



Solving the Multi-Objective Problem of IoT Service Placement in Fog Computing Using Reinforcement Learning Approaches

Zahra Saadati¹, Mani Zarei^{2*}

1. MSc. Student, Department of Computer Engineering, Science and Research Branch, Islamic Azad University, Tehran, Iran.
2. Assistant Professor, Department of Computer Engineering, Shahr-e-Qods Branch, Islamic Azad University, Tehran, Iran. (*Corresponding Author*) m.zarei@srbiau.ac.ir

Abstract

Introduction: The data generated in the Internet of Things (IoT) ecosystem requires continuous and timely processing. Transferring generated data to cloud data centers is costly and unsuitable for real-time applications. To increase the speed of service delivery, resources should be placed as close as possible to the user, i.e. at the edge of the network. A new paradigm called fog computing was introduced and added as a layer in the IoT architecture to meet this challenge. Fog computing provides the processing and storage of IoT data locally in the vicinity of IoT devices rather than in the cloud. Fog computing can provide less latency and better service quality for real-time applications than cloud computing. In general, there are theoretical foundations for fog computing, but the issue of locating IoT services to fog nodes remains a challenge and has attracted a great deal of research.

Method: In this research, a conceptual computing framework based on cloud-fog control software is proposed to optimally locate IoT services. The proposed model is formulated as an autonomous planning model for managing service requests due to some constraints, considering the heterogeneity of programs and resources. To solve the problem of locating IoT services, an autonomous evolutionary approach based on enhanced learning approaches has been proposed to make maximum use of fog-based resources and improve service quality. A heterogeneous advantage operator-criterion algorithm is used as a new reinforcement learning approach aimed at maximizing long-term cumulative reward.

Results: The results of the comparisons showed that the proposed reinforcement learning-enabled framework performs better than the advanced methods of the literature. The results of the proposed method compared to FSP-ODMA, SPP-GWO, CSA-FSPP, and GA-FSP methods indicate 4.6%, 2.4%, 3.4%, and 1.1% improvement, respectively.

Discussion: Experimental studies were performed on a simulated artificial environment based on various metrics including fog usage, services performed, response time, and service delay. The proposed reinforcement learning-enabled framework outperforms the previous works and shows better scalability.

Analysis of parallel heuristic algorithms to find a more accurate localization than evolutionary approaches is another aspect of future work. We intend to consider new reinforcement learning approaches such as the Asynchronous Advantage Actor Critic (A3C) algorithm along with the long-term cumulative reward maximization policy for locating services. Also, future efforts will explore reinforcement learning approaches for failure recovery towards Cloud-Fog-IoT architecture, where parallel processing architecture of IoT services can be considered in the location process.

Keywords: Internet of Things, Fog Computing, Service Placement, Reinforcement Learning, Autonomous Planning.



انجمن علمی تجارت الکترونیکی ایران

سامانه‌های پردازشی و ارتباطی چندرسانه‌ای هوشمند

Intelligent Multimedia Processing and Communication Systems (IMPCS)



واحد ریحان

حل مسئله چندهدفه مکان‌یابی سرویس‌های اینترنت اشیاء در محاسبات مه با استفاده از رویکردهای یادگیری تقویتی

دوره چهارم، پاییز ۱۴۰۲
شماره سوم، صص: ۲۹-۴۱

تاریخ دریافت: ۱۹/۰۴/۱۴۰۲
تاریخ پذیرش: ۳۱/۰۵/۱۴۰۲

زهرا سعادت^۱، مانی زارعی^{۲*}

۱- دانشجوی کارشناسی ارشد، گروه مهندسی کامپیوتر، واحد علوم تحقیقات، دانشگاه آزاد اسلامی، تهران، ایران.

۲- استادیار، گروه مهندسی کامپیوتر، واحد شهرقدس، دانشگاه آزاد اسلامی، تهران، ایران. (نویسنده مسئول) m.zarei@srbiu.ac.ir

چکیده: اینترنت اشیاء داده بسیاری را تولید می‌کند و پردازش حجم زیادی از این اطلاعات در مراکز داده ابری صورت می‌گیرد. انتقال داده‌ها به ابر، منجر به ایجاد تأخیر بسیاری در سرویس‌های اینترنت اشیاء می‌شود. بنابراین، به جهت افزایش سرعت در ارائه سرویس‌ها، باید منابع تا جای ممکن نزدیک به کاربران قرار داده شوند. در این مقاله، یک چارچوب محاسبات مفهومی مبتنی بر میان‌افزار کنترل ابر-مه به منظور مکان‌یابی بهینه سرویس‌های اینترنت اشیاء پیشنهاد می‌شود. در روش پیشنهادی با در نظر گرفتن ناهمگن بودن برنامه‌ها و منابع، یک مدل برنامه‌ریزی خودمختار برای مدیریت درخواست‌های سرویس با توجه به برخی محدودیت‌ها به روش تحلیلی ارائه می‌شود. برای حل مسئله مکان‌یابی سرویس‌های اینترنت اشیاء، یک رویکرد تکاملی خودمختار مبتنی بر رویکردهای یادگیری تقویتی با هدف استفاده حداکثر از منابع مه و بهبود کیفیت سرویس پیشنهاد می‌شود. لذا از الگوریتم رویکرد یادگیری تقویتی جدید با هدف حداکثرسازی پاداش جمعی بلندمدت استفاده شده است. مطالعات تجربی بر روی یک محیط مصنوعی شبیه‌سازی شده بر اساس فاکتورهای ارزیابی مختلف شامل استفاده از مه، هزینه سرویس، زمان پاسخ و تأخیر سرویس انجام شده است. به طور میانگین، نتایج روش پیشنهادی نسبت به روش‌های FSP-ODMA، SPP-GWO، CSA-FSPP و GA-FSP به ترتیب ۶٪، ۴٪، ۲٪، ۴٪، ۳٪، ۴٪ و ۱٪، ۱٪ برتری را نشان می‌دهد.

واژه‌های کلیدی: اینترنت اشیاء، محاسبات مه، مکان‌یابی سرویس، یادگیری تقویتی، محاسبات ابر.

۱. مقدمه

با پیشرفت سریع برنامه‌های اینترنت اشیا (IoT) در سال‌های اخیر، استفاده از آن‌ها برای تسهیل فعالیت‌های روزمره در حوزه‌های مختلف برای افزایش کیفیت زندگی انسان به‌طور چشمگیری افزایش یافته است. الگوی محاسبات مه به عنوان یک راه حل محاسباتی توزیع شده برای ارائه این برنامه‌ها با استفاده از گره‌های مه موجود در نزدیکی دستگاه‌های اینترنت اشیا پدیدار شده است. از آنجاکه برنامه‌های کاربردی IoT در قالب چندین سرویس IoT با الزامات مختلف کیفیت خدمات (QoS) توسعه یافته‌اند که می‌توانند بر روی گره‌های مه با قابلیت‌های منابع مختلف در اکوسیستم مه مستقر شوند، یافتن یک طرح کارآمد قرار دادن خدمات یکی از مشکلات چالش برانگیز است که باید در نظر گرفته شود. این تحقیق به مسئله مکان‌یابی سرویس‌های IoT روی گره‌های مه می‌پردازد [1,2].

با توجه به پیشرفت‌های اخیر در فناوری، تعداد دستگاه‌های اینترنت اشیا (IoT) به‌طور قابل توجهی افزایش یافته است. طبق پیش‌بینی Cisco تا سال ۲۰۳۰ بیش از ۵۰۰ میلیارد دستگاه اینترنت اشیا به اینترنت متصل می‌شوند [2,3]. در این راستا، در پارادایم اینترنت اشیا دستگاه‌های ناهمگن با یکدیگر ارتباط برقرار می‌کنند، مقدار زیادی داده تولید می‌کنند و بنابراین به پردازش و ذخیره‌سازی نیاز دارند. با این وجود، دستگاه‌های اینترنت اشیا در ارتباط با سرورهای ابری از مشکل پهنای باند و تأخیر رنج می‌برند [3].

علاوه بر این، داده‌های بزرگ ایجاد شده توسط اینترنت اشیا منجر به ازدحام در سرورهای ابر می‌شود. مراکز داده ابری از نظر جغرافیایی متمرکز هستند و در مسافت طولانی اغلب تأمین نیازها با تأخیر زیاد، کیفیت پایین و ازدحام شبکه مواجه می‌شود [4,5]. برای غلبه بر این محدودیت‌ها و تأمین مؤثر نیازهای برنامه‌های اینترنت اشیا، محاسبات مه معرفی شده است. به‌طور کلی، مجازی‌سازی دستگاه‌های اینترنت اشیا و استفاده از منابع مجازی برای پردازش داده‌ها به‌عنوان محاسبات مه شناخته می‌شود [6,7]. برای پیاده‌سازی محاسبات مه می‌توان از محاسبات خودمختار استفاده کرد. محاسبات خودمختار مفهومی است که بسیاری از زمینه‌های محاسبات را با هدف ایجاد سیستم‌های محاسبات خود مدیر و کاهش مداخله انسان، گرد هم می‌آورد [6,8].

IBM برای انجام محاسبات خودمختار مدل حلقه‌های کنترل خودکار MADE-k را پیشنهاد کرد [7,9,10]. در این مدل، ارکستراتور سرویس یک مؤلفه نرم‌افزاری برای نظارت بر سیستم، تجزیه و تحلیل معیارها و مدیریت منابع است. در این راستا، یک چارچوب محاسبات مفهومی مبتنی بر ترکیب ابر و مه با معرفی میان‌افزار کنترل ابر-مه ارائه شده که درخواست‌های سرویس را با توجه به برخی محدودیت‌ها مدیریت می‌کند. این میان‌افزار وظیفه تصمیم‌گیری و اجرای درخواست‌ها بر اساس مدل پیشنهادی را برعهده دارد. مدل پیشنهادی مبتنی بر حلقه کنترل MADE-k است و از چهار مرحله نظارت (M)، تجزیه و تحلیل (A)، تصمیم‌گیری (D) و اجرای حلقه کنترل (E) با یک پایگاه دانش

مشترک (k) تشکیل شده است. مرحله نظارت وظیفه کنترل وضعیت منابع و سرویس‌ها را برعهده دارد. اولویت‌بندی سرویس‌ها با توجه به مهلت زمانی در مرحله تجزیه و تحلیل انجام می‌شود. مکان‌یابی سرویس روی منابع مناسب با استفاده از رویکردهای یادگیری تقویتی در مرحله تصمیم‌گیری انجام می‌شود و سرانجام در مرحله اجرا، مکان‌یابی‌های تعیین شده اجرامی شوند. به دلیل انعطاف‌پذیری و سادگی، روش‌های فراابتکاری در حل طیف وسیعی از مسائل بهینه‌سازی (اغلب مسائل NP-Hard) تمرکز دارند [8,11,13].

با این حال، این روش‌ها به دلیل فرایند بهینه‌سازی برای مسائل در مقیاس بالا دارای پیچیدگی و همگرایی ضعیف هستند. به همین دلیل رویکردهای یادگیری تقویتی به‌عنوان راه‌حلی مناسب برای حل این مسئله مطرح شده است. در این پژوهش، چارچوبی بر اساس مدل کنترل حلقه MADE-k ارائه می‌شود که مسائل مربوط به پیش‌بینی منابع را پوشش می‌دهد و پویایی سیستم‌های اینترنت اشیا را برای بررسی عملکرد آن‌ها ارزیابی می‌کند.

چارچوب پیشنهادی، سه لایه شامل دستگاه‌های اینترنت اشیا، لایه مه و لایه ابری وجود دارد. برای ارائه پویا منابع از یادگیری تقویتی برای فعال کردن کنترل خودکار حلقه‌ها استفاده می‌شود که در محیط ابری برای دستیابی به محاسبات پویا اجرامی شود. حلقه MADE کنترل یک عامل هوشمند است که تصمیمات لازم را با استفاده از حسگرها و استفاده از این مفاهیم برای تعیین اقداماتی مورد نیاز در محیط ابر اعمال می‌کند. در این پژوهش، برای حل مسئله مکان‌یابی سرویس که برنامه‌های اینترنت اشیا را به سلول‌های مه نگاشت می‌دهد از الگوریتم یادگیری تقویتی Q استفاده شده است که به‌عنوان یک مسئله چندهدفه مدل‌سازی می‌شود. به‌عنوان ماژول برنامه‌ریزی MADE-K، الگوریتم یادگیری Q برای حل این مسئله پیشنهاد شده است. این الگوریتم‌ها از رویکردهای یادگیری تقویتی ارزش محور هستند که برای پیدا کردن سیاست انتخاب عمل بهینه از تابع Q استفاده می‌کنند [9,12,14].

محاسبات مه یک پارادایم جدید محاسباتی است که منابع را در مجاورت دستگاه‌های اینترنت اشیا فراهم می‌کند، جایی که در آن تمام سلول‌های مه در لایه شبکه قرار دارند [15,16]. به همین خاطر در این پژوهش، چارچوب محاسبات پویا و توزیع شده مبتنی بر ترکیب ابر و مه برای مکان‌یابی سرویس‌های اینترنت اشیا را گسترش داده و مقیاس-پذیری و قابلیت اطمینان در مراکز ابر-مه با استفاده از مکان‌یابی خودمختار سرویس و همچنین رضایت کاربران و درآمد ارائه‌دهندگان سرویس‌ها در این مراکز را با استفاده از بهبود فرایند مکان‌یابی سرویس-های اینترنت اشیا افزایش داده است.

۲. پیشینه پژوهش

در سال‌های اخیر، مطالعات زیادی برای حل مسئله مکان‌یابی سرویس در محیط مه ارائه شده‌اند، اما همچنان به تحقیقات بیشتری نیاز است، زیرا عدم موفقیت مکان‌یابی سرویس، باعث افزایش هزینه و تأخیر می‌شود [17,21]. اغلب رویکردهای معرفی شده برای حل مسئله

مکان‌یابی سرویس‌های IoT مبتنی بر فراابتکاری و یادگیری تقویتی هستند. در ادامه برخی از این تحقیقات بررسی شده‌اند. در [17]، سه الگوریتم تکاملی برای مسئله مکان‌یابی سرویس در محاسبات مه مقایسه شدند. این الگوریتم‌ها شامل الگوریتم ژنتیک جمع وزنی، الگوریتم ژنتیک مرتب‌سازی مغلوب و الگوریتم تکاملی چندهدفه بر اساس تجزیه (MOEA/D) هستند. نتایج عدم کارایی این الگوریتم‌ها را نشان می‌دهد، زیرا آن‌ها اولویت‌بندی برنامه‌ها را در زمان اجرا در نظر نمی‌گیرند.

در [18]، یک الگوریتم زمان‌بندی برای درخواست‌های سرویس اینترنت اشیا با استفاده از برنامه‌نویسی عدد صحیح ارائه شد. این روش از یک الگوریتم ژنتیک بهبودیافته با هدف حداقل‌سازی کاهش تأخیر استفاده می‌کند تا ماهیت پویای محیط مه را در نظر بگیرد. نتایج تأخیر این روش را بین ۲۱/۹٪ تا ۴۶/۶٪ نشان می‌دهد که امیدوارکننده است. در [19]، یک رویکرد خودمختار برای مکان‌یابی سرویس اینترنت اشیا با استفاده از بهینه‌سازی گرگ خاکستری در محیط محاسبات مه پیشنهاد شده است. این روش عملکرد سیستم را با در نظر گرفتن هزینه‌های اجرا افزایش داده و با رفتار پویای محیط سازگار می‌شود. نتایج شبیه‌سازی میانگین زمان انتظار ۱۰۰٪ را برای سرویس‌ها گزارش داده است. در [20]، یک چارچوب مفهومی برای تأمین منابع در محاسبات مه ارائه شده است. هدف این تحقیق بهینه‌سازی مکان‌یابی سرویس اینترنت اشیا با در نظر گرفتن تأخیر در تخصیص منابع محاسباتی است، جایی که ناهمگنی برنامه و منابع لحاظ شده است. برای حل این مسئله یک روش فراابتکاری مبتنی بر الگوریتم ژنتیک ارائه شده که باعث کاهش تأخیرهای ارتباطی و استفاده بهتر از منابع مه می‌شود.

در [21]، یک روش مکان‌یابی برای برنامه‌های متقارن اینترنت اشیا در محیط‌های لبه و مه بر اساس الگوریتم ممتیک پیشنهاد شده است. این روش یک مدل هزینه وزنی برای حداقل‌سازی زمان اجرا و مصرف انرژی است که در محیطی شامل دستگاه‌های اینترنت اشیا و همچنین سرورهای مه، لبه و ابر اعمال می‌شود. نتایج بهبود هزینه وزنی برنامه‌های اینترنت اشیا را تا ۶۵٪ نسبت به رویکردهای مشابه نشان می‌دهد. در [22]، یک تکنیک یادگیری تقویتی عمیق توزیع شده برای مکان‌یابی برنامه‌های اینترنت اشیا در محیط‌های محاسباتی لبه و مه ارائه شده است. این تکنیک بر اساس معماری‌های بازیگر-آموزنده با در نظر گرفتن وزن کار می‌کند. نتایج ارزیابی این روش نشان می‌دهد که به طور قابل توجهی هزینه اجرای برنامه‌های اینترنت اشیا را تا ۳۰٪ در مقایسه با هم‌تایان خود بهبود می‌بخشد.

در [23]، یک رویکرد مکان‌یابی تطبیقی را در محیط مه-ابر معرفی شد که QoS را از نظر تأخیر و مصرف انرژی بهبود می‌دهد. نویسندگان از یک سیاست مبتنی بر یادگیری تقویتی برای تعیین نگاشت بهینه بین گره‌های مه و سرویس‌های اینترنت اشیا استفاده می‌کنند. در [24]، یک چارچوب محاسبات توزیع شده برای مدیریت منابع در محاسبات مه معرفی شده است. در اینجا، حلقه کنترل MADE-k وظیفه تأمین سرویس برنامه‌های اینترنت اشیا را برعهده دارد. در این راستا، تکنیک

یادگیری تقویتی برای مازول تصمیم‌گیری و تکنیک رگرسیون بردار پشتیبان برای مازول تجزیه و تحلیل استفاده شده است. نتایج این روش میانگین هزینه و تأخیر بهتری را در مقایسه با الگوریتم‌های مشابه نشان می‌دهد. در مطالعه‌ای دیگر، یک معماری برای فراهم‌سازی سرویس‌ها در کلونی‌های مه معرفی و سپس چند الگوریتم برای استفاده در این معماری پیشنهاد شده است [25].

هدف این الگوریتم‌ها بهینه‌سازی نرخ استفاده از منابع مه است. در [26] رویکردی برای مکان‌یابی سرویس‌ها در شبکه‌های IoT را با کمک مه بلادرنگ مطرح کرده‌اند. نویسندگان از یک الگوریتم ابتکاری مبتنی بر خوشه‌بندی گره‌های مه برای حل مسئله استفاده می‌کنند. این الگوریتم بر اساس QoS نتایج امیدوارکننده‌ای را گزارش داده است.

در پژوهشی دیگر نویسندگان رویکرد CSA-FSP را پیشنهاد دادند که از الگوریتم جستجوی فاخته (CSA) برای حل مسئله مکان‌یابی سرویس مه (FSP) استفاده می‌کند [27]. مکان‌یابی سرویس‌ها را با استفاده از یک میان‌افزار کنترل ابر-مه متمرکز انجام می‌دهد و ارتباطات بین ابر و مه را برای کاهش تبادل داده‌ها مدیریت می‌کند. در اینجا، FSP به‌عنوان یک مسئله بهینه‌سازی چندهدفه مبتنی بر آرشیو پارتو در نظر گرفته شده است.

در [28]، این رویکرد از یک چارچوب محاسبات مفهومی سه‌لایه برای توصیف تعاملات بین اجزای سیستم استفاده می‌کند. FSP-ODMA هزینه استقرار برنامه‌های IoT را به‌طور میانگین تا ۱۰٪ نسبت به رویکردهای فراابتکاری مشابه کاهش می‌دهد.

در [29] یک رویکرد مقرون به صرفه با استفاده از الگوریتم بهینه‌سازی نهنگ (WOA) برای مکان‌یابی سرویس مه پیشنهاد شده است (WOA-FSP). WOA-FSP یک رویکرد مکان‌یابی خودمختار مبتنی بر مدل MADE-k است که SPP را با در نظر گرفتن مصرف انرژی و توان عملیاتی در حین ارضا الزامات QoS حل می‌کند. محققان در [30] بر روی یکپارچه‌سازی ابر و اینترنت اشیا مطالعه داشته‌اند. آن‌ها جداگانه، چندین مطالعه ادبی مثل کیفیت‌های اساسی، فناوری و تحقیقات آینده را روی Cloud و IoT بررسی کرده‌اند. متأسفانه، ما هنوز به‌طور کامل تحلیل مفاهیم ابر و اینترنت اشیا را درک نکرده‌ایم. به همین خاطر نیاز به یکپارچه شدن آن‌ها، و مسائلی که اتفاق می‌افتد و اینکه مطالعات قبلی چگونه به این مسائل می‌پردازد، را مورد آنالیز قرار می‌دهیم. این تحقیق، چندین حوزه از محاسبات ابری را که بر اینترنت اشیا تسلط دارند، تجزیه و تحلیل خواهد کرد و چالش‌های باز و پاسخ‌های احتمالی برای اینترنت آینده در رایانش ابری را مشخص می‌شود.

۳. مدل پیشنهادی

شرکت ماشین‌های تجاری بین‌المللی (IBM) برای پیاده‌سازی محاسبات مه مدل خودمختار MADE-k را معرفی کرد [7]. یکی از مهمترین مسائل در مدیریت منابع در ابر و مه، تصمیم‌گیری درباره فراهم‌سازی سرویس‌ها یا مکان‌یابی سرویس‌ها به‌صورت خودمختار است. هدف از حل این مسئله مکان‌یابی توزیع شده اجزای برنامه‌های اینترنت اشیا (سرویس‌ها) در



شکل ۱: نمودار جریان مدل پیشنهادی

۱.۳. معماری مدل پیشنهادی

چارچوب محاسبات مه پیشنهادی شامل معماری چهار لایه ابر-مه-لایه ارتباطات-دستگاه‌های اینترنت اشیا است، همان‌طور که در شکل ۲ آمده. ابر در بالاترین سطح است و پس از آن لایه مه قرار دارد و بعد لایه ارتباطات و در پایین‌ترین سطح دستگاه‌های اینترنت اشیا است. لایه دستگاه‌های اینترنت اشیا مجموعه‌ای از اشیاء هوشمند است که داده‌ها را جمع‌آوری کرده و به لایه ارتباطات ارسال می‌کند و بعد درخواست‌های دریافتی توسط لایه ارتباطات به لایه مه منتقل می‌شود. لایه مه درخواست‌های دریافتی از لایه دستگاه‌های اینترنت اشیا را برای فراهم‌سازی سرویس پردازش می‌کند. بر اساس برخی مشخصه‌ها (برای مثال بلادرنگ بودن)، هر درخواست می‌تواند در همان سطح (لایه مه) اجرا شود یا به لایه ابر ارسال گردد.

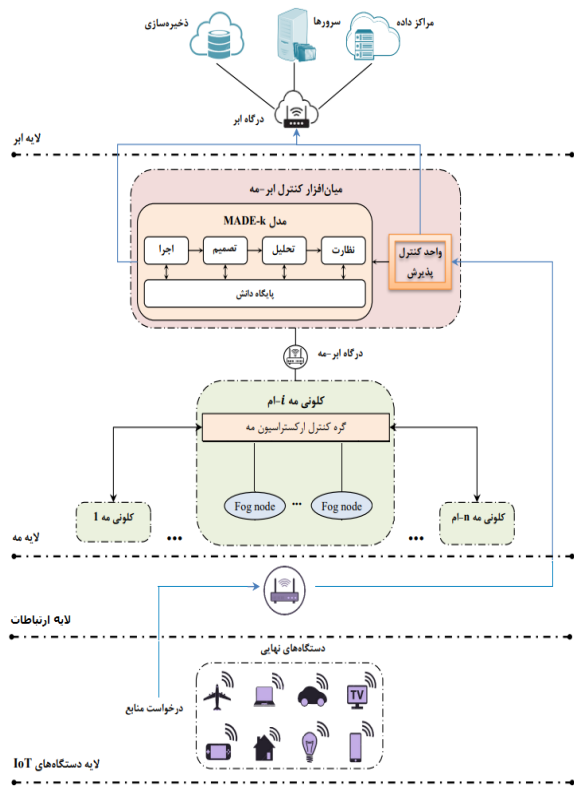
چارچوب محاسبات مفهومی پیشنهادی مبتنی بر یک میان‌افزار کنترل ابر-مه است که قابلیت مدیریت بهینه منابع را فراهم می‌کند. این میان‌افزار وظیفه کنترل چشم‌انداز مه تابع و تعامل با لایه ابر را برعهده دارد. میان‌افزار کنترل ابر-مه بر اساس مدل MADE-k می‌تواند برنامه‌ها/درخواست‌های دریافتی از دستگاه‌های اینترنت اشیا را پردازش و بدون درگیری با ابر، بستر مناسب برای سرویس‌دهی آن‌ها را مشخص کند. لایه مه از چندین کلونی مه تشکیل شده است. هر کلونی مه شامل یک گره کنترل ارکستراسیون مه و تعدادی گره مه است. گره ارکستراتور وظیفه مدیریت و نظارت بر گره‌های مه تابع را برعهده دارد و توسط میان‌افزار کنترل ابر-مه پشتیبانی می‌شود.

میان‌افزار کنترل ابر-مه دارای یک واحد کنترل پذیرش است که می‌تواند درخواست‌ها را برای فراهم‌سازی منابع پردازش کند. درخواست‌های دستگاه‌های اینترنت اشیا از طریق درگاه مه به واحد کنترل پذیرش ارسال می‌شود. این واحد دارای یک آستانه زمان پاسخ برای شناسایی

منابع ابر و مه است که باعث نگاشت بین گره‌های مه و برنامه‌های IoT برای استفاده از منابع می‌شود. از مزایای فراهم‌سازی پویای سرویس‌ها می‌توان به کاهش استفاده از پهنای باند، مدیریت بهینه منابع، کاهش هزینه‌ها، دسترس‌پذیری، کاهش تأخیر، افزایش قابلیت اطمینان، کاهش انرژی، بهبود QoS و برآورد زمان پاسخ موردنیاز برای برنامه‌های بلادرنگ اشاره کرد.

مطالعات زیادی در ادبیات برای بهبود تصمیم‌گیری و استقرار بهینه سرویس‌های اینترنت اشیا روی گره‌های مه وجود دارد. هدف اغلب این مطالعات بهینه‌سازی یک تابع هدف (برای مثال، مصرف انرژی، زمان پاسخ، تأخیر، هزینه و استفاده از منابع مه) است، با این حال، برخی از مطالعات نیز بهینه‌سازی را با در نظر گرفتن چندین هدف (چندهدفه) انجام می‌دهند. حل مسئله به صورت چندهدفه می‌تواند تصمیم‌گیری بهتری را برای نگاشت گره‌های مه به برنامه‌های اینترنت اشیا ارائه دهد. از اینرو، در این تحقیق، یک الگوریتم چندهدفه برای مسئله مکان‌یابی سرویس‌های اینترنت اشیا در چشم‌انداز مه با هدف برقراری مصالحه میان هزینه، تأخیر و استفاده از منابع پیشنهاد شده است. به دلیل انعطاف‌پذیری و سادگی، رویکردهای یادگیری تقویتی در حل طیف وسیعی از مسائل بهینه‌سازی تمرکز دارند. بنابراین، در این تحقیق برای حل مسئله مکان‌یابی سرویس‌های اینترنت اشیا از یک رویکرد یادگیری تقویتی استفاده شده است.

مدل پیشنهادی ما شامل چهار مرحله نظارت، تجزیه و تحلیل، تصمیم‌گیری و اجرا است که با یک پایگاه دانش (k) مشترک شده‌اند. کنترل سرویس‌های IoT و منابع مه در مرحله نظارت انجام می‌شود. مرحله تجزیه و تحلیل شامل اولویت‌بندی اجرای سرویس‌ها بر اساس مهلت زمانی است. در مرحله تصمیم‌گیری برنامه‌ریزی برای مکان‌یابی سرویس‌ها و سرانجام در مرحله اجرا، مکان‌یابی برنامه‌ریزی شده اعمال می‌گردد. بر اساس این مدل، در این تحقیق یک چارچوب محاسبات مفهومی خودمختار مبتنی بر میان‌افزار کنترل ابر-مه برای مدیریت بهینه منابع در ابر و مه ارائه می‌شود. این میان‌افزار به صورت متمرکز مسئول اعمال مدل MADE-k روی کل کلونی‌های مه است. مرحله تصمیم‌گیری از مدل MADE-k برای برنامه‌ریزی بهینه استقرار سرویس‌ها روی گره‌های مه بهره‌می‌گیرد که با استفاده از یک رویکرد یادگیری تقویتی نظیر یادگیری Q انجام می‌شود. شکل ۱ نمودار جریان روش پیشنهادی را نشان می‌دهد.



شکل ۲: معماری مدل پیشنهادی

۱،۲،۳. استفاده مه (UF)

این معیار تعداد مکان‌های سرویس استفاده‌شده برای منابع مه را نشان می‌دهد که باید حداکثر شود. معیار UF بر اساس تأخیر ارتباط و مکان‌یابی مرتبط با ارسال سرویس به نزدیکترین کلونی همسایه یا ابر محاسبه می‌شود، زیرا این امر می‌تواند منجر به نقض مهلت برنامه‌ها شود. از این رو، برای محاسبه UF باید اولویت برنامه‌ها بر اساس مهلت محاسبه شود. بر این اساس، نرخ استفاده از منابع مه بر مبنای رابطه (۲) محاسبه می‌شود [20,25].

$$UF = \sum_{A^k \in AP} \left(P(A^k) * \sum_{a^l \in A^k} \left(\sum_{f^j \in Res^{a^l}(F^i)} x_{a^l}^{f^j} + x_{a^l}^O + x_{a^l}^N \right) \right) \quad (3)$$

جایی که، AP مجموعه برنامه‌های درخواست‌شده، a^l سرویس l -ام از A^k ، f^j گروه مه z -ام، F^i کلونی مه i -ام، $Res^{a^l}(F^i)$ مجموعه گره‌های مه مرتبط با a^l در F^i است. $P(A^k)$ اولویت برنامه A^k است که توسط رابطه (۱) محاسبه می‌شود. همچنین، مطابق با هر گره مه که آماده اجرای سرویس a^l است، سه متغیر تصمیم به صورت باینری تعریف شده‌است شامل $x_{a^l}^{f^j}$ ، $x_{a^l}^O$ و $x_{a^l}^N$ که به ترتیب به گره مه z -ام، گره کنترل ارکستراسیون مه و نزدیکترین کلونی همسایه اشاره دارند. متغیرهای تصمیم نشان می‌دهند که سرویس a^l به کدام منبع نگاشت شده‌است.

برنامه‌های بلادرنگ است. مدل MADE-k در میان‌افزار در مورد استقرار هر سرویس IoT تصمیم‌گیری می‌کند و این کار را برای هر کلونی مه به صورت توزیع‌شده انجام می‌دهد. گره‌های مه دارای قدرت محاسباتی (یعنی، CPU، RAM، فضای ذخیره‌سازی) هستند، بنابراین می‌توانند سرویس‌های IoT را اجرا کنند.

در مرحله نظارت میزان استفاده از CPU، RAM و ذخیره‌سازی تجزیه و تحلیل شده و سپس، وضعیت گره‌های مه، سرویس‌های IoT و دستگاه‌های IoT در پایگاه دانش ذخیره می‌شوند. در مرحله تجزیه و تحلیل مطابق رابطه (۱) اولویت اجرای برنامه‌ها تعیین می‌گردد. این کار برای کاهش تأخیر در اجرای برنامه‌ها است و بر اساس فاصله زمانی تا مهلت اجرا انجام می‌شود [19].

$$P(A_k) = \alpha \times \frac{1}{D_{A_k}} + (1 - \alpha) \times \frac{1}{t - R_{A_k}} \quad (1)$$

جایی که، $P(A_k)$ اولویت برنامه A_k را نشان می‌دهد. R_{A_k} و D_{A_k} به ترتیب مهلت زمانی و زمان درخواست سرویس برای برنامه A_k است. همچنین، t زمان فعلی و α ضریب وزنی برای در نظر گرفتن تأثیر پارامترهای R_{A_k} و D_{A_k} است.

الگوریتم یادگیری تقویتی برای برنامه‌ریزی استقرار برنامه‌ها (سرویس‌های اینترنت اشیا) روی گره‌های مه اعمال می‌شود. برنامه‌ریزی استقرار مرتبط با مرحله تصمیم‌گیری از مدل MADE-k است. در نهایت، در مرحله اجرا، طرح برنامه‌ریزی ایجادشده روی کلونی‌های مه اعمال می‌شود.

۲.۳. تابع هدف

در این تحقیق، مسئله مکان‌یابی سرویس به عنوان یک مسئله بهینه‌سازی چند هدفه برای به حداکثر رساندن استفاده از مه و توان عملیاتی و به حداقل رساندن زمان پاسخ و هزینه سرویس در محاسبات مه فرموله می‌شود. بنابراین، تابع هدف پیشنهادی برای برنامه‌ریزی استقرار شامل چهار هدف می‌باشد، همان‌طور که در رابطه (۲) نشان داده شده‌است.

$$\min \quad \xi_{RT} * RT + \xi_{SC} * SC - \xi_{UF} * UF - \xi_{TP} * TP \quad (2)$$

که UF ، TP ، ST و SC به ترتیب استفاده از مه، توان عملیاتی، زمان پاسخ و هزینه سرویس است. همچنین، ξ_{UF} ، ξ_{TP} ، ξ_{RT} و ξ_{SC} به ترتیب وزن‌های این اهداف هستند. در ادامه فرایند محاسبه UF ، TP ، ST و SC بحث می‌شود.

۲.۲.۳. توان عملیاتی (TP)

توان عملیاتی بر اساس فاکتورهای مختلفی محاسبه می‌شود. این معیار می‌تواند به معنای حداکثر کردن تعداد سرویس‌های اجرا شده نسبت به کل سرویس‌های ارسال شده به کلونی‌های می باشد. همچنین، SC می‌تواند به معنای حداقل کردن هزینه اجرای این سرویس‌ها نیز باشد. در اینجا، بر اساس تفسیرهای مختلف از یک رویکرد ترکیبی برای محاسبه توان عملیاتی استفاده می‌شود. رابطه (۴) برای محاسبه توان عملیاتی تعریف شده است [19].

$$TP = TP_A - TP_B \quad (4)$$

جایی که، TP_A توان عملیاتی مرتبط با تعداد سرویس‌های اجرا شده نسبت به کل سرویس‌های ارسالی به کلونی‌های می باشد. همچنین، TP_B هزینه عملیاتی مرتبط با اجرای این سرویس‌ها می‌باشد که پس از استقرار کامل در نظر گرفته می‌شود. TP_A و TP_B بر اساس روابط (۵) و (۶) محاسبه می‌شوند.

$$TP_A = \frac{Srv_D}{Srv_{TD}} \quad (5)$$

$$TP_B = Cost_{Srv_D} \quad (6)$$

به طوری که، Srv_D تعداد سرویس‌هایی است که قبل از مهلت روی گره‌های می مستقر و اجرا شده‌اند و Srv_{TD} به کل سرویس‌های ارسالی به کلونی می اشاره دارد. همچنین، $Cost_{Srv_D}$ به مجموع هزینه سرویس‌های اجرا شده در کلونی می قبل از مهلت اشاره دارد.

۳.۲.۳. زمان پاسخ (RT)

به طور کلی، زمان پاسخ سرویس a^l به فاصله زمانی بین دریافت سرویس توسط گره کنترل ارکسترسیون می تا اجرای آن اشاره دارد. این زمان به عنوان RT_{a^l} تعریف و مطابق رابطه (۷) محاسبه می‌شود [25].

$$RT_{a^l} = WT_{a^l} + EC_{a^l} \quad \forall a^l \in A^k, A^k \in AP \quad (7)$$

که WT_{a^l} زمان انتظار برای استقرار سرویس و EC_{a^l} زمان اجرای سرویس روی گره می است. WT_{a^l} بر اساس مجموع زمان انتظار برای شروع پررود زمانی بعدی و زمان لازم برای یافتن گره می با کمترین هزینه اجرای سرویس محاسبه می‌شود. علاوه بر این، EC_{a^l} بر اساس هزینه اجرای سرویس روی یک گره می در نظر گرفته می‌شود.

۴.۲.۳. هزینه (SC)

هزینه سرویس مرتبط با ارتباطات و محاسبات بر اساس استفاده از منابع می است، همانطور که در رابطه (۸) نشان داده شده است [19].

$$SC = SC_{comm} + SC_{comp} \quad (8)$$

جائیکه، SC_{comm} هزینه ارتباطات بین دو سرویس متوالی (برای مثال، t_α و t_β) روی یک گره می است و SC_{comp} به هزینه محاسبات اشاره دارد. این هزینه‌ها در روابط (۹) و (۱۰) تعریف شده است.

$$SC_{comm} = CP \cdot \frac{cv_{\alpha,\beta}}{BW} \quad (9)$$

$$SC_{comp} = PP(t_\beta - t_\alpha) \quad (10)$$

جایی که، CP و PP به ترتیب بهای واحد ارتباطات و محاسبات برای یک گره می است. $cv_{\alpha,\beta}$ اندازه داده بین درخواست‌های t_α و t_β و BW به پهنای باند بین دو گره می است. سعیدی و همکاران (۲۰۲۰) $CP = 0.1\$$ و $BW = 20 Mbps$ را پیشنهاد می‌دهند.

۳.۳. طرح مکان‌یابی پیشنهادی در محاسبات می

هدف از حل مسئله مکان‌یابی سرویس‌های IoT، بهینه‌سازی نگاشت بین گره‌های می و سرویس‌ها است به نحوی که برخی محدودیت‌ها ارضاء شوند. در این تحقیق از الگوریتم یادگیری Q به عنوان یک رویکرد یادگیری تقویتی برای کار مکان‌یابی سرویس‌های اینترنت اشیا استفاده شده است. از این رو، مسئله مکان‌یابی سرویس به عنوان یک مسئله بهینه‌سازی چندهدفه برای به حداکثر رساندن استفاده از می و به حداقل رساندن زمان پاسخ و هزینه سرویس در محاسبات می فرموله می‌شود. الگوریتم یادگیری Q به عنوان یک طرح برنامه‌ریزی استقرار در فاز تصمیم‌گیری از مدل MADE-k پیکربندی می‌شود. در اینجا، درخواست‌های ارسالی از دستگاه‌های IoT در صف موجود در میان‌افراز کنترل ابر-می قرار می‌گیرند و هر ρ دوره زمانی برنامه‌ریزی می‌شوند. در یادگیری Q، ابتدا مقدار $Q(s, a)$ را به ازای تمام حالات و اعمال برابر صفر در نظر گرفته می‌شود. سپس، حالت فعلی سیستم (s) محاسبه و پس از آن، بر اساس الگوریتم ϵ -greedy، یک عمل (a) انتخاب می‌شود. عمل a را روی محیط انجام می‌شود و منتظر امتیاز عمل خود (r) می‌شود. سپس، حالت جدید سیستم را که پس انجام عمل سیستم به آن می‌رود (s') محاسبه می‌گردد. در نهایت، بر اساس رابطه (15) مقدار $Q(s, a)$ را به روزرسانی می‌شود. در آخر بررسی می‌شود که عامل به هدف خود دست یافته است یا خیر. اگر پاسخ منفی بود، الگوریتم از مرحله سوم بر اساس موقعیت s' دوباره تکرار می‌شود. در غیر این صورت الگوریتم خاتمه می‌یابد.

۳. ۱. ۳. تنظیمات الگوریتم یادگیری Q

که، R^{mem} حافظه مربوط به پاداش‌های گذشته است. به‌طور کلی، هدف روش پیشنهادی حداکثرسازی پاداش تجمعی بلند مدت است که از طریق رابطه (۱۴) محاسبه می‌شود.

$$\psi_{\tau}^* = \arg \max_{\psi_{\tau}} \mathbb{E}_{\psi_{\tau}} \left[\sum_{\tau=0}^{\infty} \omega * R_{\tau}(S_{\tau}, A_{\tau}) \right] \quad (14)$$

که ω فاکتور میرایی برای پاداش آینده است. حالت S_{τ} با دریافت هر درخواست جدید ایجاد می‌شود. به‌طور مشابه، شبکه صف حافظه خود را بر اساس درخواست جدید به‌روزمی‌کند. سپس، عامل عمل A_{τ} را بر اساس سیاست تعیین‌شده انجام می‌دهد. در همین حال، $R_{\tau}(S_{\tau}, A_{\tau})$ با توجه به بازخورد از محیط محاسبه می‌شود. در هر تکرار تابع ارزش در یادگیری Q به صورت رابطه (۱۵) به‌روزمی‌شود [31].

$$\hat{Q}(s, a) \leftarrow \alpha [r(s, a) + \gamma \max_{\hat{a}} \hat{Q}(s, \hat{a})] \quad (15)$$

در این رابطه α عددی بین ۰ و ۱ برای تعیین نرخ یادگیری است و سرعت یادگیری را تنظیم می‌کند. بدیهی است هرچه مقدار α بیشتر باشد، سرعت یادگیری هم بالاتر خواهد بود، ولی باید توجه داشت که مقادیر خیلی بزرگ α یادگیری را ناپایدار خواهد کرد. γ عددی بین ۰ و ۱ برای تعیین نرخ تنزیل است و همچنین این متغیر مانع از واگرایی تابع کیفیت هنگام یادگیری می‌شود. علاوه بر این $\max_{\hat{a}} \hat{Q}(s, \hat{a})$ کیفیت بهینه‌ترین عمل در موقعیت جدید سیستم (s) است. علاوه بر این، روش پیشنهادی شامل یک مکانیسم بازبایی آزاد کردن منابع اشغال‌شده توسط درخواست جاری در هنگام مکان‌یابی ناموفق است. مکانیسم بازبایی مسئول ریست کردن محیط به شرایط قبل از مکان‌یابی درخواست جاری است.

۳. ۲. ۳. شبکه صف

توزیع کارآمد استخراج منابع استفاده‌شده در گره‌ها در طول زمان می‌تواند منجر به افزایش تعداد درخواست‌های پذیرفته، شود. روش پیشنهادی از یک شبکه صف برای استخراج توزیع منابع در گره‌ها استفاده می‌کند. این صف شامل یک حافظه و چندین LSTM است. LSTM می‌تواند رابطه بین موقعیت نسبی بین گره‌های تخصیصی را کشف کند و به‌طور گسترده برای پیش‌بینی مسائل سری‌های زمانی استفاده می‌شود. در اینجا، حافظه صف برای ذخیره منابع محاسباتی مرتبط با هر درخواست در طول زمان استفاده می‌شود. در اینجا، حافظه صف برای ذخیره منابع محاسباتی مرتبط با هر درخواست SFC در طول زمان استفاده می‌شود (یعنی، S_{τ}). در این تحقیق از پارامتر عمق زمانی (TD) برای نشان دادن طول حافظه صف استفاده می‌شود. با توجه به محدودیت طول صف، آخرین لایه حافظه صف (یعنی، $S_{\tau-TD}$) با اضافه شدن هر درخواست جدید حذف می‌شود.

عمل، حالت، پاداش / تنبیه و سیاست از اجزای اصلی یادگیری Q هستند. در اینجا، پاداش منفی به‌منزله تنبیه است. در هر مرحله از یادگیری، عامل بر اساس مشاهده حالت S_{τ} و مطابق با سیاست تعریف‌شده عمل A_{τ} را انجام می‌دهد. τ به یک دوره زمانی در یادگیری اشاره دارد. بر اساس تغییر محیط، عامل پاداش $R_{\tau}(S_{\tau}, A_{\tau})$ را دریافت کرده و به حالت بعدی $S_{\tau+1}$ به‌روزمی‌شود. سرانجام، عامل سیاست را به‌روز می‌کند تا بتواند پاداش تجمعی بلندمدت بیشتری به‌دست آورد.

حالت: جزئیات شبکه شامل منابع باقیمانده همه گره‌های در دسترس می‌باشد. این شامل حافظه باقی‌مانده (\bar{f}_j^M)، منابع ذخیره‌سازی باقی‌مانده (f_j^S) و منابع محاسباتی باقیمانده (\bar{f}_j^U) است (در اینجا z به گره z -ام در کلونی جاری اشاره دارد). علاوه بر این، مشخصات درخواست جاری شامل حافظه مورد نیاز (M_{at})، منابع ذخیره‌سازی مورد نیاز (S_{at})، منابع محاسباتی مورد نیاز (U_{at}) و مهلت (D_{at}) نیز به‌عنوان حالت در نظر گرفته می‌شود (در اینجا l به سرویس l -ام اشاره دارد).

عمل: عامل در هر مرحله می‌تواند $a^l \in A_k$ را روی یک گره مه مستقر کند. بنابراین، عمل A_{τ} شامل مجموعه همه گره‌های مه در دسترس در کلونی جاری می‌باشد، طبق رابطه (۱۱):

$$A_{\tau} \in Res(F^l) \quad (11)$$

پاداش: در هر مرحله عامل یک عمل که شامل انتخاب گره $f^j \in Res(F^l)$ برای میزبانی از $a^l \in A_k$ است را انجام می‌دهد. مکان‌یابی موفقیت‌آمیز است، هنگامی که f^j منابع مورد نیاز a^l را برآورد کند. بنابراین، عامل یک پاداش دریافت می‌کند. در صورتی که عمل A_{τ} نتواند محدودیت‌ها را برآورد کند، مکان‌یابی با شکست مواجه می‌شود. در الگوریتم یادگیری Q پیشنهادی، پاداش بر اساس تابع هدف تعریف‌شده در رابطه (۲) محاسبه می‌شود، طبق رابطه (۱۲):

$$R_{\tau} = \xi_{UF} * UF + \xi_{TP} * TP - \xi_{RT} * RT - \xi_{SC} * SC \quad (12)$$

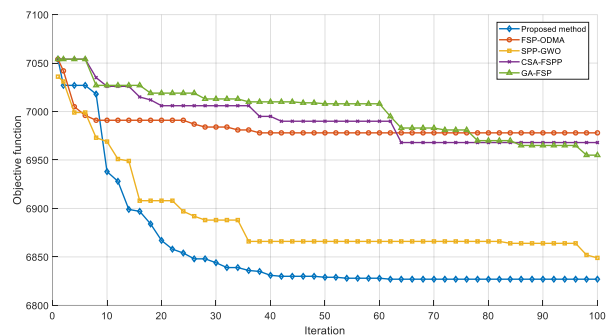
جایی که، UF ، TP ، ST و SC به ترتیب استفاده از مه، توان عملیاتی، زمان پاسخ و هزینه سرویس است. همچنین، ξ_{UF} ، ξ_{TP} ، ξ_{RT} و ξ_{SC} به ترتیب وزن‌های این اهداف هستند. از طرف دیگر، اگر عامل موفق به استقرار موفقیت‌آمیز سرویس $a^l \in A_k$ نشود، و در این صورت شکست در مکان‌یابی تلقی شده و باید تنبیه شود. در اینجا، یک مقدار به‌عنوان پاداش منفی R_{τ} اعمال می‌شود. پاداش منفی برای A_{τ} بر اساس رابطه (۱۳) محاسبه می‌شود.

$$R_{p\tau} = -\xi_{TP} * TP - \xi_{SC} * SC - \sum_{R_k \in R^{mem}} R_k \quad (13)$$

۴. بحث و مقایسه

برای مقایسه عملکرد الگوریتم پیشنهادی از چهار الگوریتم تکاملی شامل FSP-ODMA و همچنین روش پیشنهادی در حل مسئله مکان‌یابی سرویس‌های IoT روی منابع مه و ابر استفاده شده است. الگوریتم‌ها با متریک‌های کارایی مختلف ارزیابی می‌شوند. این متریک‌ها شامل سرعت همگرایی، زمان اجرا، مکان استقرار سرویس‌ها، تعداد سرویس‌های انجام شده، میانگین زمان انتظار، تعداد سرویس‌های ناموفق، تعداد سرویس‌های باقیمانده هستند. نتایج الگوریتم‌ها برای ۱۰ دوره متوالی و ۱۰۰۰ سرویس مقایسه می‌شوند.

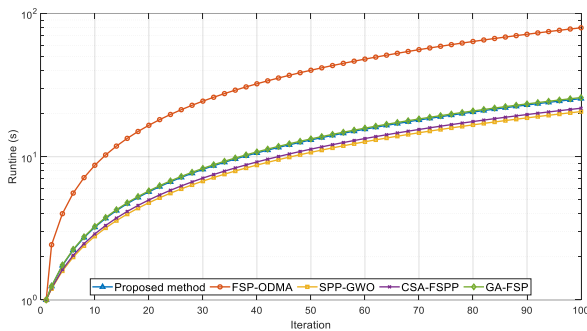
در ابتدا همگرایی الگوریتم‌های مختلف در کاهش تابع برازندگی مقایسه می‌شوند. همگرایی سرعت رسیدن به جواب نهایی را نشان می‌دهد و باعث توقف تکامل می‌شود. مقایسه سرعت همگرایی الگوریتم‌های مختلف در شکل ۳ آمده است. این مقایسه بر اساس ۱۰۰ تکرار انجام شده است، به طوری که در هر تکرار برای هر الگوریتم تابع هدف گزارش شده است. طبق نمودار فوق، کارایی و سرعت همگرایی روش پیشنهادی برای یافتن برنامه‌ریزی استقرار سرویس‌ها در کلونی مه در مقایسه با سایر الگوریتم‌ها بهتر است. روش پیشنهادی تنها با ۴۰ تکرار موفق کاهش تابع برازندگی تا حدود ۶۸۳۰ شده است. با این حال، SPP-GWO در تکرار ۳۵ همگرا شده است، اما کارایی این الگوریتم بیش از ۵٪ از روش پیشنهادی کمتر است. پس از روش پیشنهادی و SPP-GWO، به ترتیب الگوریتم‌های GA-FSP، CSA-FSP و FSP-ODMA نتایج مطلوب‌تری ارائه داده‌اند.



شکل ۳: نتایج الگوریتم‌های مختلف بر اساس سرعت همگرایی

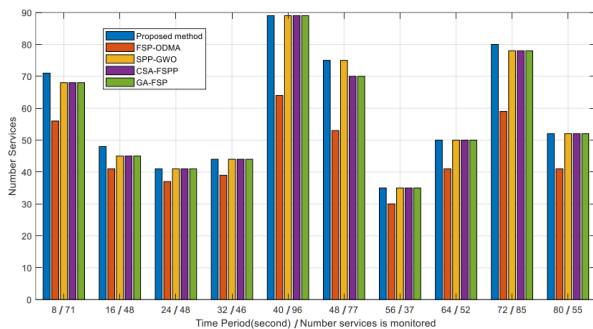
اگرچه راه‌حل ارائه شده توسط روش پیشنهادی از سرعت و هزینه کمتری برخوردار است، اما با توجه به پویایی محیط و نیاز به تصمیم‌گیری زمان واقعی باید متریک زمان اجرا نیز بررسی شود. به طور کلی، زمان اجرای پایین برای برنامه‌ریزی باعث کاهش زمان انتظار سرویس‌ها برای استقرار روی گره‌های مه می‌شود. از این رو، شکل ۴ نتایج مربوط به زمان اجرای الگوریتم‌های مختلف را در ۱۰۰ تکرار مستقل گزارش می‌دهد. نتایج به وضوح عدم کارایی FSP-ODMA را در حل مسائل زمان واقعی و محیط‌های پویا نشان می‌دهد. این الگوریتم با ۱۱۱ ثانیه زمان اجرا بدترین عملکرد را در بین سایر الگوریتم‌ها دارد. دلیل این می‌تواند خصوصیت FSP-ODMA در محاسبه تابع برازندگی پی در پی و ایجاد

راه‌حل‌های جدید باشد. سایر الگوریتم‌ها از زمان اجرای تقریباً مشابهی برخوردار هستند، با این حال، نتایج SPP-GWO اندکی بهتر است.



شکل ۴: نتایج الگوریتم‌های مختلف بر اساس زمان اجرا

در آزمایش دیگری الگوریتم‌ها بر اساس تعداد سرویس‌های انجام شده نسبت به تعداد کل سرویس‌های نظارت شده مقایسه می‌شوند. نتایج مقایسه در شکل ۵ برای ۱۰ دوره متوالی گزارش شده است. پس از پایان دوره ۱۰، در مجموع تعداد کل سرویس‌های برنامه‌ریزی شده برای روش پیشنهادی ۵۸۵ است. پس از روش پیشنهادی، GA-FSP با ۵۷۵ سرویس عملکرد بهتری داشته است. علاوه بر این، تعداد سرویس‌های انجام شده قبل از مهلت برای الگوریتم‌های FSP-ODMA، SPP-GWO و CSA-FSP به ترتیب ۵۷۲، ۴۶۱ و ۵۷۲ است.



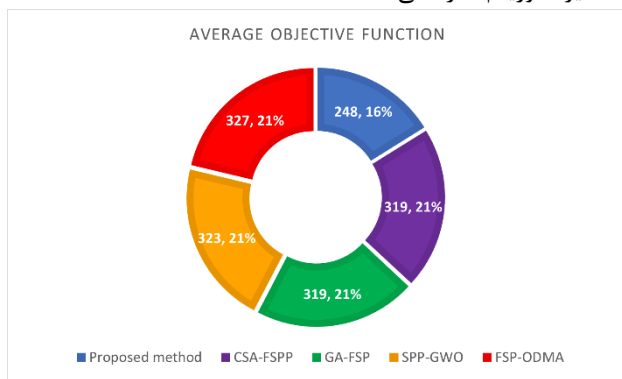
شکل ۵: نتایج بر اساس تعداد سرویس‌های انجام شده قبل از مهلت

شکل ۶ نتایج مقایسه الگوریتم‌های مختلف را بر اساس زمان اجرا در ۱۰ دوره زمانی نشان می‌دهد. هدف بهینه‌سازی یافتن یک برنامه ریزی استقرار مناسب با کمترین زمان اجرا است. به طور کلی، پس از پایان ۱۰ دوره، مجموع زمان اجرای روش پیشنهادی برابر ۷،۵۱ ثانیه است که نسبت به GA-FSP، FSP-ODMA، SPP-GWO و CSA-FSP به ترتیب با ۸،۱۷، ۳۴،۱۱، ۸،۴۳ و ۸،۹۱ ثانیه عملکرد بهتری دارد.

میانگین زمان انتظار سرویس‌های انجام شده برای اجرا روی کلونی مه در شکل ۷ برای ۱۰ دوره نشان داده شده است. زمان انتظار به مجموع تأخیرهای پردازش (تأخیر صف) و زمان برنامه‌ریزی استقرار اشاره دارد. روش پیشنهادی دارای حداقل زمان انتظار برای سرویس‌ها در همه دوره‌ها است. پس از پایان ۱۰ دوره، میانگین زمان انتظار سرویس‌های انجام شده برای روش پیشنهادی برابر ۵،۳۷ ثانیه است که در مقایسه با سایر الگوریتم‌ها کمترین است. پس از روش پیشنهادی، SPP-GWO با ۵،۴۸ ثانیه دارای میانگین زمان انتظار است. نتایج این معیار برای الگوریتم‌های

روش پیشنهادی تعداد سرویس‌های ناموفق کمتری در همه دوره‌ها فراهم کرده‌است. در ضمن، روش پیشنهادی تنها الگوریتمی است که در دوره دوم موفق به برنامه‌ریزی همه سرویس‌های نظارت شده قبل از مهلت شده‌است. به‌طور کلی، پس از پایان ۱۰ دوره، مجموع سرویس‌های ناموفق برای روش پیشنهادی برابر ۲۸ است که نسبت به FSP-ODMA، SPP-GWO، CSA-FSPP و GA-FSP به ترتیب با ۴۱، ۳۶، ۳۷ و سرویس ناموفق عملکرد برتری دارد.

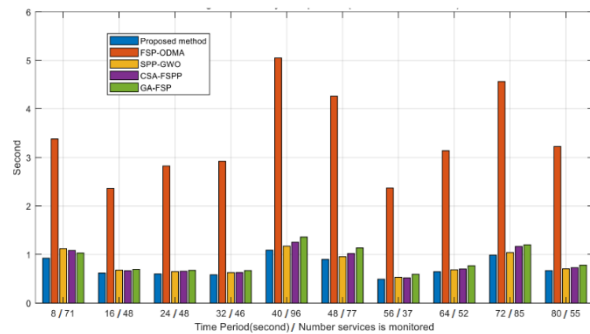
شکل ۹ نتایج تابع برازندگی را برای الگوریتم‌های مختلف در ۱۰ دوره نشان می‌دهد. با توجه به تعریف تابع برازندگی، هدف بهینه‌سازی در الگوریتم‌ها حداقل‌سازی می‌باشد. این متریک بر اساس تعداد سرویس‌های انجام‌شده نسبت به کل سرویس‌های مانیتور شده در نظر گرفته می‌شود. نتایج این مقایسه برای همه دوره‌های زمانی نشان می‌دهد که روش پیشنهادی کارایی بهتری نسبت به سایر الگوریتم‌ها دارد. در پایان ۱۰ دوره زمانی، میانگین تابع هدف برای روش پیشنهادی، FSP-ODMA، SPP-GWO، CSA-FSPP و GA-FSP به ترتیب ۲۴۸، ۳۲۲، ۳۲۷، ۳۱۹، ۴ و ۳۱۹، ۴ است. طبق شکل ۱۰، روش پیشنهادی کمترین مقدار تابع هدف را نسبت به سایر الگوریتم‌ها ارائه می‌دهد.



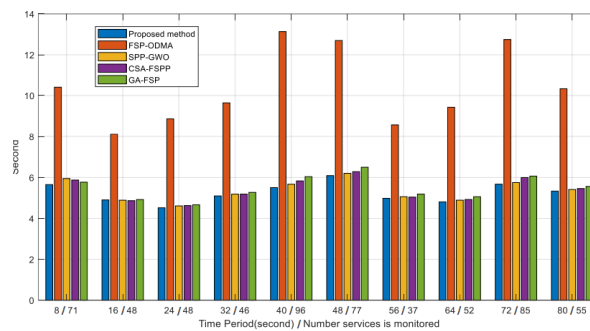
شکل ۱۰: مقایسه نتایج الگوریتم‌های مختلف بر اساس تابع هدف پس از پایان ۱۰ دوره

در آزمایش دیگری، تعداد سرویس‌های باقیمانده ارسالی به دوره یازدهم برای الگوریتم‌های مختلف مقایسه شده‌است و نشان می‌دهد که منابع آزاد گره‌های مه برای اجرای برخی از سرویس‌ها کافی نیست و باید در دوره بعدی اجرا شوند. جدول ۱ نتایج این مقایسه را برای الگوریتم‌های مختلف نشان می‌دهد، جایی که برخی سرویس‌ها باید در دوره یازدهم اجرا شوند. ستون اول به الگوریتم‌ها اشاره دارد، ستون دوم تعداد کل سرویس‌های ارسالی به مه را نشان می‌دهد، ستون سوم مرتبط با تعداد سرویس‌های اجرا شده پس از پایان ۱۰ دوره زمانی است، ستون چهارم به تعداد سرویس‌های ناموفق اشاره دارد و در نهایت ستون آخر نشان می‌دهد که چه تعداد سرویس باقیمانده وجود دارد که باید در دوره ۱۱ برنامه‌ریزی شوند. نتایج این مقایسه نشان می‌دهد که روش پیشنهادی، FSP-ODMA، SPP-GWO و CSA-FSPP با تنها ۲ سرویس باقیمانده نسبت به سایر الگوریتم‌ها عملکرد بهتری دارد. با این حال روش

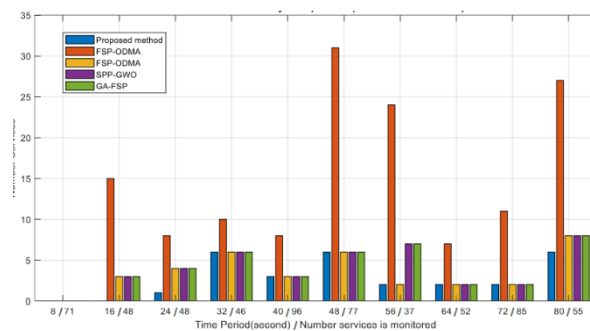
FSP-ODMA، CSA-FSPP و GA-FSP به ترتیب ۱۰، ۵، ۵، ۵، ۴ و ۵، ۶، ۴ گزارش شده‌است.



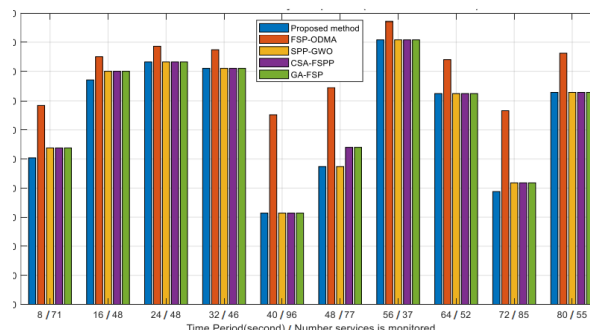
شکل ۶: نتایج بر اساس زمان اجرا قبل از مهلت



شکل ۷: نتایج بر اساس میانگین زمان انتظار قبل از مهلت



شکل ۸: مختلف بر اساس تعداد سرویس‌های ناموفق



شکل ۹: نتایج الگوریتم‌های مختلف بر اساس تابع هدف

تعداد سرویس‌های ناموفق نسبت به کل سرویس‌های مانیتور شده در شکل ۸ برای الگوریتم‌های مختلف آمده‌است. این نتایج برای ۱۰ دوره مستقل و ۱۰۰۰ درخواست است. بر اساس نتایج الگوریتم‌های مختلف،

پیشنهادی در مقایسه با این الگوریتم‌ها موفق‌تر است، زیرا تنها ۲۸ سرویس ناموفق دارد.

جدول ۱: نتایج الگوریتم‌های مختلف بر اساس تعداد سرویس‌های باقیمانده برای برنامه‌ریزی در دوره یازدهم

روش‌ها	مجموع سرویس‌ها	سرویس‌های انجام شده تا دوره ۱۰	تعداد سرویس‌ها ناموفق	سرویس‌های باقیمانده برای دوره یازدهم
GA-FSP	۶۱۵	۵۷۵	۳۷	۳
FSP-ODMA	۶۱۵	۴۶۱	۱۴۱	۱۳
SPP-GWO	۶۱۵	۵۷۲	۳۶	۷
CSA-FSPP	۶۱۵	۵۷۲	۴۱	۲
روش پیشنهادی	۶۱۵	۵۸۵	۲۸	۲

در جدول ۲ نتایج اجرای روش پیشنهادی و الگوریتم‌های GA-FSP، FSP-ODMA، SPP-GWO و CSA-FSPP خلاصه شده است. این نتایج برای متریک‌های مختلف بر اساس ۱۰۰۰ درخواست و ۱۰ دوره زمانی ارائه شده است. متریک‌ها شامل تعداد سرویس‌های انجام شده، زمان اجرا، میانگین زمان انتظار، تعداد سرویس‌های ناموفق، تابع هزینه و تعداد سرویس‌های باقی مانده هستند. نتایج به وضوح برتری روش پیشنهادی به سایر الگوریتم‌ها را نشان می‌دهد. پس از روش پیشنهادی، به ترتیب الگوریتم‌های GA-FSP، CSA-FSPP، SPP-GWO، FSP-ODMA در رتبه‌های بعدی قرار دارند. الگوریتم‌هایی نظیر روش پیشنهادی و FSP-ODMA خیلی کم مهلت برنامه‌ها را نقض می‌کند، اما FSP-ODMA در مقایسه با روش پیشنهادی از منابع ابری بیشتری استفاده می‌کند. استفاده از منابع ابری به دلیل فاصله فیزیکی مرکز داده ابر منجر به هزینه اجرای بیشتر می‌شود، جایی که در نتایج FSP-ODMA مشخص است. به طور میانگین، نتایج روش پیشنهادی نسبت به SPP-GWO، CSA-FSPP و GA-FSP به ترتیب ۴/۶٪، ۲/۴٪، ۳/۴٪ و ۱/۱٪ برتری را گزارش می‌دهد.

جدول ۲: خلاصه از نتایج مقایسه الگوریتم‌های مختلف

متریک ارزیابی	روش پیشنهادی	GA-FSP	FSP-ODMA	SPP-GWO	CSA-FSPP
تعداد سرویس‌های نظارت شده	۶۱۵	۶۱۵	۶۱۵	۶۱۵	۶۱۵
میانگین زمان اجرا	۷/۵	۸/۱۷	۳۴/۱۱	۸/۴۳	۸/۹۱
تعداد سرویس‌های انجام شده (%)	۹۵/۱۲	۹۳/۴۹	۷۴/۹۶	۹۳/۰۱	۹۳/۰۱
میانگین زمان انتظار	۵/۳۶	۵/۶۴	۱۰/۷۴	۵/۴۸	۵/۵۴
سرویس‌های ناموفق (%)	۴/۵۵	۶/۰۲	۲۲/۹۳	۶/۶۷	۴/۵۵
سرویس‌های باقیمانده (%)	۰/۳۳	۰/۴۸	۲/۱۱	۱/۱۴	۰/۳۳
هزینه سرویس	۲۴۸	۳۱۹	۳۲۷	۳۲۳	۳۱۹

۵. نتیجه‌گیری

در این تحقیق، یک چارچوب محاسبات مفهومی کارآمد مبتنی بر میان-افزار کنترل ابر-مه به منظور مکان‌یابی مکان‌یابی سرویس‌های IoT در لایه مه معرفی شد. مسئله مکان‌یابی سرویس‌ها در مه به عنوان یک مسئله بهینه‌سازی رسمی شد که هدف آن کاهش تأخیر، کاهش هزینه و حداکثرسازی استفاده از مه است. این چارچوب از مدل MADE-k (نظارت، تجزیه و تحلیل، تصمیم‌گیری، اجرا - پایگاه دانش) برای حل مسئله استفاده می‌کند. این مدل در میان‌افزار کنترل ابر-مه پیگیری شده و وظیفه مدیریت کل کلونی‌های مه را برعهده دارد، جایی که هر کلونی شامل تعدادی گره مه و یک گره کنترل اراکستراسیون مه است. در مرحله نظارت، همه سرویس‌های IoT و گره‌های مه ارزیابی شده و جزئیاتشان در پایگاه دانش ذخیره می‌شود.

اولویت‌بندی زمان اجرا سرویس‌های IoT در مرحله تجزیه و تحلیل انجام می‌شود. فرایند مکان‌یابی با اعمال رویکرد یادگیری تقویتی به صورت خودکار در مرحله تصمیم‌گیری انجام شده و سپس طرح مکان‌یابی در مرحله اجرا روی چشم‌انداز اعمال می‌گردد. نتایج به وضوح برتری روش پیشنهادی را نشان می‌دهد. پس از روش پیشنهادی، به ترتیب الگوریتم‌های GA-FSP، CSA-FSPP، SPP-GWO، FSP-ODMA در رتبه‌های بعدی قرار دارند. الگوریتم‌هایی نظیر روش پیشنهادی و FSP-ODMA خیلی کم مهلت برنامه‌ها را نقض می‌کند، اما FSP-ODMA در مقایسه با روش پیشنهادی از منابع ابری بیشتری استفاده می‌کند. استفاده از منابع ابری به دلیل فاصله فیزیکی مرکز داده ابر منجر به هزینه اجرای بیشتر می‌شود، جایی که در نتایج FSP-ODMA مشخص است. به طور میانگین، نتایج روش پیشنهادی نسبت به SPP-GWO، CSA-FSPP و GA-FSP به ترتیب ۴/۶٪، ۲/۴٪، ۳/۴٪ و ۱/۱٪ برتری را گزارش می‌دهد. به طور کلی، از مقایسه روش پیشنهادی با سایر الگوریتم‌ها می‌توان نتیجه گرفت که:

- راه‌حل‌های روش پیشنهادی به دلیل استراتژی نگهداری بهتر تنوع قابل توجهی نسبت به راه‌حل‌های سایر الگوریتم‌ها دارند.
- روش پیشنهادی از توزیع‌پذیری بهتری برخوردار است که به همگرایی و کاهش زمان اجرا کمک می‌کند.
- روش پیشنهادی در فضاهای هدف نسبت به سایر الگوریتم‌ها به خصوص GA-FSP بهتر عمل می‌کند و از مقیاس‌پذیری بهتری برخوردار است.

در ادامه یافتن یک سیاست چندهدفه بر مبنای آرشیو پارتو به عنوان کارهای آتی پیشنهاد می‌شود. همچنین، تحلیل الگوریتم‌های ابتکاری موازی برای یافتن یک مکان‌یابی دقیق‌تر نسبت به رویکردهای تکاملی جنبه دیگری از کارهای آینده است. به عنوان کار آینده، قصد داریم رویکردهای یادگیری تقویتی جدید نظیر الگوریتم عملگر-نقاد مزیت ناهمگن (A3C) را به همراه سیاست حداکثرسازی پاداش تجمعی بلندمدت برای مکان‌یابی سرویس‌ها در نظر بگیریم. همچنین، تلاش‌های

- [14] Sami, H., Mourad, A., Otrok, H., & Bentahar, J. (2020, June). Fscaler: Automatic resource scaling of containers in fog clusters using reinforcement learning. In *2020 international wireless communications and mobile computing (IWCMC)* (pp. 1824-1829). IEEE.
- [15] Suryadevara, N. K. (2021). Energy and latency reductions at the fog gateway using a machine learning classifier. *Sustainable Computing: Informatics and Systems*, 31, 100582.
- [16] Gazori, P., Rahbari, D., & Nickray, M. (2020). Saving time and cost on the scheduling of fog-based IoT applications using deep reinforcement learning approach. *Future Generation Computer Systems*, 110, 1098-1115.
- [17] Guerrero, C., Lera, I., & Juiz, C. (2019). Evaluation and efficiency comparison of evolutionary algorithms for service placement optimization in fog architectures. *Future Generation Computer Systems*, 97, 131-144.
- [18] Aburukba, R. O., AliKarrar, M., Landolsi, T., & El-Fakih, K. (2020). Scheduling Internet of Things requests to minimize latency in hybrid Fog-Cloud computing. *Future Generation Computer Systems*, 111, 539-551.
- [19] Salimian, M., Ghobaei-Arani, M., & Shahidinejad, A. (2021). Toward an autonomic approach for Internet of Things service placement using gray wolf optimization in the fog computing environment. *Software: Practice and Experience*, 51(8), 1745-1772.
- [20] Skarlat, O., Schulte, S., Borkowski, M., & Leitner, P. (2016, November). Resource provisioning for IoT services in the fog. In *2016 IEEE 9th international conference on service-oriented computing and applications (SOCA)* (pp. 32-39). IEEE.
- [21] Goudarzi, M., Wu, H., Palaniswami, M., & Buyya, R. (2020). An application placement technique for concurrent IoT applications in edge and fog computing environments. *IEEE Transactions on Mobile Computing*, 20(4), 1298-1311.
- [22] Goudarzi, M., Palaniswami, M. S., & Buyya, R. (2021). A distributed deep reinforcement learning technique for application placement in edge and fog computing environments. *IEEE Transactions on Mobile Computing*.
- [23] Murtaza, F., Akhuzada, A., ul Islam, S., Boudjadar, J., & Buyya, R. (2020). QoS-aware service provisioning in fog computing. *Journal of Network and Computer Applications*, 165, 102674.
- [24] Faraji Mehmandar, M., Jabbehdari, S., & Haj Seyyed Javadi, H. (2020). A dynamic fog service provisioning approach for IoT applications. *International Journal of Communication Systems*, 33(14), e4541.
- [25] Skarlat, O., Nardelli, M., Schulte, S., Borkowski, M., & Leitner, P. (2017). Optimized IoT service placement in the fog. *Service Oriented Computing and Applications*, 11(4), 427-443.
- [26] Khosroabadi, F., Fotouhi-Ghazvini, F., & Fotouhi, H. (2021). SCATTER: Service Placement in Real-Time Fog-Assisted IoT Networks. *Journal of Sensor and Actuator Networks*, 10(2), 26.
- [27] Liu, C., Wang, J., Zhou, L., & Rezaeipanah, A. (2022). Solving the Multi-Objective Problem of IoT Service Placement in Fog Computing Using Cuckoo Search
- آینده، رویکردهای یادگیری تقویتی را برای بازیابی شکست به سمت معماری ابر-مه-IoT بررسی خواهد کرد که می‌توان در آن، معماری پردازش موازی سرویس‌های IoT در فرایند مکان‌یابی را لحاظ کرد.
- ### مراجع
- [1] Hassan, H. O., Azizi, S., & Shojafar, M. (2020). Priority, network and energy-aware placement of IoT-based application services in fog-cloud environments. *IET communications*, 14(13), 2117-2129.
- [2] Brogi, A., Forti, S., & Ibrahim, A. (2018, March). Optimising QoS-assurance, resource usage and cost of fog application deployments. In *International Conference on Cloud Computing and Services Science* (pp. 168-189). Springer, Cham.
- [3] Chen, Y., Li, Z., Yang, B., Nai, K., & Li, K. (2020). A Stackelberg game approach to multiple resources allocation and pricing in mobile edge computing. *Future Generation Computer Systems*, 108, 273-287.
- [4] Dastjerdi, A. V., & Buyya, R. (2016). Fog computing: Helping the Internet of Things realize its potential. *Computer*, 49(8), 112-116.
- [5] Ghobaei-Arani, M., Souri, A., Safara, F., & Norouzi, M. (2020). An efficient task scheduling approach using moth-flame optimization algorithm for cyber-physical system applications in fog computing. *Transactions on Emerging Telecommunications Technologies*, 31(2), e3770.
- [6] Goswami, P., Mukherjee, A., Maiti, M., Tyagi, S. K. S., & Yang, L. (2021). A Neural Network Based Optimal Resource Allocation Method for Secure IoT Network. *IEEE Internet of Things Journal*.
- [7] Askarzadeh, A. (2016). A novel metaheuristic method for solving constrained engineering optimization problems: crow search algorithm. *Computers & Structures*, 169, 1-12.
- [8] Ibrahim, A., Noshay, M., Ali, H. A., & Badawy, M. (2020). PAPS0: A power-aware VM placement technique based on particle swarm optimization. *IEEE Access*, 8, 81747-81764.
- [9] Jia, B., Hu, H., Zeng, Y., Xu, T., & Yang, Y. (2018). Double-matching resource allocation strategy in fog computing networks based on cost efficiency. *Journal of Communications and Networks*, 20(3), 237-246.
- [10] Karatas, F., & Korpeoglu, I. (2019). Fog-based data distribution service (F-DAD) for Internet of Things (IoT) applications. *Future Generation Computer Systems*, 93, 156-169.
- [11] Etemadi, M., Ghobaei-Arani, M., & Shahidinejad, A. (2020). Resource provisioning for IoT services in the fog computing environment: An autonomic approach. *Computer Communications*, 161, 109-131.
- [12] Nayeri, Z. M., Ghafarian, T., & Javadi, B. (2021). Application placement in Fog computing with AI approach: Taxonomy and a state of the art survey. *Journal of Network and Computer Applications*, 185, 103078.
- [13] Yan, S., Jiao, M., Zhou, Y., Peng, M., & Daneshmand, M. (2020). Machine-learning approach for user association and content placement in fog radio access networks. *IEEE Internet of Things Journal*, 7(10), 9413-9425.

Algorithm. Neural Processing Letters.
<https://doi.org/10.1007/s11063-021-10708-2>.

- [28] Zhao, D., Zou, Q., & Boshkani Zadeh, M. (2022). A QoS-Aware IoT Service Placement Mechanism in Fog Computing Based on Open-Source Development Model. *Journal of Grid Computing*, 20(2), 1-29.
- [29] Ghobaei-Arani, M., & Shahidinejad, A. (2022). A cost-efficient IoT service placement approach using whale optimization algorithm in fog computing environment. *Expert Systems with Applications*, 200, 117012.
- [30] Jayadeva, S. M., Al Ayub Ahmed, A., Malik, R., Shaikh, A. A., Siddique, M., & Naved, M. (2023). Roles of Cloud Computing and Internet of Things in Marketing Management: A Critical Review and Future Trends. In *Proceedings of Second International Conference in Mechanical and Energy Technology* (pp. 165-173). Springer, Singapore.
- [31] Kaelbling, L. P., Littman, M. L., & Moore, A. W. (1996). Reinforcement learning: A survey. *Journal of artificial intelligence research*, 4, 237-285.
- [32] Natesha, B. V., & Guddeti, R. M. R. (2021). Adopting elitism-based Genetic Algorithm for minimizing multi-objective problems of IoT service placement in fog computing environment. *Journal of Network and Computer Applications*, 178, 102972.