

دانشگاه تهران
پردیس دانشکده‌های فنی
دانشکده مهندسی برق و کامپیوتر

هوشمندی توزیع شده در شبکه‌های اینترنت اشیاء

نگارش

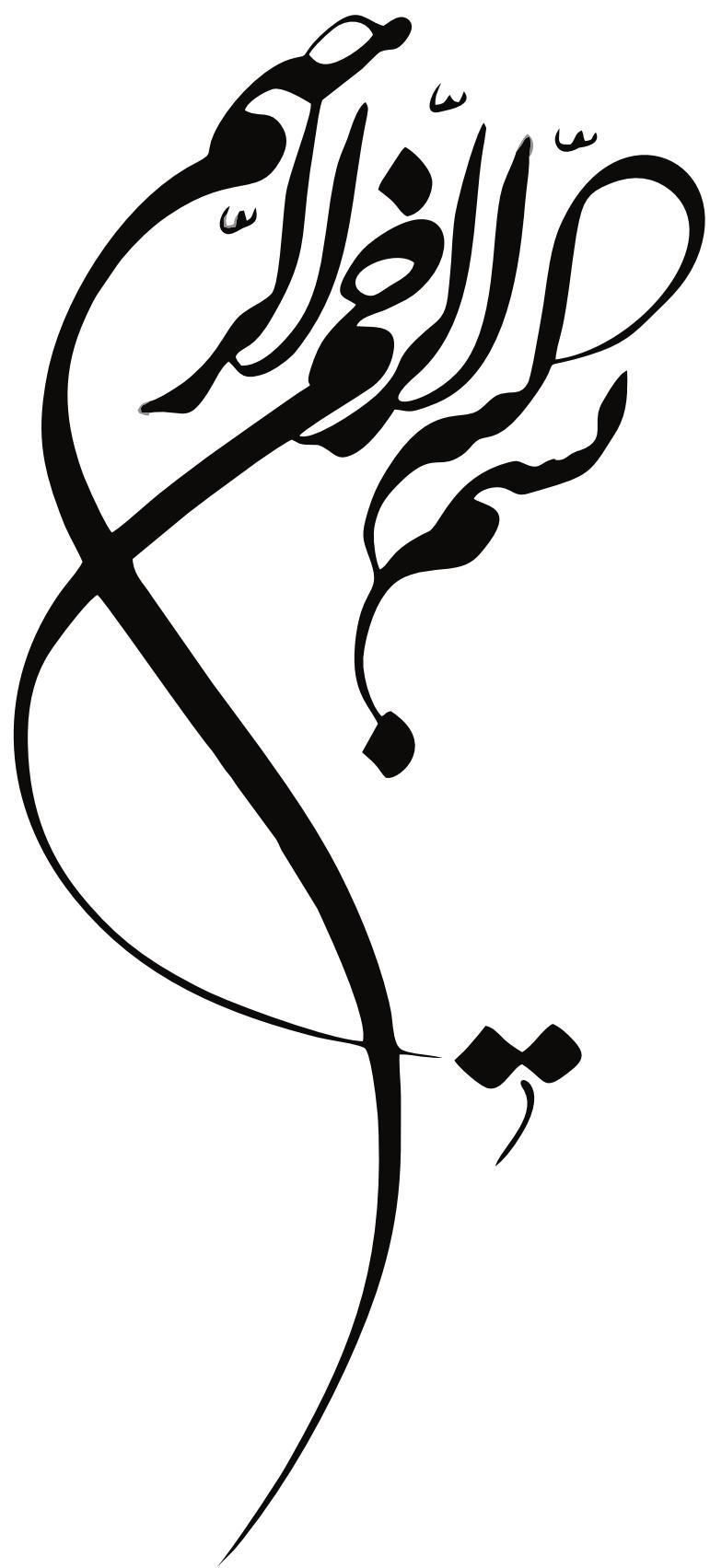
محمد محمودیان

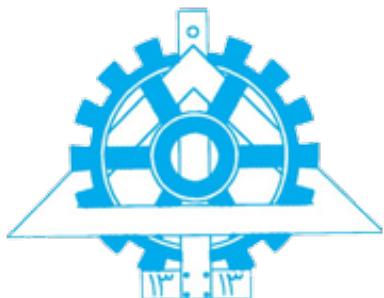
استاد راهنما

دکتر وحید شاهمنصوری

پایان‌نامه برای دریافت درجه کارشناسی ارشد در رشته مهندسی برق
گرایش شبکه‌های مخابراتی

۱۳۹۹ مهر





دانشگاه تهران
پردیس دانشکده‌های فنی
دانشکده مهندسی برق و کامپیوتر

هوشمندی توزیع شده در شبکه‌های اینترنت اشیاء

نگارش

محمد محمودیان

استاد راهنما

دکتر وحید شاهمنصوری

پایان‌نامه برای دریافت درجه کارشناسی ارشد در رشته مهندسی برق
گرایش شبکه‌های مخابراتی

۱۳۹۹ مهر

تأییدیه‌ی هیأت داوران جلسه‌ی دفاع از پایان‌نامه

نام دانشکده: پردیس دانشکده‌های فنی

نام دانشجو: محمد محمودیان

عنوان پایان نامه: هوشمندی توزیع شده در شبکه های اینترنت اشیاء

تاریخ دفاع: مهر ۱۳۹۹

رشته: مهندسی برق

گرایش: شبکه‌های مخابراتی

ردیف	سمت	نام و نام خانوادگی	مرتبه دانشگاهی	دانشگاه یا مؤسسه	امض
۱	استاد راهنمای	دکتر وحید شاه منصوری	استادیار	دانشگاه تهران	
۲	استاد مدعو خارجی	دکتر نادر مکاری یامچی	استادیار	دانشگاه تربیت مدرس	
۳	داور و نماینده تحصیلات تکمیلی دانشکده	دکتر احمد خونساری	دانشیار	دانشگاه تهران	

تعهدنامه اصالت اثر

اینجانب محمد محمودیان تائید می نمایم که مطالب مندرج در این پایان نامه حاصل کار پژوهشی اینجانب است و به دستاوردهای پژوهشی دیگران که در این نوشه از آن ها استفاده شده است، طبق مقررات ارجاع گردیده است. این پایان نامه قبل احراز هیچ مدرک هم سطح یا بالاتر ارائه نشده است.

کلیه حقوق مادی و معنوی این اثر متعلق به دانشکده فنی دانشگاه تهران است.

نام و نام خانوادگی: محمد محمودیان

تاریخ و امضا:

تقدیم به:

پدر، مادر و خواهرم

قدردانی

سپاس خداوندگار حکیم را که با لطف بی‌کران خود، آدمی را زیور عقل آراست.
در آغاز وظیفه خود می‌دانم از زحمات بی‌دریغ استاد راهنمای خود، جناب آقای دکتر وحید شاهمنصوري،
صمیمانه تشکر و قدردانی کنم که قطعاً بدون راهنمایی‌های ارزنده ایشان، این مجموعه به انجام نمی‌رسید.
در پایان، بوسه می‌زنم بر دستان خداوندگاران مهر و مهربانی، پدر و مادر عزیزم و بعد از خدا، ستایش می‌کنم
وجود مقدس‌شان را و تشکر می‌کنم از خانواده عزیزم به پاس عاطفه سرشار و گرمای امیدبخش وجودشان،
که بهترین پشتیبان من بودند.

محمد محمودیان

۱۳۹۹ مهر

چکیده

شهر هوشمند^۱ یکی از کاربردهای مهم اینترنت اشیاء (IoT^۲) است که اخیراً مورد توجه محققان قرار گرفته است. در شهر هوشمند هدف این است که تمام دستگاه‌های موجود امکان اتصال به شبکه اینترنت را داشته باشند. اطلاعات تولید شده توسط این دستگاه‌ها در ابتدا نیاز به پردازش و تحلیل دارد. از این‌رو باید این اطلاعات به منابع پردازشی ارسال و نتیجه پردازش به صورت تصمیم‌هایی به دستگاه‌های مربوطه بازگشت داده شود. یکی از چالش‌های مهم در پیاده سازی الگوی شهر هوشمند، مکان و نحوه پردازش اطلاعات دریافتی است. ساده‌ترین راه حل ارائه شده برای این چالش، پردازش کلیه اطلاعات در یک محل متمرکز به نام فضای ابری^۳ است. با گسترش تعداد دستگاه‌ها، افزایش حجم اطلاعات و نیازهای پردازشی متفاوت سرویس‌ها، این روش متمرکز پردازش اطلاعات، پاسخگوی نیاز تمام دستگاه‌های شهر هوشمند نخواهد بود؛ به همین منظور پردازش لبه^۴ و پردازش مه^۵ به عنوان دو الگوی پردازشی جدید مورد توجه محققان قرار گرفته‌اند. مهم‌ترین مزیت این دو الگو نسبت به پردازش ابری افزایش سرعت در پاسخ‌دهی به اطلاعات دریافتی است؛ اما با اضافه شدن این دو الگو در کنار پردازش ابری مسئله تخصیص منابع برای پردازش اطلاعات به یک چالش مهم تبدیل شده است. لذا در این پایان‌نامه مدل‌سازی و حل مسئله تخصیص منابع با درنظر گرفتن سه لایه لبه، مه و ابری مورد بررسی قرار گرفته است. مسئله اصلی در حالت کلی به صورت یک مسئله بهینه سازی غیرخطی ترکیب عدد صحیح^۶، با هدف کمینه کردن هزینه‌های پردازشی مدل‌سازی شده است. با توجه به NP-hard^۷ بودن مسئله بهینه‌سازی اولیه، مسئله مورد نظر خطی سازی شده و به سه روش متمرکز، غیرمتمرکز و توزیع شده مورد بررسی قرار گرفته است. با توجه به اینکه سه روش فوق نیاز به خطی سازی مسئله دارند، یک راه حل زیربهینه به صورت اکتشافی با کمک از الگوریتم ویتبی (VTP)^۸ ارائه شده است که مسئله غیرخطی اولیه را مستقیماً حل می‌کند. با توجه به شبیه‌سازی‌های انجام شده می‌توان گفت که روش‌های متمرکز و غیرمتمرکز به جواب بهینه می‌رسند. روش توزیع شده در تعداد محدودی از تبادل اطلاعات بین گره‌ها به جواب زیربهینه با دقت دلخواه^۹ می‌رسد. این روش در شبکه‌های غیرهمزمان^{۱۰} نیز به جواب می‌رسد. دو روش غیرمتمرکز و توزیع شده بر خلاف روش متمرکز در مدت زمان چند جمله‌ای به جواب می‌رسند و در شبکه‌های خیلی بزرگ به خوبی می‌توانند مورد استفاده قرار بگیرند. راه حل VTP نیز در زمان چند جمله‌ای و خیلی سریع‌تر از سه روش دیگر به جواب مسئله غیرخطی اولیه می‌رسد.

واژگان کلیدی: اینترنت اشیاء، شهر هوشمند، پردازش لبه، تخصیص منابع، راه حل غیرمتمرکز، راه حل توزیع شده

¹Smart city

²Internet of Things

³Cloud

⁴Edge computing

⁵Fog computing

⁶Mixed-integer non-linear programming

⁷Viterbi-based Task Placement

⁸Asynchronous

فهرست مطالب

ت

فهرست تصاویر

ج

فهرست جداول

۱

فصل ۱: مقدمه

۳

۱.۱ اینترنت اشیاء و شهر هوشمند

۵

۲.۱ خدمات شهر هوشمند

۵

۱.۲.۱ نظارت بر سلامت ساختمان‌ها

۵

۲.۲.۱ مدیریت پسماند

۶

۳.۲.۱ نظارت بر کیفیت هوای

۶

۴.۲.۱ نظارت بر آلودگی صوتی

۶

۵.۲.۱ مدیریت ترافیک

۷

۶.۲.۱ نظارت بر مصرف انرژی در شهر

۷

۷.۲.۱ پارک هوشمند

۷

۸.۲.۱ روشنایی هوشمند

۸

۹.۲.۱ اتوماسیون ساختمان‌های عمومی

۸

۳.۱ سرویس‌ها و وظیفه‌ها

۹

۱.۳.۱ انواع تاخیر در شبکه‌های سوئیچ بسته

۱۰

۴.۱ پردازش لبه

۱۱

۱.۴.۱ چرا به پردازش لبه نیاز داریم؟

فهرست مطالب

۱۲	۲.۴.۱ پردازش لبه چیست؟
۱۳	۳.۴.۱ مزایای پردازش لبه
۱۴	۴.۴.۱ کاربرد پردازش لبه در شهر هوشمند
۱۴	۵.۱ پردازش مه
۱۵	۱.۵.۱ عملکرد پردازش مه چگونه است؟
۱۵	۲.۵.۱ مزایای پردازش مه
۱۶	۶.۱ انواع لایه در شبکه
۱۷	۷.۱ نوآوری‌های پایان نامه
۱۷	۸.۱ ساختار پایان نامه
۱۹	فصل ۲: مروری بر مطالعات انجام شده
۱۹	۱.۲ تخصیص منابع پردازشی در شبکه اینترنت اشیاء
۲۹	فصل ۳: تخصیص منابع پردازشی در شبکه اینترنت اشیاء به صورت متتمرکز و غیرمتتمرکز
۲۹	۱.۳ مقدمه
۳۰	۲.۳ مدل سیستم
۳۴	۱.۲.۳ متغیرها
۳۵	۲.۲.۳ قیدها
۴۲	۳.۲.۳ تابع هدف
۴۴	۳.۳ راه حل غیرمتتمرکز
۵۱	۴.۳ بررسی همگرایی و پیچیدگی
۵۲	۵.۳ نتایج شبیه‌سازی
۵۷	۶.۳ جمع‌بندی و نتیجه‌گیری
۵۹	فصل ۴: راه حل‌های اکتشافی و توزیع شده
۵۹	۱.۴ مقدمه
۶۰	۲.۴ مقدمه‌ای بر ویتری

فهرست مطالب

۶۱	۳.۴ راه حل اکتشافی مبتنی بر ویترینی (VTP)
۶۷	۴.۴ راه حل توزیع شده
۶۸	۱.۴.۴ الگوریتم توزیع شده
۷۴	۵.۴ بررسی همگرایی و پیچیدگی
۷۵	۶.۴ نتایج شبیه سازی
۷۹	۷.۴ جمع بندی و نتیجه گیری
۸۱	فصل ۵: نتیجه گیری و کارهای آینده
۸۱	۱.۵ خلاصه و جمع بندی
۸۲	۲.۵ کارهای آینده
۸۳	مراجع

فهرست تصاویر

۱۲	الگوی پردازش ابری [۱]	۱.۱
۱۳	الگوی پردازش لبه [۱]	۲.۱
۲۰	مدل سیستم مقاله [۲]	۱.۲
۲۱	مدل سیستم مقاله [۳] با چهار زیر سیستم	۲.۲
۲۲	یک گراف سرویس در [۴]	۳.۲
۲۳	مدل سیستم مقاله [۵]	۴.۲
۲۴	مدل سیستم مقاله [۶]	۵.۲
۲۵	مدل سیستم مقاله [۷]	۶.۲
۲۶	مدل سیستم مقاله [۸]	۷.۲
۲۷	مدل سیستم مقاله [۹]	۸.۲
۳۰	دید کلی از مدل سیستم فصل (۳)	۱.۳
۵۲	دید کلی از توپولوژی شبکه	۲.۳
۵۳	مقدار تابع هدف(هزینه) دربرابر تعداد کل گره‌های موجود در شبکه برای دو روش متمرکز و غیرمتمرکز	۳.۳
۵۴	نحوه همگرا شدن اندازه ضرایب لاگرانژین در دو گره تصادفی در روش غیرمتمرکز	۴.۳
۵۵	مقدار تابع هدف(هزینه) دربرابر میانگین حداکثر تاخیر قابل قبول برای وظیفه‌ها δ_t	۵.۳
۵۵	درصد استفاده از گره‌های لایه‌های مختلف دربرابر میانگین حداکثر تاخیر قابل قبول برای وظیفه‌ها δ_t	۶.۳

- ۷.۳ میانگین تعداد شکسته شدن وظیفه‌ها بین گره‌های مختلف دربرابر میانگین نرخ تولیدی وظیفه‌ها در گره‌های حسگری $\lambda_{t,s}$ ۵۶
- ۱.۴ مثالی از نحوه عملکرد الگوریتم (۱.۴) ۶۵
- ۲.۴ مثالی از چند وجهی P که قسمت خاکستری رنگ است و همچنین مجموعه قابل قبول جواب مسئله یعنی P_I که با خطوط پرنگ قرمز مشخص شده است و مجموعه پوش محدب از جواب قابل قبول یعنی $\text{conv}(P_I)$ که ناحیه هاشور خورده آبی رنگ است. فضای استفاده شده به صورت دو بعدی است. ۶۹
- ۳.۴ نحوه همگرا شدن متغیر محلی در دو گره تصادفی در روش توزیع شده ۷۶
- ۴.۴ نحوه همگرا شدن اندازه‌ی متغیر محلی مسئله در دو روش توزیع شده و غیرمت مرکز برای دو گره تصادفی ۷۷
- ۵.۴ نحوه همگرا شدن مقدار خطأ در دو روش توزیع شده و غیرمت مرکز ۷۷
- ۶.۴ مقدار تابع هدف (هزینه) برای چهار روش مختلف دربرابر تعداد کل گره‌های شبکه ۷۸
- ۷.۴ زمان نسبی آماده شدن جواب نهایی در چهار روش مختلف دربرابر تعداد گره‌های شبکه ۷۹

فهرست جداول

۱.۳ نمادهای استفاده شده در فصل (۳) ۳۳

فصل ۱

مقدمه

اینترنت اشیاء (IoT^۱) امکان اتصال اشیاء هوشمند به یکدیگر را ممکن می‌سازد. اتصال این دستگاه‌های هوشمند به یکدیگر باعث می‌شود که این دستگاه‌ها بتوانند با یکدیگر و با اینترنت داده مبادله کنند. تبادل اطلاعات این امکان را ایجاد می‌کند که بتوانند خدمات متنوعی را برای کاربران فراهم کنند که قبلاً امکان ارائه آن‌ها نبوده است. پیشرفت دستگاه‌های هوشمند و ظهور روش‌های جدید پردازش داده، اینترنت اشیاء را به عنوان گزینه‌ای مناسب برای استفاده در شهر هوشمند^۲، شبکه هوشمند انرژی^۳، خانه هوشمند^۴ و سلامتی هوشمند^۵ قرار داده است. امروزه یکی از مسائل پرطرفدار در اینترنت اشیاء، شهر هوشمند است که در آن هدف این است که کلیه وسائل موجود به نحوی هوشمند شده و ارتباطات بین آن‌ها به دور از نقش انسان و به صورت کاملاً مکانیزه انجام شود. به دلیل علاقه دولت‌ها برای استفاده از اینترنت اشیاء در بهبود مدیریت امور عمومی، شهر هوشمند به عنوان بهترین راهکار تحقق وسیع اینترنت اشیاء در نظر گرفته می‌شود. یکی از مهم‌ترین بخش‌های اینترنت اشیاء، پردازش داده‌هایی است که توسط حسگرهای مختلف جمع‌آوری شده‌اند. به طور سنتی پردازش ابری^۶ روشی کارا برای پردازش داده بوده است چرا که ظرفیت پردازشی ابر بسیار بیشتر از دیگر روش‌های پردازشی است. با این حال، افزایش چشمگیر در تعداد دستگاه‌های هوشمند و

¹Internet of Things

²Smart city

³Smart grid

⁴Smart home

⁵Smart health

⁶Cloud Computing

افزایش وسیع در میزان حجم اطلاعات موجود جهت پردازش و همچنین کافی نبودن پهنانی باند شبکه‌ها برای انتقال حجم انبوی داده‌های تولید شده در اینترنت اشیاء باعث می‌شود که انتقال داده‌ها و پردازش همه‌ی آن‌ها در فضای ابری به عنوان گلوبال پردازش ابری باشد. به همین دلیل، ارسال همه‌ی داده‌ها برای پردازش به فضای ابری، می‌تواند باعث زمان پاسخ طولانی سرویس‌ها شود که برای بسیاری از سرویس‌ها قابل قبول نیست. از پردازش لبه^۱ و پردازش مه^۲ به عنوان راه حل‌هایی برای این مشکل یاد می‌شود. در پردازش لبه هدف این است که پردازش داده‌ها تا حد ممکن در جایی که داده‌ها تولید می‌شوند انجام شود. پردازش مه نیز مشابه‌تر زیادی به پردازش لبه دارد با این تفاوت که فاصله گره‌های پردازشی تا حسگرها نسبت به پردازش لبه بیشتر است.

در یک شبکه اینترنت اشیاء تعداد بسیار زیادی حسگر^۳ و فعال‌کننده^۴ مانند حسگرها دود^۵، حسگرهای دما^۶، حسگرهای حرکت^۷، دوربین‌های نظارتی و هشدارهای خطر آتش وجود دارند. همچنین سرویس‌های^۸ زیادی هم وجود دارند که از داده‌های این حسگرها استفاده و با پردازش این داده‌ها، نتیجه‌هایی را تولید می‌کنند. برای هر سرویس، داده‌های حسگرها باید به یک منبع پردازشی ارسال شوند و بعد از پردازش نتیجه به مقصد های مورد نظر فرستاده شوند. این مقصد های می‌توانند فعال‌کننده‌ها یا ذخیره‌سازهای ابری و غیره باشند. به عنوان نمونه می‌توان سرویس تشخیص آتش سوزی را نام برد. در این سرویس، ورودی‌ها می‌توانند حسگرهای دود، حسگرهای دما و دوربین‌های ویدیویی باشند و هشدار دهنده‌های آتش، خروجی باشند. قسمت پردازش، ورودی حسگرها و دوربین‌ها را مورد بررسی قرار می‌دهد و هشدار دهنده‌ها را فعال می‌کند یا می‌تواند به ایستگاه‌های آتشنشانی اطلاع دهد. به عنوان نمونه‌ی دیگر، سرویس‌های امنیت ساختمان‌ها را در نظر بگیرید. در این سرویس‌ها، داده‌های حسگرهای حرکتی و دوربین‌های نظارتی پردازش می‌شوند و در صورت تشخیص نفوذ غیر مجاز هشدار دهنده‌ها فعال می‌شوند و به پلیس اطلاع داده می‌شود. اطلاعات ورودی هر سرویس را می‌توان در قالب مجموعه‌ای از بسته‌ها و تحت عنوان وظیفه^۹ در نظر گرفت که هر وظیفه ویژگی‌ها و قیدهای مشخص و مخصوص به خود را دارد.

¹Edge Computing

²Fog Computing

³Sensor

⁴Actuator

⁵Smoke sensors

⁶Temperature sensors

⁷Motion sensors

⁸Services

⁹Task

هرچقدر تعداد اطلاعات ارسالی مربوط به یک محیط بیشتر باشد می‌توان گفت که احتمال بروز خطا در محاسبات کمتر می‌شود زیرا تصمیم‌گیری صرفا براساس اطلاعات مربوط به یک حسگر انجام نمی‌شود. سرویس‌ها برای پردازش داده‌های خود باید در منابع پردازشی مناسب پردازش شوند بهطوری‌که تمام قیدهای مربوط به آن‌ها به بهترین وجه ممکن برآورده شود. تعداد بسیار زیاد سرویس‌ها و منابع پردازشی در شبکه اینترنت اشیاء باعث می‌شود که مسئله پیدا کردن منبع پردازشی بهینه برای سرویس‌ها، یک مسئله پیچیده باشد. به همین دلیل در این پایان نامه به بررسی و ارائه راه حل برای حل مسئله اختصاص منابع پردازشی در اینترنت اشیاء می‌پردازیم.

۱.۱ اینترنت اشیاء و شهر هوشمند

واژه‌ی اینترنت اشیاء برای اولین بار توسط کوین اشتون^۱ در یک ارائه برای استفاده از بازشناسی با امواج رادیویی^۲ در مدیریت زنجیر تأمین^۳ استفاده شد[۱۰]. اینترنت اشیاء امکان اتصال هر کسی در هر مکان و زمانی به هر چیزی در هر مکانی و هر زمانی را فراهم می‌کند. با پیشرفت تکنولوژی به سمت جامعه‌ای پیش می‌رویم که همه افراد و همه اشیاء متصل خواهند بود[۱۱]. ایده‌ی اصلی اینترنت اشیاء این است که امکان اتصال خودکار و امن و انتقال داده بین دستگاه‌های فیزیکی و برنامه‌های کاربردی را فراهم می‌کند. در واقع اینترنت اشیاء این امکان را ایجاد می‌کند که اشیاء فیزیکی بتوانند بینند، بشنوند، و با صحبت کردن با یکدیگر بتوانند تصمیم‌گیری کنند و کارهایی را انجام دهند[۱۲]. در طول زمان انتظار می‌رود اینترنت اشیاء کاربردهای خانگی و تجاری فراوانی داشته باشد، کیفیت زندگی افراد را بهبود ببخشد و باعث رشد اقتصاد جهانی بشود. هدف اینترنت اشیاء این است که اینترنت را فراگیرتر و همه جانبه‌تر کند. علاوه بر این، به وسیله‌ی دسترسی آسان و تعامل با طیف گسترده‌ای از دستگاه‌هایی مانند لوازم خانگی، دوربین‌های نظارتی، حسگرهای فعال‌کننده‌ها، نمایشگرها، خودروها و غیره، اینترنت اشیاء به توسعه‌ی کاربرد داده‌های تولید شده توسط این دستگاه‌ها برای فراهم کردن خدمات به شهروندان، شرکت‌ها و اداره‌ی امور عمومی کمک می‌کند. این الگو، کاربردهایی در حوزه‌های مختلف مانند اتوماسیون خانگی، اتوماسیون صنعتی، کمک‌های پزشکی، سلامت همراه، نگهداری از افراد سالم‌مند، مدیریت هوشمند انرژی و شبکه هوشمند، وسایل نقلیه، مدیریت ترافیک و

¹Kevin Ashton

²RFID

³Supply Chain

موارد دیگر دارد [۱۳].

در چنین عرصه‌ی ناهمگونی از کاربردها، پیدا کردن یک راه حل که بتواند به همه‌ی نیازهای کاربردهای مختلف پاسخ بدهد، چالش برانگیز خواهد بود. دشواری پیدا کردن این راه حل باعث ایجاد راه حل‌های متفاوت و بعض‌اً ناسازگار برای تحقق سامانه‌های اینترنت اشیاء شده است [۱۴]. بنابر این از دید سیستمی، به دلیل نوآوری‌ها و پیچیدگی‌های اینترنت اشیاء، نیاز به تحقق یک شبکه‌ی اینترنت اشیاء، به همراه شبکه سرویس‌ها و شبکه دستگاه‌ها وجود دارد. علاوه بر مشکلات فنی، عدم وجود یک مدل تجاری مورد قبول که بتواند سرمایه‌گذاران را برای ترویج استقرار این تکنولوژی‌ها جذب کند مانع توسعه سریع استفاده از اینترنت اشیاء شده است [۱۵].

در این سناریوی پیچیده، کاربرد اینترنت اشیاء در محیط‌های شهری، از محبوبیت بالایی برخوردار است چرا که به تقاضای بسیاری از دولت‌ها برای استفاده از فناوری اطلاعات و ارتباطات در مدیریت امور عمومی پاسخ می‌دهد و باعث می‌شود مفهومی که از آن با عنوان شهر هوشمند یاد می‌شود، تحقق یابد [۱۶]. در حالی که هنوز تعریفی که موردن قبول همگان برای شهر هوشمند باشد وجود ندارد، هدف نهایی استفاده‌ی بهتر از منابع عمومی، افزایش کیفیت خدمات ارائه شده به شهروندان و کاهش هزینه‌های عملیاتی مدیریت امور عمومی است. این اهداف به کمک اینترنت اشیاء شهری قابل دستیابی خواهند بود. منظور از اینترنت اشیاء شهری، زیرساخت ارتباطی است که دسترسی یکپارچه، ساده و اقتصادی را به مجموعه‌ای از خدمات عمومی فراهم می‌کند. اینترنت اشیاء شهری می‌تواند مزایایی در مدیریت و بهینه‌سازی خدمات عمومی مرسوم مانند حمل و نقل و پارک خودروها، روشناهی، نظارت و نگهداری اماکن عمومی، حفظ آثار باستانی، جمع آوری پسماند، بیمارستان‌ها و مدارس داشته باشد. علاوه بر این، وجود انواع مختلف داده‌های جمع آوری شده توسط یک اینترنت اشیاء شهری فراگیر، می‌تواند برای افزایش شفافیت استفاده شود، اقدامات تصمیم‌گیران محلی را ترویج دهد، آگاهی شهروندان راجع به وضعیت شهرشان را بیشتر کند، شهروندان را به مشارکت فعال در مدیریت عمومی تشویق کند و باعث خلق سرویس‌های جدیدی علاوه بر سرویس‌های فراهم شده توسط اینترنت اشیاء بشود [۱۷]. بنابراین استفاده از اینترنت اشیاء به صورت خاص برای تصمیم‌گیران محلی و منطقه‌ای جذاب است و باعث می‌شود که آن‌ها اولین استفاده کننده‌های این تکنولوژی‌ها باشند و به عنوان کاتالیزور برای انطباق الگوی اینترنت اشیاء در مقیاس‌های بزرگ‌تر عمل کنند.

۲.۱ خدمات شهر هوشمند

در ادامه برخی از خدماتی که امکان ارائه آن‌ها توسط اینترنت اشیاء شهری فراهم می‌شود را مرور می‌کنیم.

۱.۲.۱ نظارت بر سلامت ساختمان‌ها

نگهداری مناسب ساختمان‌های تاریخی شهرها نیاز به نظارت مداوم وضعیت واقعی ساختمان‌ها و پیداکردن مکان‌هایی که بیشترین تاثیر را از عوامل خارجی دریافت می‌کنند دارد. اینترنت اشیاء شهری می‌تواند یک پایگاه داده‌ی توزیع شده از اندازه‌گیری‌های یکپارچگی ساختاری ساختمان‌ها فراهم کند که به وسیله‌ی حسگرهای مناسبی که در نقاط مختلف ساختمان نصب شده‌اند، اندازه‌گیری شده‌اند. این حسگرهای می‌توانند، لرزش و تغییر شکل، رطوبت و دما را اندازه‌گیری کنند و شرایط محیطی را به طور کامل مشخص کنند[۱۸]. این پایگاه داده باید بتواند هزینه بالای لازم برای آزمایش دوره‌ای مقاومت ساختمان‌ها توسط عوامل انسانی را کاهش دهد و امکان نظارت فعال بر وضعیت ساختمان‌ها را فراهم کند. همچنین امکان ترکیب کردن اطلاعات لرزش ساختمان‌ها و اطلاعات مربوط به زمین لرزه‌های کوچک برای بررسی و فهم تاثیر آن‌ها بر ساختمان‌ها فراهم می‌شود. با این وجود تحقق عملی این خدمت، نیاز به نصب حسگرهایی در ساختمان‌ها و محیط‌های اطراف و ارتباطشان با یک مرکز کنترل دارد که ممکن است نیاز به یک سرمایه‌گذاری اولیه برای ایجاد این زیرساخت‌ها داشته باشد.

۲.۲.۱ مدیریت پسماند

هزینه‌ی بالای خدمات مدیریت پسماند و مشکلات نگهداری پسماند در محل‌های دفن زباله باعث می‌شود که مدیریت پسماند یکی از مهم‌ترین مشکلات در بیشتر شهرهای بزرگ می‌باشد. نفوذ بیشتر راه حل‌های مبتنی بر فناوری‌های ارتباطات و اطلاعات در این حوزه می‌تواند به طور چشمگیری باعث کاهش هزینه‌ها بشود و بهبودهایی در زمینه‌های اقتصادی و زیست محیط داشته باشد. برای مثال استفاده از سطل‌های زباله هوشمند که امکان اندازه‌گیری وزن محتويات درون سطل را دارند و امکان بهینه‌سازی فرآيند جمع‌آوری پسماند را فراهم می‌کنند، می‌توانند هزینه‌ی جمع‌آوری پسماند را کاهش دهند و کیفیت بازیافت پسماند را بیشتر کنند [۱۹]. برای رسیدن به یک سرویس جمع‌آوری پسماند هوشمند، اینترنت اشیاء باید بتواند سطل‌های زباله هوشمند را به یک مرکز کنترل وصل کند. در این مرکز کنترل یک نرم‌افزار بهینه‌سازی اجرا می‌شود تا مدیریت بهینه

کامیون‌های جمع‌آوری پسماند را مشخص کند.

۳.۲.۱ ناظارت بر کیفیت هوا

اتحادیه اروپا به صورت رسمی اهدافی را برای کاهش تغییرات اقلیمی در دهه آینده تعیین کرده است. از آن‌ها می‌توان به کاهش ۲۰ درصدی در تولید گازهای گلخانه‌ای در سال ۲۰۲۰ میلادی نسبت به سال ۱۹۹۰، کاهش ۲۰ درصدی مصرف انرژی با بهبود بهره‌وری انرژی و افزایش ۲۰ درصدی استفاده از انرژی‌های تجدیدپذیر را نام برد. در زمینه کیفیت هوا، اینترنت اشیاء می‌تواند روش‌هایی برای پایش کیفیت هوای مناطق پر از دحام و پارک‌ها ارائه دهد [۲۰]. علاوه بر این‌ها امکانات ارتباطی می‌توانند برای برنامه‌های روی دستگاه‌های هوشمند دوندگان امکان ارتباط با زیرساخت‌های لازم را فراهم کنند. به کمک این زیرساخت‌ها، افراد می‌توانند سالم‌ترین مسیر برای فعالیت‌های خارج از منزل را پیدا کنند. تحقق خدماتی از این قبیل نیازمند نصب حسگرهای آلودگی و کیفیت هوا در نقاط مختلف شهر و قرارگیری داده‌های این حسگرها به صورت عمومی در اختیار شهروندان است.

۴.۲.۱ ناظارت بر آلودگی صوتی

مسئولین شهری معمولاً قوانینی برای کاهش آلودگی صوتی در مراکز شهر برای برخی ساعت‌های تصویب می‌کنند. اینترنت اشیاء شهری می‌تواند یک سرویس ناظارت بر آلودگی صوتی ارائه دهد که وظیفه‌ی آن اندازه‌گیری سطح آلودگی صوتی در ساعتی مشخص در مناطقی که این سرویس در آن‌ها برقرار است، باشد [۲۱]. همچنین این سرویس با استفاده از الگوریتم‌های تشخیص صدا می‌تواند شکستن شیشه‌ها و یا صدای نزاع خیابانی را تشخیص دهد و با این کار باعث افزایش امنیت بشود. با این حال نصب حسگرهای صوتی به دلیل مسائل مربوط به حریم شخصی می‌تواند حساسیت بر انگیز باشد.

۵.۲.۱ مدیریت ترافیک

از دیگر خدمات شهر هوشمند که به وسیله‌ی اینترنت اشیاء شهری قابل دستیابی است، به ناظارت بر ترافیک شهر می‌توان اشاره کرد. با این که ناظارت ویدیویی ترافیک در حال حاضر در بیشتر شهرها استفاده می‌شود، استفاده از شبکه‌های گستردۀ کم توان می‌تواند منبع انبوهی از اطلاعات را در اختیار قرار دهد. ناظارت بر

ترافیک می‌تواند با نصب سامانه‌های موقعیت یاب بر روی خودروهای جدید تحقق یابد [۲۲] و با خدمات نظارت بر کیفیت هوا و نظارت بر آلودگی صوتی ترکیب شود. این اطلاعات، اهمیت زیادی برای مسئولین و شهروندان خواهد داشت چرا که باعث افزایش نظم و برنامه ریزی بهتر می‌شود.

۶.۲.۱ نظارت بر مصرف انرژی در شهر

در کنار نظارت بر کیفیت هوا، اینترنت اشیاء شهری می‌تواند خدماتی برای نظارت بر مصرف انرژی تمام شهر ارائه کند. با این کار شهروندان و مسئولین می‌توانند یک گزارش دقیق از میزان انرژی مورد نیاز برای سرویس‌های مختلف (روشنایی عمومی، حمل و نقل، چراغ‌های راهنمایی و رانندگی، دوربین‌های نظارتی، گرمایش و سرمایش ساختمان‌های عمومی و غیره) داشته باشند. این کار می‌تواند امکان شناسایی منابع اصلی مصرف انرژی را فراهم سازد تا بتوان برای بهینه‌سازی رفتار آن‌ها چاره‌ای پیدا کرد. برای ممکن ساختن این خدمات، دستگاه‌های نظارت بر میزان مصرف انرژی باید با شبکه‌های قدرت یکپارچه بشوند. هم چنین امکان کنترل فعال منابع تولید انرژی مانند صفحات خورشیدی می‌تواند باعث بهبود این خدمات بشود.

۷.۲.۱ پارک هوشمند

خدمات پارک هوشمند، مبتنی بر حسگرهای خیابان‌ها و نمایشگرهای هوشمند است که رانندگان خودروها را به بهترین جای پارک در شهر هدایت می‌کند [۲۳]. از مزایای این خدمت می‌توان کاهش زمان لازم برای برای پیدا کردن جای پارک و در نتیجه کاهش انتشار گاز CO از ماشین‌ها، کاهش ترافیک و افزایش رضایت شهروندان اشاره کرد. این خدمات می‌توانند به صورت مستقیم در زیرساخت اینترنت اشیاء شهری ادغام شوند چرا که تولید کنندگان، دستگاه‌های لازم برای آن را آماده کرده‌اند. این خدمات امکان شناسایی افراد معلوم و ارائه خدمت ویژه به آن‌ها را فراهم می‌کند. مثلاً اجازه استفاده از پارکینگ‌های خاص و یا ارائه ابزارهایی برای مشخص کردن سریع استفاده غیر مجاز از پارکینگ‌ها.

۸.۲.۱ روشنایی هوشمند

برای کاهش انرژی مصرفی شهر، استفاده بهینه از روشنایی بسیار مهم است. به طور خاص این خدمت می‌تواند شدت روشنایی خیابان‌ها را مطابق ساعت روز، وضعیت هوا و حضور افراد تنظیم کند. این خدمت برای این

که به درستی کار کند نیاز به این دارد که اطلاعات روشنایی شهر را در زیرساخت شهر هوشمند ترکیب کند. از دیگر مزایای آن این است که امکان شناسایی ساده خطای در سامانه روشنایی شهر در این روش وجود خواهد داشت.

۹.۲.۱ اتوماسیون ساختمان‌های عمومی

یکی از کاربردهای تکنولوژی‌های اینترنت اشیاء نظارت بر مصرف انرژی ساختمان‌های عمومی (مدارس، دفاتر مدیریت عمومی و موزه‌ها) و بهبود شرایط محیطی برای فعالیت افراد است. برای این منظور از انواع متفاوتی از حسگرها و فعال‌کننده‌هایی که نور، دما و رطوبت را کنترل می‌کنند استفاده می‌شود. با کنترل این پارامترها، علاوه بر کاهش هزینه‌ی گرمایش و سرمایش، سطح آسایش افرادی که در این محیط‌ها کار می‌کنند افزایش پیدا می‌کند که خود می‌تواند باعث افزایش بهره‌وری افراد شود [۲۳].

۳.۱ سرویس‌ها و وظیفه‌ها

سرویس‌های موجود در شهر هوشمند را می‌توان از چند نظر تقسیم‌بندی کرد. یکی از انواع این تقسیم‌بندی‌ها تقسیم‌بندی برمبنای کیفیت مورد انتظار (QoS^۱) است. به این صورت که یک سری معیار به عنوان معیارهای کمی مشخص می‌شود که لازم است این سرویس‌ها به گونه‌ای باشند که این معیارها را تا حد خوبی برآورده سازند. از جمله‌ی این معیارهای کمی می‌توان به تاخیر مربوط به انجام سرویس اشاره کرد. تاخیر هر سرویس خود چند بخش مختلف دارد. در مورد سرویس اطفا حریق می‌توان گفت که ابتدا لازم است که حسگرها اطلاعات پیرامون خود را جمع‌آوری کنند سپس این اطلاعات جمع‌آوری شده به مرکز پردازشی ارسال شود و در آنجا این اطلاعات پردازش شود و نتیجه‌ی اطلاعات به فعال‌کننده ارسال شود و درنهایت آن‌ها شروع به اقدام کنند در تمام این مراحل تاخیرهای مختلف وجود دارد. با توجه به رشد خیلی خوب این روزها در ساختار دستگاه‌های الکترونیکی می‌توان گفت که دستگاه‌های خیلی دقیق و سریعی در بازار موجود است که با کمترین خطای درحال دریافت اطلاعات و همچنین انجام کارهای مربوطه هستند و بحث مربوط به افزایش بازدهی در این دستگاه‌ها در گرایش‌های مربوط به رشته‌ی الکترونیک می‌تواند بحث جذابی باشد، لذا در این پایان نامه از تاخیرهای مربوط به عملکرد حسگرها و فعال‌کننده‌ها صرف‌نظر شده است و فرض براین است که

¹Quality of Service

این دستگاهها به محض دریافت اطلاعات بلا فاصله کار خود را انجام می‌دهند. اما در مورد انواع دیگر تاخیر لازم است که دقت لازم انجام گیرد که در ادامه به تفصیل بیشتر در این مورد صحبت خواهد شد.

۱.۳.۱ انواع تاخیر در شبکه‌های سوئیچ بسته

در این بخش قرار است که انواع مختلف تاخیر در شبکه‌های سوئیچ بسته بررسی شود. در این شبکه‌ها یک بسته ارسالی، سفر خود را از یک میزبان (مبدأ) شروع می‌کند، از تعدادی سوئیچ و مسیریاب می‌گذرد و در پایان سفر خود، به میزبان دیگر (مقصد) می‌رسد. با حرکت بسته از یک گره^۱ (میزبان یا مسیریاب) به گره دیگر در طول مسیر، در هر گره انواع مختلفی از تاخیر می‌تواند برای بسته در حال ارسال اتفاق بیفتد. این تاخیرها عبارتند از

۱. تاخیر پردازش: مدت زمانی است که طول می‌کشد یک گره، بسته دریافتی را تحلیل کند و بفهمد که آیا لازم است آن را به سمت یک گره دیگر هدایت کند و یا اینکه خودش آن را پردازش کند. همچنین اگر زمانی صرف بررسی سالم بودن بیت‌های موجود در بسته شود این زمان جزء این دسته از تاخیر قرار می‌گیرد.

۲. تاخیر صفت: تاخیری است که بسته‌ها در صفت ورودی منتظر می‌شوند تا نوبت پردازش آنها برسد که میزان این تاخیر با توجه به فرمول‌های مربوط به تئوری صفت و نوع صفت و همچنین مدل پردازشی گره محاسبه می‌شود. همچنین در صورتی که گره موجود، یک گره مسیریاب باشد یعنی عملکرد آن به صورتی باشد که بسته‌های دریافتی را به سایر گره‌ها هدایت می‌کند ممکن است در خروجی این گره نیز یک صفت تشکیل شود و بسته در آن صفت خروجی هم مقداری منتظر بماند.

۳. تاخیر انتقال: این تاخیر برابر است با مدت زمان لازم برای انتقال کلیه‌ی بیت‌های موجود در بسته بر روی لینک‌های موجود که همانطور که می‌دانیم در حالت ساده شده با فرض اینکه بسته ارسالی شامل L بیت باشد و همچنین آهنگ انتقال لینک به اندازه R بیت بر ثانیه باشد آنگاه تاخیر انتقال این بیت به اندازه $\frac{L}{R}$ ثانیه خواهد بود.

۴. تاخیر انتشار: به محض آنکه مسیریاب A یک بیت را روی لینک خروجی فرستاد، این بیت باید تا

¹Node

مسیریاب B منتشر شود. زمان لازم برای منتشر شدن این بیت از ابتدای لینک تا مسیریاب B را تاخیر انتشار می‌گویند. سرعت حرکت بیت‌ها روی یک لینک در واقع همان سرعت انتشار امواج الکترومغناطیسی در لینک است، که به نوع رسانه فیزیک مورد استفاده در لینک (فیبر نوری، زوج به هم تابیده، بی‌سیم و ...) بستگی دارد. تاخیر انتشار در یک لینک ارتباطی برابر است با فاصله بین دو مسیریاب (طول لینک که با d نشان می‌دهند) تقسیم بر سرعت انتشار s یعنی $\frac{d}{s}$.

تاخیر انتقال زمان لازم برای بیرون دادن تمامی بیت‌های یک بسته توسط مسیریاب است و به صورت تابعی از طول بسته و آهنگ انتقال به لینک خروجی در مسیریاب تعریف می‌شود؛ در نتیجه هیچ ارتباطی با فاصله بین دو مسیریاب (یا همان طول لینک) ندارد. از طرفی، تاخیر انتشار زمانی است که طول می‌کشد تا یک بیت از یک مسیریاب به مسیریاب بعدی برسد، این تاخیر تابعی از فاصله بین دو مسیریاب و سرعت انتشار امواج الکترومغناطیسی در رسانه فیزیکی لینک می‌باشد و هیچ ارتباطی با طول بسته یا آهنگ انتقال لینک ندارد. معمولاً در مقالات مربوط به شبکه‌های بی‌سیم از تاخیر انتشار در مقابل تاخیر انتقال صرف نظر می‌شود و یا این دو تاخیر را به عنوان یک تاخیر در نظر می‌گیرند. در این پایان‌نامه نیز این دو تاخیر به صورت جدا درنظر گرفته نشده‌اند و عملاً از تاخیر انتشار صرف نظر شده است.

۴.۱ پردازش لبه

گسترش اینترنت اشیاء و موفقیت سرویس‌های ابری باعث ایجاد الگوی جدیدی در پردازش داده‌ها به نام پردازش لبه^۱ شده است. در این الگوی پردازشی سعی بر این است که پردازش داده‌ها در لبه شبکه انجام شود. طبق برآوردهای انجام شده در [۲۴] تعداد دستگاه‌های متصل شده به شبکه ۵۰ میلیارد عدد خواهد بود. بعضی از کاربردهای اینترنت اشیاء نیاز به زمان پاسخ کوتاه دارند، بعضی ممکن است دارای داده‌های محروم‌انه و شخصی باشند و بعضی از این کاربردها می‌توانند بار سنگینی برای شبکه داشته باشند و پردازش ابری ممکن است روش مناسبی برای این کاربردها نباشد.

¹Edge Computing

۱.۴.۱ چرا به پردازش لبه نیاز داریم؟

در حال حاضر نرخ تولید داده در لبه شبکه در حال افزایش می‌باشد. بنابراین پردازش این داده‌ها در لبه شبکه روش کارامدتری خواهد بود. در ادامه دلایلی برای لزوم استفاده از پردازش لبه را بر می‌شماریم [۱]:

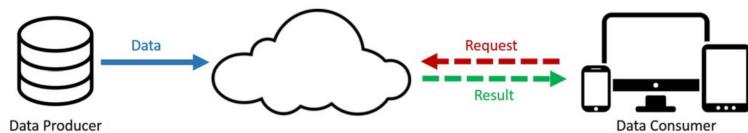
فشار از سمت پردازش ابری: قرار دادن همه وظایف پردازش در ابر به عنوان یک روش کارآمد برای پردازش داده‌ها شناخته شده است چرا که قدرت پردازشی ابر قابل مقایسه با قدرت پردازشی اشیاء در لبه شبکه نیست. با این حال، در مقایسه با پیشرفت سریع سرعت پردازش داده‌ها، پهنای باند شبکه‌ها تقریباً ثابت مانده‌اند. افزایش تولید داده در لبه شبکه باعث شده است که انتقال این داده‌ها به گلوگاه پردازش ابری تبدیل شود. به عنوان مثال یک اتومبیل خودران را در نظر بگیرید. در هر ثانیه ۱ گیگابایت داده توسط آن تولید می‌شود و پردازش بلاذرنگ^۱ این داده‌ها برای تصمیم‌گیری درست لازم است. اگر قرار باشد که همه این داده‌ها برای پردازش به ابر فرستاده شوند، زمان پاسخ بسیار طولانی خواهد بود. علاوه بر این، پهنای باند و قابلیت اطمینان شبکه‌های فعلی برای پردازش داده‌های تعداد زیادی خودرو در یک منطقه کافی نخواهد بود. در این موارد داده‌ها باید در لبه شبکه پردازش شوند. این کار زمان پاسخ کوتاه‌تر، پردازش کارامدتر و فشار کم‌تر بر شبکه را به ارمغان می‌آورد.

کشش از سمت اینترنت اشیاء: انتظار می‌رود که تعداد اشیاء لبه شبکه به میلیارد‌ها دستگاه برسد. در نتیجه داده‌های خام تولید شده توسط آن‌ها حجم بسیار بزرگی خواهند داشت و این حجم بزرگ باعث می‌شود که روش‌های مرسوم پردازش ابری مناسب برای پردازش این حجم از داده‌ها نباشند. در نتیجه می‌توان در نظر گرفت که بیشتر این داده‌های تولید شده در اینترنت اشیاء هیچ وقت برای پردازش به ابر ارسال نمی‌شوند و در لبه شبکه پردازش می‌شوند.

شکل (۱.۱) ساختار مرسوم در پردازش ابری را نشان می‌دهد. تولید کننده‌های داده، داده‌های خام را به ابر انتقال می‌دهند. مصرف کننده‌ها هم با ارسال درخواست نتیجه را از ابر به دست می‌آورند. با این حال این ساختار برای استفاده در اینترنت اشیاء مناسب نیست. اولاً حجم داده‌ی تولید شده در لبه شبکه بسیار زیاد است که پهنای باند شبکه و منابع پردازشی زیادی را طلب می‌کند. ثانیاً حفظ حریم شخصی به عنوان یک مانع برای پردازش ابری ایجاد می‌کند. در انتهای، بسیاری از گره‌های انتهایی در اینترنت

^۱Real-Time

اشیاء دارای انرژی محدودی هستند و مأذول‌های مخابرات بی‌سیم معمولاً مصرف انرژی بالای نیاز دارند. در نتیجه انجام وظایف پردازشی در لبه شبکه می‌تواند از لحاظ مصرف انرژی بهینه باشد.



شکل ۱.۱: الگوی پردازش ابری [۱]

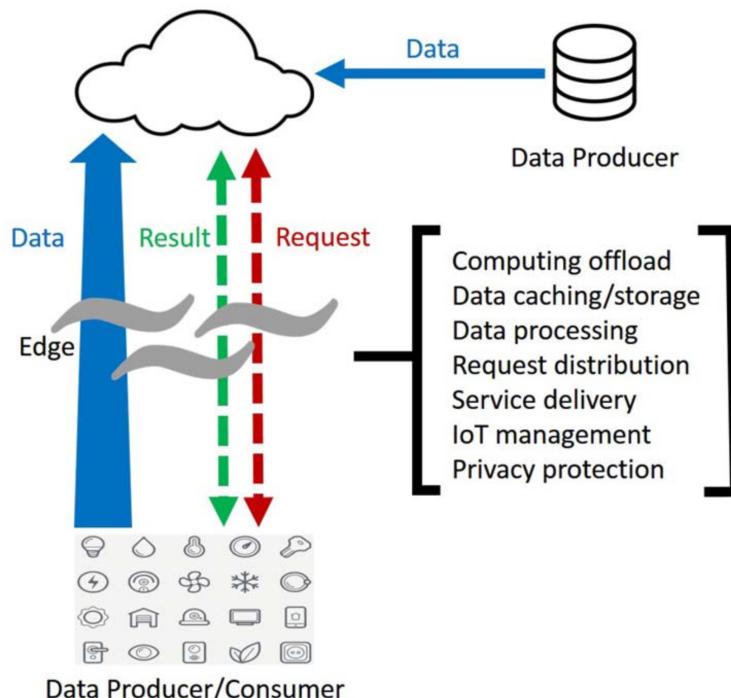
تغییر از مصرف کننده داده به تولید کننده داده: در الگوی پردازش ابری، دستگاه‌های انتهايي در لبه شبکه معمولاً نقش مصرف کننده داده را دارند. به عنوان نمونه می‌توان تماشاي يك ويديو را مثال زد. با اين حال امروزه مردم در حال تولید داده توسط دستگاه‌هاي‌شان هستند. برای مثال امروزه بسیار عادي است که افراد عکس‌ها و ویدیوهایی که توسط خودشان ضبط شده است را در سرویس‌های اینترنتی مختلف به اشتراک بگذارند. در [۲۵] اشاره شده است که در هر دقیقه در twitter ۵۱۱۲۰۰ توییت جدید ارسال می‌شود و در instagram ۵۵۱۴۰ عکس جدید بارگذاری می‌شود. باید توجه داشت که این تصاویر و ویدیوها می‌توانند حجم زیادی داشته باشند و پهناي باند زیادی را برای بارگذاری استفاده کنند. در این موارد ویدیوها باید ابتدا حجمشان در لبه شبکه کاهش پیدا کند تا به وضوح تصویر^۱ مناسب برای بارگذاری در شبکه برسند. به عنوان نمونی دیگر می‌توان دستگاه‌های مربوط به سلامت را مثال زد. داده‌های جمع‌آوری شده توسط این دستگاه‌ها معمولاً خصوصی است و پردازش این داده‌ها در لبه شبکه به جای ارسال داده‌ها برای پردازش به ابر به حفظ حریم خصوصی افراد کمک می‌کند.

۲.۴.۱ پردازش لبه چیست؟

پردازش لبه به همه تکنولوژی‌هایی اطلاق می‌شود که امکان انجام پردازش داده‌ها در لبه شبکه را فراهم می‌کنند. در اینجا منظور از لبه، همه منابع پردازشی و شبکه‌ای است که بین منبع داده‌ها (جایی که داده‌ها در آن جا تولید می‌شوند) و مراکز داده‌ی ابری قرار دارند. برای مثال یک دروازه‌ی شبکه در یک خانه هوشمند، می‌تواند یک منبع پردازشی لبه بین اشیاء و مرکز داده‌ی ابری باشد یا یک مرکز داده‌ی کوچک، می‌تواند یک منبع پردازشی لبه بین دستگاه‌های سیار و ابر در نظر گرفته شود. منطق در پردازش لبه این است که پردازش داده‌ها باید در

^۱Resolution

همسایگی منبع داده‌ها انجام شود. با این منطق پردازش لبه و پردازش مه دو مفهوم یکسان را خواهند داشت با این تفاوت که پردازش لبه تمرکزش سمت اشیاء است ولی پردازش مه تمرکزش سمت زیرساخت است.



شکل ۲.۱: الگوی پردازش لبه [۱]

شکل (۲.۱) جریان پردازش دو طرفه را در پردازش مه نشان می‌دهد. در الگوی پردازش لبه، اشیاء نه تنها مصرف کننده داده بلکه تولید کننده داده نیز هستند. در واقع در لبه، اشیاء نه تنها می‌توانند سرویس و محظوظ از ابر درخواست نمایند بلکه می‌توانند وظایف پردازشی را هم انجام دهند.

۳.۴.۱ مزایای پردازش لبه

هدف از پردازش لبه قراردادن پردازش در مجاورت منبع تولید داده‌ها است. این کار مزایایی نسبت به روش‌های مبتنی بر پردازش ابری مرسوم دارد. در [۲۶] برنامه‌ای برای تشخیص چهره ساخته شده است که با انتقال پردازش از ابر به لبه شبکه، زمان پاسخ برنامه از ۹۰۰ میلی ثانیه کاهش پیدا کرده است. در [۲۷] نویسنده‌گان از واحدهای پردازشی ابری کوچک^۱ برای پردازش وظایف دستیار شناختی پوشیدنی^۲ استفاده کرده

^۱Cloudlet

^۲Wearable Cognitive Assistance

اند و نتایج نشان می‌دهد که زمان پاسخ بین ۸۰ تا ۲۰۰ میلی ثانیه کاهش پیدا کرده است. روش ارائه شده در [۲۸] برای اجرای برنامه‌های تلفن همراه به کمک استفاده همزمان از پردازش ابری و پردازش لبه توanstه زمان اجرا و انرژی مصرفی را تا ۲۰ برابر کاهش دهد.

۴.۴.۱ کاربرد پردازش لبه در شهر هوشمند

ویژگی‌های زیر باعث می‌شوند که پردازش لبه برای استفاده در شهر هوشمند مناسب باشد.

۱. حجم زیاد داده: تخمین زده می‌شود که در سال ۲۰۱۹ میلادی یک شهر با جمعیت ۱ میلیون نفر، ۱۸۰ پتابایت داده در هر روز تولید می‌کند [۲۹]. این داده‌ها توسط امنیت عمومی، سلامت، تجهیزات شهری و حمل و نقل تولید می‌شوند. ساختن مراکز داده‌ی متمرکز برای رسیدگی به همه‌ی این داده‌ها یک راهکار غیر واقعی است چرا که از این حجم از داده نیاز به قدرت پردازشی بسیار زیادی دارد. در این مورد پردازش لبه می‌تواند یک راه حل بهینه برای پردازش این حجم عظیم از داده‌ها باشد.

۲. تاخیر کم: برای کاربردهایی که نیاز به تأخیر قابل پیش‌بینی و کم دارند مانند اورژانس سلامتی و امنیت عمومی، پردازش لبه یک الگوی مناسب است چرا که زمان انتقال داده‌ها را کاهش می‌دهد و ساختار شبکه را ساده‌تر می‌کند.

۳. آگاهی از مکان: برای کاربردهای مبتنی بر موقعیت جغرافیایی مانند حمل و نقل و مدیریت تجهیزات شهری پردازش لبه به دلیل آگاهی از مکان، بهتر از پردازش ابری عمل می‌کند.

۵.۱ پردازش مه

شرکت Cisco برای اولین بار، پردازش مه را در ماه ژانویه سال ۲۰۱۴ معرفی کرد. ایده‌ی پشت پردازش مه نسبت به پردازش ابری، افزایش سرعت پردازش اطلاعات بود. ریشه پردازش مه، همان پردازش ابری است، با این تفاوت که به دلیل سرعت بالاتر انتقال اطلاعات، در فناوری‌هایی مثل اینترنت اشیا مورد استفاده قرار می‌گیرد.

هم‌زمان با رشد و نفوذ دستگاه‌های اینترنت اشیا در حوزه‌های مختلف، استفاده از خدمات پردازش مه نیز در این حوزه اهمیت بیشتری یافت. پیش‌بینی می‌شود فناوری پردازش مه که محاسبات مه هم گفته می‌شود،

در آینده رشد قابل توجهی داشته باشد. گزارشی از ۴۵۱ تحقیق انجام شده در این زمینه نشان داد که بازار پردازش مه تا سال ۲۰۲۲ به ارزشی برابر ۴/۶ میلیارد دلار خواهد رسید.

۱.۵.۱ عملکرد پردازش مه چگونه است؟

دستگاههایی که در مه وجود دارند تحت عنوان گره شناخته می‌شوند. هر دستگاه با ارتباط شبکه‌ای، محاسباتی و ذخیره‌سازی می‌تواند یک گره باشد که در هر جایی با یک ارتباط شبکه‌ای می‌تواند قرار گیرند. دستگاههای مختلف از کنترل‌کننده‌ها تا مسیریاب‌ها و دوربین‌های فیلمبرداری، می‌توانند به عنوان یک گره مه عمل کنند. این گره‌ها می‌توانند در مناطق هدف مانند دفتر کار یا در یک وسیله نقلیه به کار گرفته شوند. وقتی یک دستگاه اینترنت اشیا داده‌هایی تولید می‌کند، این داده‌ها می‌توانند از طریق یکی از این گره‌ها دریافت شوند و در شبکه، پردازش سپس به مراکز داده ابری منتقل شوند.

تفاوت اصلی میان پردازش ابری و پردازش مه این است که محاسبات مه نزدیکی جغرافیایی بیشتری به کاربرنهایی دارد و توزیع جغرافیایی وسیع‌تری ایجاد می‌کند.

۲.۵.۱ مزایای پردازش مه

دلایل مختلفی برای به کارگیری پردازش مه وجود دارد. این دلایل در نهایت باعث افزایش بهره‌وری سازمانی می‌شوند. در ادامه به برخی از مهم‌ترین مزایای به کارگیری پردازش مه می‌پردازیم.

- کاهش زمان تاخیر: یکی از بزرگ‌ترین مزایای پردازش مه، کاهش زمان تاخیر است. دیگر لازم نیست داده‌ها برای پردازش به مراکز داده ابری فرستاده شوند و از بین رفتان این مشکل باعث می‌شود که تحلیل و پردازش داده‌ها بسیار بهتر و موثرتر انجام شود.
- کاهش ملزمومات عملکردی: عدم ارسال داده‌ها به مراکز داده پردازش ابری علاوه بر صرفه‌جویی در زمان، می‌تواند میزان پهنای باند لازم برای این کار را هم کاهش دهد و در مقابل، این میزان پهنای باند برای ارتباط با حسگرها و مراکز داده ابری بکار برده شود. این روش در نهایت باعث کاهش ملزمومات عملکرد می‌شود.
- توزیع جغرافیایی گسترده: استفاده از پردازش مه با تمرکز زدایی از شبکه، امکان توزیع جغرافیایی

گستردگی را نسبت به شبکه‌سازی سنتی یا پردازش ابری فراهم می‌آورد. این کار به ارایه سرویس با کیفیت‌تر برای کاربر نهایی منجر می‌شود.

- تحلیل در لحظه: در بسیاری از محیط‌ها، توانایی تحلیل فوری داده‌ها، اهمیت بسیار زیادی دارد. حذف عوامل ناکارآمدی و تاخیر که در سرویس‌های خدمات ابری وجود دارند، به معنای آن است که کاربر می‌تواند تحلیل معتبر و لحظه‌ای داده‌ها را در اختیار داشته باشد.

۶.۱ انواع لایه در شبکه

با توجه به مطالب گفته شده می‌توان گفت که لایه‌های شبکه را می‌توان به دو دسته تقسیم کرد. دسته‌ی اول لایه‌های مربوط به تولید و جمع‌آوری اطلاعات و دسته‌ی دوم لایه‌های مربوط به پردازش اطلاعات. در این پایان‌نامه لایه‌های مربوط به دسته‌ی اول را یک لایه در نظر می‌گیریم که شامل کلیه‌ی حسگرها و فعال‌کننده‌هاست. در مورد دسته دوم سه لایه‌ی لبه، مه و ابری را در نظر می‌گیریم. که به ترتیب فاصله از لایه‌ی حسگر قرار گرفته‌اند.

در مورد تفاوت بین لایه لبه و لایه مه علاوه بر بحث تفاوت در فاصله و میزان تاخیر می‌توان گفت که

ظرفیت پردازشی در لایه‌ی مه از لایه‌ی لبه بیشتر است. همچنین در این پایان‌نامه یک هزینه پردازشی برای گره‌های محاسباتی در نظر گرفته شده‌است به این صورت که به صورت میانگین هزینه‌ی پردازشی در گره‌های لبه از دولایه دیگر بیشتر است و گره‌های موجود در لایه‌ی ابری ارزان‌ترین هزینه پردازشی را دارند.

در بعضی از تعاریف لایه‌ی لبه را یک مجازی بر روی لایه‌ی حسگر در نظر می‌گیرند به این صورت که اطلاعات حسگرها فراتر از لایه‌ی آن‌ها نمی‌رود و در همان لایه پردازش می‌شود مزیتی که این فرض دارد جنبه‌ی امنیت و محافظت از اطلاعات است. اما در این پایان‌نامه لایه‌ی لبه را جدا از لایه‌ی جسگر در نظر گرفته‌ایم. حال صورت مسئله‌ای که ایجاد می‌شود این است که با توجه به حق انتخاب‌های موجود و با توجه به اطلاعات حسگرها که لازم است پردازش شوند مناسب ترین مکان برای پردازش این اطلاعات کجاست.

انتخاب مکان مناسب برای پردازش بستگی به تعریف دقیق صورت مسئله و همچنین تعریف دقیق از کیفیت سرویس‌ها دارد. به عنوان مثال می‌توان هدف را این‌گونه در نظر گرفت که شبکه به صورتی کار کند که اطلاعات حسگرها در کمترین زمان ممکن پردازش شود که در این صورت احتمالاً قیدی بر روی هزینه‌ی لازم برای پردازش اطلاعات لازم است که در نظر گرفته شود که در نتیجه‌ی آن، نمی‌توان گفت که شبکه در

ارزانترین حالت خود به سر می‌برد. در یک حالت دیگر می‌توان صورت مسئله را به‌گونه‌ای طراحی کرد که هزینه‌های کلی شبکه در بهینه‌ترین حالت ممکن باشد. در این صورت لازم است که شرایط مربوط به تاخیر به عنوان قید به صورت مسئله اضافه شوند. در هر دو حالت مدلسازی قبل این بحث به وجود می‌آید که آیا شبکه توان پردازش کلیه‌ی اطلاعات حاصل از لایه‌ی حسگرها را دارد یا نه، که لازم است این قسمت نیز به عنوان قید به مسئله اصلی اضافه شود. در حالتی دیگر می‌توان صورت مسئله را به این صورت تعریف کرد که تعداد پردازش‌ها بیشترین مقدار ممکن باشد.

۷.۱ نوآوری‌های پایان نامه

دستاوردهای این پایان نامه را می‌توان به صورت زیر خلاصه کرد: در بخش اول، صورت مسئله‌ی یافتن مکان مناسب برای پردازش اطلاعات با هدف بهینه کردن هزینه‌های کل شبکه مدل‌سازی شده است و مسئله به صورت یک مسئله بهینه‌سازی غیرخطی ترکیبی عدد صحیح نوشته شده است. ابتدا مسئله خطی‌سازی شده و به صورت مرکزی حل شده است. سپس یک راه حل غیرمت مرکز ارائه شده است.

در بخش دوم، ابتدا یک راه حل اکتشافی برای صورت مسئله غیرخطی نوشته شده است. در ادامه صورت مسئله خطی شده به صورت توزیع شده حل می‌شود.

۸.۱ ساختار پایان نامه

ساختار پایان نامه به شرح زیر است.

فصل (۲) به مروری بر مطالعات انجام شده در زمینه تخصیص منابع اینترنت اشیاء می‌پردازد.

فصل (۳) به بررسی مسئله اختصاص منابع پردازشی با توجه به مدل ارائه شده می‌پردازد. در این فصل مسئله بهینه‌سازی همراه با تمام قیدهای موجود نوشته می‌شود، درنهایت مسئله خطی‌سازی می‌شود، به صورتی که که حل کردن آن راحت‌تر باشد. در ادامه مسئله به صورت غیرمت مرکز بازنویسی می‌شود و یک الگوریتم غیرمت مرکز ارائه می‌شود.

در فصل (۴) مسئله تخصیص منابع پردازشی به دو صورت دیگر مورد بررسی قرار گرفته است. ابتدا یک

الگوریتم اکتشافی مبتنی بر روش ویتربی^۱ ارائه می شود، مزیت این روش این است که نیازی نیست خطی سازی انجام شود و از اولین فرم مسئله که غیرخطی است، می توان استفاده کرد. در ادامه یک الگوریتم توزیع شده ارائه می شود و مجددا مسئله توسط آن الگوریتم حل و بررسی می شود.
در انتها در فصل (۵) به بیان نتیجه گیری و کارهای آینده می پردازیم.

¹Viterbi

فصل ۲

مروری بر مطالعات انجام شده

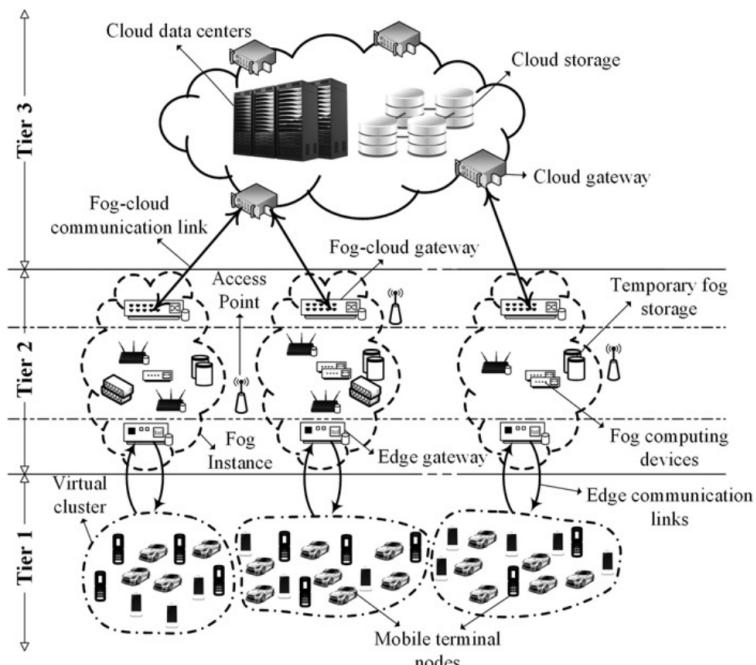
در این فصل ابتدا مدل سیستم این پایان نامه را معرفی می‌کنیم و سپس مقالاتی را که به تخصیص منابع در شبکه اینترنت اشیاء پرداخته‌اند بررسی می‌کنیم.

در این پایان نامه فرض می‌کنیم که تعدادی وظیفه وجود دارند. هر وظیفه، مشخصات معین مربوط به خود را دارد و لازم است اطلاعات مربوط به این وظیفه پردازش شود. برای پردازش داده‌های وظیفه‌ها باید منابع پردازشی به آن‌ها اختصاص پیدا کنند. مسئله تخصیص منابع در این پایان نامه به صورت یک مسئله بهینه سازی فرمول بندی شده است که تابع هدف آن کمینه کردن مجموع هزینه منابع پردازشی می‌باشد. در این تابع هدف، هزینه هر گره، با توجه به نوع و تعداد وظیفه‌هایی که پردازش می‌کند و با توجه به نوع گره مدل‌سازی می‌شود.

۱۰.۲ تخصیص منابع پردازشی در شبکه اینترنت اشیاء

مقاله [۲]، یک فرمول بندی ریاضی برای پردازش مه ارائه می‌دهد. برای این کار اجزاء مختلف را تعریف می‌کند و به مطالعه تفاوت‌های پردازش مه با پردازش ابری مرسوم می‌پردازد. مدل سیستم این مقاله در شکل (۱.۲) رسم شده است. همانطور که از این شکل مشخص است، در این مقاله سیستم از سه لایه تشکیل شده است. در لایه اول دستگاه‌های اینترنت اشیاء قرار دارند که وظیفه جمع‌آوری داده‌های حسگرها و واقعی و ارسال داده‌های خام به گره بعدی را دارند. لایه دوم که به عنوان لایه مه معرفی شده است از دستگاه‌هایی مانند

مسیریاب‌ها و نقاط دسترسی^۱ تشکیل شده است که می‌توانند داده‌ها را پردازش و به صورت موقت ذخیره کنند. این دستگاه‌ها به چهارچوب‌های ابری متصل هستند و مسئول ارسال اطلاعات به صورت متناوب به ابر می‌باشد. در لایه سوم پردازش ابری قرار دارد که بالاترین لایه می‌باشد و شامل تعدادی منبع پردازشی با ظرفیت بالا می‌باشد که می‌توانند حجم زیادی از داده‌ها را پردازش و ذخیره سازی کنند. با بررسی انجام شده، پردازش مه به کمک پردازش ابری می‌تواند پاسخ گوی نیاز نسل بعدی کاربردهای اینترنت اشیاء باشد. نتایج مورد مطالعه نشان می‌دهند که زمانی که ۲۵٪ کاربردهای اینترنت اشیاء نیاز به سرویس‌های بلاذرنگ با تاخیر کم دارند، استفاده از پردازش مه باعث کاهش ۴۰/۴۸٪ در انرژی مصرفی می‌شود. با این حال این مقاله هیچ استراتژی بهینه سازی معرفی نکرده است.

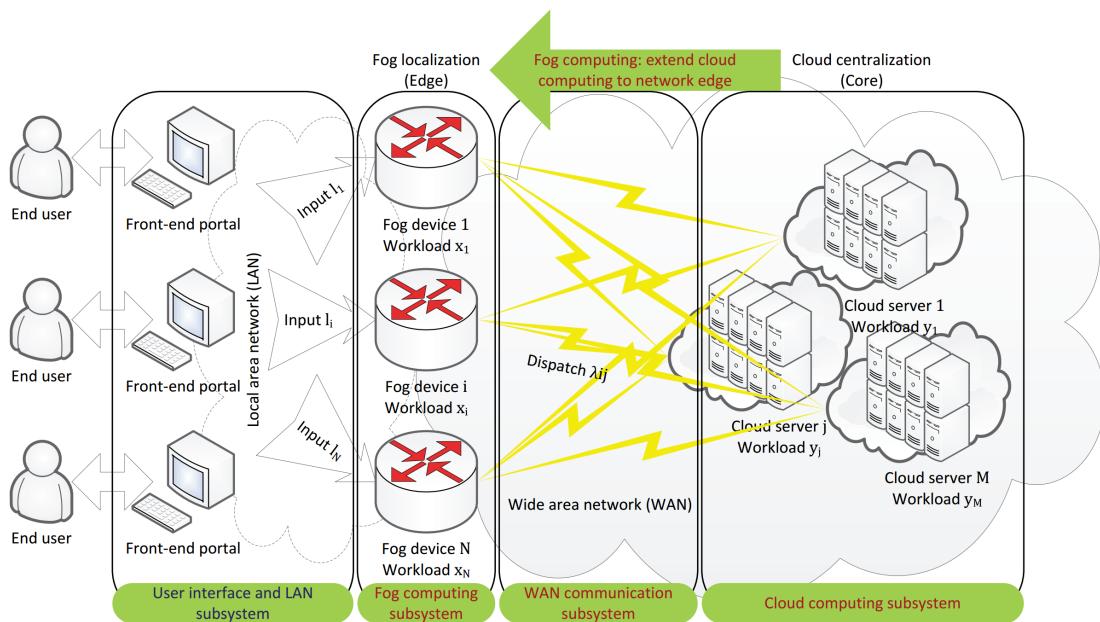


[۲] : مدل سیستم مقاله [۲]

نویسنده‌گان در [۳] چهارچوبی پیاده سازی کرده‌اند که در اختصاص منابع پردازشی در یک شبکه ابری و مه، تاخیر انتقال و مصرف انرژی را بهینه می‌کند. در مدل آن‌ها، لایه‌ی پردازش مه، بین لایه کاربران و لایه ابر قرار می‌گیرد و به پردازش ابری کمک می‌کند تا سرویس‌هایی با تاخیر کمتر و نرخ پردازش بیشتر به کاربران ارائه دهد. در این چهارچوب، بررسی اثر متقابل و همکاری پردازش ابری و پردازش لبه از اهمیت خاصی برخوردار

^۱Access Point

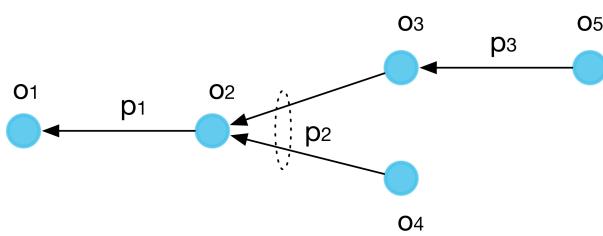
است. نویسنده‌گان تخصیص منابع را به صورت یک مسئله بهینه سازی فرمول‌بندی کرده‌اند که مصرف انرژی سرویس‌ها را با محدودیتی در تاخیر آن‌ها بهینه می‌کند. برای حل تقریبی مسئله بهینه سازی، مسئله اصلی به سه زیر مسئله‌ی مربوط به هر زیر سیستم تفکیک شده که هر کدام قابل حل می‌باشد. شکل (۲.۲) سیستم مدل این مقاله را با چهار زیر سیستم نشان می‌دهد. در انتها مبتنی بر نتایج شبیه‌سازی، نویسنده‌گان نتیجه گرفته‌اند که با کاهش استفاده از بخشی از منابع پردازشی برای کاهش پهنای‌باند مورد نیاز و کاهش تاخیر انتقال، پردازش لبه به صورت چشمگیری کارایی پردازشی ابری را افزایش می‌دهد.



شکل ۲.۲: مدل سیستم مقاله [۳] با چهار زیر سیستم

در [۴] نویسنده‌گان مسئله توزیع پردازش سرویس‌ها را به صورت مسئله بهینه سازی برای کمینه کردن هزینه فرمول‌بندی کرده‌اند که با استفاده از برنامه ریزی خطی قابل حل می‌باشد. تمرکز آن‌ها روی مصرف انرژی است که امروزه بخشنده شبکه‌ها و فراهم‌کننده‌های ابری را تشکیل می‌دهد. آن‌ها ابتدا یک مدل ریاضی برای شبکه اینترنت اشیاء ابری ارائه می‌دهند که ظرفیت، کارایی و قابلیت اطمینان حسگرهای، منابع پردازشی و منابع شبکه را در محدوده دستگاه‌های انتهایی، دستگاه‌های دسترسی به شبکه و لایه ابری در نظر می‌گیرند. سپس هر سرویس اینترنت اشیاء را با یک گراف ریشه‌دار جهت دار مدل می‌کنند که رابطه‌ی بین تابع‌هایی که باید روی اطلاعات سرویس اجرا شوند تا نتیجه نهایی مورد دلخواه کاربر بحسب باید را در خود نگه می‌دارد. شکل (۳.۲) نمونه‌ای از این گراف را نشان می‌دهد که سه تابع p_1 , p_2 و p_3 باید روی داده‌های ورودی ۰۵ و

۰۴ اجرا بشوند تا نتیجه یا همان o_1 تولید بشود. پس از آن مسئله توزیع سرویس‌های اینترنت اشیاء را برای پیدا کردن مکان اجرا شدن تابع‌های سرویس‌های اینترنت اشیاء و مسیریابی داده‌های سرویس‌ها در شبکه اینترنت اشیاء معرفی می‌کنند که هدف آن کمینه کردن هزینه با در نظر گرفتن نیاز کاربران است. آن‌ها شبکه را به صورت یک گراف مدل می‌کنند که گره‌ها منابع پردازشی و ذخیره سازی هستند و هدف پیدا کردن گره‌های مناسب برای انجام گره‌های گراف سرویس‌ها است به طوری که محدودیت‌های پردازش و انتقال گره‌ها برقرار باشد. در انتهای به بررسی مدل پیشنهادی و تاثیر پارامترهای مختلف پرداخته‌اند. نتایج نشان می‌دهند که مدل بهینه‌سازی و راه حل ارائه شده استفاده قابل انعطاف از شبکه اینترنت اشیاء را برای میزبانی، مدیریت و بهینه کردن طیف وسیعی از سرویس‌های اینترنت اشیاء فراهم می‌کند. در مقایسه با راه حل‌های مرسوم مبتنی بر ابر، مدل ارائه شده باعث کاهش مصرف انرژی تا ۸۰٪ در کنار کاهش تاخیر سرویس‌ها می‌شود.



شکل ۳.۲: یک گراف سرویس در [۴]

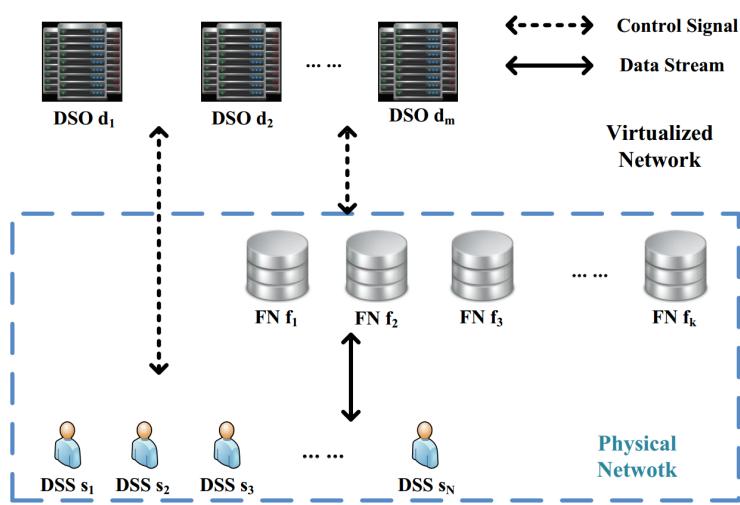
[۵] به بررسی تخصیص منابع پردازشی در یک شبکه پردازش مه می‌پردازد. در این مقاله شبکه با سه لایه مدل می‌شود که شامل لایه‌ی گره‌های مه، لایه‌ی اپراتورهای سرویس داده^۱ و لایه مشتریان سرویس‌های داده^۲ است. مدل سیستم ارائه شده در این مقاله در شکل (۴.۲) نشان داده شده است. هر اپراتور سرویس داده، مجموعه‌ای از گره‌های مه را کنترل می‌کند و گره‌های مه هم سرویس‌های داده لازم را برای مشتریان فراهم می‌کنند. نحوه تخصیص منابع پردازشی محدود گره‌های مه به مشتریان سرویس‌های داده برای رسیدن به حالت بهینه و پایدار، یک مسئله مهم معرفی شده است. آن‌ها یک چهارچوب بهینه سازی مشترک بین همه‌ی اپراتورهای سرویس‌های داده، گره‌های مه و مشتریان سرویس‌های داده ارائه می‌دهند تا به یک تخصیص منابع بهینه به صورت توزیع شده برسند. در این چهارچوب آن‌ها ابتدا از بازی استکلبرگ^۳ برای آنالیز مسئله قیمت گذاری برای اپراتورهای سرویس داده و مسئله تخصیص منابع برای مشتریان سرویس‌های داده استفاده می‌کنند.

¹Data Service Operator (DSO)

²Data Service Subscribers (DSS)

³Stackelberg Game

در سناریویی که اپراتورهای مراکز داده، میزان منابع پردازشی که مشتریان سرویس‌های داده می‌خرند را بدانند از تئوری تطبیق^۱ برای پیدا کردن ارتباط گره‌های مه و اپراتورهای سرویس‌های داده استفاده شده است. در انتها در بین گره‌های مه که اپراتور سرویس‌های داده مشترکی دارند دوباره از تئوری تطبیق برای پیدا کردن ارتباط گره‌های مه و مشتریان سرویس‌های داده استفاده شده است. نتایج شبیه سازی نشان می‌دهند که چهارچوب ارائه شده، کارایی شبکه اینترنت اشیاء را به میزان چشمگیری افزایش می‌دهد.

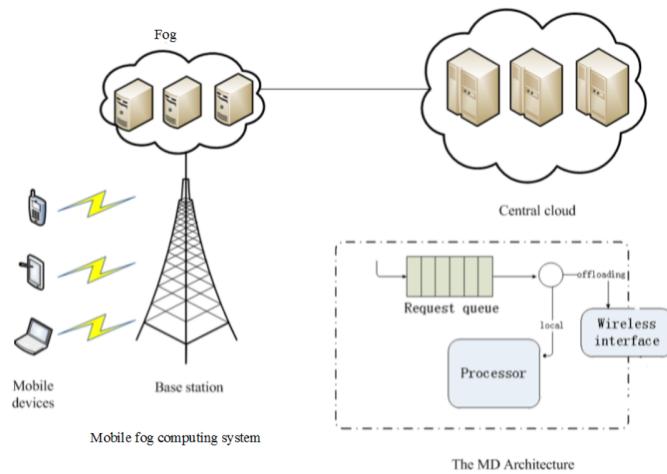


شکل ۴.۲: مدل سیستم مقاله [۵]

در [۶] از پردازش مه برای کاهش بار پردازشی مراکز داده استفاده شده است. مسئله به صورت یک بهینه‌سازی فرمول‌بندی شده است که تلاش می‌کند، تأخیر، مصرف انرژی و هزینه را کمینه کند. در این مقاله، برای بررسی کامل مصرف انرژی، تاخیر و هزینه‌ی پرداختی از تئوری صف استفاده شده است. سه مدل مختلف صف برای دستگاه‌های لبه شبکه، گره‌های مه و مراکز داده ابری استفاده شده است و نرخ داده‌ها و مصرف انرژی اتصالات بیسیم مورد بررسی قرار گرفته‌اند. برای این منظور یک سیستم پردازش ابری مبتنی بر مه مورد بررسی قرار گرفته است و برای بررسی دقیق مصرف انرژی و تاخیر مدل‌های مختلف صف به اجزاء شبکه اعمال شده است. به صورت خاص، اتصالات بیسیم و منابع پردازشی برای مدل کردن مصرف انرژی، تاخیر و هزینه پرداختی مدنظر قرار گرفته‌اند. همانطور که در شکل (۵.۲) مشخص است، در خواست‌ها برای هر کدام از منابع در صف قرار گرفته و توسط منبع مربوطه پردازش می‌شوند. پس از تبدیل مسئله به یک مسئله

^۱Matching Theory

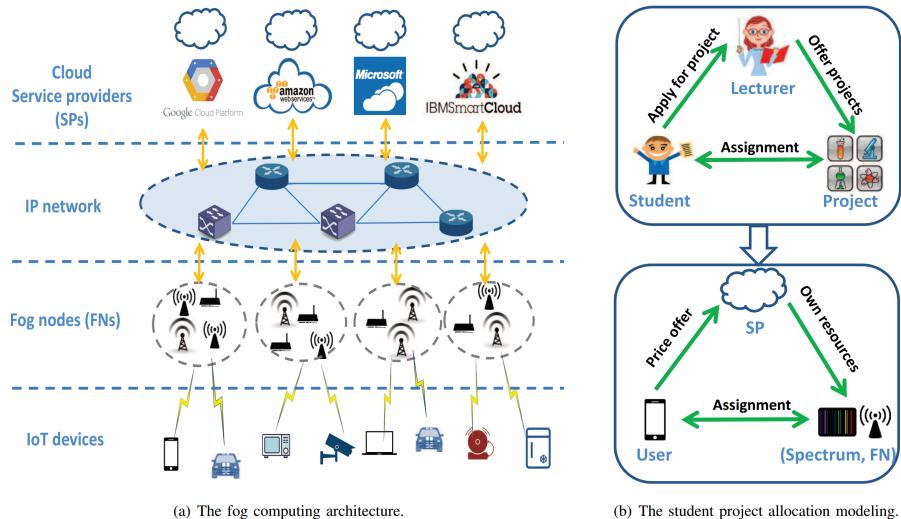
بهینه سازی با یک تابع هدف، به بررسی نتایج شبیه سازی پرداخته شده است. این شبیه سازی ها نشان می دهند که راه حل ارائه شده، کارایی مناسبی دارد و نسبت به راه حل های ارائه شده توسط دیگران مناسب تر است.



شکل ۵.۲: مدل سیستم مقاله [۶]

نویسندهای در [۷] مسئله‌ی تخصیص منابع رادیویی و پردازشی را بررسی کردند. قسمت اول شکل (۶.۲) مدل سیستم استفاده شده در این مقاله را نشان می‌دهد. در این مقاله تخصیص منابع پردازشی و رادیویی به صورت همزمان مورد مطالعه قرار گرفته است تا کارایی سیستم را بهینه کند و رضایت کاربران را افزایش دهد. عوامل مهمی مانند تاخیر، کیفیت ارتباطات و نیاز کاربران در نظر گرفته شده است. در این مقاله کاربران نیاز خود را در قالب تاخیر و اندازه داده‌ها به فراهم کنندگان ابری اعلام می‌کنند. فراهم کنندگان ابری با ارتباط با کاربران سعی در پیدا کردن گره‌های مه مناسب می‌کنند تا پردازش‌های مورد نیاز کاربران را به همراه تخصیص طیف رادیویی به آن‌ها انتقال بدهند. برای بیشینه کردن رضایت کاربران، مسئله به صورت یک مسئله برنامه‌ریزی غیرخطی عدد صحیح مخلوط فرمول بندی شده است. در فرمول بندی محدودیت‌هایی مانند تاخیر سرویس‌ها، کیفیت انتقال، کنترل توان و غیره در نظر گرفته شده است. از بازی تخصیص پروژه به دانش آموzan در تئوری تطبیق برای پیدا کردن تخصیص منابع بهینه استفاده شده است. در این بازی، تعدادی پروژه وجود دارد و دانش آموzan می‌توانند لیستی از پروژه‌های مورد علاقه خود را انتخاب کنند. در نهایت مربی پروژه‌ها را به دانش آموzan اختصاص می‌دهد به طوری که رضایت همه دانش آموzan از پروژه اختصاص یافته به خود بیشینه باشد. همانطور که در قسمت دوم شکل (۶.۲) مشخص است در اینجا فراهم کنندگان ابری به عنوان مربی و منابع پردازشی و رادیویی به عنوان پروژه‌ها و کاربران به عنوان دانش آموzan مدل شده‌اند.

با ارائه استراتژی‌هایی، تاثیر انتخاب بازیگران بر یکدیگر حذف و رسیدن به یک تعادل پایدار تضمین شده و کارایی سیستم بهبود یافته است. نتایج شبیه‌سازی‌های ارائه شده هم نمایانگر کارایی نزدیک به بهینه از دید کاربران و سیستم می‌باشد.



شکل ۶.۲: مدل سیستم مقاله [۷]

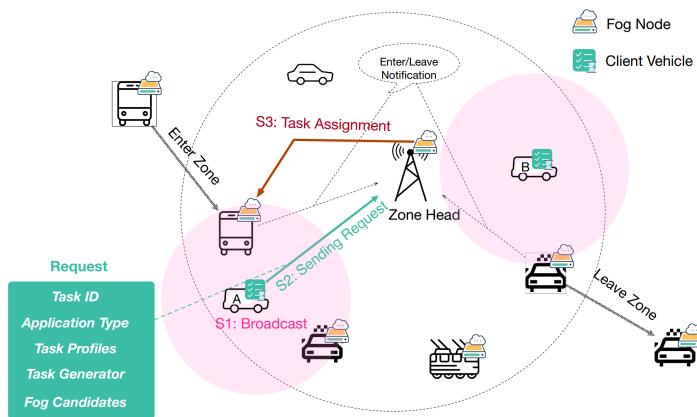
مقاله [۳۰] به بررسی مسئله تخصیص منابع پردازشی به همراه توان ارسالی و پهنه‌باند رادیویی می‌پردازد به طوری که عدالت بین کاربران و حداکثر تاخیر قابل تحمل رعایت شود. این مسئله بهینه سازی به صورت کمینه کردن بیشینه هزینه وزن دار فرمول بندی شده است و هزینه شامل تاخیر و انرژی مصرفی بین همه کاربران است. بهینه سازی ارائه شده یک مسئله برنامه‌ریزی غیرخطی عدد صحیح مخلوط است که به سختی قابل حل است و برای حل آن یک الگوریتم با پیچیدگی کم ارائه شده است. برای این کار مسئله ابتدا به یک مسئله QCQP^۱ تبدیل می‌شود. سپس با استفاده از برنامه‌ریزی کسری^۲ مسئله به یک بهینه سازی محدب تبدیل می‌شود. نتایج شبیه‌سازی‌های ارائه شده نشان دهنده مناسب بودن همگرایی الگوریتم ارائه شده و عدالت بین کاربران و کارایی در زمینه تاخیر و انرژی مصرفی حاصل از آن در برابر الگوریتم‌های موجود می‌باشد.

در مقاله [۸] نویسنده‌گان به ارائه چهارچوبی برای بهینه سازی تاخیر و کیفیت تخصیص وظایف پردازشی در پردازش مه خودروها می‌پردازند. چهارچوب ارائه شده، تحرک خودروها که شامل خودروهایی که نیاز به پردازش وظایف دارند و خودروهایی که به عنوان گره‌های مه عمل می‌کنند را شامل می‌شود. مدل سیستم این

¹Quadratically Constrained Quadratic Programming

²Fractional Programming

مقاله در شکل (۷.۲) نشان داده شده است. خودروهای دارای وظیفه‌ی پردازشی و خودروهای گرهای مه در شکل مشخص هستند. مسئله تخصیص وظایف به صورت یک مسئله بهینه سازی معرفی شده و تابع هدف آن شامل تاخیر و کیفیت است. مانند بقیه مقاله‌ها، مسئله ارائه شده در این مقاله هم نیاز به زمان زیادی برای حل دارد و به همین دلیل مقاله یک الگوریتم مبتنی بر برنامه ریزی خطی برای حل آن ارائه کرده است. نتایج شبیه سازی‌های ارائه شده در این مقاله نشان می‌دهند که چهارچوب ارائه شده نسبت به روش تخصیص وظایف تصادفی، تاخیر را تا ۲۷٪ و کیفیت را تا ۵۶٪ بهبود می‌بخشد.



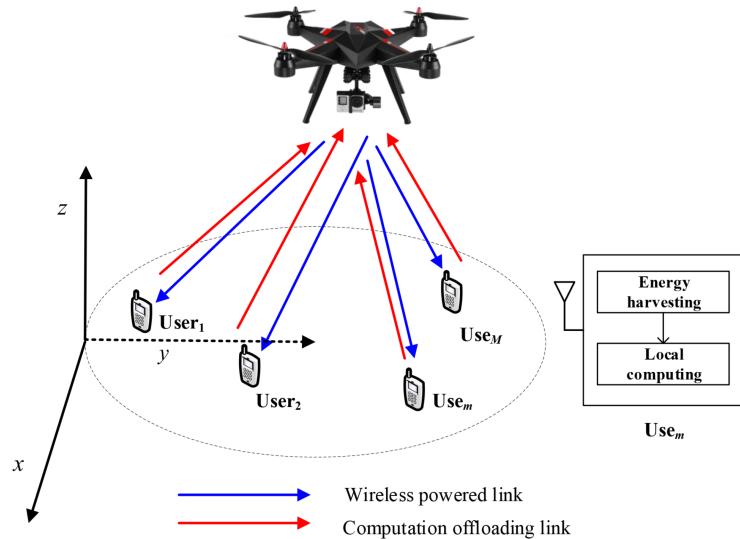
شکل ۷.۲: مدل سیستم مقاله [۸]

مقاله [۹] به استفاده از هوایپیماهای بدون سرنشین^۱ در پردازش لبه می‌پردازد. شکل (۸.۲) مدل سیستم این مقاله را نشان می‌دهد. در این مقاله کاربران باید تصمیم بگیرند که پردازش اطلاعات توسط خودشان انجام شود یا از هوایپیمای بدون سرنشین که پردازنده قدرتمندی دارد استفاده کنند. مسئله بیشینه کردن نرخ پردازش در دو حالت کلی و جزئی با محدودیت‌های انرژی و سرعت حرکت این پرنده‌ها در نظر گرفته شده است. مانند باقی مقاله‌ها برای حل مسئله در زمان مناسب، یک الگوریتم دو مرحله‌ای و یک الگوریتم سه مرحله‌ای ارائه شده است. نتایج شبیه‌سازی‌های ارائه شده نشان می‌دهند که روش تخصیص منابع ارائه شده بهبود قابل توجهی نسبت به سایر روش‌ها دارد. همچنین پیچیدگی محاسباتی و همگرایی سریع از دیگر مزایای روش ارائه شده می‌باشند.

در [۳۱] نویسنده‌گان یک روش تخصیص منابع مبتنی بر یادگیری تقویتی عمیق^۲ ارائه داده‌اند. الگوریتم ارائه شده شامل مسیریابی (تخصیص منابع شبکه) و تخصیص منابع پردازشی در شبکه پردازش لبه متحرک است.

¹Unmanned Aerial Vehicle

²Deep Reinforcement Learning



شکل ۲: مدل سیستم مقاله [۹]

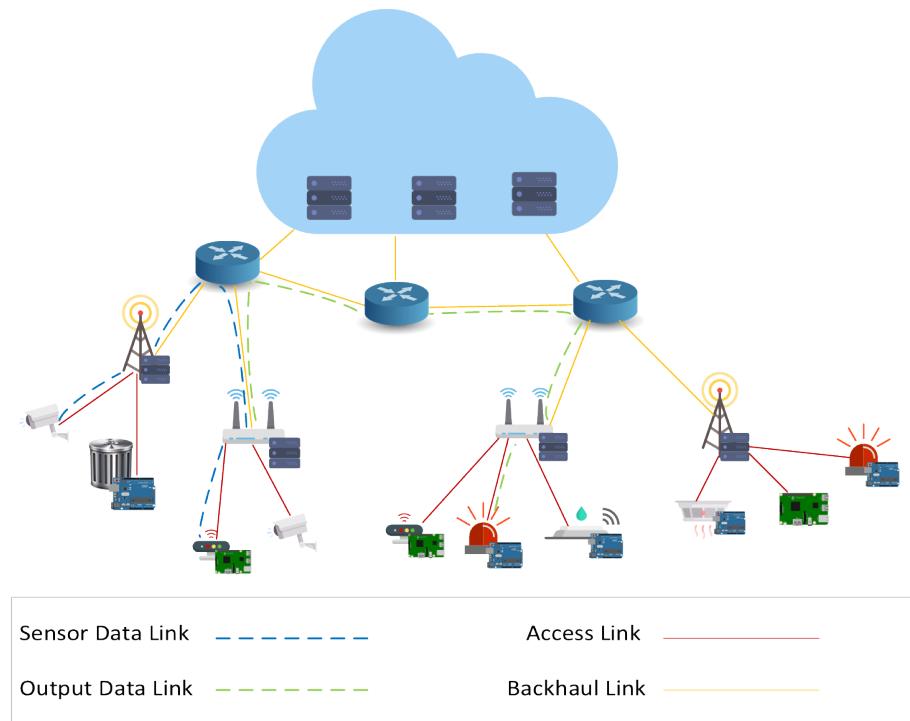
با اعمال تکنولوژی‌های شبکه‌های نرم‌افزاری در مدل ارائه شده، از مزایای آن مانند کنترل متمرکز زیرساخت شبکه بهره‌مند شده‌اند. در این مقاله تاخیر مسیریابی شبکه و تاخیر پردازشی به عنوان تاخیر سرویس در نظر گرفته شده‌اند و هدف مقاله کمینه کردن تاخیر به همراه ایجاد تعادل در استفاده از منابع پردازشی و منابع شبکه است. نتایج شبیه‌سازی‌های بررسی شده نشان می‌دهند که راه حل ارائه شده بهتر از روش‌های که قبلاً برای این مسئله ارائه شده‌اند عمل می‌کند.

فصل ۳

تخصیص منابع پردازشی در شبکه اینترنت اشیاء به صورت مرکز و غیر مرکز

۱.۳ مقدمه

در این فصل ابتدا مدل سیستم مربوط به تخصیص منابع پردازشی در شبکه اینترنت اشیاء را شرح می‌دهیم، سپس مدلسازی ریاضی مربوطه را ارائه می‌دهیم. در این نوع تخصیص منابع، تعدادی وظیفه^۱ و منبع پردازشی داریم که وظیفه‌ها باید برای پردازش داده‌های جمع‌آوری شده توسط حسگرهای خود به این منابع پردازشی ارسال شوند و در آن‌جا پردازش شوند. پس از معرفی و فرمول‌بندی مسئله، چالش‌ها و دلایل پیچیدگی حل این مسئله را بررسی می‌کنیم.



شکل ۱.۳ : دید کلی از مدل سیستم فصل (۳)

۲.۳ مدل سیستم

شکل (۱.۳) مدل سیستم این فصل را نشان می‌دهد. همانطور که دیده می‌شود مدل سیستم از چهار لایه‌ی اصلی تشکیل می‌شود که از پایین به بالا عبارتند از لایه حسگر^۱، لبه^۲، مه^۳ و ابر^۴ که در شکل (۱.۳) این لایه‌ها دیده می‌شوند. در این شکل گره‌های لایه سوم می‌توانند نقش هدایتگری نیز داشته باشند. در پایین‌ترین لایه حسگرها^۵ و عملگرها^۶ وجود دارند. حسگرها موجود در این لایه مدام درحال جمع‌آوری اطلاعات از محیط پیرامون خود هستند. اطلاعات جمع‌آوری شده توسط این گره‌ها به صورت مجموعه‌ای از بسته‌های اطلاعاتی تحت عنوان وظیفه درنظر گرفته می‌شوند. لازم است که اطلاعات موجود در کلیه وظیفه‌ها پردازش شود.

¹Task

²Sensor

³Edge

⁴Fog

⁵Cloud

⁶Sensors

⁶Actuators

پردازش اطلاعات این وظیفه‌ها بر عهده سه لایه بالاتر است که با توجه به ویژگی‌های هر وظیفه، مشخص می‌شود که هر وظیفه در کدام گره پردازشی پردازش شود. گاهای این قابلیت وجود دارد که یک وظیفه به چند بخش تقسیم شود و هر بخش آن توسط یک گره پردازشی پردازش شود، این قابلیت در این پایان‌نامه نیز وجود دارد. هدف این فصل این است که مشخص کند هر وظیفه به چه نحوی و در کدام گره پردازشی، پردازش شود. پس از پردازش وظیفه‌ها در گره‌های پردازشی لازم است که نتیجه پردازش مشخص شود. این نتیجه‌ها به عنوان مجموعه‌ای از وظیفه‌های جدید به گره‌های عملگر در لایه اول ارسال می‌شود. در ادامه به مدلسازی دقیق‌تر می‌پردازیم.

مجموعه گره‌های لایه‌ی ابر، مه، لبه و حسگر را به ترتیب با C , F , E و S نمایش می‌دهیم. همچنین مجموعه منابع موجود در هر گره پردازشی را با R نمایش می‌دهیم که در این پایان‌نامه این مجموعه را شامل سه منبع پردازشی پردازنده^۱, حافظه^۲ و همچنین فضای دیسک^۳ در نظر می‌گیریم.

$$C = \{v_1^c, v_2^c, \dots, v_{|C|}^c\} \quad (آ.۳)$$

$$F = \{v_1^f, v_2^f, \dots, v_{|F|}^f\} \quad (ب.۳)$$

$$E = \{v_1^e, v_2^e, \dots, v_{|E|}^e\} \quad (ج.۳)$$

$$S = \{v_1^s, v_2^s, \dots, v_{|S|}^s\} \quad (د.۳)$$

$$R = \{CPU, RAM, Storage\} \quad (ه.۳)$$

مقدار منبع پردازشی موجود در هر گره پردازشی، همچنین نوع آن منبع، لازم است که با یک نماد مشخص شود. در اینجا از نماد σ برای این کار استفاده می‌کنیم. به عنوان مثال می‌توان گفت که σ_c^r برابر است با مجموع ظرفیت پردازشی موجود از منبع $r \in R$ در گره پردازشی $c \in C$. همچنین در مورد مجموعه‌های تعریف شده در رابطه (۱.۳) معادل حرف کوچک از هر مجموعه را به عنوان نماینده‌ای از اعضای آن مجموعه در نظر می‌گیریم، مثلاً به این صورت که c نشان‌دهنده‌ی یکی از گره‌های موجود در لایه‌ی ابر یعنی C است. همچنین تعداد اعضای موجود در هر مجموعه را با علامت $\|\cdot\|$ مشخص کردہ‌ایم.

¹CPU

²RAM

³Storage

مجموعه‌ی مهم دیگری که تعریف می‌شود مجموعه‌ی وظیفه‌های موجود در شبکه است. هر وظیفه به عنوان یک بسته اطلاعاتی تعریف می‌شود که ویژگی‌ها مشخص و مخصوص به خود را دارد. برای هر وظیفه چهار ویژگی درنظر گرفته می‌شود که به صورت رابطه (۲.۳) تعریف می‌شود.

$$t := (w_t, \delta_t, N_t, f_t^r(\lambda_t)) \quad (2.3)$$

در رابطه (۲.۳) w_t ، δ_t و N_t به ترتیب میزان حجم پردازنده مورد نیاز، بیشترین تعداد از وظیفه‌های مشابه و همچنین حداقل زمان مورد نیاز برای پردازش کامل، برای وظیفه $t \in T$ را نشان می‌دهد. واحد اندازه‌گیری در این سه متغیر می‌تواند به سلیقه‌ی استفاده کننده تغییر کند اما به عنوان مثال می‌توان گفت که واحد اندازه‌گیری میزان حجم پردازنده را گیگاهرتز^۱ در نظر گرفت و یا واحد اندازه‌گیری در حداقل زمان مورد نیاز میلی‌ثانیه انتخاب کرد. در میان ویژگی‌های مربوط به یک وظیفه یکتابع به نام f_t^r نیز دیده می‌شود که این تابع نرخ استفاده وظیفه از منبع $r \in R$ را نشان می‌دهد که ورودی آن متغیری به نام λ است که نشان‌دهنده‌ی نرخ ورود وظیفه به گره پردازشی است. تعداد وظیفه‌های رسیده به یک گره پردازشی را به عنوان یک فرآیند تصادفی گستته در نظر می‌گیریم. جنس این فرآیند تصادفی مشابه جنس تعداد وظیفه‌های تولیدی در گره‌های حسگری است. در این پایان‌نامه تعداد وظیفه‌های تولیدشده در گره‌های حسگری را به عنوان یک فرآیند تصادفی پوآسون با پارامتر λ در نظر می‌گیریم. نرخ تولید وظیفه بسته به نوع وظیفه و گره حسگری می‌تواند متفاوت باشد لذا لازم است پارامتر λ متناسب با نوع وظیفه و گره حسگری تعریف شود. $\lambda_{t,s}$ به عنوان نرخ پوآسون تولید وظیفه در گره $s \in S$ درنظر گرفته می‌شود. مقادیر مربوط به این نرخ‌ها به عنوان ورودی مسئله هستند.

نهایتاً مجموعه وظیفه‌های موجود در شبکه را با T نشان می‌دهیم.

$$T = \{t_1, t_2, \dots, t_{|T|}\} \quad (3.3)$$

در ادامه لازم است که تاخیر بین گره‌ها مدل‌سازی شود، همانطور که در قسمت مقدمه گفته شد در این پایان‌نامه در مورد تاخیرها، فقط تاخیر انتقال را مدنظر قرار می‌دهیم، بدین صورت که مثلاً $\tau_{t,s,c}^{tr}$ برابر است با میزان تاخیر انتقال از گره حسگری $S \in s$ تا گره پردازشی $C \in c$ برای وظیفه $t \in T$.

¹GHz

نماد دیگری که در ادامه از آن استفاده می‌شود نماد π است که به میزان هزینه‌های مربوط به پردازش را در گره‌های پردازشی نشان می‌دهد. برای جلوگیری از پیچیدگی بیشتر صورت مسئله برای هر گره پردازشی یک هزینه برای تمامی منابع آن گره در نظر گرفته می‌شود که به عنوان یک ضریب در رابطه‌های بعدی از آن استفاده خواهد شد. به عنوان مثال تمامی هزینه‌های پردازشی در گره $c \in C$ با نماد π_c نمایش داده می‌شود و مفهوم آن بدین صورت است که مثلاً برای استفاده از پردازنده در این گره به ازای هر واحد پردازشی در ثانیه لازم است که مقدار π_c واحد پول پرداخته شود.

جدول (۱.۳) به صورت خلاصه پارامترها و متغیرهای استفاده شده در این فصل را معرفی می‌کند.

جدول ۱.۳: نمادهای استفاده شده در فصل (۳)

توضیح	نشانه
مجموعه گره‌های پردازشی لایه ابر	C
مجموعه گره‌های پردازشی لایه مه	F
مجموعه گره‌های پردازشی لایه لبه	E
مجموعه گره‌های لایهی حسگر	S
مجموعه وظیفه‌های موجود در شبکه	T
مجموعه منابع پردازشی موجود در شبکه	R
نماد مربوط به گره‌های پردازشی لایه ابر	c
نماد مربوط به گره‌های پردازشی لایه مه	f
نماد مربوط به گره‌های پردازشی لایه لبه	e
نماد مربوط به گره‌های پردازشی لایهی حسگر	s
نماد مربوط به وظیفه‌ها	t
نماد مربوط به منابع پردازشی	r
واحد قیمت پردازشی در گره پردازشی	π_c
نرخ پوآسون تولید وظیفه در حسگر	$\lambda_{t,s}$
نرخ پوآسون ورود وظیفه به گره پردازشی	$\lambda_{t,c}$
ظرفیت منبع پردازشی در گره پردازشی	σ_c^r
یک عدد مثبت کوچک	ϵ
تأخیر انتقال اطلاعات مربوط به یک وظیفه تا گره پردازشی	$\tau_{t,s,c}^{tr}$
نرخ مصرف منبع در گره پردازشی جهت پردازشی وظیفه	$f_t^r(\lambda_{t,e})$
ضریب‌های استفاده شده در تابع f_t^r	k_1^r, k_2^r
حجم پردازنده مورد نیاز جهت پردازش وظیفه	w_t
حداقل زمان مورد نیاز جهت پردازش کامل وظیفه	δ_t

۱۰۳ متغیرها

در این بخش قرار است در مورد متغیرهای استفاده شده در مسئله به تفصیل صحبت شود. در مورد محل پردازش وظیفه ها، سه متغیر دودویی به صورت زیر تعریف می شود:

$$x_{t,c} = \begin{cases} 1 & \text{وظیفه } c \in C \text{ در گره } t \in T \text{ پردازش می‌شود} \\ 0 & \text{در غیر این صورت} \end{cases} \quad (\tilde{\alpha}_{4.3})$$

$$x_{t,f} = \begin{cases} 1 & \text{وظیفه } f \in F \text{ در گره } t \in T \text{ پردازش می‌شود} \\ 0 & \text{در غیر این صورت} \end{cases} \quad (4.3.b)$$

$$x_{t,e} = \begin{cases} 1 & \text{وظیفه } e \in E \text{ در گره } t \in T \text{ پردازش می شود} \\ 0 & \text{در غیر این صورت} \end{cases} \quad (4.3) \text{ ج}$$

دسته دوم متغیر که با β نشان داده می‌شود متغیر پیوسته مسئله است که در رابطه (۵.۳) معرفی می‌شود و مربوط است به نرخ لحظه‌ای جریان ارسالی وظیفه‌ها بین گره‌های حسگری و پردازشی، به عنوان مثال $\beta_{t,s,c}$.
برابر است با نرخ جریان ارسالی از نوع وظیفه‌ی $T \in t$ ، از گره حسگری $S \in s$ به گره پردازشی $c \in C$.

$$0 \leq \beta_{t,s,c} \leq \lambda_{t,s} \quad \forall t \in T, \forall s \in S, \forall c \in C \quad (\text{I}\delta.\text{r})$$

$$0 \leq \beta_{t,s,f} \leq \lambda_{t,s} \quad \forall t \in T, \forall s \in S, \forall f \in F \quad (\text{Def. 3})$$

$$0 \leq \beta_{t,s,e} \leq \lambda_{t,s} \quad \forall t \in T, \forall s \in S, \forall e \in E \quad (\text{জ৫.৩})$$

متغیرهای اصلی در مسئله دو دسته هستند که دسته اول از جنس متغیر گسسته دودویی هستند و دسته دوم متغیرهای پیوسته. تا اینجای کار احتمالا خواننده می‌تواند حدرس بزند که مسئله بهینه‌سازی که در ادامه شکل

می‌گیرد از نوع مسئله‌های مخلوط عدد صحیح^۱ خواهد بود. در مورد خطی بودن یا نبودن نیز می‌دانیم که کار کردن با مسئله بهینه‌سازی خطی به مراتب راحت‌تر از غیرخطی خواهد بود، لذا به نظر می‌رسد که لازم است مسئله به‌گونه‌ای مدل شود که خطی باشد و یا اینکه خطی‌سازی شود. در ادامه قیدهای موجود در مدل‌سازی بررسی می‌شود.

۲.۲.۳ قیدها

اولین قید مورد بررسی به عنوان یک متغیر میانی معرفی می‌شود به این صورت که مجموع جریان‌های ورودی به یک گره پردازشی به عنوان یک متغیر میانی که در رابطه (۶.۳) نشان داده شده است نمایش داده می‌شود. درواقع این متغیر نرخ ورود وظیفه به گره‌های پردازشی را نشان می‌دهد.

$$\lambda_{t,c} = \sum_{s \in S} \beta_{t,s,c} \quad \forall t \in T, \forall c \in C \quad (\text{آ}6.3)$$

$$\lambda_{t,f} = \sum_{s \in S} \beta_{t,s,f} \quad \forall t \in T, \forall f \in F \quad (\text{ب}6.3)$$

$$\lambda_{t,e} = \sum_{s \in S} \beta_{t,s,e} \quad \forall t \in T, \forall e \in E \quad (\text{ج}6.3)$$

قید رابطه (۷.۳) به این صورت است که رابطه‌ی بین متغیرهای پیوسته و گسسته (دودویی) مسئله را مشخص می‌کند. بدین صورت که اگر سمت راست نامساوی صفر باشد آنگاه سمت چپ نیز ناچاراً لازم است که صفر باشد و این یعنی جریانی وجود ندارد، همچنین اگر سمت چپ غیرصفر باشد آنگاه متغیر دودویی سمت راست به ناچار باید مقدارش یک باشد. ممکن است این سوال پیش بیاید که اگر سمت چپ صفر شد و سمت راست یک، آنگاه چه اتفاقی خواهد افتاد و یا اصلاً آیا امکان دارد که این اتفاق بیفتد؟ می‌توان گفت که اگر این اتفاق بیفتد یعنی یک گره قرار است یک وظیفه با نرخ صفر را پردازش کند، در این حالت یک هزینه پایه برای آن گره درنظر گرفته می‌شود، ازانجاییکه جلوتر در بخش تابع هدف خواهیم دید، هدف کمینه کردن هزینه پردازشی در کل گره‌های پردازشی است، لذا این اتفاق یک اتفاق به صرفه نیست و می‌توان ادعا

^۱Mixed-Integer

کرد که به کمک وجود تابع هدف این اتفاق رخ نخواهد داد.

$$\frac{\lambda_{t,c}}{\sum_{s \in S} \lambda_{t,s}} \leq x_{t,c} \quad \forall t \in T, \forall c \in C \quad (\tilde{A}7.3)$$

$$\frac{\lambda_{t,f}}{\sum_{s \in S} \lambda_{t,s}} \leq x_{t,f} \quad \forall t \in T, \forall f \in F \quad (B7.3)$$

$$\frac{\lambda_{t,e}}{\sum_{s \in S} \lambda_{t,s}} \leq x_{t,e} \quad \forall t \in T, \forall e \in E \quad (C7.3)$$

قید رابطه (۸.۳) مربوط به مدیریت منابع است به این صورت که مجموع منابع استفاده شده در هر گره پردازشی لازم است از کل منابع موجود در آن گره کمتر باشد. در این قسمت در مورد تابع $f_t^r(\lambda)$ بیشتر صحبت می‌کنیم.

$$\sum_{t \in T} x_{t,e} f_t^r(\lambda_{t,e}) \leq \sigma_e^r \quad \forall r \in R, \forall e \in E \quad (\tilde{A}8.3)$$

$$\sum_{t \in T} x_{t,f} f_t^r(\lambda_{t,f}) \leq \sigma_f^r \quad \forall r \in R, \forall f \in F \quad (B8.3)$$

$$\sum_{t \in T} x_{t,c} f_t^r(\lambda_{t,c}) \leq \sigma_c^r \quad \forall r \in R, \forall c \in C \quad (C8.3)$$

همانطور که در رابطه (۹.۳) مشاهده می‌شود این تابع یک تابع خطی از ورودی اش می‌باشد و نشان می‌دهد که وظیفه t با چه نرخی منبع پردازشی از نوع r را مصرف می‌کند با براین لازم است که واحد آن از جنس مقدار منبع بر ثانیه باشد. در مورد ضرایب استفاده شده در این تابع می‌توان گفت که دو ضریب k_1^r و k_2^r بسته با نوع منبع و همچنین مشخصات گره محاسباتی تبیین می‌شوند که مقادیر آن‌ها در جدولی در ادامه خواهد آمد. در نهایت می‌توان گفت که مقدار خروجی این تابع وابسته به نوع وظیفه، نوع گره و میزان جریان ورودی به گره است.

$$f_t^r(\lambda_{t,c}) = k_1^r \lambda_{t,c} + k_2^r s \quad (9.3)$$

در رابطه (۱۰.۳) دیده می‌شود دو متغیر مسئله یعنی x و λ که مجموع خطی از β است، به صورت حاصلضرب درآمده‌اند که این اتفاق مسئله را از حالت خطی خارج می‌کند و همانجاًی است که لازم است مسئله خطی‌سازی

فصل ۳. تخصیص منابع پردازشی در شبکه اینترنت اشیاء به صورت متمرکز و غیرمتمرکز

شود. برای این کار یک متغیر میانی در رابطه (۱۰.۳) تعریف می‌شود. با این کار حاصلضرب ایجاد شده از بین می‌رود و قیدهای مسئله تاکنون همچنان خطی باقی می‌مانند. اما همانطور که می‌دانیم ایجاد یک متغیر جدید هزینه‌هایی هم دارد که باید بهای آن پرداخته شود و این بها به وجود آمدن قیدهای جدید در مسئله است.

$$x_{t,c} f_t^r(\lambda_{t,c}) = k_1^r x_{t,c} \lambda_{t,c} + k_2^r x_{t,c} \quad (10.3)$$

$$\psi_{t,c} \triangleq x_{t,c} \lambda_{t,c} \Rightarrow 0 \leq \psi_{t,c} \leq \lambda_{t,c} \quad (10.3)$$

قبل از بررسی قیدهای اضافه شده لازم است که یک نماد دیگر معرفی شود. این نماد که با حرف Q نشان داده شده است اولین بار در رابطه (۱۱.۳) دیده شده است که برابر است با مقدار بالقوه‌ی بیشترین جریان ورودی به یک گره محاسباتی. در خاص‌ترین حالت می‌توان گفت که تمام جریان تولیدی وظیفه‌های موجود در شبکه برای پردازش به یک گره ارسال شوند در این صورت Q برابر است با مجموع جریان تولید شده‌ی همه‌ی وظیفه‌ها در تمام حسگرها که در رابطه (۱۱.۳) نحوه‌ی محاسبه آنچه که گفته شد دیده می‌شود.

$$Q(x_{t,c} - 1) + \lambda_{t,c} \leq \psi_{t,c} \leq x_{t,c} Q \quad (11.3)$$

$$\begin{aligned} Q &= \max_{t \in T, c \in C} \lambda_{t,c} \\ &= \max_{t \in T, c \in C} \sum_{s \in S} \beta_{t,s,c} \\ &= \sum_{s \in S} \max_{t \in T, c \in C} \beta_{t,s,c} \\ &= \sum_{s \in S} \lambda_{t,s} \end{aligned} \quad (11.3)$$

در مورد متغیر میانی تعریف شده در رابطه (۱۰.۳ ب) قیدهای مربوطه در رابطه (۱۲.۳) دیده می‌شود.

$$0 \leq \psi_{t,c} \leq \lambda_{t,c} \quad (12.3)$$

$$Q(x_{t,c} - 1) + \lambda_{t,c} \leq \psi_{t,c} \leq x_{t,c}Q \quad \forall t \in T, \forall c \in C \quad (12.3)$$

$$0 \leq \psi_{t,f} \leq \lambda_{t,f} \quad (12.3)$$

$$Q(x_{t,f} - 1) + \lambda_{t,f} \leq \psi_{t,f} \leq x_{t,f}Q \quad \forall t \in T, \forall f \in F \quad (12.3)$$

$$0 \leq \psi_{t,e} \leq \lambda_{t,e} \quad (12.3)$$

$$Q(x_{t,e} - 1) + \lambda_{t,e} \leq \psi_{t,e} \leq x_{t,e}Q \quad \forall t \in T, \forall e \in E \quad (12.3)$$

حال با تعریف متغیر رابطه (۱۰.۳ ب) می‌توان قید موجود در رابطه (۸.۳) را به صورت رابطه (۱۳.۳) دوباره نویسی کرد.

$$\sum_{t \in T} k_1^r \psi_{t,c} + k_2^r x_{t,c} \leq \sigma_c^r \quad \forall r \in R, \forall c \in C \quad (13.3)$$

$$\sum_{t \in T} k_1^r \psi_{t,f} + k_2^r x_{t,f} \leq \sigma_f^r \quad \forall r \in R, \forall f \in F \quad (13.3)$$

$$\sum_{t \in T} k_1^r \psi_{t,e} + k_2^r x_{t,e} \leq \sigma_e^r \quad \forall r \in R, \forall e \in E \quad (13.3)$$

قید بعدی که مورد بررسی قرار می‌گیرد قید مربوط به تاخیرهاست. ابتدا تاخیر موجود در شبکه از زمانی که وظیفه ارسال می‌شود تا لحظه‌ای که وظیفه پردازش می‌شود به صورت رابطه (۱۴.۳) تعریف می‌شود، همانطور که دیده می‌شود این تعریف از دو بخش تاخیر انتقال و تاخیر صفت تشکیل شده است. پارامتر $\tau_{t,s,c}^{tr}$ همانطور که قبلاً معرفی شد به عنوان تاخیر ثابت بین لایه‌ها درنظر گرفته می‌شود.

$$\tau_{t,c} = \tau_{t,s,c}^{tr} + \frac{1}{\mu_{t,c} - \lambda_{t,c}} \quad (14.3)$$

با توجه به اینکه ورودی هر گره پردازشی یک فرآیند پواسون است. اگر بخواهیم هر گره پردازشی را به صورت یک سیستم در نظر بگیریم، با فرض اینکه سرور موجود در هرگره به صورتی باشد که مدت زمان

فصل ۳. تخصیص منابع پردازشی در شبکه اینترنت اشیاء به صورت متمرکز و غیرمتمرکز

پردازش وظیفه‌ها یک متغیر تصادفی نمایی باشد، آنگاه سیستم درنظر گرفته شده از نوع $M/M/1$ خواهد بود. همانطور که می‌دانیم برای محاسبه‌ی تاخیر صفت در این مدل سیستم، دو متغیر نرخ ورود بسته‌ها λ و نرخ پردازش آن‌ها μ لازم است. در مدل ما نرخ ورود وظیفه‌ها در هر گره به صورت یک متغیر میانی تعریف شده است اما در مورد نرخ پردازش بسته‌ها به توجه به نوع بسته و همچنین میزان قدرت پردازشی گره رابطه‌ی رابطه (۱۵.۳) در نظر گرفته شده است.

$$\frac{1}{\mu_{t,c}} = \frac{w_t}{f_t^{cpu}(\lambda_{t,c})} \quad (15.3)$$

با جایگزین کردن موارد گفته شده رابطه‌ی نهایی تاخیر موجود در گره‌ها به صورت رابطه (۱۶.۳ ب) نمایش داده می‌شود.

$$f_t^{cpu}(\lambda_{t,c}) = k_1^{cpu} \lambda_{t,c} + k_2^{cpu} \quad (16.3)$$

$$\tau_{t,c} = \tau_{t,s,c}^{tr} + \frac{w_t}{(k_1^{cpu} - w_t)\lambda_{t,c} + k_2^{cpu}} \quad (16.3 \text{ ب})$$

در نهایت قید مربوط به تاخیر در رابطه (۱۷.۳ آ) آورده شده است که با جایگذاری و ساده سازی به صورت رابطه (۱۸.۳) نمایش داده می‌شود.

$$x_{t,c} \tau_{t,c} \leq \delta_t \quad \forall t \in T, \forall s \in S, \forall c \in C \quad (17.3)$$

$$x_{t,f} \tau_{t,f} \leq \delta_t \quad \forall t \in T, \forall s \in S, \forall f \in F \quad (17.3 \text{ ب})$$

$$x_{t,e} \tau_{t,e} \leq \delta_t \quad \forall t \in T, \forall s \in S, \forall e \in E \quad (17.3 \text{ ج})$$

$$\begin{aligned} & x_{t,c} \lambda_{t,c} (k_1^{cpu} - w_t) \tau_{t,s,c}^{tr} + x_{t,c} k_2^{cpu} \tau_{t,s,c}^{tr} + w_t x_{t,c} - k_2^{cpu} \delta_t \\ & - (k_1^{cpu} - w_t) \delta_t \lambda_{t,c} \leq 0 \quad \forall t \in T, \forall s \in S, \forall c \in C \end{aligned} \quad (\tilde{1}18.3)$$

$$\begin{aligned} & x_{t,f} \lambda_{t,f} (k_1^{cpu} - w_t) \tau_{t,s,f}^{tr} + x_{t,f} k_2^{cpu} \tau_{t,s,f}^{tr} + w_t x_{t,f} - k_2^{cpu} \delta_t \\ & - (k_1^{cpu} - w_t) \delta_t \lambda_{t,f} \leq 0 \quad \forall t \in T, \forall s \in S, \forall f \in F \end{aligned} \quad (18.3)$$

$$\begin{aligned} & x_{t,e} \lambda_{t,e} (k_1^{cpu} - w_t) \tau_{t,s,e}^{tr} + x_{t,e} k_2^{cpu} \tau_{t,s,e}^{tr} + w_t x_{t,e} - k_2^{cpu} \delta_t \\ & - (k_1^{cpu} - w_t) \delta_t \lambda_{t,e} \leq 0 \quad \forall t \in T, \forall s \in S, \forall e \in E \end{aligned} \quad (ج 18.3)$$

در صورت استفاده از متغیر میانی رابطه (۱۰.۳) در قید رابطه (۱۸.۳) می‌توان این قید را به صورت رابطه (۱۹.۳) به صورت نهایی نمایش داد.

$$\begin{aligned} & \psi_{t,s,c} (k_1^{cpu} - w_t) \tau_{t,s,c}^{tr} + x_{t,c} k_2^{cpu} \tau_{t,s,c}^{tr} + w_t x_{t,c} - k_2^{cpu} \delta_t \\ & - (k_1^{cpu} - w_t) \delta_t \lambda_{t,c} \leq 0 \quad \forall t \in T, \forall s \in S, \forall c \in C \end{aligned} \quad (\tilde{1}19.3)$$

$$\begin{aligned} & \psi_{t,s,f} (k_1^{cpu} - w_t) \tau_{t,s,f}^{tr} + x_{t,f} k_2^{cpu} \tau_{t,s,f}^{tr} + w_t x_{t,f} - k_2^{cpu} \delta_t \\ & - (k_1^{cpu} - w_t) \delta_t \lambda_{t,f} \leq 0 \quad \forall t \in T, \forall s \in S, \forall f \in F \end{aligned} \quad (19.3)$$

$$\begin{aligned} & \psi_{t,s,e} (k_1^{cpu} - w_t) \tau_{t,s,e}^{tr} + x_{t,e} k_2^{cpu} \tau_{t,s,e}^{tr} + w_t x_{t,e} - k_2^{cpu} \delta_t \\ & - (k_1^{cpu} - w_t) \delta_t \lambda_{t,e} \leq 0 \quad \forall t \in T, \forall s \in S, \forall e \in E \end{aligned} \quad (ج 19.3)$$

قید بعدی که مورد بررسی قرار می‌گیرد قید مربوط به پایداری صفت در گره‌هاست، همانطور که می‌دانیم شرط کافی برای پایداری بودن یک سیستم $M/M/1$ این است که نرخ ورود اطلاعات از نرخ سرویس دادن کمتر باشد. در مدل ما این قید به صورت $\lambda_{t,c} < \mu_{t,c}$ برای وظیفه $t \in T$ و گره $c \in C$ نوشته می‌شود. نکته مدنظر این است که این شرط در صورتی برای این وظیفه و این گره نوشته می‌شود که وظیفه مورد نظر در گره مورد نظر پردازش شود، درواقع یعنی $x_{t,c} = 1$ لذا طرفین نامساوی گفته شده را در $x_{t,c}$ ضرب می‌کنیم. نکته مدنظر بعدی اینکه معمولاً نامساوی را به صورت نامساوی محض درنظر نمی‌گیرند، بلکه به صورت کوچکتری نامساوی

نوشته می شود. برای انجام این تبدیل می توانیم به سمت چپ نامساوی یک عدد مثبت خیلی کوچک اضافه کنیم. با توجه به نکات گفته شده قید نهایی به صورت رابطه (۲۰.۳) آورده شده است.

$$x_{t,c} \times (\lambda_{t,c} + \epsilon) \leq x_{t,c} \times (\mu_{t,c}) \quad \forall t \in T, \forall c \in C \quad (۱۲۰.۳)$$

$$x_{t,f} \times (\lambda_{t,f} + \epsilon) \leq x_{t,f} \times (\mu_{t,f}) \quad \forall t \in T, \forall f \in F \quad (۲۰.۳\text{ب})$$

$$x_{t,e} \times (\lambda_{t,e} + \epsilon) \leq x_{t,e} \times (\mu_{t,e}) \quad \forall t \in T, \forall e \in E \quad (۲۰.۳\text{ج})$$

با اعمال متغیر میانی تعریف شده در رابطه (۱۰.۳\text{ب}) بر روی قید رابطه (۲۰.۳)، این قید به صورت رابطه (۲۱.۳) نوشته می شود.

$$x_{t,c}(\epsilon w_t - k_2^{cpu}) + \psi_{t,c}(w_t - k_1^{cpu}) \leq 0 \quad \forall t \in T, \forall c \in C \quad (۱۲۱.۳)$$

$$x_{t,f}(\epsilon w_t - k_2^{cpu}) + \psi_{t,f}(w_t - k_1^{cpu}) \leq 0 \quad \forall t \in T, \forall f \in F \quad (۲۱.۳\text{ب})$$

$$x_{t,e}(\epsilon w_t - k_2^{cpu}) + \psi_{t,e}(w_t - k_1^{cpu}) \leq 0 \quad \forall t \in T, \forall e \in E \quad (۲۱.۳\text{ج})$$

تا اینجا بین تمام قیدهای گفته شده یک ویژگی مشترک وجود داشت و این بود که در تمامی قیدهای گفته شده قابلیت قید هر گره محاسباتی از سایر گرهها جدا بود و نمی توان قیدها را به گونه ای تجزیه کرد که هر گره محاسباتی صرفاً قید مربوط به خودش را بیند که این اتفاق یک اتفاق خواشایند است که در ادامه علت خواشایند بودن آن مشخص خواهد شد. اما در دوقید بعد خواهیم دید که بین گره های محاسباتی به نوعی اتصال ایجاد خواهد شد. اولین قید اتصالی که مورد بررسی قرار می گیرد مربوط به وجود تعادل بین جریان ورودی وظیفه ها و جریان خروجی وظیفه هاست، بدین صورت که نرخ تولید یک وظیفه خاص در یک گره حسگری برابر است با مجموع نرخ ارسالی بین آن گره حسگری و تمام گره های پردازشی برای آن وظیفه خاص. این قید در رابطه (۲۲.۳) آورده شده است. همانطور که می بینید در این قید نمی توانیم برای هر گره پردازشی یک رابطه جدا مشخص کنیم.

$$\lambda_{t,s} = \sum_{e \in E} \beta_{t,s,e} + \sum_{f \in F} \beta_{t,s,f} + \sum_{c \in C} \beta_{t,s,c} \quad \forall t \in T, \forall s \in S \quad (۲۲.۳)$$

قید بعدی مربوط به شرط لازم برای پردازش همه‌ی وظیفه‌های است بدين صورت که هر وظیفه حداقل باید در یکی از گره‌های پردازشی موجود در شبکه پردازش شود. در این پایان‌نامه این فرض در نظر گرفته شده‌است که یک وظیفه می‌تواند در چندین گره پردازشی نیز پردازش شود این فرض این قابلیت را به مسئله اضافه می‌کند که وظیفه‌ها به صورت افقی مقیاس‌پذیر^۱ باشند به این معنی که بر روی تعداد وظیفه‌های از جنس یک سرویس می‌تواند محدودیتی وجود ندارد. برای این کار یک متغیر تحت عنوان N_t در نظر گرفته می‌شود که هرچقدر این عدد بزرگتر باشد مقیاس‌پذیری سیستم بیشتر است. قید مربوطه در رابطه (۲۳.۳) آورده شده‌است. به وضوح در این قید نیز دیده می‌شود که نمی‌توان رابطه‌ی مستقلی برای هر یک از گره‌های پردازشی پیدا کرد.

$$1 \leq \sum_{e \in E} x_{t,e} + \sum_{f \in F} x_{t,f} + \sum_{c \in C} x_{t,c} \leq N_t \quad \forall t \in T \quad (23.3)$$

۳.۲.۳ تابع هدف

همانطور که در ابتدای فصل (۳) توضیح داده شد، قرار است که مدل‌سازی ریاضی به صورت یک مسئله بهینه‌سازی انجام شود. تا ایجا متغیرهای مربوط به این مسئله، همچنین قیدهای مربوطه مشخص شد. در گام بعدی لازم است که برای این مسئله بهینه‌سازی یک تابع هدف^۲ مشخص شود. برای این کار لازم که ابتدا هدف از صورت مسئله مشخص شود. اولین هدف در این مقاله کم کردن هزینه‌های مربوط به کل شبکه است. می‌دانیم که پردازش وظیفه‌ها در گره‌های پردازشی یک فرآیند هزینه‌بر است، حال صورت مسئله بهینه‌سازی با این هدف تعریف می‌شود که مجموع کل هزینه‌های موچود در شبکه کمترین مقدار ممکن باشد. برای این کار لازم است از متغیرهای دودویی مسئله^x، تابع مشخص کننده نرخ مصرف منبع پردازشی^f و همچنین واحد

¹Scalable

²Objective function

هزینه‌های پردازشی π استفاده شود. درنهایت تابع هدف به صورت رابطه (۲۴.۳) تعریف می‌شود.

$$\begin{aligned} & \min \sum_{t \in T} \sum_{e \in E} (x_{t,e} \pi_e \sum_{r \in R} f_t^r(\lambda_{t,e})) \\ & + \sum_{t \in T} \sum_{f \in F} (x_{t,f} \pi_f \sum_{r \in R} f_t^r(\lambda_{t,f})) \\ & + \sum_{t \in T} \sum_{c \in C} (x_{t,c} \pi_c \sum_{r \in R} f_t^r(\lambda_{t,c})) \end{aligned} \quad (24.3)$$

برای سادهسازی در نوشتار یک متغیر میانی به نام Γ به صورت رابطه (۲۵.۳) تعریف می‌شود.

$$\begin{aligned} \Gamma_{t,e} & \triangleq \pi_e \sum_{r \in R} f_t^r(\lambda_{t,e}) \\ & = \pi_e ((k_1^{cpu} + k_1^{ram} + k_1^{storage}) \lambda_{t,e} \\ & + k_2^{cpu} + k_2^{ram} + k_2^{storage}) \\ & = \pi_e (K_1 \lambda_{t,e} + K_2) \end{aligned} \quad (25.3)$$

با جایگذاری متغیر تعریف شده در رابطه (۲۵.۳) در رابطه (۲۴.۳) تابع هدف به صورت رابطه (۲۶.۳) بازنویسی می‌شود.

$$\begin{aligned} & \min \sum_{t \in T} \sum_{e \in E} x_{t,e} \Gamma_{t,e} \\ & + \sum_{t \in T} \sum_{f \in F} x_{t,f} \Gamma_{t,f} \\ & + \sum_{t \in T} \sum_{c \in C} x_{t,c} \Gamma_{t,c} \end{aligned} \quad (26.3)$$

همچنین به کمک متغیر تعریف شده در رابطه (۱۰.۳ ب) می‌توان رابطه زیر را نوشت.

$$\begin{aligned} x_{t,e}\Gamma_{t,e} &= K_1\pi_e x_{t,e}\lambda_{t,e} + K_2\pi_e x_{t,e} \\ &= K_1\pi_e \psi_{t,e} + K_2\pi_e x_{t,e} \end{aligned} \quad (۲۷.۳)$$

درنهایت با ساده‌سازی‌های انجام شده و همچنین استفاده از متغیرهای کمکی می‌توان گفت که مسئله بهینه‌سازی نهایی به صورت رابطه (۲۸.۳) نوشته می‌شود. این مسئله بهینه‌سازی به صورت متمرکز^۱ قابل حل خواهد بود و ابزارهای مختلفی برای حل آن وجود دارد.

$$\begin{aligned} \min \sum_{t \in T} \sum_{e \in E} & K_1\pi_e \psi_{t,e} + K_2\pi_e x_{t,e} \\ & + \sum_{t \in T} \sum_{f \in F} K_1\pi_f \psi_{t,f} + K_2\pi_f x_{t,f} \\ & + \sum_{t \in T} \sum_{c \in C} K_1\pi_c \psi_{t,c} + K_2\pi_c x_{t,c} \\ \text{subj. to } & (3.6) (3.7) (3.12) (3.13) \\ & (3.19) (3.21) (3.22) (3.23) \end{aligned} \quad (۲۸.۳)$$

۳.۳ راه حل غیرمتمرکز

در این بخش قرار است که مسئله بهینه‌سازی نهایی که در رابطه (۲۸.۳) نوشته شد به صورت غیرمتمرکز^۲ حل شود. همانطور که در قسمت بررسی قیدهای مربوط به مسئله اصلی گفته شد، بجز دو قید از قیدهای مسئله که به صورت متصل بودند بقیه قیدها این قابلیت را داشتند که بین گره‌ها تجزیه شوند.

در این بخش ابتدا با ارجاع به یکی از مقاله‌های موجود در این زمینه در مورد نحوه حل مسائل MILP به صورت غیرمتمرکز صحبت می‌شود و در ادامه این راه حل ارائه شده بر روی مسئله اصلی اجرا می‌شود. در مقاله [۳۲] به صورت کامل در مورد روش غیرمتمرکز صحبت شده است با ارجاع به این مقاله در ادامه این

^۱Centralized

^۲Decentralized

روش را مورد بررسی قرار می‌دهیم.

فرض کنید که مسئله MILP به صورت رابطه (۲۹.۳) نوشته شده باشد.

$$\begin{aligned} & \min_{x_1, \dots, x_m} \sum_{i=1}^m c_i^T x_i \\ \text{subj. to } & \sum_{i=1}^m A_i x_i \leq b \\ & x_i \in X_i, i = 1, \dots, m \end{aligned} \quad (29.3)$$

$$b \in \mathbb{R}^p$$

$$X_i = \{x_i \in \mathbb{R}^{n_{c,i}} \times \mathbb{Z}^{n_{d,i}} : D_i x_i \leq d_i\} \quad (29.3)$$

اگر برای مسئله رابطه (۲۹.۳) رابطه‌ی مسئله‌ی تجزیه دوگان^۱ را بنویسیم، آنگاه به رابطه (۳۰.۳) می‌رسیم.

$$\max_{\lambda \geq 0} -\lambda^T b + \sum_{i=1}^m \min_{x_1, \dots, x_m} (c_i^T + \lambda^T A_i) x_i \quad (30.3)$$

بعد از حل مسئله رابطه (۳۰.۳) می‌توان متغیرهای اصلی یعنی x را به صورت رابطه (۳۱.۳) یافت.

$$\begin{aligned} x(\lambda^*) &= [x_1(\lambda^*)^T, \dots, x_m(\lambda^*)^T]^T \\ x_i(\lambda) &\in \arg \min_{x_i \in \text{vert}(X_i)} (c_i^T + \lambda^T A_i) x_i \end{aligned} \quad (31.3)$$

در مقاله [۳۲] با یک مثال نشان داده می‌شود که با طی کردن مراحل بالا برای یافتن جواب بهینه x ، می‌توان تضمین کرد که شرایط محلی یعنی $x_i \in X_i$ حتماً ارضامی شود اما مtasفانه نمی‌توان حتماً تضمین کرد که قیدهای متصل نیز ارضامی شوند. راه حل جایگزین استفاده از راه حل ارائه شده در مقاله [۳۳] است که در آنجا مسئله را از حالت LP به حالت MILP تبدیل می‌کند، با این کار تضمین می‌شود که قیدهای متصل حتماً ارضام شوند اما از آن طرف به دلیل وجود متغیرهای گسسته این احتمال وجود دارد که قیدهای محلی به طور

^۱Dual decomposition

کامل ارضاء نشوند. بنابراین از این راه حل نیز نمی‌توان استفاده کرد.

Algorithm 3.1 Decentralized MILP from [32]

```

1: Input: Matrix A defined in (3.29)
2: Output: Lagrangian coefficient  $\lambda$  defined in (3.30), variable  $x$ 
3: procedure
4:    $\lambda(0) = 0$ 
5:    $\bar{s}_i(0) = -\infty, i = 1, \dots, m$ 
6:    $\underline{s}_i(0) = +\infty, i = 1, \dots, m$ 
7:    $k = 0$ 
8:   repeat
9:     for  $i = 1$  to  $m$  do
10:     $x_i(k+1) \leftarrow \arg \min_{x_i \in \text{vert}(X_i)} (c_i^T + \lambda(k)^T A_i)x_i$ 
11:   end for
12:    $\bar{s}_i(k+1) = \max\{\bar{s}_i(k), A_i x_i(k+1)\}, i = 1, \dots, m$ 
13:    $\underline{s}_i(k+1) = \min\{\underline{s}_i(k), A_i x_i(k+1)\}, i = 1, \dots, m$ 
14:    $\rho_i(k+1) = \bar{s}_i(k+1) - \underline{s}_i(k+1), i = 1, \dots, m$ 
15:    $\rho(k+1) = p \max\{\rho_1(k+1), \dots, \rho_m(k+1)\}$ 
16:    $\lambda(k+1) = [\lambda(k) + \alpha(k)(\sum_{i=1}^m A_i x_i(k+1) - b + \rho(k+1)]_+$ 
17:    $k \leftarrow k + 1$ 
18:   until some stopping criterion is met.
19: end procedure

```

بنابراین در این بخش لازم است که الگوریتم (۱.۳) را بر روی مسئله خودمان اعمال کنیم. برای این کار با توجه به شکل مسئله رابطه (۲۹.۳) لازم است که تغییراتی را در صورت مسئله خودمان ایجاد کنیم. همانطور که قبل ذکر شد در مسئله ما دو قید به صورت متصل وجود دارد که عبارتند از قیدهای رابطه (۲۳.۳) و رابطه (۲۲.۳) اما مشکلی که هست این است که قید رابطه (۲۲.۳) به صورت تساوی است در حالی که با توجه به مسئله رابطه (۲۹.۳) لازم است که کلیه قیدهای متصل به صورت نامساوی باشند لذا لازم است که قید مربوطه تغییر کند، برای این کار این قید به صورت رابطه (۳۲.۳ب) و یا رابطه (۳۲.۳ج) بازنویسی

می شود.

$$\lambda_{t,s} = \sum_{e \in E} \beta_{t,s,e} + \sum_{f \in F} \beta_{t,s,f} + \sum_{c \in C} \beta_{t,s,c} \quad \forall t \in T, \forall s \in S \quad (\text{آ}۳۲.۳)$$

$$\lambda_{t,s} \leq \sum_{e \in E} \beta_{t,s,e} + \sum_{f \in F} \beta_{t,s,f} + \sum_{c \in C} \beta_{t,s,c} \leq \lambda_{t,s} + \epsilon \quad \forall t \in T, \forall s \in S \quad (\text{ب}۳۲.۳)$$

$$\lambda_{t,s} - \epsilon \leq \sum_{e \in E} \beta_{t,s,e} + \sum_{f \in F} \beta_{t,s,f} + \sum_{c \in C} \beta_{t,s,c} \leq \lambda_{t,s} + \epsilon \quad \forall t \in T, \forall s \in S \quad (\text{ج}۳۲.۳)$$

که در رابطه های فوق ϵ به عنوان یک عدد کوچک در نظر گرفته می شود.

درنهایت می توان گفت که فرم استاندارد شده مسئله که بتوان الگوریتم (۱.۳) را بر روی آن اعمال کرد

به صورت رابطه (۳۳.۳) نوشته می شود.

$$\begin{aligned}
& \min \sum_{t \in T} \sum_{e \in E} K_1 \pi_e \psi_{t,e} + K_2 \pi_e x_{t,e} \\
& + \sum_{t \in T} \sum_{f \in F} K_1 \pi_f \psi_{t,f} + K_2 \pi_f x_{t,f} \\
& + \sum_{t \in T} \sum_{c \in C} K_1 \pi_c \psi_{t,c} + K_2 \pi_c x_{t,c} \\
& \text{subj. to} \\
& - \sum_{e \in E} x_{t,e} - \sum_{f \in F} x_{t,f} - \sum_{c \in C} x_{t,c} \leq -1 \quad \forall t \in T \\
& \sum_{e \in E} x_{t,e} + \sum_{f \in F} x_{t,f} + \sum_{c \in C} x_{t,c} \leq N_t \quad \forall t \in T \\
& - \sum_{e \in E} \beta_{t,s,e} - \sum_{f \in F} \beta_{t,s,f} - \sum_{c \in C} \beta_{t,s,c} \leq -\lambda_{t,s} + \epsilon \quad \forall t \in T, \forall s \in S \\
& \sum_{e \in E} \beta_{t,s,e} + \sum_{f \in F} \beta_{t,s,f} + \sum_{c \in C} \beta_{t,s,c} \leq \lambda_{t,s} + \epsilon \quad \forall t \in T, \forall s \in S
\end{aligned}
\tag{3.6) (3.7) (3.12) (3.13) (3.19) (3.21) (۳۳.۳)
}$$

لاگرانژین صورت مسئله‌ی فوق به صورت رابطه (۳۴.۳) نوشته خواهد شد. در این رابطه متغیرهای مسئله به صورت برداری و ماتریسی نمایش داده شده‌اند، دو متغیر اصلی مسئله یعنی x و β به صورت ماتریس نمایش داده شده‌اند که به ترتیب ماتریس‌های دو بعدی و سه بعدی هستند. نحوه‌ی چینش اعضاء در این ماتریس‌ها لزوماً قرارداد خاصی ندارد، اما بعد از چینش به صورت یک قرارداد باید آن را تا آخرین مرحله حفظ کرد، به عنوان مثال می‌توان برای ماتریس مربوط به متغیر x ، سطرها را متناسب با وظیفه‌ها و ستون‌ها را متناسب با گره‌های پردازشی درنظر گرفت. کل گره‌های پردازشی را می‌توان به ترتیب از بالا به پایین، یعنی از لایه ابری به سمت لایه لبه در کنار هم قرارداد. در مورد ضرایب لاگرانژ نیز اتفاق مشابه رخ می‌دهد، با توجه به رابطه (۳۳.۳) می‌توان گفت که چهار دسته ضریب لاگرانژ لازم است که به ترتیب با اسم‌های η_1 , η_2 , ν_1 و ν_2 مشخص شده‌اند. نحوه چینش اعضاء این ماتریس‌ها و بردارها با توجه به قراردادهای انجام شده در مورد متغیرهای

مسئله، انجام می‌گیرد.

$$\begin{aligned}
 L(\underline{\underline{x}}, \underline{\underline{\beta}}, \underline{\eta_1}, \underline{\eta_2}, \underline{\underline{\nu_1}}, \underline{\underline{\nu_2}}) = & \sum_{t \in T} \sum_{e \in E} K_1 \pi_e \psi_{t,e} + K_2 \pi_e x_{t,e} \\
 & + \sum_{t \in T} \sum_{f \in F} K_1 \pi_f \psi_{t,f} + K_2 \pi_f x_{t,f} \\
 & + \sum_{t \in T} \sum_{c \in C} K_1 \pi_c \psi_{t,c} + K_2 \pi_c x_{t,c} \\
 & + \sum_{t \in T} \eta_{1,t} \left(1 - \sum_{e \in E} x_{t,e} - \sum_{f \in F} x_{t,f} - \sum_{c \in C} x_{t,c} \right) \\
 & + \sum_{t \in T} \eta_{2,t} \left(\sum_{e \in E} x_{t,e} + \sum_{f \in F} x_{t,f} + \sum_{c \in C} x_{t,c} - N_t \right) \\
 & + \sum_{t \in T} \sum_{s \in S} \nu_{1,t,s} (\lambda_{t,s} - \epsilon - \sum_{e \in E} \beta_{t,s,e} - \sum_{f \in F} \beta_{t,s,f} - \sum_{c \in C} \beta_{t,s,c}) \\
 & + \sum_{t \in T} \sum_{s \in S} \nu_{2,t,s} (\sum_{e \in E} \beta_{t,s,e} + \sum_{f \in F} \beta_{t,s,f} + \sum_{c \in C} \beta_{t,s,c} - \lambda_{t,s} - \epsilon) \quad (۳۴.۳)
 \end{aligned}$$

در رابطه (۳۵.۳) سعی شده است که کل تابع موجود بر روی گرهای پردازشی تجزیه شود. بدین صورت که هر گره پردازشی تابع مخصوص به خود را داشته باشد.

$$\begin{aligned}
 L = & \sum_{e \in E} \sum_{t \in T} (x_{t,e} (K_2 \pi_e - \eta_{1,t} + \eta_{2,t}) + K_1 \pi_e \psi_{t,e} + \sum_{s \in S} \beta_{t,s,e} (\nu_{2,t,s} - \nu_{1,t,s})) \\
 & + \sum_{f \in F} \sum_{t \in T} (x_{t,f} (K_2 \pi_f - \eta_{1,t} + \eta_{2,t}) + K_1 \pi_f \psi_{t,f} + \sum_{s \in S} \beta_{t,s,f} (\nu_{2,t,s} - \nu_{1,t,s})) \\
 & + \sum_{c \in C} \sum_{t \in T} (x_{t,c} (K_2 \pi_c - \eta_{1,t} + \eta_{2,t}) + K_1 \pi_c \psi_{t,c} + \sum_{s \in S} \beta_{t,s,c} (\nu_{2,t,s} - \nu_{1,t,s})) \\
 & + \sum_{t \in T} (\eta_{1,t} - N_t \eta_{2,t} + \sum_{s \in S} \lambda_{t,s} (\nu_{1,t,s} - \nu_{2,t,s}) - \epsilon (\nu_{1,t,s} + \nu_{2,t,s})) \quad (\tilde{۳}۵.۳)
 \end{aligned}$$

$$L = L_0 + \sum_{e \in E} L_e + \sum_{f \in F} L_f + \sum_{c \in C} L_c \quad (\text{ب} ۳۵.۳)$$

Algorithm 3.2 Decentralized milp from [32]

```

1: procedure
2:    $\eta_1(0) = 0$ 
3:    $\eta_2(0) = 0$ 
4:    $\nu_1(0) = 0$ 
5:    $\nu_2(0) = 0$ 
6:    $\bar{s}_{\eta_1,i}(0) = -\infty, i = 1, \dots, l$ 
7:    $\underline{s}_{\eta_1,i}(0) = +\infty, i = 1, \dots, l$ 
8:    $\bar{s}_{\eta_2,i}(0) = -\infty, i = 1, \dots, l$ 
9:    $\underline{s}_{\eta_2,i}(0) = +\infty, i = 1, \dots, l$ 
10:   $\bar{s}_{\nu_1,i}(0) = -\infty, i = 1, \dots, l$ 
11:   $\underline{s}_{\nu_1,i}(0) = +\infty, i = 1, \dots, l$ 
12:   $\bar{s}_{\nu_2,i}(0) = -\infty, i = 1, \dots, l$ 
13:   $\underline{s}_{\nu_2,i}(0) = +\infty, i = 1, \dots, l$ 
14:  k = 0
15: repeat
16:   for  $i = 1$  to  $l$  do
17:      $x_i(k+1), \beta_i(k+1), \psi_i(k+1) \leftarrow \arg \min_{x_i, \beta_i, \psi_i \in \text{vert}(X_i)} \sum_{t \in T} (x_i(K_2\pi_i - \eta_{1,t} +$ 
         $\eta_{2,t}) + K_1\pi_i\psi_{t,i} + \sum_{s \in S} \beta_{t,s,i}(\nu_{2,t,s} - \nu_{1,t,s}))$ 
      subj. to
      (3.6) (3.7) (3.12) (3.13) (3.19) (3.21)
18:      $\bar{s}_{\eta_1,i}(k+1) = \max\{\bar{s}_{\eta_1,i}(k), -x_{t,i}(k+1)\}$ 
19:      $\underline{s}_{\eta_2,i}(k+1) = \max\{\bar{s}_{\eta_2,i}(k), +x_{t,i}(k+1)\}$ 
20:      $\bar{s}_{\nu_1,i}(k+1) = \max\{\bar{s}_{\nu_1,i}(k), -\beta_{t,s,i}(k+1)\}$ 
21:      $\underline{s}_{\nu_2,i}(k+1) = \max\{\bar{s}_{\nu_2,i}(k), +\beta_{t,s,i}(k+1)\}$ 
22:      $\underline{s}_{\eta_1,i}(k+1) = \min\{\underline{s}_{\eta_1,i}(k), -x_{t,i}(k+1)\}$ 
23:      $\underline{s}_{\eta_2,i}(k+1) = \min\{\underline{s}_{\eta_2,i}(k), +x_{t,i}(k+1)\}$ 
24:      $\underline{s}_{\nu_1,i}(k+1) = \min\{\underline{s}_{\nu_1,i}(k), -\beta_{t,s,i}(k+1)\}$ 
25:      $\underline{s}_{\nu_2,i}(k+1) = \min\{\underline{s}_{\nu_2,i}(k), +\beta_{t,s,i}(k+1)\}$ 
26:      $\rho_{\eta_1,i}(k+1) = \bar{s}_{\eta_1,i}(k+1) - \underline{s}_{\eta_1,i}(k+1)$ 
27:      $\rho_{\eta_2,i}(k+1) = \bar{s}_{\eta_2,i}(k+1) - \underline{s}_{\eta_2,i}(k+1)$ 
28:      $\rho_{\nu_1,i}(k+1) = \bar{s}_{\nu_1,i}(k+1) - \underline{s}_{\nu_1,i}(k+1)$ 
29:      $\rho_{\nu_2,i}(k+1) = \bar{s}_{\nu_2,i}(k+1) - \underline{s}_{\nu_2,i}(k+1)$ 
30:   end for

```

```

31:    $\rho_{\eta_1}(k+1) = |T| \max\{\rho_{\eta_1,1}(k+1), \dots, \rho_{\eta_1,l}(k+1)\}$ 
32:    $\rho_{\eta_2}(k+1) = |T| \max\{\rho_{\eta_2,1}(k+1), \dots, \rho_{\eta_2,l}(k+1)\}$ 
33:    $\rho_{\nu_1}(k+1) = |S||T| \max\{\rho_{\nu_1,1}(k+1), \dots, \rho_{\nu_1,l}(k+1)\}$ 
34:    $\rho_{\nu_2}(k+1) = |S||T| \max\{\rho_{\nu_2,1}(k+1), \dots, \rho_{\nu_2,l}(k+1)\}$ 
35:    $\eta_1(k+1) = [\eta_1(k) + \alpha(k)(1 - \sum_{e \in E} x_{t,e}(k+1) - \sum_{f \in F} x_{t,f}(k+1) - \sum_{c \in C} x_{t,c}(k+1) + \rho_{\eta_1}(k+1))]_+$ 
36:    $\eta_2(k+1) = [\eta_2(k) + \alpha(k)(\sum_{e \in E} x_{t,e}(k+1) + \sum_{f \in F} x_{t,f}(k+1) + \sum_{c \in C} x_{t,c}(k+1) - N_t + \rho_{\eta_2}(k+1))]_+$ 
37:    $\nu_1(k+1) = [\nu_1(k) + \alpha(k)(\lambda_{t,s} - \epsilon - \sum_{e \in E} \beta_{t,s,e}(k+1) - \sum_{f \in F} \beta_{t,s,f}(k+1) - \sum_{c \in C} \beta_{t,s,c}(k+1) + \rho_{\nu_1}(k+1))]_+$ 
38:    $\nu_2(k+1) = [\nu_2(k) + \alpha(k)(\sum_{e \in E} \beta_{t,s,e}(k+1) + \sum_{f \in F} \beta_{t,s,f}(k+1) + \sum_{c \in C} \beta_{t,s,c}(k+1) - \lambda_{t,s} - \epsilon + \rho_{\nu_2}(k+1))]_+$ 
39:    $k \leftarrow k+1$ 
40: until some stopping criterion is met.
41: end procedure

```

حال می‌توانیم الگوریتم (۱.۳) را بر روی مسئله خودمان اعمال کنیم همچنین در الگوریتم پیشنهادی حاصل جهت بهینه کردن میزان محاسبات تغییراتی داده شده است بدین صورت که در الگوریتم (۱.۳) یک حلقه وجود دارد که متغیرهای محلی را حساب می‌کند و در مرحله بعد سه حلقه‌ی دیگر به همان اندازه استفاده می‌شود که متغیر اضافه شده به قیدها حساب شوند اما در الگوریتم (۲.۳) که الگوریتم پیشنهادی است تمام این محاسبات در یک حلقه انجام می‌شود. در الگوریتم (۲.۳) بعضی متغیرها به صورت برداری هستند، اما جهت خوانایی بیشتر از گذاشتن علامت بردار و یا ماتریس بر روی متغیرها صرف نظر شده است. برخی قراردادهای جدید در رابطه (۳۶.۳) آورده شده است که از آن‌ها در الگوریتم (۲.۳) استفاده شده است.

$$l = |E| + |F| + |C| \quad (36.3)$$

۴.۳ بررسی همگرایی و پیچیدگی

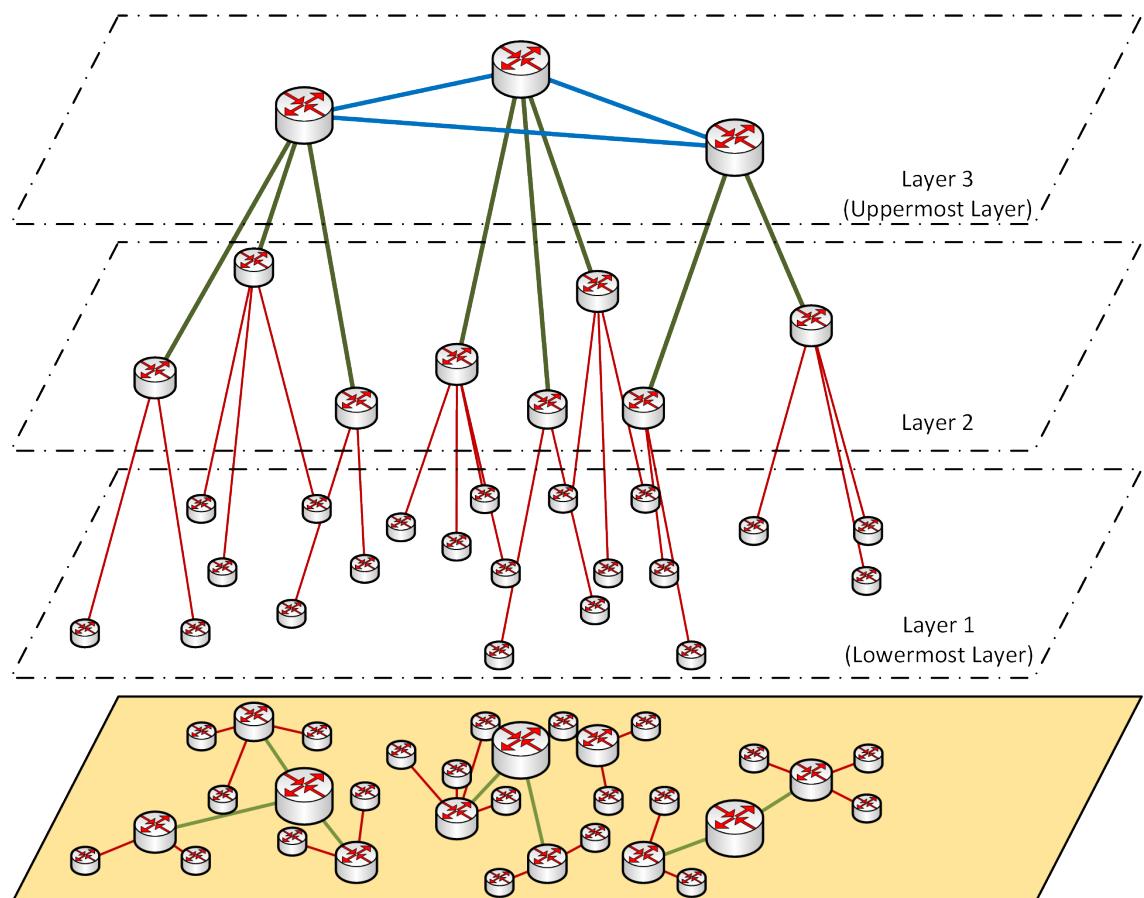
در این بخش ابتدا لازم است که در مورد همگرایی روش غیرمتمرکز و همچنین میزان پیچیدگی حل مسئله در دو روش ذکر شده صحبت شود. با توجه به [۳۴] می‌توان گفت که الگوریتم (۱.۳) و درنتیجه الگوریتم (۲.۳) به جواب بهینه همگرا خواهد شد.

همانطور که می‌دانیم مسئله خطی ترکیبی عدد صحیح از نوع مسئله NP-hard است که در مدت زمان

فصل ۳. تخصیص منابع پردازشی در شبکه اینترنت اشیاء به صورت متتمرکز و غیرمتتمرکز

چندجمله‌ای قابل حل نیست. اما به صورت سرانگشتی در مورد میزان پیچیدگی حل مسئله، تعداد کل وظیفه‌ها $|T|$ است و تعداد کل گره‌های محاسباتی $|N_t|$ است، همچنین حداکثر تعداد قابل تقسیم برای هر وظیفه t است؛ آنگاه میزان پیچیدگی محاسباتی برای یافتن جواب به کمک روش جستجو فراگیر^۱ از حدود $O(l^{(|T||N_t|)})$ خواهد بود. همانطور که دیده می‌شود میزان پیچیدگی نسبت به تعداد وظیفه‌ها به صورت نمایی است. در [۳۴] نویسنده‌گان نشان داده‌اند که مدت زمان آماده شدن جواب نسبت به توپولوژی مسئله خطی است.

۵.۳ نتایج شبیه‌سازی

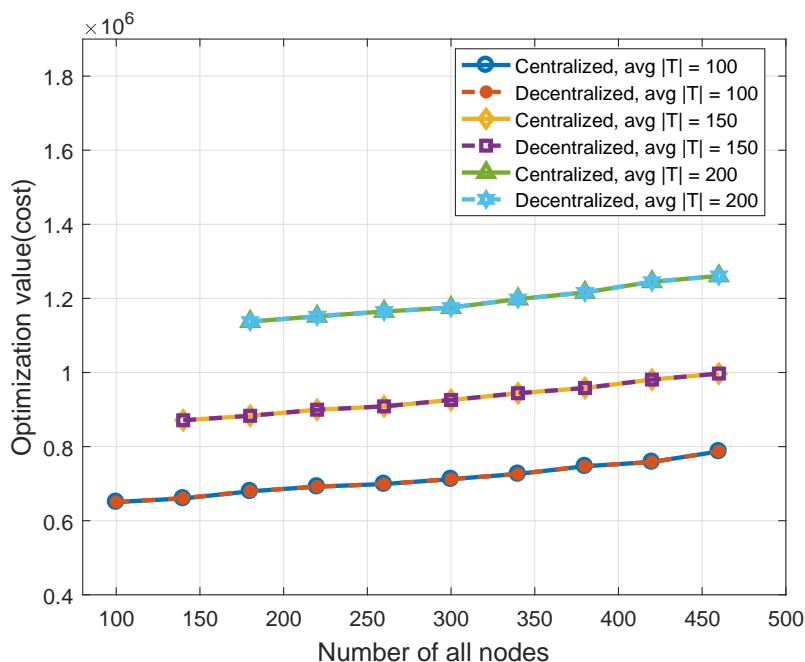


شکل ۲.۳: دید کلی از توپولوژی شبکه

در این قسمت نتایج شبیه‌سازی را بیان می‌کنیم. در همه موارد این فصل و فصل بعد نتایج برای ۲۰۰

^۱Brute force

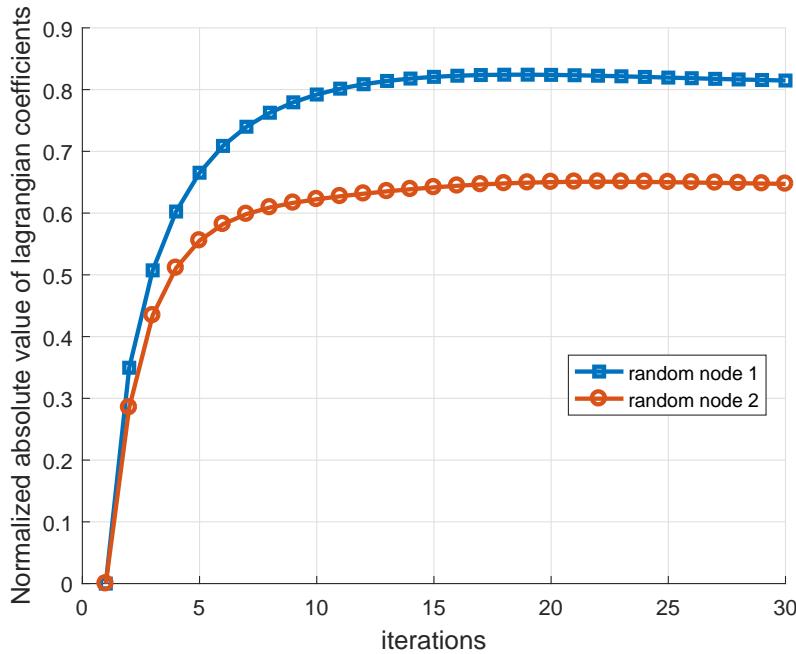
بار اجرا شده‌اند و نتایج به صورت میانگین بیان شده‌اند. همانند مدل سیستم یک شبکه چهار لایه به صورت شکل (۲.۳) در نظر می‌گیریم، تعداد گره‌های هر لایه هر بار به صورت تصادفی انتخاب می‌شوند، تعداد وظیفه‌ها و همچنین نرخ تولید وظیفه‌ها به صورت تصادفی در یک محدوده عددی معقول تولید می‌شوند.



شکل ۳.۳: مقدار تابع هدف(هزینه) دربرابر تعداد کل گره‌های موجود در شبکه برای دو روش متمرکز و غیرمتمرکز

برای تاخیر مسیریاب‌ها فرض می‌کنیم که به صورت میانگین عبور بسته‌ها از مسیریاب‌های لایه ۱، ۲ میلی ثانیه، مسیریاب‌های لایه ۲، ۴ میلی ثانیه و مسیریاب‌های لایه ۳، ۱۰ میلی ثانیه طول می‌کشد. شکل (۲.۳) توپولوژی شبکه را که در شبیه‌سازی استفاده شده است نشان می‌دهد.

ظرفیت مربوط به هریک از گره‌ها به نجوم انتخاب شده است که بیشترین ظرفیت مربوط به گره‌های ابری و کمترین مربوط به گره‌های لبه باشد. که به ترتیب به نسبت اعداد ۱۰، ۹ و ۸ درنظر گرفته شده است. حجم پردازنده مورد نیاز برای وظیفه‌ها نیز به صورت تصادفی در محدوده عدد ۴۰ واحد در نظر گرفته شده‌اند. حداقل زمان لازم جهت پردازش وظیفه‌ها به صورت تصادفی و با میانگین حدود ۱۵ میلی ثانیه فرض شده است. واحد قیمت پردازشی برای گره‌ها به صورتی در نظر گرفته شده است که ارزان‌ترین هزینه مربوط به گره‌های لایه ابری و گرانترین مربوط به گره‌های لایه لبه باشد، این اعداد به صورت تصادفی و به ترتیب به نسبت اعداد



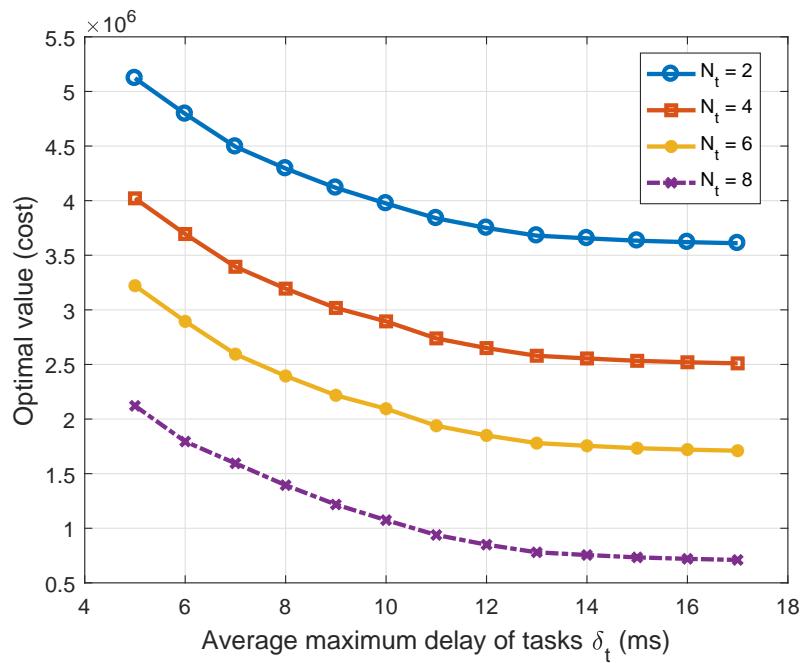
شکل ۴.۳: نحوه همگرا شدن اندازه ضرایب لگرانژین در دو گره تصادفی در روش غیرمتتمرکز

۱، ۴ و ۹ واحد در نظر گرفته شده است. نرخ پوآسون تولید وظیفه در گره های حسگری به صورت تصادفی و در حدود ۵۰ واحد بر ثانیه برای هر وظیفه در هر گره حسگری درنظر گرفته شده است.

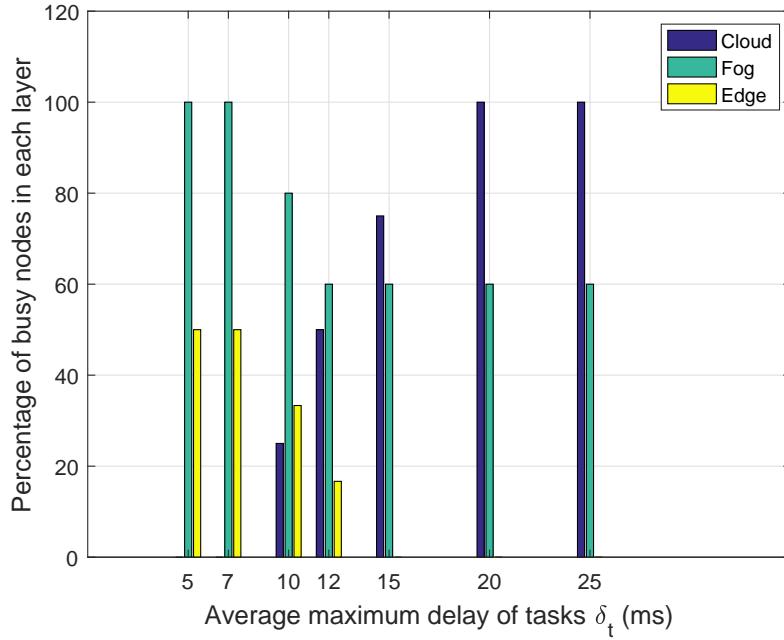
ابتدا لازم است که دقت دو روش متتمرکز و غیرمتتمرکز مقایسه شود. در شکل (۳.۳) مقدار تابع هدف در این دو روش در برابر تعداد کل گره های موجود در شبکه از جمله گره های حسگری و گره های پردازشی دیده می شود. همانطور که دیده می شود این نمودار برای سه حالت مختلف از تعداد وظیفه ها رسم شده است و در هر سه حالت مقدار بهینه در روش متتمرکز و غیرمتتمرکز یکسان شده است. با افزایش تعداد کل وظیفه ها مقدار تابع هدف افزایش می یابد، همچنین با زیاد شدن کل گره های موجود در شبکه با فرض ثابت بودن تعداد وظیفه ها انتظار می رود که میزان تابع هدف تقریباً ثابت باشد، اما به دلیل اینکه با افزایش تعداد گره ها، تعداد گره های حسگری نیز افزایش می یابد، مقدار بهینه نیز اندکی افزایش می یابد.

در مورد روش غیرمتتمرکز و نحوه همگرا شدن آن، دو گره به صورت تصادفی انتخاب شده است و در این دو گره مجموع اندازه ضرایب لگرانژ یعنی $\sqrt{|\eta_1|^2 + |\eta_2|^2 + |\nu_1|^2 + |\nu_2|^2}$ به صورت یکه شده در طول زمان رسم شده است. در شکل (۴.۳) می توانید این نتایج را ببینید.

در شبیه سازی بعدی تابع هدف بر اساس حداکثر تاخیر های قابل قبول متفاوت برای وظیفه ها رسم شده

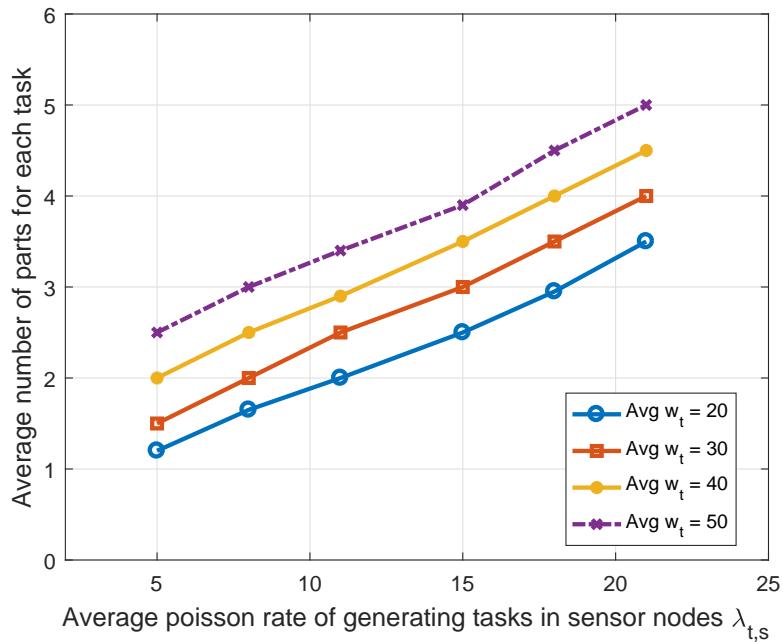


شکل ۵.۳: مقدار تابع هدف(هزینه) دربرابر میانگین حداکثر تاخیر قابل قبول برای وظیفه‌ها δ_t



شکل ۶.۳: درصد استفاده از گره‌های لایه‌های مختلف دربرابر میانگین حداکثر تاخیر قابل قبول برای وظیفه‌ها δ_t

است. با توجه به شکل (۵.۳) می‌توان گفت که با افزایش میانگین حداکثر تاخیر قابل قبول برای وظیفه‌ها (δ_t) هزینه کل شبکه کاهش می‌یابد که دلیل این موضوع این است که استفاده از گره‌های ابری نسبت به گره‌های مه و لبه بیشتر می‌شود و از آنجایی که هزینه استفاده از گره‌های ابری کمتر است، هزینه کل شبکه کاهش می‌یابد. این نمودار برای چهار حالت مختلف از حداکثر تعداد قابل قبول برای تقسیم شدن وظیفه‌ها (N_t) رسم شده‌است. با توجه به شکل (۳.۳) با کاهش N_t این امکان کمتر می‌شود که وظیفه‌ها بین گره‌های مختلف شکسته شوند و هر وظیفه باید به صورت بخش‌های بزرگتری پردازش شود، درنتیجه استفاده از گره‌های مه و لبه بیشتر می‌شود و همین باعث می‌شود که هزینه کل شبکه بیشتر شود.



شکل ۷.۳: میانگین تعداد شکسته شدن وظیفه‌ها بین گره‌های مختلف دربرابر میانگین نرخ تولیدی وظیفه‌ها در گره‌های حسگری $\lambda_{t,s}$

در شبیه سازی بعدی میزان استفاده از گره‌های لایه‌های مختلف مورد تحلیل و بررسی قرار می‌گیرد. در شکل (۶.۳) میانگین حداکثر تاخیر قابل قبول برای وظیفه‌ها در محور افقی قرار گرفته است و برای هر تاخیر، درصد استفاده از گره‌های هر لایه در محور عمودی نشان داده شده است. همانطور که دیده می‌شود با کاهش حداکثر میزان تاخیر قابل قبول لازم است که وظیفه‌ها بیشتر در سمت لبه پردازش شوند و این درصد استفاده از گره‌های این لایه را بیشتر می‌کند و بالعکس.

در آخرین شبیه‌سازی در این فصل قرار است که میزان شکسته شدن وظیفه‌ها بین گره‌های مختلف مورد بررسی قرار گیرد. برای این‌کار میزان نرخ تولید وظیفه‌ها در گره‌های حسگری تغییر کرده است و در هر مرحله میانگین تعداد شکسته شدن گره‌ها مورد بررسی قرار گرفته است. با توجه به شکل (۷.۳) می‌توان گفت که با افزایش میزان نرخ تولید وظیفه‌ها احتمال شکسته شدن وظیفه‌ها بین گره‌ها بیشتر می‌شود و میانگین تعداد تقسیم وظیفه‌ها افزایش می‌یابد. این نمودار برای میانگین حجم پردازشی وظیفه‌ها رسم شده است، که می‌توان گفت با افزایش میانگین حجم پردازشی، پردازش وظیفه‌ها سخت‌تر و درنتیجه میانگین تعداد تقسیم‌ها بیشتر می‌شود.

۶.۳ جمع‌بندی و نتیجه‌گیری

در این فصل مدل سیستم برای مسئله تخصیص منابع پردازشی نوشته شد و مورد بررسی قرار گرفت. این مسئله غیرخطی بود که برای راحت‌تر شدن حل آن خطی‌سازی شد. درادامه مسئله به کمک روش متمرکز حل شد و در نهایت یک الگوریتم برای حل مسئله به صورت غیرمتمرکز ارائه شد. هر دو روش گفته شده به جواب بهینه می‌رسند با این تفاوت که روش غیرمتمرکز در زمان چندجمله‌ای به جواب می‌رسد.

فصل ۴

راه حل های اکتشافی و توزیع شده

۱.۴ مقدمه

در فصل (۳) مدل سیتم به صورت کامل توضیح داده شد، مسئله بهینه سازی تبیین شد و پس از خطی سازی، ابتدا به روش متمرکز و سپس در بخش (۳.۳) به روش غیر متمرکز بررسی و حل شد. در این فصل قرار است دو راه حل دیگر برای مدل سیستم ارائه شده در فصل (۳) ارائه شود. در قسمت اول قرار است یک راه حل اکتشافی^۱ ارائه و بررسی گردد، مزیت این راه حل این است که نیازی نیست مسئله به صورت خطی تبیین شود، اما عیب آن نسبت به دو راه حل ارائه شده در فصل (۳) این است که این راه حل جواب زیر بهینه^۲ ارائه می‌دهد. این راه حل اکتشافی مبتنی بر یکی از الگوریتم‌های معروف به نام ویتربی^۳ است، که در [۳۵] تبیین شده است. در قسمت بعد از این فصل یک راه حل توزیع شده مورد بررسی قرار می‌گیرد، این راه حل از مدل خطی شده در فصل (۳) یعنی رابطه (۲۸.۳) استفاده می‌کند. این راه حل به صورت کاملاً توزیع شده بین گره‌های شبکه و بدون نیاز به هیچ واحد سومی^۴ ارائه می‌شود.

در ادامه این فصل، ابتدا مقدمه‌ای در مورد الگوریتم ویتربی گفته می‌شود، سپس مسئله بهینه سازی اصلی به صورتی که در این بخش قابل استفاده باشد بازنویسی می‌شود و درنهایت الگوریتم اکتشافی مربوطه ارائه

¹Heuristic

²Sub-optimal

³Viterbi

⁴Third-party

می شود. در ادامه راه حل توزیع شده^۱ مورد بررسی قرار می گیرد. درنهایت پس از بررسی همگرایی و پیچیدگی دو راه حل ارائه شده در این فصل، نتایج مربوط به این دو راه حل و همچنین در مقایسه با راه حل های ارائه شده در فصل (۳) نمایش داده می شود و جمع بندی لازم ارائه می گردد.

۲.۴ مقدمه ای بر ویتربی

ابتدا لازم است که در مورد الگوریتم ویتربی و نحوه کار آن صحبت شود. این الگوریتم یک الگوریتم پویا^۲، برای پیدا کردن محتمل ترین مسیر از حالت های پنهان، با داشتن یک توالی از مشاهدات است. این الگوریتم اغلب در مواردی به کار می رود که با داشتن یک مدل پنهان مارکف و توالی از مشاهدات، می خواهیم بدانیم چه توالی از حالت ها (مسیر) این مشاهدات را تولید کرد هاند. به عبارت دیگر ما به دنبال محتمل ترین مسیر به وجود آورنده مشاهدات در یک مدل پنهان مارکف هستیم.

به عنوان ابتدایی ترین پاسخ می توانیم تمامی مسیر های ممکن که مشاهده ما را تولید می کنند، پیدا کنیم، سپس با محاسبه احتمال آنها، محتمل ترین مسیر را بدست آوریم اما می توان نشان داد که پیچیدگی زمانی این راه حل نسبت به طول توالی n ، از اندازه نمایی ($O(a^n)$) است. الگوریتم ویتربی یک راه حل ارائه می دهد که هزینه محاسباتی آن به صورت خطی با طول توالی افزایش می یابد، اصطلاحا هزینه آن از اندازه چندجمله ای است. فرم نهایی مسئله را می توان بدین صورت نوشت: می خواهیم مسیر بهینه ای مانند π^* را $\pi = \arg \max_{\pi} P(x, \pi)$ داشته باشیم، که در آن $x = (x_1, \dots, x_L)$ یکه توالی از مشاهدات و (π_1, \dots, π_L) یک توالی از متغیرهای پنهان باشد.

نحوه عملکرد این الگوریتم بدین صورت است که مسئله را به صورت یک گراف چند مرحله ای^۳ در نظر می گیرد. تعداد مرحله^۴ ها برابر است با تعداد مشاهدات و در هر مرحله یکی از مشاهدات مورد بررسی قرار می گیرد. در هر مرحله تعدادی حالت^۵ درنظر گرفته می شود که هر حالت یک پیشامد ممکن در آن مرحله را بیان می کند. حال لازم است که دو مدل تابع هزینه^۶ تعریف شود یک تابع به عنوان هزینه بودن در هر حالت در

¹Distributed

²Dynamic

³Multistage

⁴Stage

⁵State

⁶Cost function

هر مرحله و یک تابع به عنوان هزینه انتقال از یک حالت در یک مرحله به یک حالت دیگر در مرحله بعد است، در واقع این دوتابع به نحوی باید مکمل یکدیگر باشند. در هر مرحله، برای هریک از حالت‌های آن مرحله یک مسیر به عنوان مسیر نجات‌یافته^۱ تعریف می‌شود که برابر است با مسیری از اولین مرحله تا مرحله فعلی که کمترین هزینه ممکن را دارد. در آخرین مرحله نیز هریک از حالت‌ها یک مسیر نجات‌یافته دارد، در این مرحله حالتی که کمترین هزینه ممکن را دارد، مسیر نجات‌یافته‌اش به عنوان مسیر ویتربی^۲ مشخص می‌شود که این مسیر در واقع پاسخ مسئله است.

۳.۴ راه حل اکتشافی مبتنی بر ویتربی (VTP)

در این بخش قرار است با توجه به مقدمه‌ی گفته شده متغیرهای مربوط به الگوریتم ویتربی را برای مسئله اصلی تعریف کنیم. ابتدا صورت مسئله اصلی که به صورت غیرخطی است در رابطه (۱.۴) آورده می‌شود.

$$\begin{aligned}
 & \min \sum_{t \in T} \sum_{e \in E} K_1 \pi_e x_{t,e} \lambda_{t,e} + K_2 \pi_e x_{t,e} \\
 & + \sum_{t \in T} \sum_{f \in F} K_1 \pi_f x_{t,f} \lambda_{t,f} + K_2 \pi_f x_{t,f} \\
 & + \sum_{t \in T} \sum_{c \in C} K_1 \pi_c x_{t,c} \lambda_{t,c} + K_2 \pi_c x_{t,c} \\
 & \text{subj. to (3.6) (3.7) (3.8)} \\
 & (3.17) (3.20) (3.22) (3.23) \tag{1.4}
 \end{aligned}$$

حال لازم است که متغیرهای مربوط به الگوریتم ویتربی متناسب با مسئله ما تعریف شوند، می‌دانیم که هدف از مسئله ما این است که تعدادی وظیفه به تعدادی گره پردازشی ارسال شوند؛ در واقع قرار است هر کدام از این وظیفه‌ها در یک یا چند گره پردازشی، پردازش شوند، با این هدف که کل هزینه موجود در شبکه کمترین مقدار ممکن باشد. حال می‌توانیم اینگونه فرض کنیم که هر وظیفه یک مشاهده است و هر گره پردازشی نیز یک پیشامد است یعنی هر وظیفه را به عنوان یک مرحله در نظر بگیریم و در هر مرحله، هر حالت برابر

¹Survived path

²Viterbi path

است با یک گره پردازشی، پس درواقع با توجه به رابطه (۳۶.۳) / حالت مختلف داریم. در نهایت الگوریتم ویتربی باید طوری طراحی شود که مسیر ویتربی نشان دهنده این باشد که هر وظیفه در کدام گره پردازش شود یا درواقع مسیر ویتربی کمترین هزینه موجود در شبکه را نشان دهد. نکته ای که باید مدنظر قرار بگیرد قابلیت پردازش یک وظیفه در چند گره است، همانطور که قبل توضیح داده شد، این قابلیت بدین صورت است که هر وظیفه این امکان را دارد که شکسته شود و در دو یا چند گره پردازشی پردازش شود و حداکثر می تواند در N_t گره پردازش شود. برای اضافه شدن این قابلیت به الگوریتم مربوطه، در تعداد مرحله های الگوریتم یک تغییر کوچک ایجاد می کنیم به این صورت که به جای اینکه هر وظیفه را یک مرحله در نظر بگیریم، هر بخش شکسته شده از هر وظیفه را یک مرحله در نظر می گیریم، پس در نهایت $L_V = \sum_{t \in T} N_t$ مرحله خواهیم داشت. برای حالت های هر مرحله نیز لازم است که یک تغییر کوچک ایجاد شود؛ برای هر وظیفه در بخش اول، حالت های ممکن برابر است با همان حالت های قبل یعنی / حالت، اما در مورد بخش های بعدی یک حالت دیگر که حالت صفر (0) نام دارد در نظر گرفته می شود. تعداد حالت های ممکن با توجه به شماره هی مرحله در رابطه (۲۰.۴) دیده می شود.

$$Z_n = \begin{cases} \{0\} \cup S, & u = (n \mod N_t) = 1 \\ S, & \text{در غیر این صورت} \end{cases}, n = 1, \dots, L_V \quad (\tilde{A}2.4)$$

$$S = E \cup F \cup C \quad (2.4b)$$

بنابراین تا اینجای کار تعداد مرحله ها و همچنین حالت های هر مرحله مشخص شد. در ادامه لازم است که چند تابع مختلف تعریف شود یکی از آنها تابع هزینه انتقال از یک حالت در یک مرحله به یک حالت دیگر در مرحله بعد است، که به صورت رابطه (۳۳.۴) تعریف می شود.

$$\Theta_{n-1,n}^{i,j} = \Gamma_{t,u,j} + \phi_{n-1}^i \quad (\tilde{A}3.4)$$

$$u \in \{1, \dots, N_t\} \quad (3.4b)$$

در رابطه (۳۳.۴)، n نشان دهنده مرحله، t نشان دهنده وظیفه و u نشان دهنده شماره بخش مربوط به هر وظیفه است که درواقع برای وظیفه t عددی است بین یک تا N_t . i و j نشان دهنده حالت هستند. همچنین

ϕ_{n-1}^i هزینه‌ی بودن در مرحله $1 - n$ و حالت i است. متغیر Γ در رابطه (۲۵.۳) تعریف شده است.

در ادامه یک متغیر دیگر به نام $T_{n-1,n}^{i,j}$ تعریف می‌کنیم که نشان‌دهنده‌ی تاخیر سیستم برای زمانی است که وظیفه مورد نظر در گره j پردازش شود و الگوریتم ویتربی از حالت n در مرحله قبل به این حالت باید. این متغیر در رابطه (۴.۴) تعریف شده است.

$$T_{n-1,n}^{i,j} = \begin{cases} 0, & j = 0 \\ \tau_{t,u,j}, & \text{در غیر این صورت} \end{cases}, n = 1, \dots, L_V \quad (4.4)$$

در معادله بالا بعد از مشخص شدن وظیفه و گره پردازشی τ را به راحتی می‌توان به کمک رابطه (۱۶.۳ ب) محاسبه کرد. تنها قسمتی که لازم است در مورد آن صحبت شود مقدار λ در رابطه (۱۶.۳ ب) است. برای هر وظیفه نرخ کل جریان تولیدی در حسگرها مقداری مشخص است که لازم است در طی حل مسئله کل این نرخ تولیدی برای پردازش به گره‌های پردازشی برسد؛ پس در هر مرحله از الگوریتم لازم است مشخص شود چه حجمی از کل جریان تولیدی مورد بررسی قرار می‌گیرد. ایده‌ی به کار رفته در الگوریتم به این صورت است که ابتدا فرض می‌شود کل جریان تولیدی مربوط به هر وظیفه در مرحله مربوط به بخش اول از آن وظیفه در نظر گرفته می‌شود، در صورتیکه در یک مرحله هیچ گره‌ای پیدا نشود که منبع کافی برای پردازش کامل وظیفه را داشته باشد، آنگاه وظیفه شکسته می‌شود، یعنی بخشی از جریان تولیدی برای بخش‌های بعدی در نظر گرفته می‌شود و الگوریتم از بخش اول آن وظیفه دوباره اجرا می‌شود. جزئیات دقیق‌تری در مورد آن‌چه که گفته شد در الگوریتم (۱.۴) آمده است.

حال با توجه به متغیرهای تعریف شده برای هر حالت در هر مرحله می‌توان مسیر نجات یافته را به صورت رابطه (۵.۴) پیدا کرد.

$$I_n^j = \arg \min_{i \in XP_n^j} (D_{n-1,n}^{i,j}) \quad (5.4)$$

$$\begin{aligned} D_{n-1,n}^{i,j} &= \Theta_{n-1,n}^{i,j} + M_k \times (T_{n-1,n}^{i,j} - OD_n^j) \times I(T_{n-1,n}^{i,j} - OD_n^j) \\ &\quad + M_k \times (\lambda_{t,u,j} - \mu_{t,u,j}) \times I(\lambda_{t,u,j} - \mu_{t,u,j}) \end{aligned} \quad (6.4)$$

در رابطه (۵.۴) $D_{n-1,n}^{i,j}$ به عنوان معیار نهایی تصمیم‌گیری برای انتخاب مسیر بین حالت i از مرحله $1 - n$ و

حالات z از مرحله n در نظر گرفته می شود. I_n^j به عنوان شمارنده مسیر نجات یافته و XP_n^j به عنوان مجموعه می توان از مرحله $1 - n$ که منابع کافی جهت ورود به حالت z را دارند، در نظر گرفته می شود. گفت که $XP_n^j \subset Z_{n-1}$, یعنی گره هایی که منبع کافی جهت انتقال به حالت جدید را ندارند از کل مجموعه حالت های مرحله قبل حذف می شوند. M_k به عنوان یک هزینه سنگین در نظر گرفته می شود برای حالت های که تاخیر پردازش در آنها از حداقل تاخیر قابل قبول برای پردازش وظیفه بیشتر است. یک متغیر دیگر به نام OD_n^j دیده می شود، که می توان گفت بیانگر تاخیر درخواستی^۱ برای وظیفه مورد نظر است که در رابطه (۷.۴) آورده شده است در این رابطه این تاخیر برابر با میزان تاخیر درخواستی هر وظیفه در نظر گرفته شده است، در حالیکه می توانیم برای اطمینان بیشتر این تاخیر را اندکی کمتر از تاخیر درخواستی وظیفه در آن مرحله در نظر بگیریم که از همگرا شدن الگوریتم اطمینان بیشتری حاصل کنیم. قید مربوط به پایداری صفت در گره ها نیز در این عبارت گنجانده شده است، به صورتی که اگر در یک حالت از یک مرحله مقدار متغیرها به گونه ای باشد که باعث شود صفت موجود در گره ناپایدار شود، برای ورود به آن حالت هزینه زیادی در نظر گرفته می شود.

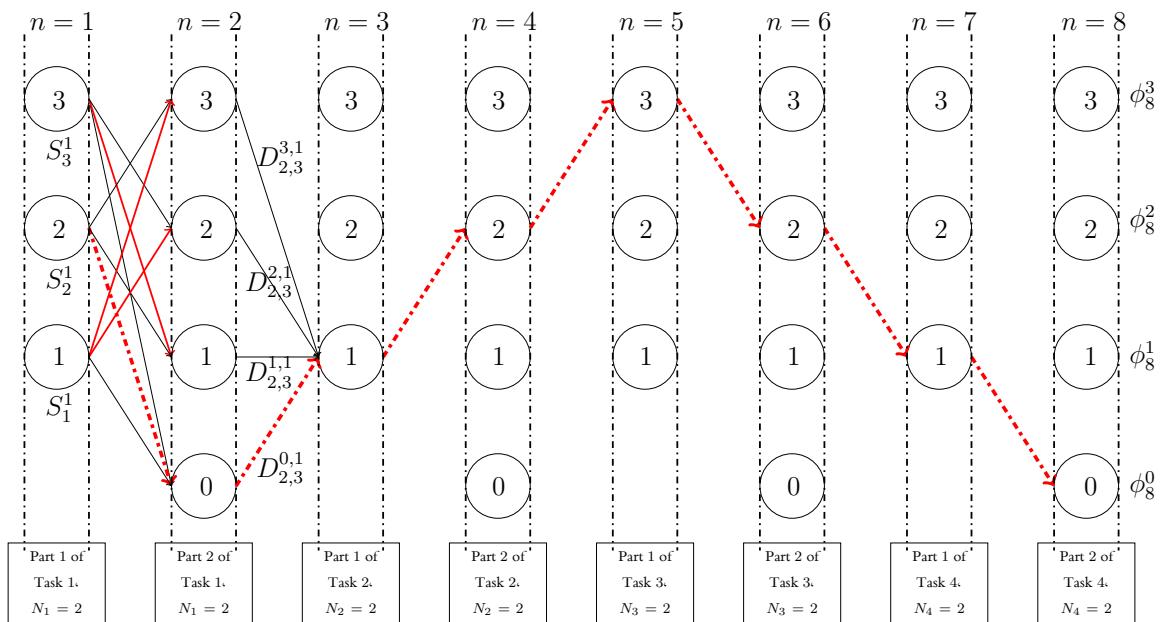
$$OD_n^j = \begin{cases} \infty, & j = 0 \\ \delta_t, & \text{در غیر این صورت} \end{cases} \quad (7.4)$$

در هر مرحله از الگوریتم لازم است که مسیر نجات یافته را برای هر حالت ذخیره کنیم برای این کار از یک بردار به نام Λ استفاده می کنیم، که به صورت رابطه (۸.۴) بروزرسانی می شود.

$$\Lambda_n^j[u] = \begin{cases} \Lambda_n^{I_n^j}[u], & 1 \leq u \leq n-1 \\ j, & u = n \end{cases} \quad (8.4)$$

عملکرد رابطه (۸.۴) بدین صورت است که در هر حالت پس از مشخص شدن شمارنده مربوط به حالت نجات یافته، مسیر نجات یافته برای حالت فعلی برابر است به مسیر نجات یافته برای حالت نجات یافته، به اضافه خود حالت فعلی. در قسمت بعد لازم است که هزینه بودن در هر حالت بعد از مشخص شدن مسیر

^۱Ordinary delay



شکل ۱.۴: مثالی از نحوه عملکرد الگوریتم (۱.۴)

نجات یافته بروزرسانی شود. این بروزرسانی به صورت رابطه (۹.۴) انجام می‌گیرد.

$$\phi_n^j = \Theta_{n-1,n}^{I_n^j,j} \quad (9.4)$$

در آخر لازم است که در هر مرحله بتوانیم مسیر ویتری را بیابیم، فرض کنیم که در مرحله H هستیم، در میان مسیرهای نجات یافته در این مرحله، مسیر ویتری را می‌توان به کمک رابطه (۱۰.۴) یافت.

$$P = \Lambda_H^\zeta \quad (\tilde{10.4})$$

$$\zeta = \arg \min_{i \in Z_H} \phi_H^i \times M_k \quad (10.4)$$

در حالتی که الگوریتم به خوبی تمام شود و تمام مراحل پیموده شوند می‌توان گفت که $H = L_V$.

در شکل (۱.۴) یک مثال کوچک از نحوه کار الگوریتم اکتشافی آورده شده است. در این مثال چهار وظیفه و سه گره پردازشی وجود دارد و حداقل تعداد قابل تقسیم برای هر وظیفه دو بخش است.

درنهایت راه حل ارائه شده برای یافتن مسیر ویتری به صورت کامل در الگوریتم (۱.۴) دیده می‌شود. پس از پیدا شدن مسیر ویتری لازم است که قیدهای مربوط به حداقل تاخیر قابل قبول برای وظیفه‌ها و پایداری

Algorithm 4.1 Viterbi-based Task Placement (VTP) Algorithm

```

1:  $Z_0 = 0, \phi_0^0 = 0, ResourceIndicator = true, H = L_V, NodeResource_0^0 = \sigma.$ 
2: for  $n = 1 : L_v$  do
3:   Determine the index of considered task,  $t$ , the index of considered part of task,  $u$ .
4:   Determine the set of states  $Z_n$  using (4.2)
5:    $RemovedStates = \{\}, RepeatIndicator = false$ 
6:   if  $u = 1$  then
7:     update backups
8:   end if
9:   for  $j \in Z_n$  do
10:     $XP_n^j = Z_{n-1}, OD_n^j = \infty$ 
11:    if  $j \neq 0$  then
12:      for  $i \in XP_n^j$  do
13:        if  $NodeResource_{n-1}^i < f_t^r(\lambda_{t,u,j})$  then
14:           $XP_n^j = XP_n^j - \{i\}$ 
15:        end if
16:      end for
17:       $OD_n^j = \delta_t$ 
18:    end if
19:    if  $XP_n^j \neq \emptyset$  then
20:      for  $i \in XP_n^j$  do
21:        Calculate  $T_{n-1,n}^{i,j}$  using (4.4)
22:        Calculate  $\Theta_{n-1,n}^{i,j}$  using (4.3)
23:      end for
24:      Calculate  $I_n^j$  using (4.5) and  $\Lambda_n^j$  using (4.8)
25:      Calculate  $\phi_n^j$  using (4.9)
26:      if  $j \neq 0$  then
27:         $NodeResource_n^j = NodeResource_n^j - f_t^r(\lambda_{t,u,j})$ 
28:      end if
29:    else
30:       $RemovedStates = RemovedStates + \{j\}$ 
31:    end if
32:  end for
33:  if  $|RemovedStates| > 0.7|Z_n|$  then
34:     $numOfIters[n] = numOfIters[n] + 1$ 
35:    if  $numOfIters[n] < 7$  then
36:       $RepeatIndicator = true$ 
37:      divide  $\lambda_{t,u,j}$  into smaller pieces.
38:      update parameters with backup data.
39:      break
40:    end if
41:  end if

```

```

42: if RemovedStates =  $Z_n$  then
43:     Set  $H = \sum_{m=1}^t N_m$  and ResourceIndicator = false
44:     Determine the Viterbi path  $P$ , using (4.10) with  $H$ .
45:     break
46: else
47:     Remove all states in the RemovedStates from the  $Z_n$ 
48: end if
49: end for
50: if RepeatIndicator = true then
51:     go to 2, repeat the loop from first part of last task.
52: end if
53: if ResourceIndicator = true then
54:     Determine the Viterbi path  $P$ , using (4.10) with  $H$ .
55: end if
56: Determine the task scheduling using Viterbi path  $P$ 

```

صف در گره های پردازشی مجددا چک شود و مسیر ویتری ب در صورتی قابل قبول است که این قیدها ارضاء شده باشند. الگوریتم لازم برای انجام این کار در الگوریتم (۲.۴) ارائه شده است.

Algorithm 4.2 Viterbi Path Scheduling

```

1: for  $i = 1 : H$  do
2:   Determine the index of considered task,  $t$ , the index of considered part of task,  $u$ .
3:   Determine the index of considered computational node  $n$ 
4:   if  $P(i) != 0$  then
5:      $x_{t,n} = 1$ 
6:      $\lambda_{t,n} = \lambda_{t,n} + \lambda_{t,u,n}$ 
7:   end if
8:   if  $(\tau_{t,u,n} > \delta_t)$  or  $(\lambda_{t,n} \geq \mu_{t,n})$  then
9:      $x_{t,n} = 0$ 
10:     $\lambda_{t,n} = 0$ 
11:   end if
12: end for

```

۴.۴ راه حل توزیع شده

در این بخش قرار است یک راه حل دیگر برای حل مسئله اصلی خطی شده یعنی رابطه (۲۸.۳) ارائه شود. در این راه حل مسئله بهینه سازی به طور کامل شکسته می شود و به صورت توزیع شده حل می شود. برای این کار از

الگوریتم ارائه شده در [۳۶] استفاده می شود. همانند راه حل های قبلی ابتدا مقدمه ای در مورد این الگوریتم و نحوی کار کردن آن گفته می شود، سپس همین الگوریتم برای مسئله اصلی نوشته می شود و درنهایت نتایج شبیه سازی به نمایش گذاشته می شود.

۱.۴.۴ الگوریتم توزیع شده

فرض کنید که مسئله بهینه سازی خطی ترکیب عدد صحیح به صورت رابطه (۱۱.۴) باشد.

$$\min_z c^T z \quad (11.4)$$

$$\text{subj. to} \quad a_i^T z \leq b_i, i = 1, \dots, n \quad (11.4)$$

$$z \in \mathbb{Z}^{d_Z} \times \mathbb{R}^{d_R} \quad (11.4)$$

$$d = d_Z + d_R \quad (11.4)$$

$$a_i \in \mathbb{R}^d, b_i \in \mathbb{R}, c \in \mathbb{R}^d \quad (11.4)$$

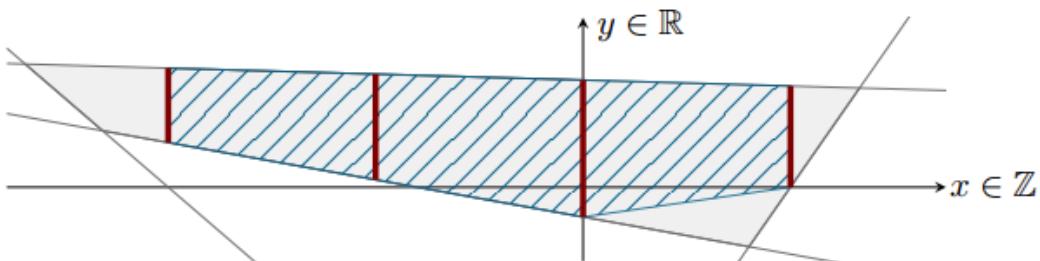
همانطور که مشخص است متغیر اصلی مسئله یعنی z از دو بخش گسته و پیوسته تشکیل شده است و همچنین n قید مختلف نیز وجود دارد. حال فرض کنید که شبکه‌ی گره‌ها شامل N گره باشد و میخواهیم مسئله فوق را به صورت توزیع شده توسط این گره‌ها حل کنیم. می‌توان نشان داد که در صورتی که $n \geq N$ آن‌گاه این مسئله به جواب می‌رسد یعنی تعداد گره‌ها از تعداد قیدها بیشتر نباشد، در این صورت می‌توان گفت که در هر گره حداقل یک قید بررسی می‌شود. در این صورت هر گره نیازی ندارد که در مورد قید یا قیدهای گره‌های دیگر چیزی بداند. گره‌های موجود در شبکه با یک گراف جهت دار کاملاً متصل مدل‌سازی می‌شود. در ادامه در مورد قیدهای موجود در مسئله رابطه (۱۱.۴) بیشتر صحبت خواهیم کرد، فرض کنید که کل قیدهای مسئله به صورت رابطه (۱۲.۴) نوشته شود با این تفاوت که متغیر اصلی مسئله کاملاً پیوسته در نظر گرفته شود. همانطور که می‌دانیم شکل هندسی این قید در فضای پیوشه d بعدی یک چندوجهی^۱ خواهد بود. حال اگر اشتراک این مجموعه و شرط گسته بودن بعضی از اعضای متغیر اصلی را در نظر بگیریم آن‌گاه یک مجموعه جدید به نام P_I شکل می‌گیرد که در رابطه (۱۳.۴) نشان داده شده است. همانطور که می‌دانیم

¹Polyhedron

مجموعه P_I لزوماً محدب نخواهد بود در نتیجه مسئله بهینه سازی که بر روی این مجموعه تعریف می شود محدب نخواهد بود، لذا برای بررسی مسائلی که مجموعه قیدهای آنها به صورت فوق باشد تلاش می شود به نجوى مجموعه قیدها محدب شود که بتوان از مزیتهای مربوط به مسائل بهینه سازی محدب استفاده برد. برای این کار معمولاً از مفهوم پوش محدب^۱ استفاده می شود. در شکل (۲.۴) یک مثال از چند وجهی و پوش محدب آورده شده است.

$$P := \bigcap_{i=1}^n \{z \in \mathbb{R}^d : a_i^T z \leq b_i\} \quad (12.4)$$

$$P_I = P \cap \{\mathbb{Z}^{d_Z} \times \mathbb{R}^{d_R}\} \quad (13.4)$$



شکل ۲.۴: مثالی از چند وجهی P که قسمت خاکستری رنگ است و همچنین مجموعه قابل قبول جواب مسئله یعنی P_I که با خطوط پرنگ قرمز مشخص شده است و مجموعه پوش محدب از جواب قابل قبول یعنی $\text{conv}(P_I)$ که ناحیه هاشور خورده آبی رنگ است. فضای استفاده شده به صورت دو بعدی است.

باتوجه به نکات گفته شده یک مسئله خطی پیوسته در رابطه (۱۴.۴) معرفی می کنیم که جواب بهینه این مسئله با جواب بهینه مسئله رابطه (۱۱.۴) یکسان خواهد بود. به صورت شهودی با توجه به شکل (۲.۴) می توان گفت که جواب یک مسئله بهینه سازی خطی در گوشه های منطقه قابل قبول^۲ آن خواهد بود. بنابراین

¹Convex hull

²Feasible region

نقطه بهینه^۱ در این دو مسئله یکسان خواهد بود.

$$\begin{aligned} & \min_z c^T z \\ & \text{subj. to } z \in \text{conv}(P_I) \\ & z \in \mathbb{R}^d \end{aligned} \tag{۱۴.۴}$$

با توجه به آنچه گفته شد از این ایده برای حل کردن مسائل بهینه‌سازی خطی ترکیب عدد صحیح می‌توان استفاده کرد، به این صورت که ابتدا مسئله را به صورت خطی پیوسته در نظر می‌گیرند و هر بار یک قید جدید به مسئله اضافه می‌کنند به صورتی که فضای قابل قبول برای مسئله کوچک و کوچکتر شود و در نهایت با شروع از P به $\text{conv}(P_I)$ می‌رسند.

در ادامه نحوه عملکرد این دسته از الگوریتم‌ها که به الگوریتم‌های صفحه برش^۲ معروف هستند به صورت کلی آورده می‌شود.

Algorithm 4.3 Centralized Cutting-Plane Meta-Algorithm

- 1: Initialization: $P := \bigcap_{i=1}^n \{z \in \mathbb{R}^d : a_i^T z \leq b_i\}$
 - 2: LP solver: Find an optimal solution $z^{LP} = (x^{LP}, y^{LP})$ of the LP relaxation of (4.11) with polyhedron P .
 - 3: Check feasibility: if $x^{LP} \in \mathbb{Z}^{d_Z}$, go to 6.
 - 4: Cutting-Plane: $h = \text{CUTORACLE}(z^{LP}, P, c)$.
 - 5: Update: $P := P \cap h$ and go to 2.
 - 6: Output: z^{LP} .
-

در الگوریتم (۳.۴) از مفهوم *CUTORACLE* استفاده شده است که نشان‌دهنده یک تابع است که با توجه به نقطه بهینه فعلی و آخرین چندوجهی مربوط به قیدها، یک قید جدید تولید می‌کند به طوریکه اشتراک چندوجهی قبلی و این قید جدید شامل پوش محدب مجموعه گسسته $\text{conv}(P_I)$ باشد اما شامل اخرين نقطه بهینه (x^{LP}, y^{LP}) نباشد.

در ادامه لازم است که چند مفهوم دیگر تعریف شود. یک جعبه محدود به صورت رابطه (۱۵.۴) تعریف

¹Optimal point

²Cutting-plane

می شود که در ادامه از آن استفاده خواهد شد. فرض می شود که $P \subset H_M$

$$H_M := \bigcap_{l=1}^d (\{z_l \leq M\} \cap \{z_l \geq -M\}) \quad (15.4)$$

یک مفهوم دیگر به نام پایه^۱ تعریف می شود، فرض کنید که یک مسئله بهینه سازی خطی پیوسته با مجموعه قید $P = \bigcap_{i=1}^n P_i$ داریم که هریک از P_i ها یک نیم فضای^۲ است، پایه B مجموعه تشکیل شده از تقاطع حداقل تعداد نیم فضای P_{l_1}, \dots, P_{l_q} است به صورتی که $d \leq q$ و جواب بهینه مسئله بهینه سازی بر روی مجموعه قید P با جواب همان مسئله بر روی مجموعه قید B برابر است. اگر برای حل کردن مسئله بهینه سازی از الگوریتم $LEXOPTIMAL$ که نوعی الگوریتم سیمپلکس^۳ است استفاده شود، می توان نشان داد که B دقیقاً از تقاطع d نیم فضای تشکیل می شود. الگوریتم $LEXOPTIMAL$ به تفصیل در [۳۷] توضیح داده شده است.

Algorithm 4.4 Distributed Meta-Algorithm

State $(z^{[i]}, B^{[i]})$

Initializing

1: $h^{[i]} = h_{i0} \cap H_M$

2: $(z^{[i]}, B^{[i]}) = LPLEXSOLV(h^{[i]}, c)$

Evolution

3: $(h_{MIG}(t), h_c(t)) = CUTORACLE(z^{[i]}(t), B^{[i]}(t), c)$

4: $H_{TMP}(t) = \left(\bigcap_{j \in N_i(t)} B^{[j]}(t) \right) \cap B^{[i]}(t) \cap h^{[i]} \cap h_{MIG}(t) \cap h_c(t)$

5: $(z^{[i]}(t+1), B^{[i]}(t+1)) = LPLEXSOLV(H_{TMP}, c)$

در الگوریتم (۴.۴) خروجی تابع CUTORACLE به صورت دو قسمتی است. که لازم است در مورد آن صحبت شود. در [۳۸] یک روش برش به نام MIG^۴ ارائه شده است و اثبات شده است که در صورتی که تابع هدف مسئله بهینه سازی به صورت گسسته باشد، این روش صفحه برش می تواند مسئله را حل کند. خروجی تابع که به صورت h_{MIG} نشان داده شده است، مربوط به این نحوه برش می باشد. خروجی دیگر

¹Basis

²Half-space

³Simplex

⁴Mixed-Integer Gomory

که با نماد h_c نشان داده شده است در رابطه (۱۶.۴) آورده شده است.

$$h_c = \{c^T z > \lceil c^T z^{LP} \rceil\} \quad (16.4)$$

بنابراین از الگوریتم (۴.۴) برای حل کردن مسئله خطی ترکیب عدد صحیح به صورت کاملاً توزیع شده استفاده می‌شود. نحوه کار این الگوریتم بدین صورت است ابتدا هریک از گره‌های موجود در شبکه یک یا چند قید اختصاصی برای شروع دارد. در اولین مرحله هر گره مسئله خطی معادل را با توجه به قیدهای اولیه حل می‌کند و برش‌ها و مجموعه پایه متناسب را می‌سازد. در هر مرحله هر گره پایه تولیدی همسایه‌هایش را دریافت می‌کند و با اخرين پایه و برش‌های تولیدی خودش اشتراک می‌گیرد و در مرحله بعد از اين پایه برای حل مسئله خطی معادل استفاده می‌کند. برای کارکرد درست الگوریتم این شرط اساسی وجود دارد که تابع هدف مسئله بهینه‌سازی به صورت گسسته (عدد صحیح) باشد. برای حل مشکل شرط گسسته بودن تابع هدف یک الگوریتم دیگر ارائه می‌شود که در الگوریتم (۵.۴) آورده شده است. ابتدا مسئله اصلی نوشته شده در رابطه (۱۱.۴) به صورت رابطه (۱۷.۴) بازنویسی می‌شود، که مدل epigraph نامیده می‌شود.

$$\min_{\rho, z} \rho \quad (17.4)$$

$$\text{subj. to } a_i^T z \leq b_i, i = 1, \dots, n \quad (17.4)$$

$$c^T z \leq \rho \quad (17.4)$$

$$\rho \in \mathbb{R}, z \in \mathbb{Z}^{d_Z} \times \mathbb{R}^{d_R} \quad (17.4)$$

$$(e) \quad (17.4)$$

می‌دانیم که ρ لزوماً به صورت عدد صحیح نیست، بنابراین یک متغیر دیگر به نام ρ_I که فرض می‌کنیم به صورت عدد صحیح است، به صورت $\rho = \rho_I + \epsilon$ تعریف می‌شود که ϵ یک عدد مثبت کوچک است. حال مسئله بهینه‌سازی مجدداً به فرم رابطه (۱۸.۴) بازنویسی می‌شود و درنهایت الگوریتم توزیع شده برای این مسئله نوشته

می شود.

$$\min_{\rho_I, z} \rho_I \quad (18.4)$$

$$\text{subj. to} \quad a_i^T z \leq b_i, i = 1, \dots, n \quad (18.4)$$

$$c^T z \leq \epsilon \rho_I \quad (18.4)$$

$$\rho_I \in \mathbb{Z}, z \in \mathbb{Z}^{d_Z} \times \mathbb{R}^{d_R}, \epsilon > 0 \quad (18.4)$$

$$(5) \quad (18.4)$$

درنهایت می توان الگوریتم (5.۴) را به صورت زیر نوشت که برای تمام مسئله های خطی ترکیب عدد صحیح جواب می دهد. توضیح بیشتر در مورد جزئیات این الگوریتم این که هر یک از گره ها قید اولیه و محلی خود را به صورت $h_{i0} = \{a_i^T z \leq b_i\} \cap \{c^T z \leq \epsilon \rho\}$ در نظر می گیرد. زوج متغیر ρ_I ، z را نیز به عنوان هزینه جدید در نظر می گیریم و با e_1 نشان می دهیم.

Algorithm 4.5 Distributed Algorithm

State $((\rho_I^{[i]}, z^{[i]}), B^{[i]})$

Initializing

$$1: h^{[i]} = h_{i0} \cap H_M$$

$$2: ((\rho_I^{[i]}, z^{[i]}), B^{[i]}) = LPLEXSOLV(h^{[i]}, e_1)$$

Evolution

$$3: h_{MIG}(t) = MIGORACLE((\rho_I^{[i]}(t), z^{[i]}(t)), B^{[i]}(t), c)$$

$$4: h_c(t) = \{\rho_I \geq \lceil \rho_I^{[i]}(t) \rceil\}$$

$$5: H_{TMP}(t) = (\bigcap_{j \in N_i(t)} B^{[j]}(t)) \cap B^{[i]}(t) \cap h^{[i]} \cap h_{MIG}(t) \cap h_c(t)$$

$$6: ((\rho_I^{[i]}(t+1), z^{[i]}(t+1)), B^{[i]}(t+1)) = LPLEXSOLV(H_{TMP}, e_1)$$

در هر دو الگوریتم توزیع شده ارائه شده، هر گره متغیر اصلی مسئله را به صورت محلی ذخیره می کند و در هر مرحله بعد از تبادل اطلاعات با همسایه های خود و دریافت قید (پایه) های جدید، متغیر اصلی مسئله را بروز رسانی می کند. سوالی که پیش می آید، این است که هر گره چگونه بداند فرآیند حل مسئله به اتمام رسیده است یا نه، در [36] گفته شده است که هر وقت یک گره از همسایه های خود هیچ قید جدیدی دریافت نکرد، می توان گفت که فرآیند حل مسئله به پایان رسیده است.

در این قسمت لازم است که مسئله مربوط به این پایان نامه را به فرم رابطه (۱۱.۴) و سپس رابطه (۱۷.۴) بنویسیم. ابتدا لازم است که متغیر اصلی یعنی z را تعریف کنیم، که در رابطه (۱۹.۴) انجام شده است. در این رابطه ابتدا کلیه متغیرهای گستته (دودویی) مسئله یعنی x و سپس متغیرهای پیوسته یعنی β به صورت یک بردار در کنار هم قرار داده می شود. در رابطه (۱۹.۴) جهت نمایش درست از قراردادهای استاندارد نرم افزار مطلب^۱ استفاده شده است.

$$z = (\underline{x_e}^T, \underline{x_f}^T, \underline{x_c}^T, \underline{\beta_e}(:)^T, \underline{\beta_f}(:)^T, \underline{\beta_c}(:)^T)^T \quad (۱۹.۴)$$

$$\underline{x_e} = (x_{e,1}, \dots, x_{e,|T|})^T \quad (۱۹.۴\text{ ب})$$

$$[\underline{\beta_e}]_j^i = \beta_{i,j,e} \quad (۱۹.۴\text{ ج})$$

در قدم بعدی لازم است که قیدها را بین گره ها تقسیم کنیم، در مورد قیدهایی که تفکیک پذیر هستند، یعنی قیدهای رابطه (۶.۳)، رابطه (۷.۳)، رابطه (۱۲.۳)، رابطه (۱۳.۳)، رابطه (۱۹.۳) و رابطه (۲۱.۳) وضعیت مشخص است و هر گره قید مختص به خود را دارد، اما در مورد دو قید رابطه (۲۳.۳) و رابطه (۳۲.۳) که به صورت مشترک هستند لازم است تصمیم گرفته شود. تعداد قیدهای دسته اول $|T|$ است و تعداد قیدهای دسته دوم $|S| \times |T|$ است، در حالی که تعداد گره های پردازشی از رابطه (۳۶.۳)^۱ است. حال لازم است که این دو دسته قید بین گره ها توزیع شوند، برای توزیع بهتر می توان از این ایده کمک گرفت که توزیع قیدها را از گره هایی که توانایی پردازش بیشتری دارند شروع کرد. به صورت میانگین به هر گره $\frac{|T| \times (|S|+1)}{l}$ قید می رسد. در نهایت ρ به راحتی با مقایسه باتابع هدف در رابطه (۲۸.۳) قابل استخراج است.

۵.۴ بررسی همگرایی و پیچیدگی

در این بخش نوع همگرایی و میزان پیچیدگی دو الگوریتم گفته شده در این فصل مورد بررسی قرار می گیرد. در [۳۶] نویسنده اثبات کرده اند که الگوریتم (۵.۴) جواب زیربهینه^۲ ارائه می دهد به این معنی که مقدار زیربهینه به دست آمده حداقل به اندازه^۳ از مقدار بهینه اصلی بیشتر است. اگر^۴ z جواب زیربهینه به دست

¹Matlab

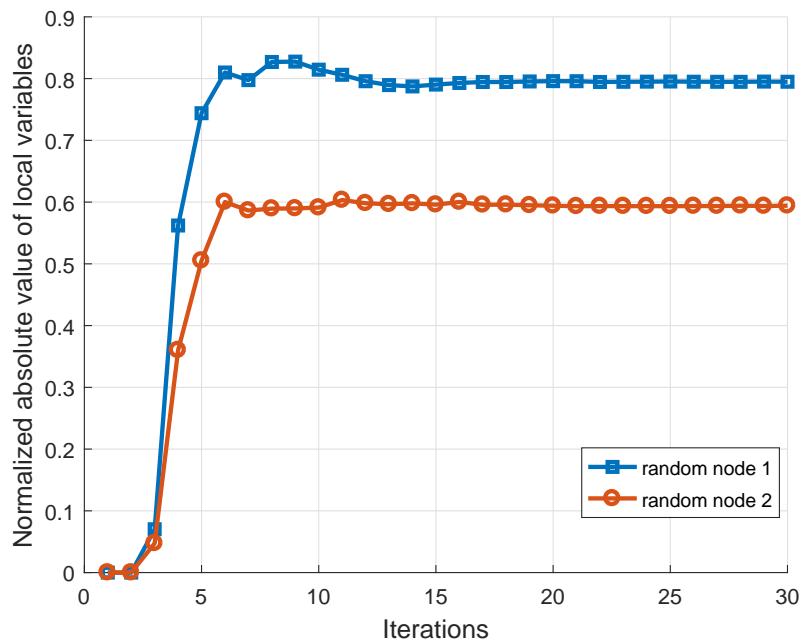
آمده از الگوریتم (۵.۴) باشد و z^* جواب بهینه مسئله باشد آنگاه $\epsilon < c^T z^* - c^T z^\epsilon$. در این منبع در مورد میزان پیچیدگی الگوریتم توزیع شده نیز اشاره شده است که این الگوریتم در زمان خطی نسبت به توپولوژی مسئله (مثلًا تعداد گرهها) به جواب می‌رسد.

در مورد راه حل VTP ارائه شده نیز می‌توان گفت با فرض اینکه تعداد کل وظیفه‌ها $|T|$ باشد و تعداد کل گره‌های محاسباتی l باشد، همچنین حداکثر تعداد قابل تقسیم برای هر وظیفه N_t باشد. آنگاه میزان پیچیدگی محاسباتی برای یافتن جواب به کمک روش جستجو اکتشافی از حدود $O(l \times (|T|N_t)^2)$ خواهد بود که نسبت به ابعاد مسئله چندجمله‌ای است.

۶.۴ نتایج شبیه‌سازی

در این قسمت نتایج شبیه‌سازی را بیان می‌کنیم. در همه موارد این فصل همانند فصل (۳) نتایج برای ۲۰۰ بار اجرا شده‌اند و نتایج به صورت میانگین بیان شده‌اند. همانند مدل سیستم یک شبکه چهار لایه به صورت شکل (۲.۳) در نظر می‌گیریم، تعداد گره‌های هر لایه هر بار به صورت تصادفی انتخاب می‌شوند، تعداد وظیفه‌ها و همچنین نرخ تولید وظیفه‌ها به صورت تصادفی در یک محدوده عددی معقول تولید می‌شوند. برای تاخیر مسیریاب‌ها فرض می‌کنیم که به صورت میانگین عبور بسته‌ها از مسیریاب‌های لایه ۱، ۲ میلی ثانیه، مسیریاب‌های لایه ۲، ۴ میلی ثانیه و مسیریاب‌های لایه ۳، ۱۰ میلی ثانیه طول می‌کشد. شکل (۲.۳) توپولوژی شبکه را که در شبیه‌سازی استفاده شده است نشان می‌دهد. ظرفیت مربوط به هریک از گره‌ها به نجوم انتخاب شده است که بیشترین ظرفیت مربوط به گره‌های ابری و کمترین مربوط به گره‌های لبه باشد. که به ترتیب به نسبت اعداد ۱۰، ۹ و ۸ درنظر گرفته شده است. حجم پردازنده مورد نیاز برای وظیفه‌ها نیز به صورت تصادفی در محدوده عدد ۴۰ واحد در نظر گرفته شده‌اند. حداقل زمان لازم جهت پردازش وظیفه‌ها به صورت تصادفی و با میانگین حدود ۱۵ میلی ثانیه فرض شده است. واحد قیمت پردازشی برای گره‌ها به صورتی در نظر گرفته شده است که ارزان‌ترین هزینه مربوط به گره‌های لایه ابری و گرانترین مربوط به گره‌های لایه لبه باشد، این اعداد به صورت تصادفی و به ترتیب به نسبت اعداد ۱، ۴ و ۹ واحد در نظر گرفته شده است. نرخ پواسون تولید وظیفه در گره‌های حسگری به صورت تصادفی و در حدود ۵۰ واحد بر ثانیه برای هر وظیفه در هر گره حسگری درنظر گرفته شده است.

ابتدا نحوه همگرایی در روش توزیع شده را بررسی می‌کنیم. در شکل (۳.۴) نحوه همگرایی برای دو



شکل ۳.۴: نحوه همگرا شدن متغیر محلی در دو گره تصادفی در روش توزیع شده

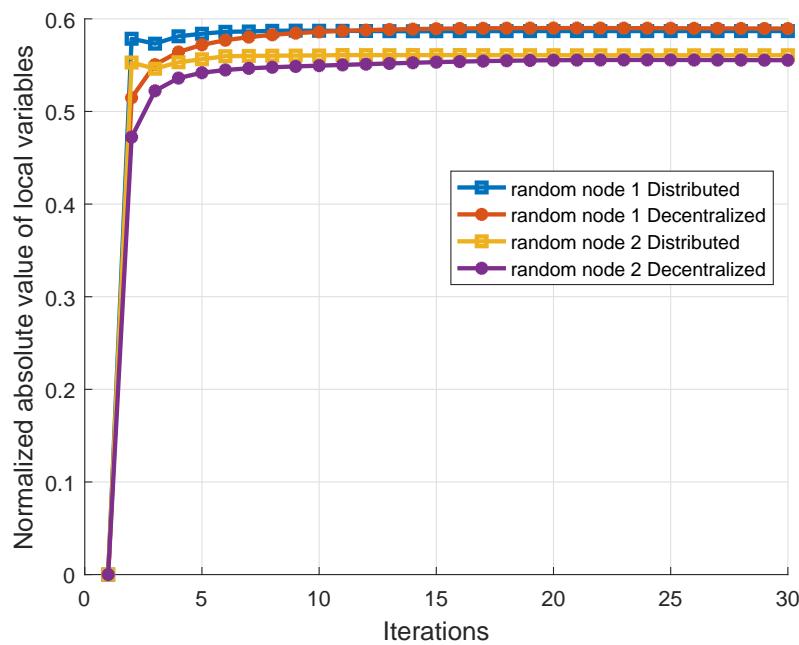
گره تصادفی از شبکه در روش توزیع شده به نمایش گذاشته شده است. در این نمودار اندازه متغیر محلی برای این دو گره در طول زمان نشان داده شده است.

در شکل (۴.۴) نحوه همگرایی متغیر محلی در دو روش غیرمت مرکز و توزیع شده مقایسه شده است. همانطور که دیده می شود در روش غیرمت مرکز سرعت همگرایی اندکی بیشتر است، که دلیل آن وجود واحد هدایتگر^۱ در روش غیرمت مرکز است که باعث می شود سرعت همگرایی بیشتر شود و همچنین نحوه همگرایی صاف تر باشد.

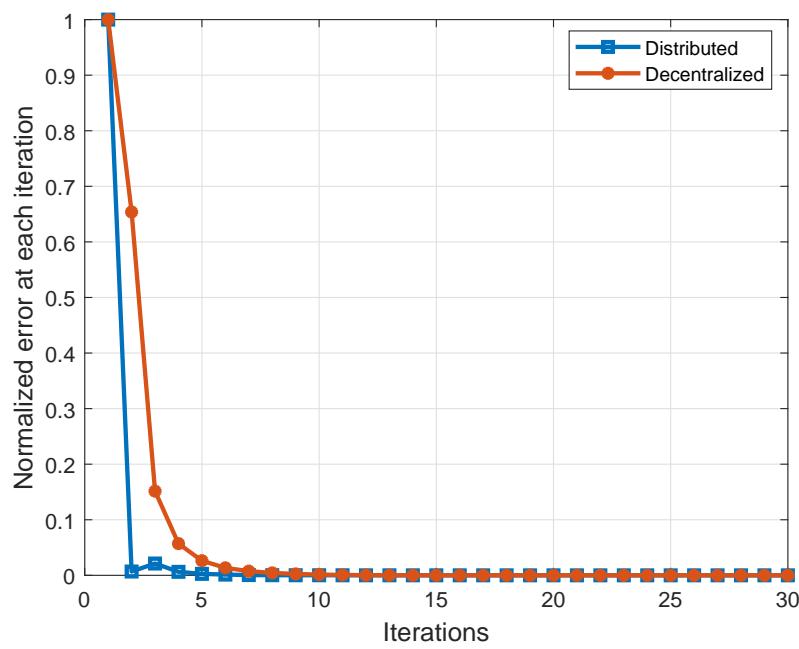
در شکل (۵.۴) نیز نحوه همگرایی میزان خطا در دو روش غیرمت مرکز و توزیع شده به نمایش گذاشته شده است. در این نمودار در هر مرحله اختلاف مقدار تابع هدف با مرحله قبل به صورت خطا در نظر گرفته شده است که این خطا به صورت یکه شده نمایش داده شده است.

در شبیه سازی بعدی که در شکل (۶.۴) دیده می شود هر چهار روش موجود بر روی شبکه اجرا شده است، همانطور که از این نمودار دیده می شود مقدار تابع هدف در دو راه حل مت مرکز و غیرمت مرکز بهینه است و در راه حل توزیع شده نیز با فاکتور دلخواه قابل کنترل است، اما در مورد راه حل اکتشافی می بینیم که این راه حل

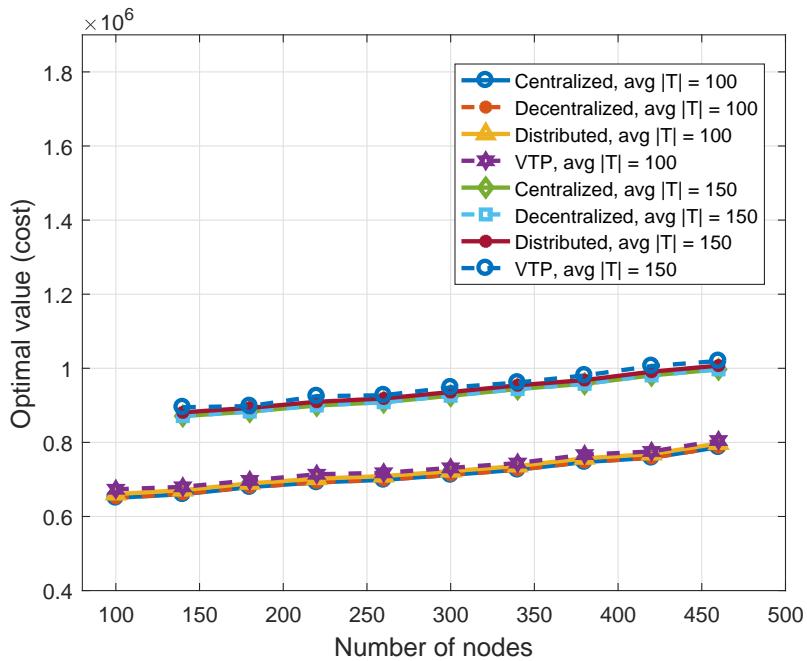
¹Coordinator



شکل ۴.۴: نحوه همگرا شدن اندازه‌ی متغیر محلی مسئله در دو روش توزیع شده و غیرمت مرکز برای دو گره تصادفی



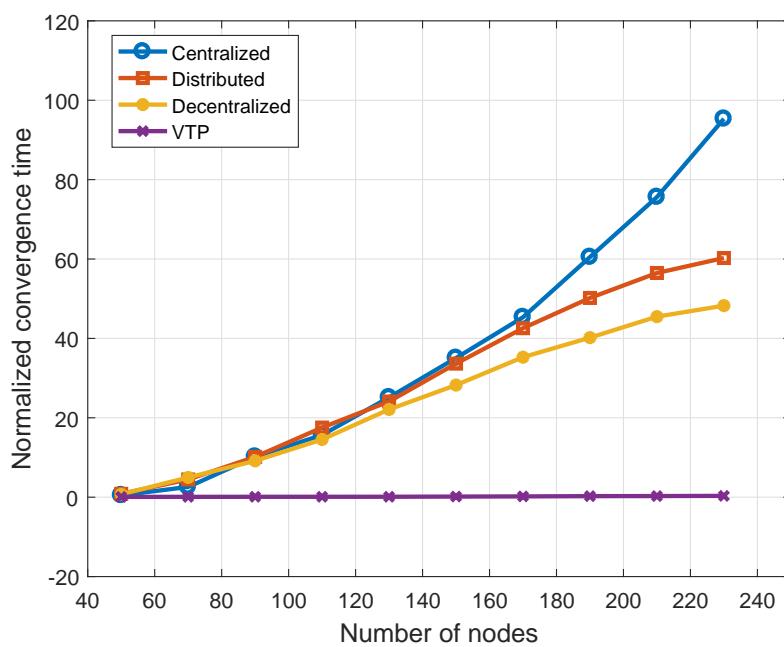
شکل ۵.۴: نحوه همگرا شدن مقدار خطأ در دو روش توزیع شده و غیرمت مرکز



شکل ۶.۴: مقدار تابع هدف (هزینه) برای چهار روش مختلف دربرابر تعداد کل گره های شبکه

جواب زیربهینه را می دهد.

آخرین تحلیل مربوط به زمان آماده شدن جواب در چهار راه حل ارائه شده در این پایان نامه است. سعی شده است که شبیه سازی ها در شرایط کاملاً یکسان انجام شود، از جمله میزان حافظه موجود بر روی دستگاه کامپیوتری که تست ها بر روی انجام شده است و یا دمای پردازنده در هنگام تست و همانطور که دیده می شود به وضوح زمان محاسبات در راه حل اکتشافی کم است. در مورد دو راه حل غیر مرکز و توزیع شده نیز ذکر این نکته لازم است که شرایط واقعی شبیه سازی زمانی است که محاسبات در گره های مختلف و به صورت همزمان انجام شود در حالیکه در تست های انجام شده به دلیل محدودیت منابع کلیه شبیه سازی ها بر روی یک دستگاه انجام شده است که این خود باعث می شود خاصیت اصلی این دو روش به خوبی دیده نشود، با این وجود در شکل (۷.۴) دیده می شود که زمان نسبی آماده شدن جواب در سه روش غیر مرکز، توزیع شده و اکتشافی به صورت خطی نسبت به اندازه توپولوژی مسئله است.



شکل ۷.۴: زمان نسبی آماده شدن جواب نهایی در چهار روش مختلف در برابر تعداد گره های شبکه

۷.۴ جمع بندی و نتیجه گیری

در این فصل دو روش دیگر برای حل مسئله اصلی ارائه شد. روش اول یک راه حل اکتشافی مبتنی بر الگوریتم ویتربی بود که مسئله اصلی را بدون نیاز به خطی سازی و به صورت اکتشافی حل می کند. روش دوم یک روش کاملا توزیع شده است که مسئله بین تمام گره های پردازشی تقسیم می شود و از قدرت پردازشی آنها برای حل مسئله کمک گرفته می شود. هردو روش ارائه شده در این فصل جواب زیربهینه ارائه می کردند. در قسمت نتایج دیده شد که هر دوروش در زمان خطی به نتیجه می رسند و چهار روش ارائه شده در این پایان نامه از جنبه های مختلف با یکدیگر مقایسه شد.

فصل ۵

نتیجه‌گیری و کارهای آینده

در این فصل ابتدا مروجی داریم بر کارهای انجام شده در این پایان نامه و سپس برخی از جهت‌های تحقیقاتی ممکن برای ادامه‌ی این فعالیت پژوهشی بیان خواهند شد.

۱.۵ خلاصه و جمع‌بندی

در این پایان‌نامه، ابتدا به معرفی اینترنت اشیاء و کاربردهای آن در شهر هوشمند پرداختیم. پس از آن برخی خدمات شهر هوشمند را برشمردیم. سپس به معرفی پردازش لبه و مه و دلایل برتری آن‌ها نسبت به پردازش ابری پرداختیم. در ادامه کارهای انجام شده در اختصاص منابع در شبکه اینترنت اشیاء را بررسی کردیم.

در فصل (۳) صورت مسئله تخصیص منابع جهت پردازش وظیفه‌ها به صورت یک مسئله بهینه‌سازی مدل‌سازی شد. در این نوع از تخصیص منابع، منابع پردازشی به تعدادی وظیفه اختصاص پیدا می‌کنند. در این فصل مسئله تخصیص منابع به صورت یک مسئله بهینه سازی فرمول بندی شد که هدف آن کمینه کردن مجموع هزینه پردازشی در کل گره‌های شبکه است. با توجه به حل دشوار مسئله مورد نظر در ابتدا این مسئله خطی‌سازی شد. جواب بهینه مسئله به روش مرکز به دست آمد. در ادامه یک روش غیر مرکز برای حل مسئله ارائه شد و در نهایت میزان پیچیدگی و نحوه همگرایی در این روش بررسی شد. در آخر نتایج مقایسه‌ای برای این دو روش ارائه شد.

در فصل (۴) ابتدا صورت مسئله غیرخطی اولیه بازنویسی شد. به کمک الگوریتم ویتری بیک راه حل

اکتشافی به نام VTP برای حل مسئله اولیه به صورت زیربهینه ارائه شد. در ادامه یک روش توزیع شده نیز برای حل مسئله خطی شده ارائه شد. این روش نیز جواب زیربهینه ارائه می‌کرد. پس از بررسی همگرایی و پیچیدگی دو الگوریتم ارائه شده، شبیه‌سازی‌هایی برای بررسی آن‌ها ارائه شد. درنهایت چهار روش موجود در این پایان نامه از نظر زمان رسیدن به جواب مقایسه شد که مشخص شد روش VTP سریع‌ترین روش است.

۲.۵ کارهای آینده

در این قسمت چند پیشنهاد برای ادامه این کار مطرح می‌کنیم. اولین پیشنهاد که به صورت یک پیشنهاد عملی است این است که دو روش غیرمتمرکز و توزیع شده به عنوان قراردادهای هوشمندی در شبکه زنجیره‌بلوک^۱ عملیاتی شوند. برای این‌کار لازم است که کلیه گره‌های موجود در شبکه به عنوان یک گره در شبکه زنجیره‌بلوک در نظر گرفته شوند. برای ایجاد امنیت بیشتر در این شبکه می‌توان از زنجیره‌های بلوک خصوصی استفاده کرد.

به عنوان پیشنهاد دوم، با ترکیب کردن این مدل‌ها با شبکه‌های مبتنی بر نرم‌افزار^۲ می‌توان به مدل بهتری رسید. همچنین ترکیب این کار با تخصیص منابع رادیویی برای انتقال داده‌های حسگرها و فعال کننده‌ها هم می‌تواند به واقعی‌تر شدن مدل ارائه شده کمک کند.

به عنوان پیشنهاد آخر، در مدل ارائه شده در فصل (۳) می‌توان مسئله را به صورت یک بازی^۳ تعریف کرد و از روش‌های حل مسئله برای این دسته از مسائل استفاده کرد.

¹Blockchain

²Software Defined Networks (SDN)

³Game

مراجع

- [1] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, “Edge computing: Vision and challenges,” *IEEE Internet of Things Journal*, vol.3, no.5, pp.637–646, 2016.
- [2] S. Sarkar and S. Misra, “Theoretical modelling of fog computing: a green computing paradigm to support iot applications,” *Iet Networks*, vol.5, no.2, pp.23–29, 2016.
- [3] R. Deng, R. Lu, C. Lai, T. H. Luan, and H. Liang, “Optimal workload allocation in fog-cloud computing toward balanced delay and power consumption,” *IEEE Internet of Things Journal*, vol.3, no.6, pp.1171–1181, 2016.
- [4] M. Barcelo, A. Correa, J. Llorca, A. M. Tulino, J. L. Vicario, and A. Morell, “Iot-cloud service optimization in next generation smart environments,” *IEEE Journal on Selected Areas in Communications*, vol.34, no.12, pp.4077–4090, 2016.
- [5] H. Zhang, Y. Xiao, S. Bu, D. Niyato, F. R. Yu, and Z. Han, “Computing resource allocation in three-tier iot fog networks: A joint optimization approach combining stackelberg game and matching,” *IEEE Internet of Things Journal*, vol.4, no.5, pp.1204–1215, 2017.
- [6] L. Liu, Z. Chang, X. Guo, S. Mao, and T. Ristaniemi, “Multiobjective optimization for computation offloading in fog computing,” *IEEE Internet of Things Journal*, vol.5, no.1, pp.283–294, 2017.
- [7] Y. Gu, Z. Chang, M. Pan, L. Song, and Z. Han, “Joint radio and computational resource allocation in iot fog computing,” *IEEE Transactions on Vehicular Technology*, vol.67, no.8, pp.7475–7484, 2018.
- [8] C. Zhu, J. Tao, G. Pastor, Y. Xiao, Y. Ji, Q. Zhou, Y. Li, and A. Ylä-Jääski, “Folo: Latency and quality optimized task allocation in vehicular fog computing,” *IEEE Internet of Things Journal*, 2018.

- [9] F. Zhou, Y. Wu, R. Q. Hu, and Y. Qian, “Computation rate maximization in uav-enabled wireless-powered mobile-edge computing systems,” *IEEE Journal on Selected Areas in Communications*, vol.36, no.9, pp.1927–1941, 2018.
- [10] K. Ashton, “That internet of things thing,” *RFID j.*, vol.22, no.7, pp.97–114, 2009.
- [11] J. Zheng, D. Simplot-Ryl, C. Bisdikian, and H. T. Mouftah, “The internet of things [guest editorial],” *IEEE Communications Magazine*, vol.49, no.11, pp.30–31, 2011.
- [12] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, “Internet of things: A survey on enabling technologies, protocols, and applications,” *IEEE communications surveys & tutorials*, vol.17, no.4, pp.2347–2376, 2015.
- [13] P. Bellavista, G. Cardone, A. Corradi, and L. Foschini, “Convergence of manet and wsn in iot urban scenarios,” *IEEE Sensors Journal*, vol.13, no.10, pp.3558–3567, 2013.
- [14] A. Zanella, N. Bui, A. Castellani, L. Vangelista, and M. Zorzi, “Internet of things for smart cities,” *IEEE Internet of Things journal*, vol.1, no.1, pp.22–32, 2014.
- [15] A. Laya, V.-I. Bratu, and J. Markendahl, “Who is investing in machine-to-machine communications?,” 2013.
- [16] H. Schaffers, N. Komninos, M. Pallot, B. Trousse, M. Nilsson, and A. Oliveira, “Smart cities and the future internet: Towards cooperation frameworks for open innovation,” in *The future internet assembly*, pp.431–446, Springer, 2011.
- [17] D. Cuff, M. Hansen, and J. Kang, “Urban sensing: out of the woods,” *Communications of the ACM*, vol.51, no.3, p.24, 2008.
- [18] J. P. Lynch and K. J. Loh, “A summary review of wireless sensors and sensor networks for structural health monitoring,” *Shock and Vibration Digest*, vol.38, no.2, pp.91–130, 2006.
- [19] T. Nuortio, J. Kytöjoki, H. Niska, and O. Bräysy, “Improved route planning and scheduling of waste collection and transport,” *Expert systems with applications*, vol.30, no.2, pp.223–232, 2006.
- [20] A. Al-Ali, I. Zualkernan, and F. Aloul, “A mobile gprs-sensors array for air pollution monitoring,” *IEEE Sensors Journal*, vol.10, no.10, pp.1666–1671, 2010.

- [21] N. Maisonneuve, M. Stevens, M. E. Niessen, P. Hanappe, and L. Steels, “Citizen noise pollution monitoring,” in *Proceedings of the 10th Annual International Conference on Digital Government Research: Social Networks: Making Connections between Citizens, Data and Government*, pp.96–103, Digital Government Society of North America, 2009.
- [22] X. Li, W. Shu, M. Li, H.-Y. Huang, P.-E. Luo, and M.-Y. Wu, “Performance evaluation of vehicle-based mobile sensor networks for traffic monitoring,” *IEEE transactions on vehicular technology*, vol.58, no.4, pp.1647–1653, 2008.
- [23] S. Lee, D. Yoon, and A. Ghosh, “Intelligent parking lot application using wireless sensor networks.,” in *CTS*, pp.48–57, 2008.
- [24] “Cisco edge-to-enterprise iot analytics for electric utilities,” 2018.
- [25] “Data never sleeps 7.0,”
- [26] S. Yi, Z. Hao, Z. Qin, and Q. Li, “Fog computing: Platform and applications,” in *2015 Third IEEE Workshop on Hot Topics in Web Systems and Technologies (HotWeb)*, pp.73–78, IEEE, 2015.
- [27] K. Ha, Z. Chen, W. Hu, W. Richter, P. Pillai, and M. Satyanarayanan, “Towards wearable cognitive assistance,” in *Proceedings of the 12th annual international conference on Mobile systems, applications, and services*, pp.68–81, ACM, 2014.
- [28] B.-G. Chun, S. Ihm, P. Maniatis, M. Naik, and A. Patti, “Clonecloud: elastic execution between mobile device and cloud,” in *Proceedings of the sixth conference on Computer systems*, pp.301–314, ACM, 2011.
- [29] C. G. C. Index, “Forecast and methodology, 2014–2019, 2015,” *Forrás: www.cisco.com/c/en/us/solutions/collateral/service-provider/global-cloud-index-gci/Cloud_Index_White_Paper.pdf* (2015. 10. 01.).
- [30] 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,” *IEEE Transactions on Communications*, vol.66, no.4, pp.1594–1608, 2018.
- [31] J. Wang, L. Zhao, J. Liu, and N. Kato, “Smart resource allocation for mobile edge computing: A deep reinforcement learning approach,” *IEEE Transactions on Emerging Topics in Computing*, 2019.

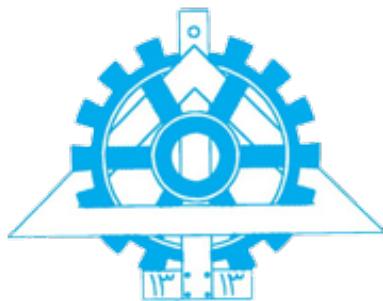
- [32] A. Falsone, K. Margellos, and M. Prandini, “A decentralized approach to multi-agent milps: Finite-time feasibility and performance guarantees,” *Automatica*, vol.103, 06 2017.
- [33] N. Z. Shor. *Minimization methods for non-differentiable functions*, vol.3. Springer Science & Business Media, 2012.
- [34] R. Vujanic, P. M. Esfahani, P. J. Goulart, S. Mariéthoz, and M. Morari, “A decomposition method for large scale milps, with performance guarantees and a power system application,” *Automatica*, vol.67, pp.144–156, 2016.
- [35] G. D. Forney, “The viterbi algorithm,” *Proceedings of the IEEE*, vol.61, no.3, pp.268–278, 1973.
- [36] A. Testa, A. Rucco, and G. Notarstefano, “Distributed mixed-integer linear programming via cut generation and constraint exchange,” *IEEE Transactions on Automatic Control*, vol.65, no.4, pp.1456–1467, 2019.
- [37] C. N. Jones, E. C. Kerrigan, and J. M. Maciejowski, “Lexicographic perturbation for multiparametric linear programming with applications to control,” *Automatica*, vol.43, no.10, pp.1808–1816, 2007.
- [38] R. Gomory, “An algorithm for the mixed integer problem,” tech. rep., RAND CORP SANTA MONICA CA, 1960.

Abstract:

Smart city is one of the most important applications of IoT¹ which attracts lots of attention from industry and academy. The main purpose of smart city is to provide the possibility of Internet connection for all of the devices in the network. The generated data of these devices should be processed by the network. Therefore, all of the generated data is transmitted to the processing nodes and the result of the processed data is informed to the devices in form of some actions. One of the important challenges of implementing smart city is the determination of the processing nodes for each generated data. The naive solution for this challenge is processing all of the data in a centralized node which is named cloud. However, this solution is not appropriate in the case of the next-generation network in which there are massive amounts of devices with different processing requirements. To this end, edge computing and fog computing are two of the proposed novel patterns. The most characteristic of these two patterns is the decreasing of the response time to the generated data. However, resource allocation has become challenging when there are different layers including, edge, fog, and cloud for processing the data. Therefore, we have introduced a novel model for resource allocation in smart for the scenario in which there are possible options including, edge, fog, and cloud nodes for processing the data. This problem is modeled as a mixed-integer non-linear programming (MINLP), to minimize the processing cost. According to the NP-Hard nature of the initial optimization problem, the main problem is linearized, and solved by centralized, decentralized, and distributed methods. Due to the necessity of linearization for these methods, a heuristic solution using the idea of the Viterbi algorithm is presented for solving the initial non-linear optimization problem which is named Viterbi-based task placement (VTP). According to the conducted simulations, we indicate that the centralized and decentralized methods converge to the optimal solution. The distributed method with few information exchanges between the devices converges to a sub-optimal solution with the desired error. It should be mentioned that the distributed method can be used in the asynchronous networks. The decentralized and distributed methods unlike the centralized method converge in polynomial time, and therefore, can be used in the network with a massive number of devices. Finally, it is worth noting that the introduced VTP algorithm converges in polynomial time and much faster than the centralized, decentralized, and distributed methods.

Keywords: Internet of things (IoT), Smart city, Edge computing, Fog computing, Cloud computing, Distributed optimization, Resource allocation

¹Internet of things



University of Tehran
School of Electrical and Computer Engineering

Distributed Intelligence in IoT Networks

By:

Mohammad Mahmoodian

Supervisor:

Dr. Vahid Shah-Mansouri

A Thesis Submitted in Partial Fulfillment of the Requirement for the Degree of
Master of Science in Electrical Engineering

October 2020