به نام خدا

درس سیستم‌های نهفته

مستندات پروژه

نگار باباشاه

ایمان محمدی

محمدمهدی میرزایی

بهار ۱۴۰۳

تخصیص وظایف در معماری ابر، مه، لبه با در نظر گرفتن پدیدهٔ مهاجرت (پروژه ۱۷)

مقدمه

امروزه، با پیشرفت چشمگیر فناوری‌های ارتباطی و توسعه سریع اینترنت اشیا (IoT)، استفاده از دستگاه‌های متحرک در زندگی روزمره و صنعت به شکل گسترده‌ای افزایش یافته است. این دستگاه‌ها که شامل گوشی‌های هوشمند، خودروهای متصل، وسایل نقلیه هوشمند، و دیگر تجهیزات پرتابل هستند، نیازمند سیستم‌های پردازشی قوی و کارآمدی برای انجام وظایف محوله در زمان واقعی می‌باشند.

با این حال، محدودیت‌های مربوط به پردازش و ذخیره‌سازی داده‌ها در این دستگاه‌ها از چالش‌های اصلی است که به سمت استفاده از معماری‌های محاسباتی مدرن مانند محاسبات ابری، مه و لبه سوق داده است. در این میان، معماری سه لایه ابر، مه، و لبه به عنوان یک راهکار امیدوار کننده برای پشتیبانی از این دستگاه‌های متحرک و توانمندسازی آن‌ها با امکان پردازش داده‌ها نزدیک به منبع داده مطرح شده است.

در این معماری، لایه لبه که نزدیک‌ترین لایه به کاربران است، وظیفه جمع‌آوری داده‌ها و انجام پیش‌پردازش‌های اولیه را بر عهده دارد. لایه مه که بین لایه لبه و ابر قرار دارد، امکان پردازش‌های پیچیده‌تر و ذخیره‌سازی موقت داده‌ها را فراهم می‌کند، و در نهایت لایه ابر، داده‌ها را در مقیاس بزرگ پردازش و ذخیره می‌کند. این تقسیم‌بندی به ما اجازه می‌دهد تا داده‌ها و وظایف را به صورت مؤثرتری مدیریت کنیم، ولی با چالش‌هایی نیز همراه است.

یکی از این چالش‌ها، مسئله مهاجرت دستگاه‌های متحرک است که می‌تواند به دلیل تغییر موقعیت جغرافیایی یا نوسانات کیفیت شبکه رخ دهد. این پدیده می‌تواند بر تخصیص منابع و وظایف محاسباتی تأثیر بگذارد و نیاز به استراتژی‌های دقیق و بهینه برای مدیریت آن‌ها دارد.

پروژه حاضر به دنبال ارائه راهکارهایی برای تخصیص کارآمد وظایف در معماری سه لایه ابر – مه – لبه با در نظر گرفتن پدیده مهاجرت دستگاه‌های متحرک است. هدف از این پروژه ارائه راه‌حل‌هایی است که نه تنها به بهبود عملکرد و کارایی کمک کنند، بلکه پایداری و دسترس‌پذیری سیستم‌های مبتنی بر IoT را نیز افزایش دهند.

مفاهیم و تعریف‌ها

در این بخش، به تعریف و توضیح مفاهیم کلیدی مورد استفاده در پروژه می‌پردازیم که شامل معماری سه لایه ابر، مه و لبه، پدیده مهاجرت، و دیگر اصطلاحات فنی مرتبط می‌شود:

معماری سه لایه: این معماری یک ساختار توزیع‌شده است که به سه بخش اصلی تقسیم می‌شود:

- لایه ابر (Cloud Layer): این لایه شامل مراکز داده بزرگ است که در آنها داده‌ها به صورت متمرکز پردازش و ذخیره می‌شوند. این لایه برای انجام محاسبات پیچیده و تحلیل داده‌های حجیم مناسب است.
- لایه مه (Fog Layer): لایه مه یک لایه واسطه است که بین دستگاه‌های کاربر (لبه) و مراکز داده ابری (ابر) قرار گرفته است. این لایه شامل نودهای محاسباتی است که می‌توانند پردازش‌ها و ذخیره‌سازی‌های موقت را بر عهده بگیرند تا بار را از روی لایه ابر بکاهند و پاسخ‌دهی سریع‌تری را ارائه دهند.
- لایه لبه (Edge Layer): این لایه شامل دستگاه‌های متحرک یا ثابت نزدیک به کاربر است که داده‌ها را جمع‌آوری و پیش‌پردازش می‌کنند. لایه لبه برای کاهش تأخیر در پاسخ‌گویی به کاربران و کاهش بار شبکه طراحی شده است.

مهاجرت (Migration): در محیط‌های محاسباتی، مهاجرت به تغییر مکان دستگاه‌های متحرک از یک نقطه به نقطه دیگر اطلاق می‌شود که می‌تواند به دلایل مختلفی مانند تغییر در کیفیت شبکه، حرکت فیزیکی کاربران یا خرابی‌های سخت‌افزاری رخ دهد. مهاجرت می‌تواند چالش‌هایی را در زمینه تخصیص منابع و مدیریت وظایف ایجاد کند.

نود (Node): در معماری ما، نود به هر دستگاه یا منبع محاسباتی در هر یک از لایه‌های ابر، مه یا لبه اشاره دارد که می‌تواند وظایف محاسباتی را انجام دهد. نودها ممکن است شامل سرورها، روترها، سوئیچ‌ها یا حتی دستگاه‌های پایانی مانند گوشی‌های هوشمند یا خودروهای هوشمند باشند.

تخصیص وظایف (Task Assignment): فرایند توزیع وظایف محاسباتی میان نودهای مختلف در لایه‌های ابر، مه و لبه بر اساس معیارهایی مانند موقعیت مکانی، ظرفیت پردازشی و مصرف انرژی. این فرایند باید به گونه‌ای انجام شود که بهینه‌سازی مصرف انرژی، کاهش تاخیر و افزایش قابلیت اطمینان سیستم را به همراه داشته باشد.

پایداری و دسترس‌پذیری: در معماری سه لایه، پایداری به توانایی سیستم برای حفظ عملکرد مستمر در شرایط تغییر پذیر اشاره دارد، در حالی که دسترس‌پذیری به میزان در دسترس بودن منابع و خدمات برای کاربران نهایی می‌پردازد.

خروجی مورد انتظار

در این پروژه، هدف اصلی ارائه راه‌حل‌هایی برای تخصیص وظایف در دستگاه‌های متحرک است که با پدیده مهاجرت در معماری سه لایه ابر، مه و لبه مواجه هستند. ویژگی اصلی خروجی مدیریت مهاجرت‌ها و تخصیص صحیح تسک‌ها به نحوی است که هم ددلاین‌ها از دست نروند هم مهاجرت کمینه شود. به صورت دقیق‌تر، خروجی‌های مورد انتظار از این پروژه به شرح زیر می‌باشند:

۱. بهبود کارایی سیستم: انتظار می‌رود که الگوریتم‌های توسعه داده شده بتوانند کارایی سیستم را از طریق کاهش تاخیر در پاسخ‌دهی و بهینه‌سازی استفاده از منابع، به طور قابل توجهی افزایش دهد.

۲. کاهش تأثیر مهاجرت: با توجه به پدیده مهاجرت دستگاه‌های متحرک، یکی از اهداف این پروژه کاهش تأثیر منفی این مهاجرت‌ها بر تخصیص وظایف است. راه‌حل‌های پیشنهادی باید قادر به سازگاری سریع با تغییرات محیطی و شبکه‌ای باشند.

۳. افزایش پایداری و دسترس‌پذیری: پروژه به دنبال افزایش پایداری سیستم در برابر خطاها و اختلالات احتمالی و همچنین بهبود دسترس‌پذیری منابع برای کاربران نهایی است. این موضوع به خصوص در شرایط مهاجرت کاربران و تغییر دینامیکی توپولوژی شبکه اهمیت می‌یابد.

۴. بهینه‌سازی مصرف انرژی: یکی دیگر از ویژگی‌های کلیدی خروجی، کاهش مصرف انرژی در دستگاه‌های متحرک و نودهای مه است. الگوریتم باید سعی کند تا حد ممکن مصرف انرژی را مدیریت کند.

۵. مستندسازی و تحلیل دقیق: نتایج حاصل از اجرای الگوریتم‌ها و راه‌حل‌های پیاده‌سازی شده باید به صورت دقیق مستند و تحلیل شوند تا امکان مقایسه با دیگر روش‌ها و الگوریتم‌های موجود فراهم آید.

به طور خلاصه، این پروژه با هدف ایجاد یک محیط محاسباتی پایدار، کارآمد و کم مصرف، ضمن بهره‌گیری از فناوری‌های نوین و تکنیک‌های برنامه‌نویسی پیشرفته، پیش می‌رود تا چالش‌های اصلی موجود در معماری‌های محاسباتی مدرن را برطرف سازد.

مدل‌سازی سیستم

این پروژه از یک مدل سیستم پیچیده استفاده می‌کند که شامل اجزای مختلفی در معماری سه لایه ابر، مه و لبه می‌شود. در اینجا جزئیات کاملی از سیستم و اجزای آن ارائه می‌دهیم:

معماری کلی:

سیستم ما شامل سه لایه اصلی است: لایه ابر، لایه مه و لایه کاربران (لبه). هر لایه وظایف خاصی را بر عهده دارد و با استفاده از زبان برنامه‌نویسی پایتون و چارچوب‌های مرتبط، شبیه‌سازی شده است.

لایه ابر (Cloud Layer):

این لایه برای انجام محاسبات سنگین و ذخیره‌سازی داده‌ها در مقیاس بزرگ طراحی شده است. کلاس CloudLayer مسئول مدیریت نودهای ابری است و امکان افزودن نود به این لایه را فراهم می‌کند. این نودها از کلاس Node ایجاد می‌شوند که خصوصیاتی مانند شناسه، مکان، قدرت پردازشی و پوشش را دارند.

لایه مه (Fog Layer):

لایه مه نقش پل ارتباطی بین لایه ابر و دستگاه‌های لبه را دارد و در کلاس FogLayer پیاده‌سازی شده است. این لایه شامل دو نوع نود است: نودهای ثابت و نودهای سیار، که هر دو به وسیله کلاس Node مدیریت می‌شوند و می‌توانند در حین حرکت وظایف پردازشی را انجام دهند. نکته‌ی مهم در لایه‌ی مه همین قابلیت حرکت گره‌ها است که باید پیش‌بینی شود.

لایه کاربران (Users Layer):

لایه کاربران شامل دستگاه‌های متحرک کاربران است که داده‌ها را جمع‌آوری و اولین سطح پردازش‌ها را انجام می‌دهند. این لایه در کلاس UsersLayer تعریف شده و به طور مستقیم با گراف حرکت دستگاه‌ها، که در کلاس MobilityGraph مدل‌سازی شده است، ارتباط دارد.

گراف (Mobility Graph):

MobilityGraph تمام نودها و حرکت آن‌ها را در زمان واقعی مدیریت می‌کند. این کلاس با استفاده از داده‌های XML و پارسر SUMO (SumoXMLParser)، حرکت دستگاه‌ها و تغییرات مکانی نودها را شبیه‌سازی می‌کند.

مدیریت وظایف و تخصیص (Task Management and Assignment):

وظایف و تخصیص آن‌ها از طریق کلاس‌های ServiceZone و Topology مدیریت می‌شوند. Task خصوصیات مربوط به یک وظیفه مانند نام، نیاز به قدرت پردازشی، حجم و زمان مرگ را تعریف می‌کند. Topology مسئول هماهنگی بین لایه‌ها و تخصیص مناسب وظایف به نودها بر اساس الگوریتم‌های تعریف شده است. در ServiceZone در واقع ZoneManagerها قرار دارند. هدف ZoneManager مدیریت تسک‌ها و تخصیص آن‌ها به نودهای مناسب و همچنین مدیریت مهاجرت‌های موجود در منطقه تحت پوشش خود (Zone) است.

ارزیابی و نظارت (Evaluation and Monitoring):

کلاس Evaluator برای ارزیابی عملکرد سیستم طراحی شده است و شامل متریک‌هایی مانند تعداد مهاجرت‌ها، تعداد مواعد از دست رفته و کل وظایف است. این کلاس وضعیت سیستم را بررسی و گزارش می‌دهد.

پیکربندی سیستم (System Configuration):

تنظیم‌های مربوط به سیستم در کلاس Config تعریف شده‌اند، که شامل تنظیمات مربوط به پوشش (coverage)، طول تایم‌اسلات، دوره شبیه‌سازی، و دیگر پارامترهای مرتبط می‌شوند.

این مدل سیستم به صورت یکپارچه طراحی شده تا امکان شبیه‌سازی عملکرد دستگاه‌های متحرک در یک محیط محاسباتی توزیع شده را فراهم آورد و بتواند به طور مؤثر به چالش‌های موجود پاسخ دهد.

روند الگوریتم

همان طور که گفته شد، قرار است چیزی شبیه به یک سیستم خودران هوشمند طراحی شود که در لایه‌ی کاربر یک سری خودروی متحرک قرار دارند. این خودروها در هر زمانی ممکن است یک تسک تولید کنند. در ادامه چند بخش اصلی و توضیحات مربوط به هر یک را بررسی می‌کنیم:

تولید یک تسک از سمت یک خوردو

تسک از سمت خودرو تولید می‌شود. حال قرار است اجرا کننده‌ی تسک مشخص شود. روند به این صورت است که خودرو یک پکت شامل پروفایل تسک (حاوی سایز تسک، توان مصرفی، و مشخصات خودرو شامل موقعیت، سرعت و زاویه) خود را broadcast می‌کند. این پیغام درخواست به zone managerهایی که خودرو در محدوده‌ی پوشش آن‌ها است ارسال می‌شود.

حال هر zone manager که در محدوده مناسب باشد، در بین fog nodeهایی که در پوشش دارد و می‌توانند تسک را انجام دهند می‌گردد و یک پکت تحت عنوان offer به خودرو ارسال می‌کند. offer شامل مختصات و ویژگی‌های fog node ای است که آن zone manager پیشنهاد می‌دهد.

در نهایت خودرو از میان پیشنهادهای دریافت شده یکی را accept می‌کند و به zone manager مربوطه ارسال می‌کند. پس از آن zone manager دوباره بررسی می‌کند و در صورتی که مشکلی نبود (مثلا fog node اشغال نشده بود) تسک را به این گره اساین می‌کند. نتیجه‌ی موفقیت آمیز بودن یا نبودن تخصیص پس از آن به خودرو اعلام می‌شود.

در صورتی که در هر حال تسک به گره مهی تخصیص داده نشد، تسک به سمت گره cloud فرستاده می‌شود. در صورتی که این کار هم ممکن نباشد، به صورت کلی تسک miss خواهد شد.

- روند تصمیم‌گیری برای **تعیین ناحیه‌هایی** که درخواست تخصیص تسک به آن‌ها ارسال شود:
  ابتدا ناحیه‌هایی که خودرو در حال حاضر تحت پوشش آن‌ها هستند پیدا می‌شوند. برای جلوگیری از بروز مهاجرت، مختصات خودرو در لحظه‌ای که قرار است تسک به اتمام برسد نیز تخمین زده می‌شود. سپس ناحیه‌هایی که احتمال می‌رود خودرو در آن لحظه تحت پوشش آن‌ها قرار داشته باشد نیز پیدا می‌شود. ناحیه‌ی ایده‌آل ناحیه‌ای است که در اشتراک این نواحی باشد. در صورتی که ناحیه‌ی ایده‌آلی وجود نداشت، صرفا همان ناحیه‌هایی که ابتدا بدست آمده بودند بررسی می‌شوند.

- روند تصمیم‌گیری برای تخصیص به **گره‌های مه** درون یک ناحیه:

دستگاه zone manager با آمدن یک تسک، ابتدا همه‌ی fog nodeهایی که در محدوده‌ی خودرو هستند را پیدا می‌کند. سپس مانند حالت قبلی پیش‌بینی می‌کند که کدام یک از fog nodeها در لحظه‌ی اتمام تسک هم می‌توانند خودرو را پوشش دهند و با توجه به این موارد گزینه‌ی مطلوب را انتخاب می‌کند. نکته‌ی مهم و متفاوت این است که باید توجه کنیم fog nodeها هم دو نوع متحرک و ثابت دارند و برای پیش‌بینی پوشش دادن یا ندادن تسک‌ها، باید سرعت و زاویه‌ی آن‌ها را هم در نظر گرفت. در نهایت، هیوریستیک انتخاب شده فاصله است و هر گرهی که در این بین فاصله‌ی کمینه داشته باشد انتخاب می‌شود.

اولویت انتخاب پیشنهاد (offer) توسط خودرو هم بر اساس نزدیک بودن گره مه است.

اتمام پردازش یک تسک در گره مه بدون مهاجرت:

در این حالت قرار است نتیجه‌ی تسک به خودروی صاحب وظیفه منتقل شود. گره مه باید این که تسک تمام شده است را به zone manager اعلام کند. این پیام (که شامل آیدی خودروی صاحب وظیفه است) به zone managerای که تسک را در ابتدا به او داده بود ارسال می‌کند.

حال zone manager چک می‌کند که خودرو همچنان در محدوده‌ی پوشش او هست یا خیر. اگر قرار است مهاجرت نداشته باشیم، یعنی همچنان در همان محدوده مانده است. بنابراین zone manager می‌تواند مختصات خودرو را گرفته و به گره مه ارسال کند. پس از آن، گره مه خروجی تسک را برای خودرو ارسال می‌کند. در این جا همچنین می‌توان چک کرد که ددلاین تسک از دست رفته است یا خیر.

اتمام پردازش یک تسک در گره مه با مهاجرت:

شرایط مثل مورد قبل است با این تفاوت که zone manager متوجه می‌شود خودرو دیگر در محدوده‌ی پوشش او نیست. در این صورت به zone managerهای دیگر که در محدوده‌اش هستند، پیغامی می‌دهد که به دنبال خودروی با آیدی گفته شده است. zone managerهای دیگر هم اگر خودرو در محدوده‌شان بود که پاسخ می‌دهند وگرنه می‌توانند از دیگر zoneها بپرسند.

در نهایت وقتی zone manager متوجه شد که خودرو مهاجرت کرده است و دیگر در محدوده‌ی پوشش او و گره مه نیست، خروجی تسک را از گره مه می‌گیرد و به صورت غیرمستقیم (از طریق دیگر zone manageها) به دست خودرو می‌رساند. توجه کنید که در این حالت به تاخیر تسک مقداری زمان اضافه می‌شود که باید در نظر گرفته شود. البته در پروژه ما این تاخیر را ثابت در نظر گرفته‌ایم (در صورت وجود مهاجرت).

مهاجرت یک گره مه و خروج از یک ناحیه:

در طول اجرای الگوریتم و به صورت متناوب، با آپدیت شدن گراف، هر ناحیه‌ای چک می‌کند که گره‌های مهی که در پوشش داشت همچنان تحت پوشش باشند. اگر نبودند، آن‌ها را از لیست خود خارج می‌کند. برعکس آن نیز رخ می‌دهد و هر واحد زمانی حرکت گره‌های مه، ممکن است تحت پوشش ناحیه‌های دیگری در آیند و این نیز آپدیت می‌شود. توجه کنید که اگر تسک یک گره مه تمام شد و zone managerای که قبلا تسک را به او ارسال کرده بود دیگر در منطقه‌ی پوشش گره مه نبود، نتیجه‌ی تسک به zone managerای که تحت پوشش هست ارسال می‌شود.

نتیجه اجرای الگوریتم

ورودی‌ها:

- یک توزیع از خودروها و حرکت آن‌ها در هر زمان (در کل ۹۹ واحد زمانی)
- یک توزیع از تسک‌های تولید شده توسط خودروها به همراه مشخصات (در کل ۳۰۰ تسک)
- یک توزیع از گره‌های مه ثابت (در کل ۱۰ گره)
- یک توزیع از گره‌های مه متحرک (در کل ۲۰۰۰ گره)
- توزیعی از ناحیه‌ها و zone manager مربوط به هر یک (در کل ۲۰ ناحیه)
- یک گره ابر

نتیجه‌ی خروجی:

مهاجرت‌های کل: **۱۰ درصد**

ددلاین‌های از دست رفته‌ی کل: **۱۹ درصد**

```
Total migrations:   30
Total deadline misses:  58
Total cloud tasks:  0
Total tasks:    298
Migration ratio:    0.10067114093959731
Deadline miss ratio:    0.19463087248322147
```

مقایسه با اجرای الگوریتم تصادفی

در این حالت تسک‌ها به صورت تصادفی به گره‌ها تخصیص داده می‌شوند و تنها چیزی که در طول الگوریتم چک می‌شود، تحت پوشش بودن گره‌ها است.
در این صورت، خروجی به دست آمده به این صورت خواهد بود:

```
Total migrations:   211
Total deadline misses:  278
Total cloud tasks:  0
Total tasks:    298
Migration ratio:    0.7080536912751678
Deadline miss ratio:    0.9328859060402684
```

جمع‌بندی

در این پروژه یک مدل برای سیستم و الگوریتم برای اختصاص دادن تسک‌های یک سیستم خودروهای هوشمند طراحی شده است. هدف اصلی این است که ددلاین تسک‌ها تا جای ممکن از دست نرود و همچنین علاوه بر تلاش برای جلوگیری از رخ دادن مهاجرت، در صورتی که مهاجرت رخ داد بتوان مواجهه با مهاجرت نیز داشت. در نهایت خروجی پروژه یک سیستم مدل و یک الگوریتم است که در واقع این سیستم را شبیه‌سازی می‌کند. علاوه بر آن تعدادی اسکریپت جهت ساخت توزیع مناسب از تسک‌ها و خودروها و گره‌ها نیز نوشته شده‌اند. همگی خروجی‌های پروژه، از صفحه‌ی گیتهاب پروژه نیز قابل دسترسی هستنند.

در نهایت، بابت کمک‌ها و راهنمایی‌های شما حین انجام پروژه سپاس‌گزاریم.