علوم رایانش و فناوری اطلاعات
نشریه علمی انجمن کامپیوتر ایران
مجلد ۲۲، شماره ۱، ۱۴۰۳
صفحات ۸۸-۹۷
مقاله عادی

الگوریتمی حریصانه برای نگاشت پویای سرویس‌های هوشمند در لبه شبکه خودرویی

*نویسنده مسئول، دریافت: ۰۲/۱۰/۲۶، بازنگری: ۰۲/۱۲/۱۶، پذیرش: ۰۳/۰۱/۲۸ (تاریخ‌ها توسط نشریه وارد می‌شوند)

چکیده

در رایانش لبه شبکه خودرویی[1]، پردازش بی‌درنگ درخواست‌ها در عین جابجایی خودروها بین سلول‌ها، نیازمند تخصیص پویای درخواست‌ها به سرورهاست تا بتوان تعداد پاسخ‌های بهنگام به درخواست‌های خدمت را بیشینه کرد. بهترین روش‌های پیشین همچنان زمان اجرای بالایی دارند و درنتیجه در ابعاد بزرگ، فاصله‌ی زیادی از بهترین تخصیص پیدا می‌کنند چون نمی‌توان آن‌ها را در بازه‌های زمانی کوتاه اجرا کرد. ما یک الگوریتم حریصانه ارائه می‌دهیم که در عین سرعت اجرای بالا، می‌تواند به پاسخ بهینه نزدیک شود. نتایج ارزیابی الگوریتم، با استفاده از شبیه‌سازی انجام‌شده، نشان می‌دهد که نسبت به بهترین روش رقیب، روش ما ۴۹٪ پاسخ‌های بهنگام بیشتری به درخواست‌های خودروها ارائه می‌کند و فاصله‌ی آن با بهترین تخصیص ممکن که با MILP به‌دست‌آمده، تنها ۶.۸٪ است.

کلمات کلیدی: ابعاد بزرگ، پاسخ بهنگام، خدمات بهنگام، رایانش لبه شبکه، شبکه خودرویی، نگاشت پویای درخواست‌ها.

۱- مقدمه

یکی از چالش‌های محاسبات لبه شبکه خودرویی در شهرهای بزرگ، اجرای بی‌درنگ نگاشت پویای سرویس‌های درخواستی به منابع محاسباتی لبه شبکه است. مجموع منابع محاسباتی کل شبکه ممکن است بیش از نیاز کل درخواست‌ها در هرلحظه باشد، اما اگر نگاشت پویا و بی‌درنگ به‌خوبی انجام نشود، تعداد زیادی از درخواست‌ها ممکن است پاسخ بهنگام دریافت نکنند. اهمیت این موضوع بخصوص با در نظر گرفتن جابجایی خودروها، پویایی تعداد و منابع موردنیاز هریک از درخواست‌های دریافتی از آن‌ها، بخصوص در ساعت‌های اوج، بیشتر هم می‌شود.

دریکی از بهترین روش‌های اخیر، تلپور و همکارانش[1] از ترکیب برنامه‌ریزی خطی عدد صحیح مختلط و یادگیری تقویتی برای نگاشت سرویس‌های درخواستی بر روی سرورهای لبه موبایل استفاده می‌کنند .در هنگام یادگیری، بخش Actor سیستم یادگیری تقویتی، در ابتدا از برنامه‌ریزی خطی عدد صحیح مختلط استفاده کرده و پاسخ نزدیک به بهینه را برای کمینه کردن متوسط تأخیر اجرای سرویس در سمت خودرو پیدا می‌کند. سپس بخش Critic، تصمیم‌گیری بخش Actor را ارزیابی می‌کند و به آن بازخورد می‌دهد. درنهایت پس از مرحله‌ی یادگیری، جدول ارزش کیفی Q ساخته می‌شود که Actor برای تصمیم‌گیری‌های بعدی از آن استفاده می‌کند. به‌این‌ترتیب سرعت نگاشت درخواست‌ها به سرورها در [1] بالا است چراکه فقط با بررسی یک ارزش کیفیQ ، بالاترین ارزش را انتخاب و سرور را تعیین می‌کند؛ اما مشکلی که در اینجا وجود دارد و به‌عنوان کاستی این رویکرد برای ما حائز اهمیت است، مربوط به تأخیر حاصل از به‌روزرسانی جدول Q است. در طول عملکرد سیستم، لازم است مقادیر Q در زمان‌های مختلف برحسب تغییر در جابجایی شبکه خودرویی و تغییر در تعداد درخواست‌ها و توزیع جغرافیایی آن‌ها، دوباره بروز رسانی شوند. ایشان در این بروز رسانی دوباره از راهکار برنامه‌ریزی خطی عدد صحیح مختلط برای یادگیری استفاده می‌کنند .در اینجا است که با افزایش خودروها و تعداد سرورهای لبه شبکه، طول بازه‌های زمانی بروز رسانی بزرگ می‌شود، چون بایستی راهکار برنامه‌ریزی خطی عدد صحیح مختلط برای یادگیری مجدد اجرا شود که با افزایش خودروها و تعداد سرورها زمان اجرای آن به‌صورت نمایی رشد می‌کند .با بزرگ شدن بازه‌های یادگیری و به عبارتی بزرگ شدن بازه‌های بروز رسانی، یا نگاشت سرویس دیگر به‌صورت بی‌درنگ انجام نمی‌گیرد، یا نگاشتی دور از بهینه انجام می‌شود . درنتیجه ارائه سرویس به خودروها با تغییر جابجایی و تغییر درخواست‌ها، با تأخیر مواجه می‌شود که درنهایت باعث کاهش تعداد پاسخ‌های بهنگام به درخواست‌های دریافتی می‌گردد.

ما الگوریتم حریصانه‌ای را ارائه می‌کنیم که می‌تواند راه‌حل تقریبی برای مسئله برنامه‌ریزی خطی عدد صحیح مختلط باشد. درروش برنامه‌ریزی فوق کلیه درخواست‌ها برای تمام سرویس‌ها از کلیه نقاط دسترسی و همه سرورها همگی باهم بررسی‌شده و سعی می‌شود تابع هدف بهینه شود. ولی در این روش حریصانه، ما ضمن استفاده از همان تابع هدف روش برنامه‌ریزی خطی، روش بهینه‌سازی را طوری اصلاح می‌کنیم که درخواست‌ها از سمت نقاط دسترسی یکی پس از دیگری موردبررسی قرار گیرند، به‌طوری‌که به ازای هر نقطه دسترسی، تابع هدف مربوطه را به حداقل برساند و سپس نقاط دسترسی بعدی بررسی شوند؛ یعنی اگر به‌طور مثال ۱۰ نقطه دسترسی داشته باشیم و از هرکدام درخواست‌های خودروها دریافت شوند، الگوریتم پیشنهادی ما ابتدا به نقطه دسترسی شماره ۱ نگاه می‌کند و برحسب تعداد درخواست‌ها، آن‌ها را با توجه به تابع هدف به سرورهای لبه شبکه تخصیص می‌دهد و بعد نقطه دسترسی شماره ۲ و به همین ترتیب تا آخر. هنگامی‌که چند درخواست در نقاط دسترسی به‌طور هم‌زمان می‌رسند، یک توالی ورود دلخواه به آن‌ها اختصاص داده می‌شود و سپس درخواست‌ها همچنان با استفاده از الگوریتم بالا به سرورهای محاسباتی تخصیص داده می‌شوند. به عبارتی ما فضای حالت را در الگوریتم حریصانه کمتر می‌کنیم تا بتواند سریع‌تر تصمیم بگیرد.

۲- پیشینه پژوهش

می‌توان محدوده کارهای مرتبط را به دو بخش ۱ .مدیریت سرویس‌ها در لبه موبایل و ۲ .بهبود زمان اجرای الگوریتم‌های نگاشت سرویس‌ها و تخصیص منابع در لبه شبکه برای شهرهای بزرگ تقسیم کرد.

۲-۱- مدیریت سرویس‌ها در لبه موبایل

تمرکز تعدادی از پژوهش‌های اخیر بر روی سازوکار استقرار سرویس‌های هوشمند بر روی لبه شبکه است؛ هدف، تضمین کیفیت سرویس‌ها مانند حفظ حداقل تأخیر در اجرای سرویس‌ها و یا به حداکثر رساندن درآمد ارائه‌دهندگان شبکه است. برای مثال، رویکرد پیشنهادشده در [۲] از شبکه نرم‌افزار محور با محاسبات مه به‌عنوان فناوری‌های آینده برای شبکه VANET[2] استفاده می‌کند. هدف رویکردهای ارائه‌شده در [۳،۴] ارائه یک چارچوب مبتنی بر شبکه نرم‌افزار محور با کنترل توزیع‌شده برای مدیریت سرویس‌های هوشمند باقابلیت مدیریت جابجایی کاربران است.

نویسندگان [۵] یک معماری سلسله مراتبی مبتنی بر ابر خودروها، ابر کنار جاده و یک ابر مرکزی را باهدف استفاده بهینه از منابع محاسباتی، پیشنهاد می‌دهند. در [۶]، نویسندگان یک شبکه خودرویی مبتنی بر MEC[3] جهت ارائه سرویس‌های هوشمند پیشنهاد کردند که در آن از شبکه نرم‌افزار محور استفاده می‌شود. شاه و همکارانش در مقاله حاضر، مسئله تضمین کیفیت سرویس‌ها را با ارائه ماژول‌های نرم‌افزاری که در کنترلر شبکه نرم‌افزار محور توسعه داده می‌شود، بررسی کرده‌اند.

این دسته پیشنهادها از یک مشکل مشترک رنج می‌برند: آن‌ها توضیح نمی‌دهند که با افزایش تعداد خودروها، تعداد سرویس‌ها و سرورهای لبه شبکه در شهرهای بزرگ، چگونه و با چه سازوکاری منابع پردازشی موجود در سرورهای لبه شبکه باید به درخواست‌های سرویس از جانب خودروها اختصاص یابند تا بتوان حداکثر بازدهی در پاسخگویی به کاربران را فراهم نمود.

۲-۲- الگوریتم‌های نگاشت سرویس‌ها و تخصیص منابع در لبه شبکه خودرویی

مرجع [۷] بررسی مقیاس‌پذیری شبکه را به‌عنوان یکی از چالش‌ها و کارهای آینده معرفی می‌کند. رویکرد پیشنهادشده در [۸] یک مسئله برنامه‌ریزی خطی عدد صحیح برای قرار دادن بهینه سرویس‌های V2X[4] با در نظر گرفتن تأخیر و الزامات محاسباتی در محیطی که شامل گره‌های لبه و ابر است، ارائه کرده است. هدف در مرجع [۹] به حداقل رساندن استفاده از زیرساخت‌های فناوری اطلاعات برای قرار دادن ماشین‌های مجازی ارائه‌کننده سرویس در محاسبات لبه ابری است به صورتی که نیازهای تأخیر ناهمگن سرویس‌های مختلف را برآورده کند. مرجع [۱۰] به‌قرار دادن کارآمد سرویس‌های V2X با در نظر گرفتن محدودیت‌های تأخیر آن‌ها و منابع محاسباتی محدود در لبه می‌پردازد. آن‌ها یک مدل برنامه‌ریزی خطی عدد صحیح دودویی را برای به حداقل رساندن تأخیر اجرای سرویس‌ها برای پنج سرویس مختلف پیشنهاد می‌کنند. در مرجع [11]، یک مکانیسم جدید با نام "نگاشت خدمات با تمرکز بر میکرو سرویس‌ها" یا MOSP[5] برای اینترنت اشیاء وسایل نقلیه (IoV)که از محاسبات لبه موبایل (MEC) پشتیبانی می‌کند، پیشنهادشده است. هدف از این مکانیسم، کاهش تأخیر سرویس، کاهش مصرف منابع و اطمینان از پایداری درازمدت است. مرجع [12] به بررسی نگاشت سرویس‌های حساس به تأخیر و نیازمند به محاسبات گسترده در لبه، باهدف کاهش تأخیر شبکه می‌پردازد. گره‌های مه غبار (VNFs)، گره‌های مه غبار سیار که توسط وسایل نقلیه حمل می‌شوند، برای مدیریت تغییرات مکانی-زمانی درخواست‌ها پیشنهادشده‌اند. در مرجع [13]، یک معماری سلسله مراتبی ابر-لبه برای ارائه خدمات به خودروها از طریق زنجیره‌های توابع شبکه مجازی موردبررسی قرارگرفته است. هر خدمت دارای نیازهای خاص محاسباتی و تأخیر معینی است که نیاز به نگاشت مناسب زنجیره و تخصیص منابع محاسباتی دارد. همچنین، ممکن است برای دستیابی به تأخیر قابل‌قبول برای خدمات، مهاجرت زنجیره نیز لازم باشد. این مسئله به‌صورت ریاضی مدل‌سازی شده است تا یک راه‌حل قابل‌قبول برای قرارگیری زنجیره و تخصیص منابع پیدا کند به صورتی که در آن، مهاجرت، پهنای باند و هزینه محاسباتی کمینه شود.

نویسندگان در [۱۴]، چارچوبی را برای ارائه خدمات پویا و آگاه از کیفیت خدمات[6] (QoS) با استفاده از محاسبات مه پیشنهاد کرده‌اند. دستاورد مقاله حاضر یک مسئله بهینه‌سازی برنامه‌ریزی غیرخطی عدد صحیح و دو الگوریتم حریصانه برای توسعه پویای خدمات لبه و در عین حال به حداقل رساندن هزینه و برآورده کردن محدودیت‌های QoS است. نویسندگان در مرجع [۱] یک استراتژی استقرار مجدد سرویس‌های هوشمند را پیشنهاد کرده‌اند که الگوریتمActor-Critic را برای به حداقل رساندن تأخیر اجرای سرویس‌ها و مصرف منابع پردازشی به کار گرفته‌اند. در قسمت Actor آن که تصمیم‌گیری انجام می‌گیرد، از یک مدل برنامه‌ریزی خطی عدد صحیح برای یادگیری مدل استفاده می‌کند و در قسمتCritic ، تصمیم‌گیری بخش Actor ارزیابی می‌شود. نویسندگان در مرجع [۱۵] یک مدل پویا برای قرار دادن سرویس‌های اضطراری ازجمله سرویس اورژانس در شهرهای هوشمند بر اساس برخی از الزامات کیفیت خدمات، مانند تعادل بار و تأخیر خدمات ارائه کرده‌اند.

پیشنهادهای بالا از مشکلات زیر رنج می‌برند: ۱) در یکسری از کارهای مرتبط بیان‌شده [۱۰-۲] راهکار ایستا برای قرار دادن سرویس‌های هوشمند روی سرورهای لبه ارائه کرده‌اند. درحالی‌که راهکارهای ایستا برای کاربرانی که در حال حرکت می-باشند کارآمد نیستند. ۲) در استفاده کارآمد از مدل‌های محاسباتی لبه موبایل که درمراجع [۱۵-۱۱ و ۱] بالا بیان گردید، با افزایش تعداد خودروها، تعداد سرورهای لبه و تعداد سرویس‌های هوشمند، مسئله تخصیص منابع لبه موبایل به‌صورت پویا و در زمان قابل‌قبول، اصلی‌ترین چالش پیش روی آن‌ها است.

با توجه به این مشکلات، ما در این پژوهش یک الگوریتم کارآ برای تخصیص منابع لبه موبایل در شبکه‌های خودرویی ارائه می‌دهیم که مبتنی بر تحرک خودروها و پویایی در درخواست‌ها عمل می‌کند و زمان اجرای عملیات تخصیص سرورها به سرویس‌ها را در شهرهای بزرگ بهبود می‌دهد.

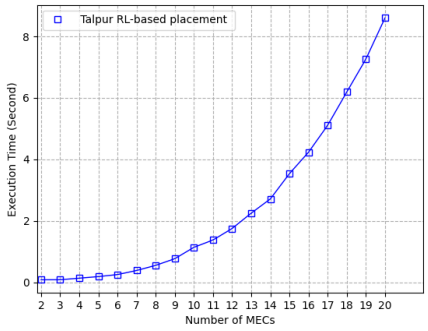
۳- انگیزه

بهترین راهکار فعلی [۱]، از برنامه‌ریزی خطی عدد صحیح برای یافتن بهترین نگاشت در بخش یادگیری استفاده کرده و در بازه‌های زمانی خاصی این نگاشت را بازنگری می‌کند. ما نشان می‌دهیم که فاصله زیادی بین پاسخ این روش و بهترین نگاشت ممکن وجود دارد و الگوریتم ما به کاهش آن شکاف خواهد پرداخت. آزمایش اول به بررسی تأثیر تعداد سرورهای لبه موبایل بر روی زمان حل مسئله در [۱] می‌پردازد. در آزمایش دوم نشان می‌دهیم که رشد نمایی زمان حل مسئله در [۱] باعث می‌شود تعداد کاربرانی که پاسخ بی‌درنگ دریافت می‌کنند از تعداد بیشینه فاصله‌ی بزرگی پیدا کند و مهم‌تر این‌که این فاصله با افزایش تعداد سرورهای لبه رو به فزونی می‌گذارد.

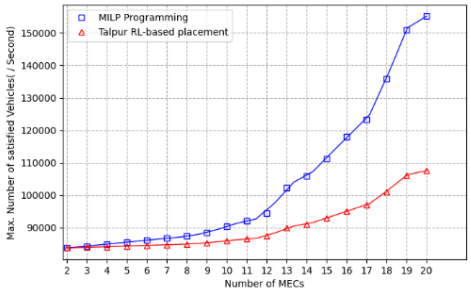
پیکربندی آزمایش‌ها در بخش ۱-۵ آمده است. آزمایش اول، مسئله بهینه‌سازی مطرح در [۱] را باهمان روش پیشنهادی‌شان، یعنی برنامه‌ریزی خطی عدد صحیح، حل کرده و زمان رسیدن به پاسخ را گزارش می‌کند. شکل ۱ نمودار زمان حل مسئله نسبت به تعداد سرورهای لبه شبکه را نمایش می‌دهد؛ زمان اجرا به‌صورت نمایی با بیشتر شدن تعداد سرورهای لبه موبایل رشد می‌کند.

با تغییر در جابجایی بار شبکه خودرویی یا تغییر در تعداد درخواست‌ها، بهترین نگاشت خدمات به سرورها، برای بیشینه کردن تعداد کاربرانی که پاسخ بهنگام دریافت می‌کنند، دچار تغییر می‌شود. برای مقایسه، ما به‌صورت برون‌خط، بهترین نگاشت ممکن را در هر حالت پیداکرده و درنتیجه بیشترین تعداد کاربرانی را که می‌توانند پاسخ بهنگام دریافت کنند، محاسبه می‌کنیم. برای این کار یک مسئله برنامه‌ریزی خطی عدد صحیح مختلط تعریف کرده و آن را حل کردیم. عدد حاصل در نمودار شکل ۲ به رنگ آبی دیده می‌شود. تعدادی را نیز که پاسخ [۱] برآورده می‌کند، در همان شکل به رنگ قرمز نمایش داده‌ایم .نتایج نشان می‌دهند که با

افزایش تعداد سرورهای لبه‌ی شبکه، فاصله‌ی پاسخ [۱] از بهترین پاسخ رو به فزونی می‌گذارد. دلیل این امر، همان افزایش نمایی زمان حل مسئله بهینه‌سازی است که باعث می‌شود [۱] نتواند با سرعت کافی به تغییرات در شرایط مسئله واکنش نشان دهد.

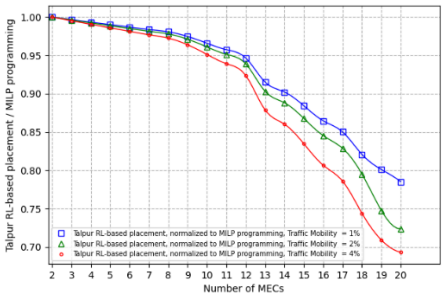


شکل ۱: زمان اجرای الگوریتم MILP با افزایش تعداد سرورهای لبه موبایل



شکل ۲: تعداد پاسخ‌های بهنگام برای الگوریتم‌ها (بهینه و تلپور)

در سناریوی دیگری برای نمایش تأثیر جابجایی خودروها بر تعداد پاسخ‌های بهنگام، فرض می‌کنیم هر سرور MEC می‌تواند ۱۰۰۰۰ خودرو را کنترل کند و ۱ درصد این تعداد خودرو در ثانیه، می‌توانند از مرزهای مناطق تحت کنترل سرورهای MEC، جابجا شوند و آن را به‌عنوان پارامتر Traffic Mobilityدر نظر می‌گیریم. نمودار شکل ۳، برای مقادیر مختلف جابجایی کاربران در هر ثانیه به‌صورت هنجار[۷] شده نسبت به بهترین پاسخ رسم شده است. با افزایش درصد جابجایی کاربران و به ترتیب دو و چهار برابر کردن آن نسبت به مقدار یک درصد، دو نمودار دیگر به ترتیب بارنگ‌های سبز و قرمز کشیده شده‌اند. نمودار آبی برای Traffic Mobility یک درصد است. در اینجا نیز مشاهده می‌شود که با افزایش تعداد جابجایی کاربران، پاسخ [۱] نسبت به بهترین پاسخ فاصله حتی بیشتری پیدا می‌کند.



شکل ۳: نمودار هنجار شده پاسخ [۱] نسبت به بهترین پاسخ برای درصدهای مختلف جابجایی خودروها

٤- الگوریتم حریصانه

مسئله مدنظر ما، بیشینه کردن تعداد پاسخ‌های بهنگام به درخواست‌های سرویس‌های هوشمند در شبکه خودرویی شهرهای بزرگ است با در نظر گرفتن جابجایی خودروهای درخواست‌کننده و تغییر در تعداد درخواست‌ها. استفاده از الگوریتم حریصانه به دلیل سرعت اجرا و کارایی در مسائل مرتبط با شبکه‌های خودرویی با توجه به کارهای مرتبط بررسی‌شده در این پژوهش مدنظر ما قرار گرفت.

ازآنجایی‌که ترافیک درخواست‌های ورودی در طول زمان در حال تغییر است، نگاشت سرویس‌ها به سرورها بایستی به‌طور پویا در طول زمان تنظیم شود. یک رویکرد برای انجام این کار، حل مسئله به‌صورت دوره‌ای در بازه‌های زمانی مشخص است. در این حالت درخواست‌ها و بارکاری سرورها در طی هر بازه‌ی زمانی، «ثابت» فرض می‌شود.

در دو قسمت بعدی، ابتدا صورت‌مسئله بهینه‌سازی در یک بازه زمانی ارائه می‌شود. سپس، یک الگوریتم حریصانه را ارائه می‌کنیم که به‌صورت دوره‌ای برای حل مسئله بالا فراخوانی می‌شود.

٤-١- تعریف مسئله

از نمادهای جدول‌های ۱ و ۲ در تعریف مسئله استفاده می‌کنیم.

جدول ۱: نمادهای مورداستفاده برای تعیین پارامترهای ورودی

توصیف	پارامترهای ورودی
مجموعه سرورهای MEC	M
مجموعه سرویس‌ها	S
مجموعه نقاط دسترسی	A
ظرفیت سخت‌افزاری موجود در سرور m	H_m
میزان ظرفیت سخت‌افزاری که برای پاسخ به یک نمونه از سرویس s مصرف می‌شود	C_s
تأخیر انتشار بین هر دو جفت سرور MEC (ثانیه)	$l_{m1,m2}$
آستانه تأخیر سرویس s (ثانیه)	T_s
نرخ ارائه سرویس s در سرور m (req/s)	μ^s_m
نرخ درخواست سرویس s در نقطه دسترسی a (req/s)	R^s_a

جدول ۲: نمادهای مورداستفاده برای تعیین پارامترهای قابل‌کنترل

توصیف	پارامترهای قابل‌کنترل
تعداد واحد سخت‌افزاری تخصیص داده‌شده در سرور m برای ارائه سرویس s	N^s_m
یک نشانه تخصیص صفر و یا یک است که مشخص می‌کند که آیا $SR^s_{src,dst}$ قابل تخصیص به سرور مقصد است و یا نه	$X^s_{src,dst}$
تعداد درخواست‌های سرویس S از سرور مبدأ که در سرور مقصد پاسخ‌داده‌شده است. (پاسخ بهنگام) (ResponedServiceRequest: RSR)	$RSR^s_{src,dst}$
مقدار پردازش تخصیص داده‌شده در سرور مقصد برای $RSR^s_{src,dst}$ (AllocatedService: AS)	$AS^s_{src,dst}$

مسئله بهینه‌سازی مدنظر مقاله حاضر عبارت است از:

در بازه‌ی زمانی جاری، به ازای مقادیر داده‌شده برای مجموعه سرورهای لبه موبایل، مجموعه نقاط دسترسی، مجموعه سرویس‌های هوشمند، آستانه تأخیر برای هر سرویس، ظرفیت سخت‌افزار هر سرور، بارکاری فعلی هر سرور و تأخیر انتشار مابین هر جفت سرور، مجموع سرویس‌های درخواستی در بازه‌ی زمانی جاری در تمام نقاط دسترسی را به‌گونه‌ای به سرورهای لبه موبایل تخصیص بدهد که تعداد پاسخ‌های بهنگام به درخواست‌های کاربران بیشینه شود.

مسئله فوق را می‌توان به شکل زیر مدل‌سازی ریاضی کرد:

هدف:

$$\max \sum_{src\in A}\sum_{dst\in M}\sum_{s\in S} RSR^s_{src,dst} * X^s_{src,dst} \quad (1)$$

محدودیت‌ها:

$$\sum_{s\in S} C_s * N^s_m \le H_m \; m \in M \quad (2)$$

$$\sum_{dst\in M} \left(RSR^s_{src,dst}\right) \le R^s_{src} \; src \in A, dst \in M, s \in S \quad (3)$$

$$\sum_{src\in A} \left(AS^s_{src,dst}\right) \le N^s_{dst} * \mu^s_{dst} \; dst \in M, s \in S \quad (4)$$

$$RSR^{s}_{src,dst} \le AS^{s}_{src,dst} - \frac{X^{s}_{src,dst}}{T_s - 2 * l_{src,dst}} \quad src \in A,\ dst \in M,\ s \in S \ (5)$$

$$X^{s}_{src,dst} * (T_s - 2 * l_{src,dst}) \ge 0 \quad src \in A,\ dst \in M,\ s \in S \ (6)$$

$$X^{s}_{src,dst} = 0 \leftrightarrow RSR^{s}_{src,dst} = 0 \quad src \in A,\ dst \in M,\ s \in S \ (7)$$

هدف (۱) مجموع تعداد درخواست‌های پاسخ‌داده‌شده برای تمام درخواست‌های خودروها را محاسبه می‌کند و این مدل می‌خواهد آن را به حداکثر برساند. محدودیت ظرفیت سخت‌افزار در (۲) مدل شده و تضمین می‌کند که منابع سخت‌افزاری اشغال‌شده توسط نمونه‌های سرویس در هر سرور، از ظرفیت سخت‌افزار آن تجاوز نکند. نامعادله (۳) الزام می‌کند که مجموع درخواست‌های پاسخ‌داده‌شده از هر نقطه دسترسی، نمی‌تواند از کل درخواست‌های ورودی بیشتر شود. نامعادله‌ی (۴) توانایی سرویس‌دهی به ازای هر واحد سخت‌افزار را لحاظ می‌کند؛ این محدودیت الزام می‌کند که مقدار پردازش استفاده‌شده برای سرویس S در سرور مقصد، نمی‌تواند از نرخ سرویس قابل‌ارائه توسط سرور فوق تجاوز کند. قید (۵) تضمین می‌کند که تأخیر سرویس کمتر و یا مساوی آستانه تأخیر باشد. نامعادله (۶) تضمین می‌کند که درصورتی‌که تأخیر رفت‌وبرگشت بین مبدأ و مقصد بزرگ‌تر از آستانه باشد مقدار نشانه تخصیص صفر می‌شود. گزاره (۷) نیز بیان می‌کند که درصورتی‌که نشانه تخصیص صفر باشد تعداد درخواست پاسخ‌داده‌شده در مقصد نیز بایستی صفر گردد.

۴-۲- الگوریتم حریصانه پیشنهادی

شبهه کد شکل ۴، الگوریتم پیشنهادی ما را نمایش می‌دهد.

Algorithm: Greedy Service-to-Server Mapper

Input:
- $A = \{A_1, A_2, ..., A_{|A|}\}$ access points in an arbitrary order,
- $S = \{S_1, S_2, ..., S_{|S|}\}$ service requests of each type at each access point,
- $M = \{M_1, M_2, ..., M_{|M|}\}$ list of MEC servers,
- $l_{M1,M2}$, MEC-to-MEC latencies,
- R^{s}_{a}, Request rate for service s at access point a,
- H_m, Available hardware resource at MEC server m,
- μ^{s}_{m}, Service Rate s in each Hardware Unit at MEC server m,
- T_s, latency threshold for service s.

Output:
1, Mappings of each service request to nearest MEC servers: allocated[s][a][m] s∈S, a∈A, m∈M
2, Where the algorithm fails to find a MEC server for the request, notallocated[s][a] represents a failure

1: Initialize allocated array; allocated[s][a][m] = {0} and notallocated[s][a] = {0} for all s∈S and a∈A and m∈M
۲: Initialize the number of the request for service s at access point a ; $R^{s,a}_{num} = R^{s}_{a}$ for all s∈S and a∈A
۳: Initialize Remind processing capacity; $RemCap_m = H_m * \mu^{s}_{m}$ for all m∈M
٤: Sort the service requests, S, based on ascending order of their latency thresholds.
٥: **for** s in S **do**
٦: **for** a in A **do**// A can be in any arbitrary order
۷: **while**,$R^{s,a}_{num}$ **do**
۸: R_M = Set of MEC Servers, m, reachable within the acceptable latency threshold ($T_s - 2 * l_{a,m} > 0$) and $RemCap_m > 0$
۹: **if** (R_M == NULL)
10: notallocated[s][a] = notallocated[s][a] +$R^{s,a}_{num}$ // represents a failure
11:$R^{s,a}_{num}$ = 0
12: **else**
13: Sort R_M in ascending order of propagation delay from a
14: **for** m in R_M **do**
۱۵: **if** ($RemCap_m - (1/(T_s - 2 * l_{a,m})) > R^{s,a}_{num}$) then// The idea is that ($1/(T_s - 2 * l_{a,m})$) represents the additional service-rate required to compensate for the roundtrip communication delay between a and m
16: allocated[s][a][m] = allocated[s][a][m] +$R^{s,a}_{num}$ +($1/(T_s - 2 * l_{a,m})$) // Assign s on m
17:$R^{s,a}_{num}$ = 0
18: $RemCap_m = RemCap_m - R^{s,a}_{num} - (1/(T_s - 2 * l_{a,m}))$
19: **else**
20: allocated[s][a][m] = allocated[s][a][m] +$R^{s,a}_{num}$ +($1/(T_s - 2 * l_{a,m})$) - $RemCap_m$// Assign a part of s on m
21:$R^{s,a}_{num} = R^{s,a}_{num} - RemCap_m$
22: $RemCap_m$ = 0
23: **end if**
24: **end for**
25: **end if**
26: **end while**
27: **end for**
28: **end for**
29: **Return** allocated[s][a][m], notallocated[s][a]

شکل ۴: الگوریتم حریصانه پیشنهادی

در این قسمت، الگوریتم حریصانه‌ای را ارائه می‌کنیم که جواب‌های تقریباً بهینه را تولید می‌کند. روش برنامه‌ریزی خطی کلیه درخواست‌ها را برای تمام سرویس‌ها از کلیه نقاط دسترسی و همه سرورها همه باهم بررسی کرده، تابع هدف را بهینه می‌کرد. ولی در مورد روش حریصانه آنلاین، ما رویه ارائه‌شده در مسئله MILP را طوری اصلاح می‌کنیم که درخواست‌ها از سمت نقاط دسترسی جداجدا موردبررسی قرار گیرند، به‌طوری‌که به ازای هر نقطه دسترسی تابع هدف مدنظر (معادله ۱) را به حداکثر برساند؛ یعنی اگر به‌طور مثال ۱۰ نقطه دسترسی داشته باشیم و از هرکدام درخواست‌های خودروها دریافت شوند، الگوریتم آنلاین ابتدا به نقطه دسترسی شماره ۱ نگاه می‌کند و برحسب تعداد درخواست‌ها، آن‌ها را با توجه به تابع هدف در معادله شماره ۱ به سرورهای لبه موبایل تخصیص می‌دهد و بعد نقطه دسترسی شماره ۲ و الی تا آخر. هنگامی‌که درخواست‌ها در نقاط دسترسی به‌طور هم‌زمان می‌رسند، یک توالی ورود دلخواه به‌صورت تصادفی به آن‌ها اختصاص داده می‌شود و درخواست‌ها همچنان با استفاده از الگوریتم آنلاین ذکرشده به سرورهای محاسباتی لبه موبایل تخصیص داده می‌شوند. الگوریتم فوق در بازه‌های زمانی متوالی تکرار می‌شود و تخصیص درخواست‌ها از ابتدا اجرا می‌گردند.

خطوط ۵-۲۱ در الگوریتم به‌طور تکرارشونده اجرا می‌شود. در هر مرحله منابع محاسباتی موردنیاز برای هر درخواست سرویس که از نقاط دسترسی دریافت شده است را به سرورهای لبه شبکه تخصیص می‌دهد. برای هر سرویس (خط ۵)، درخواست‌ها از نقاط دسترسی به ترتیب شماره آن‌ها (خط ۶) وارد الگوریتم می‌شوند. الگوریتم فوق، فهرستی از سرورهای لبه موبایلی که ازنظر محدودیت تأخیر و فضای پردازشی آزاد قابل‌دسترس هستند را مهیا می‌کند (خط ۸). الگوریتم، درخواست‌های فوق را به نزدیک‌ترین گره محاسباتی تخصیص می‌دهد (خطوط ۱۳- ۲۵). در هر بار تخصیص پارامترهای کنترلی شامل تعداد درخواست‌های باقیمانده، تعداد درخواست پاسخ‌داده‌شده و مقدار منابع محاسباتی آزاد به‌روز می‌شوند. اگر گرهی پیدا نشد (خط ۹ – ۱۲) مقدار باقی‌مانده از درخواست‌ها در ارائه خروجی درخواست‌های پاسخ داده نشده ذخیره می‌شود. کل تعداد درخواست‌های پاسخ‌داده‌شده و پاسخ داده نشده خروجی الگوریتم است (خط ۲۹).

۵- ارزیابی

۵-۱- برپا سازی آزمایش‌ها

در جدول ۳ پارامترهای مهم در آزمایش‌ها بیان‌شده است. در بعضی از پارامترها نحوه انتخاب مقدار برای آن‌ها از مراجعی بوده که در کنار آن‌ها عنوان‌شده است. پارامتر «نرخ ارائه سرویس» برای سرویس‌های مختلف در جدول فوق با این استدلال به‌دست‌آمده‌اند. با توجه به مقدار آستانه تأخیر هر سرویس، دوسوم مقدار فوق را به‌عنوان زمان پردازش سرویس در نظر گرفته‌ایم. به‌این‌ترتیب مثلاً برای سرویس شماره ۱، آستانه آن ۱۰ میلی‌ثانیه است و دوسوم آن ۶.۶۶۶ = ۲۰/۳ میلی‌ثانیه می‌شود و «نرخ ارائه سرویس» عکس این مقدار است که می‌شود: ۱۵۰ = ۳۰۰۰/۲۰ که به عبارتی ۱۵۰ درخواست در ثانیه برای نرخ ارائه سرویس فوق به دست می‌آید. برای بقیه سرویس‌ها نیز به همین ترتیب است.

مقادیر پارامترهای ذکرشده در جدول بسته به نیاز در آزمایش‌ها، به‌عنوان تنظیمات پایه در نظر گرفته‌شده‌اند.

قابل‌ذکر است که تعداد نقاط دسترسی در هر سرور MEC، مساوی ۵۰ در نظر گرفته‌شده و هر نقطه دسترسی قابلیت اتصال به ۲۰۰ خودرو را دارد. لذا هر سرور MEC می‌تواند به‌طور متوسط ۱۰۰۰۰ خودرو را کنترل کند.

در سناریوی آزمایش‌هایی که نیاز به جابجایی خودروها است، فرض می‌کنیم ۴ درصد این تعداد خودرو در ثانیه، می‌توانند از مرزهای مناطق تحت کنترل سرورهای MEC، جابجا گردند و به‌عنوان پارامتر Traffic Mobility در جدول آورده شده است. این عدد به‌عنوان تنظیمات پایه برای آزمایش‌های دیگر در نظر گرفته‌شده و در قسمت ۵-۵ اثر مقادیر مختلف آن بررسی می‌شود.

تعداد درخواست‌ها برای همه آزمایش‌ها بر اساس اینکه هر سرور ۵۰ نقطه دسترسی دارد و هر نقطه دسترسی حدود ۲۰۰ خودرو به آن‌ها وصل می‌شود و با در

نظر گرفتن ۲۰ سرور لبه، ماکزیمم ۲۰۰ هزارتا در نظر گرفته‌شده که در صورت تغییر در هر آزمایش مقدار جدید ذکرشده است.

ما از Python و Pulp[15] برای راه‌اندازی محیط شبیه‌سازی استفاده می‌کنیم. نتایج حل مسئله بهینه‌سازی از PULP CBC Optimization به‌دست‌آمده است، درحالی‌که نتایج دیگر از شبیه‌ساز مبتنی بر پایتون در آزمایشگاه ما هستند. PuLP یک کتابخانه برای زبان برنامه‌نویسی پایتون است که به کاربران امکان می‌دهد برنامه‌های ریاضی را توصیف کنند. بسیاری از حل‌کننده‌های برنامه‌ریزی خطی عدد صحیح مختلط (MILP) قابل‌استفاده در PuLP هستند. در PuLp، حل‌کننده پیش‌فرض CBC(COIN Branch and Cut solver) است.

در شبیه‌سازی، ۲۰ ناحیه جغرافیایی مجهز به سرورهای MEC را در نظر گرفته‌ایم تا پوشش خودروها را فراهم کنند. هر منطقه مجهز به ۵۰ نقطه دسترسی جهت جمع‌آوری داده از خودروها بوده و به یک سرور MEC متصل می‌باشند.

نتایج شبیه‌سازی در شکل‌های ۵ الی ۱۵ نشان داده‌شده است. در تمام شکل‌ها، برچسب MILP Programming نشان می‌دهد که در آن درخواست‌هایی که از جانب خودروها می‌آیند از طریق حل مسئله بهینه‌سازی بر روی سرورهای MECای که در جواب تعیین می‌شود قرار می‌گیرند و به‌عنوان بهترین جواب مسئله است. نتایج شبیه‌سازی با ۱۰۰ مقدار تصادفی مختلف برای ورودی‌های تصادفی اجراشده است.

در نمودارها نام Our greedy placement نشان‌دهنده الگوریتم حریصانه پیشنهادی است و Talpur RL-based placement یکی از جدیدترین روش‌های رقیب است که در مرجع [۱] مطرح‌شده و برای مقایسه بکار گرفته‌شده است.

در مرجع [۱]، تعداد درخواست‌ها ۲۰۰، تعداد گره‌های لبه محاسباتی موبایل ۶ و تعداد سرویس‌ها ۸ بوده است.

۵-۲- پاسخ‌های بهنگام

جدول ۳: پارامترهای مهم در آزمایش‌ها

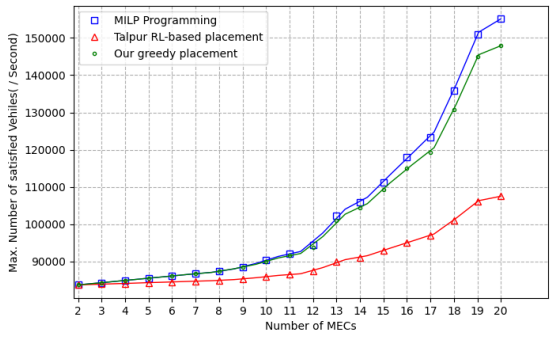
پارامتر	مقدار
Number of MEC servers	20
Number of Access Points controlled by each MEC server	50
Average number of vehicles at each Access Point	200
Available computation resources at MEC (unit)	100
Number of services	7
Maximum number of vehicles	200000
Service Rate Per Hardware Unit (req / sec)	[150, 75, 50, 37.5, 30, 25, 21.5]
Service Latency Threshold (millisecond) [13]	[10, 20, 30, 40, 50, 60, 70]
Network Propagation Latency Between MECs (millisecond) [14]	Table 4(در پیوست)
Traffic Mobility (Vehicles / sec / border)	۴٪ تعداد خودروها در هر MEC
timeslot duration: time interval between two instances of executing our greedy algorithm (second)	1
کامپیوتر استفاده‌شده در آزمایش‌ها	Core i7, 4-Core Intel حافظه ۱۶ گیگابایت و سیستم‌عامل ویندوز ۱۰

این آزمایش به بررسی تأثیر تعداد سرورهای لبه‌ی موبایل و جابجایی خودروها بر روی تعداد پاسخ بهنگام در [۱] و در مدل پیشنهادی می‌پردازد. با جابجا شدن خودروها، بهترین نگاشت خدمات به سرورها، برای بیشینه کردن تعداد کاربرانی که پاسخ بهنگام دریافت می‌کنند، دچار تغییر می‌شود.

در این آزمایش، برای مقایسه، ما به‌صورت برون‌خط، بهترین نگاشت ممکن را در هر حالت پیداکرده و درنتیجه بیشترین تعداد کاربرانی را که می‌توانند پاسخ بهنگام دریافت کنند، محاسبه می‌کنیم. برای این کار یک مسئله برنامه‌ریزی خطی عدد صحیح ترکیبی تعریف کرده و آن را حل کردیم. عدد حاصل در نمودار شکل ۵ به رنگ آبی دیده می‌شود. تعدادی را نیز که پاسخ [۱] برآورده می‌کند، در همان شکل به رنگ قرمز نمایش داده‌ایم. در مقایسه دیگر، با استفاده از الگوریتم پیشنهادی، بهترین نگاشت ممکن را در هر حالت پیداکرده و بیشترین تعداد کاربرانی را که می‌توانند پاسخ بهنگام دریافت کنند محاسبه می‌کنیم. این تعداد در شکل ۵ به رنگ سبز نشان داده‌شده است.

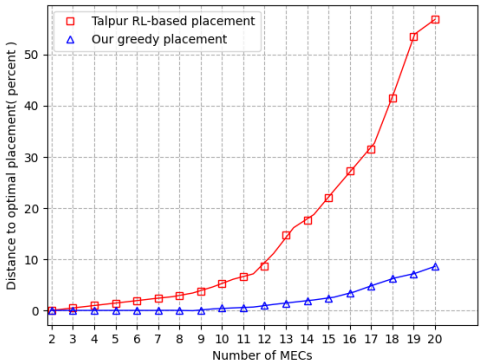
همچنین شکل ۵ به‌نوعی ارزیابی الگوریتم پیشنهادی و مقایسه آن نسبت به الگوریتم بهینه، می‌پردازد. به‌عبارت‌دیگر، مقایسه به ارزیابی نزدیکی عملکرد الگوریتم پیشنهادی به بهترین جایگذاری ممکن است.

هر چه تعداد پاسخ‌های بهنگام الگوریتم نزدیک‌تر به الگوریتم بهینه باشد، عملکرد آن بهتر ارزیابی می‌شود. در شکل ۵ در بدترین حالت نسبت تعداد پاسخ‌های بهنگام الگوریتم بهینه به الگوریتم پیشنهادی ۱.۰۷ است که نشان‌دهنده نزدیکی عملکرد الگوریتم پیشنهادی به بهترین جایگذاری ممکن است و به عبارتی فاصله بیشتری مابین این دو جواب با افزایش تعداد گره‌های MECاز ۲ تا ۲۰ ایجاد نمی‌شود. (وجود همگرایی در نسبت تعداد پاسخ‌های بهنگام الگوریتم بهینه به الگوریتم پیشنهادی)



شکل ۵: تعداد پاسخ‌های بهنگام هر الگوریتم (بهینه و تلپور و پیشنهادی) با افزایش تعداد سرورهای محاسباتی لبه موبایل

در شکل ۶ میزان فاصله پاسخ بهینه با روش پیشنهادی و روش مقاله پایه رسم شده است.



شکل ۶: درصد اختلاف بین الگوریتم بهینه با الگوریتم تلپور و الگوریتم پیشنهادی با افزایش تعداد سرورهای محاسباتی لبه موبایل

نتایج نشان می‌دهند که با افزایش تعداد سرورهای لبه‌ی شبکه، فاصله‌ی پاسخ [۱] از بهترین پاسخ رو به فزونی می‌گذارد. دلیل این امر، همان افزایش نمایی زمان حل مسئله بهینه‌سازی است که باعث می‌شود [۱] نتواند با سرعت کافی به تغییرات در شرایط مسئله واکنش نشان دهد. برعکس، فاصله پاسخ الگوریتم پیشنهادی از بهترین پاسخ کم است. دلیل آن به خاطر اجرای کوتاه‌مدت الگوریتم پیشنهادی در مقایسه بازمان حل مسئله بهینه‌سازی است که بنابراین الگوریتم فوق سریع‌تر به تغییرات در شرایط مسئله واکنش نشان می‌دهد.

در این آزمایش تعداد درخواست‌ها ۱۶۰۰۰۰ و جابجایی کاربران ۴ درصد تعداد خودروهای تحت کنترل هر سرور MEC است. در راهکار پیشنهادی، تعداد پاسخ بهنگام درزمانی که تعداد سرورهایMEC ، ۲۰ است، ۱۴۹۰۰۰ است که تفاوت آن

با بهترین پاسخ حدوداً ۶.۸ درصد است.

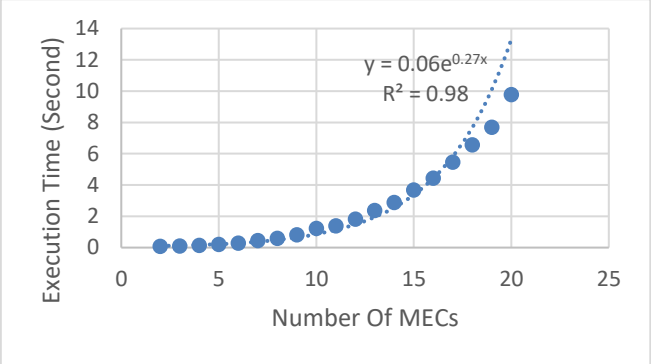
۵-۳- مدت‌زمان حل مسئله (اجرای الگوریتم)

این آزمایش به بررسی تأثیر تعداد سرورهای لبه‌ی موبایل بر روی زمان حل مسئله در [۱] و زمان اجرای الگوریتم در مدل پیشنهادی می‌پردازد. این آزمایش، مسئله بهینه‌سازی مطرح در [۱] را باهمان روش پیشنهادی‌شان، یعنی برنامه‌ریزی خطی عدد صحیح، حل کرده و زمان اجرای آن را گزارش می‌کند. شکل ۷ نمودار زمان حل مسئله نسبت به تعداد سرورهای لبه‌ی شبکه را نمایش می‌دهد؛ زمان اجرا به‌صورت نمایی ($y = 0.06e^{0.27x}$) با بیشتر شدن تعداد سرورهای لبه موبایل رشد می‌کند. تعداد سرورهای لبه شبکه در بازه ۱ تا ۲۰، به تعداد ۲۰ نقطه در نظر گرفته‌شده است.

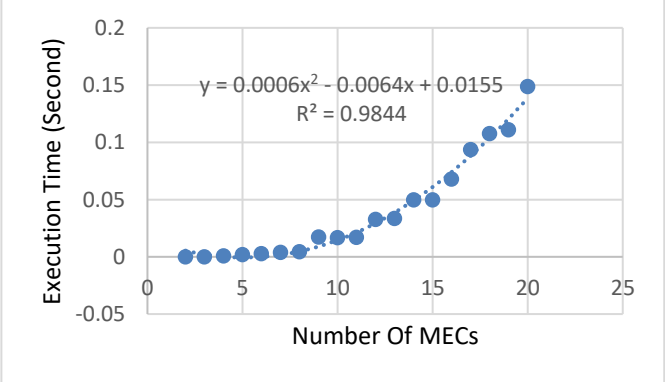
همچنین این آزمایش، الگوریتم پیشنهادی در بخش قبل را در شبیه‌ساز سیستم حل کرده و زمان اجرای آن را به‌عنوان خروجی ارائه می‌کند.

شکل ۸ زمان اجرای الگوریتم در مدل پیشنهادی را نمایش می‌دهد؛ زمان اجرا قابل‌مقایسه با یک معادله درجه ۲ بوده ($y = 0.0006x^2 - 0.0064x + 0.0155$) و با نرخ بسیار کمتری رشد می‌کند.

در نمودارها محور افقی تعداد سرورهای محاسباتی لبه موبایل و محور عمودی زمان اجرا برحسب ثانیه است.



شکل ۷: زمان اجرای الگوریتم مقاله پایه با افزایش تعداد سرورهای لبه موبایل



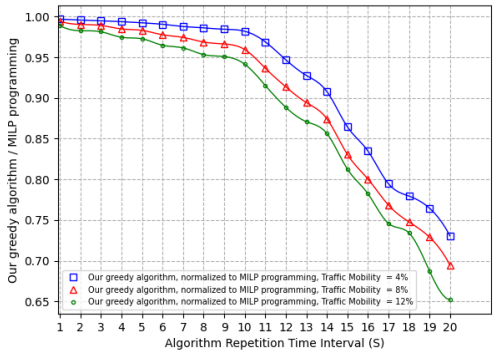
شکل ۸: نمودار زمان اجرای روش پیشنهادی با افزایش تعداد سرورهای لبه موبایل

نتیجه‌ای که از این آزمایش گرفته می‌شود آن است که مدل پیشنهادی با افزایش تعداد سرورهای لبه شبکه مدت‌زمان اجرای کوتاه‌تری نسبت به [۱] دارد. علت این کاهش این بوده است که درروش پیشنهادی درخواست‌ها از سمت نقاط دسترسی جداجدا، یکی پس از دیگری و به ترتیب اهمیت موردبررسی قرار می‌گیرند، به‌طوری‌که به ازای هر نقطه دسترسی بهترین نگاشت ممکن را در هر حالت پیدا می‌کنیم ولیکن در [۱] به‌صورت برون‌خط با بررسی کلیه درخواست‌ها باهم، بهترین نگاشت ممکن به دست می‌آید که زمان‌بر است و نمی‌تواند نسبت به تغییر مداوم محل خودروها با سرعت کافی واکنش نشان دهد.

۵-۴- اثر زیاد یا کم کردن طول بازه زمانی تکرار الگوریتم

شکل ۹ تأثیر طول بازه زمانی تکرار الگوریتم را بر روی روش پیشنهادی ما نشان می‌دهد. طول دوره تکرار باید به‌اندازه کافی کوچک باشد تا اجرای مکرر الگوریتم حریصانه بتواند پویایی تغییرات را به‌صورت برخط دنبال کند. این زمان همچنین باید به‌اندازه کافی طولانی باشد تا الگوریتم پیشنهادی کارش تمام شود. ما باید طول بازه زمانی تکرار الگوریتم را با توجه به مصالحه‌ی بالا انتخاب کنیم. با آزمایشی که مدت‌زمان اجرای الگوریتم را به دست می‌آورد، زمان اجرای الگوریتم پیشنهادی برای حالت‌های مختلف اندازه‌گیری می‌شود و بنابراین حداقل این زمان به دست می‌آید.

برای مقایسه، ما ۲۰ حالت مختلف مابین بازه ۱ تا ۲۰ ثانیه برای اجرای مجدد الگوریتم پیشنهادی در نظر گرفته‌ایم. در حالت‌های مختلف، با در نظر گرفتن جابجایی کاربران در هر ثانیه و هر بار با افزایش زمان اجرای مجدد الگوریتم، با استفاده از نگاشت قبلی، بیشترین تعداد کاربرانی را که می‌توانند پاسخ بهنگام دریافت کنند را محاسبه می‌کنیم. عدد حاصل در نمودار شکل ۹ برای Traffic Mobility مساوی ۴٪، به رنگ آبی رسم شده است. با افزایش درصد جابجایی کاربران و به ترتیب دو و سه برابر کردن آن نسبت به مقدار پایه، دو نمودار دیگر به ترتیب بارنگ‌های قرمز و سبز کشیده شده‌اند. همچنین جهت مقایسه با بهترین پاسخ (MILP Programming)، نمودارهای فوق به‌صورت نرمالایز شده نسبت به پاسخ فوق رسم شده‌اند.



شکل ۹: اثر زیاد یا کم کردن طول بازه زمانی تکرار الگوریتم

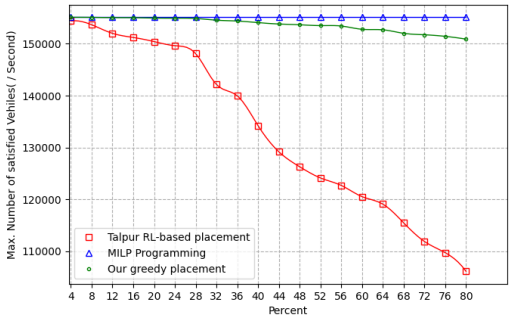
نتایج نشان می‌دهند که با افزایش جابجایی تعداد خودروها در هر بازه زمانی و با افزایش طول بازه‌ی زمانی تکرار الگوریتم، فاصله‌ی پاسخ روش پیشنهادی ما از بهترین پاسخ، افزایشی می‌گردد. دلیل این امر، دیر اجراشدن الگوریتم پیشنهادی است که باعث می‌شود مانتوانیم با سرعت کافی به تغییرات در شرایط مسئله واکنش نشان دهیم. همچنین دلیل کاهشی‌تر شدن پاسخ الگوریتم پیشنهادی با افزایش تعداد جابجایی کاربران به این خاطر است که ارتباط بین MECها که برای جابجایی درخواست‌ها استفاده می‌شود، بیشتر شده و توان بیشتری از سرورها صرف پوشش تأخیر انتشار شبکه می‌شود و به سبب آن تعداد کمتری از درخواست‌های خودروها را می‌توان پاسخ داد.

۵-۵- اثر تعداد خودروهایی که در هر بازه زمانی از مرز نقاط دسترسی عبور می‌کنند

شکل ۱۰، اثر تعداد خودروهایی که در هر ثانیه از مرز سرورهای MEC عبور می‌کنند را نمایش می‌دهد. در این آزمایش، ما با تغییر پارامتر Traffic Mobility، اثر آن را بر روی تعداد کاربرانی که می‌توانند پاسخ بهنگام دریافت کنند، بررسی می‌کنیم.

در نمودار، محور افقی، مقادیر مختلف Traffic Mobility است و مابین بازه ۴ تا ۸۰ درصد تغییر می‌کند. محور عمودی تعداد پاسخ‌های بهنگام است. برای مقایسه،

در حالت‌های مختلف، با استفاده از مسئله برنامه‌ریزی خطی، تعداد کاربرانی را که می‌توانند پاسخ بهنگام دریافت کنند، محاسبه می‌کنیم. عدد حاصل در نمودار شکل ۱۰ به رنگ آبی دیده می‌شود. تعدادی را نیز که پاسخ [۱] برآورده می‌کند، در همان شکل به رنگ قرمز نمایش داده‌ایم. در مقایسه دیگر، با استفاده از الگوریتم پیشنهادی، تعداد کاربرانی را که می‌توانند پاسخ بهنگام دریافت کنند را محاسبه می‌کنیم. این تعداد در شکل ۱۰ به رنگ سبز نشان داده‌شده است.



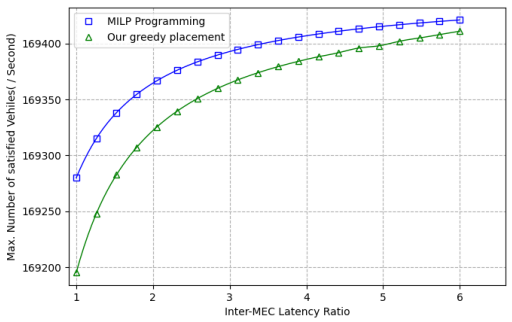
شکل ۱۰: اثر پارامتر Traffic Mobility

نتایج نشان می‌دهند که با افزایش جابجایی تعداد خودروها، فاصله‌ی پاسخ [۱] از بهترین پاسخ، افزایشی می‌گردد. دلیل این امر، همان افزایش نمایی زمان حل مسئله بهینه‌سازی است که باعث می‌شود [۱] نتواند با سرعت کافی به تغییرات در شرایط مسئله واکنش نشان دهد. برعکس، فاصله پاسخ الگوریتم پیشنهادی از بهترین پاسخ کم است. دلیل آن به خاطر اجرای کوتاه‌مدت الگوریتم پیشنهادی در مقایسه بازمان حل مسئله بهینه‌سازی است که در اینجا الگوریتم فوق سریع‌تر به تغییرات در شرایط مسئله واکنش نشان می‌دهد.

۵-۶- اثر تأخیر کمتر (یا پهنای باند بیشتر) بین سرورها

شکل ۱۱، اثر تأخیر کمتر بین سرورهای لبه شبکه را نمایش می‌دهد. در این آزمایش ما پارامتر " نسبت مقدار تأخیر انتشار به تنظیمات پایه (I_MEC_LR) " را تعریف می‌کنیم که برای اعمال حساسیت نسبت به تنظیمات پایه استفاده می‌شود.

در این آزمایش، با تقسیم پارامتر پایه‌ی "تأخیر انتشار بین هر دو جفت سرور MEC " بر عدد I_MEC_LR، می‌توان اثر تأخیر کمتر بین سرورهای لبه شبکه را بررسی کرد. مقدار این پارامتر پایه در آزمایش، در جدول ۴ آمده است. در نمودار، محور افقی، مقدار I_MEC_LR است و محور عمودی تعداد پاسخ‌های بهنگام است. برای مقایسه، در حالت‌های مختلف، با افزایش مقدار I_MEC_LR در بازه ۱ تا ۶، با استفاده از مسئله برنامه‌ریزی خطی و الگوریتم پیشنهادی، تعداد کاربرانی را که می‌توانند پاسخ بهنگام دریافت کنند، محاسبه می‌کنیم. اعداد حاصل در نمودار شکل ۱۱ نشان داده‌شده است.



شکل ۱۱: اثر تأخیر کمتر بین سرورهای لبه موبایل

طبق نتایج پژوهش مدل پیشنهادی با کاهش تأخیر بین سرورهای لبه شبکه، پاسخ خود را به پاسخ بهینه نزدیک‌تر می‌کند. دلیل نزدیک‌تر شدن این دو پاسخ به این خاطر است که، چون درروش پیشنهادی درخواست‌ها از سمت نقاط دسترسی جداجدا، یکی پس از دیگری موردبررسی قرار می‌گیرند، ممکن است بعضی از درخواست‌ها، برخلاف بهترین پاسخ، به سرورهای دورتری بروند و این کار باعث می‌شود اتلاف توان سرورها نسبت به بهترین پاسخ بیشتر گردد. حال هر چه این تأخیر انتشار کمتر باشد، توان کمتری از سرورها صرف پوشش تأخیر درروش پیشنهادی می‌شود و به سبب آن این دو پاسخ به هم نزدیک‌تر می‌شوند.

همچنین همان‌طور که مشاهده می‌کنیم، با کاهش تأخیر بین سرورهای لبه شبکه، تعداد پاسخ‌های بهنگام افزایش می‌یابد. دلیل این افزایش به این خاطر است که با کاهش تأخیر بین سرورهای لبه موبایلی، توان کمتری از سرورها صرف پوشش تأخیر شود و به سبب آن تعداد بیشتری از درخواست‌های خودروها را می‌توان پاسخ داد.

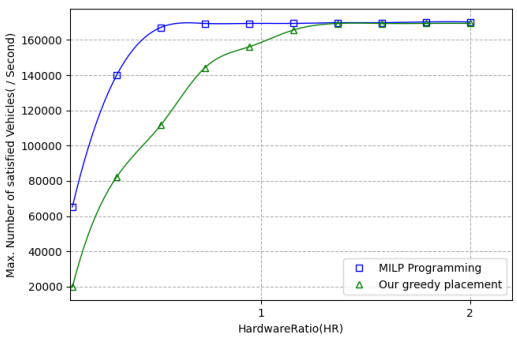
۵-۷- اثر تعداد خودروهایی که در هر بازه زمانی از مرز نقاط دسترسی عبور می‌کنند

شکل ۱۲، اثر ناشی از تغییر قدرت پردازشی سرورهای لبه شبکه بین مدل پیشنهادی و پاسخ بهینه را نمایش می‌دهد. در این آزمایش ما پارامتر " نسبت ظرفیت سخت‌افزار به تنظیمات پایه (HR) " را تعریف می‌کنیم که برای اعمال حساسیت نسبت به تنظیمات پایه استفاده می‌شود.

در این آزمایش، با ضرب HR در پارامتر پایه‌ی Available resources at MEC می‌توان اثر تغییر قدرت پردازشی سرورهای لبه شبکه را بررسی کرد. در نمودار، محور افقی مقدار HR است و محور عمودی تعداد پاسخ‌های بهنگام است. برای مقایسه، در حالت‌های مختلف، با افزایش مقدار HR در بازه ۰.۰۱ تا ۲، ما به‌صورت برون‌خط، با استفاده از مسئله برنامه‌ریزی و سپس با استفاده از الگوریتم پیشنهادی، تعداد کاربرانی را که می‌توانند پاسخ بهنگام دریافت کنند، محاسبه می‌کنیم. اعداد حاصل در نمودار شکل ۱۲ نشان داده‌شده‌اند.

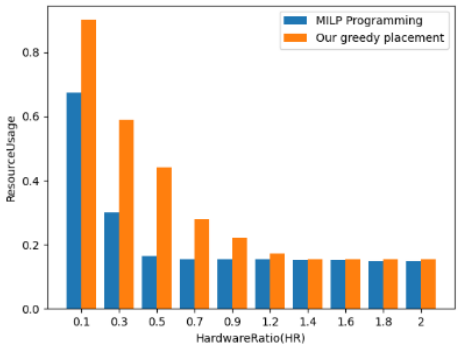
در این شکل، با کاهش HR (از راست به چپ)تعداد پاسخ‌های بهنگام کاهش می‌یابد. دلیل این امر به این خاطر است که، درخواست‌ها از جانب خودروها، نیاز به سخت‌افزار پردازشی دارند و با توجه به کاهش مقدار سخت‌افزار و کم شدن قدرت پردازشی سرورها، باعث ازدحام در سرورهای MEC می‌شوند و برخی از درخواست‌ها به MECهای راه دور می‌روند (اینکه سرور محلی دریافت‌کننده درخواست جایی برای سرویس دادن ندارد و درخواست را به سرور MEC دیگری می‌فرستد). ارتباط بین MECها که برای جابجایی درخواست‌ها استفاده می‌شود، تأخیر انتشار شبکه را ایجاد می‌کند و زمان باقی‌مانده برای صف‌بندی و پردازش را کمتر می‌کند؛ بنابراین، ازدحام فوق باعث کاهش پاسخ‌های بهنگام می‌گردد.

دلیل دورتر شدن این دو پاسخ از همدیگر زمانی که ظرفیت سخت‌افزاری کاهش می‌یابد به این خاطر است که، چون درروش پیشنهادی درخواست‌ها از سمت نقاط دسترسی جداجدا، یکی پس از دیگری موردبررسی قرار می‌گیرند، ممکن است بعضی از درخواست‌ها، برخلاف بهترین پاسخ، به سرورهای دورتری بروند و این کار باعث می‌شود اتلاف توان سرورها نسبت به بهترین پاسخ بیشتر گردد؛ بنابراین، توان بیشتری از سرورها صرف پوشش تأخیر درروش پیشنهادی می‌شود و به سبب آن این دو پاسخ از هم دورتر می‌شوند.



شکل ۱۲: اثر تغییر قدرت پردازشی سرورهای لبه شبکه

برای نشان دادن مسئله ازدحام، می‌توان یک نمودار دیگر رسم کرد که در آن با مورداستفاده قرار می‌گیرد تا جایی که تمام ظرفیت استفاده می‌شود (اشباع) و تعداد پاسخ‌های بهنگام ثابت می‌ماند. برای انجام این کار آزمایش فوق را به این صورت تکرار می‌کنیم که یک معیار جدید به نام ResourceUsage تعریف می‌کنیم که این مقدار برابر است با مجموع منابع سخت‌افزاری استفاده‌شده جهت سرویس به درخواست‌ها، تقسیم‌بر کل تعداد منابع. اگر این مقدار به یک نزدیک شود نشانه آن است که بخش بیشتری از کل ظرفیت سخت‌افزار مورداستفاده قرارگرفته است. درصورتی‌که یک شود یعنی کل منابع استفاده‌شده و از آن به بعد دیگر تعداد پاسخ‌های بهنگام با افزایش تعداد درخواست‌ها تغییر نمی‌کند. نزدیک‌تر شدن به صفر نیز می‌تواند نشان از استفاده بیشتر منابع محلی (در همان سرور محل درخواست سرویس) جهت ارائه سرویس باشد. حال برای آزمایش فوق این نمودار در شکل ۱۳ نشان داده‌شده است.



شکل ۱۳: اثر تغییر قدرت پردازشی سرورهای لبه شبکه بر روی مقدار ResourceUsage

۵-۸- اثر ناشی ناهمگن بودن درخواست‌ها در نقاط دسترسی

در سناریوی آزمایشی که برای بررسی درخواست‌های ناهمگن است، درخواست‌های هر منطقه مربوط به هر سرور MEC، به‌صورت تصادفی و با توزیع یکنواخت $U(1000, 9000)$ ایجادشده است. عدد به‌دست‌آمده تعداد درخواست‌های آن ناحیه را نشان می‌دهد و به‌عنوان تنظیمات پایه برای این آزمایش در نظر گرفته می‌شود. تعداد درخواست‌ها در جدول ۵ نشان داده‌شده است.

شکل ۱۴، اثر ناشی از زیادشدن تعداد درخواست‌های ناهمگن در سرورهای لبه شبکه را نمایش می‌دهد. در این آزمایش ما پارامتر " نسبت تعداد درخواست‌های

جدول ۵: تعداد درخواست‌های ناهمگن

(به‌صورت تصادفی با توزیع یکنواخت بین ۱۰۰۰ تا ۹۰۰۰)

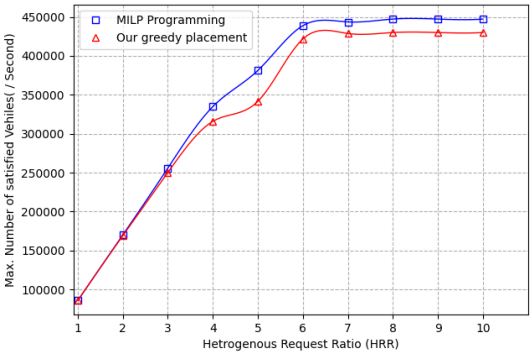
منطقه تحت کنترل سرور MEC	تعداد درخواست‌ها
۱	٤٠٢٠
۲	١٠٤٢
۳	٥١٠٣
٤	٦١٦١
٥	٧٩٢٦
٦	٢٥٦٥
۷	٢٦٣٠
۸	١٨٩٠
۹	٤٤٥١
۱۰	٣٣١٥
۱۱	٦٧٧٨
۱۲	٤٠٦٩
۱۳	٧٤٦٠
١٤	٣٤٠٦
١٥	٢١١٦
١٦	٣٦١١
١٧	١٧٤٣
١٨	١٥٨٥
١٩	٣٤٠٠
٢٠	٥٦٣٧

کاهش ظرفیت سخت‌افزاری سرورها، بخش بیشتری از کل ظرفیت سخت‌افزاری

ناهمگن به تنظیمات پایه(HRR) " را تعریف می‌کنیم که برای اعمال حساسیت نسبت به تنظیمات پایه استفاده می‌شود.

در این آزمایش، با ضرب پارامتر HRR در جدول ۵، می‌توان اثر زیادشدن تعداد درخواست‌های ناهمگن در سرورهای لبه شبکه را بررسی کرد. در نمودار، محور افقی مقدار HRR است و محور عمودی تعداد پاسخ‌های بهنگام است. برای مقایسه، در حالت‌های مختلف، با افزایش مقدار HRR در بازه ۱ تا ۶، ابتدا، ما به‌صورت برون‌خط، با استفاده از مسئله برنامه‌ریزی خطی و سپس با استفاده از الگوریتم پیشنهادی، تعداد کاربرانی را که می‌توانند پاسخ بهنگام دریافت کنند را محاسبه می‌کنیم. این تعداد در شکل ۱۴ نشان داده‌شده‌اند.

در این شکل، با افزایش HRR ابتدا تعداد پاسخ‌های بهنگام افزایش می‌یابد ولی سپس به یک نقطه اشباع می‌رسد؛ زیرا زمانی که درخواست سرویس از جانب خودروها افزایش می‌یابد، قاعدتاً افزایش پاسخ‌های بهنگام را به همراه دارد؛ اما با این افزایش، مقدار پردازش بیشتری در سرورها استفاده می‌شود. با توجه به بالا رفتن مقدار پردازش، ممکن است باعث ازدحام در سرورهای MEC شوند و برخی از درخواست‌ها به MECهای راه دور بروند (اینکه سرور محلی دریافت‌کننده درخواست جایی برای سرویس دادن ندارد و درخواست را به سرور MEC دیگری می‌فرستد). ارتباط بین MECها که برای جابجایی درخواست‌ها استفاده می‌شود، تأخیر انتشار شبکه را ایجاد می‌کند و زمان باقی‌مانده برای صف‌بندی و پردازش را کمتر می‌کند؛ بنابراین، ازدحام فوق باعث می‌گردد بعد از مدتی تعداد پاسخ‌های بهنگام به یک عدد معین برسند و دیگر افزایش نیابد چراکه پردازشی برای سرورها دیگر باقی نمانده است. به عبارتی سیستم Infeasible می‌شود.



شکل ۱۴: اثر زیادشدن تعداد درخواست‌های ناهمگن در سرورهای لبه شبکه

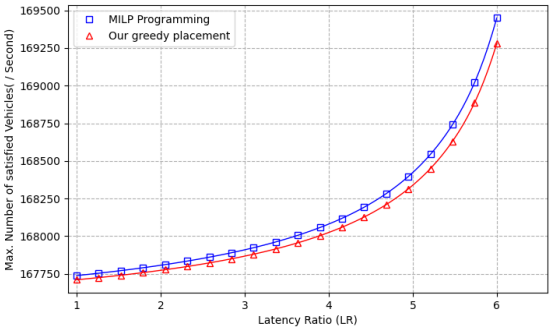
۵-۹- اثر ناشی از زیادشدن آستانه تأخیر سرویس‌ها

شکل ۱۵، اثر ناشی از زیادشدن آستانه تأخیر سرویس‌ها را نمایش می‌دهد. در این آزمایش ما پارامتر " نسبت آستانه تأخیر سرویس به تنظیمات پایه (LR) " را تعریف می‌کنیم که برای اعمال حساسیت نسبت به تنظیمات پایه استفاده می‌شود.

در این آزمایش، با ضرب LR در پارامتر پایه‌ی Service Latency Threshold می‌توان اثر زیادشدن آستانه تأخیر سرویس‌ها را بررسی کرد. در نمودار، محور افقی مقدار LR است و محور عمودی تعداد پاسخ‌های بهنگام است. برای مقایسه، در حالت‌های مختلف، با افزایش مقدار LR در بازه ۱ تا ۶، ابتدا با استفاده از مسئله برنامه‌ریزی خطی و سپس با استفاده از الگوریتم پیشنهادی تعداد کاربرانی را که می‌توانند پاسخ بهنگام دریافت کنند را محاسبه می‌کنیم. این تعداد در شکل ۱۵ نشان داده‌شده‌اند.

در این شکل، با افزایش LR تعداد پاسخ‌های بهنگام افزایش می‌یابد؛ زیرا زمانی که سرویس‌ها آستانه تأخیر بیشتری دارند، این امکان وجود دارد که درخواست‌هایی

که تاکنون به خاطر محدودیت آستانه تأخیر (نامعادله ۶) نمی‌توانستند به سرورهای MEC دورتری بروند، اکنون می‌توانند در آن‌ها سرویس بگیرند.



شکل ۱۵: اثر زیادشدن آستانه تأخیر سرویس‌ها

۵-۱۰- تحلیل و بحث

همان‌طور که قبلاً نیز بحث شد، حل مسئله بهینه‌سازی به‌صورت دوره‌ای ممکن است امکان‌پذیر نباشد، به‌خصوص برای شبکه‌های بزرگ. در اینجا می‌توان الگوریتم حریصانه پیشنهادی را به‌صورت دوره‌ای به‌عنوان رویکرد جایگزین استفاده نمود.

طول بازه زمانی تکرار الگوریتم به‌عنوان یک پارامتر مهم، باید به‌اندازه کافی نیز بزرگ باشد تا الگوریتم پیشنهادی بتواند تخصیص سرویس‌ها را در طول این فاصله به پایان برساند.

بااین‌وجود، اجرای سریع الگوریتم حریصانه نیز ممکن است کارآمد نباشد، زیرا همیشه خوب نیست که تخصیص سرویس‌ها را در پاسخ به اوج و یا کاهش ترافیک کوتاه‌مدت و یا درزمانی که نرخ ترافیک به‌سرعت در نوسان است انجام داد.

یکی از راه‌های رسیدگی به مشکل بالا این است که مقدار بیشتری را برای طول بازه زمانی تکرار الگوریتم انتخاب کنیم و میانگین ترافیک در آن مدت‌ها را به‌عنوان ترافیک ورودی به الگوریتم در نظر بگیریم. روش دیگر استفاده از الگوریتم‌های یادگیری آنلاین برای پیش‌بینی ترافیک است. اگر ترافیک از قبل پیش‌بینی‌شده باشد، اوج‌ها و افت‌ها را می‌توان پیش‌بینی کرد و تصمیم‌گیری برای اجرای الگوریتم‌های حریصانه را می‌توان هوشمندانه‌تر گرفت.

۶- خلاصه و نتیجه‌گیری

یکی از چالش‌های محاسبات لبه شبکه خودرویی در شهرهای بزرگ، اجرای بی درنگ نگاشت پویای سرویس‌های درخواستی به منابع محاسباتی لبه شبکه است. اگر نگاشت پویا و بی‌درنگ به‌خوبی انجام نشود، تعداد زیادی از درخواست‌ها ممکن است پاسخ بهنگام دریافت نکنند. در طول عملکرد سیستم، لازم است در زمان‌های مختلف برحسب جابجایی خودروها و تغییر در تعداد درخواست‌ها و توزیع جغرافیایی آن‌ها، نگاشت‌ها، دوباره بروز رسانی شوند. در کارهای قبلی با افزایش خودروها و تعداد سرورهای لبه شبکه، طول بازه‌های زمانی بروز رسانی بزرگ می‌شود، چون از راهکار برنامه‌ریزی خطی برای یادگیری مجدد استفاده می‌شد که با افزایش خودروها و تعداد سرورها زمان اجرای آن به‌صورت نمایی رشد می‌کرد. ما الگوریتم حریصانه‌ای را پیشنهاد کردیم که می‌تواند راه‌حل تقریبی برای مسئله برنامه‌ریزی خطی باشد. درروش برنامه‌ریزی خطی کلیه درخواست‌ها برای تمام سرویس‌ها از کلیه نقاط دسترسی و همه سرورها همگی باهم بررسی‌شده و سعی می‌شود تابع هدف بهینه شود. ولی در این روش حریصانه، ما ضمن استفاده از همان تابع هدف روش برنامه‌ریزی خطی، روش بهینه‌سازی را طوری اصلاح کردیم که درخواست‌ها از سمت نقاط دسترسی یکی پس از دیگری موردبررسی قرار می‌گیرند، به‌طوری‌که به ازای هر نقطه دسترسی، تابع هدف مربوطه را به حداقل برساند و سپس نقاط دسترسی بعدی بررسی می‌شوند. کارایی مدل پیشنهادی، ازنظر دستیابی به اهداف کاهش زمان اجرا و تعداد پاسخ‌های بهنگام به درخواست‌های دریافتی در مقایسه با روش مقاله پایه مورد ارزیابی قرار گرفت. تحلیل و ارزیابی نمودار حاصل از نتایج شبیه‌سازی نشان می‌دهد که مدل پیشنهادی در پارامترهای بیان‌شده در تعداد پاسخ‌های بهنگام ۲۶.۸٪ و در زمان اجرا ۹۰٪ بهبود نسبت به [1] داشته است و تفاوت آن با بیشینه پاسخ‌های بهنگام، تنها 6.8%است. به‌این‌ترتیب توانسته‌ایم که برای تعداد بیشتری از کاربران، سرویس‌های هوشمند را ارائه کنیم .

افزودن تاریخچه‌ی زمانی و مکانی حرکت‌های قبلی خودروها در تصمیم‌گیری برای نگاشت سرویس‌ها و استفاده از الگوریتم‌های یادگیری برای پیش‌بینی ترافیک و تصمیم‌گیری برای اجرای الگوریتم حریصانه به‌عنوان کار آینده در نظر گرفته‌شده است.

۷- مآخذ

[1] A. Talpur and M. Gurusamy, “Reinforcement learning-based dynamic service placement in vehicular networks,” in IEEE Internet of Things Journal, Volume: 9, Issue: 8, 15 April 2022

[2] N. B. Truong, G. M. Lee and Y. Ghamri-Doudane, "Software defined networking-based vehicular Adhoc Network with Fog Computing," 2015 IFIP/IEEE International Symposium on Integrated Network Management (IM), Ottawa, ON, 2015, pp. 1202-1207.

[3] A. Kaul, K. Obraczka, M. A. S. Santos, C. E. Rothenberg, and T. Turletti, “Dynamically distributed network control for message dissemination in ITS,” in IEEE/ACM 21st International Symposium on DSRT, 2017, pp. 1–9.

[4] A. Kaul, L. Xue, K. Obraczka, M. A. S. Santos, and T. Turletti, “Handover and Load Balancing for Distributed Network Control: Applications in ITS Message Dissemination,” in 27th ICCCN, 2018, pp. 1–8.

[5] Zhang, S. Gjessing, W. Xia, and K. Yang, “Toward cloud based vehicular networks with efficient resource management,” IEEE Netw. vol. 27, no. 5, pp. 48–55, Sep./Oct. 2013

[6] S. D. A. Shah, M. A. Gregory, S. Li, R. Fontes, and L. Hou, “SDNBased Service Mobility Management in MEC-Enabled 5G and Beyond Vehicular Networks,” IEEE Internet of Things Journal, 2022.

[7] Hadi Tabatabaee Malazi, Saqib Rasool Chaudhry, Aqeel Kazmi, Andrei Palade, Christian Cabrera, Gary White, and Siobhán Clarke. Dynamic service placement in multi-access edge computing: A systematic literature review. IEEE Access 10 (2022), 32639–32688.

[8] A. Moubayed, A. Shami, P. Heidari, A. Larabi, and R. Brunner, “Edgeenabled V2X service placement for intelligent transportation systems,” IEEE Trans. Mobile Comput. vol. 20, no. 4, pp. 1380–1392, Apr. 2021.

[9] Wei Wang, Massimo Tornatore, Yongli Zhao, Haoran Chen, Yajie Li, Abhishek Gupta, Jie Zhang, and Biswanath Mukherjee, “Infrastructure-efficient Virtual-Machine Placement and Workload Assignment in Cooperative Edge-Cloud Computing over Backhaul Networks,” in IEEE Transactions on Cloud Computing, August 2021, pp. 1–6.

[10] Shaer I, Haque A, Shami A: Multi-Component V2X Applications Placement in Edge Computing Environment. ICC 2020-2020 IEEE International Conference on Communications (ICC). 2020; (pp. 1–6). IEEE.

[11] Leilei Wang, Xiaoheng Deng, Jinsong Gui, Xuechen Chen, and Shaohua Wan. 2023. Microservice-Oriented Service Placement for Mobile Edge Computing in Sustainable Internet of Vehicles. IEEE Transactions on Intelligent Transportation Systems. September 2023.

[12] Wencan Mao, Ozgur Umut Akgul, Byungjin Cho, Yu Xiao, and Antti Ylä-Jääski. On-demand Vehicular Fog Computing for Beyond 5G Networks. Accepted for publication in IEEE Transactions on Vehicular Technology, 1-17 pages, June 2023.

[13] Cohen, I. Chiasserini, C.F. Giaccone, P. Scalosub, G. Dynamic service provisioning in the edge-cloud continuum with bounded resources. IEEE ACM Trans. Netw. 2023, 1–16.

[14] A. Yousefpour et al. "FOGPLAN: A Lightweight QoS-Aware Dynamic Fog Service Provisioning Framework," in IEEE Internet of Things Journal, vol. 6, no. 3, pp. 5080-5096, June 2019, doi: 10.1109/JIOT.2019.2896311

[15] C. H. C. Jojoa, S. Svorobej, A. Palade, A. Kazmi, and S. Clarke, ‘‘MAACO: A dynamic service placement model for smart cities,’’ IEEE Trans. Services Comput. early access, Jan. 13, 2022.

[16] "5G White Paper", [online] Available: www.ngmn.org / uploads /media / NGMN_5G_White_Paper_V1_0.pdf

[17] Wondernetwork, [online] Available: https:// wondernetwork.com /pings

[18] Mitchell Stuart, Consulting Stuart Mitchell, OâĂŹSullivan Michael, Dunning Iain. PuLP: A Linear Programming Toolkit for Python.: Department of Engineering Science, The University of Auckland, Auckland, New Zealand; 2011

۸- معرفی نویسندگان:

دکتر مازیار گودرزی دانشیار دانشکده مهندسی کامپیوتر، دانشگاه صنعتی شریف است. وی مدرک‌های کارشناسی، کارشناسی ارشد و دکتری را نیز از همان دانشگاه و دانشکده دریافت کرده است و درزمینه معماری کامپیوتر، هم طراحی سخت‌افزار/نرم‌افزار و رایانش سبز پژوهش می‌نماید.

سید علیرضا عمرانیان دانشجوی دکترا دانشکده مهندسی کامپیوتر، دانشگاه صنعتی شریف است. حوزه‌های علاقه وی شامل محاسبات لبه موبایل، محاسبات ابری و مرکز داده توزیع‌شده است.

MEC Numbers	۱	۲	۳	۴	[illegible]	[illegible]	[illegible]	۸	۹	۱۰	۱۱	۱۲	۱۳	۱۴	۱۵	۱۶	۱۷	۱۸	۱۹	۲۰
۱	۰	۷٫۸۵۵	۱۶٫۷۶۹	۸٫۷۲۱	[illegible]	[illegible]	[illegible]	۲۳٫۲۲۸	۳۵٫۰۵	۲۲٫۳۸	۳۴٫۰۸	۴۵٫۶۲	۳۸٫۷۳	۲۲٫۶۳	۴۴٫۷۹	۳۶٫۵۳	۲۷٫۵۵	۳۹٫۸۴	۵۷٫۶۶	۳۹٫۶۹
۲	۷٫۸۵۵	۰	۹٫۰۹۸	۰٫۹۰۵	[illegible]	[illegible]	[illegible]	۲۲٫۸۶۷	۲۵٫۹۸	۲۶٫۶۴	۱۸٫۸	۱۸٫۳۱	۳۶٫۰۱	۱۱٫۲۹	۲۷٫۴۴	۳۲٫۱۹	۳۴٫۲	۴۰٫۶۷	۴۳٫۰۱	۴۲٫۰۵
۳	۱۶٫۷۶۹	۹٫۰۹۸	۰	۹٫۸۸۸	[illegible]	[illegible]	[illegible]	۲۱٫۸۵۳	۳۰٫۰۴	۲۴٫۳۳	۳۶٫۴۴	۴۴٫۲۹	۲۵٫۲۷	۲۶٫۷۶	۴۷٫۳۶	۳۸٫۱۳	۲۹٫۵۳	۳۶٫۶۵	۲۷٫۴۴	۴۲٫۰۹
۴	۸٫۷۲۱	۰٫۹۰۵	۹٫۸۸۸	۰	[illegible]	[illegible]	[illegible]	۲۳٫۸۹۸	۲۵٫۵	۲۷٫۵۹	۲٫۳۵	۲۳٫۲۲	۴۰٫۴۲	۱۴٫۲۵	۳۱٫۵۸	۱۶٫۸۵	۳۳٫۷۲	۴۰٫۹۱	۴۴٫۶۴	۳۶٫۶۳
۵	۱۱٫۳۲	۴٫۳۲۱	۱۹٫۵۵۷	۴۶۶۱	[illegible]	[illegible]	[illegible]	۲۲٫۰۴۸	۳۲٫۵۵	۳۴٫۷	۳۵٫۴۱	۳۱٫۴	۳۳٫۹۸	۱۸٫۴۳	۴۳٫۸	۳۲٫۰۲	۴۶٫۷۱	۴۷٫۹۸	۳۳٫۷۷	۴۲٫۹
۶	۱۹٫۴۵	۹٫۱۰۸	۱۷٫۹۶۱	۱۴۶۰۴	[illegible]	[illegible]	[illegible]	۱۸٫۶۲۴	۲۵٫۹۸	۲۶٫۶۳	۱۸٫۷۸	۱۸٫۳۳	۴۱٫۸۶	۱۱٫۱۴	۲۷٫۴۵	۳۲٫۰۵	۳۴٫۱	۴۰٫۶۸	۴۳٫۰۴	۴۲٫۰۴
۷	۱۹٫۰۴۸	۹٫۷۷۵	۱۸٫۱۶	۹٫۵۰۵	[illegible]	[illegible]	[illegible]	۱۹٫۱۷	۲۵٫۳۱	۲۷٫۵۳	۲۰٫۱۱	۲۳٫۲۲	۴۴٫۸۶	۱۴٫۳۷	۲۸٫۹۹	۲۱٫۴۱	۳۳٫۸۴	۴۵٫۶۹	۴۷٫۳۰	۳۷٫۴۹
۸	۲۳٫۲۲۸	۲۲٫۸۶۷	۲۱٫۸۵۳	۲۳٫۸۹۸	[illegible]	[illegible]	[illegible]	۰	۲۷٫۴۶	۲۹٫۲۳	۶٫۵۷	۱۶٫۲۸	۴۸٫۸۹	۳۳٫۷۳	۱۴٫۲۴	۱۸٫۴۹	۴۱٫۱۷	۴۲٫۶۹	۴۷٫۹۸	۳۸٫۹۸
۹	۳۵٫۰۵	۲۵٫۹۸	۳۰٫۰۴	۲۵٫۵	[illegible]	[illegible]	[illegible]	۲۷٫۴۶	۰	۲٫۸۴	۳۱٫۹۲	۵۲٫۳۵	۲۳٫۷۵	۳۸٫۷۶	۴۰٫۴۱	۴۴٫۷۶	۱۰٫۴۲	۱۶٫۵۹	۲۴٫۱۳	۱۲٫۶۶
۱۰	۲۲٫۳۸	۲۶٫۶۴	۲۴٫۳۳	۲۷٫۵۹	[illegible]	[illegible]	[illegible]	۲۹٫۲۳	۲٫۸۴	۰	۳۲٫۸۸	۳۴٫۳۳	۲۱٫۵۵	۳۶٫۹۱	۲۷٫۷۸	۱۴٫۳۷	۱۱٫۵۷	۱۸٫۷۵	۲۰٫۵۷	۱۶٫۷۳
۱۱	۳۴٫۰۸	۱۸٫۸	۳۶٫۴۴	۲٫۳۵	[illegible]	[illegible]	[illegible]	۶٫۵۷	۳۱٫۹۲	۳۲٫۸۸	۰	۱۲٫۷۲	۶۳٫۰۳	۲۷٫۸۷	۱۱٫۴۸	۲۱٫۶۹	۴۰٫۰۲	۴۶٫۲۱	۵۱٫۵۳	۴۲٫۲۹
۱۲	۴۵٫۶۲	۱۸٫۳۱	۴۴٫۲۹	۲۳٫۲۲	[illegible]	[illegible]	[illegible]	۱۶٫۲۸	۵۲٫۳۵	۳۴٫۳۳	۱۲٫۷۲	۰	۴۷٫۷۷	۳۵٫۰۷	۲۳٫۰۳	۴۴٫۹۲	۴۱٫۵۸	۶۲٫۰۳	۵۰٫۶۹	۵۰٫۳۵
۱۳	۳۸٫۷۳	۳۶٫۰۱	۲۵٫۲۷	۴۰٫۴۲	۳۳٫۹۸	۴۱٫۸۶	۴۴٫۸۶	۴۸٫۸۹	۲۳٫۷۵	۲۱٫۵۵	۶۳٫۰۳	۴۷٫۷۷	۰	۵۰٫۷۰	۵۴٫۱۶	۲۸٫۰۳	۲۶٫۸۶	۱۷٫۴۸	۷٫۸۷	۱۶٫۹۶
۱۴	۲۲٫۶۳	۱۱٫۲۹	۲۶٫۷۶	۱۴٫۲۵	۱۸٫۴۳	۱۱٫۱۴	۱۴٫۳۷	۳۳٫۷۳	۳۸٫۷۶	۳۶٫۹۱	۲۷٫۸۷	۳۵٫۰۷	۵۰٫۷۰	۰	۴۰٫۶۰	۲۵٫۹۴	۴۴٫۳۱	۶۱٫۱۷	۵۴٫۱۱	۵۴٫۳۳
۱۵	۴۴٫۷۹	۲۷٫۴۴	۴۷٫۳۶	۳۱٫۵۸	۴۳٫۸	۲۷٫۴۵	۲۸٫۹۹	۱۴٫۲۴	۴۰٫۴۱	۲۷٫۷۸	۱۱٫۴۸	۲۳٫۰۳	۵۴٫۱۶	۴۰٫۶۰	۰	۴۶٫۴۹	۴۱٫۴۰	۵۴٫۱۷	۶۵٫۷۸	۵۱٫۴۹
۱۶	۳۶٫۵۳	۳۲٫۱۹	۳۸٫۱۳	۱۶٫۸۵	۳۲٫۰۲	۳۲٫۰۵	۲۱٫۴۱	۱۸٫۴۹	۴۴٫۷۶	۱۴٫۳۷	۲۱٫۶۹	۴۴٫۹۲	۲۸٫۰۳	۲۵٫۹۴	۴۶٫۴۹	۰	۲۱٫۱۸	۲۹٫۳۷	۳۲٫۰۴	۲۷٫۱۱
۱۷	۲۷٫۵۵	۳۴٫۲	۲۹٫۵۳	۳۳٫۷۲	۴۶٫۷۱	۳۴٫۱	۳۳٫۸۴	۴۱٫۱۷	۱۰٫۴۲	۱۱٫۵۷	۴۰٫۰۲	۴۱٫۵۸	۲۶٫۸۶	۴۴٫۳۱	۴۱٫۴۰	۲۱٫۱۸	۰	۳۲٫۶۹	۲۷٫۸۸	۲۶٫۷۵
۱۸	۳۹٫۸۴	۴۰٫۶۷	۳۶٫۶۵	۴۰٫۹۱	۴۷٫۹۸	۴۰٫۶۸	۴۵٫۶۹	۴۲٫۶۹	۱۶٫۵۹	۱۸٫۷۵	۴۶٫۲۱	۶۲٫۰۳	۱۷٫۴۸	۶۱٫۱۷	۵۴٫۱۷	۲۹٫۳۷	۳۲٫۶۹	۰	۲۶٫۷۰	۱۱٫۹۵
۱۹	۵۷٫۶۶	۴۳٫۰۱	۲۷٫۴۴	۴۴٫۶۴	۳۳٫۷۷	۴۳٫۰۴	۴۷٫۳۰	۴۷٫۹۸	۲۴٫۱۳	۲۰٫۵۷	۵۱٫۵۳	۵۰٫۶۹	۷٫۸۷	۵۴٫۱۱	۶۵٫۷۸	۳۲٫۰۴	۲۷٫۸۸	۲۶٫۷۰	۰	۱۹٫۲۱
۲۰	۳۹٫۶۹	۴۲٫۰۵	۴۲٫۰۹	۳۶٫۶۳	۴۲٫۹	۴۲٫۰۴	۳۷٫۴۹	۳۸٫۹۸	۱۲٫۶۶	۱۶٫۷۳	۴۲٫۲۹	۵۰٫۳۵	۱۶٫۹۶	۵۴٫۳۳	۵۱٫۴۹	۲۷٫۱۱	۲۶٫۷۵	۱۱٫۹۵	۱۹٫۲۱	۰

[1] Vehicular Edge Computing Network
[2] Vehicle Adhoc Network
[3] Mobile Edge Computing
[4] Vehicle to Everything Service
[5] Microservice-Oriented Service Placement
[6] Quality of Service
[7] Normal