زمان‌بندی وظایف برنامه‌های کاربردی اینترنت اشیا در محیط رایانش مه با استفاده از یادگیری تقویتی عمیق

پگاه گازری، دادمهر رهبری و محسن نیک‌رأی

چکیده: همزمان با فراگیرشدن تکنولوژی اینترنت اشیا در سال‌های اخیر، تعداد دستگاه‌های هوشمند و به تبع آن حجم داده‌های جمع‌آوری‌شده توسط آنها به سرعت در حال افزایش است. از سوی دیگر، اغلب برنامه‌های کاربردی اینترنت اشیا نیازمند تحلیل بلادرنگ داده‌ها و تأخیر اندک در ارائه خدمات هستند. تحت چنین شرایطی، ارسال داده‌ها به مراکز داده ابری جهت پردازش، پاسخ‌گوی نیازمندی‌های برنامه‌های کاربردی مذکور نیست و مدل رایانش مه، انتخاب مناسب‌تری محسوب می‌گردد. با توجه به آن که منابع پردازشی موجود در مدل رایانش مه دارای محدودیت هستند، استفاده مؤثر از آنها دارای اهمیت ویژه‌ای است.

در این پژوهش به مسئله زمان‌بندی وظایف برنامه‌های کاربردی اینترنت اشیا در محیط رایانش مه پرداخته شده است. هدف اصلی در این مسئله، کاهش تأخیر ارائه خدمات است که جهت دستیابی به آن، از رویکرد یادگیری تقویتی عمیق استفاده شده است. روش ارائه‌شده در این مقاله، تلفیقی از الگوریتم Q-Learning، یادگیری عمیق و تکنیک‌های بازپخش تجربه و شبکه هدف است. نتایج شبیه‌سازی‌ها نشان می‌دهد که الگوریتم DQLTS از لحاظ معیار ASD، ۷۶٪ بهتر از الگوریتم QLTS و ۶/۵٪ بهتر از الگوریتم RS عمل می‌نماید و نسبت به QLTS زمان همگرایی سریع‌تری دارد.

کلیدواژه: **اینترنت اشیا، رایانش مه، زمان‌بندی وظایف، یادگیری تقویتی عمیق.**

۱- مقدمه

طبق اظهار سیسکو تا پایان سال ۲۰۲۰ میلادی حدود ۵۰ میلیارد شیء امکان اتصال به اینترنت را خواهند داشت [۱]. این اشیای هوشمند که در هر مکانی مستقر هستند [۲]، به تولید و جمع‌آوری حجم عظیمی از داده‌های خام می‌پردازند. چنانچه این داده‌ها، جهت پردازش و یا ذخیره‌سازی به مراکز داده ابری منتقل شوند، قطعاً موجب اشغال پهنای باند زیادی خواهند شد و علاوه بر آن زمان پاسخ برنامه‌های کاربردی اینترنت اشیا (IoT)[1] نیز افزایش خواهد یافت. از سوی دیگر، در برخی از صنایع، انتقال داده‌ها به خارج از مرزهای سازمان، نقض امنیت محسوب می‌گردد. تحت چنین شرایطی، انتقال داده‌ها به مراکز داده ابری چندان عملی و مقرون به صرفه نیست و می‌بایست از مدل رایانشی دیگری به این منظور استفاده گردد [۱]. در سال ۲۰۱۲ میلادی، شرکت سیسکو با هدف ارائه راه حلی برای چالش‌های مربوط به برنامه‌های کاربردی اینترنت اشیا در بستر رایانش ابری[2]، مدل جدیدی را تحت عنوان رایانش مه[3] معرفی نمود [۳]. این مدل، استقرار خدمات و برنامه‌های کاربردی اینترنت اشیا را تسهیل می‌نماید و طبق اظهار کنسرسیوم OpenFog، موجب کاهش تأخیر زمانی و به تبع آن امکان‌پذیرشدن پردازش و تحلیل بلادرنگ، کاهش اتلاف پهنای باند شبکه‌های ارتباطی و کاهش هزینه‌ها می‌گردد [۴].

یکی از نیازمندی‌های ضروری کلیه سیستم‌های رایانشی که منجر به بهبود عملکرد شبکه‌های رایانه‌ای نیز می‌گردد، مدیریت منابع و زمان‌بندی وظایف[4] به شیوه‌ای مؤثر است [۵] و [۶] و عدم استفاده از یک زمان‌بند مناسب می‌تواند منجر به ناکارامدی سخت‌افزار گردد [۶]. هدف اصلی در این پژوهش، کاهش میانگین تأخیر ارائه خدمات مربوط به برنامه‌های کاربردی اینترنت اشیا در بستر رایانش مه است. در این راستا قصد داریم که با بهره‌گیری از رویکرد یادگیری تقویتی عمیق (DRL)[5]، روش جدیدی را جهت زمان‌بندی وظایف مربوط به برنامه‌های کاربردی ایجاد نماییم. این روش قادر خواهد بود به گونه‌ای خودکار و با تکیه بر تجربیات کسب‌شده پیشین خود، به مرور زمان به یک استراتژی مؤثر جهت زمان‌بندی دست یابد. مسئله زمان‌بندی در حوزه مسایل تصمیم‌گیری قرار دارد و به دلیل پویایی شرایط حاکم بر شبکه‌های رایانه‌ای امروزی و دشواری مدل‌سازی آنها، حل چنین مسایلی نیازمند روشی برخط[6] و تطبیق‌پذیر است. لذا استفاده از روش‌های اکتشافی[7] و مبتنی بر قانون به این منظور، چندان عملی نیست.

چندی است که رویکرد یادگیری ماشین (ML)[8] جهت حل برخی از مسایل شبکه‌های رایانه‌ای (همچون مدیریت منابع، مسیریابی و کنترل ازدحام) مورد استفاده قرار می‌گیرد. با بهره‌گیری از یادگیری ماشین، می‌توان با استفاده مستقیم از داده‌ها و بدون به کارگیری قوانین از پیش تعیین شده، مدل‌هایی تخمینی و با دقتی قابل قبول را از سیستم‌هایی که رفتاری پیچیده دارند، ایجاد نمود [۷].

در این پژوهش از تلفیق الگوریتم Q-Learning، یادگیری عمیق (DL)[9] و تکنیک‌های بازپخش تجربه[10] و شبکه هدف[11] استفاده شده است. طبق [۸]، به کارگیری یادگیری عمیق جهت تطابق با تغییر در

این مقاله در تاریخ ۱۲ اردیبهشت ماه ۱۳۹۸ دریافت و در تاریخ ۲۲ بهمن ماه ۱۳۹۸ بازنگری شد.

پگاه گازری، دانشکده مهندسی کامپیوتر و فناوری اطلاعات، دانشگاه قم، قم، (email: p.gazori@stu.qom.ac.ir).

دادمهر رهبری (نویسنده مسئول)، دانشکده مهندسی کامپیوتر و فناوری اطلاعات، دانشگاه قم، قم، (email: d.rahbari@stu.qom.ac.ir).

محسن نیک‌رأی، دانشکده مهندسی کامپیوتر و فناوری اطلاعات، دانشگاه قم، قم، (email: m.nickray@qom.ac.ir).

1. Internet of Things
2. Cloud Computing
3. Fog Computing
4. Task Scheduling
5. Deep Reinforcement Learning
6. Online
7. Huristic
8. Machine Learning
9. Deep Learning
10. Experience Replay
11. Target Network

شرایط شبکه و همچنین مؤثرساختن انتخاب عمل در مسایلی با فضای حالت و عمل گسترده و استفاده از تکنیک‌های بازپخش تجربه در [۹] تا [۱۲] و شبکه هدف در [۱۱] و [۱۳] به ترتیب منجر به بهبود یادگیری عامل و کاهش نوسان و واگرایی در عملکرد عامل‌ها می‌گردد [۱۴]. در پژوهش‌هایی همچون [۵] و [۱۵] تا [۱۷]، به دلیل عدم امکان گسترش و یا تغییر در شرایط شبکه و کوچک‌بودن مقیاس فضاهای حالت و عمل، از یادگیری عمیق و تکنیک‌های مذکور استفاده نشده است.

یادگیری تقویتی، یکی از شاخه‌های یادگیری ماشین است که در آن موجودیتی تحت عنوان عامل[۱] از طریق تعامل پیوسته با محیط[۲] پیرامون خود یاد می‌گیرد که چگونه بهترین اعمال ممکن را با هدف بیشینه‌ساختن مقدار پاداش تجمعی[۳] انتخاب نماید. یادگیری عمیق نیز یکی از زیرحوزه‌های یادگیری ماشین است که به واسطه توانایی بازنمایی روابط ذاتی مابین ورودی و خروجی‌های سیستم، انتخاب مناسبی جهت یادگیری به صورت برخط است [۷]. لذا می‌توان نتیجه‌گیری نمود که یادگیری تقویتی عمیق انتخاب مناسبی جهت تصمیم‌گیری به صورت خودکار و برخط در مسئله زمان‌بندی وظایف برنامه‌های کاربردی اینترنت اشیا در بستر رایانش مه محسوب می‌گردد.

در این مسئله، دو چالش نیز وجود دارد: اولین چالش مربوط به توزیع مؤثر وظایف در میان تعداد محدودی ماشین مجازی و دومین چالش نیز محدودیت منابع سخت‌افزاری دروازه‌ها[۴] است. در بخش‌های آتی، در خصوص راهکار ارائه‌شده برای این چالش‌ها توضیح داده خواهد شد. به علاوه، نوآوری‌های این مقاله نیز عبارتند از:

- ارائه‌ی چارچوبی جهت زمان‌بندی وظایف با استفاده از تلفیق الگوریتم Q-Learning، یادگیری عمیق و تکنیک‌های بازپخش تجربه و شبکه هدف.
- استفاده از چهار زمان‌بند مجزا جهت ارائه خدمات به گروه‌هایی متمایز از دستگاه‌های پایانی و با هدف کاهش ابعاد فضاهای حالت[۵] و عمل[۶]، توزیع مؤثر وظایف ورودی و در نهایت، جهت استفاده مؤثر از منابع گره‌های پردازشی.
- نتایج آزمایش‌های شبیه‌سازی نشان می‌دهد که روش پیشنهادی این مقاله از لحاظ میانگین زمان انتظار، پاسخ، اتمام[۷] و تکمیل[۸] وظایف بهتر از سه الگوریتم مبنای مورد بررسی عمل کرده و با توزیع مناسب وظایف، به گونه‌ای مؤثر از منابع استفاده می‌نماید.

سازماندهی این مقاله بدین صورت است که در بخش ۲، پژوهش‌های مرتبط مورد بررسی قرار می‌گیرند. بخش ۳ به تشریح مدل سیستم و بخش ۴ به بیان جزئیات روش پیشنهادی اختصاص یافته‌اند. در بخش ۵ به ارزیابی عملکرد روش ارائه‌شده خود می‌پردازیم و در بخش ۶ جمع‌بندی، نتیجه‌گیری نهایی و کارهای آتی را خواهیم داشت.

۲- پژوهش‌های مرتبط

در این بخش به برخی از پژوهش‌های صورت‌گرفته در حوزه مدیریت منابع و زمان‌بندی وظایف یا کارها با بهره‌گیری از رویکردهای یادگیری تقویتی و یادگیری تقویتی عمیق می‌پردازیم.

۲-۱ حوزه یادگیری تقویتی

در [۵] و [۱۵] به زمان‌بندی کارها به ترتیب در محیط‌های رایانش توری[۹] و رایانش ابری و با اهداف کمینه‌سازی میانگین زمان تکمیل وظایف و زمان انتظار پرداخته شده است. در این دو پژوهش، به عنوان الگوریتم مبنا از Q-Learning استفاده شده است. نتایج ارزیابی‌ها نشان می‌دهد که روش‌های ارائه‌شده در [۵] و [۱۵] به ترتیب تحت شرایطی با مهلت زمانی برای کارها و پویایی بارهای کاری، با ایجاد تعادل میان بارها، به نتایج مطلوبی در مقایسه با چند الگوریتم مبنا دست یافته‌اند.

زمان اجرای وظایف و زمان پاسخ به درخواست‌های ورودی، دو نمونه از مهم‌ترین اهداف در مسایل زمان‌بندی و مدیریت منابع هستند که در [۱۶] و [۱۷] به ترتیب به این دو پرداخته شده است. در [۱۶]، سیستم‌های توزیع‌شده و در [۱۷] رایانش ابری مورد بررسی قرار گرفته‌اند و در هر دو پژوهش مذکور، الگوریتم Q-Learning به عنوان مبنا به کار گرفته شده است. ارزیابی عملکرد روش‌های ارائه‌شده نشان می‌دهد که در [۱۶]، با انتخاب قدرتمندترین منبع در دسترس و در [۱۷] با توزیع بهینه بارهای کاری به اهداف معین‌شده دست یافته‌اند.

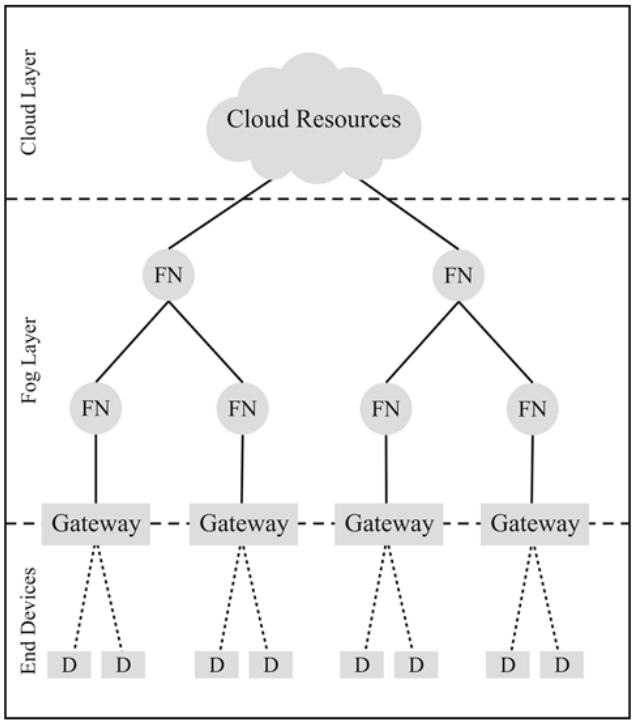
در [۱۸] به مسئله تخلیه محاسبات[۱۰] و تخصیص منابع[۱۱] دستگاه‌های سیار به وظایف در بستر رایانش لبه[۱۲] پرداخته شده است. چارچوب برخط ارائه‌شده در این پژوهش با بهره‌گیری از استراتژی بهینه‌سازی Lyapunov و الگوریتم Q-Learning ایجاد شده است. ارزیابی عملکرد روش پیشنهادی، توانایی آن را در کاهش هزینه‌های عملیاتی و همچنین زمان اجرای وظایف نشان می‌دهد.

۲-۲ حوزه یادگیری تقویتی عمیق

طبق [۷]، روش DeepRM [۸]، اولین پژوهشی است که در آن از رویکرد یادگیری تقویتی عمیق جهت زمان‌بندی کارها در خوشه‌های چندمنبعی استفاده شده است. هدف این پژوهش، کمینه‌سازی یک معیار وابسته به زمان با عنوان Slowdown کارها است که طبق آزمایش‌ها، الگوریتم DeepRM با بهره‌گیری از یک شبکه تماماً متصل[۱۳] و در مقایسه با چند الگوریتم مبنا، عملکرد مطلوبی را به نمایش گذاشته است.

یکی دیگر از اهداف مطرح در مسایل زمان‌بندی و مدیریت منابع، کاهش مصرف انرژی یا توان مصرفی است که این دو مورد به ترتیب در [۹] و [۱۰] مورد توجه قرار گرفته‌اند. محیط مورد مطالعه در [۹]، سیستم‌های بلادرنگ و در [۱۰] رایانش ابری است و هر دو پژوهش، روش پیشنهادی خود را بر مبنای رویکرد DRL و با تلفیق یک SAE[۱۴] و الگوریتم Q-Learning ایجاد نموده‌اند. به علاوه در [۱۰] به منظور مدیریت توان مصرفی از یک شبکه LSTM استفاده شده است. آزمایش‌های ارزیابی عملکرد، کارایی روش‌های ارائه‌شده در این دو پژوهش و همچنین توانایی روش ارائه‌شده در [۱۰] را جهت ایجاد توازن میان تأخیر زمانی و مصرف انرژی نشان می‌دهد.

1. Agent
2. Environment
3. Cumulative Reward
4. Gateway
5. State
6. Action
7. Finish Time
8. Makespan Time
9. Grid Computing
10. Computation Offloading
11. Resource Allocation
12. Edge Computing
13. Fully Connected
14. Stacked Auto Encoder



شکل ۱: هم‌بندی شبکه پیشنهادی.

در [۱۹] و [۱۱]، هدف اصلی پژوهش، کاهش تأخیر در ارائه خدمات به ترتیب در محیط اینترنت وسایل نقلیه (IoV)[۱] و اینترنت اشیای مبتنی بر رایانش مه است که در این راستا، در هر دو پژوهش از یادگیری تقویتی عمیق مبتنی بر الگوریتم A۳C[۲] و علاوه بر آن در [۱۱]، از روش گرادیان سیاست طبیعی[۳] نیز استفاده شده است. در نهایت، افزایش وابستگی وظایف در [۱۹]، به بهبود عملکرد روش ارائه‌شده در مقایسه با یک الگوریتم حریصانه منجر گشته است. همچنین چارچوب ارائه‌شده در [۱۱] نیز عملکرد بهتری را از لحاظ تأخیر در ارائه خدمات نشان می‌دهد.

در [۱۳]، دو هدف مصرف انرژی و کاهش هزینه‌های ناشی از تأخیر در ارائه خدمات در قالب تخلیه هم‌زمان ترافیک و محاسبات در شبکه‌های وسایل نقلیه[۴] در بستر رایانش مه، به طور هم‌راستا دنبال می‌شوند. در این پژوهش با بهره‌گیری از الگوریتم Q-Learning عمیق دوگانه[۵] (DDQL) به عملکردی مطلوب‌تر نسبت به چند الگوریتم مبنا دست یافته‌اند.

در [۱۲] و [۲۰] به ترتیب به مسایل تخصیص منابع به محاسبات و مدیریت منابع و محاسبات در بستر رایانش لبه سیار (MEC)[۶] و شبکه‌های دسترسی مبتنی بر مه (F-RANs)[۷] پرداخته شده است. روش‌های ارائه‌شده در [۱۲] و [۲۰] به ترتیب بر مبنای الگوریتم Q-Learning عمیق و تلفیق Q-Learning عمیق و یادگیری انتقالی (TL)[۸] توسعه داده شده‌اند که طبق نتایج آزمایش‌ها و در مقایسه با چند الگوریتم مبنا، به قابلیت اطمینان در خدمات انتها به انتها [۱۲] و کمینه‌سازی توان مصرفی سیستم در طول زمان [۲۰] دست یافته‌اند. لازم به ذکر است که استفاده از یادگیری انتقالی در [۲۰] منجر به تسریع فرایند یادگیری شده است.

به علاوه به منظور بهبود عملکرد در [۹] تا [۱۲] از تکنیک بازپخش تجربه، در [۱۱] و [۱۳] از تکنیک شبکه هدف و در [۱۳] از تکنیک بازپخش تجربه اولویت‌بندی‌شده (PER)[۹] استفاده شده است.

۳- مدل سیستم

در این بخش، اجزای شبکه پیشنهادی خود و نحوه تبدیل مسئله زمان‌بندی به یک مسئله یادگیری تقویتی را بیان می‌نماییم و جهت تعریف اجزای سیستم زمان‌بندی و تعاملات آن با محیط، از چارچوب MDP[۱۰] بهره خواهیم گرفت.

۳-۱ هم‌بندی شبکه

همان گونه که در شکل ۱ ملاحظه می‌گردد، معماری شبکه پیشنهادی دارای سه لایه منطقی است و ساختار درختی دارد. در پایین‌ترین لایه، دستگاه‌های پایانی[۱۱] مستقر هستند که با فواصل زمانی معینی به جمع‌آوری داده‌ها پرداخته و این داده‌ها را از طریق زیرساخت ارتباطی بی‌سیم به سمت نقاط دسترسی[۱۲] و سپس دروازه‌های متناظر خود ارسال می‌کنند. درون هر دروازه، یک زمان‌بند وظیفه مستقر است که تصمیم‌گیری را در خصوص این که هر یک از وظایف ورودی به کدام یک از گره‌های پردازشی لایه مه و یا ابر تخصیص یابند برعهده دارد. در لایه دوم، گره‌های مه[۱۳] مستقر هستند که به یکدیگر و همچنین مرکز داده ابری در لایه سوم متصل‌اند.

با توجه به محدودیت منابع رایانشی موجود در لایه‌های دوم و سوم، این منابع با استفاده از تکنولوژی مجازی‌سازی[۱۴] به چندین ماشین مجازی تقسیم می‌شوند. مقصود از منابع در این ماشین‌های مجازی، پردازنده[۱۵]، حافظه[۱۶] و دستگاه ذخیره‌سازی[۱۷] است. در این پژوهش فرض شده است که مقدار این منابع در هر یک از ماشین‌های مجازی، معین و بدون تغییر هستند.

همان گونه که پیش از این ذکر شد، یکی از چالش‌های موجود در این مسئله، محدودیت منابع سخت‌افزاری دروازه‌ها جهت اجرای روش پیشنهادی است. به منظور غلبه بر این چالش و همچنین بهبود عملکرد، شبکه به چهار مسیر تقسیم شده است. منابع هر یک از این مسیرها توسط یک زمان‌بند مجزا مدیریت می‌گردد. به این ترتیب با کاهش فضاهای حالت و عمل، از حجم پردازش‌های هر یک از زمان‌بندها کاسته شده و به سخت‌افزار کمتری جهت اجرای عملیات زمان‌بندی نیاز خواهد بود.

۳-۲ وظایف ورودی

مجموعه وظایف ورودی به هر یک از زمان‌بندهای شبکه به صورت $Tasks = \{t_1, t_2, t_3, \ldots, t_n\}$ نمایش می‌یابد که اندیس n نشان‌دهنده تعداد وظایف است. به علاوه، نیازمندی‌های هر یک از وظایف ورودی به

1. Internet of Vehicles
2. Asynchronous Advantage Actor-Critic
3. Natural Policy Gradient
4. Vehicular Networks
5. Double Deep Q-Learning
6. Mobile Edge Computing
7. Fog Radio Access Networks
8. Transfer Learning
9. Prioritized Experience Replay
10. Markov Decision Process
11. End Devices
12. Access Point
13. Fog Node
14. Virtualization
15. CPU
16. Memory
17. Storage

صورت $Task_{REQ}=\{dl,cr\}$ تعریف می‌گردد که مقصود از dl، اندازه داده مربوط به وظیفه و مقصود از cr نیز مقدار منبع پردازشی مورد نیاز جهت اجرای وظیفه است. به منظور محاسبه مدت زمان اجرای هر وظیفه از (۱) در [۲۱] استفاده می‌شود

$$ET_j^i(a)=\frac{DL_j^i}{PC_i^a}. \quad (۱)$$

در (۱)، DL_j^i (بر حسب تعداد چرخه‌های[1] مورد نیاز پردازنده) نشان‌دهنده حجم داده‌های وظیفه j در مسیر i و PC_i^a (بر حسب تعداد چرخه در واحد ثانیه) نیز نشان‌دهنده ظرفیت پردازشی ماشین مجازی a در مسیر i است. در نهایت، $ET_j^i(a)$ (بر حسب میلی‌ثانیه) برابر با زمان اجرای وظیفه j با استفاده از ماشین مجازی a و در مسیر i است. در این مقاله، طبق [۲۲] فرض می‌کنیم که پردازش هر بایت از داده، تقریباً به ۱۰۰۰ چرخه پردازنده نیاز داشته باشد و به این ترتیب می‌توان صورت رابطه را به واحد چرخه پردازنده تبدیل نمود. به علاوه طبق [۲۳]، ظرفیت پردازشی یک پردازنده سری Xeon شرکت اینتل با معماری ۶۴بیتی، با یک هسته[2] و سرعت ساعت[3] ۲ گیگاهرتز، معادل با ۱۰۰۰۰ واحد پردازنده است. لذا با استفاده از این رابطه و با فرض این که تمامی پردازنده‌های گره‌های پردازشی ویژگی‌های مذکور را دارند، می‌توان ظرفیت پردازشی ماشین‌های مجازی را به واحد چرخه پردازنده بر ثانیه[4] تبدیل نمود.

۳-۳ محیط و عامل

در این مقاله، ما تنها بخشی از محیط پیرامون عامل را که هر زمان‌بند با آن در تعامل است به عنوان محیط در نظر می‌گیریم. به عبارت دیگر، محیط هر زمان‌بند آن بخشی از منابع گره‌های پردازشی موجود در استخر منابع[5] است که زمان‌بند در هر گام زمانی و پیش و پس از اجرای عمل، حالت آن را مشاهده می‌کند و پاداش نیز از سوی آن بخش به عامل بازگردانده می‌شود. به علاوه در این پژوهش، مقصود از عاملِ تصمیم‌گیرنده، زمان‌بند وظیفه است. لازم به ذکر است که عامل مورد استفاده در این پژوهش، از نوع مبتنی بر DQN[6] است. طبق [۲۴]، DQN نوعی عامل هوشمند است که یادگیری تقویتی را با یک رده از شبکه‌های عصبی مصنوعی تلفیق می‌کند. با توجه به آن که از شبکه عصبی جهت تقریب‌زدن[7] تابع Q استفاده می‌شود، شبکه عصبی مورد استفاده در این عامل Q-Network نامیده دارد [۱۴].

۳-٤ فضای حالت

فضای حالت در نظر گرفته شده برای هر عامل، میزان مصرف منابع[8] در مسیر تحت مدیریت آن عامل است که در قالب یک بردار نمایش داده می‌شود. به منظور محاسبه هر یک از عناصر این بردار، به ترتیب از (۲)، (۳) و (۴) در [۹] استفاده خواهد شد. در هر یک از این روابط، صورت کسر نشان‌دهنده منبع در حال مصرف در لحظه ورود وظیفه j به مسیر i و مخرج کسر نشان‌دهنده مجموع کل مقادیر منبع مورد نظر در مسیر i است

$$uCPU_j^i=\frac{cCPU_j^i}{tCPU_i}, \quad (۲)$$

$$uMemory_j^i=\frac{cMemory_j^i}{tMemory_i}, \quad (۳)$$

$$uStorage_j^i=\frac{cStorage_j^i}{tStorage_i}. \quad (۴)$$

۳-۵ فضای عمل

فضای عمل در این پژوهش به صورت مجموعه‌ای از شناسه‌های[9] ماشین‌های مجازی قابل دسترس و ممکن در مسیر مورد نظر از شبکه در نظر گرفته شده است.

۳-٦ اعتبارسنجی عمل

اجرای هر عمل به معنای تخصیص ماشین مجازی انتخاب‌شده به وظیفه ورودی است. در بخش ۳-۲ ذکر کردیم که هر وظیفه ورودی دارای نیازمندی‌هایی است و چنانچه ماشین مجازی انتخاب‌شده منابعی بزرگ‌تر یا مساوی با این نیازمندی‌ها داشته باشد، آن ماشین مجازی یک انتخاب معتبر و در غیر این صورت، نامعتبر تلقی می‌گردد.

۳-۷ تابع پاداش

تابع پاداش مورد نظر در این پژوهش، تأخیر خدمت[10] است. پاداش آنی[11] محاسبه‌شده، نقش یک سیگنال بازخورد[12] را ایفا می‌نماید که پس از اجرای وظیفه، به زمان‌بند بازگردانده می‌شود. همان گونه که پیش از این نیز بیان شد، هدف اصلی این مقاله کاهش میانگین تأخیر ارائه خدمات برنامه‌های کاربردی است که تمرکز بر این هدف منجر به کاهش تأخیرهای ارسال[13]، انتشار[14]، انتظار و اجرای وظایف خواهد شد

$$IR_j^i(a)=\frac{1}{nSD_j^i}. \quad (۵)$$

در (۵) که در ارتباط با محاسبه پاداش آنی است، nSD_j^i در [۲۵] نشان‌دهنده فرم نرمال‌شده تأخیر خدمت برای وظیفه j در مسیر i است. طبق [۲۵]، تأخیر خدمت شامل تأخیرهای ارسال، انتشار، انتظار و اجرای وظیفه است که در (۵)، مجموع نرمال‌شده حاصل‌جمع این عناصر مورد استفاده قرار می‌گیرد. لازم به ذکر است که مقادیر تأخیر ارسال و انتشار در تمامی ارتباطات نقطه به نقطه (P2P)[15] مسیرهای رفت و برگشت و تأخیرهای انتظار و اجرا تنها در ماشین مجازی مقصد محاسبه می‌شوند و مجموع این مقادیر به عنوان مقدار نهایی در (۵) قرار داده می‌شود. به منظور محاسبه مقادیر تأخیرهای ارسال و انتشار (بر حسب میلی‌ثانیه) در هر یک از ارتباطات نقطه به نقطه نیز به ترتیب از (۶) و (۷) در [۲۵] استفاده می‌شود

1. Cycle
2. Core
3. Clock Speed
4. Cycle Per Second
5. Resource Pool
6. Deep Q-Network
7. Approximate
8. Resource Usage
9. Identifier
10. Service Delay
11. Immediate Reward
12. Feedback
13. Transmission
14. Propagation
15. Point to Point

جدول ۱: مفهوم نمادهای مورد استفاده در چارچوب پیشنهادی.

نماد	مفهوم
vmn	تعداد ماشین‌های مجازی موجود در شبکه
scn	تعداد زمان‌بندهای موجود در شبکه
rmc	ظرفیت حافظه بازپخش
$step_k$	شمارنده تعداد گام‌های آموزش
epn	تعداد تکرارها
CR	آرایه حاوی پاداش تجمعی زمان‌بندها
tsn	تعداد وظایف ورودی به هر زمان‌بند
s و $state_j^i$	حالت جاری مسیر i در لحظه ورود وظیفه j
$flag$	وضعیت اعتبار عمل انتخاب‌شده
a و $action_j^i$	عمل انتخاب‌شده برای وظیفه j در مسیر i
s' و $state_j^{ri}$	حالت بعدی مسیر i پس از تخصیص منابع به وظیفه j
r	پاداش آنی اجرای عمل a
BS	تعداد نمونه‌های انتخاب‌شده از حافظه بازپخش
UF	دوره تناوب به روز رسانی پارامترهای شبکه هدف
α	نرخ یادگیری* مدل
γ	فاکتور کاهش**
DR	نرخ کاهش ε

* Learning Rate
** Discount Factor

$$TD_j^i(a) = \frac{DL_j^i}{TR_j^i} \qquad (۶)$$

در (۶)، DL_j^i (بر حسب گیگابایت) نشان‌دهنده طول داده وظیفه j در مسیر i و TR_j^i (بر حسب گیگابیت بر ثانیه) نیز نشان‌دهنده نرخ ارسال داده‌ها[1] در رسانه ارتباطی مسیر i است

$$PD_j^i(a) = \frac{LL_i}{PS_i} \qquad (۷)$$

در (۷)، LL_i (بر حسب کیلومتر) نشان‌دهنده طول پیوند[2] ارتباطی میان مبدأ و مقصد در مسیر i و PS_i (بر حسب متر بر ثانیه) نیز نشان‌دهنده سرعت انتشار[3] در رسانه ارتباطی در مسیر i است. رابطه مورد نیاز جهت محاسبه زمان انتظار (بر حسب میلی‌ثانیه) نیز به صورت زیر است

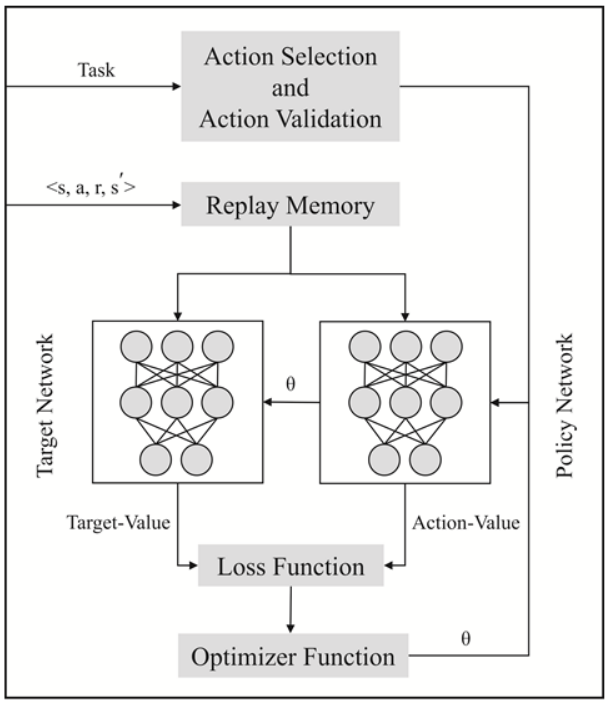
$$WT_j^i(a) = VAT_j^i - TAT_j^i \qquad (۸)$$

در (۸) از [۵]، VAT_j^i نشان‌دهنده زمان تخصیص ماشین مجازی a به وظیفه j در مسیر i و TAT_j^i نیز برابر با زمان ورود وظیفه j به مسیر i است. در خصوص (۱) که مربوط به محاسبه زمان اجرا است نیز پیش از این در بخش ۳-۲ توضیح داده‌ایم.

٤- روش پیشنهادی

در این بخش، به تشریح جزئیات روش پیشنهادی خود برای زمان‌بندی وظایف خواهیم پرداخت. در جدول ۱، کلیه نمادهای مورد استفاده در چارچوب پیشنهادی همراه با مفهوم هر یک آورده شده است.

1. Transmission Rate
2. Link
3. Propagation Speed



شکل ۲: فرایندهای داخلی هر یک از زمان‌بندهای شبکه.

٤-١ نحوه عملکرد زمان‌بند

روش پیشنهادی این مقاله جهت حل مسئله زمان‌بندی وظایف برنامه‌های کاربردی اینترنت اشیا در رایانش مه، DQLTS[4] نام دارد. نحوه عملکرد این روش در الگوریتم‌های اول و دوم شرح داده شده است.

شکل ۲ فرایندهای داخلی هر یک از زمان‌بندها را نشان می‌دهد. به هنگام ورود یک وظیفه، زمان‌بند حالت جاری محیط را مشاهده می‌کند و بر اساس آن و با کمک سیاست $\varepsilon-$greedy یک عمل را انتخاب می‌نماید. به کارگیری سیاست $\varepsilon-$greedy منجر به ایجاد تعادل میان استراتژی‌های اکتشاف[5] و بهره‌برداری[6] از تجربیات پیشین می‌گردد. استراتژی اکتشاف موجب انتخاب یک عمل به صورت تصادفی شده و رویکرد بهره‌برداری نیز با استفاده از شبکه Policy و با پیش‌بینی مقادیر Q، مناسب‌ترین عمل را در حالت جاری انتخاب می‌نماید.

همان گونه که پیش از این نیز ذکر شد، یکی از چالش‌های مطرح در این مسئله، توزیع مناسب وظایف در میان ماشین‌های مجازی به شیوه‌ای است که بهره‌برداری مناسب و مؤثری از این منابع صورت بگیرد. استراتژی اکتشاف در سیاست $\varepsilon-$greedy با انتخاب اعمال تصادفی، بر این چالش غلبه خواهد نمود. پس از اتمام انتخاب، عمل انتخاب‌شده اعتبارسنجی می‌شود. سپس مقدار بعدی پارامتر ε در سیاست $\varepsilon-$greedy که مقداری متغیر است، با استفاده از (۹) در [۱۴] محاسبه می‌گردد

$$\varepsilon_j^i = \max(\varepsilon_{\min}, \varepsilon_{\max} - (\varepsilon_{\max} - \varepsilon_{\min}) \times \frac{step_k}{DS}), \qquad (۹)$$

در (۹)، $\varepsilon_{\min}$ برابر با کمترین مقدار ε، $\varepsilon_{\max}$ برابر با بیشترین مقدار ε، $step_k$ برابر با شمارنده گام‌های آموزش زمان‌بند و DS نیز نشان‌دهنده اندازه گام کاهش ε است.

4. Deep Q-Learning Task Scheduling
5. Exploration
6. Exploitation

در این مقاله، فرض بر این است که هر ماشین مجازی در آنِ واحد قادر به اجرای یک وظیفه است و شیوه زمان‌بندی نیز انحصاری[1] است و نمی‌توان منبع اختصاص داده شده به هر وظیفه را پیش از پایان اجرا از آن دریافت نمود. در صورتی که ماشین مجازی انتخاب‌شده، توسط وظیفه دیگری اشغال[2] باشد، وظیفه ورودی فعلی باید در صف منتظر بماند. شیوه صف‌بندی در این مسئله، اولین ورودی اولین خروجی (FIFO)[3] است و وظیفه‌ای که از لحاظ زمانی زودتر به زمان‌بند وارد گردد، سریع‌تر یک منبع را دریافت خواهد نمود. پس از اتمام اجرای وظیفه، پاداش آنی و حالت بعدی توسط محیط به زمان‌بند بازگردانده شده و مقادیر حالت اولیه (s)، عمل انتخاب‌شده (a)، پاداش آنی (r) و حالت بعدی محیط (s') به عنوان تجربه در حافظه زمان‌بند ذخیره می‌گردد.

به منظور دستیابی به سیاستی مؤثر در زمان‌بندی، مرحله‌ای جهت آموزش شبکه Policy در نظر گرفته شده است. در این مرحله، مقادیر s به عنوان نمونه‌های ورودی به شبکه عصبی داده می‌شوند و مقادیر هدف نیز با استفاده از (۱۰) در [۲۴] و شبکه Target تعیین می‌گردند. جهت ارزیابی عملکرد شبکه عصبی از یک تابع خطا[4] استفاده می‌شود. این تابع میزان اختلاف مقادیر پیش‌بینی شده و مقادیر هدف را محاسبه نموده و حاصل را به تابع بهینه‌ساز[5] ارسال می‌کند. تابع بهینه‌ساز نیز با استفاده از رویکرد شیب کاهشی[6]، وزن‌های شبکه Policy را به گونه‌ای تغییر می‌دهد که مقدار خطا کاهش یابد.

به این ترتیب، وزن‌های شبکه Policy در پایان هر مرحله از آموزش و در راستای دستیابی به یک سیاست زمان‌بندی کارامد، به روز رسانی می‌شوند. لازم به ذکر است که وزن‌های شبکه Target با یک تناوب مشخص (UF) و با استفاده از مقادیر وزن‌های شبکه Policy به روز رسانی می‌شوند

$$Y(s,a)^i_j = \begin{cases} r^i_j & , \ j == tsn \\ r^i_j + \gamma \max_a Q(s',a)^i_j & , \ \text{otherwise} \end{cases} \qquad (۱۰)$$

در (۱۰) جهت محاسبه بخش $Q(s',a)^i_j$ از شبکه هدف استفاده می‌شود.

۴-۲ چارچوب زمان‌بندی پیشنهادی

چارچوب[7] پیشنهادی این مقاله، دارای دو جزء است. جزء اول (الگوریتم ۱) به تنظیم محیط و عامل‌ها اختصاص دارد و جزء دوم (الگوریتم ۲) که در واقع بخش اصلی چارچوب پیشنهادی محسوب می‌گردد، تلفیقی از الگوریتم DQL [۲۴] و چارچوب زمان‌بندی وظایف ارائه‌شده در این مقاله است. در ادامه این بخش، شبه‌کدها و نحوه عملکرد هر دو جزء شرح داده خواهند شد.

الگوریتم ۱: تنظیمات محیط و عامل‌ها

۱: هم‌بندی شبکه را ایجاد و تعداد ماشین‌های مجازی را تعیین نمایید.
۲: **آغاز حلقه اول:** برای $i = ۱$ تا vmn انجام دهید:
۳: پیکربندی منابع ماشین‌های مجازی را تعیین نمایید.
۴: **پایان حلقه اول.**
۵: **آغاز حلقه دوم:** برای $j = ۱$ تا scn انجام دهید:
۶: منابع مسیرهای شبکه را تعیین نمایید.
۷: مقادیر $tCPU_j$، $tMemory_j$ و $tStorage_j$ را محاسبه نمایید.
۸: شبکه‌های Q_{Policy} و Q_{Target} را برای زمان‌بند j ایجاد نمایید.
۹: **پایان حلقه دوم.**
۱۰: **آغاز حلقه سوم:** برای $k = ۱$ تا scn انجام دهید:
۱۱: زمان‌بند را ایجاد و پارامترهای آن را مقداردهی اولیه نمایید.
۱۲: حافظه بازپخشِ زمان‌بند را با ظرفیت rmc ایجاد نمایید.
۱۳: Q_{Policy} و Q_{Target} را با وزن‌های تصادفی مقداردهی نمایید.
۱۴: متغیر $step_k$ را با صفر مقداردهی اولیه نمایید.
۱۵: **پایان حلقه سوم.**
۱۶: محیط زمان‌بندها را ایجاد و پارامترهای آن را مقداردهی نمایید.
۱۷: **آغاز حلقه چهارم:** برای $l = ۱$ تا epn انجام دهید:
۱۸: پارامترهای محیط را به حالت اولیه بازنشانی کنید.
۱۹: چهار زمان‌بند را به طور هم‌زمان و با الگوریتم ۲ اجرا نمایید.
۲۰: **پایان حلقه چهارم.**
۲۱: آرایه CR را ذخیره نمایید.

در پیکربندی ماشین‌های مجازی مستقر در گره‌های مه و مرکز داده ابری، مقادیر ظرفیت پردازشی $(tCPU_j)$، حافظه $(tMemory_j)$ و ظرفیت ذخیره‌سازی $(tStorage_j)$ تعیین شده و به هر ماشین مجازی یک شناسه اختصاص می‌یابد. در ادامه برای هر زمان‌بند دو شبکه عصبیِ Policy و Target ایجاد می‌شوند. سپس زمان‌بندها ایجاد شده و دو شبکه عصبی مذکور همراه با ابعاد فضای حالت و عمل، اندازه دسته نمونه‌ها (BS)، تناوب به روز رسانی شبکه Target (UF)، نرخ یادگیری (α)، نرخ کاهش ε (DR) و فاکتور کاهش (γ) به آنها تخصیص داده می‌شود. به علاوه، ظرفیت حافظه بازپخش (rmc) نیز برای هر عامل تعیین می‌گردد و پس از آن، شبکه Policy با وزن‌های تصادفی و شبکه Target نیز با وزن‌های شبکه Policy مقداردهی اولیه می‌شوند. مقدار شمارنده گام‌های آموزش $(step_k)$ برای هر عامل نیز با صفر مقداردهی اولیه می‌گردد. سپس محیط مسئله ایجاد شده و پارامترهای بیشینه و کمینه ε، تعداد وظایف و ظرفیت استخر منابع برای آن تنظیم می‌شوند. در ابتدای هر تکرار از زمان‌بندی، پارامترهای محیط به مقادیر اولیه خود بازنشانی می‌شوند و سپس تمامی زمان‌بندها به طور هم‌زمان و با استفاده از الگوریتم ۲ اجرا می‌شوند.

الگوریتم ۲: DQLTS

۱: **آغاز حلقه اول:** برای $i = ۱$ تا scn انجام دهید:
۲: متغیر CR_i را با صفر مقداردهی اولیه نمایید.
۳: مجموعه وظایف پایه را به صورتی تصادفی درهم‌ریخته کنید.
۴: **آغاز حلقه دوم:** برای $j = ۱$ تا tsn انجام دهید:
۵: متغیر $step_k$ را با یک واحد افزایش دهید.
۶: بردار $state^i_j$ را با استفاده از (۲) تا (۴) محاسبه نمایید.
۷: متغیر $flag$ را با $false$ مقداردهی اولیه نمایید.
۸: **آغاز حلقه سوم:** مادامی که $flag == false$ انجام دهید:
۹: عمل $action^i_j$ را با سیاست $\varepsilon - greedy$ انتخاب نمایید.
۱۰: نتیجه اعتبارسنجی $action^i_j$ را در $flag$ قرار دهید.
۱۱: **آغاز شرط اول:** اگر مقدار $flag == true$ است:
۱۲: از حلقه سوم خارج شوید.
۱۳: **پایان شرط اول.**
۱۴: **پایان حلقه سوم.**

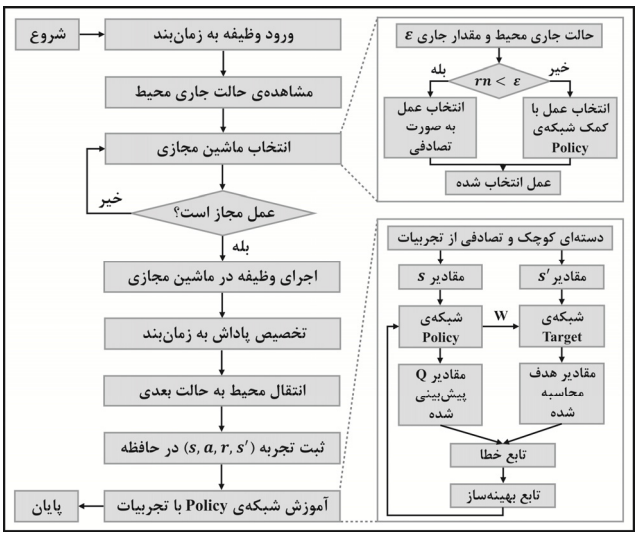
1. Non-Preemptive
2. Busy
3. First In First Out
4. Loss Function
5. Optimizer Function
6. Gradient Descent
7. Framework

۱۵: ماشین مجازی منتخب را به وظیفه j تخصیص دهید.
۱۶: بردار $state'^{i}_{j}$ را با استفاده از (۲) تا (۴) محاسبه نمایید.
۱۷: مقدار $ET^{i}_{j}(a)$ را با استفاده از (۱) محاسبه نمایید.
۱۸: مقدار $WT^{i}_{j}(a)$ را با استفاده از (۸) محاسبه نمایید.
۱۹: ماشین مجازی منتخب را از وظیفه j پس بگیرید.
۲۰: $TD^{i}_{j}(a)$ و $PD^{i}_{j}(a)$ را با (۶) و (۷) محاسبه نمایید.
۲۱: $IR^{i}_{j}(a)$ را با (۵) محاسبه و به متغیر CR_i اضافه نمایید.
۲۲: مقادیر s، a، r و s' را به حافظه اضافه کنید.
۲۳: مقدار جاری ε را با استفاده از (۹) به روز رسانی نمایید.
۲۴: یک دسته با اندازه BS را از حافظه انتخاب نمایید.
۲۵: مقادیر $Y(s,a)^{i}_{j}$ را با استفاده از (۱۰) محاسبه نمایید.
۲۶: شبکه Q_{Policy} را با استفاده از نمونه‌ها آموزش دهید.
۲۷: **آغاز شرط دوم:** اگر حاصل $step_k\%UF$ برابر با صفر بود:
۲۸: وزن‌های Q_{Target} را با Q_{Policy} جایگزین نمایید.
۲۹: **پایان شرط دوم.**
۳۰: **پایان حلقه دوم.**
۳۱: **پایان حلقه اول.**
۳۲: آرایه CR مربوط به تکرار جاری را ذخیره نمایید.

این الگوریتم برای هر تکرار از یک مجموعه وظیفه جداگانه استفاده می‌کند. در ابتدای ورود هر وظیفه، شمارنده گام یک واحد افزایش می‌یابد. سپس حالت جاری محیط با استفاده از (۲) تا (۴) محاسبه شده و با کمک سیاست $\varepsilon-greedy$، یک عمل انتخاب می‌گردد. نحوه عملکرد این سیاست بدین گونه است که ابتدا یک عدد به صورت تصادفی از بازه [۱-۰] انتخاب می‌شود. در صورتی که این عدد کوچک‌تر از مقدار جاری ε باشد، عمل اکتشاف و در غیر این صورت عمل بهره‌برداری صورت می‌گیرد. در عمل بهره‌برداری، حالت جاری به شبکه Policy ارسال شده و بهترین عمل ممکن (عملی با مقدار بیشینه برای Q) با استفاده از این شبکه انتخاب می‌گردد. پس از انتخاب عمل، اعتبارسنجی صورت می‌گیرد تا مشخص گردد که آیا اجرای این عمل در ماشین مجازی انتخاب‌شده امکان‌پذیر است یا خیر.

سپس ماشین مجازی انتخاب‌شده به وظیفه ورودی تخصیص می‌یابد و پس از آن حالت بعدی محیط با استفاده از (۲)، تا (۴) محاسبه می‌شود. وظیفه بر روی ماشین مجازی اجرا می‌گردد و پس از آن زمان اجرا و زمان انتظار وظیفه به ترتیب با استفاده از (۱) و (۸) محاسبه می‌شوند. پس از اتمام اجرای وظیفه، ماشین مجازی آزاد می‌شود. سپس مقادیر تأخیرهای ارسال و انتشار به ترتیب با استفاده از (۶) و (۷) محاسبه می‌شوند و در نهایت مقدار پاداش آنی با استفاده از (۵) محاسبه شده و به مقدار پاداش تجمعی اضافه می‌گردد. در ادامه، تجربه مربوط به این تبدیل حالات در حافظه بازپخش ذخیره و مقدار بعدی ε با استفاده از (۹) محاسبه می‌گردد. در مرحله آموزش، یک دسته کوچک از تجربیات با اندازه BS به صورت تصادفی از حافظه بازپخش انتخاب شده و به شبکه Policy وارد می‌گردد. جهت محاسبه برچسب نمونه‌ها نیز از (۱۰) استفاده می‌شود. در انتها در صورتی که مقدار شمارنده گام‌های آموزش مضربی از پارامتر UF باشد، وزن‌های شبکه Target با وزن‌های شبکه Policy جایگزین می‌گردد. خروجی این الگوریتم، مقدار پاداش تجمعی مربوط به تکرار جاری برای هر یک زمان‌بندها و به صورت تفکیک‌شده است.

در شکل ۳، کلیه مراحلی که هر یک از زمان‌بندها در هر یک از گام‌های تصمیم‌گیری و به محض ورود هر وظیفه طی می‌کنند، در قالب یک روندنما نمایش داده شده است. این مراحل، پیش از این و در



شکل ۳: روندنمای هر یک از گام‌های تصمیم‌گیری توسط زمان‌بند.

بخش‌های ۴-۱ و ۴-۲ با جزئیات بیشتر مورد بررسی قرار گرفته‌اند.

لازم به ذکر است که قسمت مربوط به جزئیات "آموزش شبکه Policy با تجربیات" در شکل ۲، پس از به روز رسانی وزن‌های شبکه Policy (در همه تکرارها) و Target (هر چند تکرار یک بار) به پایان می‌رسد.

۵- ارزیابی کارایی روش ارائه‌شده

در این بخش به معرفی معیارها، الگوریتم‌های مبنا و نتایج ارزیابی کارایی الگوریتم DQLTS می‌پردازیم.

۵-۱ محیط شبیه‌سازی

در این مقاله، محیط شبیه‌سازی با استفاده از زبان برنامه‌نویسی Python و کتابخانه‌های SimPy [۲۶] و Keras [۲۷] ایجاد شده که به ترتیب جهت ساخت چارچوب زمان‌بندی و ساخت شبکه عصبی مورد استفاده قرار می‌گیرد.

۵-۲ طرح شبیه‌سازی

۵-۲-۱ زیرساخت شبکه

زیرساخت مورد نیاز برای اجرای آزمایش‌ها، تعداد اجزا و هم‌بندی آنها مطابق با شکل ۱ شبیه‌سازی شده است. همان گونه که پیش از این نیز ذکر شد در هر دروازه، یک زمان‌بند وظیفه مستقر شده است. گره‌های مه و سرورهای مستقر در مرکز داده ابری نیز گره پردازشی در نظر گرفته شده و منابع موجود در آنها با استفاده از تکنولوژی مجازی‌سازی، میان چند ماشین مجازی تقسیم می‌شود. این طرح در مجموع شامل ۲۹ ماشین مجازی است که ۵ مورد متعلق به مرکز داده ابری و ۲۴ مورد دیگر متعلق به گره‌های مه هستند. در هر یک از گره‌های مه، چهار ماشین مجازی در نظر گرفته شده است. در این طرح، چهار مسیر در شبکه وجود دارد که هر مسیر شامل یک دروازه، یک زمان‌بند و دو گره مه است و در نهایت به مرکز داده ابری منتهی می‌گردد. بنابراین منابع مرکز داده ابری در میان هر چهار مسیر و گره‌های مه سطح دوم در میان دو مسیر مشترک هستند.

در این مقاله، به منظور تنظیم پیکربندی ماشین‌های مجازی از مسیریاب‌های سری ISR 4K شرکت سیسکو (ISR ۴۳۲۱، ISR ۴۳۳۱ و ISR ۴۴۵۱) [۲۸] و جهت پیکربندی سرورهای مستقر در مرکز داده ابری نیز از سرورهای سری UCS شرکت سیسکو (C۲۲۰ LFF M۴،

C۲۲۰ LFF M۵، C۲۴۰ LFF M۴ و C۲۴۰ SFF M۵) استفاده می‌کنیم [۲۹] تا [۳۲].

اتصال حسگرها به دروازه‌ها به صورت بی‌سیم و استاندارد ارتباطی مابین آنها IEEE ۸۰۲.۱۱n [۳۳] و سایر اتصالات شبکه نیز فیبر نوری فرض می‌شوند. فاصله میان حسگرها تا دروازه‌های متناظرشان برابر با ۱۴۰ متر، فاصله میان دروازه‌ها تا گره‌های مه سطح اول برابر با ۱ کیلومتر، فاصله گره‌های مه سطح اول و دوم برابر با ۳ کیلومتر و فاصله گره‌های مه سطح دوم تا مرکز داده ابری برابر با ۱۲ کیلومتر در نظر گرفته می‌شود. نرخ انتقال[1] در ارتباطات حسگرها به دروازه‌ها، دروازه‌ها به گره‌های مه، گره‌های مه به یکدیگر و گره‌های مه به مرکز داده ابری به ترتیب برابر با ۶۰۰ مگابیت بر ثانیه [۳۳]، ۱ گیگابیت بر ثانیه، ۱/۵ گیگابیت بر ثانیه و ۲ گیگابیت بر ثانیه و تمامی سرعت‌های انتشار نیز برابر با ۳×۱۰^{۸} متر بر ثانیه در نظر گرفته شده است.

۵-۲-۲ وظایف

در این مقاله، وظایف مورد استفاده جهت زمان‌بندی به صورت مصنوعی تولید می‌شوند و نیازمندی‌های آنها به گونه‌ای است که امکان اجرای تمامی وظایف در گره‌های مه وجود داشته باشد. نیازمندی‌های پردازشی وظایف، از بازه [۴۰۰-۵۰] واحد و اندازه داده‌ها نیز از بازه [۱۰۰-۱۰] کیلوبایت انتخاب شده است. در اینجا فرض می‌شود که حجم داده‌های ورودی و خروجی یکسان هستند. شایان ذکر است که به منظور عمومیت‌بخشیدن به استراتژی زمان‌بندی پیشنهادی در این پژوهش، در ابتدای هر یک از تکرارها، مجموعه وظایف درهم‌ریخته[2] می‌شود. به این ترتیب، زمان‌بندها هر یک از تکرارها را از یک حالت اولیه تصادفی آغاز می‌کنند [۳۴].

۵-۲-۳ شبکه عصبی عمیق

شبکه‌های عصبی عمیق مورد استفاده در این پژوهش از نوع تماماً متصل هستند و هر یک در مجموع ۵۵۷ پارامتر قابل یادگیری دارند. به عنوان تابع فعال‌ساز مورد استفاده در لایه‌های ماقبل آخر از ReLU، به عنوان تابع بهینه‌ساز از Adam و به عنوان تابع خطا از میانگین مربع خطا (MSE)[3] استفاده شده است. هر زمان‌بند در طی ۱۰۰ تکرار، ۱۰۰۰۰ گام آموزش را طی می‌نماید. مقادیر پارامترهای شبکه‌ها نیز با استفاده از چند آزمایش تنظیم خواهند شد و بهترین مقادیر حاصل در آزمایش‌های مقایسه‌ای نهایی مورد استفاده قرار خواهند گرفت.

۵-۳ معیارهای ارزیابی

در این مقاله از شش معیار ارزیابی جهت بررسی عملکرد روش ارائه‌شده و سپس مقایسه آن با الگوریتم‌های مبنا استفاده می‌شود. این معیارها عبارتند از میانگین زمان انتظار (AWT)[4]، میانگین زمان پاسخ (ART)[5]، میانگین تأخیر ارائه خدمات (ASD)[6]، میانگین زمان تکمیل وظایف (AMST)[7]، میانگین زمان اتمام وظایف (AFT)[8] و میانگین نرخ استقرار وظایف. به علاوه، جهت بررسی عملکرد الگوریتم پیشنهادی در تصمیم‌گیری از میانگین خطای مطلق (AMAE)[9] استفاده می‌شود.

۵-۴ روش‌های مبنا جهت مقایسه

الگوریتم‌های مبنای مورد استفاده جهت ارزیابی عملکرد روش پیشنهادی این مقاله عبارتند از:

QLTS[10]: این الگوریتم، تلفیقی از الگوریتم Q-Learning و چارچوب زمان‌بندی وظایف ایجادشده در این پژوهش است. مطابق با [۳۵]، الگوریتم Q-Learning یکی از پرکاربردترین الگوریتم‌های مبتنی بر تکنیک اختلاف زمانی (TD)[11] و از نوع بدون سیاست[12] است و زمان همگرایی قابل قبولی دارد. لازم به ذکر است که در [۵] و [۱۵] تا [۱۷] از الگوریتم Q-Learning جهت توسعه روش‌های پیشنهادی و در [۳۷] نیز جهت مقایسه با روش پیشنهادی استفاده شده است.

RS[13]: این الگوریتم، در چارچوب زمان‌بندی ایجادشده مقاله، هر وظیفه را به صورت تصادفی به یکی از ماشین‌های مجازی موجود در استخر منابع اختصاص می‌دهد. با توجه به آن که در بخش اکتشاف سیاست $\varepsilon-$greedy، انتخاب‌ها صرفاً به صورت تصادفی انجام می‌شوند، لذا لازم است که عملکرد روش پیشنهادی این مقاله با الگوریتم RS مقایسه شود تا به این ترتیب میزان هوشمندی عامل و همچنین میزان تأثیر یادگیری از تجربیات قبلی در عملکرد عامل مشخص گردد. الگوریتم RS در [۱۷]، [۱۱]، [۱۲]، [۲۰] و [۳۶] جهت مقایسه با روش‌های پیشنهادشده مورد استفاده قرار گرفته است.

FF[14]: در این الگوریتم، به هنگام ورود هر یک از وظایف، ابتدا دسترس‌پذیری[15] منابع بررسی می‌گردد و در نهایت وظیفه جاری، جهت پردازش به نزدیک‌ترین ماشین مجازیِ در دسترس تخصیص می‌یابد. لازم به ذکر است که در این الگوریتم، ماشین‌های مجازی مستقر در گره‌های مه از اولویت بالاتری جهت انتخاب برخوردارند. هدف از انتخاب این الگوریتم آن است که به بررسی این پرسش بپردازیم که آیا اولویت‌دادن به اجرای اغلب وظایف در لایه رایانش مه موجب حصول نتایج مؤثرتری از لحاظ تابع هدف مسئله خواهد شد و یا انتخاب ماشین مجازی بر حسب شرایط شبکه می‌تواند به نتیجه مطلوب‌تری منجر گردد. این الگوریتم در [۳۷] معرفی شده است و گونه‌ای از آن در [۱۷] جهت مقایسه مورد استفاده قرار گرفته است.

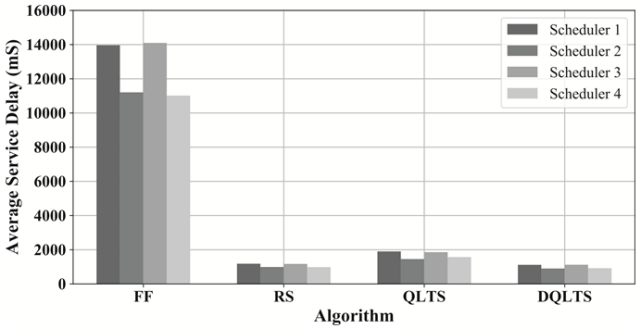
۵-۵ نتایج آزمایش‌های مقایسه‌ای ارزیابی کارایی

در این زیربخش، نتایج حاصل از ارزیابی روش پیشنهادی ارائه خواهد شد و بر روی نتایج حاصل بحث خواهیم نمود. لازم به ذکر است که نمودارهای ترسیم‌شده نتایج حاصل را به تفکیک هر یک از زمان‌بندها نمایش می‌دهند. ارتفاع هر یک از میله‌ها با محاسبه میانگین معیار ارزیابی مورد نظر در ۱۰۰ تکرار و برای ۱۰۰ وظیفه به دست آمده است.

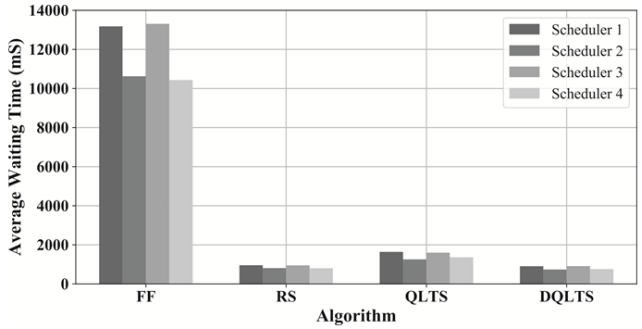
۵-۵-۱ پارامترهای مسئله

در این مقاله، ابتدا پارامترهای الگوریتم DQLTS طی چندین آزمایش اولیه تنظیم شده‌اند و اجرای نهایی این الگوریتم جهت مقایسه با

1. Transmission Rate
2. Shuffle
3. Mean Square Error
4. Average Waiting Time
5. Average Response Time
6. Average Service Delay
7. Average Makespan Time
8. Average Finish Time
9. Average Mean Absolute Error
10. Q-Learning Task Scheduling
11. Temporal Difference
12. Off-Policy
13. Random Selection
14. First Fit
15. Availability



شکل ۴: مقایسه کارایی الگوریتم‌ها از لحاظ میانگین تأخیر ارائه خدمات.



شکل ۵: مقایسه کارایی الگوریتم‌ها از لحاظ میانگین زمان انتظار.

جدول ۲: بهترین مقادیر پارامترهای مسئله.

Parameter	α	γ	DR	BS	UF
Best Value	۰٫۰۱	۰٫۹۵	۳۰۰۰۰	۶۴	۵۰۰

الگوریتم‌های مبنا با استفاده از این مقادیر (جدول ۲) انجام شده است. به منظور مقایسه بهتر، پارامترهای الگوریتم QLTS نیز مشابه با الگوریتم DQLTS تنظیم شده‌اند. لازم به ذکر است که نرخ جمع‌آوری داده توسط حسگرها برابر با ۱ میلی‌ثانیه، تعداد تکرارها برابر با ۱۰۰ و ظرفیت حافظه هر زمان‌بند نیز برابر با ۱۰۰۰۰ نمونه در نظر گرفته شده است. در آزمایش‌های تنظیم پارامتر، طبق [۳۸] از اعداد موجود در بازه ۰٫۹۵ الی ۰٫۹۹ استفاده شده است.

۵-۵-۲ میانگین تأخیر ارائه خدمات

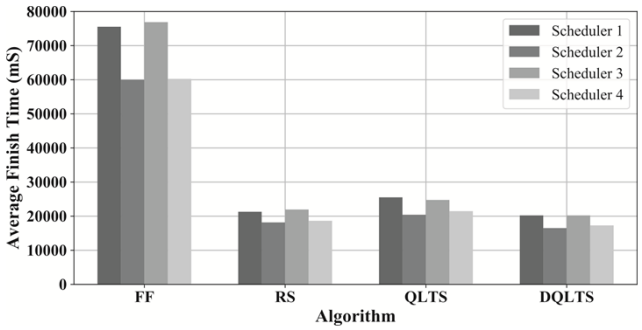
شکل ۴ به مقایسه الگوریتم‌های مورد آزمایش از لحاظ تأخیر ارائه خدمات اختصاص یافته است. همان گونه که ملاحظه می‌گردد، کمترین مقادیر به الگوریتم DQLTS تعلق دارند. دلیل این امر آن است که تأخیر ارائه خدمات، همان تابع پاداش الگوریتم DQLTS است و تمرکز بر روی این هدف موجب شده که این الگوریتم دارای کمترین مقدار ASD در مقایسه با سایرین باشد.

۵-۵-۳ میانگین زمان انتظار

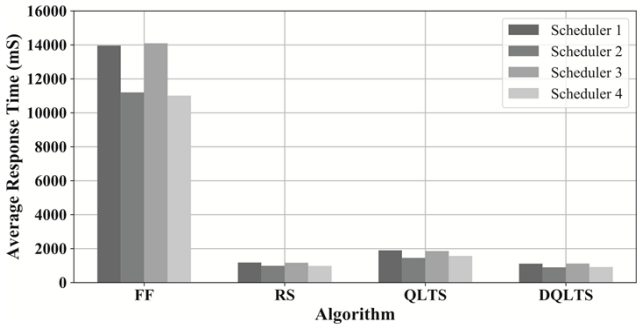
شکل ۵ مقایسه عملکرد الگوریتم پیشنهادی با الگوریتم‌های مبنای دیگر را از لحاظ میانگین زمان انتظار نشان می‌دهد. همان گونه که ملاحظه می‌گردد، زمان‌بندهای الگوریتم پیشنهادی از لحاظ میانگین زمان انتظار کمترین مقدار را در مقایسه با سایرین دارند. دلیل این امر آن است که در الگوریتم DQLTS، زمان انتظار بخشی از تابع پاداش است و به این ترتیب، یکی از اهداف هر عامل کاهش میزان زمان انتظار وظیفه جهت دریافت منبع است.

۵-۵-۴ میانگین زمان اتمام وظایف

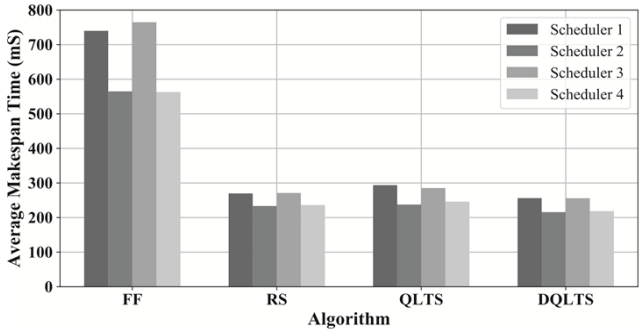
مطابق با شکل ۶، کمترین میزان زمان اتمام وظایف مربوط به



شکل ۶: مقایسه کارایی الگوریتم‌ها از لحاظ میانگین زمان اتمام وظایف.



شکل ۷: مقایسه کارایی الگوریتم‌ها از لحاظ میانگین زمان پاسخ.



شکل ۸: مقایسه کارایی الگوریتم‌ها از لحاظ میانگین زمان تکمیل وظایف.

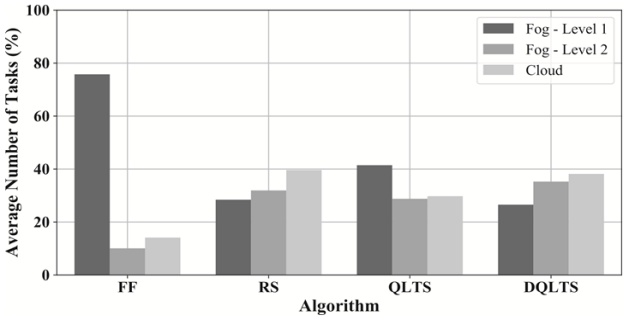
الگوریتم پیشنهادشده در این مقاله است. دلیل این امر آن است که زمان اجرای وظایف، بخشی از تابع هدف زمان‌بندها است و همین امر موجب می‌گردد که زمان‌بندها تصمیمات خود را در راستای کاهش زمان اجرای وظایف اتخاذ نمایند. طبق [۳۹]، زمان اتمام وظایف به صورت مجموع زمان صرف‌شده جهت اجرای تمامی وظایف در نظر گرفته شده است.

۵-۵-۵ میانگین زمان پاسخ

در شکل ۷، عملکرد الگوریتم پیشنهادی این مقاله از لحاظ میانگین زمان پاسخ در مقایسه با الگوریتم‌های مبنا به تصویر کشیده شده است. شکل ۷ نشان می‌دهد که الگوریتم DQLTS در مقایسه با سایر الگوریتم‌ها، بهترین عملکرد را از لحاظ زمان پاسخ دارد. دلیل این امر آن است که زمان پاسخ برابر با فاصله زمانی میان لحظه ورود وظیفه به زمان‌بند تا خروج آن است. این فاصله، شامل زمان انتظار و همچنین زمان اجرای وظیفه است و به دلیل آن که زمان‌های اجرا و انتظار، بخشی از تابع هدف الگوریتم پیشنهادی این مقاله هستند، لذا کمترین میزان زمان پاسخ را در الگوریتم DQLTS شاهد هستیم.

۵-۵-۶ میانگین زمان تکمیل وظایف

شکل ۸، میانگین زمان تکمیل وظایف را برای چهار الگوریتم مورد بررسی در این مقاله نشان می‌دهد. طبق این نمودار، الگوریتم DQLTS به واسطه استفاده مؤثر از منابع پردازشی موجود و همچنین به دلیل داشتن



شکل ۹: مقایسه کارایی الگوریتم‌ها از لحاظ میانگین نرخ استقرار وظایف.

جدول ۳: مقایسه میانگین عملکرد زمان‌بندهای شبکه.

Metric (mS)	FF	RS	QLTS	DQLTS
ASD	۱۲۵۷۰٫۳۸	۱۰۸۲٫۴۶	۱۶۹۷٫۵۸	۱۰۱۶٫۱۵
AWT	۱۱۸۸۶٫۹۱	۸۸۰٫۰۸	۱۴۶۵٫۰۶	۸۲۸٫۲۷
AFT	۶۸۱۶۶٫۵۱	۲۰۰۱۷٫۲۴	۲۳۰۴۴٫۶۱	۱۸۵۶۷٫۱۹
ART	۱۲۵۶۹٫۰۵	۱۰۸۰٫۷۳	۱۶۹۵٫۹۸	۱۰۱۴٫۴۲
AMST	۶۵۸٫۱۴	۲۵۲٫۶۱	۲۶۵٫۶۹	۲۳۶٫۶۰

کمترین میانگین زمان اتمام وظایف (بخش ۵-۵-۴) دارای کمترین زمان تکمیل وظایف در مقایسه با سه الگوریتم مبنای دیگر است.

۵-۵-۷ میانگین نرخ استقرار وظایف

شکل ۹ به مقایسه نرخ استقرار وظایف[1] در سه سطح از هم‌بندی شبکه (گره‌های مه سطح اول، گره‌های مه سطح دوم و مرکز داده ابری) می‌پردازد. مطابق با شکل ۹، الگوریتم پیشنهادی این مقاله به شیوه مؤثرتری وظایف را توزیع می‌نماید و موجب می‌گردد که علاوه بر زمان‌بندی وظایف با کمترین میزان تأخیر ارائه خدمات، وظایف به گونه‌ای در میان ماشین‌های مجازی موجود توزیع گردند که زمان بیکاری[2] آنها کاهش و مصرف منابع[3] افزایش یابد. در شبکه پیشنهادی این مقاله، چنانچه درصد بالایی از وظایف، بدون توجه به شرایط محیطی شبکه به گره‌های مه سطح اول ارسال شوند، (مانند الگوریتم FF) در این صورت عملکرد مطلوبی را از لحاظ معیارهای ارزیابی شاهد نخواهیم بود.

۵-۵-۸ میانگین خطای مطلق

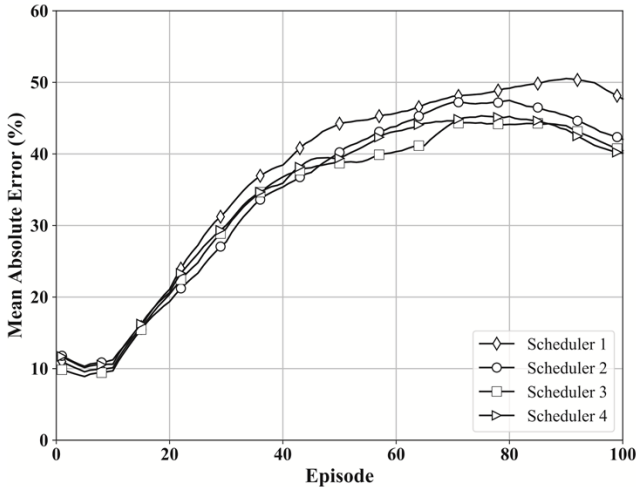
شکل ۱۰، روند تغییرات خطای زمان‌بندها (قدر مطلق اختلاف مقدار پیش‌بینی شده و هدف) را در روش پیشنهادی این مقاله نمایش می‌دهد.

همان گونه که ملاحظه می‌گردد از حدود تکرار ۴۰، روند افزایش خطا کندتر شده و از تکرار ۷۰ به بعد، در نیمی از زمان‌بندها، شاهد روند کاهشی خطا هستیم. کاهش تدریجی خطا نشان می‌دهد که به مرور زمان، یادگیری زمان‌بندها بهبود یافته و تصمیم‌گیری‌ها مؤثرتر شده است.

۵-۵-۹ میانگین عملکرد زمان‌بندها از لحاظ معیارهای ارزیابی

در جدول ۳، نتیجه محاسبه میانگین عملکرد زمان‌بندهای شبکه در چهار الگوریتم مورد بررسی در مقاله نشان داده شده است. این جدول به مقایسه الگوریتم پیشنهادی مقاله و سه الگوریتم مبنا از لحاظ پنج معیار ارزیابی می‌پردازد. همان گونه که ملاحظه می‌گردد، الگوریتم پیشنهادی مقاله دارای کمترین مقادیر میانگین در مقایسه با سه الگوریتم دیگر است و به این ترتیب کارایی الگوریتم DQLTS تأیید می‌گردد.

1. Task Placement Ratio
2. Idle Time
3. Resource Utilization



شکل ۱۰: بررسی عملکرد الگوریتم DQLTS از لحاظ میانگین خطای مطلق.

۶- نتیجه‌گیری و کارهای آتی

در این مقاله به مسئله زمان‌بندی وظایف ایجادشده توسط برنامه‌های کاربردی اینترنت اشیا پرداخته شد. با توجه به ماهیت اغلب برنامه‌های کاربردی اینترنت اشیا، تأخیر زمانی از اهمیت ویژه‌ای برخوردار است. لذا در این پژوهش نیز بر روی این فاکتور به عنوان هدف اصلی تمرکز شده و تأخیر در ارائه خدمات به عنوان تابع هدف انتخاب گشته است. به منظور حل این مسئله، از تلفیق الگوریتم Q-Learning، یادگیری عمیق و تکنیک‌های بازپخش تجربه و شبکه هدف استفاده شده است. به علاوه، جهت غلبه بر محدودیت‌های سخت‌افزاری گره‌های مه، چهار زمان‌بند مجزا برای گروه‌های معینی از حسگرهای شبکه در نظر گرفته شده است. سپس به منظور ارزیابی عملکرد روش پیشنهادی مقاله، از شش معیار ارزیابی (AFT، AMST، ART، ASD، AWT و Placement Level) و سه الگوریتم مبنا (QLTS، RS و FF) استفاده شده است. نتایج حاصل از اجرای آزمایش‌ها نشان می‌دهد که الگوریتم DQLTS از لحاظ معیار ASD، ۷۶٪ بهتر از الگوریتم QLTS و ۶٫۵٪ بهتر از الگوریتم RS عمل می‌نماید. عملکرد ضعیف QLTS نسبت به DQLTS در شرایط مشابه را نیز می‌توان به همگرایی سریع‌تر روش پیشنهادی این مقاله نسبت داد.

با توجه به نتایج برخی از تحقیقات اخیر، استفاده از عامل‌های تصادفی و پیش‌آموزش‌دیده[4] در ابتدای عملیات زمان‌بندی می‌تواند موجب بهبود عملکرد زمان‌بندها گردد. به علاوه، استفاده از تکنیک PER نیز با اولویت‌بندی تجربیات موجود در حافظه منجر به بهبود فرایند یادگیری عامل‌ها خواهد شد. لذا در پژوهش‌های آتی به این دو مورد پرداخته خواهد شد.

مراجع

[1] Cisco, Fog Computing and the Internet of Things: Extend the Cloud to Where the Things Are, Cisco White Paper, 2015.
[2] M. Iorga, *et al.*, *Fog Computing Conceptual Model*, NIST SP-500-325, 2018.
[3] R. Mahmud, R. Kotagiri, and R. Buyya, "Fog computing: a taxonomy, survey and future directions," *Internet of Everything*, pp. 103-130, Springer, 17 Oct. 2018.
[4] OpenFog Consortium and Others, *OpenFog Reference Architecture for Fog Computing*, Architecture Working Group, 2017.

4. Pre-Trained

[5] D. Cui, Z. Peng, W. Lin, *et al.*, "A reinforcement learning-based mixed job scheduler scheme for grid or iaas cloud," *IEEE Trans. on Cloud Computing*, vol. 8, no. 4, pp. 1030-1039, Oct.-Dec. 2017.

[۶] س. حورعلی، ش. جمالی و ف. حورعلی، "ارائه یک الگوریتم توازن بار نامتمرکز در محیط‌های ناهمگن،" *نشریه مهندسی برق و مهندسی کامپیوتر ایران، ب- مهندسی کامپیوتر*، صص. ۱۵۴-۱۴۷، دوره ۱۴، شماره ۲، تابستان ۱۳۹۵.

[7] M. Wang, Y. Cui, X. Wang, S. Xiao, and J. Jiang, "Machine learning for networking: workflow, advances and opportunities," *IEEE Network*, vol. 32, no. 2, pp. 92-99, Nov. 2018.

[8] H. Mao, M. Alizadeh, I. Menache, and S. Kandula, "Resource management with deep reinforcement learning," in *Proc. of the 15th ACM Workshop on Hot Topics in Networks*, *HotNet'16*, pp. 50-56, Nov. 2016.

[9] Q. Zhang, M. Lin, L. T. Yang, Z. Chen, and P. Li, "Energy-efficient scheduling for real-time systems based on deep Q-learning model," *IEEE Trans. on Sustainable Computing*, vol. 4, no. 1, pp. 132-141, Aug. 2017.

[10] N. Liu, *et al.*, "A hierarchical framework of cloud resource allocation and power management using deep reinforcement learning," in *Proc. IEEE 37th Int. Conf. on Distributed Computing Systems, ICDCS'17*, vol. 1, pp. 372-382, 2017.

[11] Y. Wei, F. R. Yu, M. Song, and Z. Han, "Joint optimization of caching, computing, and radio resources for fog-enabled iot using natural actor-critic deep reinforcement learning," *IEEE Internet of Things J.*, vol. 6, no. 2, pp. 2061-2073, Oct. 2018.

[12] T. Yang, Y. Hu, M. C. Gursoy, A. Schmeink, and R. Mathar, "Deep reinforcement learning based resource allocation in low latency edge computing networks," in *Proc. IEEE 15th Int. Symp. on Wireless Communication Systems, ISWCS'18,* , 5 pp., Lisbon, Portugal, 28-31 Aug. 2018.

[13] Y. Wang, K. Wang, H. Huang, T. Miyazaki, and S. Guo, "Traffic and computation co-offloading with reinforcement learning in fog computing for industrial applications," *IEEE Trans. on Industrial Informatics*, vol. 15, no. 2, pp. 976-986, Nov. 2018.

[14] S. Ravichandiran, Hands-on Reinforcement Learning with Python: Master Reinforcement and Deep Reinforcement Learning Using OpenAI Gym and TensorFlow, Packt Publishing Ltd, p. 303, 2018.

[15] M. H. Moghadam and S. M. Babamir, "Makespan reduction for dynamic workloads in cluster-based data grids using reinforcement-learning based scheduling," *J. of Computational Science*, vol. 24, pp. 402-412, Jan. 2018.

[16] A. I. Orhean, F. Pop, and I. Raicu, "New scheduling approach using reinforcement learning for heterogeneous distributed systems," *J. of Parallel and Distributed Computing*, vol. 117, pp. 292-302, Jul. 2018.

[17] Z. Peng, D. Cui, J. Zuo, Q. Li, B. Xu, and W. Lin, "Random task scheduling scheme based on reinforcement learning in cloud computing," *Cluster Computing*, vol. 18, no. 4, pp. 1595-1607, Dec. 2015.

[18] G. Qiao, S. Leng, and Y. Zhang, "Online learning and optimization for computation offloading in d2d edge computing and networks," *Mobile Networks and Applications*, 12 pp., Jan. 2019.

[19] Q. Qi, J. Wang, Z. Ma, H. Sun, Y. Cao, L. Zhang, and J. Liao, "Knowledge-driven service offloading decision for vehicular edge computing: a deep reinforcement learning approach," *IEEE Trans. on Vehicular Technology*, vol. 68, no. 5, pp. 4192-4203, Jan. 2019.

[20] Y. Sun, M. Peng, and S. Mao, "Deep reinforcement learning based mode selection and resource management for green fog radio access networks," *IEEE Internet of Things J.*, vol. 6, no. 2, pp. 1960-1971, Sept. 2018.

[21] L. Yin, J. Luo, and H. Luo, "Tasks scheduling and resource allocation in fog computing based on containers for smart manufacturing," *IEEE Trans. on Industrial Informatics*, vol. 14, no. 10, pp. 4712-4721, Jun. 2018.

[22] A. P. Miettinen and J. K. Nurminen, "Energy efficiency of mobile clients in cloud computing," *HotCloud*, vol. 10, pp. 4-4, Jun. 2010.

[23] Cisco, *IoX App Concepts: Application Resource Profiles, Developer, Cisco*, 2019, URL: https://developer.cisco.com/docs/iox/#!application-resource-profiles/resource-profiles

[24] V. Mnih, *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529-533, Feb. 2015.

[25] A. Yousefpour, G. Ishigaki, R. Gour, and J. P. Jue, "On reducing IoT service delay via fog offloading," *IEEE Internet of Things J.*, vol. 5, no. 2, pp. 998-1010, Jan. 2018.

[26] T. SimPy, *Simpy: Discrete Event Simulation for Python*, Python Package Version, vol. 3, no. 9, 2017. URL: https://simpy.readthedocs.io/en/latest/

[27] F. Chollet, *et al.*, *Keras: The Python Deep Learning Library*, Astrophysics Source Code Library, 2018.

[28] Cisco, Cisco 4000 Family Integrated Services Router, Cisco Datasheet, 2018.

[29] Cisco, *Cisco UCS C220 M4 Rack Server*, Cisco Datasheet, 2018.

[30] Cisco, *Cisco UCS C220 M5 Rack Server*, Cisco Datasheet, 2019.

[31] Cisco, *Cisco UCS C240 M4 Rack Server*, Cisco Datasheet, 2016.

[32] Cisco, *Cisco UCS C240 M5 Rack Server*, Cisco Datasheet, 2019.

[33] S. Sendra, M. Garcia Pineda, C. Turro Ribalta, and J. Lloret, "Wlan ieee 802.11 a/b/g/n indoor coverage and interference performance study," *International J. on Advances in Networks and Services*, vol. 4, no. 1, pp. 209-222, 2011.

[34] C. Zhang, O. Vinyals, R. Munos, and S. Bengio, *A Study on Overfitting in Deep Reinforcement Learning*, arXiv preprint arXiv:1804.06893, 2018.

[35] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, MIT Press, 2018.

[36] J. Zhu, *et al.*, "A new deep-Q-learning-based transmission scheduling mechanism for the cognitive internet of Things," *IEEE Internet of Things J.*, vol. 5, no. 4, pp. 2375-2385, Aug. 2017.

[37] O. Skarlat, *et al.* "Optimized IoT service placement in the fog," *Service Oriented Computing and Applications*, vol. 11, no. 4, pp. 427-443, Dec. 2017.

[38] E. Akhtar and S. Farrukh, Practical Reinforcement Learning: Develop self-evolving, Intelligent Agents with OpenAI Gym, Python and Java, Packt Publishing, p. 146, 2017.

[39] A. Kapsalis, P. Kasnesis, I. S. Venieris, D. I. Kaklamani, and C. Z. Patrikakis, "A cooperative fog approach for effective workload balancing," *IEEE Cloud Computing*, vol. 4, no. 2, pp. 36-45, Apr. 2017.

پگاه گازری در سال ۱۳۹۳ مدرک کارشناسی مهندسی فناوری اطلاعات خود را از دانشگاه پیام نور تهران دریافت نموده و از سال ۱۳۹۶، دوره کارشناسی ارشد مهندسی فناوری اطلاعات را در دانشگاه قم آغاز کرده و هم‌اکنون مشغول به تحصیل است. زمینه‌های مطالعاتی و تحقیقاتی وی عبارتند از: مسیریابی، سوئیچینگ، یادگیری ماشین و زمان‌بندی وظایف در رایانش مه و رایانش ابری.

دادمهر رهبری در سال ۱۳۸۵ مدرک کارشناسی مهندسی کامپیوتر خود را از دانشگاه علم و صنعت ایران و در سال ۱۳۸۸ مدرک کارشناسی ارشد مهندسی کامپیوتر خود را از دانشگاه آزاد اسلامی واحد مشهد دریافت نموده است. وی در سال ۱۳۹۵ به دوره دکتری مهندسی فناوری اطلاعات در دانشگاه قم وارد گردید و هم‌اکنون به صورت تمام‌وقت مشغول به تحصیل است. زمینه‌های علمی مورد علاقه وی عبارتند از: زمان‌بندی و مدیریت منابع در محیط ابر و مه، یادگیری ماشین و امنیت اطلاعات.

محسن نیک‌رأی در سال ۱۳۸۱ مدرک کارشناسی مهندسی کامپیوتر خود را از دانشگاه علم و صنعت ایران و مدرک کارشناسی ارشد و دکتری مهندسی کامپیوتر خود را در سال‌های ۱۳۸۵ و ۱۳۹۲ از دانشگاه تهران دریافت نموده است. دکتر نیک‌رأی از سال ۱۳۹۵ در گروه مهندسی کامپیوتر و فناوری اطلاعات دانشگاه قم به عنوان هیأت علمی مشغول به فعالیت است. زمینه‌های علمی مورد علاقه وی عبارتند از: زمان‌بندی و مدیریت منابع در محیط ابر و مه.