ICTF : زمان‌بندی وظایف مبتنی بر الگوریتم رقابت استعماری در محیط محاسبات مه

*حسین مؤمنی[۱] و علی یاوری[۲]
[۱] گروه مهندسی کامپیوتر، دانشگاه گلستان، گرگان، ایران
[۲] گروه مهندسی کامپیوتر، دانشگاه اراک، اراک، ایران

چکیده

محاسبات مه برای حل چالش‌های متعدّد محیط محاسبات ابری مانند زمان تأخیر بالا، ظرفیّت کم و نقص شبکه ارایه شده است. در محیط محاسبات مه، دستگاه‌های اینترنت اشیا به‌عنوان یک ماشین محاسباتی کوچک با قابلیّت پردازش و ارسال و دریافت اطّلاعات، زیرساخت یک مه را تشکیل می‌دهند. در محیط مه، پردازش کارها و وظایف و ذخیرهٔ داده‌های اینترنت اشیا به‌جای ارسال برای سرورهای دور در مراکز دادهٔ ابری به‌صورت محلی در دستگاه‌های اینترنت اشیا انجام می‌شود؛ که این قابلیّت منجر به ارایهٔ پاسخ سریع‌تر، با تأخیر کمتر و افزایش کیفیّت ارایه خدمات در محیط مه می‌شودY بنابراین، می‌توان گفت که محاسبات مه بهترین انتخاب برای فعال کردن اینترنت اشیا در راستای ارایهٔ خدمات کارآمد و امن برای بسیاری از کاربران در لبهٔ شبکه محسوب می‌شود. در محاسبات مه، مدیریّت منابع و زمان‌بندی کار با در نظر گرفتن محدودیّت‌های انرژی، زمان، تأخیر چالش بزرگی محسوب می‌شود. در این مقاله راهکار زمان‌بندی وظایف مبتنی بر الگوریتم رقابت استعماری برای محاسبات مه ارائه شده است. در راه‌کار پیشنهادی جمعیّت اولیّه به‌طور تصادفی شامل وظایف و ماشین‌ها شکل می‌گیرد و تابع ارزیابی بر اساس معیارهای انرژی، زمان و هزینهٔ تعریف‌شده و با به‌کارگیری دو عملگر جذب و انقلاب، الگوریتم زمان‌بندی وظایف مبتنی بر رقابت استعماری (ICTF) ارایه می‌شود. نتایج شبیه‌سازی نشان می‌دهد که ICTF در معیارهای Makespan ، بهره‌وری منابع، مصرف انرژی و انرژی باقی‌مانده نسبت‌به سایر روش‌های مشابه کارایی بالاتری دارد.

واژگان کلیدی: محاسبات مه، مدیریّت منابع، زمان‌بندی وظایف، الگوریتم رقابت استعماری.

ICTF: Imperialist Competitive Algorithm-based Task Scheduling in Fog Computing

Hossein Momeni[1*] & Ali Yavari[2]
[1]Computer Engineering Department, Golestan University, Gorgan, Iran
[2] Computer Engineering Department, Arak University, Arak, Iran

Abstract

Fog computing address numerous cloud computing challenges such as high latency, low capacity and network failure. The cloud computing infrastructure includes a large number of IoT devices with the ability to process in the cloud environment. In fog computing, processing and storage provided on IoT devices locally instead of remote servers; therefore, fog computing is the best choice to enable IoT in order to provide efficient, faster and secure services for many users on the edge of the network. Fog computing have a variety of challenges. One of these important challenges is resource management and task scheduling such that solving this problem has a great impact on system efficiency and service quality. In this paper, we present a task scheduling approach based on the imperialist competition algorithm namely, Imperialist Competitive Algorithm-based Task Scheduling in Fog Computing (ICTF). In the proposed method, we consider the search space as a directional graph. Assume that each task that contains a set of tasks is a graph with a root node and an end leaf node whose middle nodes are the task set. Each path in this graph that starts at the root node and ends at the leaf is a solution represented by a string. This solution is modeled as a country. Therefore, in the proposed method, the

* Corresponding author
*نویسندهٔ عهده‌دار مکاتبات

• تاریخ ارسال مقاله: ۱۳۹۹/۹/۱۷ • تاریخ پذیرش: ۱۴۰۰/۳/۹ • تاریخ انتشار: ۱۴۰۲/۵/۲۰ • نوع مطالعه: پژوهشی

[Downloaded from jsdp.rcisp.ac.ir on 2026-08-11]
[DOI: 10.61186/jsdp.20.1.25]

concept of country includes the tasks of a job along with the fog nodes that are assigned to these tasks. The initial population consists of a random number of these solutions. ICTF presents a cost function consisting of three important criteria for assessing the initial population of countries and determining the imperialists and colonies countries include energy, execution time and execution cost. The assimilation operation is performed on two different members of countries, namely the imperialists and colonies country, and two new types of members are created called children. The countries participating in this process are among the best countries and are selected using the cost function. In this process, the best offspring produced are passed on to the next generation, and this operation continues until the final population of the countries is obtained. The assimilation operator has different models and, in this article, we use the two-point assimilation operator. The name of the revolution operator used in this algorithm is the inverse of the task. This operator randomly selects two tasks belonging to a fog node and moves them together. The above operation is repeated until the population converges and reaches the final answer. We show that our proposed approach is more efficient in terms of makespan, resource utilization, energy consumption and remaining energy compared to the similar approach.

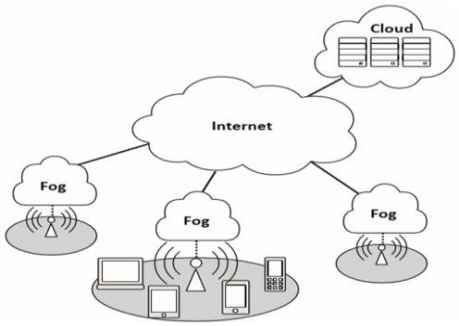
Keywords: Fog computing, resource management, task scheduling, imperialist competitive algorithm.

۱- مقدمه

محیط محاسبات مه یک معماری جدید محاسباتی برای مدیریّت داده‌های تولیدشده با اینترنت اشیا است که شباهت بسیاری به محیط محاسبات ابری دارد. مهم‌ترین مشکل در محاسبات ابری، عدم پاسخ مناسب به درخواست‌های بی‌درنگ است که در محاسبات مه برطرف شده است. بنابراین، باتوجّه‌به‌این‌که ضعف‌های محاسبات ابری در محاسبات مه وجود ندارد، در سال‌های اخیر پژوهشگران بسیاری راهکارهای مفیدی در محیط محاسبات مه ارایه نموده‌اند[۱].

همان‌طورکه شکل (۱) نشان می‌دهد محاسبات مه لایهٔ میانی مجازی بین کاربران و ابر است. مه، یک فناوری مجازی است که شبیه به ابر است و شامل ارائه اطّلاعات، محاسبات، ذخیره‌سازی و خدمات شبکه بین کاربران و سرورهای ابر است. محاسبات مه دارای قابلیّت‌های حرکت منابع محاسباتی، پروتکل‌های ارتباطی، نامتجانس بودن رابط، استفاده از ابر و تجزیه و تحلیل داده‌های توزیع‌شده در محل استقرار برنامه‌های کاربردی است[2,3].

با تنوع درخواست‌ها و وجود منابع متعدد محاسباتی یکی از مهم‌ترین چالش‌ها در محاسبات مه، زمان‌بندی کار و وظایف است. منظور از زمان‌بندی کار در محاسبات مه، این است که کارهای درخواستی از کاربران مختلف بر اساس یک راهبرد مشخص به گره‌های مه موجود تخصیص داده شود به گونه‌ای که زمان تکمیل درخواست کاربر و انرژی مصرفی تجهیزات سمت کاربر به کمترین حدّ ممکن برسد[4-6]. محیط محاسبات مه یک زیرساخت مناسب را برای به‌اشتراک‌گذاری پویای منابع در مقیاس‌های مختلف فراهم می‌کند. وظایف ارسال‌شده به مه به‌طور عموم، به‌صورت نامتجانس بوده و برای اجرا نیازمندی‌های مختلفی مانند میزان حافظه، پردازنده و غیره دارد. در مسألهٔ زمان‌بندی وظایف هدف یافتن بهترین منابع برای پردازش وظایف است. ازآنجاکه برای هر تعداد از وظایف چندین پردازنده می‌توانند کاندید باشند، یافتن بهترین نگاشت میان منابع و وظایف امری زمان‌بر است. مسأله زمان‌بندی وظایف با در نظر داشتن چند پردازنده یک مسألهٔ NP-Complete است[۷].



(شکل- ۱): معماری محیط محاسبات مه[۲]
(Figure-1): Fog Computing Architecture [2]

منابعی مانند حافظه، پردازنده و غیره باارزش هستند و استفادهٔ بهینه‌تر از آن‌ها یک چالش همیشگی در محیط‌های توزیعی و به‌ویژه در محیط محاسبات مه است[8,9]. ازاین‌رو مسئلهٔ زمان‌بندی کار و وظایف در رایانش مه، مسئله‌ای بسیار مهم محسوب می‌شود و اغلب زمان‌بندی‌های سنّتی پاسخ و خروجی مناسبی تولید نمی‌کنند[10,11]. از طرفی، محیط محاسبات مه، سامانه پیچیده با تعداد بسیار زیاد منابع مشترک برای درخواست‌های پیش‌بینی‌نشده است که از وقایع بیرونی غیرقابل‌کنترل تأثیر می‌گیرند. مدیریّت منابع مه نیاز به سیاست‌های پیچیده و تصمیم‌گیری‌هایی برای بهینه‌سازی چند منظوره دارد[12,۱۳] .

الگوریتم‌های بهینه‌سازی شامل مجموعه‌ای از الگوریتم‌های هوش مصنوعی هستند که بر فضای ناشناختهٔ مسأله تسلّط یافته و با روش جستجو راه‌حل‌های مناسبی ایجاد می‌کنند[14]. الگوریتم‌های

[Downloaded from jsdp.rcisp.ac.ir on 2026-08-11]
[DOI: 10.61186/jsdp.20.1.25]

بهینه‌سازی از لحاظ ماهیّت دارای مزایا و معایبی نسبت‌به همدیگر هستند؛ ازاین‌رو در کارایی نیز متفاوتند[15]. انتخاب یک الگوریتم بهینه‌سازی مناسب به‌طور کامل وابسته به شرایط و ماهیّت مسأله است. در مسألهٔ زمان‌بندی وظایف که یک مسألهٔ NP-Complete است و معیارهای متعدّدی تحت‌تأثیر کارایی الگوریتم قرار دارند؛ به همین دلیل، تابع هدفی که باید با الگوریتم بهینه‌سازی شود، نیز، باید از لحاظ ماهیّت منطبق بر ماهیّت الگوریتم باشد[16,17].

الگوریتم‌های فراابتکاری مبتنی بر جمعیّت اولیّه نظیر الگوریتم رقابت استعماری در زمینهٔ زمان‌بندی کارایی مناسبی از خود نشان داده‌اند[18]. در این مقاله به ارایهٔ روش پیشنهادی زمان‌بندی وظایف مبتنی بر الگوریتم رقابت استعماری در محیط محاسبات مه با عنوان ICTF[1] می‌پردازیم. روش پیشنهادی بر اساس کمینه شدن تابع هدف متشکل از سه معیار انرژی مصرفی، هزینهٔ اجرا و زمان اجرای وظایف است. این روش، نخست، جمعیّت اولیّه از راه‌حل‌های ممکن جهت اجرای یک مجموعهٔ وظیفه را به‌عنوان کشورهای اولیّه تعیین می‌کند؛ سپس هر کشور با تابع هزینهٔ پیشنهادی ارزیابی می‌شود. کشورها بر اساس نتیجه تابع هدف به دو دستهٔ استعمارگر و مستعمره تبدیل شده و در ادامه روال بهینه‌سازی با دو عملگر جذب و انقلاب انجام می‌شود. عملگر جذب وظیفهٔ نزدیک کردن مستعمرات به استعمارگر را دارد. در اینجا استعمارگر کشوری است که کمترین مقدار تابع هزینه را داشته است و مستعمرات سایر کشورها هستند که با هدف بهبود تابع هزینه به‌سمت استعمارگر حرکت می‌کنند. عملگر انقلاب برای خروج از بهینه محلی طراحی شده است. نوآوری این مقاله ارایهٔ راهکار بهینه مبتنی بر الگوریتم رقابت استعماری برای زمان‌بندی وظایف است که در مقایسه با سایر روش‌های مطرح در دنیا در معیارهای زمان و انرژی و هزینه عملکرد بهتری دارد.

ادامهٔ این مقاله به صورت زیر سازمان‌دهی شده است: در بخش دوم، کارهای پیشین ارایه می‌شود. در بخش سوم الگوی سامانه تشریح می‌شود. بخش چهارم به بیان روش پیشنهادی می‌پردازد. بخش پنجم، ارزیابی راه‌کار پیشنهادی را نشان می‌دهد و در بخش ششم نتیجه‌گیری ارائه خواهد شد.

[1] Imperialist Competitive Algorithm-based Task Scheduling in Fog Computing

۲-کارهای پیشین

محاسبات مه یک الگوی جدید محاسباتی است که توجّه بسیاری را به خود جلب کرده است. ارائهٔ عملکرد محاسبات رضایت‌بخش و کیفیّت خدمات (سرویس) یک چالش بزرگ در محیط محاسبات مه است. در [۱۹] الگوریتم I-Apriori با هدف بهبود الگوریتم Apriori ارائه شده است. در این روش قوانین انجمنی با الگوریتم I-Apriori تولید می‌شود که به‌عنوان شاخص مهم عمل می کند. قانون انجمنی بیان می‌دارد اگر پردازندهٔ ماشین مجازی i از زمان لازم برای پردازش وظیفه j بیشتر و حافظهٔ آزاد این ماشین مجازی از اندازه وظیفه j بیشتر باشد، آنگاه ماشین مجازی i را به وظیفهٔ j تخصیص دهد.

روش جدید با دسته‌بندی ماشین‌های مجازی و تولید قوانین انجمنی بر اساس این دسته‌ها، کمیت و کیفیّت قوانین را بهبود می دهد. نتایج شبیه‌سازی نشان داده که روش I-Apriori در زمان کل اجرای کار و میانگین زمان انتظار، عملکرد بهتری نسبت‌به سایر الگوریتم های مشابه دارد.

در [۲۰] الگوریتم زمان‌بندی کار در محاسبات مه با استفاده از الگوریتم بهینه‌سازی زنبور ارائه شد. الگوریتم بهینه‌سازی زنبور یکی از الگوریتم‌های هوش جمعی است که به‌عنوان یک راهکار حل مسائل بهینه‌سازی مطرح شد. نتایج شبیه‌سازی این روش نشان می‌دهد که زمان اجرا، زمان پاسخ و زمان انتظار نسبت‌به روش‌های دیگر بهبود یافته است.

در [۲۱] الگوریتم زمان‌بندی کار در محیط محاسبات مه با استفاده از روش بهینه‌سازی اجتماع مورچگان ارائه شده است. هر مورچه نشان‌دهندهٔ تعدادی کار بر تعدادی ماشین است. جمعیّت اولیّه از مورچه‌ها بر اساس میزان توازن باری که هر مورچه ایجاد می‌کند، محاسبه خواهد شد. در ادامه مورچه‌ها بر اساس مقدار فرمون که همان تابع برازندگی است به سمت مورچه‌های بهینه حرکت کرده، و در صورت وجود مانع کمی مسیر خود را تغییر می‌دهند. این فرآیند تا زمان رسیدن به غذا تکرار خواهد شد. نتایج شبیه‌سازی نشان‌دهندهٔ بهبود مناسب توازن بار است.

در [۲۲] الگوریتم ترکیبی NBIHA برای بهبود کیفیّت سرویس در محاسبات مه ارائه شده است. این الگوریتم ترکیبی از بهینه‌سازی ذرّات اصلاح‌شده (MPSO[2]) و بهینه‌سازی گربهٔ اصلاح‌شده

[2] Modified particle swarm optimization (MPSO)

[Downloaded from jsdp.rcisp.ac.ir on 2026-08-11]
[DOI: 10.61186/jsdp.20.1.25]

(MCSO[1]) اســـت. در روش پیشـــنهادی از الگـــوریتم MPSO برای زمان‌بندی وظایف میـان دسـتگاه‌هـای مـه استفاده شده است و از ترکیبـی MPSO و MCSO بـرای مدیریّت منابع در سطح دسـتگاه مـه اسـتفاده مـی‌شـود. هدف اصلی این روش، کاهش زمان متوسط پاسخ و بهبود بهره‌برداری از منابع است. تابع هزینهٔ این روش بر اسـاس زمان پاسخ است. نتایج نشان مـی‌دهـد کـه ایـن روش از نظر مصرف انرژی، زمان اجرا و میـانگین زمـان پاسـخ در مقایسه با سایر الگوریتم‌های مشابه نتایج بهتری دارد.

در [۲۳] برای زمان‌بندی کـار و افـزایش تـوازن بـار از الگوریتم بهینه‌سـازی ژنتیـک بهبودیافتـه اسـتفاده شـده است. در این روش تمامی منابع مه به سـه دسـتهٔ منـابع ذخیـرهٔ داده، منـابع محاسـباتی و منـابع پهنـای بانـد دسته‌بندی می‌شوند. الگوی توازن بار این روش بر اسـاس کارکرد پردازنده، میزان حافظ آزاد و پهنای بانـد مصـرفی است و تـابع برازنـدگی کـه از کرومـزوم‌هـا اعتبارسـنجی می‌کند نیـز، بـر ایـن اسـاس تنظـیم شـده اسـت. نتـایج پیاده‌سازی نشان‌دهندهٔ این است که روش پیشـنهادی در عناصر تشکیل‌دهندهٔ تابع برازندگی نسبت‌به سایر روش‌ها کارایی بالاتری دارد.

در [۲۴] روش زمان‌بندی ترکیبی FPFTS مبتنـی بـر الگوریتم بهینه‌سازی ازدحـام ذرّات و منطـق فـازی بـرای زمان‌بندی وظایف در محاسبات مه ارائه شد. در این روش انتخاب منـابع بـا منطـق فـازی و بهبـود تـابع هزینـه بـا الگوریتم ازدحام ذرات انجام می‌شود. منطـق فـازی بـرای انتخاب منابع از توان پردازشی، میزان حافظهٔ آزاد، میـزان پهنـای بانـد مصـرفی دسـتگاه و مجموعـه‌ای از قـوانین استنتاج فازی استفاده می‌کند. الگوریتم ازدحام ذرّات نیز از تابع هزینهٔ مبتنی بر این سه فاکتور جهت بهینه‌سـازی زمان‌بنـدی اسـتفاده می‌کنـد. نتـایج آزمایش‌هـا نشـان می‌دهد روش FPFTS در مقایسه با سایر الگوریتم‌هـا در تأخیر و پهنای باند مصـرفی و زمـان اجـرا نتـایج بهتـری داشته است.

در [۲۵] یک روش آگاه از انـرژی بـا اسـتفاده از روش پویایی ولتاژ و مقیاس فرکـانس (DVFS[2]) بـرای کـاهش مصرف انرژی در محاسبات مه ارائه شده اسـت. عـلاوه‌بـر این، به‌منظـور زمان‌بنـدی مناسـب وظـایف، از الگـوریتم تکاملی علف‌های هرز مهـاجم ترکیبـی (IWO-CA[3]) نیـز استفاده شده است. در ایـن روش بـا تـابع برازنـدگی کـه نشان‌دهندهٔ میزان مصرف انرژی است، تعیـین برازنـدگی انجام می‌شود و در هر تکرار از الگوریتم علف هرز مهاجم، الگوریتم DVFS که وظیفهٔ انتخاب بهترین جواب‌ها را بـا در نظر گرفتن میزان پویایی ولتاژ و مقیـاس فرکـانس بـر عهده دارد، اجرا می‌شود. نتایج شبیه‌سازی نشان می‌دهـد که الگوریتم پیشنهادی از نظـر مصـرف انـرژی، نسـبت‌به الگـوریتم هـای HEFT-B[4] و HEFT-T [۲۶]، EES[5] [۲۷]، DEWTS[6] [۲۸] بهتر عمل می‌کند.

در الگوریتم HEFT-B اساس کار بـر مبنـای انتخـاب بزرگ‌ترین وظیفه و تخصیص قـوی‌ترین پردازنـده بـه آن اسـت. در HEFT-T وظـایف حیـاتی را روی پردازنـده‌ای زمان‌بندی می‌کند که زمان اجرای کل وظایف مهم را بـه کمترین حـد برسـاند. در EES کارهـای حیـاتی نخسـت، زمان‌بندی و اجرا می‌شوند و بـرای کارهـای غیرحیـاتی از روش slack room استفاده می‌شود. در این روش کارهای غیرحیـاتی در محلّـی جـدا جمـع‌آوری شـده و بـا یـک زمان‌بند عمومی زمان‌بندی می‌شوند. الگـوریتم DEWTS نخست، ترتیب زمان‌بندی اولیّـه همـهٔ کارهـا را محاسـبه می‌کند و کل زمان و مهلت را بر اساس الگـوریتم HEFT به دست می‌آورد. سپس، با شناسـایی پردازنـده‌هـای کـم مصرف و بـا در نظـر گـرفتن زمان‌بنـدی و مهلـت زمـانی وظایف به پردازنده‌ها تخصیص داده می‌شوند.

در [۲۹] از منطق فازی برای زمان‌بندی کار بـا هـدف افزایش توازن بار در محاسبات مه استفاده شده است. این الگوریتم مبتنـی بـر منطـق فـازی بـرای انجـام تجزیـه و تحلیل پیوند ارتباطی بـه‌عنوان اتُصـال بـه‌همراه مـدیریت ترافیک پیاده‌سازی شده است. در این روش کیفیّت مسیر ارتباطی و ترافیک به‌عنوان دو مؤلّفه جهت انتخاب مسـیر مناسب و زمـان اجـرای وظیفـه بـه‌عنوان مؤلّفـهٔ انتخـاب ماشـین مجـازی جهـت اجـرای کـار و عمـل زمان‌بنـدی بـه‌عنوان معیارهـای فـازی اسـتفاده شـده اسـت. نتـایج شبیه‌سازی این روش بیانگر کـارایی بـالاتر در تـوازن بـار نسبت‌به سایر روش‌های موجود است.

با مطالعهٔ روش‌های موجود در زمینهٔ زمان‌بنـدی کـار در محاسبات مه در می‌یابیم که بیشتر روش‌های موجود

با تمرکز بر روی یک معیـار نسـبت‌به سـایر روش‌هـا، بهبود ایجاد کرده‌اند. در برخی از روش‌ها بهبود یک معیار کمّی یا کیفی مانند زمـان اجـرای کـار در ازای از دسـت دادن شرایط مناسب برای معیارهای دیگر بـوده اسـت. از این رو مسألهٔ زمان‌بندی کار در محاسبات مـه بایـد بـا در نظر گرفتن چندین شاخص و معیار حل شود. در ادامه بـا

1 Modified cat swarm optimization
2 Dynamic Voltage and Frequency Scaling
3 Invasive Weed Optimization and Culture algorithm (IWO-CA)
4 Earliest-Finish-Time
5 Enhanced Energy-Efficient Scheduling
6 DVFS-enabled Energy-efficient Workflow Task Scheduling algorithm

[Downloaded from jsdp.rcisp.ac.ir on 2026-08-11]
[DOI: 10.61186/jsdp.20.1.25]

به‌کارگیری الگوریتم رقابت‌های استعماری که در مسایل بهینه‌سازی به‌طور گسترده استفاده می‌شود[۳۰]، راهکاری برای زمان‌بندی وظایف ارایه خواهد شد.

۳- الگوی سامانه

۳-۱- الگوی منابع در مه

محیط محاسبات مه شامل مجموعه‌ای از منابع، با عنوان ماشین‌های فیزیکی است که طبق رابطهٔ (۱) تعریف می‌شود:

$$PM=\{ [\![pm]\!]_j, j=1,2,\ldots,n\} \quad n=|PM| \qquad (۱)$$

که در این رابطه n برابر کل تعداد ماشین‌های فیزیکی موجود در محیط محاسبات مه است. هر ماشین فیزیکی با مجموعه ویژگی‌های رابطهٔ (۲) مشخص می‌شود:

$$[\![PM]\!]_j=\{M_j,R_j,E_j,[\![VM]\!]_j\} \qquad (۲)$$

که در این رابطه M_j ظرفیّت پردازندهٔ ماشین فیزیکی jام بر حسب میلیون دستور در ثانیه (MIPS)، R_j ظرفیّت حافظهٔ ماشین فیزیکی jام، E_j بیشترین توان انرژی ماشین فیزیکی و $[\![VM]\!]_j$ مجموعه ماشین‌های مجازی مستقر بر روی ماشین فیزیکی jام است. مجموعه ماشین‌های مجازی مستقر بر روی هر ماشین فیزیکی طبق رابطهٔ (۳) تعریف می‌شود:

$$[\![VM]\!]_j=\{ [\![vm]\!]_{j1}, [\![vm]\!]_{j2},\ldots, [\![vm]\!]_{jk}\} k=|[\![VM]\!]_{j1}| \qquad (۳)$$

که در این رابطه K برابر کل تعداد ماشین‌های مجازی مستقر بر روی ماشین فیزیکی j ام است.

ویژگی‌های یک ماشین مجازی طبق رابطهٔ (۴) بیان می‌شود:

$$[\![VM]\!]_j=\{[\![CPUUsage]\!]_{jk},R_{jk},\ldots,P_{jk}\} \qquad (۴)$$

که در این رابطه $[\![CPUUsage]\!]_{jk}$ میزان مصرف پردازنده، R_{jk} ظرفیّت حافظه‌ای که به ماشین مجازی $[\![vm]\!]_{jk}$ اختصاص یافته است، P_{jk} زمان آماده بودن ماشین مجازی است.

۳-۲- الگوی وظایف در مه

در محیط محاسبات مه مجموعهٔ کارها به‌صورت رابطه (۵) بیان می‌شود:

$$J=\{j_1,j_2,j_3,\ldots,j_S\} \qquad (۵)$$

در رابطه بالا S تعداد کارهای موجود در مجموعه J است. هر کار شامل مجموعه‌ای از وظایف T است که به‌صورت رابطهٔ (۶) بیان می‌شود:

$$T=\{t_1,t_2,t_3,\ldots,t_W\} \qquad (۶)$$

در رابطهٔ بالا W تعداد وظایف موجود در مجموعه T است. هدف نگاشت بین وظایف و ماشین‌های مجازی، یعنی یک مسأله بهینه‌سازی‌ست. نگاشت این وظایف بر روی ماشین‌های مجازی را می‌توان به‌صورت رابطهٔ (۷) نشان داد:

$$[\![Vm]\!]_k=\{j_{(s_{(t_w)})} \mid s=1,2,3,\ldots,S, w=1,2,3,\ldots,W, k=1,2,3,\ldots,k \qquad (۷)$$

در رابطهٔ بالا $[\![Vm]\!]_k$ حالتی از ماشین مجازی k را نشان می‌دهد که وظیفهٔ w ام از کار sام را برای پردازش قبول کرده است.

۳-۳- الگوی زمان‌بندی

هر $[\![Job]\!]_s$ در مجموعهٔ کارها می‌تواند شامل مجموعه ای از w وظیفه باشد که هر وظیفه w طبق رابطه (۸) به یکی از k ماشین مجازی جهت اجرا ارسال می‌شود.

$$[\![jobs]\!]_{s\ task}=\{[\![jtask]\!]_{s\ 1}^{a},[\![jtask]\!]_{s\ 2}^{b},\ldots,[\![jtask]\!]_{s\ w}^{k}\} \qquad (۸)$$

در این رابطه $[\![jobs]\!]_{s\ task}$ مجموعهٔ وظایف در کار jobs است که بر روی m ماشین مجازی قرار گرفته است. برای مثال $[\![jtask]\!]_{s\ w}^{k}$ به معنای قرار گرفتن وظیفه wام از jobs بر روی ماشین مجازی k است.

همان‌طورکه بیان شد، هر ماشین مجازی $[\![Vm]\!]_k$ در مه می‌تواند یک مجموعه از وظایف را اجرا کند. مجموعه وظایفی که ماشین مجازی k انجام می‌دهد، با رابطهٔ (۹) قابل نمایش است:

$$VMT_k = \{jtask_{ax}^{k}, jtask_{by}^{k}, \ldots, jtask_{sw}^{k}\} \qquad (۹)$$

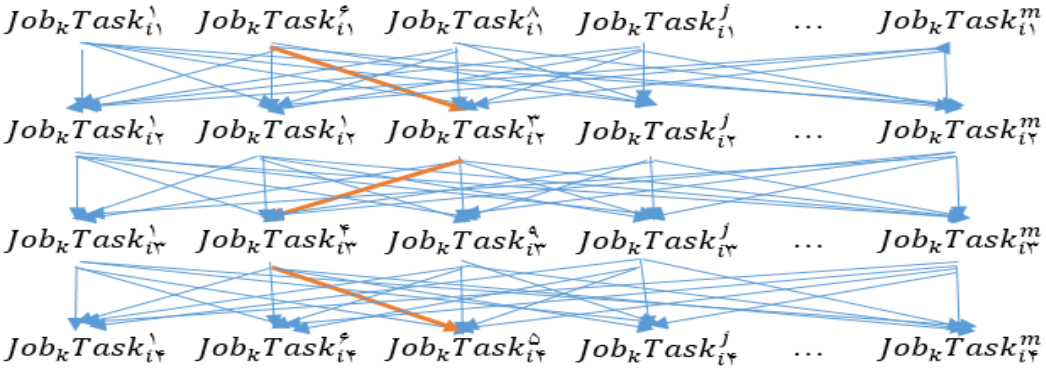
که در این رابطه k بیانگر ماشین مجازی kام است و $jtask_{sw}^{k}$ نشان می‌دهد که وظیفهٔ w از کار s باید با ماشین مجازی k پردازش شود.

۴- راه‌کار پیشنهادی ICTF

در این بخش به ارایهٔ راهکار پیشنهادی برای حل مسألهٔ زمان‌بندی کار در محیط محاسبات مه که به‌اختصار ICTF می‌نامیم، خواهیم پرداخت. ازآنجاکه در زمان‌بندی کارها در محیط محاسبات مه معیارهای متعددی شامل میزان مصرف انرژی، زمان اجرای وظایف، زمان پاسخ، زمان انتظار، میزان مصرف پهنای باند و بهره‌وری منابع مطرح است، به‌عنوان یک مسألهٔ بهینه‌سازی از الگوریتم رقابت استعماری به دلیل سرعت همگرایی بالا، توانایی بهینه‌سازی توابع با تعداد متغیّرهای زیاد و سرعت بالا در یافتن جواب بهینه، استفاده خواهیم کرد.

[Downloaded from jsdp.rcisp.ac.ir on 2026-08-11]

[DOI: 10.61186/jsdp.20.1.25]



(شکل -۲): حالت های مختلف تشکیل یک جواب از وظایف و ماشین‌ها
(Figure-2): Different Solutions of Tasks and Machines

الگوریتم رقابت استعماری مانند دیگر الگوریتم‌های تکاملی، با تعدادی جمعیّت اولیّهٔ تصادفی که هرکدام از آنها یک کشور نامیده می‌شوند، شروع می‌شود. تعدادی از بهترین عناصر جمعیّت به‌عنوان امپراطوری انتخاب می‌شوند. باقی‌ماندهٔ جمعیّت نیز به‌عنوان مستعمره، در نظر گرفته می‌شوند. در بهینه‌سازی، هدف یافتن یک جواب بهینه بر حسب متغیرهای مسئله است. ما یک آرایه از متغیّرهای مسئله را که باید بهینه شوند، ایجاد می‌کنیم و آن را یک کشور می‌نامیم. کشورها به‌طور معمول به‌صورت آرایه و ماتریس الگوسازی می‌شوند.

برای شروع الگوریتم، تعداد N کشور اولیّه را به شکل یک ماتریس ایجاد می‌کنیم. کشورهای استعمارگر از بهترین اعضای این جمعیت (کشورهای دارای کمترین مقدار تابع هزینه) را به‌عنوان امپراطوری انتخاب می‌کنیم. باقی‌ماندهٔ کشورها، مستعمراتی را تشکیل می‌دهند که هرکدام به یک امپراطوری تعلّق دارند. دو عملگر الگوریتم رقابت استعماری شامل جذب و انقلاب، وظیفهٔ ایجاد بهبود میان کشورها را بر عهده دارند. عملگر جذب، کشورهای مستعمره را به استعمارگرها نزدیک می‌کند و درصورتی‌که بهینهٔ محلی رخ دهد، الگوریتم انقلاب با تغییر در ویژگی کشورها، خروج از بهینه محلی را تضمین می‌کند. در ادامه نحوهٔ استفاده از الگوریتم رقابت استعماری و الگوسازی زمان‌بندی وظایف این الگوریتم را ارائه می‌کنیم.

برای حل مسئله زمان‌بندی کار در محاسبات مه، فضای جستجو را به شکل یک گراف جهت‌دار در نظر می‌گیریم. از آنجا که توالی و وابستگی میان وظایف باید پوشش داده شود، برای الگوسازی مسأله از گراف جهت‌دار استفاده شده است. فرض می‌کنیم هر کار که شامل یک مجموعه از وظایف است، یک گراف با یک گره ریشه و یک گره برگ پایانی باشد که گره‌های میانی مجموعهٔ وظایف آن کار هستند. هر مسیر در این گراف که از گره ریشه آغاز شده و به برگ ختم می‌شود، یک راه‌حل است که با یک رشته نمایش داده می‌شود.

برای مثال job_i با ۴ وظیفه را در نظر بگیرید. چندین حالت مختلف برای تخصیص وظایف این کارها به m گره محاسباتی مه وجود دارد. برای مثال، در شکل (۲) با در نظر گرفتن همه راه‌حل های موجود، یک راه‌حل به‌طور مجزا نشان داده شده است.

این راه‌حل به‌عنوان یک کشور الگوسازی شده و در رابطه (۱۰) قابل مشاهده است. بنابراین، در روش ICTF مفهوم کشور شامل وظایف یک کار به‌همراه گره‌های مه که به این وظایف تخصیص داده می‌شوند، است. مجموعهٔ اولیّه شامل تعداد تصادفی از این راه‌حل‌هاست.

مجموعه‌ای از حالات مختلف برای وظایف هر کار

$$c_1 = Job_k Task_{i1}^{6}, Job_k Task_{i2}^{3}, Job_k Task_{i3}^{4}, Job_k Task_{i4}^{5} \quad (۱۰)$$

به‌عنوان مجموعهٔ کشورهای اولیّه در نظر گرفته می‌شود. برای هر جمعیّت N راه‌حل وجود دارد که هرکدام به شکل یک مجموعه از گره‌های مه به‌صورت رابطه (۱۱) است.

$$FNTasks = \{FN_1Tasks, FN_2Tasks, \dots, FN_m\} \quad (۱۱)$$

هر گره مه مسئول اجرای وظایفی است که به آن تخصیص داده شده است؛ به‌عنوان مثال، وظایف گره مه j مانند رابطهٔ (۱۲) نشان داده می‌شود.

$$FN_jTasks = \{JTask_{ax}^{j}, JTask_{by}^{j}, \dots, JTask_{ik}^{j}, \dots JTask_{nr}^{j}\} \quad (۱۲)$$

در این الگوریتم از دو عملگر جذب و انقلاب در فاز تولید مثل برای افزایش تنوع اعضای جمعیّت استفاده می شود.

[Downloaded from jsdp.rcisp.ac.ir on 2026-08-11]
[DOI: 10.61186/jsdp.20.1.25]

۴-۱- تابع هزینه

در روش ICTF یک تابع هزینه متشکّل از سه معیار مهم برای ارزیابی جمعیّت اولیّه از کشورها و تعیین کشورهای مستعمره و استعمارگر را ارایه می‌کنیم. این سه معیار شامل انرژی، زمان اجرای کار و هزینه اجراست. همان‌طورکه مشخص است هرکدام از این سه معیار به‌تنهایی در حالت کمینه مناسب هستند؛ مجموع این سه نیز باید کمینه شود. برای محاسبهٔ انرژی مصرفی با پردازنده‌ها طبق رابطهٔ (۱۳) عمل می‌شود:

$$E = \sum_{t=1}^{t} \sum_{i=1}^{m} E_i(t) \quad (۱۳)$$

به‌طوری‌که

$$E_i(t) = (p_{max} - p_{min}) \times \frac{u_i(t)}{100} + p_{min} \quad (۱۴)$$

$$u_i(t) = \sum_{i=1}^{m} u(i,j) \quad (۱۵)$$

در رابطه‌های (۱۴) و (۱۵) m نشان‌دهندهٔ تعداد ماشین‌های مجازی و t نشان‌دهندهٔ زمان، و $E_i(t)$ بیانگر انرژی مصرفی ماشین مجازی i در زمان t، و $u(i,j)$ نشان‌دهندهٔ میزان مصرف وظیفه j از ماشین i است.

طبق رابطهٔ (۱۶) کاهش انرژی به تعداد منابع استفاده‌شده در کار در زمان t بستگی دارد. این رابطه برای محاسبهٔ کل زمان پردازش کار به‌عنوان بخش دوم از تابع هدف برای کارهای T استفاده خواهد شد.

$$T = \prod_{j=1}^{n} \max(ETC(tj.2) + (tj.4) \quad (۱۶)$$

همان‌طورکه در رابطهٔ (۱۶) مشاهده می‌شود، T کل زمان پردازش و برابر با آخرین زمان شروع کار با زمان پردازش آن در ماشین مجازی است. در رابطهٔ بالا n تعداد کل کارها را نشان می‌دهد. در روش ICTF، هزینهٔ منبع معیار سوم است که باید کمینه‌سازی شود. از این رو قسمت سوم از رابطهٔ تابع هزینه برابر با هزینه منبع در اجرای وظایف است. این هزینه با رابطهٔ (۱۷) محاسبه می‌شود.

$$C = \sum_{i=1}^{m} Costi \times Time_i \quad (۱۷)$$

در این رابطه C بیانگر هزینه کل پردازنده‌هاست. هزینهٔ پردازش وظایف برابر با میزان استفاده آن از پردازنده در واحد زمان است که در اینجا واحد زمان- ثانیه است. بنابراین، مقدار پایین‌تر بهتر خواهد بود.

طبق روابط بالا، ارزش هر راه‌حل (کشور) برابر با مجموع ارزش سه معیار انرژی، زمان و هزینه است و مقدار کمتر دارای ارزش بالاتری است. به عبارت دیگر، کشوری که دارای هزینهٔ کمتری باشد، به‌عنوان راه‌حل بهتر شناخته می‌شود. در رابطهٔ (۱۸) تابع هزینه نشان داده شده است.

که در این رابطه مقادیر p1 ، p2 وp3 نشان‌دهندهٔ ضرایب عددی تأثیر هرکدام از معیارها هستند. این ضرایب عددی بین صفر و یک است که صفر برای

$$Minimize\ Fitness = (T \times p_1) + (E \times p_2) + (C \times p_3) \quad (۱۸)$$

ارزش و تأثیر کمتر و یک برای ارزش و تأثیر بالاتر به کار می‌رود. مقدار P1 برابر با ۰٫۵، مقدار P2 برابر با ۰٫۴ و مقدار P3 برابر با ۰٫۱ در نظر گرفته شده است.

۴-۲- عملگر جذب

در راستای این عملگر، کشور مستعمره، به اندازهٔ x واحد در جهت خط واصل مستعمره به استعمارگر، حرکت کرده و به موقعیّت جدید کشانده می‌شود. x عددی تصادفی با توزیع یکنواخت (یا هر توزیع مناسب دیگر) است. اگر فاصلهٔ میان استعمارگر و مستعمره با d نشان داده شود، آنگاه طبق رابطه (۱۹) داریم:

در این رابطه ß عددی بزرگتر از یک و نزدیک به دو است. یک انتخاب مناسب می‌تواند ß =۲ باشد؛ که البتّه باید با انجام آزمون و خطا مقدار مناسب آن

$$x \sim U(0, ß * d) \quad (۱۹)$$

تعیین شود. ضریب ß≥۱ باعث می‌شود تا کشور مستعمره در حین حرکت به سمت کشور استعمارگر، از جهت‌های مختلف به آن نزدیک شود. همچنین، در کنار این حرکت، یک انحراف زاویه‌ای کوچک نیز با توزیع یکنواخت به مسیر حرکت افزوده می‌شود.

عملگر جذب بر روی دو عضو متفاوت از کشورها، یعنی کشور استعمارگر و کشور مستعمره انجام، و دو نوع جدید از اعضا به نام فرزندان ایجاد می‌شوند. کشور استعمارگر و کشور مستعمرهٔ شرکت‌کننده در این فرآیند از بین بهترین کشورها بوده و با استفاده از تابع هزینه انتخاب می‌شوند. در این فرآیند، بهترین فرزندان تولیدشده به نسل بعد منتقل می‌شوند و این عملیّات آن قدر ادامه می‌یابد تا جمعیّت نهایی از کشورها حاصل شود. عملگر جذب الگوهای مختلفی دارد، که در این مقاله از عملگر جذب دو نقطه‌ای استفاده می‌شود. در این عملگر در والدها که به شکل رشته نمایش داده می‌شوند، دو نقطهٔ تصادفی انتخاب، و رشته والدها به سه بخش تقسیم می‌شود. بخش اول و سوم بدون تغییر به فرزندان منتقل می‌شود، ولی بخش دوم دو کشور والد با هم جابه‌جا شده و به کشورهای جدید به‌عنوان فرزندان منتقل می‌شود. این عملیات در شکل (۳) نشان داده شده است.

[Downloaded from jsdp.rcisp.ac.ir on 2026-08-11]

[DOI: 10.61186/jsdp.20.1.25]

۴-۳- عملگر انقلاب

در اساس، در همهٔ الگوریتم‌های فراابتکاری یک عملگر برای جلوگیری از گیر افتادن در بهینهٔ محلی و در صورت نیاز خروج از بهینه‌های محلّی طراحی می‌شود. در الگوریتم رقابت استعماری این وظیفه مهم بر عهدهٔ عملگر انقلاب است. در واقع، این عملگر با ایجاد تغییر در موقعیّت کشورها از طریق دست‌کاری ویژگی‌های آنها باعث این مهم می‌شود. عملگر انقلاب در الگوریتم رقابت استعماری یک عملگر یگانی است که ویژگی‌های یک کشور فرزند تولیدشده را از عمل جذب تغییر می‌دهد. این تغییر باید بسیار کوچک و بر اساس احتمال pm باشد. ازآنجاکه تغییرات بسیار اندک است، فرزند جدید تفاوت زیادی با فرزند اصلی ندارد. نام عملگر انقلاب استفاده‌شده در این الگوریتم، معکوس‌کنندهٔ وظیفه است.

کشورهای والد، پیش از عملیات جذب

C_i	$Job_kTask^i_{qw}$	$Job_kTask^i_{bv}$	$Job_kTask^i_{mn}$	$Job_kTask^i_{qa}$	$Job_kTask^i_{sa}$	$Job_kTask^i_{za}$
C_{i+1}	$Job_kTask^j_{nb}$	$Job_kTask^j_{cv}$	$Job_kTask^j_{sx}$	$Job_kTask^j_{az}$	$Job_kTask^j_{hj}$	$Job_kTask^j_{sk}$
C_{i+2}	$Job_wTask^m_{cx}$	$Job_wTask^m_{bv}$	$Job_wTask^m_{mn}$	$Job_wTask^m_{qa}$	$Job_wTask^m_{sa}$	$Job_wTask^m_{za}$
C_{i+3}	$Job_wTask^n_{bh}$	$Job_wTask^n_{gi}$	$Job_wTask^n_{lr}$	$Job_wTask^n_{tn}$	$Job_wTask^n_{bt}$	$Job_wTask^n_{pb}$

First Point — Second point

کشورهای فرزند پس از عمل جذب

$newC_i$	$Job_kTask^i_{qw}$	$Job_kTask^i_{bv}$	$Job_wTask^m_{mn}$	$Job_wTask^m_{qa}$	$Job_wTask^m_{sa}$	$Job_kTask^i_{za}$
$newC_{i+1}$	$Job_kTask^j_{nb}$	$Job_kTask^j_{cv}$	$Job_wTask^n_{lr}$	$Job_wTask^n_{tn}$	$Job_wTask^n_{bt}$	$Job_kTask^j_{sk}$
$newC_{i+2}$	$Job_wTask^m_{cx}$	$Job_wTask^m_{bv}$	$Job_kTask^i_{mn}$	$Job_kTask^i_{qa}$	$Job_kTask^i_{sa}$	$Job_wTask^m_{za}$
$newC_{i+3}$	$Job_wTask^n_{bh}$	$Job_wTask^n_{gi}$	$Job_kTask^j_{sx}$	$Job_kTask^j_{az}$	$Job_kTask^j_{hj}$	$Job_wTask^n_{pb}$

First Point — Second point

(شکل-۳): عملگر جذب دونقطه‌ای

(Figure-3): Two-point Assimilation Operator

کشور فرزند قبل از عملگر انقلاب

$newC_{i+1}$	$Job_kTask^j_{nb}$	$Job_kTask^j_{cv}$	$Job_wTask^n_{lr}$	$Job_wTask^n_{tn}$	$Job_wTask^n_{bt}$	$Job_kTask^j_{sk}$
$newC_{i+2}$	$Job_wTask^m_{cx}$	$Job_wTask^m_{bv}$	$Job_kTask^i_{mn}$	$Job_kTask^i_{qa}$	$Job_kTask^i_{sa}$	$Job_wTask^m_{za}$

کشور فرزند بعد از عملگر انقلاب

$newerC_{i+1}$	$Job_kTask^j_{nb}$	$Job_kTask^j_{qa}$	$Job_wTask^n_{lr}$	$Job_wTask^n_{tn}$	$Job_wTask^n_{bt}$	$Job_kTask^j_{sk}$
$newerC_{i+2}$	$Job_wTask^m_{cx}$	$Job_wTask^m_{bv}$	$Job_kTask^i_{mn}$	$Job_kTask^i_{cv}$	$Job_kTask^i_{sa}$	$Job_wTask^m_{za}$

(شکل-۴): عملگر انقلاب

(Figure-4): Revolution Operator

این عملگر دو وظیفهٔ متعلّق به یک گره مه را به شکل تصادفی انتخاب کرده، و آنها را با هم جابه‌جا می‌کند. این فرآیند برای سایر گره‌های مه نیز تکرار می‌شود. شکل (۴) یک مثال از عملگر انقلاب را نشان می‌دهد.

۴-۴- الگوریتم ICTF

زمان‌بندی وظایف در محیط محاسبات مه مبتنی بر الگوریتم رقابت استعماری بر اساس الگوریتم ICTF طبق شبه کد جدول (۱) انجام می‌شود. همان‌طورکه مشخّص است الگوریتم ICTF در هر مرحله، فهرستی از وظایف

[Downloaded from jsdp.rcisp.ac.ir on 2026-08-11]

[DOI: 10.61186/jsdp.20.1.25]

جهت نگاشت به منابع محیط مه را انتخاب و اجرا می‌کند. خروجی این الگوریتم بهترین کشور، یا در واقع، نحوهٔ تخصیص منابع به وظایف مذکور است.

(جدول-۱): شبه‌رمز الگوریتم ICTF

(Table-1): ICTF Pseudocode Algorithm

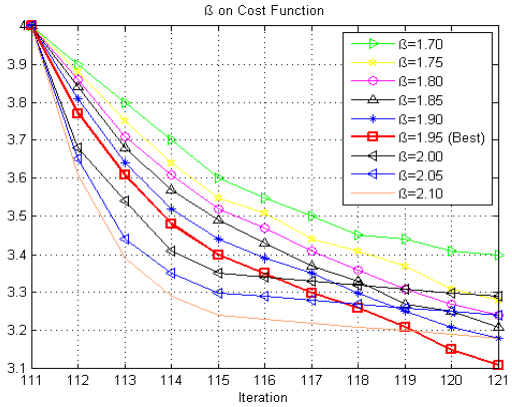
```
Get the Task(i);
Add the Task(i) to the Task list (TL);
until there are unscheduled tasks in TL
{
Select Sub Task List STL(i) ∈ Task list(TL);
 Create a Country Ci Model by Eq. 10
Initial the population P by Eq. 11 and STL(i);
while Stop Condition () do
  {
For each country Ci in P
      {
Compute the E by Eq. 13;
      Compute the Ei(t) by Eq. 14;
      Compute the ui(t) by Eq. 15
      Compute the T by Eq. 16
      Compute the C by Eq. 17;
      Compute the Fitness for all countries by Eq. 18;
      }
    Imperialist= Min ( Fitness of countries);
    Assimilation (According Eq. 19)
    Revolution()
    IF ( there the cost of country less than imperialist)
    Exchange the position of country and imperialist
     If ( convergence )
      Dispatch all the mapped tasks and execution;
      Calculate the Average (Makespan, Energy
Consumption, Resource Utilization, Remaining Energy)
  }
  Update the Task list(TL);
 }
 Void  Revolution (population P)
        {
        Selected population=Select 1/2 P randomly;
        For( i= 1; i<= Selected population; i++)
        {
                Candidate_country_a =
select a random C in Selected_population;
                Candidate_country_b =
        select a random C in Selected_population;
Candidate_country_a.loc
= select a random attribute in Candidate_country_a;
Candidate_country_b.loc
= select a random attribute in Candidate_country_b;
                Swap
        (
        Candidate_country_a.loc, Candidate_country_b
        );
        }
        }
Void Assimilation (population P)
{
        For each Ci country in P
        {
        Ci_loc = select two random points in Ci;
        Imperialist_loc
        = select two random points in Imperialist ;
        Swap (Ci_loc, Imperialist_loc);
        }
}
```

۵- ارزیابی راه‌کار پیشنهادی

در این بخش به ارزیابی روش ICTF در مقایسه با روش‌های دیگر می‌پردازیم. روش‌های انتخابی برای مقایسه با ICTF با توجه به ماهیّت فراابتکاری بودن الگوریتم پیشنهادی، شامل الگوریتم IWO-CA، DEWTS، EES، HEFT-B و HEFT-T هستند.

برای ارزیابی ICTF و مقایسهٔ آن با روش‌های دیگر از شبیه‌ساز iFogsim به‌عنوان یک کتابخانهٔ توسعه‌یافته از شبیه‌ساز Cloudsim جهت پردازش مه استفاده شده است[۳۱-۳۴].

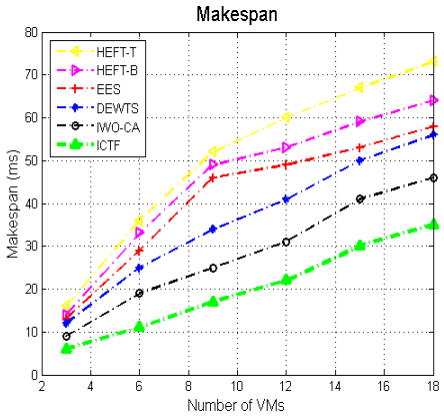
برای ارزیابی ICTF فرض کرده‌ایم تعداد ماشین‌های مجازی محیط محاسبات مه برابر با ۳، ۶، ۹، ۱۲، ۱۵، و ۱۸ و تعداد وظایف به‌ترتیب برابر با ۲۰، ۴۰، ۸۰، ۱۶۰ و ۴۰۰ باشد. یکی از شاخص‌های مهم در ارزیابی ICTF شاخص β (ضریب جذب کشورهای مستعمره به استعمارگر) است که برای تعیین مقدار مناسب آن، آزمون و خطا انجام شده است. شکل (۵) نشان‌دهندهٔ تأثیر شاخص β بر تابع هزینه است. طبق این شکل مقدار مناسب برای این شاخص برابر با ۱/۹۵ است.



(شکل-۵): تعیین مقدار مناسب شاخص β

(Figure-5): β Parameter Value

شکل (۶) مقایسه الگوریتم ICTF با سایر روش‌ها را نشان می‌دهد. همان‌طورکه در شکل مشاهده می‌شود، روش ICTF در همهٔ حالت‌ها و آزمایش‌ها کمترین مقدار Makespan را داشته است. علّت این امر دقت بالای روش ICTF در انتخاب ماشین‌های مجازی جهت اجرای وظایف است.



(شکل-۶): میزان Makespan

(Figure-6): Makespan

میزان Makespan روش‌های موردارزیابی بر اساس تعداد ماشین‌های مجازی در جدول (۲) ارائه شده است.

[Downloaded from jsdp.rcisp.ac.ir on 2026-08-11]

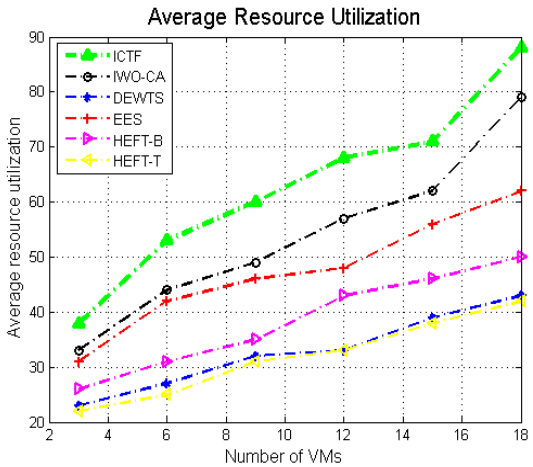
[DOI: 10.61186/jsdp.20.1.25]

(جدول-۲): نتایج Makespan
(Table-2): Makespan Results

HEFT-T	HEFT-B	EES	DEWTS	IWO-CA	ICTF	No. VMs
16	16	13	12	9	6	3
36	33	29	25	19	11	6
52	49	46	34	25	14	9
60	53	49	41	31	22	12
67	59	53	50	41	30	15
73	64	58	56	46	35	18
46.2	45.33	38	36.33	28.5	20.1	AVG
26.03	25.16	17.83	16.16	8.33	IMP	

همان‌طورکه در این جدول در ردیف مربوط به میانگین نتایج(AVG) مشاهده می‌شود، میانگین Makespan در روش پیشنهادی ICTF برابر ۲۰/۱ است که نسبت‌به سایر روش‌ها کارآیی بهتری دارد و همچنین، میزان بهبود روش پیشنهادی ICTF نسبت‌به سایر روش‌ها در ردیف مربوط به میزان بهبود (IPM) جدول نشان داده شده است.

شکل (۷) نشان‌دهندهٔ میزان بهره‌وری منابع در ICTF در مقایسه با روش‌های دیگر است. همان‌طورکه در شکل مشاهده می‌شود، ICTF بالاترین کارایی را در میان روش‌های موجود دارد. در روش پیشنهادی تابع هدف بر اساس معیارهای زمان، هزینه و انرژی مصرفی پیشنهاد شده است. به همین علت الگوریتم سعی در انتخاب ماشین‌هایی دارد که به بهترین شکل ممکن از منابع استفاده کند و بهره‌وری را بهبود دهد. از این رو ICTF نسبت‌به سایر روش‌ها عملکرد بهتری در بهره‌وری منابع داشته است.



(شکل-۷): میزان بهره‌وری منابع

(Figure-7): Resource Utilization

همچنین نتایج عددی بهره‌وری منابع روش‌های موردارزیابی در جدول (۳) ارائه شده است.

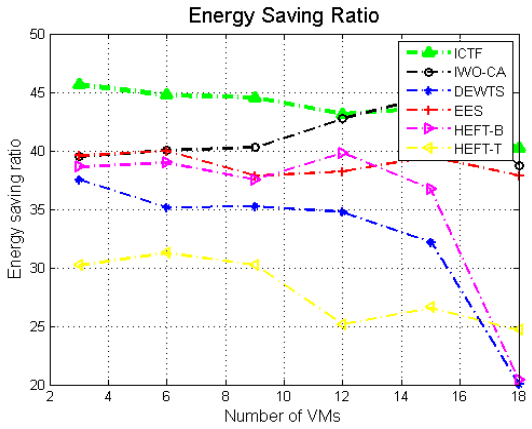
(جدول-۳): نتایج بهره‌وری منابع
(Table-3): Resource Utilization Results

HEFT-T	HEFT-B	EES	DEWTS	IWO-CA	ICTF	No. VMs
22	23	26	31	33	38	3
25	27	31	42	44	53	6
31	32	35	46	49	60	9
33	33	43	48	57	68	12
38	39	46	56	62	71	15
42	43	50	62	79	88	18
29.8	32.83	36.2	47.5	54	63	AVG
33.2	30.16	26.8	15.5	9	IMP	

همان‌طورکه در این جدول مشاهده می‌شود میانگین بهره‌وری منابع روش پیشنهادی ICTF برابر ۶۳ است که نسبت‌به سایر روش‌ها کارآیی بهتری دارد و میزان بهبود آن نسبت به سایر روش‌ها در ردیف آخر جدول نشان داده شده است.

شکل (۹) نشان‌دهندهٔ انرژی باقی‌مانده ICTF نسبت‌به سایر روش‌هاست. همان‌طورکه در شکل مشاهده می‌شود، روش ICTF در آزمایش‌ها با سناریوهای مختلف انرژی کمتری مصرف کرده است.

همچنین، میزان انرژی باقی‌مانده در جدول (۵) ارائه شده است. همان‌طورکه در این جدول مشاهده می‌شود، میانگین انرژی باقی‌مانده در روش ICTF برابر ۴۸/۶ است که نسبت‌به سایر روش‌ها کارآیی بهتری دارد و میزان بهبود آن در ردیف آخر جدول نشان داده شده است.



(شکل-۹): انرژی باقی‌مانده
(Figure-9): Remaining Energy

[Downloaded from jsdp.rcisp.ac.ir on 2026-08-11]
[DOI: 10.61186/jsdp.20.1.25]

(جدول-۵): نتایج انرژی باقی‌مانده

(Table-5): Remaining Energy Results

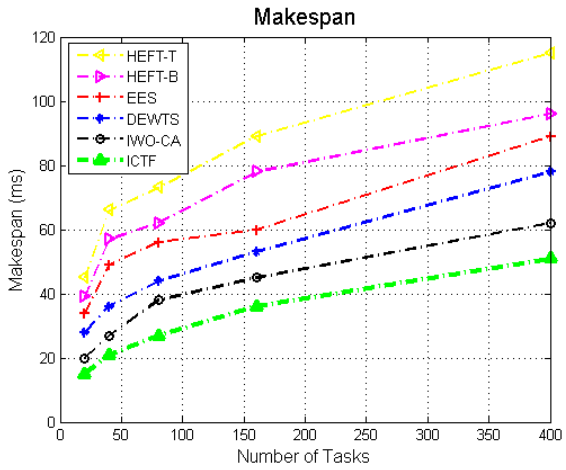
HEFT-T	HEFT-B	EES	DEWTS	IWO-CA	ICTF	No. VMs
30.2	37.5	38.6	39.6	39.5	45.6	3
31.2	35.1	38.9	40	40	44.7	6
30.2	35.2	37.5	37.8	40.3	74.5	9
25.1	34.7	39.8	38.2	42.7	43.1	12
26.5	32.1	36.7	39.5	44.6	43.6	15
24.6	20	20.3	37.8	38.7	40.2	18
28.6	32.4	38.3	38.8	40.9	48.6	AVG
19.9	16.1	10.3	9.8	7.6	IMP	

در ادامهٔ این بخش رفتار روش پیشنهادی در تغییر تعداد وظایف بررسی می‌شود. شکل (۱۰) مقایسهٔ الگوریتم ICTF با سایر روش‌ها را از منظر تغییرات تعداد وظایف نشان می‌دهد. همان‌طورکه در شکل مشاهده می‌شود، روش ICTF در همه حالت‌ها و آزمایش‌ها کمترین مقدار Makespan را داشته است.

(جدول-۶): نتایج Makespan بر اساس تعداد وظایف

(Table-6): Makespan Results based on Number of Tasks

HEFT-T	HEFT-B	EES	DEWTS	IWO-CA	ICTF	No. Tasks
45	39	34	28	20	15	
66	57	49	36	27	21	
73	62	56	44	38	27	
89	78	60	53	45	36	
115	96	89	78	62	51	
77.6	66.4	57.6	47.8	38.4	30	AVG
47.6	36.4	27.6	17.8	8.4	IMP	



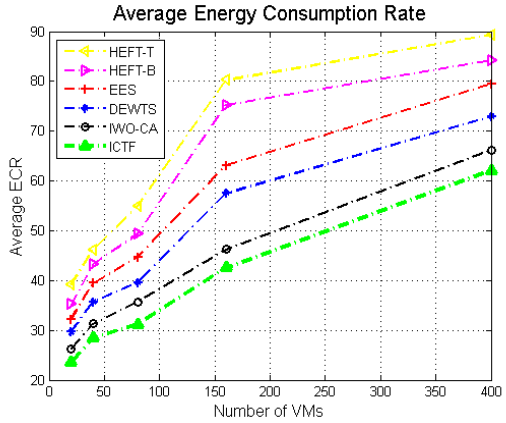
(شکل-۱۰): میزان Makespan بر اساس تعداد وظایف

(Figure-10): Makespan based on Number of Tasks

میزان Makespan روش‌های موردارزیابی بر اساس تعداد وظایف در جدول (۶) ارائه شده است. همان‌طورکه مشاهده می‌شود، روش پیشنهادی ICTF در مقایسه با سایر روش‌ها با افزایش تعداد وظایف میزان Makespan پایین‌تری دارد.

همانطورکه در این جدول در ردیف مربوط به میانگین نتایج(AVG) مشاهده می‌شود، میانگین Makespan به ازای تغییرات وظایف در روش پیشنهادی ICTF برابر ۳۰ است که نسبت‌به سایر روش‌ها کارآیی بهتری دارد و همچنین میزان بهبود روش پیشنهادی ICTF نسبت‌به سایر روش‌ها در ردیف مربوط به میزان بهبود(IPM) جدول نشان داده شده است.

شکل(۱۱) مقایسه میزان انرژی مصرفی الگوریتم ICTF نسبت‌به سایر روش‌ها از منظر تغییرات تعداد وظایف را نشان می‌دهد. همانطورکه در شکل مشاهده می‌شود روش ICTF در همه حالت‌ها و آزمایش‌ها کمترین مقدار مصرف انرژی را داشته است.



(شکل-۱۱): میزان مصرف انرژی بر اساس تعداد وظایف

(Figure-11): Energy Consumption based on Number of Tasks

میزان مصرف انرژی روش‌های موردارزیابی بر اساس تعداد وظایف در جدول (۷) ارائه شده است.

(جدول-۷): نتایج مصرف انرژی بر اساس تعداد وظایف

(Table-7): Energy Consumption Results

HEFT-T	HEFT-B	EES	DEWTS	IWO-CA	ICTF	No. Tasks
39.2	35.2	32.2	29.6	26.3	23.6	
46.1	43.2	39.4	35.7	31.2	28.4	
54.9	49.3	44.7	39.6	35.7	31.1	
80.2	75.2	63.1	57.5	46.2	42.5	
98.3	84.2	79.5	72.9	66.1	62.1	

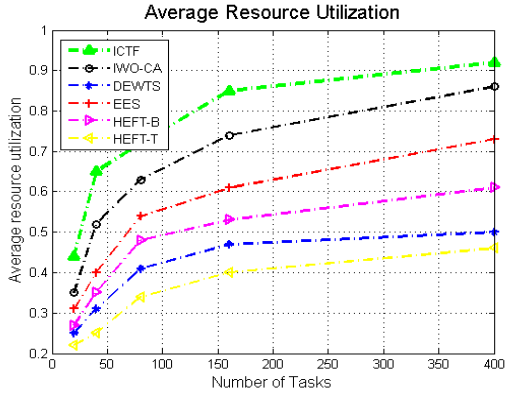
[Downloaded from jsdp.rcisp.ac.ir on 2026-08-11]

[DOI: 10.61186/jsdp.20.1.25]

61.9	57.4	51.8	47.1	41.1	37.5	AVG
24.3	19.8	14.2	9.5	3.5	IMP	

همان‌طورکه در این جدول در ردیف مربوط به میانگین نتایج(AVG) مشاهده می‌شود میانگین مصرف انرژی به ازای تغییرات وظایف در روش پیشنهادی ICTF برابر 37.5 می‌باشد که نسبت‌به سایر روش‌ها کارآیی بهتری دارد و همچنین میزان بهبود روش پیشنهادی ICTF نسبت‌به سایر روش‌ها در ردیف مربوط به میزان بهبود(IPM) جدول نشان داده شده است.

شکل(۱۲) مقایسه میزان بهره‌وری الگوریتم ICTF نسبت‌به سایر روش‌ها از منظر تغییرات تعداد وظایف را نشان می‌دهد. همانطورکه در شکل مشاهده می‌شود روش ICTF در همه حالت‌ها و آزمایش‌ها بیشترین میزان بهره‌وری از منابع را داشته است.



(شکل-۱۲): میزان بهره‌وری از منابع بر اساس تعداد وظایف

(Figure-12): Resource Utilization based on Number of Tasks

میزان مصرف انرژی روش‌های موردارزیابی بر اساس تعداد وظایف در جدول (۸) ارائه شده است.

(جدول-۸): نتایج بهره‌وری از منابع بر اساس تعداد وظایف

(Table-8): Resource Utilization Results

HEFT-T	HEFT-B	EES	DEWTS	IWO-CA	ICTF	No. Tasks
0.22	0.25	0.27	0.31	0.35	0.44	
0.25	0.31	0.35	0.40	0.52	0.63	
0.34	0.41	0.48	0.54	0.63	0.72	
0.40	0.47	0.53	0.61	0.74	0.85	
0.46	0.50	0.61	0.73	0.86	0.92	
0.33	0.38	0.44	0.51	0.62	0.71	AVG
0.37	0.32	0.26	0.19	0.09	IMP	

همان‌طورکه در این جدول در ردیف مربوط به میانگین نتایج (AVG) مشاهده می‌شود، میانگین بهره‌وری از منابع به‌ازای تغییرات وظایف در روش پیشنهادی ICTF برابر 0.71 است که نسبت‌به سایر روش‌ها کارآیی بهتری دارد و همچنین، میزان بهبود روش پیشنهادی ICTF نسبت‌به سایر روش‌ها در ردیف مربوط به میزان بهبود (IPM) جدول نشان داده شده است.

۶- نتیجه‌گیری

محیط محاسبات مه، به‌عنوان یک بستر توزیعی، الگویی برای دسترسی و استفاده از منابع اشتراکی در لبهٔ شبکه با هدف افزایش کارآیی و کاهش تأخیر انجام کارها است. تخصیص و زمان‌بندی منابع که به‌طور معمول در لبهٔ شبکه اختصاص داده می‌شود، یک چالش اساسی در محیط محاسبات مه است. یکی از اهداف زمان‌بندی استفاده از منابع، به‌طور کارآمد و به دست آوردن بیشینهٔ بهره‌وری است. ازآنجاکه مسألهٔ زمان‌بندی یک مسأله بهینه‌سازی است، در این مقاله از الگوریتم فراابتکاری رقابت استعماری استفاده کردیم. در الگوریتم پیشنهادی ICTF جمعیّتی تصادفی از حالت‌های مختلف نگاشت وظایف بر ماشین‌های محیط مه ایجاد می‌کند. تابع برازندگی که جمعیّت کشورها را ارزیابی می‌کند، از ترکیب سه معیار مصرف انرژی، هزینه و زمان اجرای وظایف تشکیل شد. نتایج آزمایش‌ها نشان‌دهندهٔ کارایی بالای روش ICTF در معیارهای Makespan، بهره‌وری منابع، میزان مصرف انرژی و انرژی باقی‌مانده نسبت‌به سایر روش‌های مشابه است.

۷- مراجع 7-Refrence

[1] D. Tychalas and H. Karatza, “A scheduling algorithm for a fog computing system with bag-of-tasks jobs: Simulation and performance evaluation,” Simul. Model. Pract. Theory, vol. 98, no. 101982, p. 101982, 2020.

[2] M. Yang, H. Ma, S. Wei, Y. Zeng, Y. Chen, and Y. Hu, “A multi-objective task scheduling method for fog computing in cyber-physical-social services,” IEEE Access, vol. 8, pp. 65085–65095, 2020.

[3] S. Wang, T. Zhao, and S. Pang, “Task scheduling algorithm based on improved firework algorithm in fog computing,” IEEE Access, vol. 8, pp. 32385–32394, 2020.

[4] M. Ghobaei-Arani, A. Souri, F. Safara, and M. Norouzi, “An efficient task scheduling approach using moth-flame optimization algorithm for cyber-physical system

[Downloaded from jsdp.rcisp.ac.ir on 2026-08-11]

[DOI: 10.61186/jsdp.20.1.25]

[15] H. Momeni, A. Yavari, F. Goli, and M. A. Chakoli, “Optimality Evaluation of Maintenance Strategy Using LVQ Neural Network”.

[16] A. Yavari, M. Golbaghi, and H. Momeni, “Assessment of effective risk in software projects based on Wallace’s classification using fuzzy logic,” Int. j. inf. eng. electron. bus., vol. 5, no. 4, pp. 58–64, 2013.

[17] A. Yavari, M. Musavi, H. Momeni, and M. Hamzehnia, “Measuring the Failure Rate in Service Oriented Architecture Using Fuzzy Logic,” Journal of mathematics and computer Science, vol. 7, no. 3, pp. 160–170, 2013.

[18] R. Mahmud, S. N. Srirama, K. Ramamohanarao, and R. Buyya, “Profit-aware application placement for integrated Fog–Cloud computing environments,” J. Parallel Distrib. Comput., vol. 135, pp. 177–190, 2020.

[19] L. Liu, D. Qi, N. Zhou, and Y. Wu, “A Task Scheduling algorithm based on classification mining in Fog Computing environment,” Wirel. Commun. Mob. Comput., vol. 2018, pp. 1–11, 2018.

[20] S. Bitam, S. Zeadally, and A. Mellouk, “Fog computing job scheduling optimization based on bees swarm,” Enterp. Inf. Syst., vol. 12, no. 4, pp. 373–397, 2018.

[21] R. Beraldi, C. Canali, R. Lancellotti, and G. P. Mattia, “A random walk based load balancing algorithm for fog computing,” in 2020 Fifth International Conference on Fog and Mobile Edge Computing (FMEC), 2020.

[22] H. Rafique, M. A. Shah, S. U. Islam, T. Maqsood, S. Khan, and C. Maple, “A novel bio-inspired hybrid algorithm (NBIHA) for efficient resource management in fog computing,” IEEE Access, vol. 7, pp. 115760–115773, 2019.

[23] Y. Li, W. Ma, J. Zhang, J. Wu, J. Ma, and X. Dang, “Efficient fog node resource allocation algorithm based on taboo genetic algorithm,” in Advances in Intelligent Systems and Computing, Singapore: Springer Singapore, 2021, pp. 1565–1573.

[24] S. Javanmardi, M. Shojafar, V. Persico, and A. Pescapè, “FPFTS: A joint fuzzy particle swarm optimization mobility-aware approach to fog task scheduling algorithm for Internet of Things devices,” Softw. Pract. Exp., no. spe.2867, 2020.

[25] Z. Tang, L. Qi, Z. Cheng, K. Li, S. U. Khan, and K. Li, “An energy-efficient task scheduling applications in fog computing,” Trans. emerg. telecommun. technol., vol. 31, no. 2, 2020.

[5] F. Murtaza, A. Akhunzada, S. ul Islam, J. Boudjadar, and R. Buyya, “QoS-aware service provisioning in fog computing,” J. Netw. Comput. Appl., vol. 165, no. 102674, p. 102674, 2020.

[6] J. C. Guevara, R. da S. Torres, and N. L. S. da Fonseca, “On the classification of fog computing applications: A machine learning perspective,” J. Netw. Comput. Appl., vol. 159, no. 102596, p. 102596, 2020.

[7] A. Bose, T. Biswas, and P. Kuila, “A novel genetic algorithm-based scheduling for multi-core systems,” in Smart Innovations in Communication and Computational Sciences, Singapore: Springer Singapore, 2019, pp. 45–54.

[8] B. Jamil, M. Shojafar, I. Ahmed, A. Ullah, K. Munir, and H. Ijaz, “A job scheduling algorithm for delay and performance optimization in fog computing,” Concurr. Comput., vol. 32, no. 7, 2020.

[9] M. Etemadi, M. Ghobaei-Arani, and A. Shahidinejad, “Resource provisioning for IoT services in the fog computing environment: An autonomic approach,” Comput. Commun., vol. 161, pp. 109–131, 2020.

[10] M. S. Aslanpour, S. S. Gill, and A. N. Toosi, “Performance evaluation metrics for cloud, fog and edge computing: A review, taxonomy, benchmarks and standards for future research,” Internet of Things, vol. 12, no. 100273, p. 100273, 2020.

[11] P. Kanani and M. Padole, “Exploring and optimizing the fog computing in different dimensions,” Procedia Comput. Sci., vol. 171, pp. 2694–2703, 2020.

[12] M. H. Shahid, A. R. Hameed, S. ul Islam, H. A. Khattak, I. U. Din, and J. J. P. C. Rodrigues, “Energy and delay efficient fog computing using caching mechanism,” Comput. Commun., vol. 154, pp. 534–541, 2020.

[13] R. O. Aburukba, M. AliKarrar, T. Landolsi, and K. El-Fakih, “Scheduling Internet of Things requests to minimize latency in hybrid Fog–Cloud computing,” Future Gener. Comput. Syst., vol. 111, pp. 539–551, 2020.

[14] H. Momeni and A. Yavari, “Complexity evaluation of aspect-oriented software with adaptive neuro-fuzzy inference system,” Int J Basic Sci Appl Res, vol. 3, pp. 22–30, 2014.

[Downloaded from jsdp.rcisp.ac.ir on 2026-08-11]

[DOI: 10.61186/jsdp.20.1.25]

algorithm in DVFS-enabled cloud environment," J. Grid Comput., vol. 14, no. 1, pp. 55–74, 2016.

[26] P. Hosseinioun, M. Kheirabadi, S. R. Kamel Tabbakh, and R. Ghaemi, "A new energy-aware tasks scheduling approach in fog computing using hybrid meta-heuristic algorithm," J. Parallel Distrib. Comput., vol. 143, pp. 88–96, 2020.

[27] Q. Huang, S. Su, J. Li, P. Xu, K. Shuang, and X. Huang, "Enhanced energy-efficient scheduling for parallel applications in cloud," in 2012 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (ccgrid 2012), 2012.

[28] S. A. A. Naqvi, N. Javaid, H. Butt, M. B. Kamal, A. Hamza, and M. Kashif, "Metaheuristic optimization technique for load balancing in cloud-fog environment integrated with smart grid," in Advances in Network-Based Information Systems, Cham: Springer International Publishing, 2019, pp. 700–711.

[29] S. P. Singh, A. Sharma, and R. Kumar, "Design and exploration of load balancers for fog computing using fuzzy logic," Simul. Model. Pract. Theory, vol. 101, no. 102017, p. 102017, 2020.

[30] A. Chagari, M.R. Feizi Derakhshi, "Automatic Clustering using Improved Imperialist Competitive Algorithm" Journal of Signal and Data Processing" Vol. 14. No. 2, pp. 159-169, 2017.

آرش چاقری و محمدرضا فیضی درخشی، خوشه‌بندی خودکار داده‌ها با بهره‌گیری از الگوریتم رقابت استعماری، مجلهٔ « پردازش علایم و داده‌ها»، دورهٔ ۱۴ شماره ۲، صفحه ۱۶۹-۱۵۹، ۱۳۹۷

[31] H. Gupta, A. Vahid Dastjerdi, S. K. Ghosh, and R. Buyya, "iFogSim: A toolkit for modeling and simulation of resource management techniques in the Internet of Things, Edge and Fog computing environments: IFogSim: A toolkit for modeling and simulation of internet of things," Softw. Pract. Exp., vol. 47, no. 9, pp. 1275–1296, 2017.

[32] R. Mahmud and R. Buyya, "Modelling and simulation of Fog and edge computing environments using iFogSim toolkit," arXiv [cs.DC], 2018.

[33] D. Seo et al., "Dynamic iFogSim: A framework for full-stack simulation of dynamic resource management in IoT systems," in 2020 International Conference on Omni-layer Intelligent Systems (COINS), 2020.

[34] M. I. Bala and M. A. Chishti, "Offloading in cloud and fog hybrid infrastructure using iFogSim," in 2020 10th International Conference on Cloud Computing, Data Science & Engineering (Confluence), 2020.

حسین مؤمنی دارای مدرک کارشناسی رشته مهندسی کامپیوتر، گرایش نرم‌افزار از دانشگاه فردوسی مشهد و مدرک کارشناسی ارشد و دکترای رشته مهندسی کامپیوتر گرایش نرم‌افزار از دانشگاه علم و صنعت ایران است. زمینه‌های پژوهشی ایشان سامانه‌های تحمل‌پذیر خطا و مدیریّت بهینهٔ منابع در سامانه‌های توزیعی است. ایشان در حال حاضر استادیار دانشگاه گلستان است. نشانی رایانامهٔ ایشان عبارت است از:

h.momeni@gu.ac.ir

علی یاوری دارای مدرک کارشناسی مهندسی نرم‌افزار و کارشناسی ارشد مهندسی فناوری اطّلاعات از دانشگاه علوم و فنون مازندران است. ایشان در حال حاضر دانشجوی دورهٔ دکترا در رشته مهندسی کامپیوتر گرایش نرم‌افزار و الگوریتم در دانشگاه اراک است. زمینه‌های پژوهشی ایشان محاسبات توزیعی، الگوریتم‌های بهینه‌سازی و داده‌کاویاست. ایشان در حال حاضر مدرس دانشگاه فنی و حرفه‌ای نیز هست. نشانی رایانامهٔ ایشان عبارت است از:

aliyavari68@gmail.com
yavari@ustmb.ac.ir

[Downloaded from jsdp.rcisp.ac.ir on 2026-08-11]

[DOI: 10.61186/jsdp.20.1.25]