7th International Conference on Web Research
May 19-20, 2021; Tehran, IRAN ICWR2021

Optimization of fog resource allocation in IoT using Deep Reinforcement Learning

Danoosh Cahamani[1], Parsa Vafaei[1], Zahra Movahedi[2,*]

[1]BSc Graduate in Computer Engineering
Department of Computer Engineering, Faculty of Engineering, College of Farabi, University of Tehran, Iran
{danoosh.chamani, parsa.vafaei}@ut.ac.ir

[2]Assistant Professor
Department of Computer Engineering, Faculty of Engineering, College of Farabi, University of Tehran, Iran
{zmovahedi}@ut.ac.ir

Abstract

Fog computing is an emerging paradigm that extends the cloud concept to the edge. It provides computing, storage, control, and networking capabilities for realizing the Internet of Things (IoT) applications. In the fog computing concept, the IoT devices offload its data or computationally expensive tasks to the fog nodes within its proximity, instead of distant cloud. In this paper, we address the problem of optimal allocating the limited resources of fog nodes to the IoT applications. Current approaches of fog resource allocation are not sufficiently adaptable in noisy and uncertain environments. Using learning-based algorithms is therefore essential. Resource allocation problem can be considered as an online decision-making system where fog nodes should decide whether processing locally the receiving requests from IoT devices or sending them to distant cloud nodes. We model the fog resource allocation problem as a Markov decision process and solve it by the Deep reinforcement learning approach. Based on policy-gradient algorithm, fog nodes learn how to schedule the IoT tasks in an optimal way .

The proposed method is compared with the non-learning approach in which tasks are assigned to fog nodes based on their length and without the consideration of task priority. The obtained results, according to the cumulative reward during the implementation process of the proposed algorithm, indicate that the resource allocation policy has been learned online. This improves the average slowdown and average slowdown in difficult conditions for a system with different task priorities, when compared to the non-learning method.

Keywords: Fog Computing, Task Scheduling, Internet of Things, Deep Reinforcement Learning, Actor-critic

7th International Conference on Web Research
May 19-20, 2021; Tehran, IRAN ICWR2021

بهینه‌سازی تخصیص منابع مه در اینترنت اشیا با رویکرد یادگیری تقویتی عمیق

دانوش چمنی[۱]، پارسا وفایی[۱]، زهرا موحدی [۲،*]

[۱] دانش‌آموخته‌ی کارشناسی مهندسی کامپیوتر-هوش مصنوعی،
گروه مهندسی کامپیوتر، دانشکده‌ی مهندسی پردیس فارابی، دانشگاه تهران
{danoosh.chamani, parsa.vafaei}@ut.ac.ir

[۲] استادیار، گروه مهندسی کامپیوتر، دانشکده‌ی مهندسی پردیس فارابی، دانشگاه تهران
{zmovahedi}@ut.ac.ir

چکیده

محاسبات مه یک پارادایم نوظهور است که مفهوم ابر را تا لبه گسترش می دهد. این پارادایم، منابع محاسباتی، ذخیره سازی، کنترل و قابلیت‌های شبکه را برای تحقق برنامه‌های کاربردی اینترنت اشیا فراهم می‌کند. در مفهوم محاسبات مه، دستگاه‌های اینترنت اشیا، داده ها و محاسبات پیچیده را به گره‌های مه در اطرافشان بارگذاری می‌کنند. در این مقاله، به مسئله‌ی تخصیص بهینه منابع محدود گره‌های مه به برنامه‌های اینترنت اشیا می‌پردازیم. در واقع، مسئله‌ی تخصیص منابع را می توان به صورت یک سیستم تصمیم‌گیری آنلاین در نظر گرفت که در آن گره‌های مه باید تصمیم بگیرند که آیا درخواست‌های دریافتی از دستگاه‌های اینترنت اشیا را به صورت محلی پردازش کنند یا آن‌ها را به گره‌های ابر در فواصل دور فرستند. رویکردهای کنونی برای تخصیص منابع مه از انطباق پذیری کافی در محیط‌های دارای نویز و عدم قطعیت برخوردار نیستند. به همین منظور، وجود الگوریتم‌های متکی به یادگیری در این حوزه امری ضروری است. در این مقاله، در مرحله‌ی اول مسئله تخصیص منابع مه به عنوان یک فرایند تصمیم گیری مارکوف مدل‌سازی شده است. سپس روشی بر پایه‌ی رویکرد یادگیری تقویتی عمیق جهت حل این مسئله پیشنهاد شده است. در واقع، بر اساس الگوریتم گرادیان سیاست، گره‌های مه یاد می‌گیرند که چگونه وظایف IoT را به روشی بهینه برنامه‌ریزی کنند.

روش پیشنهادی با رویکرد غیر یادگیری مقایسه شده است که در آن وظایف بر اساس طول اجرایشان و بدون در نظر گرفتن اولویت وظایف، به گره‌های مه تخصیص داده می‌شوند. نتایج به‌دست‌آمده با توجه به پاداش تجمعی در طول فرایند اجرای الگوریتم پیشنهادی، حاکی از یادگیری سیاست تخصیص منابع به‌صورت برخط است. این امر منجر به بهبود معیارهای میانگین تاخیر و میانگین تاخیر در شرایط سخت برای سیستمی با اولویت‌های مختلف وظایف، در مقایسه با روش غیر یادگیری می‌شود.

کلمات کلیدی

محاسبات مه، تخصیص منابع، اینترنت اشیا، یادگیری تقویتی عمیق، عملگر-نقاد

۱- مقدمه

اینترنت اشیا پدیده ایست که به طور فزاینده ای در برنامه‌های کاربردی روزمره زندگی انسان مانند اتوماسیون خانگی، نظارت بر ترافیک، مراقبت‌های بهداشتی، نظارت بر محیط زیست و غیره حضور دارد. اشیا اشاره به دستگاه‌های فیزیکی با توان محاسباتی و حافظه‌ی کم دارد که به‌صورت مدام می‌بایست وظایف (task) محاسباتی را جهت انجام برنامه‌های اینترنت اشیا اجرا کنند. از آن‌جایی‌که، این برنامه‌ها، اغلب نیازمند محاسبات پیچیده در مدت زمان کم هستند، اجرای آن‌ها توسط اشیا امکان‌پذیر نمی‌باشد. همچنین انتقال داده‌ها و وظایف به سرورهای مرکزی ابر با توجه به افزایش چشمگیر تعداد اشیا و داده‌های آن‌ها، موجب افزایش شدت مصرف انرژی، تاخیر و تضعیف عملکرد شبکه می‌شود.

محاسبات مه در ادامه‌ی محاسبات ابر به عنوان مدلی که در آن داده‌ها و پردازش‌ها در تجهیزات توزیع شده در لبه‌ی شبکه (network edge) متمرکز می‌شوند، نه کاملا در ابر، به‌وجود آمده است[2] ,[1]. با توجه به ویژگی های مه (نزدیکی آن به کاربر، تراکم توزیع جغرافیایی، پویایی و ذخیره‌سازی و محاسبات ارزان قیمت)، استفاده از منابع مه، قابلیت دسترسی سریع و تاخیر کم در پاسخگویی به نیازهای کاربران برنامه‌های اینترنت اشیا را فراهم می‌کند و ترافیک شبکه و همچنین انرژی مصرفی به‌طرز چشمگیری کاهش می‌یابد. بر این اساس، تعدادی گره مه با پارامترهای خاص (پردازنده، حافظه، ...) در محیط توزیع شده‌اند. هر یک از این گره‌های مه، می‌توانند تعدادی وظیفه/کار (task) از طرف اشیا را با توجه به ظرفیت منابع موجودشان انجام دهند. در این صورت، یک برنامه‌ی بیدرنگ اینترنت اشیا می‌تواند با تاخیر کم و میزان مصرف انرژی پایین، اجرا شود. کیفیت اجرای این برنامه‌ها بستگی به انتخاب مه‌های مورد استفاده جهت تخلیه بار محاسباتی اشیا دارد. به‌علاوه، وظایف اشیا دارای اولویت‌های متفاوت است که در تخصیص آن‌ها به گره‌های مه بسیار مهم است.

در همین راستا، تخصیص بهینه‌ی منابع گره‌های مه یکی از چالش‌های مطرح‌شده در رایانش مه است که در سال‌های اخیر مورد توجه بسیاری از محققان و صنعت‌گران در حوزه‌های مختلف قرار گرفته است [4] ,[3]. از یک طرف، گره‌های مه برخلاف گره‌های ابر، از توان محاسباتی پایین‌تری برخوردار هستند و از طرفی دیگر، تعداد تولید وظایف توسط اشیا که نیاز به تخصیص دارند، بسیار زیاد است. بنابراین نیاز به یک الگوریتم زمانبندی جهت تخصیص بهینه‌ی منابع مه داریم به‌طوری‌که میزان تاخیر کاهش یابد. با توجه به ماهیت ان–پی سخت مسائل زمانبندی، استفاده از روش‌های بهینه‌سازی مرسوم [6] ,[5] جهت حل نزدیک به بهینه‌ی مسئله در زمان چندجمله‌ای بسیار پرکاربرد بوده است.

چنین روش‌هایی، بدون در نظر گرفتن عدم قطعیت در محیط‌های اینترنت اشیا و مه به ارائه‌ی راهکار پرداخته‌اند. برای مثال، اطلاعاتی چون میزان منابع مورد نیاز توسط اشیا قبل از رسیدن درخواست تخصیص، نامشخص است. بنابراین، نیاز به روشی است که بتوان از فیدبک‌های مرتبط با تصمیم‌گیری‌های گذشته در زمان تخصیص منابع درس گرفت تا بتوان فرایند تصمیم‌گیری آنلاین را به سمت درست هدایت کرد.

یادگیری تقویتی عمیق، یکی از روش‌های هوش مصنوعی است که اخیرا به‌طور گسترده مورد توجه قرار گرفته است. مزیت اصلی این رویکرد، عدم نیاز به اطلاعات قبلی در رابطه با سیستم و تطبیق با اهداف متنوع است. در همین راستا، راهکارهایی جهت حل زمانبندی آنلاین وظایف بر پایه‌ی یادگیری تقویتی عمیق پیشنهاد شده است [11]–[7]. برای مثال، مرجع[7] روشی را بر پایه‌ی استفاده از گرادیان سیاست جهت یادگیری کنش‌های مناسب در طول زمان ارائه کرده است به‌طوری‌که دو فاکتور میزان تاخیر و عدالت در استفاده از منابع حفظ شود.

روش‌های موجود، علی‌رغم بهبودهای مختلف عملکردی، به‌طور کلی اولویت وظایف اشیا را به‌عنوان یکی از پارامترهای مهم در زمانبندی نادیده گرفتند. این امر به کاهش تمایل کاربران برنامه‌های کاربردی بیدرنگ در اینترنت اشیا به استفاده از سیستم مبتنی بر مه می‌انجامد.

زمانبندی آنلاین وظایف با توجه به اولویت وظایف، هنوز به‌عنوان یک چالش مهم باقی مانده است. به‌علاوه، یک زمانبندی درست باید هر دو فاکتور اولویت و تاخیر را به‌طور همزمان مورد توجه قرار دهد تا اولویت برخی وظایف موجب تاخیر بیش از اندازه در اجرای سایر وظایف با اولویت کمتر نشود.

در این مقاله، برای مقابله با چالش‌های فوق، به طراحی یک الگوریتم زمانبندی آنلاین وظایف برای سیستم‌های رایانش مه، با تکیه بر اولویت وظایف و دستیابی به تأخیر بسیار کم می‌پردازیم. در زیر نوآوری و نتایج کلیدی این مقاله آورده شده است:

- یک مدل سیستم و فرمول‌سازی را با توجه به اهداف اولویت و تاخیر پیشنهاد می‌کنیم.
- با استفاده از رویکرد یادگیری تقویتی عمیق، الگوریتم بهینه‌ای را برای زمانبندی آنلاین وظایف پیشنهاد می‌دهیم به‌گونه‌ای که با یادگیری از تصمیمات گذشته در تخصیص منابع، هر دو هدف تاخیر و اولویت وظایف را در نظر بگیرد.

ادامه‌ی مقاله به شرح زیر است: بخش دوم، به شرح کارهای پیشین می‌پردازد. مدل سیستم و فرمولسازی در بخش سوم توصیف شده است. بخش چهارم، الگوریتم زمانبندی آنلاین وظایف را شرح می‌دهد. نتایج شبیه‌سازی در بخش پنجم آمده است و بخش ششم نتیجه‌گیری و کارهای آینده را بیان می‌کند.

۲- کارهای پیشین

در سال های اخیر، مسئله ی تخصیص منابع در رایانش ابری توجه ویژه‌ای را سمت خود جلب کرده است. با ظهور رایانش مه، این مسئله اهمیت بیشتری پیدا کرده و در نهایت موجب شده است که تحقیقات گسترده ای بر روی آن صورت بگیرد. در این بخش، مقالاتی که به مدیریت منابع موجود در مه جهت بارسپاری وظایف پرداخته‌اند را بررسی می‌کنیم. مطالعات مجملی در مقالات [11]–[8] بر روی مسئله‌ی تخصیص منابع در محیط ابر و مه صورت گرفته است. در ادامه، تحقیقاتی که در زمینه‌ی مدیریت منابع مه با استفاده از رویکرد یادگیری عمیق است را بررسی خواهیم کرد.

در مقاله‌ی [12]، مولف یک الگوریتم یادگیری تقویتی عمیق را برای مسئله‌ی تخصیص منابع استفاده می‌کند. هر چند این مقاله مرتبط به تخصیص منابع درون یک سیستم کامپیوتری، و نه محیط اینترنت اشیا است، اما دیدگاه مناسبی نسبت به مسئله‌ی تخصیص منابع در آن دیده می‌شود. در این مقاله، از یک عامل هوشمند یادگیری تقویتی عمیق برای تخصیص منابع به وظایف درون یک کامپیوتر با منابع مختلف، استفاده شده است. هر وظیفه درون یک سیستم کامپیوتر به منابع مختلف و متعددی از جمله واحد پردازنده

مرکزی، حافظه، ورودی و خروجی استاندارد نیازمند است. حال هدف این عامل هوشمند این است که با توجه به وضعیت کنونی سیستم و کارهای در حال انتظار، به گونه ای منابع را به کارهای مختلف اختصاص دهد که تاخیر کلی سیستم به حداقل برسد. تاخیر در این مقاله به صورت مدت زمان بین ورود کار به سیستم تا زمان اتمام انجام آن، تعریف شده است. به همین منظور، تابع هدف عامل به گونه ای طراحی شده که یک حالت کمینه ی محلی برای تاخیر پیدا کند. مولف در این مقاله حالت کنونی سیستم را بصورت یک ماتریس دو بعدی به عامل هوشمند تحویل می دهد. این ماتریس دو بعدی حاوی اطلاعاتی از جمله کارهای در حال انجام، مدت زمان باقی مانده برای هر کار در حال انجام کارهای در حال انتظار و مدت زمان و منبع مورد نیاز برای هر کار است. نتیجه‌ی نهایی الگوریتم نشان‌دهنده‌ی عملکرد بهتر این روش نسبت به الگوریتم های Tetris، SJF و Packer می‌باشد. این رویکرد به عنوان سنگ بنای روش پیشنهادی در این مقاله برای تخصیص منابع مه در نظر گرفته شده است.

در مقاله‌ی [13]، مولف مدلی ارائه می‌دهد که در آن هر گره لبه به‌عنوان یک عامل در نظر گرفته شده است. هر گره در هر لحظه تصمیم می‌گیرد که آیا یک کار را به دیگر گره‌های محاسباتی بفرستد یا خودش انجام دهد. هدف مدل اینست که تاخیر انجام کار و انرژی مصرفی برای انجام کارها را در بلند مدت کمینه کنند. به این منظور، در این مقاله از یک الگوریتم یادگیری تقویتی عمیق مبتنی بر روش ε-greedy Q-learning برای دستیابی به تابع هدف استفاده شده است. نتایج نهایی مدل معرفی شده یک پایاپایی بهینه بین انرژی مصرفی و تاخیر کار را پیشنهاد می‌کند.

مقاله‌ی [14] یک فرموله سازی بر مبنای فرآیند تصمیم مارکوف را برای مسئله‌ی تخصیص منابع در محیط مه‌های LAN یا سرویس های اینترنت اشیا ناهمگن، ارائه می‌دهد. در این مقاله، مولفان چندین الگوریتم یادگیری تقویتی از جمله Q-learning، SARSA و Monte-carlo را به‌منظور یادگیری سیاست بهینه در محیط اینترنت اشیا به کار می‌گیرند.

در مقاله‌ی [15]، مولف یک معماری مه مبتنی بر کنترل گر SDN را معرفی می‌کند. در این معماری یک کنترل گر مرکزی با دانش بر وضعیت کل شبکه، راجع به عملیات تخلیه‌ی بار هر یک از گره‌های مه تصمیم می‌گیرد. هدف نهایی این سیستم کمینه کردن تاخیر انجام کارها و احتمال سرباری هر گره است. در این سیستم کنترل‌گر، کنش‌های بهینه‌ای را با استفاده از الگوریتم‌های یادگیری تقویتی پیدا می‌کند که به موجب آن‌ها هر گره قادر است یک گره‌ی بهینه‌ی همسایه را پیدا کرده و کارهای خود را به آن بفرستد. به موجب این توزیع بار بر روی شبکه، تأخیر اجرایی وظایف و سرباری بر روی گره‌ها کاهش می‌یابد. هرچند به دلیل اینکه این روش متکی به یک کنترلگر مرکزی است به‌طبع از مقیاس‌پذیری بالایی برخوردار نمی‌باشد.

مقاله‌ی [16] با هدف کمینه‌سازی زمان میانگین وزن‌دار انجام کار و میانگین تعداد منابع درخواستی، به حل مسئله‌ی تخصیص منابع در رایانش لبه‌ای می‌پردازد و یک روش deep Q-network را جهت حل مسئله پیشنهاد می‌دهد.

نویسندگان در مقاله‌ی [17] مسئله‌ی تخصیص منابع محدود مه را با توجه به نیازمندی‌های مختلف برنامه‌های IoT در تاخیر، بررسی کرده‌اند و پس از مدل‌سازی مسئله به‌صورت مدل مارکوف، با استفاده از روش‌های Q-learning, SARSA, Expected SARSA, and Monte Carlo به حل مسئله پرداخته‌اند.

مشکل کلی روش‌های معرفی‌شده، عدم در نظر گرفتن اولویت وظایف اشیا در زمانبندی است. از آن‌جایی‌که برای برنامه‌های کاربردی اینترنت اشیا مانند سیستم مانیتور بیمارستان و یا آتش‌سوزی، اولویت انجام وظایف بسیار بالاست و این اولویت می‌بایست در زمانبندی نظر گرفته شود، در این مقاله، با توجه به این پارامتر، زمانبندی مبتنی بر یادگیری تقویتی عمیق را پیشنهاد می‌کنیم. مقایسه‌ی روش‌های بین شده با روش پیشنهادی، در جدول (۱)آمده است.

۳- طراحی سیستم پیشنهادی

در این بخش، مدل و فرموله‌سازی مسئله‌ی زمانبندی آنلاین وظایف در سیستم‌های رایانش مه ارائه شده است.

جدول ۱: مقایسه‌ی روش پیشنهادی با سایر روش‌ها

مقاله	محیط	اولویت کار	میانگین تاخیر کار	روش
[12]	سیستم کامپیوتری	خیر	بله	Policy Gradient
[13]	اینترنت اشیا	خیر	بله	ε-greedy Q-learning
[14]	اینترنت اشیا	خیر	بله	Q-learning، SARSA و Monte-carlo
[15]	عمومی	خیر	بله	Q-learning
[16]	اینترنت اشیا	بله	خیر	deep Q-network
[17]	اینترنت اشیا	بله	خیر	Q-learning, SARSA, Expected SARSA, and Monte Carlo
این مقاله	اینترنت اشیا	بله	بله	Policy gradient و Q-learning

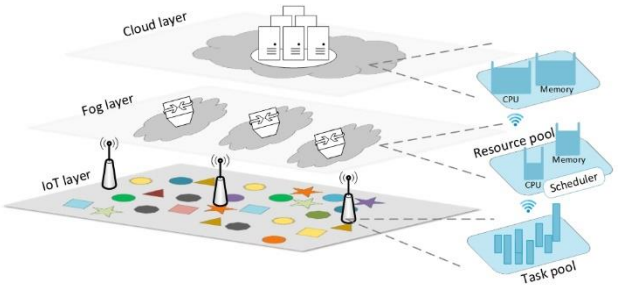
۳-۱- معماری سیستم

همانطور که در شکل (۱) نشان داده شده است، معماری سیستم پیشنهادی از سه لایه‌ی اینترنت اشیا، مه و ابر تشکیل شده است. در لایه‌ی اینترنت اشیا، تعداد زیادی گره اشیا (حسگر و عملگر) وجود دارد که به صورت مدام وظایف محاسباتی و داده تولید می‌کنند و دارای توانایی ارتباط بی‌سیم، ظرفیت و قدرت پردازش پایین هستند. همچنین در این لایه، تعدادی گره سینک وجود دارد که داده‌های گره‌های اشیا تحت پوشش خود را جهت ارسال به لایه‌های بالاتر جمع‌آوری می‌کند. در لایه‌ی مه، تعدادی سرور مه وجود دارد که وظیفه‌ی آن‌ها اجرای وظایف تخصیص داده شده‌ی اشیای لایه‌ی اینترنت اشیا است. در لایه‌ی ابر نیز با سرورها و مراکز داده‌ای کلان ابری مواجه هستیم که برای پردازش‌های پیچیده و سنگین مورد استفاده قرار می‌گیرد.

وظایف موجود در لایه‌ی اینترنت اشیا در مخزن وظایف (task pool) منتظر زمانبندی و تخصیص به منابع مه قرار دارند. در اینجا، وظایف را به صورت مجموعه‌ی $D = \{d_1, d_2, \dots, d_i, \dots, d_n\}$ نمایش می‌دهیم. هر وظیفه جهت اجرا به مقدار m_i^{cpu} واحد پردازش و m_i^{mem} واحد حافظه نیازمند است و مدت زمان ایده‌آل برای انجام کار را با T_i نشان می‌دهیم. اولویت هر وظیفه عددی بین ۰ و ۱ است (صفر کمترین و ۱ بیشترین اولویت) و با $p_i \in [0,1]$ نمایش داده می‌شود. منابع موجود (پردازنده و حافظه) در لایه‌ی مه و ابر، در مخزن منابع (resource pool) جهت پردازش وظایف تخصیص داده‌شده قرار دارند. واحد برنامه‌ریز (scheduler unit) در این لایه، زمانبندی بهینه اجرای وظایف و تخصیص آن‌ها را با توجه به اولویت وظایف انجام می‌دهد به طوری‌که میزان تاخیر حداقل شود. در اینجا، منابع را به صورت مجموعه‌ی $R = \{r_1, r_2, \dots, r_j, \dots, r_k\}$ نشان می‌دهیم. هر منبع دارای ظرفیت c_j^{mem} و c_j^{cpu} می‌باشد.

افق زمانی را به واحدهای زمانی مجزا تقسیم می‌کنیم. در هر واحد زمان، وظایف به‌صورت آنلاین وارد سیستم می‌شوند؛ به‌این معنا که هر واحد زمانی درون سیستم با وارد شدن یک کار جدید شروع می‌شود. واحد برنامه‌ریز در هر واحد زمان، تعدادی از وظایف در حال انتظار را انتخاب و بعد از زمانبندی، به آن‌ها منابع مورد نیازشان را اختصاص می‌دهد. زمانی‌که یک منبع r_j برای اختصاص وظیفه d_i انتخاب شد، با علامت a_{ij} نشان داده می‌شود که مقدار آن برابر با یک است و اگر انتخاب نشود، مقدار آن برابر با صفر خواهد بود. اولویت وظایف با هم برابر نیست؛ به‌این معنی که برخی کارهای از اولویت بالاتری نسبت به بقیه برخوردار است.

هر وظیفه d_i، دارای یک زمان ورود t_i^{arr}، یک زمان شروع t_i^{sta} و یک زمان پایان t_i^{fin} است. t_i^{sta} و t_i^{fin} تا زمان ورود به سیستم مشخص نیستند و بر اساس اولویت و اینکه چه وقتی، واحد برنامه‌ریز تصمیم به بررسی آن بر اساس میزان منابع مورد نیاز بگیرد.



شکل ۱: معماری سیستم پیشنهادی

۳-۲- فرموله‌سازی مسئله

در برنامه‌های بیدرنگ کاربردی اینترنت اشیا، کاهش تاخیر در زمان پاسخگویی سیستم یکی از مهمترین پارامترها جهت رضایت کاربران است. به‌همین دلیل، در این مقاله، هدف کمینه سازی میانگین اولویت‌دار تاخیر کار می‌باشد.

برای هر وظیفه d_i، تاخیر کار به‌صورت تاخیر از زمان ورود تا زمان پایان کار ($E = t_i^{fin} - t_i^{arr}$) تعریف می‌شود. جهت از بین بردن اثر کارهایی با نیاز کمتر به منابع، تاخیر کار را به‌وسیله‌ی مدت زمان ایده‌ال کار (T_i)، نرمال‌سازی می‌کنیم که از فرمول (۱) به‌دست می‌آید:

$$S_i = \frac{t_i^{fin} - t_i^{arr}}{T_i} \quad , where\ S_i \geq 1 \qquad (۱)$$

اولویت تاخیر کار به‌صورت فرمول (۲) تعریف می‌شود:

$$SP_i = \frac{t_i^{fin} - t_i^{arr}}{p_i * T_i} \quad , where\ SP_i \geq 1 \qquad (2)$$

در نهایت میانگین اولویت تاخیر کار را به‌صورت فرمول (۳) تعریف می‌کنیم:

$$\overline{SP_l} = \frac{1}{|D|} \sum_{d_i \in D} SP_i \qquad (۳)$$

هدف، کمینه‌سازی میانگین تاخیر کار است به‌طوری‌که اولویت کارها نیز در نظر گرفته شود. مسئله‌ی زمانبندی وظایف به صورت فرمول (۴) بیان می‌شود:

$$min \quad \overline{SP_l} = \frac{1}{|D|} \sum_{d_i \in D} \frac{t_i^{fin} - t_i^{arr}}{p_i * T_i} \qquad (۴)$$

Subject to

$$t_i^{sta} \geq t_i^{arr} \quad \forall\, d_i \in D \qquad (C1)$$

$$\sum_{d_i \in D} a_{ij}\, m_i^{cpu} \leq c_j^{cpu}, \quad \forall\, r_i \in R \qquad (C2)$$

$$\sum_{d_i \in D} a_{ij}\, m_i^{mem} \le c_j^{mem}, \qquad \forall r_i \in R \qquad (C3)$$

در اینجا، قید C1 بیان می‌کند که زمان آغاز یک وظیفه d_i بزرگتر از زمان ورود آن به سیستم است. قید C2 و C3 به ترتیب ضامن عدم عبور از ظرفیت پردازنده و ظرفیت حافظه‌ی منبع r_j برای اجرای وظایف انتخاب شده $a_{ij} = 1$جهت تخصیص می‌باشد.

٤- طراحی سیستم با استفاده از یادگیری تقویتی عمیق

محیط‌های اینترنت اشیا و مه، دارای شرایط نامشخصی هستند به‌طوری‌که اطلاعاتی مانند زمان رسیدن وظایف و میزان منابع مورد نیاز آن‌ها از قبل مشخص و در دسترس نیست. این همان عدم قطعیت و نویز محیط تلقی می‌شود. همچنین رویکردهای متداول متکی به یک کنترلگر مرکزی نیز از مقیاس‌پذیری پایینی برخوردار هستند. به همین دلایل، استفاده از الگوریتم‌های بهینه‌سازی معمول، جهت زمانبندی بهینه‌ی وظایف، ممکن نیست. با توجه به موفقیت رویکرد یادگیری تقویتی عمیق در حل مسائل مدیریت منابع در سال‌های اخیر، و همچنین توانایی تطابق با محیط‌های دارای عدم‌قطعیت، در این مقاله یک روش جدید غیر متمرکز و مبتنی بر یادگیری تقویتی عمیق جهت زمانبندی آنلاین وظایف با در نظر گرفتن اولویت آن‌ها ارائه شده است به‌طوری‌که میانگین تاخیر اجرای وظایف حداقل شود.

٤-١- پیش‌زمینه‌ی یادگیری تقویتی عمیق

همانطور که در [18] به تفصیل معرفی شده است، یادگیری تقویتی عمیق، یک روش بر مبنای یادگیری ماشین است. در هر واحد زمانی، عامل حالت S_t از محیط را مشاهده می‌کند و کنش a_t را بر روی آن انجام می‌دهد. در نتیجه‌ی این کنش بر روی محیط، حالت کنونی محیط به S_{t+1} تغییر می‌یابد و عامل پاداش r_t را دریافت می‌کند. فرض می‌شود که این تغییر حالت و پاداش دریافت شده از خصوصیت مارکوف تبعیت می‌کند؛ به این معنی که احتمالات تغییر یک حالت به حالت دیگر و همچنین پاداش دریافت شده تماما به حالت کنونی و کنش انجام شده توسط عامل بستگی دارد، پس تمامی عوامل تاثیرگذار در محیط توسط عامل مشاهده‌پذیر هست. لازم به ذکر است که عامل هیچ دانشی از قبل راجع به پاداش ها و تغییر حالت‌های محیط ندارد اما با انجام کنش بر روی محیط و مشاهده نتایج آن‌ها و با یادگیری قادر است این مقادیر را یاد بگیرد. هدف یادگیری این عامل بیشینه‌کردن مقدار مجموع پاداش‌های تخفیف یافته است $E[\sum_{t=0}^{\infty} \gamma^t\, r_t], \gamma \in (0,1]$. عامل هر کنش را بر اساس یک سیاست انتخاب می‌کند. سیاست در واقع یک توزیع احتمالاتی روی مجموعه‌ی کنش‌هاست که به‌صورت $\pi: \pi(s,a) \rightarrow$ تعریف می شود. $\pi(s,a)$ احتمال انتخاب کنش a در حالت s است. در مسائل دنیای واقعی تعداد این جفت (s,a) بسیار زیاد است، بنابراین ذخیره‌کردن این جفت‌ها درون جداول غیر ممکن است. برای حل این مشکل از توابع تقریب‌زن استفاده می‌شود. یک تابع تقریب‌زن تعدادی پارامتر (Θ) قابل تغییر دارد که این تعداد نسبت به حالت جدولی بسیار کمتر و در نتیجه قابل مدیریت است. به این پارامترها، پارامترهای سیاست نیز گفته و سیاست بر اساس این پارامترها بصورت $\pi_\theta(s,a)$ بازنویسی می‌شود. توابع تقریب‌زن بسیاری وجود دارند که می‌توان از آن‌ها استفاده کرد اما استفاده موفق از شبکه‌های عصبی عمیق به عنوان تقریب‌زن در سال‌های اخیر نشان داده است که آن‌ها بهترین انتخاب برای حل مسائل پیچیده و بزرگ هستند. به همین دلیل در این مقاله از روشی مبتنی بر یادگیری تقویتی عمیق استفاده شده است. مدل استفاده شده برای عامل یادگیری تقویتی عمیق، مدل PP0(Proximity Policy Optimization) مبتنی بر مدل Actor Critic است. PPO اجازه‌ی کنترل سیاست‌های هر حالت را نسبت به حالت قبلی می‌دهد از این رو می‌توان مطمئن بود که میزان سیاست تغییر زیادی نداشته باشد. مدل‌های Actor Critic تلفیقی از دو مدل Policy gradient و Q-learning هستند. در ادامه، ابتدا به بازنویسی سیستم بر اساس یادگیری تقویتی عمیق پرداخته و سپس مدل PP0 را جهت حل مسئله به‌کار می‌گیریم.

٤-٢- بازنمایی محیط

شاید بتوان گفت بدون وجود یک محیط خوب و کامل، دست یافتن به نتایج درست با الگوریتم های یادگیری تقویتی امکان ناپذیر است. این محیط باید اطلاعات کامل و گویایی را در اختیار عامل گذاشته تا عامل بر اساس آن‌ها یادگیری درستی داشته باشد. در این مقاله، محیط، سیستم رایانش مه است و واحد برنامه‌ریز، عامل است.

بازنمایی حالت، پاداش و مدل گذار سه مشخصه‌ی اصلی یک محیط هستند که باید به دقت کافی طراحی شوند، در غیر این صورت عامل یادگیری تقویتی الگوهای غلط را یاد گرفته و به نتیجه‌ی مطلوبی نمی‌رسد. در ادامه به معرفی سه مشخصه ی اصلی در محیط سیستم رایانش مه می‌پردازیم. شکل(۲) نمای کلی سیستم مبتنی بر یادگیری تقویتی عمیق را نشان می‌دهد. در این شکل، وظایف بصورت برخط و در واحد زمانی های جداگانه به سیستم وارد شده و عامل یادگیری تقویتی عمیق با استفاده از دانشی که از محیط دارد و پاداشی که دریافت می‌کند، اقدام به تخصیص منبع به این وظایف می‌کند.

٤-٢-١- بازنمایی حالت

بازنمایی حالت مجموعه ای از اطلاعات محیط است که به‌صورت تانسوری از اعداد در اختیار عامل یادگیری تقویتی قرار گرفته و عامل به وسیله آن کنش مناسب را انتخاب می‌کند. بازنمایی حالت باید شامل تمامی اطلاعاتی باشد که عامل برای تصمیم‌گیری به آن‌ها نیاز دارد. در مقاله‌ی [12]، مولفان از یک بازنمایی حالت به شکل یک تصویر یک کاناله استفاده کرداند. برای هر حالت از محیط، یک ماتریس دو بعدی شامل اطلاعاتی از وظایف در حال

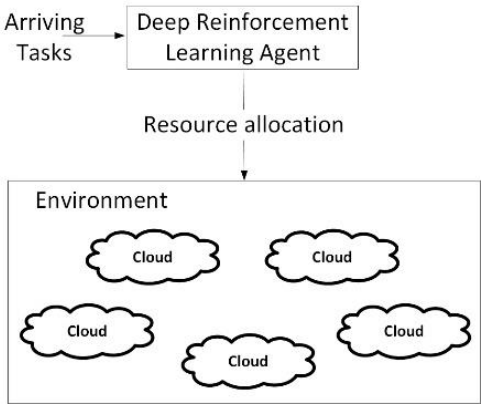
انجام،وظایف در حال انتظار، منابع و نرخ ورود وظیفه‌ی جدید، ساخته می‌شود. در این مقاله، علاوه بر اطلاعات ذکر شده در هر حالت از محیط، وظایف در صف پشتی (صفی که هر وظیفه قبل از ورود به صف انتظار در آنجا سپری می‌کند) اولویت کارها نیز در نظر گرفته می‌شود. شکل ۳شکل (۳)، دید دقیق‌تری نسبت به این بازنمایی می دهد. همانطور که مشخص است، این بازنمایی تمامی اطلاعاتی که عامل برای تصمیم گیری نیاز دارد را در اختیارش می‌گذارد. در این تصویر، تعداد m منبع وجود دارد که هر کدام از منابع دارای ظرفیت خاص می‌باشد و به‌صورت مجموعه‌ای از خانه‌ها مشخص شده است. همچنین تعداد n وظیفه در صف انتظار وجود دارد. این وظایف دارای اولویت‌های متفاوت هستند. مثلا رنگ قرمز نشانگر اولویت بالا و رنگ سبز نشانگر اولویت کمتر است. هر کدام از وظایف، به منابعی نیاز دارند که تعداد آن به‌صورت خانه‌هایی مشخص شده است. در این تصویر، نوع منبع (پردازنده یا حافظه) تفکیک نشده است؛ به‌این معنا که سیستم پیشنهادی به‌صورت کلی و با هر نوع منبع، قابل شبیه‌سازی است. همچنین حالت آغازین در سیستم همیشه ماتریسی تماما صفر است چرا که در ابتدای امر هیچ وظیفه‌ای به سیستم وارد نشده است.

٤-۲-۲- بازنمایی پاداش

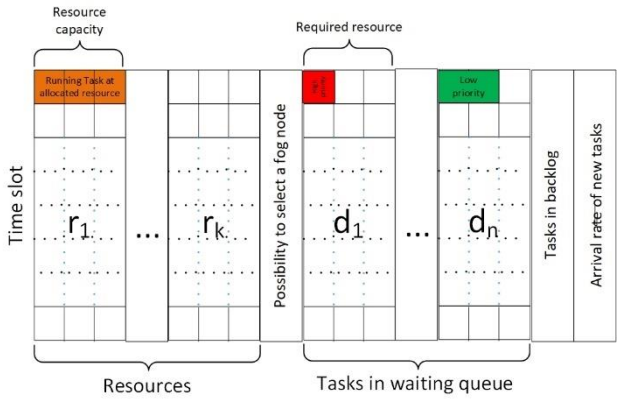
تابع پاداش در یادگیری تقویتی، پاداشی است که به عامل داده می‌شود و نقش اساسی در هدایت و نحوه‌ی یادگیری عامل دارد. این پاداش باید رابطه‌ی مستقیم با هدف نهایی سیستم داشته باشد تا عامل بتواند آن را برآورده کند؛ به‌همین جهت، ما دقیقا هدف سیستم را به‌عنوان پاداش به عامل می‌دهیم. همانطور که پیش تر دیدیم، هدف سیستم کمینه‌کردن مقدار میانگین اولویت‌دار تاخیر کار (معادله‌ی ۴) است. پس هدف عامل برابر است با بیشینه‌سازی پاداش تجمعی کمینه‌ی میانگین اولویت‌دار تاخیر کار در هر واحد زمان (هر واحد از E را یک واحد زمانی در نظر گرفته‌ایم). همچنین به علت اینکه مفهوم اولویت، زمانی که کار در حال انجام است بی معنی می‌شود، به ازای هر کار در حال انجام درون سیستم، پاداشی را به آن اختصاص می دهیم. بنابراین، پاداش نهایی که پس از هر کنش به سمت عامل یادگیری تقویتی باز می‌گردد به صورت فرمول (۵) است:

$$rew_{ag} = -\sum_{d_i \in D_t} \frac{1}{p_i * T_i} + \sum_{d_i \in D_r} \frac{1}{T_i} \qquad (5)$$

D_t وظایفی است که توسط واحد برنامه‌ریز جهت تخصیص به منابع آغاز شده ولی هنوز به اتمام نرسیده‌اند و D_r وظایف در حال اجرا را نشان می‌دهد.



شکل ۲: بازنمایی سیستم با توجه به رویکرد یادگیری تقویتی



شکل ۳: بازنمایی حالت

٤-۳- بازنمایی مدل گذار

مدل گذار بیان می‌کند که اگر در حالت s از محیط، کنش a انجام شود، حالت محیط به چه حالت جدیدی گذار خواهد کرد. این مفهوم از آن جایی که سنگ بنای کنش‌های عامل است، از اهمیت بسیار بالایی برخوردار است. همچنین شرط لازم برای کارکرد صحیح عامل‌های یادگیری تقویتی اینست که این مدل گذار برای عامل و توسط بازنمایی حالتی که از محیط دریافت می‌کند، کاملا قابل شناسایی باشد. به این شرط لازم، تبعیت کردن محیط از خصیصه‌ی مارکوف می‌گویند. با توجه به اینکه تمامی اطلاعات تاثیرگذار در مدل گذار درون بازنمایی حالت محیط جای داده شده است، این شرط لازم به خودی خود برآورده می‌شود. حال برای آشنایی بیشتر با تاثیر کنش ها بر روی محیط و مدل گذاری به تشریح کلی کنش های قابل انجام عامل می‌پردازیم.

اولین نکته‌ی قابل توجه اینست که عامل در هر واحد زمانی می‌تواند به تعداد نامحدود کنش انجام دهد. از آن‌جایی که شروع هر واحد زمانی با وارد شدن یک کار جدید به سیستم همراه است. عامل می‌تواند قبل از ورود کار جدید به هر تعداد کار در حال انتظار، در صورت امکان، منبع تخصیص دهد.

تنها زمانی یک واحد زمانی به اتمام می‌رسد که عامل صراحتا اعلام به اتمام واحد زمانی کند و یا یک کنش غیر مجاز انجام شود. برای مثال، اقدام به تخصیص منبعی کند که ظرفیت کافی برای انجام کار انتخاب شده را نداشته باشد. ذکر این نکته خالی از لطف نیست که عامل تنها می‌تواند منابع را به کارهای در صف انتظار (تعداد Q)، و نه صف پشتی، اختصاص دهد. از آنجایی که تعدا کارهای درون سیستم می تواند بسیار بزرگ شود، اگر قرار بود به ازای هر کار، یک کنش برای عامل قرار داده شود، فضای کنش‌ها بسیار بزرگ و پیچیده می‌شد؛ بنابراین تعدادی از کارها در صف پشتی و تعدادی در صف انتظار جای‌گذاری می‌شوند. در هنگامی خالی شدن یکی از خانه‌های صف انتظار، یکی از کارهای صف پشتی به این صف انتقال می‌یابد. کنش‌های عامل بطور کلی به سه دسته تقسیم می شود:

- انتخاب یک کار در حال انتظار و تخصیص منابع مورد نیاز به آن
- فعال کردن یک مه یا حالت مذاکره
- کنش خالی

کنش‌های دسته‌ی اول کنش‌های بین صفر تا q هستند که متغیر q تعداد خانه‌های صف انتظار است. انتخاب این کنش‌ها به منزله‌ی انتخاب کار در حال انتظار در خانه q_l به‌منظور اختصاص منابع به آن است. کنش‌های دسته دوم برای انتخاب و به عبارت دیگر فعال کردن یک مه است. این کنش ها از شماره ۱ تا q+x را شامل می شوند که متغیر x نمایانگر تعداد مه‌های تحت کنترل عامل است. فرض کنید عامل قصد دارد کار در حال انتظار در خانه‌ی صفر را به مه شماره یک اختصاص دهد. برای این کار، عامل ابتدا باید از طریق کنش‌های دسته‌ی دوم. مه شماره‌ی یک را انتخاب کرده و در کنش بعدی خود، کنش انتخاب کار در خانه‌ی صفرم را انجام دهد. دسته‌ی سوم در واقع متشکل از یک کنش است. این کنش همان اعلام با صراحت عامل برای اتمام یک واحد زمانی است. زمانی که عامل دیگر قصد و یا امکان تخصیص منبع نداشته باشد، می‌تواند از این کنش استفاده کند. این کنش شماره‌ی آخر در مجموعه‌ی کنش‌هاست.

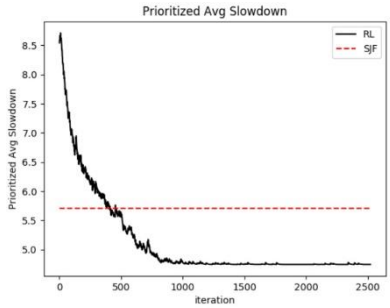
۵- نتایج تجربی

در این مقاله، از زبان پایتون و کتابخانه‌های Keras, Numpy و Matplotlib جهت پیاده‌سازی و نمایش نتایج سیستم پیشنهادی برای زمانبندی تخصیص منابع مه بر اساس اولویت وظایف با به‌کار بردن یادگیری تقویتی عمیق استفاده شده است. در سیستم پیشنهادی، اولویت عددی بین ۰ و ۱ در نظر گرفته شده است و تعداد تکرارها بین ۲۵۰۰ تا ۴۰۰۰ (با توجه به همگرایی) انتخاب شده‌است. جهت جلوگیری از کنش‌های اشتباه و غیر مجاز در حالات مختلف سیستم، از روش alphaGo Zero که در آن احتمال کنش‌های غیر مجاز صفر می‌شود، استفاده شده است.

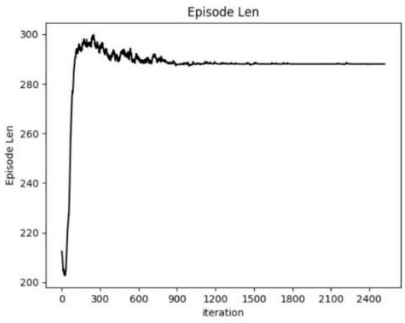
معیار ارزیابی در سیستم پیشنهادی، مقادیر میانگین اولویت دار تاخیر کارها در تکرارهای مختلف الگوریتم‌ها و اطمینان از سیر نزولی مقادیر آن‌هاست. همچنین مشاهده ی سیر طول هر مسیر نیز می‌تواند اطلاعات مفیدی راجع به سرعت تصمیم‌گیری عامل‌ها در امر تخصیص منابع بدهد. و در نهایت، مقایسه‌ی پاداش‌های تجمعی تکرارهای مختلف برنامه می‌تواند بر درست بودن معیارهای ارزیابی صحه بگذارد. این معیارها به تفصیل و در کنار نمودارهایی که آنها را نمایش می‌دهند، بررسی خواهد شد. در ارزیابی میانگین اولویت دار تاخیر کار، که کمینه کردن آن هدف اصلی سیستم است، عملکرد سیستم را با الگوریتمSJF (Shortest Job First) مقایسه می‌کنیم. بنابراین، الگوریتم SJF که یک الگوریتم نسبتا خوب برای مقاصد تخصیص منابع و زمان‌بندی است، به عنوان سنگ محک سیستم پیشنهادی استفاده می‌شود.

شکل (۴) نمایانگر میانگین اولویت‌دار تاخیر کار است. در این نمودار مشاهده می‌شود که عامل یادگیری تقویتی عمیق توانسته بعد از حدود تنها ۵۰۰ تکرار یادگیری به عملکرد الگوریتم SJF در تخصیص منابع، برسد. همچنین مشاهده می‌شود که این عامل پس از حدود ۱۱۰۰ تکرار به حدود ۱ واحد میانگین اولویت‌دار تاخیر کار کمتر از الگوریتم SJF رسیده است، اما پس از آن رشد خاصی حتی تا ۱۰۰۰ تکرار بعدتر هم نداشته است.

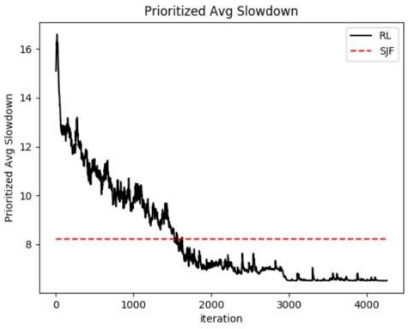
شکل(۵) بیانگر میانگین طول مسیرها در تکرارهای مختلف است. با مشاهده‌ی این نمودار و مقایسه‌ی آن با نمودار شکل ۴ می‌توان دریافت که در ابتدای کار عامل به تمامی کارها به سرعت و بصورت غیر بهینه منبع اختصاص می‌دهد ولی با گذشت زمان و آموزش دیدن سعی می‌کند تخصیص منابع را با تامل بیشتری انجام دهد و همین باعث زیاد شدن میانگین طول میسرها شده است. همچنین یک نزول در میانگین طول مسیر از تکرار ۳۰۰ تا حدود ۱۰۰۰ مشاهده می‌شود که نشان می‌دهد عامل راهی برای کم کردن طول مسیرها و همچنین افزایش بهره وری یافته است؛ به عبارت دیگر، میانبرهای مسیر را یافته است. نمودارهایشکل ۴: میانگین اولویت‌دار تاخیر کار وشکل ۵ متعلق به عاملی است که بر روی یک محیط با ٪۷۰ احتمال ورود کار جدید در هر واحد زمانی و همچنین ٪۶۰ احتمال کوچک بودن کار، آزمایش شده است. اما نمودار شکل(۶) که بیانگر میانگین اولویت دار تاخیر کار است، متعلق به عاملی است که بر روی یک محیط با ٪۸۰ احتمال ورود کار جدید در هر واحد زمانی و و همچنین ٪۵۰ احتمال کوچک بودن کار، آزمایش شده است. دلیل آزمایش بر روی دو نوع محیط این بود که میزان موفقیت عامل را در شرایط سخت و سرباری زیاد نیز آزمایش کرده و نتایج را مقایسه کنیم. این نمودار نشان می‌دهد که عامل توانسته پس از ۱۵۰۰ تکرار به عملکرد SJF در تخصیص منابع برسد. همچنین عامل پس از حدود ۴۳۰۰ تکرار توانسته به حدود ۲ واحد میانگین اولویت دار تاخیر کار کمتر از الگوریتم SJF برسد و از شمایل نمودار هم به نظر می‌رسد همچنان قابلیت یادگیری داشته باشد. با مقایسه این نمودار با نمودار شکل (۴) به این نتیجه می‌رسیم که هرچه محیط دارای شرایط سخت‌تر باشد عملکرد بهتر الگوریتم یادگیری تقویتی عمیق با الگوریتم‌هایی مثل SJF بیشتر نمایان می‌شود. همچنین جهش از حدود ۱۵ واحد میانگین اولویت دار تاخیر کار به حدود ۶ واحد میانگین اولویت دار تاخیر کار و مقایسه با جهش نمودار شکل (۴) که جهشی



شکل ٤: میانگین اولویت‌دار تاخیر کار



شکل ٥: میانگین طول مسیر



شکل ٦: میانگین اولویت‌دار تاخیر کار در شرایط سخت

۴ واحدی بوده است می‌توان نتیجه گرفت که عملکرد این الگوریتم و پاسخ نهایی، نزدیک به بهینه‌ی سراسری بوده است.

٦- نتیجه‌گیری

تخصیص منابع با ظهور و گسترش رایانش مه تبدیل به مسئله‌ی مهمی شده است؛ به همین دلیل وجود الگوریتم هایی که بتوانند منابع محدود مه را به بهترین نحو اختصاص دهند و یا بار اضافی یک مه را بر روی دیگر مه‌ها توزیع کنند، نیاز اساسی برای پیشرفت این حوزه است. پیشرفت‌های روزافزون یادگیری تقویتی عمیق و دستاوردهای بزرگ این رویکرد، ما را به استفاده از این رویکرد در حل مسئله‌ی تخصیص منابع در محیط‌های مه با توجه به ماهیت عدم قطعیت در چنین محیط‌هایی ترغیب کرد. بدین ترتیب، عامل برنامه‌ریز، با توجه به آنچه از محیط اطراف خود بر اساس کنش‌های انجام داده در طول زمان یاد می‌گیرد، می‌تواند به حل بهینه‌ی مسئله‌ی زمانبندی با توجه به اولویت کارهای اشیا بپردازد. مقایسه با روش SJF، کارا بودن راهکار پیشنهادی را تایید می‌کند. این مقایسه نشان داد که در محیطی که وظایف بصورت برخط به سیستم وارد می‌شوند و سیستم دانش پیشینی از وظایف ندارد، استفاده از رویکرد پیشنهادی تاثیر مثبتی بر روی عملکرد سیستم دارد.

برای کارهای آینده در نظر داریم تا از راهکار سیستم‌های چند عاملی در یک معماری توزیع‌شده، علاوه بر عامل یادگیری تقویتی، عامل دیگری نیز به‌عنوان عامل مذاکره داشته باشیم تا با استفاده از تعامل و مذاکره‌ی بین عامل‌ها بتوانیم عملکرد سیستم بهینه‌ساز را بهبود بخشیم. در واقع عامل مذاکره، در زمانی‌که عامل یادگیری تقویتی، قادر به تخصیص منابع در مه‌های تحت پوشش خود نیست، می‌تواند به بررسی تخصیص منابع بر روی نواحی تحت پوشش دیگر عامل‌های یادگیری تقویتی بپردازد تا میانگین تاخیر حداقل شود و مقیاس پذیری سیستم در صورت وجود وظایف زیاد افزایش یابد. همچنین استفاده و لحاظ کردن مفاهیمی چون کیفت خدمت (QoS) و به کارگیری آن‌ها، به سیستم جامعیت بیشتری می دهد. به‌علاوه، در محیط‌های مه. هر زمان یک مه قادر به انجام وظایف نباشد و هیچ مه دیگری نیز نباشد که بتواند آن کار را انجام دهد، آن وظیفه به سمت ابر، که از قدرت و ظرفیت محاسباتی بالاتری برخوردار است، فرستاده می شود. در حال حاضر چنین امکانی در سیستم پیشنهادی ما وجود ندارد. لذا اضافه کردن این قابلیت می‌تواند به افزایش جامعیت و همچنین بهره‌وری سیستم کمک شایانی کند.

مراجع

[1] S. Yi, C. Li, and Q. Li, ‘A Survey of Fog Computing: Concepts, Applications and Issues’, in *Proceedings of the 2015 Workshop on Mobile Big Data*, Hangzhou China, Jun. 2015, pp. 37–42, doi: 10.1145/2757384.2757397.

[2] H. Atlam, R. Walters, and G. Wills, ‘Fog Computing and the Internet of Things: A Review’, *Big Data Cogn. Comput.*, vol. 2, no. 2, p. 10, Apr. 2018, doi: 10.3390/bdcc2020010.

[3] D. Rahbari and M. Nickray, ‘Computation Offloading and Scheduling in Edge-Fog Cloud Computing’, *J. Electron. Inf. Syst.*, vol. 1, no. 1, Oct. 2019, doi: 10.30564/jeisr.v1i1.1135.

[4] N. Rajak and D. Shukla, ‘A Systematic Analysis of Task Scheduling Algorithms in Cloud Computing’, in *Social Networking and Computational Intelligence*, Singapore, 2020, pp. 39–49.

[5] P. Singh, M. Dutta, and N. Aggarwal, ‘A review of task scheduling based on meta-heuristics approach in cloud computing’, *Knowl. Inf. Syst.*, vol. 52, pp. 1–51, 2017.

[6] R. A. Al-Arasi and A. Saif, ‘Task scheduling in cloud computing based on metaheuristic techniques: A review

paper', *EAI Endorsed Trans. Cloud Syst. Online First*, 2020, doi: 10.4108/eai.13-7-2018.162829.

[7] S. Bian, X. Huang, and Z. Shao, 'Online Task Scheduling for Fog Computing with Multi-Resource Fairness', in *2019 IEEE 90th Vehicular Technology Conference (VTC2019-Fall)*, Honolulu, HI, USA, Sep. 2019, pp. 1–5, doi: 10.1109/VTCFall.2019.8891573.

[8] M. Ghobaei-Arani, A. Souri, and A. A. Rahmanian, 'Resource Management Approaches in Fog Computing: a Comprehensive Review', *J. Grid Comput.*, vol. 18, no. 1, pp. 1–42, Mar. 2020, doi: 10.1007/s10723-019-09491-1.

[9] Ar. Arunarani, D. Manjula, and V. Sugumaran, 'Task scheduling techniques in cloud computing: A literature survey', *Future Gener. Comput. Syst.*, vol. 91, pp. 407–415, Feb. 2019, doi: 10.1016/j.future.2018.09.014.

[10] P. Hosseinioun, M. Kheirabadi, S. R. Kamel Tabbakh, and R. Ghaemi, 'aTask scheduling approaches in fog computing: A survey', *Trans. Emerg. Telecommun. Technol.*, Jan. 2020, doi: 10.1002/ett.3792.

[11] L. Lei, Y. Tan, K. Zheng, S. Liu, K. Zhang, and X. Shen, 'Deep Reinforcement Learning for Autonomous Internet of Things: Model, Applications and Challenges', *IEEE Commun. Surv. Tutor.*, vol. 22, no. 3, pp. 1722–1760, 2020, doi: 10.1109/COMST.2020.2988367.

[12] H. Mao, M. Alizadeh, I. Menache, and S. Kandula, 'Resource Management with Deep Reinforcement Learning', in *Proceedings of the 15th ACM Workshop on Hot Topics in Networks*, Atlanta GA USA, Nov. 2016, pp. 50–56, doi: 10.1145/3005745.3005750.

[13] X. Liu, Z. Qin, and Y. Gao, 'Resource Allocation for Edge Computing in IoT Networks via Reinforcement Learning', in *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*, Shanghai, China, May 2019, pp. 1–6, doi: 10.1109/ICC.2019.8761385.

[14] Y. Sun, M. Peng, and S. Mao, 'Deep Reinforcement Learning-Based Mode Selection and Resource Management for Green Fog Radio Access Networks', *IEEE Internet Things J.*, vol. 6, no. 2, pp. 1960–1971, Apr. 2019, doi: 10.1109/JIOT.2018.2871020.

[15] J. Baek, G. Kaddoum, S. Garg, K. Kaur, and V. Gravel, 'Managing Fog Networks using Reinforcement Learning Based Load Balancing Algorithm', in *2019 IEEE Wireless Communications and Networking Conference (WCNC)*, Marrakesh, Morocco, Apr. 2019, pp. 1–7, doi: 10.1109/WCNC.2019.8885745.

[16] X. Xiong, K. Zheng, L. Lei, and L. Hou, 'Resource Allocation Based on Deep Reinforcement Learning in IoT Edge Computing', *IEEE J. Sel. Areas Commun.*, vol. 38, no. 6, pp. 1133–1146, Jun. 2020, doi: 10.1109/JSAC.2020.2986615.

[17] A. Nassar and Y. Yilmaz, 'Resource Allocation in Fog RAN for Heterogeneous IoT Environments Based on Reinforcement Learning', in *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*, Shanghai, China, May 2019, pp. 1–6, doi: 10.1109/ICC.2019.8761626.

[18] R. S. Sutton and A. G. Barto, *Reinforcement learning: an introduction*, Second edition. Cambridge, Massachusetts: The MIT Press, 2018.