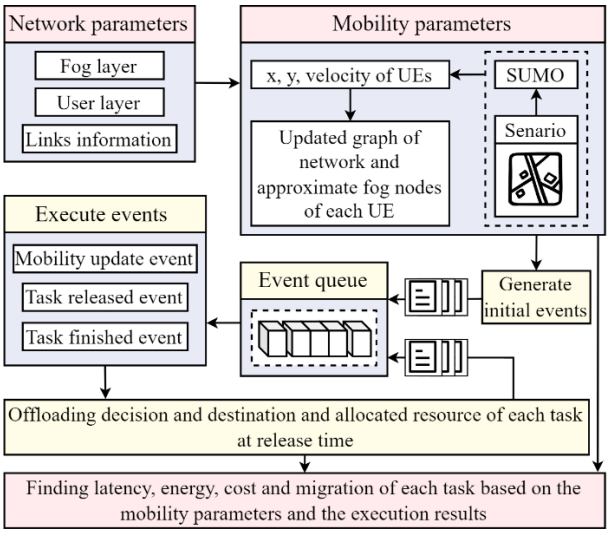
- نداده است. اگر t_1 زمانی باشد که وظیفه برای اجرا وارد سیستم می‌شود و t_2 لحظه‌ای باشد که اجرا تمام می‌شود و N_j^f واحد مه مقصد باشد، حالت ۱ برای دستگاه کاربری i مطابق زیر خواهد بود:

$$\sqrt{\left(x_i^u(t_2)-x_j^f\right)^2+\left(y_i^u(t_2)-y_j^f\right)^2}\le\rho_j^f \quad (۱۷)$$

و در نتیجه برای هزینه خواهیم داشت:

$$\phi_i^{j,1}=\phi_i^j \quad (۱۸)$$

- **حالت ۲:** اگر دستگاه کاربری از محدوده پوشش واحد مه مقصد پس از انجام وظیفه خارج شده باشد، نتیجه نمی‌تواند به صورت مستقیم به دستگاه کاربری مبدأ برگردد. در این شرایط نتیجه باید از طریق لایه ابر مسیریابی شود و به مبدأ باز گردد. در این سناریو مهاجرت رخ داده است و باعث افزایش تأخیر و مصرف انرژی کل می‌شود. تأخیر و انرژی اضافه شده به صورت هزینه حساب می‌شود و به صورت $\phi_i^{mig}=$



شکل ۵- ساختار کلی شبیه‌سازی مبتنی بر واقعه

۴-۱- مسئله

قرار است مسئله ریاضی با هدف بهینه کردن متغیر هزینه حل شود که خود شامل دو پارامتر تأخیر و انرژی است. هزینه، طبق مدل سیستم در نظر گرفته شده، در سناریوهای متفاوت، مقدار مختلفی دارد و طبق معادلات (۳)، (۱۶) و (۲۷) می‌توانیم تابع هزینه را به صورت چندضابطه‌ای تعریف کنیم:

$$\Phi_i\left(s_i, a_{ij}, a_{ic}, c_{ij}^f\right) = \begin{cases} \phi_i^{local} & , s_i = 0 \\ \phi_i^j & , s_i = 1 \ , a_{ij} = 1 \\ \phi_i^{cloud} & , s_i = 1, a_{ic} = 1 \end{cases} \quad (۲۸)$$

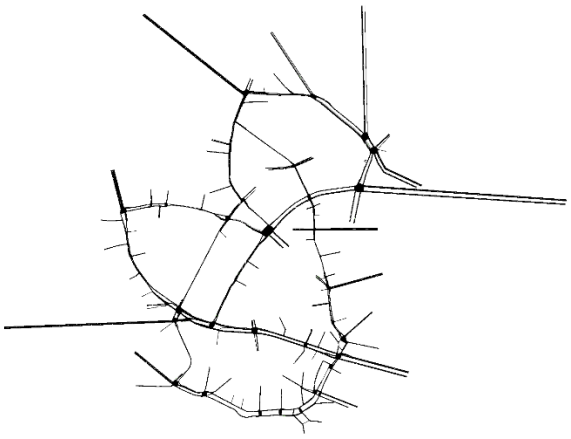
با توجه به تعریف تابع هزینه، مسئله به صورت زیر بیان خواهد شد:

$$P1: \quad \min_{S,A^f,A^c,C} \sum_{N_i^u \in \mathcal{N}^u} \Phi\left(s_i, a_{ij}, a_{ic}, c_{ij}^f\right)$$

$$s.t. \quad C1: \ s_i = \{0,1\}, \quad \forall N_i^u \in \mathcal{N}^u$$
$$C2: \ a_{ij} = \{0,1\}, \quad \forall N_i^u \in \mathcal{N}^u$$
$$C3: \ a_{ic} = \{0,1\}, \quad \forall N_i^u \in \mathcal{N}^u \quad (۲۹)$$
$$C4: \ \sum_{j=1}^{F} a_{ij} + a_{ic} \le 1, \quad \forall N_i^u \in \mathcal{N}^u$$
$$C5: \ \sum_{i=1}^{U} c_{ij}^f \le c_j^f, \quad \forall N_j^f \in \mathcal{N}^f$$

سه محدودیت اول که به ترتیب C1، C2 و C3 هستند، مقادیر صفر یا یکی تصمیمات بارسپاری، انتخاب لایه مه و انتخاب لایه ابری را نشان می‌دهند. محدودیت C4 نشان می‌دهد که هر کاربر فقط می‌تواند به یک مقصد خاص بارسپاری کند یا وظیفه خود را به صورت محلی اجرا کند. محدودیت C5 نشان می‌دهد که مجموع منابع تخصیص‌یافته به وظایف بارسپاری‌شده به یک واحد مه خاص نباید از ظرفیت حداکثری آن واحد مه تجاوز کند.

این مسئله یک مسئله برنامه‌ریزی غیرخطی مختلط (MINLP) است که NP-سخت می‌باشد. برای حل این مسئله، ما یک رویکرد ژنتیک همراه با یک الگوریتم ابتکاری به نام MOFCO طراحی کرده‌ایم که سعی می‌کند تا حد امکان گره‌های مه را با توجه به بهترین پاداشی که هر گره مه ارائه می‌دهد حذف کند و سپس با استفاده از الگوریتم ژنتیک فضای باقی‌مانده را بررسی کرده و راه‌حلی با کمترین مهاجرت‌ها پیدا کرده و هزینه نهایی هر وظیفه را کاهش می‌دهد.



شکل ۶- نقشه سناریو واقعی شبیه‌ساز

۴-۲- روش پیشنهادی

در این بخش، روش پیشنهادی برای حل این مسئله MINLP چندهدفه توصیف شده است. رویکرد اصلی یک روش دو مرحله‌ای مبتنی بر الگوریتم ژنتیک و ابتکاری است که در شکل ۴ نشان داده شده است. این روش با محاسبه بهترین حالت برای هر واحد مه نزدیک و حذف آن‌ها از گزینه‌های بارسپاری، فضای جستجوی مسئله را محدود می‌کند و سپس فضای جستجوی باقی‌مانده با استفاده از الگوریتم مبتنی بر ژنتیک و الگوهای جستجوی تصادفی بررسی می‌شود. بهترین وضعیت بارسپاری به یک گره مه زمانی است که تمام ظرفیت پردازشی آزاد را در اختیار وظیفه قرار دهد و هم‌چنین مهاجرت رخ ندهد. اگر هزینه حاصل از این حالت از تصمیم‌گیری بارسپاری به ابر و اجرای محلی بهتر شود، گره مه به عنوان گزینه تصمیم‌گیری به مجموعه کاندیدهای بارسپاری اضافه می‌شود. این روش می‌تواند به طور کارآمد به راه‌حل‌های نزدیک به بهینه دست یابد. لازم به ذکر است که در این مقاله فرض بر وجود یک کنترل‌کننده مرکزی مبتنی بر مدل‌های شبکه نرم‌افزارمحور است که در آن محاسبات تصمیم‌گیری به صورت متمرکز و خارج از مسیر داده انجام می‌شوند. در چنین معماری‌هایی، سربار محاسباتی الگوریتم در مسیر مستقیم تبادل داده‌ها قرار ندارد و تأخیر تصمیم‌گیری بر تأخیر اجرای وظایف تأثیر مستقیمی نمی‌گذارد.

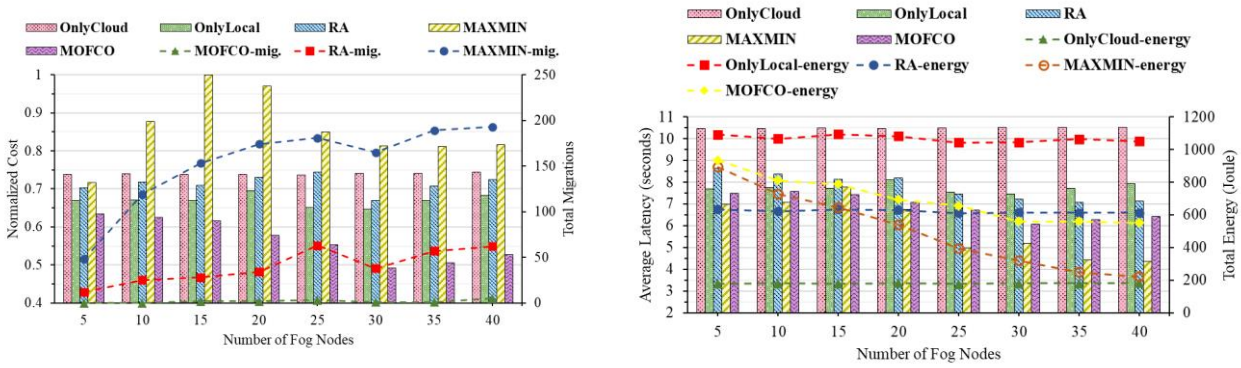
اجزای اصلی یک الگوریتم ژنتیک شامل تولید جمعیت، انتخاب والدین، ترکیب (crossover) و جهش است. هر کروموزوم در جمعیت ارزیابی شده و یک مقدار شایستگی به آن اختصاص داده می‌شود. کروموزوم‌های دارای بیشترین شایستگی به تکرار بعدی منتقل می‌شوند و این فرآیند به الگوریتم اجازه می‌دهد به تدریج به سمت راه‌حل‌های بهینه‌تر همگرا شود.

برای حذف گره‌های مه غیرقابل استفاده، همان‌طور که در بخش مثال انگیزشی مشاهده شد، باید حداکثر ظرفیت محاسباتی آزاد آن‌ها در نظر گرفته شود. علاوه بر این، بهترین حالت زمانی اتفاق می‌افتد که هیچ مهاجرتی وجود نداشته باشد، بنابراین معادله زیر برای محاسبه بهترین حالت بارسپاری به هر گره مه استفاده می‌شود:

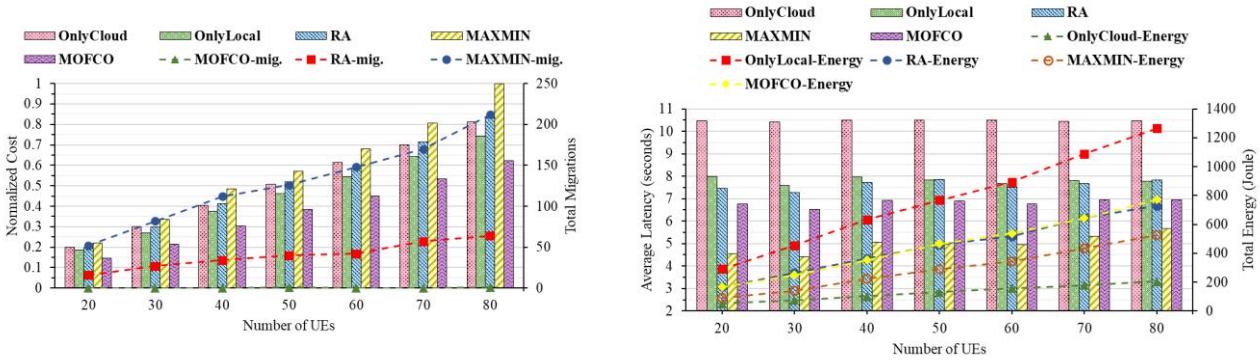
$$\phi_i^{j,best} = \phi_i^j, \quad c_{ij}^f = c_j^f - \sum_{k=1,k\neq i}^{U} c_{kj}^f \quad (۳۰)$$

که به طور کلی به این معناست که بهترین هزینه هنگام بارسپاری به یک گره مه تخصیص حداکثر منابع موجود است و فرض می‌شود که هیچ مهاجرتی رخ نمی‌دهد. الگوریتم (۱) روش پیشنهادی ما را نشان می‌دهد.

ابتدا، در خط ۴، فرآیند حذف گره‌های مه، گره‌های مه نزدیک هر دستگاه کاربری را از طریق تابع $FindProximateFogNodes(N_i^u)$ پیدا کرده و در صورتی که هزینه بهترین حالت آن‌ها از هزینه مربوط به اجرای محلی یا بارسپاری



شکل ۷- نتیجه شبیه‌سازی ۷۰ دستگاه کاربری در مدت ۳۰ دقیقه و بررسی تأثیر تعداد واحدهای مه روی هزینه کل، تعداد کل مهاجرت، میانگین تأخیر نهایی و انرژی مصرفی کل



شکل ۸- نتیجه شبیه‌سازی دستگاه‌های کاربری متغیر در مدت ۳۰ دقیقه و بررسی تأثیر تعداد دستگاه‌های کاربری روی هزینه کل، تعداد کل مهاجرت، میانگین تأخیر نهایی و انرژی مصرفی کل

به ابر بیشتر باشد، آن‌ها را حذف می‌کند. این مقایسه در خط ۷ الگوریتم (۱) قابل مشاهده است. سپس بررسی می‌کند که آیا پس از فرآیند حذف، هیچ واحد مه واجدی باقی مانده است یا خیر. در صورتی که باقی نمانده باشد، مقصد و تصمیم مشخص می‌شود.

در مرحله دوم، فضای جستجوی باقی‌مانده با استفاده از الگوریتم ژنتیک بررسی می‌شود. تابع $InitializePopulation(F,p)$ راه‌حل‌های تصادفی را با توجه به محدودیت‌های توضیح داده‌شده در معادله (۲۹) ایجاد می‌کند. سپس، برای تعداد نسل‌ها، الگوریتم بهترین نمونه‌های جمعیت را با استفاده از روش انتخاب تورنمنتی بر اساس معادله (۲۸) انتخاب می‌کند. با این حال، مقدار شایستگی به‌صورت $-\Phi_i$ تعریف شده است و نمونه‌هایی که هزینه کمتری دارند مطلوب‌تر هستند.

تابع ترکیب (crossover) دو والد را به صورت تصادفی از جمعیت انتخاب‌شده انتخاب کرده و یک ترکیب دو نقطه‌ای انجام می‌دهد، به‌طوری که فرزندان حاصل، از والدین تصادفی انتخاب‌شده در مرحله انتخاب ایجاد می‌شوند. در مرحله بعد، جمعیت مطابق الگوریتم (۲) جهش داده می‌شود.

همان‌طور که در الگوریتم (۲) نشان داده شده است، فرآیند جهش ابتدا باید از یک فیلتر متغیر تصادفی که به نرخ جهش P_m بستگی دارد عبور کند. اگر فردی از جمعیت برای جهش انتخاب شود، سپس تصمیم می‌گیرد که کدام پارامتر باید جهش یابد. این پارامتر می‌تواند تصمیم بارسپاری s_i، مقصد بارسپاری که به a_i^j و a_i^cمرتبط است، یا منبع تخصیص‌یافته به واحد مه که c_{ij}^f است باشد. لازم به ذکر است که در طول اجرای ترکیب (crossover) و جهش، محدودیت‌های مسئله بررسی شده و اطمینان حاصل می‌شود که نقض نمی‌شوند.

پس از تکرار به تعداد نسل‌ها در الگوریتم، در تابع نهایی SetRemaining(P)، این تابع مقادیر s_i ، a_i^j، a_i^c و c_{ij}^f را برای دستگاه‌های کاربری باقی‌مانده بر اساس بهترین شایستگی که به صورت $-\Phi_i$ محاسبه می‌شود، تنظیم می‌کند.

با توجه به حلقه‌های مشاهده‌شده در مرحله اول الگوریتم اصلی، ترتیب اجرای این مرحله $O(UF)$ است، زیرا در بدترین حالت الگوریتم باید برای تمام دستگاه‌هایی که در پوشش تمامی واحدهای مه قرار دارند تصمیم‌گیری کند. اگر پارامترهای الگوریتم ژنتیک p و g به‌عنوان مقادیر ثابت در نظر گرفته شوند، ترتیب اجرای مرحله دوم $O(UF)$ نیز خواهد بود، زیرا پس از هر ترکیب یا جهش، الگوریتم باید منابع موجود گره‌های مه نزدیک را بررسی کرده و مطمئن شود که محدودیت‌ها رعایت شده‌اند.

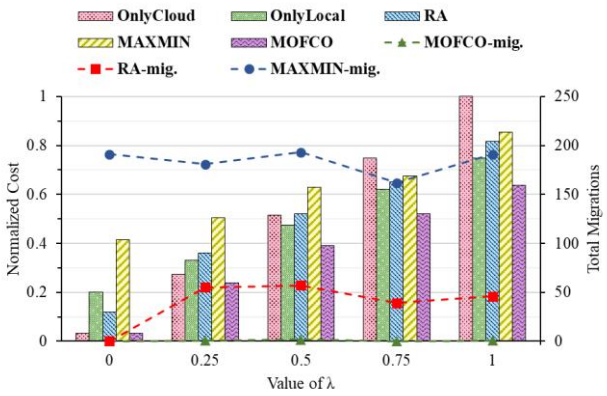
نکته‌ای که باید در نظر داشت این است که تعداد گره‌های مه نزدیک هر کاربر معمولاً به‌عنوان یک مقدار ثابت در نظر گرفته می‌شود، زیرا در لحظات معین، هر دستگاه کاربری تنها تحت پوشش تعداد محدودی از گره‌های مه قرار می‌گیرد.

۵- نتایج شبیه‌سازی

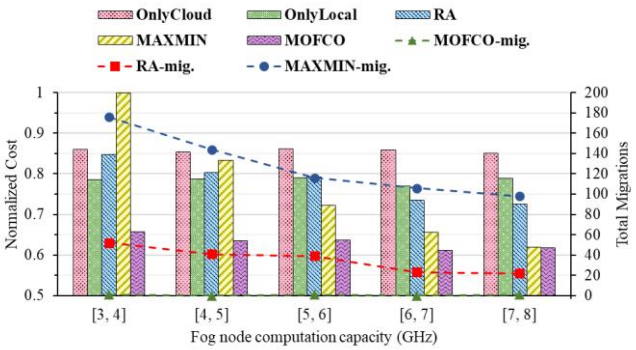
در این بخش، نتایج شبیه‌سازی طی یک پیاده‌سازی مبتنی بر واقعه که در شکل ۵ توضیح داده شده است، با روش‌های قبلی مقایسه شده‌اند.

۵-۱- شرایط سنجش

شبیه‌سازی ما با استفاده از Python 3.11.0 پیاده‌سازی شده است و مدل حرکتی و محیط‌های فیزیکی از نتایج سناریوهای پیش‌ساخته در SUMO گرفته شده‌اند. محیط فیزیکی شبیه‌سازی بر اساس سناریوی متن‌باز [۳۶] که طبق



شکل ۹- بررسی تأثیر پارامتر λ در شبیه‌سازی با ۲۵ واحد مه و ۷۰ دستگاه کاربری در مدت ۳۰ دقیقه



شکل ۱۰- بررسی تأثیر ظرفیت پردازشی گره‌های مه در شبیه‌سازی با ۲۵ واحد مه و ۷۰ دستگاه کاربری در مدت ۳۰ دقیقه

بخشی از شهر هامبورگ است، طراحی شده است، همان‌طور که در شکل ۶ نشان داده شده است. خروجی شبیه‌سازی SUMO شامل موقعیت دقیق و سرعت هر وسیله نقلیه در هر گام زمانی (بر حسب ثانیه) است. وسایل نقلیه در یک بازه زمانی ۳۰ دقیقه‌ای معادل ۱۸۰۰ ثانیه پایش شدند و الگوریتم‌های مقایسه‌شده از این وسایل نقلیه به‌عنوان دستگاه‌های کاربری استفاده کردند.

الگوریتم پیشنهادی با چهار الگوریتم پایه مقایسه شده است. اولین الگوریتم یک رویکرد حریصانه است که در آن دستگاه‌ها تنها وظایف خود را به لایه ابر بارسپاری می‌کنند (OnlyCloud). رویکرد دوم (OnlyLocal) از بارسپاری اجتناب کرده و وظایف را در صورت امکان به‌صورت محلی اجرا می‌کند (اگر دستگاه کاربری در حال اجرای نمونه قبلی وظیفه خود باشد، وظیفه را به لایه‌های بالاتر بارسپاری می‌کند و لایه مه اولویت دارد). الگوریتم سوم به‌صورت تصادفی بین بارسپاری یا اجرای محلی انتخاب کرده و در صورت نیاز مقصد بارسپاری را نیز به‌صورت تصادفی انتخاب می‌کند (RA).

برای نشان دادن تأثیر حرکت و مهاجرت، رویکرد چهارم (MAXMIN) تمایل دارد وظایف را به لایه مه بارسپاری کند و در صورتی که مقاصد مه پر باشند، وظایف را به‌صورت محلی اجرا می‌کند. مقاصد گره مه بر اساس یک اکتشافی max-min انتخاب می‌شوند، جایی که دستگاه‌های کاربری مقصدی را انتخاب می‌کند که حداقل پاداشی را که انتظار دارد دریافت کند، به حداکثر برساند.

۵-۲- تأثیر تعداد واحدهای مه

در شکل ۷، هزینه کل نرمال‌شده اجرای الگوریتم‌های مختلف با توجه به تعداد گره‌های مه نشان داده شده است. هزینه الگوریتم‌هایی مانند OnlyCloud و OnlyLocal با افزایش تعداد گره‌های مه تغییر نمی‌کند، زیرا این الگوریتم‌ها لایه مه را نادیده می‌گیرند. با این حال، الگوریتم MAXMIN بدلیل مهاجرت‌های زیاد عملکرد ضعیفی دارد و در تقریباً تمام موارد بدترین روش بوده است. تعداد مهاجرت‌ها در الگوریتم MAXMIN با افزایش تعداد گره‌های مه افزایش می‌یابد، زیرا با قرار گرفتن تعداد بیشتری گره مه در سناریو، دستگاه‌های کاربری متحرک بیشتری می‌توانند به گره‌های مه دسترسی پیدا کنند که منجر به بارسپاری بیشتر می‌شود. الگوریتم تصادفی نیز افزایش تعداد مهاجرت‌ها را نشان می‌دهد، اما الگوریتم پیشنهادی (MOFCO) به طور مداوم در هر سناریو کمتر از ۱۰ مهاجرت دارد.

بدلیل تعداد کمتر مهاجرت‌ها و انتخاب پویا بین سه لایه، روش پیشنهادی هزینه کلی کمتری را به دست می‌آورد که به طور متوسط ۱۸٪ بهتر است. در شکل ۷ نمودار دوم، تأخیر متوسط و مصرف انرژی کل نشان داده شده‌اند. مشابه هزینه، تأخیر متوسط و مصرف انرژی در روش‌های OnlyCloud و OnlyLocal در سناریوهای مختلف نسبتاً ثابت باقی می‌مانند. همان‌طور که انتظار می‌رود، مصرف انرژی در روش OnlyLocal بیشترین مقدار را دارد، در حالی که تأخیر متوسط بارسپاری هر وظیفه به لایه ابر نیز بیشترین مقدار است. بدلیل انتخاب حریصانه گره‌های مه در روش MAXMIN، تأخیر متوسط و مصرف انرژی این روش با افزایش تعداد واحدهای مه کاهش می‌یابد. به طور مشابه، تأخیر متوسط و مصرف انرژی MOFCO نیز کاهش می‌یابد، زیرا توان محاسباتی موجود در لایه مه افزایش می‌یابد.

۵-۳- تأثیر تعداد دستگاه‌های کاربری

در شکل ۸ تأثیر تعداد دستنگاه‌های کاربری نشان داده شده است. افزایش تعداد دستگاه‌ها به معنای پایش تعداد بیشتری وسیله نقلیه در بازه زمانی ۳۰ دقیقه‌ای شبیه‌سازی SUMO است که منجر به تولید وظایف بیشتری برای اجرا شده و به‌طور طبیعی هزینه کلی سیستم را افزایش می‌دهد. هر دو روش OnlyCloud و OnlyLocal افزایش هزینه را نشان می‌دهند، چون تعداد بیشتری وظیفه اجرا شده است و انرژی اضافی (برای OnlyLocal) و تأخیر (برای OnlyCloud) مورد نیاز برای اجرای آن‌ها باعث آن است.

بدلیل تعداد ثابت واحدهای مه، افزایش بار وظایف باعث افزایش مهاجرت‌ها در روش MAXMIN می‌شود، زیرا این روش در انتخاب مقصد ضعیف عمل می‌کند. با شلوغ‌تر شدن واحدهای مه، زمان‌های اجرا کندتر شده و منجر به مهاجرت‌های بیشتر می‌شود. در مقابل، MOFCO تعداد کمی از مهاجرت‌ها را حفظ کرده و در هر سناریو کمترین هزینه را به دست می‌آورد و به‌طور متوسط ۱۳٪ کارآمدتر است.

مصرف انرژی و تأخیر متوسط برای تعداد مختلف دستگاه‌ها در شکل ۸ نمودار دوم نشان داده شده‌اند. تأخیر متوسط برای هر دو روش OnlyLocal و OnlyCloud ثابت باقی می‌ماند، بدلیل منابع و تخصیص قدرت انتقال استاتیکی که در این مطالعه مدل‌سازی شده است. روش MAXMIN در همه سناریوها کمترین تأخیر را دارد، با افزایش کمی در پیشرفت سناریوها، زیرا واحدهای مه شلوغ‌تر می‌شوند.

با این حال، مصرف انرژی کل برای همه روش‌ها افزایش می‌یابد، و OnlyCloud بدلیل اتکای آن به انتقال، کمترین مصرف انرژی را دارد. بالاترین مصرف انرژی در روش OnlyLocal رخ می‌دهد، در حالی که MAXMIN پس از OnlyCloud دومین کمترین مصرف انرژی را دارد. بیشتر مصرف انرژی در MAXMIN ناشی از انتقال به علاوه اجرای محلی در شرایطی است که گره‌های مه منابع کافی ندارند.

۵-۴- تأثیر پارامترهای دیگر

مقایسه هزینه تحت مقادیر مختلف ضرایب λ^T در شکل ۹ نشان داده شده است. زمانی که $\lambda^T = 0$ باشد، هزینه فقط بدلیل مصرف انرژی و مهاجرت افزایش می‌یابد. این موضوع توضیح می‌دهد که چرا در سناریویی که $\lambda^T = 0$ است، هر دو روش MAXMIN و OnlyLocal بالاترین هزینه را دارند. باید توجه داشت که در این حالت، هزینه کل MOFCO برابر با OnlyCloud است، زیرا در مرحله حذف واحدهای مه، الگوریتم هیچ کاندید مناسبی در لایه مه پیدا نمی‌کند.

افزایش مقدار λ^T منجر به افزایش هزینه OnlyCloud می‌شود زیرا وظایف حساس‌تر به تأخیر می‌شوند. در حالتی که $\lambda^T = 1$، OnlyCloud بالاترین هزینه را در میان همه سناریوها نشان می‌دهد، در حالی که روش پیشنهادی در هر سناریو به راه‌حل‌های بهتر دست می‌یابد و به طور متوسط ۱۶٪ کارآمدتر است.

در شکل ۱۰، تأثیر ظرفیت محاسباتی مختلف گره‌های مه نشان داده شده است. بی‌تفاوتی روش‌های OnlyCloud و OnlyLocal در این شبیه‌سازی مشهود است. با این حال، هزینه‌های MOFCO، RA و MAXMIN با افزایش ظرفیت محاسباتی کاهش می‌یابد. همچنین مشاهده می‌شود که در بالاترین ظرفیت واحدهای مه، یعنی بین ۷ تا ۸ گیگاهرتز، MAXMIN تقریباً عملکردی مشابه MOFCO دارد. دلیل این موضوع تخصیص منابع کافی به وظایف در این روش است که منجر به کاهش تأثیر مهاجرت بدلیل کاهش تأخیر و انرژی کلی می‌شود. MOFCO در این سناریوها به طور متوسط ۱۶٪ بهتر است.

تأثیر هزینه مهاجرت نیز مقایسه شده و در شکل ۱۱ نشان داده شده است. ضریب پایین هزینه مهاجرت منجر به افزایش مهاجرت‌ها در روش پیشنهادی می‌شود. همان‌طور که در شکل نشان داده شده است، زمانی که $\delta = 10^{-7}$ باشد، بالاترین تعداد مهاجرت‌ها برای MOFCO رخ می‌دهد. دلیل این امر این است که هزینه مهاجرت پایین‌تر، پاداش بیشتری برای هر راه‌حل که به مه منتقل می‌شود ایجاد می‌کند و منجر به افزایش مهاجرت‌ها می‌شود. با این حال، مقدار کلی برای MOFCO ثابت باقی می‌ماند، زیرا این روش تعداد مهاجرت‌ها را به حداقل می‌رساند. در مقابل، روش‌های MAXMIN و RA به تحرک توجهی نمی‌کنند و در نتیجه، با افزایش هزینه، از مهاجرت‌ها بیشتر آسیب می‌بینند و MOFCO به طور متوسط ۲۰٪ کارآمدتر از MAXMIN و RA است.

۶- نتیجه‌گیری

در این مقاله، ما تحرک تعدادی مشخص از دستگاه کاربری را با استفاده از یک شبیه‌سازی واقعی SUMO در یک بازه زمانی معین پایش کردیم و بارسپاری وظایف برای هر دستگاه را به‌منظور کاهش تعداد کل مهاجرت‌ها تعیین کردیم. روش MOFCO برای حل مسئله بارسپاری وظایف و تخصیص منابع پیشنهاد شد. نتایج نشان می‌دهد که روش پیشنهادی ما به طور متوسط ۱۶٪ کارآمدتر است و می‌تواند هزینه کل را در برخی سناریوها تا ۵۸٪ کاهش دهد.

در کارهای آینده، قصد داریم فرآیند مهاجرت را با دقت بیشتری مدل‌سازی کنیم، هزینه آن را کاهش دهیم و از آن برای به حداقل رساندن هزینه کلی اجرا استفاده کنیم؛ به این صورت که با پیش‌بینی موقعیت‌های مهاجرت اجتناب‌ناپذیر، وظایف را به واحد مه مناسب انتقال دهیم که به هنگام خاتمه اجرای آن، دستگاه کاربری متحرک در محدوده دریافت نتیجه قرار داشته باشد.

۷- مراجع

[1] S. D. Okegbile, B. T. Maharaj and A. S. Alfa, "A Multi-User Tasks Offloading Scheme for Integrated Edge-Fog-Cloud Computing Environments," in *IEEE Transactions on Vehicular Technology*, vol. 71, no. 7, pp. 7487-7502, July 2022, doi: 10.1109/TVT.2022.3167892.

[2] P. Habibi, M. Farhoudi, S. Kazemian, S. Khorsandi and A. Leon-Garcia, "Fog Computing: A Comprehensive Architectural Survey," in *IEEE Access*, vol. 8, pp. 69105-69133, 2020, doi: 10.1109/ACCESS.2020.2983253.

[3] H. Tran-Dang and D. -S. Kim, "FRATO: Fog Resource Based Adaptive Task Offloading for Delay-Minimizing IoT Service Provisioning," in *IEEE Transactions on Parallel and Distributed Systems*, vol. 32, no. 10, pp. 2491-2508, 1 Oct. 2021, doi: 10.1109/TPDS.2021.3067654.

[4] H. Tran-Dang and D. -S. Kim, "Dynamic collaborative task offloading for delay minimization in the heterogeneous fog computing systems," in *Journal of Communications and Networks*, vol. 25, no. 2, pp. 244-252, April 2023, doi: 10.23919/JCN.2023.000008.

[5] K. Wang, Y. Zhou, J. Li, L. Shi, W. Chen and L. Hanzo, "Energy-Efficient Task Offloading in Massive MIMO-Aided Multi-Pair Fog-Computing Networks," in *IEEE Transactions on Communications*, vol. 69, no. 4, pp. 2123-2137, April 2021, doi: 10.1109/TCOMM.2020.3046265.

[6] X. Dai et al., "Task Offloading for Cloud-Assisted Fog Computing With Dynamic Service Caching in Enterprise Management Systems," in *IEEE Transactions on Industrial Informatics*, vol. 19, no. 1, pp. 662-672, Jan. 2023, doi: 10.1109/TII.2022.3186641.

[7] Q. Wu, S. Wang, H. Ge, P. Fan, Q. Fan and K. B. Letaief, "Delay-Sensitive Task Offloading in Vehicular Fog Computing-Assisted Platoons," in *IEEE Transactions on Network and Service Management*, vol. 21, no. 2, pp. 2012-2026, April 2024, doi: 10.1109/TNSM.2023.3322881.

[8] R. A. Addad, D. L. Cadette Dutra, M. Bagaa, T. Taleb and H. Flinck, "Towards a Fast Service Migration in 5G," 2018 *IEEE Conference on Standards for Communications and Networking (CSCN),* Paris, France, 2018, pp. 1-6, doi: 10.1109/CSCN.2018.8581836.

[9] A. Machen, S. Wang, K. K. Leung, B. J. Ko and T. Salonidis, "Live Service Migration in Mobile Edge Clouds," in *IEEE Wireless Communications*, vol. 25, no. 1, pp. 140-147, February 2018, doi: 10.1109/MWC.2017.1700011.

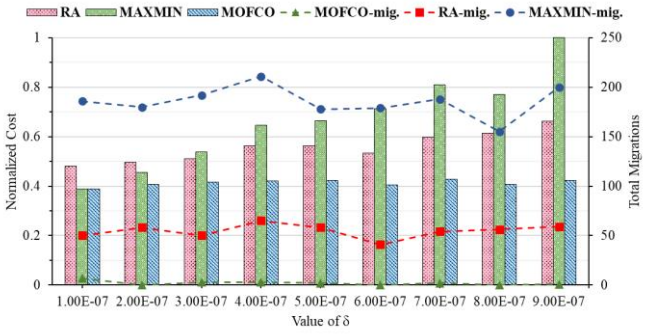
[10] P. A. Lopez et al., "Microscopic Traffic Simulation using SUMO," 2018 *21st International Conference on Intelligent Transportation Systems (ITSC)*, Maui, HI, USA, 2018, pp. 2575-2582, doi: 10.1109/ITSC.2018.8569938.

[11] L. Liu, Z. Chang and X. Guo, "Socially Aware Dynamic Computation Offloading Scheme for Fog Computing System With Energy Harvesting Devices," in *IEEE Internet of Things Journal*, vol. 5, no. 3, pp. 1869-1879, June 2018, doi: 10.1109/JIOT.2018.2816682.

[12] A. Bozorgchenani, D. Tarchi and G. E. Corazza, "Centralized and Distributed Architectures for Energy and Delay Efficient Fog Network-Based Edge Computing Services," in *IEEE Transactions on Green Communications and Networking*, vol. 3, no. 1, pp. 250-263, March 2019, doi: 10.1109/TGCN.2018.2885443.

[13] J. Du, L. Zhao, J. Feng and X. Chu, "Computation Offloading and Resource Allocation in Mixed Fog/Cloud Computing Systems With Min-Max Fairness Guarantee," in *IEEE Transactions on Communications*, vol. 66, no. 4, pp. 1594-1608, April 2018, doi: 10.1109/TCOMM.2017.2787700.

[14] J. Du, L. Zhao, X. Chu, F. R. Yu, J. Feng and C. -L. I, "Enabling Low-Latency Applications in LTE-A Based Mixed Fog/Cloud Computing Systems," in *IEEE Transactions on Vehicular*



شکل ۱۱- بررسی تأثیر پارامتر $\boldsymbol{\delta}$ در شبیه‌سازی با ۲۵ واحد مه و ۷۰ دستگاه کاربری در مدت ۳۰ دقیقه

[15] H. Shah-Mansouri and V. W. S. Wong, "Hierarchical Fog-Cloud Computing for IoT Systems: A Computation Offloading Game," in *IEEE Internet of Things Journal*, vol. 5, no. 4, pp. 3246-3257, Aug. 2018, doi: 10.1109/JIOT.2018.2838022.

[16] M. Liu, F. R. Yu, Y. Teng, V. C. M. Leung and M. Song, "Distributed Resource Allocation in Blockchain-Based Video Streaming Systems With Mobile Edge Computing," in *IEEE Transactions on Wireless Communications*, vol. 18, no. 1, pp. 695-708, Jan. 2019, doi: 10.1109/TWC.2018.2885266.

[17] H. Ye, G. Y. Li and B. -H. F. Juang, "Deep Reinforcement Learning Based Resource Allocation for V2V Communications," in *IEEE Transactions on Vehicular Technology*, vol. 68, no. 4, pp. 3163-3173, April 2019, doi: 10.1109/TVT.2019.2897134.

[18] F. Chai, Q. Zhang, H. Yao, X. Xin, R. Gao and M. Guizani, "Joint Multi-Task Offloading and Resource Allocation for Mobile Edge Computing Systems in Satellite IoT," in *IEEE Transactions on Vehicular Technology*, vol. 72, no. 6, pp. 7783-7795, June 2023, doi: 10.1109/TVT.2023.3238771.

[19] T. X. Tran and D. Pompili, "Joint Task Offloading and Resource Allocation for Multi-Server Mobile-Edge Computing Networks," in *IEEE Transactions on Vehicular Technology*, vol. 68, no. 1, pp. 856-868, Jan. 2019, doi: 10.1109/TVT.2018.2881191.

[20] Y. Shi, S. Chen and X. Xu, "MAGA: A Mobility-Aware Computation Offloading Decision for Distributed Mobile Cloud Computing," in *IEEE Internet of Things Journal*, vol. 5, no. 1, pp. 164-174, Feb. 2018, doi: 10.1109/JIOT.2017.2776252.

[21] S. -S. Lee and S. Lee, "Resource Allocation for Vehicular Fog Computing Using Reinforcement Learning Combined With Heuristic Information," in *IEEE Internet of Things Journal*, vol. 7, no. 10, pp. 10450-10464, Oct. 2020, doi: 10.1109/JIOT.2020.2996213.

[22] Z. Zhang, Z. Chen, Y. Shen, X. Dong, and N. Xi, "A Dynamic Task Offloading Scheme Based on Location Forecasting for Mobile Intelligent Vehicles," *IEEE Trans. Veh. Technol.*, vol. PP, pp. 1–15, 2024, doi: 10.1109/TVT.2024.3351224.

[23] C. Chen, Y. Zeng, H. Li, Y. Liu and S. Wan, "A Multihop Task Offloading Decision Model in MEC-Enabled Internet of Vehicles," in *IEEE Internet of Things Journal*, vol. 10, no. 4, pp. 3215-3230, 15 Feb.15, 2023, doi: 10.1109/JIOT.2022.3143529.

[24] L. Liu, M. Zhao, M. Yu, M. A. Jan, D. Lan and A. Taherkordi, "Mobility-Aware Multi-Hop Task Offloading for Autonomous Driving in Vehicular Edge Computing and Networks," in *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 2, pp. 2169-2182, Feb. 2023, doi: 10.1109/TITS.2022.3142566.

[25] Z. Zhang and F. Zeng, "Efficient Task Allocation for Computation Offloading in Vehicular Edge Computing," in *IEEE Internet of Things Journal*, vol. 10, no. 6, pp. 5595-5606, 15 March15, 2023, doi: 10.1109/JIOT.2022.3222408.

[26] L. Zhao et al., "MESON: A Mobility-Aware Dependent Task Offloading Scheme for Urban Vehicular Edge Computing," in *IEEE Transactions on Mobile Computing*, vol. 23, no. 5, pp. 4259-4272, May 2024, doi: 10.1109/TMC.2023.3289611.

[27] C. Zhu et al., "Folo: Latency and Quality Optimized Task Allocation in Vehicular Fog Computing," in *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4150-4161, June 2019, doi: 10.1109/JIOT.2018.2875520.

[28] D. Wang, Z. Liu, X. Wang and Y. Lan, "Mobility-Aware Task Offloading and Migration Schemes in Fog Computing Networks," in *IEEE Access*, vol. 7, pp. 43356-43368, 2019, doi: 10.1109/ACCESS.2019.2908263.

[29] Y. Wen, W. Zhang and H. Luo, "Energy-optimal mobile application execution: Taming resource-poor mobile devices with cloud clones," in *2012 Proceedings IEEE International Conference on Computer Communications (INFOCOM)*, Orlando, FL, 2012, pp. 2716-2720.

[30] W. Hao and S. Yang, "Small Cell Cluster-Based Resource Allocation for Wireless Backhaul in Two-Tier Heterogeneous Networks With Massive MIMO," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 1, pp. 509-523, Jan. 2018.

[31] R. Yadav, W. Zhang, O. Kaiwartya, H. Song and S. Yu, "Energy-Latency Tradeoff for Dynamic Computation Offloading in Vehicular Fog Computing," in *IEEE Transactions on Vehicular Technology*, vol. 69, no. 12, pp. 14198-14211, Dec. 2020, doi: 10.1109/TVT.2020.3040596.

[32] W. Hao and S. Yang, "Small Cell Cluster-Based Resource Allocation for Wireless Backhaul in Two-Tier Heterogeneous Networks With Massive MIMO," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 1, pp. 509-523, Jan. 2018.

[33] E. Dahlman, S. Parkvall, and J. Skold, 4G: LTE/LTEAdvanced for Mobile Broadband. New York, NY, USA: Academic, 2013.

[34] C. Tang, X. Wei, C. Zhu, Y. Wang and W. Jia, "Mobile Vehicles as Fog Nodes for Latency Optimization in Smart Cities," in *IEEE Transactions on Vehicular Technology*, vol. 69, no. 9, pp. 9364-9375, Sept. 2020, doi: 10.1109/TVT.2020.2970763.

[35] W. Nasrin and J. Xie, "SharedMEC: Sharing Clouds to Support User Mobility in Mobile Edge Computing," *2018 IEEE International Conference on Communications (ICC)*, Kansas City, MO, USA, 2018, pp. 1-6, doi: 10.1109/ICC.2018.8422241.

[36] DLR-TS, "TAVF-Hamburg," GitHub. [Online]. Available: https://github.com/DLR-TS/sumo-scenarios/tree/main/TAVF-Hamburg. [Accessed: Sep. 23, 2024].