JOURNAL OF SOUTHERN COMMUNICATION ENGINEERING
ISLAMIC AZAD UNIVERSITY BUSHEHR BRANCH

E-ISSN: 2980-9231
https://jce.bushehr.iau.ir

https://doi.org/10.30495/jce.2023.1975916.1186

Vol. 13/ No. 50/Winter 2024

Research Article

An Autonomous Planning Model for Deploying IoT Services in Fog Computing

Mansoureh Zare, PhD Student [1] | Yasser Elmi Sola, Assistant Professor[2*] | Hesam Hasanpour, Assistant Professor[3]

[1]Department of Information Technology and Computer Engineering, Sabzevar Branch, Islamic Azad University, Sabzevar, Iran,
Soori.zare67@gmail.com

[2]Department of Information Technology and Computer Engineering, Sabzevar Branch, Islamic Azad University, Sabzevar, Iran,
Elmi@iaus.ac.ir

[3]Department of Information Technology and Computer Engineering, Sabzevar Branch, Islamic Azad University, Sabzevar, Iran,
Hesam_78@yahoo.com

Correspondence
Yasser Elmi Sola, Assistant Professor of Information Technology and Computer Engineering, Sabzevar Branch, Islamic Azad University, Sabzevar, Iran,
Elmi@iaus.ac.ir

Received: 25 December 2022
Revised: 11 January 2023
Accepted: 23 January 2023

Abstract

IoT-based devices are constantly sending data to the cloud. However, the centralization of cloud data centers and the long distance to the location of data creation has reduced the efficiency of this paradigm in real-time applications. Fog computing can provide the resources needed by Internet of Things devices in a distributed manner at the edge of the network without involving the cloud. Therefore, processing, analysis and storage are closer to the place of data creation and end users, and the delay is reduced. Every Internet of Things program includes a set of Internet of Things services with different quality of service requirements, whose required resources can be provided by deploying on cloud nodes. This study deals with the challenge of locating Internet of Things services as an autonomous planning model in fog computing. We develop the colonial competition algorithm as a meta-heuristic approach to solve this problem. Since fog nodes with enough resources can host several IoT services, we consider resource distribution in the localization process. The proposed algorithm prioritizes Internet of Things services to reduce delay and solves the multi-objective positioning problem. The results of the experiments show that our algorithm can effectively improve the performance of the system and have 15% to 31% better effectiveness than the best results of the advanced algorithms in the literature.

Keywords: Fog Computing, Autonomous Planning Model, IoT services, Metaheuristic approach.

Highlights

- Development of ICA as a meta-heuristic approach to improve the SPP solution.
- Development of the SPP problem as a multi-objective optimization problem.
- Considering the distribution of using fog nodes for resource management in future requests.

Citation: M. Zare, Y. Elmi sola, and H. Hasanpour, "An Autonomous Planning Model for Deploying IoT Services In Fog Computing," *Journal of Southern Communication Engineerng*, vol. 13, no. 50, pp. 85–100, 2023, doi: 10.30495/jce.2023.1975916.1186 (in Persian).

مقاله پژوهشی

یک مدل برنامه‌ریزی خودمختار برای استقرار خدمات اینترنت اشیا در محاسبات مه: یک رویکرد مبتنی بر فرا ابتکاری

منصوره زارع [۱] | یاسر علمی سولا *[۲] | حسام حسن پور [۳]

[۱] گروه کامپیوتر و فناوری اطلاعات، واحد سبزوار، دانشگاه آزاد اسلامی، سبزوار، ایران،
Soori.zare67@gmail.com

[۲] گروه کامپیوتر و فناوری اطلاعات، واحد سبزوار، دانشگاه آزاد اسلامی، سبزوار، ایران،
Elmi@iaus.ac.ir

[۳] گروه کامپیوتر و فناوری اطلاعات، واحد سبزوار، دانشگاه آزاد اسلامی، سبزوار، ایران،
Hesam_78@yahoo.com

نویسنده مسئول
*یاسر علمی سولا، استادیار، گروه کامپیوتر و فناوری اطلاعات، واحد سبزوار، دانشگاه آزاد اسلامی، سبزوار، ایران،
Elmi@iaus.ac.ir

تاریخ دریافت: ٤ دی ١٤٠١
تاریخ بازنگری: ٢١ دی ١٤٠١
تاریخ پذیرش: ٣ بهمن ١٤٠١

https://doi.org/10.30495/jce.2023.1975916.1186

چکیده:

دستگاه‌های مبتنی بر اینترنت اشیاء دائماً در حال ارسال داده به ابر هستند. بااین‌حال، متمرکز بودن مراکز داده ابر و فاصله زیاد تا محل ایجاد داده‌ها باعث کاهش کارایی این پارادایم در کاربردهای زمان واقعی شده است. محاسبات مه می‌تواند بدون درگیر کردن ابر منابع موردنیاز دستگاه‌های اینترنت اشیاء را به‌صورت توزیع‌شده در لبه شبکه فراهم کند؛ بنابراین، پردازش، تجزیه‌وتحلیل و ذخیره‌سازی به محل ایجاد داده‌ها و کاربران نهایی نزدیک شده و تأخیر کاهش می‌یابد. هر برنامه اینترنت اشیاء شامل مجموعه از سرویس‌های اینترنت اشیاء با الزامات مختلف کیفیت سرویس است که منابع موردنیاز آن‌ها می‌توانند با استقرار روی گره‌های مه فراهم شوند. این مطالعه به چالش مکان‌یابی سرویس‌های اینترنت اشیاء به‌عنوان یک مدل برنامه-ریزی خودمختار در محاسبات مه می‌پردازد. ما الگوریتم رقابت استعماری را به‌عنوان یک رویکرد مبتنی بر فرا ابتکاری برای حل این مسئله توسعه می‌دهیم. ازآنجاکه گره‌های مه با داشتن منابع کافی می‌توانند میزبان چندین سرویس اینترنت اشیاء باشند، ما توزیع منابع را در فرایند مکان‌یابی در نظر می‌گیریم. الگوریتم پیشنهادی سرویس‌های اینترنت اشیاء را برای کاهش تأخیر اولویت‌بندی کرده و مسئله مکان‌یابی را به‌صورت چند-هدفه حل می‌کند. نتایج آزمایش‌ها نشان می‌دهد که الگوریتم ما می‌تواند به‌طور مؤثر عملکرد سیستم را بهبود داده و از بهترین نتایج الگوریتم‌های پیشرفته ادبیات بین ۱۵٪ تا ۳۱٪ اثربخشی بهتری داشته باشد.

کلید واژه‌ها: محاسبات مه، مدل خودمختار، مکان‌یابی خدمات اینترنت اشیاء، رویکرد فرا ابتکاری.

۱-مقدمه

منابع محاسباتی راه دور مختلفی نظیر رایانش مشبک، رایانش ابر و رایانش مه منجر به ظهور دستگاه‌های اینترنت اشیاء[1] (IoT) شده است [۱]. مفهوم اینترنت اشیا به‌عنوان یک دستگاه هوشمند متصل به اینترنت در اوایل دهه ۲۰۰۰ معرفی شد و امروزه میلیاردها از آن در سراسر جهان به جمع‌آوری و تبادل داده می‌پردازند [۲]. توسعه روزافزون اینترنت و فناوری محاسبات کامپیوتری باعث محبوبیت سرویس‌های آنلاین شده است. کاربران نهایی می‌توانند برای پردازش داده‌های تولیدشده توسط دستگاه‌های اینترنت اشیا از سرویس‌های آنلاین در منابع محاسباتی راه دور استفاده کنند. این باعث کاهش قدرت پردازش،

[1] Internet of Things

هزینه، مصرف حافظه و مصرف انرژی برای کاربران نهایی می‌شود [۲، ۳]. در این راستا، پارادایم رایانش ابری می‌تواند سرویس‌های دستگاه‌های اینترنت اشیا را بدون نگرانی از قدرت پردازش، مصرف حافظه، هزینه و مصرف انرژی فراهم کند [۳].

با گسترش تعداد دستگاه‌های اینترنت اشیا تا ۴۶ میلیارد در سال ۲۰۲۱ و افزایش آن تا ۷۵ میلیارد در سال ۲۰۲۵، سرعت و کیفیت انتقال داده‌ها بیش‌ازپیش حائز اهمیت شده است. امروزه، برنامه‌های اینترنت اشیا مبتنی بر پردازش داده‌های زمان واقعی بسیار رایج شده‌اند [۴]. رایانش ابری با معماری متمرکز برای پردازش این برنامه‌ها با مشکل تأخیر زیاد مواجه است، زیرا سرورهای ابری از مبدأ داده بسیار دور هستند [۱]. علاوه بر تأخیر، رایانش ابری به دلیل حجم عظیم داده‌های تولیدشده توسط دستگاه‌های اینترنت اشیا با چالش‌هایی نظیر ازدحام در شبکه، پهنای باند و زمان پاسخ روبه‌رو است [۴]. از اثبات عدم کارایی زیرساخت فعلی ابر، پارادایم جدید به نام رایانش مه تکامل یافت [۵]. رایانش مه می‌تواند سرویس‌هایی نظیر ذخیره‌سازی و پردازش را در لبه شبکه و نزدیک به کاربر نهایی فراهم کند. ایده رایانش مه استفاده کارآمد از این دستگاه‌های اینترنت اشیا است که می‌تواند ارائه سرویس‌ها را به مبدأ داده نزدیک کند [۵]؛ بنابراین، رایانش مه راه‌حل‌هایی باکیفیت خدمات بهتری در مقایسه با رایانش مه فراهم می‌کند. علاوه بر این، رایانش مه با کاهش قابل‌توجه ارسال تعداد درخواست‌ها به مراکز داده ابری باعث کاهش مصرف پهنای باند و کاهش ازدحام در شبکه می‌شود [۶، ۷].

مفهوم رایانش مه به‌عنوان زیرساختی محلی و غیرمتمرکز برای مدیریت درخواست‌های اینترنت اشیا ارائه‌شده است [۸]. اجزای اصلی محاسبات مه، دستگاه‌های اینترنت اشیا (کاربران نهایی) و گره‌های مه (دستگاه‌های ناهمگن و غیرمتمرکز) هستند. دستگاه‌های اینترنت اشیا می‌تواند هر دستگاه هوشمند متصل به اینترنت نظیر ساعت هوشمند، موبایل هوشمند، خودرو هوشمند باشد و گره‌های مه می‌توانند دستگاه‌هایی با منابع محدود نظیر سوئیچ‌ها، مسیریاب‌ها و ستاپ‌باکس‌ها باشند. گره‌های مه دارای منابع محاسباتی و ذخیره‌سازی محدودی هستند که می‌تواند میزبان سرویس‌های اینترنت اشیا باشند. هر برنامه اینترنت اشیا شامل تعدادی سرویس مستقل است که منابع موردنیاز آن‌ها می‌تواند با توسط هر گره مه در دسترس فراهم شود [۸]. مهم‌ترین چالش در تحقق رایانش مه تخصیص کارآمد گره‌های مه به سرویس‌های اینترنت اشیا است که به‌عنوان مسئله مکان‌یابی سرویس[1] (SPP) در رایانش مه شناخته می‌شود [۱].

با توجه به توزیع جغرافیایی گره‌های مه، پیچیدگی حل مشکل استقرار سرویس باهدف بهبود کیفیت سرویس یک مسئله ان پی سخت است [۳-۵]. مشکل استقرار سرویس‌ها به دلیل غیرمتمرکز منابع مه و پویا بودن برنامه‌های اینترنت اشیا دریافتی بسیار حائز اهمیت است. در اینجا، هر سرویس با الزامات مختلف کیفیت سرویس برای اجرا باید روی یک گره مه با منابع محدود مستقر شود. مدل نظارت، تجزیه‌وتحلیل، تصمیم‌گیری و اجرا با پایگاه دانش مشترک[2] (MADE-k) یکی از مدل‌های معروف محاسبات خودمختار است که توسط شرکت IBM معرفی شد [۹]. MADE-k می‌تواند مشکل استقرار سرویس را برای تصمیم-گیری در مورد استقرار سرویس‌های اینترنت اشیا فرموله سازی کند. در این مقاله، الگوریتم رقابت استعماری[3] (ICA) به‌عنوان یک الگوریتم مبتنی بر رویکردهای فرا ابتکاری برای حل کارآمد مشکل استقرار سرویس‌ها استفاده می‌شود. بر این اساس، ما الگوریتمی به نام SPP-ICA برای حل مشکل استقرار سرویس‌ها با استفاده از الگوریتم ICA پیشنهاد می‌دهیم.

مشارکت‌های اصلی این کار را می‌توان به شرح زیر خلاصه کرد:

- توسعه ICA به‌عنوان یک رویکرد فرا ابتکاری برای بهبود حل مشکل استقرار سرویس‌ها
- توسعه مسئله استقرار سرویس‌ها به‌عنوان یک مسئله بهینه‌سازی چند-هدفه
- در نظر گرفتن توزیع استفاده از گره‌های مه برای مدیریت منابع در درخواست‌های آینده

ادامه مقاله به شرح زیر سازمان‌دهی شده است: بخش ۲ به کارهای مرتبط می‌پردازد. بخش ۳ مدل‌سازی مسئله را ارائه می‌دهد. بخش ۴ الگوریتم SPP-ICA پیشنهادی توصیف می‌شود. بحث در مورد شبیه‌سازی و مقایسه‌ها در بخش ۵ آمده است و درنهایت، بخش ۶ این مقاله را به پایان می‌رساند.

[1] Service Placement Problem
[2] Monitoring, Analysis, Decision-making, and Execution with a shared knowledge
[3] Imperialist Competitive Algorithm

۲- ادبیات تحقیق

تاکنون، مطالعات گسترده‌ای روی مشکل استقرار سرویس‌ها در رایانش مه انجام‌شده است و اهمیت مسئله توسط جامعه پژوهشی درک شده است [۱۰، ۱۱]. برخلاف ابر، تحقیقات در مه همچنان نابالغ است و هنوز چالش‌ها و کمبودهایی در تحقیقات انجام‌شده وجود دارد. محققان زیادی نظیر ناتاشا و گودتی [۱۲] با در نظر گرفتن اهداف مختلف برنامه‌ریزی استقرار سرویس‌ها را انجام دادند، بااین‌حال در این تحقیقات کمیت ارتباط بین اهداف لحاظ نشده است. به‌طورکلی، مصالحه بین اهداف و محدودیت‌های مختلف در مشکل استقرار سرویس‌ها می‌تواند به بهبود عملکرد کلی سیستم منجر شود [۱۱]. فاکتورهای زیادی برای تحلیل الگوریتم‌های مرتبط با مشکل استقرار سرویس‌ها وجود دارد که می‌توان به تکنیک حل (آفلاین یا آنلاین)، الگو مکان‌یابی (متمرکز یا توزیع‌شده)، متحرک بودن (عدم پشتیبانی، پشتیبانی)، نوع سیستم (ایستا یا پویا) و نوع مدل (فرا ابتکاری یا یادگیری تقویتی) و تعداد اهداف (تک-هدفه یا چند-هدفه) اشاره داشت [۱۳].

جیا و همکاران [۱۴] از یک استراتژی تطبیق دوگانه[1] (DMS) برای بهبود مدیریت تخصیص منابع مقرون‌به‌صرفه در رایانش مه استفاده کردند. DMS یک نسخه توسعه‌یافته از الگوریتم پذیرش معوق از تطبیق دوطرفه به تطبیق سه‌طرفه است. یوسف‌پور و همکاران [۱۵] یک چارچوب ارائه سرویس مه پویا با QoS-آگاه سبک‌وزن[2] (QDFSP) را برای حل مشکل استقرار سرویس‌ها معرفی کردند. نویسندگان از دو الگوریتم ابتکاری به‌منظور کاهش پیچیدگی حل مسئله استفاده کردند، جائیکه اولی نیازهای تأخیر را ارضا می‌کند و دومی به دنبال به حداقل رساندن هزینه است. چن و همکاران [۱۶] یک چارچوب مبتنی بر بازی استکلبرگ برای مکان‌یابی سرویس‌ها در محاسبات مه معرفی کردند که تعادل را در تخصیص منابع ارائه می‌دهد. در این روش یک کاربر نهایی با منابع غیرفعال می‌تواند نقش یک ابر لبه متحرک را ایفا کند.

سلیمیان و همکاران [۱۷] الگوریتم SPP-GWO را برای حل مشکل استقرار سرویس‌ها پیشنهاد دادند که از بهینه‌سازی گرگ خاکستری[3] (GWO) برای انجام مکان‌یابی استفاده می‌کند. SPP-GWO با در نظر گرفتن هزینه سرویس و تفکیک منابع مه به‌صورت بلوک یک سیستم مدیریت منابع خودمختار ارائه می‌دهد که با رفتار دینامیکی محیط مه سازگار است. سلیمیان و همکاران [۱۸] برای حل مشکل استقرار سرویس‌ها الگوریتم بهینه‌سازی ازدحام ذرات[4] (PSO) را پیشنهاد دادند که برخی محدودیت‌ها و ناهمگنی برنامه‌ها و منابع را در مکان‌یابی در نظر می‌گیرد. این الگوریتم با نام SPP-PSO از مدل MADE-k برای مدیریت منابع استفاده می‌کند و اولویت‌بندی درخواست‌ها را بر اساس مهلت در نظر می‌گیرد. هدف SPP-PSO مصالحه میان هزینه، تأخیر و استفاده از منابع است.

لیو و همکاران [۱۹] رویکرد CSA-FSPP را پیشنهاد دادند که از الگوریتم جستجوی فاخته[5] (CSA) برای حل مسئله مکان‌یابی سرویس مه (FSPP) استفاده می‌کند. CSA-FSPP 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، FSPP به‌عنوان یک مسئله بهینه‌سازی چند-هدفه مبتنی بر آرشیو پارتو در نظر گرفته‌شده است. ژائو و همکاران [۲۰] یک رویکرد مکان‌یابی کیفیت سرویس-آگاه برای حداقل سازی زمان پاسخ، تأخیر، مصرف انرژی، هزینه سرویس و حداکثر سازی استفاده از منابع مه پیشنهاد دادند. نویسندگان از الگوریتم مدل توسعه منبع باز[6] (ODMA) به‌عنوان یک رویکرد فرا ابتکاری برای مکان‌یابی سرویس مه استفاده کردند (FSP-ODMA). این رویکرد از یک چارچوب محاسبات مفهومی سه لایه برای توصیف تعاملات بین اجزاء سیستم استفاده می‌کند. FSP-ODMA هزینه استقرار برنامه‌های اینترنت اشیا را به‌طور میانگین تا ۱۰٪ نسبت به رویکردهای فرا ابتکاری مشابه کاهش می‌دهد.

قبایی‌آرانی و شهیدی‌نژاد [۲۱] یک رویکرد مقرون‌به‌صرفه با استفاده از الگوریتم بهینه‌سازی نهنگ[7] (WOA) برای مکان‌یابی سرویس مه پیشنهاد دادند (WOA-FSP). WOA-FSP یک رویکرد مکان‌یابی خودمختار مبتنی بر مدل MADE-k است که

[1] double matching strategy
[2] lightweight QoS-aware Dynamic Fog Service Provisioning
[3] Gray Wolf Optimization
[4] Particle Swarm Optimization
[5] Cuckoo Search Algorithm
[6] Open-source Development Model Algorithm
[7] Whale Optimization Algorithm

مشکل استقرار سرویس‌ها را با در نظر گرفتن مصرف انرژی و توان عملیاتی در حین ارضا الزامات کیفیت سرویس حل می‌کند. عظیم زاده و همکاران [۲۲] رویکردی برای حل مسئله مکان‌یابی سرویس مه بر اساس ویژگی شبکه‌های پیچیده[1] (FSPCN) معرفی کردند. FSPCN با در نظر گرفتن مفهوم جامعه سعی در کاهش پیچیدگی مشکل استقرار سرویس‌ها دارد. نویسندگان برای ایجاد جوامع از ساختار شبکه و گره‌ها و ویژگی‌های پیوندها استفاده می‌کنند تا استفاده از منابع و تأخیر را بهبود دهند. FSPCN یک رویکرد مبتنی بر الگوریتم ژنتیک[2] (GA) است که از یک متریک فاصله همسایگی جدید برای ایجاد جوامع و اولویت‌بندی آن‌ها استفاده می‌کند.

۳- مدل‌سازی مسئله

به‌منظور مدیریت کارآمد، زیرساخت مه به‌صورت محلی توسط کلونی‌های مه تفکیک می‌شود [۱۸]. بنابراین، بر اساس محدوده‌های جغرافیایی و دامنه‌های غیر همپوشان می‌توان کلونی‌های مه را تشکیل داد. هر کلونی مه از چندین گره مه به همراه یک گره کنترل ارکستراسیون مه تشکیل می‌شود. هر گره مه دارای یک قدرت محاسباتی و منابع محدود برای اجرای درخواست اینترنت اشیا است. همچنین، مدیریت کلونی مه تابع توسط گره کنترل ارکستراسیون مه انجام می‌شود. اجاره دهید $F=\lfloor f_1, f_2,..., f_i,..., f_{|F|}\rfloor$ مجموعه کلونی‌های مه در دسترس باشد که در آن f_i کلونی مه i-ام و $|F|$ تعداد کل کلونی‌ها است. همچنین، اجازه دهید $f_i=\lfloor o_i, n_i^1, n_i^2,..., n_i^j,..., n_i^{|f_i|}\rfloor$ مجموعه گره‌های تابع در کلونی f_i باشد که در آن o_i گره کنترل ارکستراسیون مه، n_n^j گره مه غیرمتمرکز j-ام در f_i و $|f_i|$ تعداد گره‌های مه فعال در f_i است. برای سهولت، ما جزئیات یک کلونی مه را به‌صورت $f=\lfloor o, n^1, n^2,..., n^j,..., n^{|f|}\rfloor$ نشان می‌دهیم که o گره کنترل ارکستراسیون مه، n^j گره مه j-ام و $|f|$ تعداد گره‌های مه فعال هستند.

علاوه بر این، اجازه دهید $A=\lfloor a_1, a_2,..., a_k,..., f_{|A|}\rfloor$ مجموعه برنامه‌های اینترنت اشیا دریافت شده توسط f در یک دوره زمانی باشد که در آن a_k برنامه k-ام و $|A|$ تعداد برنامه‌ها است. در مشکل استقرار سرویس هر برنامه اینترنت اشیا از چندین سرویس اینترنت اشیا مستقل تشکیل شده است که هر سرویس باید برای اجرا روی یک گره مه مستقر شود. بنابراین، اجازه دهید $a_k=\lfloor s_k^1, s_k^2,..., s_k^l,..., s_k^{|a_k|}\rfloor$ مجموعه سرویس‌های متعلق به a_k باشد که در آن s_k^l سرویس $l-$ام در a_k و $|a_k|$ تعداد سرویس‌های a_k است. با توجه به اینکه سرویس‌های یک برنامه اینترنت اشیا غیروابسته هستند می‌تواند مجموعه کل سرویس‌ها را به‌صورت $S=\lfloor s_1, s_2,..., s_l,..., s_{|S|}\rfloor$ نشان داد، جائیکه S_l سرویس $l-$ام و $|S|$ تعداد کل سرویس‌های دریافت شده در دوره زمانی τ توسط f است. هر $s_l\in S$ دارای نیازهای پردازنده، حافظه، ذخیره‌سازی و مهلت زمانی برای اجرا است که به ترتیب با U_{s_l}، M_{s_l}، S_{s_l} و D_{s_l} نشان داده می‌شود. علاوه بر این، اجازه دهید U_o و U_{n^j} به ترتیب میزان استفاده از پردازنده توسط گره کنترل ارکستراسیون مه و گره n^j باشد. به‌طور مشابه، M_o و M_{n^j} به‌میزان استفاده از حافظه و S_o و S_{n^j} به ظرفیت ذخیره‌سازی استفاده شده اشاره دارند.

همه اجزای سیستم دارای پیوندهای ارتباطی دوجهته همراه با یک تأخیر ناچیز هستند. اجازه دهید $d_o^{n^j}$، d_o^{mw}، d_o^{IOT} و d_o^{nn} تأخیر گره کنترل ارکستراسیون مه به ترتیب در ارتباط با گره n^j، میان‌افزار کنترل ابر-مه، دستگاه‌های اینترنت اشیا و نزدیک‌ترین کلونی مه همسایه باشد. به‌طورکلی، پردازنده، حافظه و ذخیره‌سازی گره‌های مه محدود است و در مکان‌یابی سرویس‌ها باید این محدودیت در نظر گرفته شود. بنابراین، مجموع منابع موردنیاز سرویس‌های مستقر شده روی گره n^j نباید از منابع در دسترس این گره تجاوز کند. این محدودیت در رابطه ۱ تعریف شده است. همچنین، در مشکل استقرار سرویس باید مهلت اجرای سرویس‌ها باید تضمین شود. هر برنامه a_k دارای مهلت زمانی D_{a_k} است که برای همه سرویس‌های متعلق به a_k یکسان است. اجازه دهید RT_{a_K} زمان پاسخ برنامه a_k باشد که باید از D_{a_k} کمتر باشد، همانطور که در رابطه ۲ نشان داده شده است.

[1] Fog Service Placement algorithm based on Complex Networks feature
[2] Genetic Algorithm

$$\sum_{s_l \in S} RR_{s_l} \le \varepsilon\, RR_{n^j}, \forall n^j \in f, RR = \{U, M, S\} \quad (۱)$$

$$RT_{a_k} \le D_{a_k}, \qquad \forall a_k \in A \quad (۲)$$

جائیکه، ε نرخ منابع آزاد هر گره است که نمی‌تواند برای اجرای سرویس‌ها استفاده شود.

یک الگوریتم مکان‌یابی کارآمد در محیط ابر-مه می‌تواند بر اساس فاکتورهای مختلفی تحلیل شود. به‌طورکلی، مشکل استقرار سرویس را می‌توان به‌عنوان یک مسئله چند هدفه مدل‌سازی کرد تا بهترین توافق‌ها بین اهداف مختلف حاصل شود. SPP-ICA نیز مسئله را برای ایجاد مصالحه بین پنج هدف مختلف (یعنی، بهره‌وری از منابع، هزینه سرویس و هزینه تأخیر) حل می‌کند.

بهره‌وری از منابع (RU): این فاکتور بر اساس تعداد مکان‌های استفاده شده در مه توسط سرویس‌های اینترنت اشیا اشاره دارد که باید حداکثر شود. هر سرویس اینترنت اشیا می‌تواند توسط یک گره مه در کلونی جاری، گره کنترل ارکستراسیون مه، یک گره مه در نزدیک‌ترین کلونی مه همسایه یا در ابر اجرا شود. بنابراین، مهلت اجرای برنامه‌ها در محاسبه RU باید در نظر گرفته شود. رابطه ۳ فاکتور بهره‌وری از منابع را تعریف می‌کند [۲۳].

$$RU = \sum_{a_k \varepsilon A} \left[P(a_k) * \sum_{s_l \in a_k} \left(\sum_{n^j \in f} x_{n^j}^{s_l} \right) + x_o^{s_l} + x_{nn}^{s_l} + x_{mw}^{s_l} \right] \quad (۳)$$

جائیکه، $x_{n^j}^{s_l}$، x_o^{sl}، x_{nn}^{sl} و x_{mw}^{sl} متغیرهای تصمیم باینری به ترتیب برای گره مه n^j، گره کنترل ارکستراسیون مه، نزدیک‌ترین کلونی مه همسایه و میان‌افزار کنترل ابر-مه هستند. برای مثال، $x_{n^j}^{s_l} = 1$ نشان می‌دهد که سرویس s_l روی گره n^j مستقر شده است و در غیر این صورت $x_{n^j}^{s_l} = 0$ است. همچنین، $P(a_k)$ اولویت برنامه a_k است.

هزینه سرویس (SC): این فاکتور بر اساس هزینه محاسبات (هزینه پولی اجرای سرویس) و هزینه ارتباطات (یعنی زمان اجرای سرویس) محاسبه می‌شود. بااین‌حال، این هزینه‌ها با توجه به مکان اجرای سرویس متفاوت است. ازآنجاکه برای اجرای سرویس-های یک برنامه اینترنت اشیا اولویت وجود ندارد، بنابراین می‌توان SC را بر مبنای برنامه محاسبه کرد. رابطه ۴ فاکتور هزینه کل سرویس‌ها را تعریف می‌کند [۲۴].

$$sc = \sum_{a_k \in A} \sum_{s_l \in a_k} \left[x_{n^j}^{s_l} \left(cp_{n^j} + cm_{n^j} \right) + x_o^{s_l} \left(cp_o + cm_o \right) + x_{nn}^{s_l} \left(cp_{nn} + cm_{nn} \right) + x_{mw}^{s_l} \left(cp_{cc} + cm_{cc} \right) \right] \quad (۴)$$

جائیکه، cp و cm به ترتیب هزینه محاسبات و هزینه ارتباطات برای یک سرویس بر اساس دستگاه مکان‌یابی شده هستند، همانطور که در روابطِ ۵ و ۶ نشان داده شده است [۲۵].

$$cp = P_{dv}(t_\beta - t_\alpha) \quad (۵)$$

$$cm = C_{dv^*} \frac{cv_{\alpha,\beta}}{BW} \quad (۶)$$

جائیکه، P_{dv} و C_{dv} به ترتیب به بهای واحد محاسبات و ارتباطات روی دستگاه dv اشاره دارد، t_α و t_β محدوده زمانی اجرای سرویس است، $cv_{\alpha,\beta}$ اندازه داده بین t_α و t_β است و BW پهنای باند خروجی است.

هزینه تأخیر (DC): این فاکتور برای هر سرویس اینترنت اشیا بر اساس تأخیر پیوند ارتباطی و تأخیر پردازش محاسبه می‌شود. تأخیر پیوند ارتباطی مرتبط با $d_o^{n^j}$، d_o^{mw}، d_o^{IoT} و d_o^{nn} است، جائیکه به دلیل فاصله نزدیک در لبه شبکه بسیار ناچیز هستند و می‌توان از آن صرفه نظر کرد. با این وجود، تأخیر پردازش مرتبط با بارهای کاری سرویس است. یک سرویس اینترنت اشیا باید توسط گره کنترل ارکستراسیون مه پردازش شود تا سیاست مکان‌یابی آن تعیین شود. پس از آن، سرویس با توجه به دستگاه تخصیص داده شده (یعنی، گره مه، گره کنترل ارکستراسیون مه، نزدیک‌ترین دامنه مه همسایه یا ابر) می‌تواند تأخیر پردازش متفاوتی داشته باشد. رابطه ۷ فاکتور هزینه تأخیر را تعریف می‌کند [۲۶].

$$DC = \sum_{a_k \in A} \sum_{s_l \in a_k} d_o^{s_l} + \left[x_{n^j}^{s_l} . d_{n^j}^{s_l} + x_o^{s_l} . d_o^{s_l} + x_{nn}^{s_l} . d_{nn}^{s_l} + x_{mw}^{s_l} . d_{mw}^{s_l} \right] \quad (۷)$$

جائیکه، $d_{n^j}^{sl}$، d_o^{sl}، d_{nn}^{sl} و d_{mw}^{sl} به ترتیب تأخیر پردازش سرویس s_l روی گره n^j، گره کنترل ارکستراسیون مه، نزدیک‌ترین دامنه مه همسایه و ابر است. با توجه به انجام مکان‌یابی توسط گره کنترل ارکستراسیون مه ، تأخیر d_o^{sl} همراه باید در نظر گرفته شود. در اینجا، d_{mw}^{sl} با توجه به منابع غنی ابر تنها مبتنی بر فاصله (یعنی، تأخیر پیوند ارتباطی) محاسبه می‌شود. علاوه بر این، d_{nn}^{sl} شامل تأخیر پیوند ارتباطی و تأخیر پردازش است. با این وجود، $d_{n^j}^{sl}$ و d_o^{sl} تنها بر اساس تأخیر پردازش محاسبه می‌شوند. تأخیر پردازش سرویس s_l بر اساس رابطه ۸ محاسبه می‌شود.

$$d^{s_l} = \frac{SS_{s_l} . N_{s_l}}{\vartheta} \tag{۸}$$

جائیکه، N_{s_l} تعداد سیکل‌های موردنیاز برای اجرای سرویس s_l و υ فرکانس گره تخصیص داده شده است.

۴- الگوریتم مکان‌یابی پیشنهادی

هر کلونی مه شامل یک گره کنترل ارکستراسیون مه است که وظیفه اصلی آن انجام مکان‌یابی سرویس‌های اینترنت اشیا است. این گره شامل یک واحد کنترل پذیرش است که همه درخواست‌های ورودی به کلونی مه را از طریق درگاه‌های مه دریافت می‌کند. این واحد بر اساس اطلاعات مربوط به منابع کلونی مه تابع و مهلت زمانی درخواست‌ها می‌تواند در مورد مکان‌یابی درخواست روی کلونی مه جاری یا ارسال درخواست به ابر تصمیم‌گیری کند. درخواست‌هایی که توسط واحد کنترل پذیرش برای مکان‌یابی روی کلونی مه جاری در نظر گرفته می‌شوند در یک صف قرار می‌گیرند. در هر دوره زمانی τ، درخواست‌های موجود در صف توسط گره کنترل ارکستراسیون مه برای استقرار روی گره‌های مه پردازش می‌شوند گره کنترل ارکستراسیون مه با استفاده از مدل حلقه کنترل MADE-k مکان‌یابی را به‌صورت خودمختار انجام می‌دهد MADE-k شامل چهار فاز به شرح زیر است:

فاز نظارت: این فاز مسئول نظارت بر وضعیت گره‌های تابع، منابع در دسترس گره‌ها و منابع موردنیاز برنامه‌های اینترنت اشیا است. همچنین در این فاز میزان استفاده از منابع (یعنی، پردازنده، حافظه و ذخیره‌سازی) و درخواست‌های منتقل‌شده به نزدیک‌ترین کلونی مه همسایه و ابر نظارت می‌شود. نظارت در هر دوره زمانی τ انجام می‌شود و گزارش‌های مربوط به آن در پایگاه دانش ذخیره می‌شوند.

فاز تجزیه و تحلیل: در این فاز اولویت اجرای درخواست‌ها برای هر برنامه اینترنت اشیا محاسبه می‌شود. مطالعات زیادی نشان داده‌اند که فاصله زمانی تا مهلت اجرای برنامه‌ها می‌تواند به خوبی اولویت اجرای برنامه‌ها را بیان کند [۲۷]. اجازه دهید $P(a_k)$ اولویت اجرا برنامه a_k باشد، همانطور که در رابطه ۹ نشان داده شده است. در نهایت، جزئیات مربوط به اولویت‌بندی برنامه‌ها در پایگاه دانش ذخیره می‌شود.

$$P(a_k) + \lambda . \frac{1}{D_{a_k}} + (1-\lambda). \frac{1}{t - R_{a_k}} \tag{۹}$$

جائیکه، D_{a_k} مهلت اجرای برنامه a_k است، R_{a_k} زمان ارسال برنامه a_k توسط کاربر است، t زمان فعلی است و λ ضریب برای در نظر گرفتن تاثیر D_{a_k} نسبت به R_{a_k} است.

فاز تصمیم‌گیری: این فاز وظیفه مکان‌یابی سرویس‌های اینترنت اشیا را روی گره‌های مه تابع بر عهده دارد. در هر دوره زمانی τ، سرویس‌های اینترنت اشیا دریافت شده برای مکان‌یابی پردازش می‌شوند در این مقاله فرایند تصمیم‌گیری توسط ICA به‌عنوان یک رویکرد فراابتکاری انجام می‌شود پس از انجام مکان‌یابی جزئیات استقرار سرویس‌ها در پایگاه دانش ذخیره می‌شود.

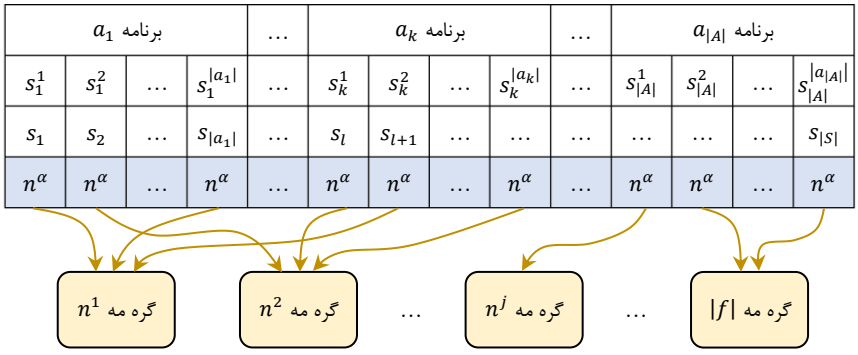
فاز اجرا: برنامه ریزی مکان‌یابی سرویس‌های اینترنت اشیا در این فاز اجرا می‌شود. در اینجا، هر سرویس اینترنت اشیا با توجه به فاز تصمیم‌گیری روی یک گره مه مستقر می‌شود. سپس، گره مه منابع موردنیاز سرویس را فراهم کرده و نتایج اجرا را در پایگاه دانش ذخیره می‌کند.

SPP-ICA تلاش دارد با ایجاد مصالحه بین اهداف مختلف عملکرد کلی سیستم و رضایت از کیفیت خدمات را بهبود دهد. به‌طورکلی، ICA همانند سایر رویکردهای تکاملی شامل گام‌های ۱) طرح کدگذاری راه‌حل‌ها، ۲) ایجاد جمعیت اولیه، ۳) محاسبه برازندگی، ۴) تکامل جمعیت و ۵) شرایط توقف می باشد. در ادامه جزئیات گام‌های ICA در الگوریتم پیشنهادی SPP-ICA به تفصیل توضیح داده شده است.

بر اساس تکنیک پیشنهادی، جمعیت اولیه شامل $N_{country}$ راه‌حل (به‌عنوان کشور) به‌صورت تصادفی-ابتکاری ایجاد می‌شود. با توجه به ICA، هر راه‌حل به‌عنوان یک کشور مستعمره یا امپریالیست معرفی می‌شود. در اینجا، N_{col} تعداد کشورهای مستعمره و N_{imp} تعداد کشورهای امپریالیست است، جائیکه، $N_{imp}+N_{col}=N_{country}$. نوع راه حل در ICA بر اساس تابع برازندگی (به‌عنوان قدرت) تعیین می‌شود. اجازه دهید $c_q, \forall_q = 1,2,...,N_{country}$ راه حل $q-$ ام از جمعیت باشد و $OF(c_q)$ به مقدار برازندگی برای c_q اشاره کند. ICA کشورها را بر اساس برازندگی مرتب‌سازی نزولی می‌کند. سپس، N_{imp} از راه‌حل‌ها با بالاترین مقدار برازندگی (یعنی، $(c_1, c_2, ..., c_{N_{imp}})$ به‌عنوان امپریالیست‌ها انتخاب می‌شوند در حالی که مابقی راه‌حل‌ها (یعنی، $c_{N_{imp}+1}, c_{N_{imp}+2}, ..., N_{country}$) به‌عنوان کشورهای مستعمره در نظر گرفته می‌شوند.

۴-۱-طرح کدگذاری راه‌حل‌ها

هر راه‌حل در ICA به‌عنوان یک کشور شناخته می‌شود. طرح کدگذاری نشان دهنده نگاشت مشکل استقرار سرویس بر اساس برنامه‌ریزی خطی عدد صحیح (ILP) است که اطلاعات لازم را برای تحلیل و جستجوی راه‌حل بهینه فراهم می‌کند. به‌طورکلی، سرویس‌های اینترنت اشیا مختلف می‌تواند به گره‌های مه یکسانی تخصیص داده شوند، جائیکه گره‌ها باید منابع موردنیاز سرویس‌ها را داشته باشند. در این راستا، SPP-ICA استفاده مجدد از گره‌های مه را در سرویس‌های اینترنت اشیا مختلف برای حصول شتاب محاسباتی پیشنهاد می‌کند. MADE-k بر اساس چارچوب SPP-ICA مکان‌یابی را تنها روی گره‌های مه در دسترس کلونی جاری انجام می‌دهد. بنابراین، هدف استقرار S سرویس اینترنت اشیا ازپ A برنامه دریافتی روی f گره مه در دسترس کلونی جاری در دوره زمانی π است. شکل ۱ طرح کدگذاری راه‌حل‌ها را در SPP-ICA نشان می‌دهد، جائیکه بردار راه‌حل با رنگ آبی برجسته شده است.



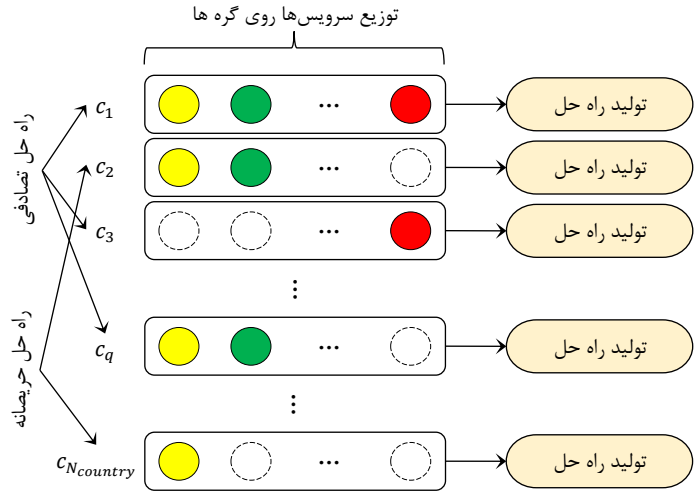
شکل ۱: طرح کدگذاری راه‌حل‌ها در SPP-ICA
Figure1. Solution encoding scheme in SPP-ICA

همانطور که نشان داده شده است، هر برنامه اینترنت اشیا نظیر a_k شامل a_k سرویس اینترنت اشیا است و در مجموع S سرویس اینترنت اشیا در دوره زمانی τ برای مکان‌یابی وجود دارد. هر سرویس اینترنت اشیا نظیر s_l باید روی یک گره مه نظیر $n^\alpha \in f$ مستقر شود. n^α می‌تواند یک گره مه با منابع کافی از بین $|f|$ گره مه موجود در کلونی جاری باشد. بنابراین، چندین سرویس اینترنت اشیا می‌تواند روی n^α مستقر شود. بر اساس طرح کدگذاری، هر راه‌حل شامل برداری به طول $|S|$ است که هر عنصر آن به یک سرویس اینترنت اشیا و ارزش هر عنصر به اندیس گره مه تخصیص داده شده اشاره دارد.

۴-۲-ایجاد جمعیت اولیه

ایجاد جمعیت اولیه به‌عنوان گام نخست در فرآیند بهینه‌سازی الگوریتم‌های تکاملی است. جمعیت زیر مجموعه ای از راه حل ها در تکرار جاری است. به‌طورکلی، جمعیت اولیه به‌صورت تصادفی یا از طریق روش های ابتکاری بر اساس طرح کدگذاری تعریف شده ایجاد می‌شود. معمولا روش تصادفی با توجه به محدودیت‌ها و الزامات مختلف در مشکل استقرار سرویس منجر به تولید راه‌حل‌های غیرممکن یا کیفیت پایین می‌شود. علاوه بر این، روش های ابتکاری نیز می‌تواند منجر به کاهش تنوع جمعیت و وقوع حالتی به نام همگرایی زودرس شود. بر این اساس، SPP-ICA ایجاد جمعیت را با یک تکنیک تصادفی-ابتکاری توصیه می‌کند.

تکنیک تصادفی-ابتکاری بر اساس توزیع منابع مدل‌سازی شده است. ازآنجاکه گره‌های مه با داشتن منابع کافی می‌توانند میزبان چندین سرویس اینترنت اشیا باشند، در نظر گرفتن توزیع منابع در ایجاد جمعیت اولیه می‌تواند منجر به افزایش تعداد مکان-یابی‌های موفق در آینده شود. در این تکنیک، اولین راه حل به‌صورت تصادفی ایجاد می‌شود. سپس، میانگین منابع محاسباتی اشغال شده از هر گره مه مرتبط با درخواست جاری در حافظه ذخیره می‌شود. این حافظه می‌تواند توزیع مصرف منابع در گره‌های مه را با توجه به استقرار سرویس‌های اینترنت اشیا نشان دهد. در این راستا، دومین راه حل به‌صورت حریصانه ایجاد می‌شود. در روش حریصانه سرویس‌ها بر مبنای میانگین منابع محاسباتی مرتب‌سازی صعودی شده و هر سرویس به ترتیب به گره مه با بیشترین منابع تخصیص داده می‌شود. این فرایند در نهایت یک راه‌حل ابتکاری برای مکان‌یابی ایجاد می‌کند که توزیع کارآمد سرویس‌ها روی گره ها را در نظر می گیرد. در ادامه سایر راه‌حل ها ایجاد می‌شوند، جائیکه پس از هر راه‌حل تصادفی یک راه‌حل به روش حریصانه ایجاد می‌شود. به‌طورکلی، توزیع کارآمد منابع مصرف شده در طول ایجاد جمعیت می‌تواند منجر به افزایش تعداد درخواست‌های پذیرفته شده شود. معماری ایجاد جمعیت اولیه با تکنیک تصادفی-ابتکاری در شکل ۲ نشان داده شده است.



شکل ۲: معماری ایجاد جمعیت اولیه با تکنیک تصادفی-ابتکاری

Figure2.Architecture of initial population generation with random-innovative technique

بر اساس تکنیک پیشنهادی، جمعیت اولیه شامل Ncountry راه حل (به‌عنوان کشور) به‌صورت تصادفی-ابتکاری ایجاد می‌شود. با توجه به ICA، هر راه‌حل به‌عنوان یک کشور مستعمره یا امپریالیست معرفی می‌شود. در اینجا، *N*col تعداد کشورهای مستعمره و *N*imp تعداد کشورهای امپریالیست است، جائیکه، *N*imp+*N*col=*N*country. نوع راه حل در ICA بر اساس تابع برازندگی (به‌عنوان قدرت) تعیین می‌شود. اجازه دهید Cq,∀q=1,2,…,Ncountry راه حل q -ام از جمعیت باشد و $OF(c_q)$ به مقدار برازندگی برای C_q اشاره کند. ICA کشورها را بر اساس برازندگی مرتب‌سازی نزولی می‌کند. سپس، Nimp از راه‌حل‌ها با بالاترین مقدار برازندگی (یعنی، $C_1,C_2,\ldots,C_{Nimp}$به‌عنوان امپریالیست‌ها انتخاب می‌شوند درحالی‌که مابقی راه‌حل‌ها (یعنی، $C_{Nimp+1},C_{Nimp+2}),\ldots,C_{Ncountry}$به‌عنوان کشورهای مستعمره در نظر گرفته می‌شوند.

۴-۳-محاسبه برازندگی

هر راه‌حل از جمعیت نشان دهنده یک طرح مکان‌یابی است که با توجه به محدودیت‌های محیط مه و الزامات کیفیت خدمات می‌تواند یک مقدار برازندگی داشته باشد. SPP-ICA برازندگی راه‌حل‌ها را به‌صورت چند-هدفه محاسبه می‌کند. هدف SPP-ICA ایجاد مصالحه بین هزینه‌ها (یعنی، بهره‌وری از منابع و هزینه سرویس) و کیفیت خدمات (یعنی، هزینه تأخیر) است که از طریق یک استراتژی مکان‌یابی کارآمد حاصل می‌شود. پارامترهای موردنیاز برای محاسبه این اهداف از طریق اطلاعات کلونی مه و جزئیات سرویس‌های اینترنت اشیا در دسترس می باشد. بنابراین، تابع هدف پیشنهادی شامل سه هدف مختلف است که مطابق رابطه ۱۰ محاسبه می‌شود و ICA به دنبال حداکثر کردن آن است.

$$OF = \frac{\xi_{sc} * SC + \xi_{dc} * DC + \xi_{\varepsilon c} * EC}{\xi_{rc} * RU} \qquad (۱۰)$$

جائیکه، RU، SC و DC به ترتیب بهره‌وری از منابع، هزینه سرویس و هزینه تأخیر هستند. همچنین، ξ_{ru}، ξ_{sc}و ξ_{dc} به‌ترتیب وزن های این اهداف هستند.

۴-۴-تکامل جمعیت

فرایند تکامل در ICA شامل تشکیل امپراطوری‌ها، سیاست جذب و سیاست رقابت است.

تشکیل امپراطوری‌ها: جمعیت در ICA شامل N_{col} مستعمره و N_{imp} امپریالیست است، جائیکه هر کشور امپریالیست دارای یک امپراطوری است. بنابراین، تعداد امپراطوری‌ها برابر تعداد امپریالیست‌ها (یعنی، N_{imp}) است و هر کشور مستعمره متعلق به یک امپراطوری است. امپریالیست‌ها بر اساس قدرت خود مستعمرات را تصاحب می کنند. تقسیم مستعمرات با توجه به قدرت (یعنی، برازندگی) هر امپریالیست انجام می‌شود، همانطور که در روابط ۱۱ و ۱۲ نشان داده شده است.

$$P_e = \frac{OF(c_\varepsilon)}{\sum_{q=1}^{N_{imp}} OF(c_q)},\ \forall e = 1,2,...,N_{imp} \qquad (۱۱)$$

$$N.c_\varepsilon = \left\| P_\varepsilon * N_{col} \right\|,\ \forall e = 1,2,...,N_{imp} \qquad (۱۲)$$

جائیکه، $OF(c_\varepsilon)$ مقدار برازندگی امپریالست c_ε است و P_ε قدرت نسبی این امپریالیست را نشان می‌دهد. همچنین، $\|*\|$ تابع گرد کردن است و $N.c_\varepsilon$ به تعداد مستعمرات اولیه برای امپریالیست c_ε اشاره دارد که به‌صورت تصادفی از بین کشورهای مستعمره انتخاب می‌شود.

سیاست جذب: سیاست جذب در ICA به مفهوم حرکت مستعمرات هر امپراطوری به سمت امپریالیست است. SPP یک مسئله گسسته است و ما برای سیاست جذب از عملگر تفاضل تکاملی[1] (DE) استفاده می‌کنیم. DE پیشنهاد شده بر اساس c_q (به‌عنوان یک مستعمره) و c_ε (به‌عنوان یک امپریالیست) مفهوم سیاست جذب را از طریق تفاوت بین ویژگی‌های c_q و c_ε مدل‌سازی می‌کند، همانطور که در رابطه ۱۳ نشان داده شده است. DE یک تابع دو بخشی است که یک بخش از آن برای هر عنصر از c_q بر اساس احتمال ϕ اعمال می‌شود.

$$c_{q,j} = \begin{cases} Selector(c_{\varepsilon,j}, c_{q,j}) with\ probability\ of\ \rho & rand(0.1) > \phi \\ \left\| c_{q,j} + \theta * \sqrt{(c_{\varepsilon,j} - c_{q,j})^2} \right\| & otherwise \end{cases}, \forall j = 1,2,..,|s_\tau| \qquad (۱۳)$$

جائیکه، $C_{q,j}$ و $C_{\varepsilon,j}$ به ترتیب گره مه تخصیص داده شده را به j-مین سرویس اینترنت اشیا برای c_q و c_ε نشان می‌دهد. همچنین، $|S_\tau|$ تعداد کل سرویس‌های IoT و θ ضریب انحراف برای حرکت c_q به سمت c_ε است که سرعت تکامل جمعیت

[1] Differential Evolution Crossover

را کنترل می‌کند. علاوه بر این، $Selector(c_{\varepsilon,j}, c_{q,j})$ تابعی است که بر اساس احتمال ρ یک عنصر از بین $c_{e,j}$ و $c_{q,j}$ است. $\rho \in [0,1]$ یک ثابت عددی است که گام حرکت c_q به سمت c_ε را کنترل می‌کند.

سیاست رقابت: سیاست رقابت در ICA به مفهوم تصاحب مستعمرات امپراطوری‌های ضعیف توسط امپراطوری‌های قوی است. در هر تکرار از ICA، ضعیفترین مستعمره از هر امپراطوری برای تصاحب توسط سایر امپراطوری‌ها به رقابت گذاشته می‌شود. مستعمره ضعیف بر اساس مقدار برازندگی تعیین می‌شود. اجازه دهید c_w ضعیف‌ترین مستعمره از امپراطوری E_ε باشد. در اینجا، P_ε احتمال پیوستن c_w به E_ε است و توسط روابطه ۱۴ و ۱۵ محاسبه می‌شود.

$$N.T.c_\varepsilon = OF(c_\varepsilon) + \frac{\xi * \sum_{c_q \in E_\varepsilon} OF(c_q)}{|E_\varepsilon|},\ \forall e = 1,2,...,N_{imp} \quad (۱۴)$$

$$P_\varepsilon = \frac{N.T.c_\varepsilon}{\sum_{q=1}^{N_{imp}} N.T.c_q},\ \forall e = 1,2,...,N_{imp} \quad (۱۵)$$

جائیکه، $N.T.c_\varepsilon$ قدرت نسبی امپراطوری E_ε است و ξ ثابت عددی که اثربخشی قدرت مستعمرات در محاسبه قدرت امپراطوری را تعیین می‌کند. همچنین، P_ε احتمال تصاحب c_w توسط E_ε است که برای همه امپراطوری‌ها محاسبه می‌شود. در این مقاله از عملگر چرخ رولت [۲۸] برای تعیین امپراطوری برنده در رقابت برای تصاحب c_w استفاده می‌شود. سیاست رقابت امپریالیستی در ICA در نهایت منجر به سقوط همه امپراطوری‌ها و تشکیل فقط یک امپراطوری قدرتمند می‌شود. در فرایند بهینه سازی امپراطوری‌های بدون مستعمره برای تصاحب بین سایر امپراطوری‌ها به‌عنوان یک مستعمره به رقابت گذاشته می‌شود.

۴-۵-شرایط توقف

ما شرایط توقف SPP-ICA را از طریق ۱) رسیدن به نقطه همگرایی و ۲) حداکثر تکرار مرسوم فراهم می‌کنیم. روش اول هنگامی ارضا می‌شود که تنها یک امپراطوری وجود داشته باشد و روش دوم با یک حداکثر تکرار($Iter_{max}$) شرط توقف را تعریف می‌کند. پس از ارضا شرایط توقف، بهترین راه‌حل بر مبنای تابع هدف به‌عنوان خروجی الگوریتم در نظر گرفته می‌شود و مکان‌یابی سرویس‌ها بر اساس آن انجام می‌شود.

۵-نتایج شبیه سازی

ا الگوریتم SPP-ICA پیشنهادی برای مکان‌یابی برنامه‌های اینترنت اشیا از ICA به‌عنوان یک الگوریتم فراابتکاری استفاده می‌کند. از اینرو، ما SPP-ICA را با الگوریتم های هم ارزی نظیر SPP-PSO [۱۸]، CSA-FSPP [۱۹] و FSP-ODMA [۲۰] مقایسه می‌کنیم. همه این الگوریتم ها روی یک محیط مه یکسان با استفاده از شبیه‌ساز متلب پیاده‌سازی شده است. بعلاوه، همه شبیه سازی ها با توجه به وجود برخی پارامترهای تصادفی بر مبنای ۱۵ اجرای مستقل انجام می‌شوند تا نتایج قابل اعتماد باشند.

۵-۱- راه اندازی شبیه‌سازی

شبیه سازی همه الگوریتم های مقایسه شده بر اساس یک محیط مه مصنوعی انجام شده است. تعداد کل کلونی‌های مه موجود برابر ۵ در نظر گرفته شده است. هر درخواست به یک کلونی مه به‌صورت تصادفی تخصیص داده می‌شود و هر کلونی مه مکان‌یابی درخواست را بر مبنای مدل برنامه ریزی تعریف شده روی گره‌های مه تابع انجام می‌دهد. در اینجا، هزینه پردازش در ابر برای هر واحد زمان صورتحساب[۲] (BTU) برابر ۰.۳ دلار است [۱۹]. هر گره شامل تعداد مشخصی منابع محاسباتی و ذخیره‌سازی است که به‌صورت بلوک در نظر گرفته می‌شود. از طرف دیگر، هر سرویس اینترنت اشیا به تعداد مشخصی از بلوک منابع نیاز دارد. با استقرار هر سرویس روی یک گره مه، بخشی از بلوک منابع آن اشغال می‌شود. به‌طورکلی، بلوک‌های منابع اشغال شده

[2] billing time unit

تا پایان شبیه‌سازی آزاد نمی‌شوند [۲۹]. علاوه بر این، تعداد انواع سرویس‌های ارائه شده توسط گره ها مه محدود است و برابر ۵ تنظیم شده است.

در این مقاله، نتایج شبیه سازی بر اساس مکان‌یابی سرویس‌ها در ۱۰ دوره زمانی متوالی (یعنی، $\tau=1,2,\ldots,10$) تحلیل می‌شود. تعداد سرویس‌های درخواست شده به همراه زمان درخواست سرویس‌ها مطابق جدول ۱ تنظیم شده است. در اینجا، شبیه سازی در ۱۰ دوره زمانی برای ۶۱۵ سرویس انجام می‌شود. گره‌های مه و سرویس‌های اینترنت اشیا دارای پارامترهای مختلفی هستند. جدول ۲ تنظیمات آزمایش ها را برای گره‌های مه و سرویس‌های اینترنت اشیا نشان می‌دهد [۱۷-۲۰]. اجازه دهید مقادیر پارامترهایی با یک محدوده مشخص به‌صورت تصادفی با توزیع یکنواخت انتخاب شوند.

جدول ۱: جزئیات سرویس‌های درخواست شده در ۱۰ دوره زمانی متوالی
Table1. Details of requested services in 10 consecutive time periods

دوره زمانی τ	۱	۲	۳	۴	۵	۶	۷	۸	۹	۱۰
زمان t	۸	۱۶	۲۴	۳۲	۴۰	۴۸	۵۶	۶۴	۷۲	۸۰
تعداد سرویس‌ها	۷۱	۴۸	۴۸	۴۶	۹۶	۷۷	۳۷	۵۲	۸۵	۵۵

جدول ۲: تنظیمات محیط مه مصنوعی در شبیه سازی
Table2. Parameter settings for fog nodes and IoT services in the simulation

پارامترها	مقادیر
تعداد انواع سرویس‌ها برای هر برنامه IoT	۲۰
تعداد بلوک های منبع برای هر گره مه	۶۰۰۰-۵۵۰۰
تعداد بلوک های منبع برای هر سرویس IoT	۳۵-۲۵
هزینه هر سرویس IoT (دلار)	۲۰-۱۰
تعداد گره‌های مه در هر کلونی	۱۵

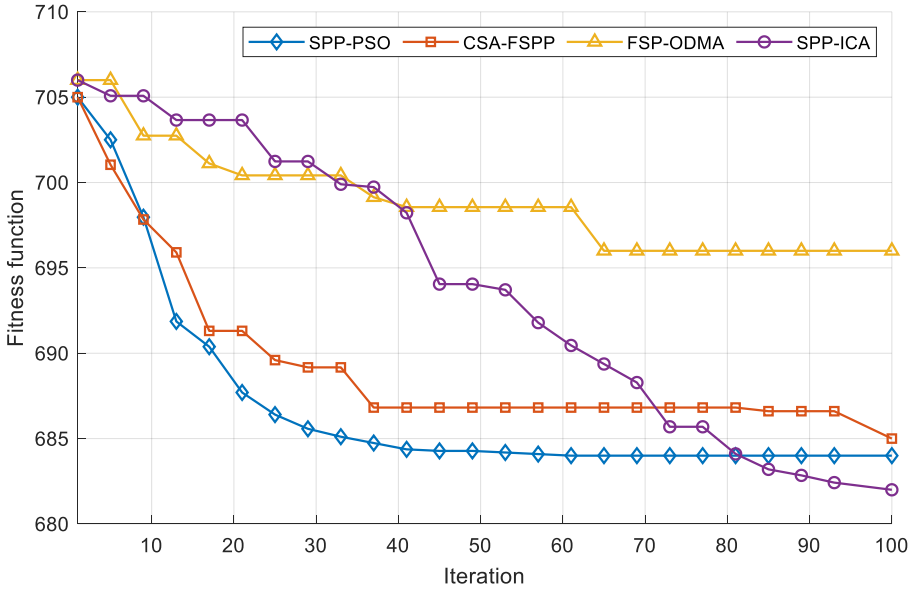
هر گره دارای شامل منابع ذخیره‌سازی، حافظه و پردازنده است. به‌طور مشابه، هر سرویس اینترنت اشیا دارای تقاضای منابع شامل ذخیره‌سازی، حافظه و پردازنده می باشد. جدول ۳ جزئیات منابع در دسترس گره ها مه و منابع موردنیاز سرویس‌ها را نشان می‌دهد. در اینجا، جزئیات منابع برای هر ۵ گره مه در یک کلونی مه یکسان در نظر گرفته شده است. همچنین، تعداد انواع برنامه‌های اینترنت اشیا برابر ۵ در نظر گرفته شده است، بنابراین هر گره مه حداکثر قابلیت پشتیبانی از ۵ نوع سرویس مختلف را دارد. علاوه بر این، هر برنامه اینترنت اشیا دارای یک مهلت است که به‌صورت تصادفی بین ۱۲۰ تا ۲۴۰ ثانیه تنظیم شده است.

جدول ۳: جزئیات منابع برای گره ها و سرویس‌ها
Table3. Resource details for nodes and services

آیتم	نوع	ذخیره‌سازی (MB)	حافظه (MB)	پردازنده (MIPS)
گره مه	۱	۲۵۶	۲۵۶	۲۰۰-۱۰۰
	۲	۵۱۲	۵۱۲	۳۰۰-۲۰۰
	۳	۱۰۲۴	۱۰۲۴	۱۴۰۰-۳۰۰
	۴	۲۰۴۸	۲۰۴۸	۱۶۰۰-۱۴۰۰
	۵	۴۰۹۶	۴۰۹۶	۳۰۰۰-۱۶۰۰
سرویس IoT	۱	۱۲۸	۱۲۸	۲۰۰-۱۰۰
	۲	۲۵۶	۲۵۶	۳۰۰-۲۰۰
	۳	۵۱۲	۵۱۲	۱۴۰۰-۳۰۰
	۴	۲۰۴۸	۱۰۲۴	۱۶۰۰-۱۴۰۰
	۵	۴۰۹۶	۴۰۹۶	۳۰۰۰-۱۶۰۰

۵-۲-نتایج و مقایسه‌ها

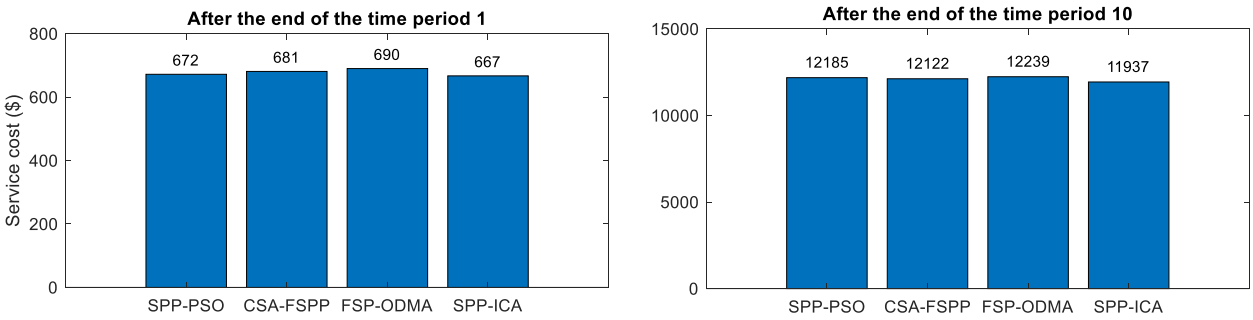
این بخش شامل نتایج شبیه سازی و مقایسه ها و بحث پیرامون آن است. الگوریتم SPP-ICA مبتنی بر یک الگوریتم فراابتکاری است که از ICA برای تصمیم‌گیری در مورد مکان‌یابی استفاده می‌کند. به همین دلیل SPP-ICA با سایر الگوریتم های فراابتکاری مشابه مقایسه می‌شود. در آزمایش اول، سرعت همگرایی الگوریتم های مختلف برای افزایش ارزش برازندگی و رسیدن به نقطه همگرایی مقایسه می‌شود. در نقطه همگرایی به دلیل کاهش تنوع جمعیت می‌توان فرایند تکامل را متوقف کرد. بااین‌حال، ما مقایسه را در هر الگوریتم برای ۱۰۰ تکرار پس از پایان دوره زمانی اول گزارش می‌کنیم، همانطور که در شکل ۳ نشان داده شده است. سرعت همگرایی SPP-PSO و CSA-FSPP مناسب است و در تکرار ۳۵ به نقطه همگرایی می‌رسند. بااین‌حال، همگرایی SPP-ICA به نتایج مطلوبتری منجر شده است، جائیکه در هر ۱۰۰ تکرار موفق به کاهش مقدار برازندگی شده است. بدترین نتایج به FSP-ODMA اختصاص دارد، زیرا این الگوریتم برای محیط های پیوسته توسعه یافته و عملکرد مناسبی برای مسائل گسسته ندارد. به‌طورکلی، SPP-PSO، CSA-FSPP و FSP-ODMA به‌ترتیب به مقادیر برازندگی ۶۸۴، ۶۸۵ و ۶۹۶ پس از ۱۰۰ تکرار دست می‌یابند.



شکل ۳: نتایج برای سرعت همگرایی در پایان دوره زمانی اول

Figure3. Results for the convergence speed at the end of the time period $\tau = 1$

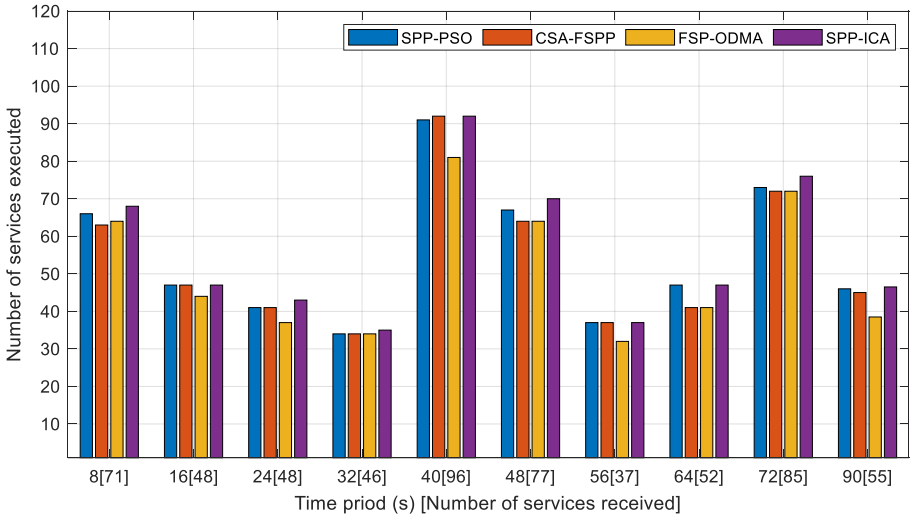
آزمایش دوم به مقایسه هزینه سرویس در الگوریتم های مختلف می‌پردازد. هزینه سرویس یکی از متریک‌های مهم در بهبود کیفیت خدمات است، جائیکه به هزینه پولی پرداخت شده توسط کاربر برای اجرای سرویس‌ها اشاره دارد. هزینه سرویس می‌تواند با استقرار درخواست‌های کاربر به گره‌های مه کاهش پیدا کند. بنابراین، الگوریتم هایی که با مکان‌یابی کارامتر می‌توانند هزینه سرویس کمتری برای کاربر فراهم کنند. شکل ۴ نتایج مرتبط با هزینه سرویس را برای یک دوره زمانی و همچنین ۱۰ دوره زمانی نشان می‌دهد. همانطور که در نمودار فوق مشاهده می‌شود، SPP-ICA در هر دو سناریو نتایج بهتری با هزینه سرویس کمتر ارائه داده است. به‌طور میانگین، SPP-ICA در مقایسه با SPP-PSO، CSA-FSPP و FSP-ODMA به ترتیب هزینه سرویس را ۲.۰۵٪، ۱.۵۲٪ و ۱.۵۱٪ کاهش داده است. دلیل این برتری تعداد سرویس‌های مکان‌یابی شده بیشتر روی کلونی مه جاری توسط SPP-ICA است.



شکل ۴: مقایسه الگوریتم های مختلف بر اساس هزینه سرویس

Figure4. Comparison of different algorithms based on service cost

در آزمایش سوم، تعداد سرویس‌های انجام شده برای هر الگوریتم در هر دوره زمانی مقایسه می‌شود. شکل ۵ نتایج این مقایسه را به تفکیک دوره زمانی برای ۶۱۵ سرویس درخواست شده نشان می‌دهد. در هر دوره زمانی تعداد مشخصی از سرویس‌ها برای مکان‌یابی دریافت می‌شوند. در اینجا، برای هر الگوریتم نتایج مرتبط با تعداد سرویس‌های انجام شده با توجه به تعداد سرویس‌های دریافت شده گزارش شده است. SPP-ICA در اغلب دوره ها نتایج بهتری را ارائه می‌دهد. بااین‌حال، در مقایسه با SPP-PSO رقابتی است. به‌طورکلی، SPP-ICA موفق به مکان‌یابی ۵۶۱ سرویس شده است. این برای الگوریتم هایSPP-PSO ، CSA-FSPP و FSP-ODMA به ترتیب ۵۴۹، ۵۳۶ و ۵۰۷ گزارش شده است. در واقع، SPP-ICA پس از دوره زمانی ۱۰ از ۶۱۵ سرویس دریافت شده موفق به مکان‌یابی ۵۶۱ سرویس شده است که در مقایسه با سایر الگوریتم‌ها برتری قابل توجهی دارد.



شکل ۵. نتایج برای تعداد سرویس‌های انجام شده در هر دوره زمانی

Figure5. Results for the number of services performed in each time period

در آخرین آزمایش، تعداد سرویس‌های باقیمانده و تعداد سرویس‌های ناموفق در مقایسه با تعداد کل سرویس‌های دریافت شده برای هر الگوریتم بررسی شده است. الگوریتم پیشنهادی SPP-ICA موفق به اجرای ۵۶۱ سرویس از ۶۱۵ سرویس دریافت شده شده است. همچنین، الگوریتم هایSPP-PSO ، CSA-FSPP و FSP-ODMA به ترتیب ۵۴۹، ۵۳۶ و ۵۰۷ سروس اجرا شده را گزارش می کنند. از میان ۵۴ سرویس اجرا نشده در SPP-ICA، ۷ سرویس به دلیل کمبود منابع موفق به مکان‌یابی نشده و هنوز مهلت برای اجرا برای آنها نقض نشده است. بنابراین این سرویس‌ها می توانند در دوره زمانی بعدی اجرا شوند. بااین‌حال، اجرای ۴۷ سرویس در SPP-ICA ناموفق بوده است. در واقع، منابع موردنیاز این سرویس‌ها قبل از مهلت فراهم نشده است. نتایج مرتبط با تعداد سرویس‌های باقیمانده و تعداد سرویس‌های ناموفق برای سایر الگوریتم ها در جدول ۴ نشان داده شده است. در اینجا، نتایج پس از پایان دوره زمانی اول و همچنین پس از پایان ۱۰ دوره زمانی گزارش شده است. نتایج نشان دهنده برتری روش پیشنهادی است، زیرا موفق با کاهش تعداد سرویس‌های باقیمانده و تعداد سرویس‌های ناموفق شده است.

جدول ۴: نتایج تعداد سرویس‌های باقیمانده و تعداد سرویس‌های ناموفق در الگوریتم های مختلف
Table4. Results of the number of remaining services and the number of failed services in different algorithms

دوره زمانی	الگوریتم‌ها	تعداد سرویس‌های دریافت شده	تعداد سرویس‌های اجرا شده	تعداد سرویس‌های ناموفق	تعداد سرویس‌های باقیمانده
$\tau = 1$	SPP-PSO	۷۱	۶۶	۰	۵
	CSA-FSPP	۷۱	۶۳	۰	۸
	FSP-ODMA	۷۱	۶۴	۰	۷
	SPP-ICA	۷۱	۶۸	۰	۳
$\tau = 10$	SPP-PSO	۶۱۵	۵۴۹	۵۷	۹
	CSA-FSPP	۶۱۵	۵۳۶	۶۲	۱۷
	FSP-ODMA	۶۱۵	۵۰۷	۸۹	۱۹
	SPP-ICA	۶۱۵	۵۶۱	۴۷	۷

با در نظر گرفتن همه متریک های کارایی، SPP-ICA نسبت به سایر الگوریتم‌های مورد مقایسه برتری دارد. پس از آن، SPP-PSO، CSA-FSPP و FSP-ODMA به ترتیب در رتبه های بعدی قرار دارند. الگوریتم های SPP-ICA و SPP-PSO کمترین نقش مهلت را دارند، اما SPP-PSO در مقایسه با SPP-ICA منابع ابری بیشتری را مصرف می‌کند. استفاده از منابع ابری منجر به افزایش تأخیر و زمان انتظار سرویس‌ها می‌شود که این به دلیل فاصله ابر تا مبدا داده‌ها است. علاوه بر این، SPP-PSO هزینه سرویس بالاتری در مقایسه با SPP-ICA گزارش می‌دهد که این به دلیل پرداخت هزینه برای استفاده از منابع ابری است. به‌طور میانگین، SPP-ICA بهترین نتایج را دارد و نسبت به SPP-PSO، CSA-FSPP و FSP-ODMA به ترتیب ۹.۷٪، ۲۱.۶٪ و ۱۷.۴٪ برتری را گزارش می‌دهد.

۶-نتیجه‌گیری

این مقاله با استفاده از رویکردهای مبتنی بر فراابتکاری و در نظر گفتن مفهوم مدل برنامه‌ریزی خودمختار راه‌حلی کارآمد برای حل مشکل استقرار سرویس پیشنهاد داد. مدل برنامه‌ریزی خودمختار پیشنهادی مبتنی بر MADE-k توسعه یافته است جائیکه تصمیم‌گیری در مورد مکان‌یابی سرویس اینترنت اشیا با استفاده از ICA به‌عنوان یک رویکرد فراابتکاری انجام می‌شود. ICA پیشنهادی با ایجاد مصالحه بین اهداف مختلف (یعنی، بهره‌وری از منابع، هزینه سرویس و هزینه تأخیر) مسئله را به‌عنوان یک مسئله چند-هدفه مدلسازی می‌کند. با توجه به فاکتورهای مختلف استفاده شده در تابع هدف، الگوریتم پیشنهادی عملکرد کلی سیستم را با ایجاد همکاری بین ابر و مه بهبود داده و باعث افزایش رضایت کیفیت خدمات شده است. علاوه بر این، الگوریتم پیشنهادی با استقرار منابع به‌صورت توزیع شده موفق به صرفه‌جویی در منابع و پذیرش تعداد درخواست‌های بیشتری شده است. شبیه‌سازی‌های گسترده روی یک محیط مه مصنوعی بر مبنای معیارهای مختلف برتری الگوریتم پیشنهادی را در مقایسه با سایر الگوریتم های فراابتکاری اثبات کرده است. به‌عنوان یک کار آینده، ما قصد داریم رویکردهای یادگیری تقویتی عمیق[3] (DRL) را به همراه سیاست حداکثرسازی پاداش تجمعی بلندمدت برای مکان‌یابی سرویس‌ها در نظر بگیریم.

مراجع

[1] R. Basir, S. Qaisar, M. Ali, M. Aldwairi, M. I. Ashraf, A. Mahmood, and M. Gidlund, "Fog computing enabling industrial internet of things: State-of-the-art and research challenges," *Sensors*, vol. 19, no. 21, p. 4807, 2019, doi: 10.3390/s19214807.

[2] T. Joyce and J. M. Herrmann, "A review of no free lunch theorems, and their implications for metaheuristic optimization," *Nature-inspired algorithms and applied optimization*, vol. 744, pp. 27-51, 2018, doi: 10.1007/978-3-319-67669-2_2.

[3] Deep Reinforcement Learning

[3] S. Marchal, M. Miettinen, T. D. Nguyen, A. R. Sadeghi and N. Asokan,"Audi: Toward autonomous IoT device-type identification using periodic communication,". *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 6, pp. 1402-1412, 2019, doi: 10.1109/JSAC.2019.2904364..

[4] M. Etemadi, M. Ghobaei-Arani and A. Shahidinejad, "Resource provisioning for IoT services in the fog computing environment: An autonomic approach,". *Computer Communications*, vol. 161, pp. 109-131, 2020, doi: 10.1016/j.comcom.2020.07.028 .

[5] S. Dlamini, J. Mwangama, N. Ventura and T. Magedanz, "Design of an Autonomous Management and Orchestration for Fog Computing," *International Conference on Intelligent and Innovative Computing Applications (ICONIC)*, Mon Tresor, Mauritius, 2018, pp. 1-6, doi: 10.1109/ICONIC.2018.8601272.

[6] M. Ghobaei-Arani, M. Shamsi and A. A. Rahmanian, "An efficient approach for improving virtual machine placement in cloud computing environment,". *Journal of Experimental & Theoretical Artificial Intelligence*, vol. 29, no. 6, pp. 1149-1171, 2017, doi: 10.1080/0952813X.2017.1310308.

[7] M. Slabicki and K. Grochla, "Performance evaluation of CoAP, SNMP and NETCONF protocols in fog computing architecture," *NOMS 2016 - 2016 IEEE/IFIP Network Operations and Management Symposium*, Istanbul, Turkey, 2016, pp. 1315-1319, doi: 10.1109/NOMS.2016.7503010.

[8] Z. Rezazadeh, D. Rahbari and M. Nickray, "Optimized Module Placement in IoT Applications Based on Fog Computing," *Electrical Engineering (ICEE), Iranian Conference on, Mashhad*, Iran, 2018, pp. 1553-1558, doi: 10.1109/ICEE.2018.8472469.

[9] H. Maaranen, K. Miettinen and A. Penttinen, "On initial populations of a genetic algorithm for continuous optimization problems," *Journal of Global Optimization*, vol. 37, no. 3, pp. 405-436, 2007, doi: 10.1007/s10898-006-9056-6.

[10] H. Sami and A. Mourad, "Dynamic On-Demand Fog Formation Offering On-the-Fly IoT Service Deployment," in *IEEE Transactions on Network and Service Management*, vol. 17, no. 2, pp. 1026-1039, June 2020, doi: 10.1109/TNSM.2019.2963643.

[11] T. Dokeroglu, E. Seinc, T. Kucukyilmaz and A. Cosar, "A survey on new generation metaheuristic algorithms,". *Computers & Industrial Engineering*, vol. 137, p. 106040, 2019, doi: 10.1016/j.cie.2019.106040.

[12] B. V. Natesha and R. M. R. Guddeti, "Adopting elitism-based Genetic Algorithm for minimizing multi-objective problems of IoT service placement in fog computing environment,". *Journal of Network and Computer Applications*, vol. 178, p. 102972, 2021, doi: 10.1016/j.jnca.2020.102972.

[13] F. A. Salaht, F. Desprez and A. Lebre, "An overview of service placement problem in fog and edge computing,". *ACM Computing Surveys (CSUR)*, vol. 53, no. 3, pp. 1-35, 2020, doi: 10.1145/3391196.

[14] B. Jia, H. Hu, Y. Zeng, T. Xu and Y. Yang, "Double-matching resource allocation strategy in fog computing networks based on cost efficiency," *Journal of Communications and Networks*, vol. 20, no. 3, pp. 237-246, 2018, doi: 10.1109/JCN.2018.000036.

[15] A. Yousefpour et al., "FOGPLAN: A Lightweight QoS-Aware Dynamic Fog Service Provisioning Framework," in *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 5080-5096, June 2019, doi: 10.1109/JIOT.2019.2896311.

[16] Y. Chen, Z. Li, B. Yang, K. Nai and K. Li, "A Stackelberg game approach to multiple resources allocation and pricing in mobile edge computing,". *Future Generation Computer Systems*, vol. 108, pp. 273-287, 2020, doi: 10.1016/j.future.2020.02.045.

[17] M. Salimian, M. Ghobaei-Arani and A. Shahidinejad, "Toward an autonomic approach for Internet of Things service placement using gray wolf optimization in the fog computing environment," *Software: Practice and Experience*, vol. 51, no. 8, pp. 1745-1772, 2021, doi: 10.1002/spe.2986.

[18] M. Salimian, M. Ghobaei-Arani and A. Shahidinejad, “An Evolutionary Multi-objective Optimization Technique to Deploy the IoT Services in Fog-enabled Networks: An Autonomous Approach,”. *Applied Artificial Intelligence*, vol. 36, no. 1, 2022, doi: 10.1080/08839514.2021.2008149

[19] C. Liu, J. Wang, L. Zhou and A. Rezaeipanah, “Solving the multi-objective problem of IoT service placement in fog computing using cuckoo search algorithm,”. *Neural Processing Letters*, vol. 54, no. 3, pp. 1823-1854, 2022, doi: 10.1007/s11063-021-10708-2.

[20] D. Zhao, Q. Zou and M. Boshkani Zadeh, “A QoS-Aware IoT Service Placement Mechanism in Fog Computing Based on Open-Source Development Model,”. *Journal of Grid Computing*, vol. 20, no. 2, pp. 1-29, 2022, doi: 10.1007/s10723-022-09604-3.

[21] M. Ghobaei-Arani and A. Shahidinejad, “A cost-efficient IoT service placement approach using whale optimization algorithm in fog computing environment,”. *Expert Systems with Applications*, vol. 200, p. 117012, 2022, doi: 10.1016/j.eswa.2022.117012.

[22] M. Azimzadeh, A. Rezaee, S. J. Jassbi and M. Esnaashari, “Placement of IoT services in fog environment based on complex network features: a genetic-based approach,”. *Cluster Computing*, vol. 25, pp. 3423-3445, 2022, doi: 10.1007/s10586-022-03571-w.

[23] R. Basir, S. Qaisar, M. Ali, M. Aldwairi, M. I. Ashraf, A. Mahmood and M. Gidlund, “Fog computing enabling industrial internet of things: State-of-the-art and research challenges,”. *Sensors*, vol. 19, no. 21, p. 4807, 2019, doi: 10.3390/s19214807.

[24] S. Marchal, M. Miettinen, T. D. Nguyen, A. -R. Sadeghi and N. Asokan, "AuDI: Toward Autonomous IoT Device-Type Identification Using Periodic Communication," in *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 6, pp. 1402-1412, June 2019, doi: 10.1109/JSAC.2019.2904364.

[25] M. Etemadi, M. Ghobaei-Arani and A. Shahidinejad, “Resource provisioning for IoT services in the fog computing environment: An autonomic approach,”. *Computer Communications*, vol. 161, pp. 109-131, 2020, doi: 10.1016/j.comcom.2020.07.028.

[26] S. Dlamini, J. Mwangama, N. Ventura and T. Magedanz, "Design of an Autonomous Management and Orchestration for Fog Computing," *International Conference on Intelligent and Innovative Computing Applications (ICONIC)*, Mon Tresor, Mauritius, 2018, pp. 1-6, doi: 10.1109/ICONIC.2018.8601272.

[27] M. Ghobaei-Arani, M. Shamsi and A. A. Rahmanian, “An efficient approach for improving virtual machine placement in cloud computing environment,”. *Journal of Experimental & Theoretical Artificial Intelligence*, vol. 29, no. 6, pp. 1149-1171, 2017, doi: 10.1080/0952813X.2017.1310308.

[28] A. Shahidinejad, M. Ghobaei-Arani and M. Masdari, “Resource provisioning using workload clustering in cloud computing environment: a hybrid approach,”. *Cluster Computing*, vol. 24, no. 1, pp. 319-342, 2021, doi: 10.1007/s10586-020-03107-0.

[29] M. S. Aslanpour, S. E. Dashti, M. Ghobaei-Arani and A. A. Rahmanian, “Resource provisioning for cloud applications: a 3-D, provident and flexible approach,”. *The Journal of Supercomputing*, vol. 74, no. 12, pp. 6470-6501, 2018, doi: 10.1007/s11227-017-2156-x.