دو فصلنامه علمی
فناوری اطلاعات و ارتباطات ایران

سال چهاردهم، شماره‌های ۵۳ و ۵۴، پاییز و زمستان
۱۴۰۱
صفحات: ۲۰۰تا ۲۲۳

Review: Application of Meta-Heuristic Algorithms in Cloud Load Balancing

Mehdi Morsali [*]، ***Abolfazl Taroghi Haqiqat*** [*] ، ***Sasan Hossein Alizadeh*** [**]

[*] Faculty of Computer Engineering and Information Technology, Qazvin Branch, Islamic Azad University, Qazvin, Iran
[**] Information Technology Research Institute, Communication and Information Technology Research Institute, Tehran, Iran

Abstract:

By widespread use of cloud computing, the need to improve performance and reduce latency in the cloud increases. One of the problems of distributed environments, especially clouds, is unbalanced load which results in reducing speed and efficiency and increasing delay in data storage and retrieval time. Various methods for load balancing in the cloud environment have been proposed, each of which has addressed the issue from its own perspective and has its advantages and disadvantages. In this research, we first provide some criteria for measuring load balance in the cloud and then examine the use of Metaheuristic methods in load balancing in the cloud environment. After introducing Metaheuristic load balancing methods, we have compared them based on the aforementioned criteria and discussed the advantages and disadvantages of each.

Ant Colony Algorithms, Artificial Ant Colony, Bee Colony, Artificial Bee Colony, Bee Foraging Algorithm, Particle Swarm, Cat Swarm, Simulated Annealing, Genetic Algorithm, Tabu Search, Fish Swarm and Hybrid Algorithms and etc. examined in this research.

Keywords: Cloud Computing, Load Balancing, Meta-heuristic Methods, Overloading, Underloading

مروری بر کاربرد الگوریتم‌های فراابتکاری در توازن بار در رایانش ابری

مهدی مرسلی* ، ابوالفضل طرقی حقیقت* ، ساسان حسینعلی زاده**

* دانشکده مهندسی کامپیوتر و فناوری اطلاعات، دکترا ،واحد قزوین، دانشگاه آزاد اسلامی، قزوین، ایران

** پژوهشکده فناوری اطلاعات، دکترا ،پژوهشگاه ارتباطات و فناوری اطلاعات، تهران، ایران

تاریخ دریافت:۱۴۰۰/۰۷/۲۶ تاریخ پذیرش: ۱۴۰۱/۰۳/۰۵

نوع مقاله: مروری

چکیده

با گسترش استفاده از رایانش ابری نیاز به بهبود کارایی و کاهش تاخیر در ابر افزایش می‌یابد. یکی از مسائل محیط‌های توزیع شده و مخصوصا ابر، عدم توازن بار و در نتیجه کاهش سرعت و کارایی و افزایش تاخیر در زمان ذخیره و بازیابی اطلاعات می‌باشد. روش‌های مختلفی برای متوازن سازی بار در محیط ابر ارائه شده‌اند که هر کدام از منظری به موضوع پرداخته‌اند و مزایا و معایب خود را دارند. ما در این کار نخست معیارهایی برای سنجش توازن بار در ابر ارائه کرده‌ایم و سپس به بررسی کاربرد روش‌های فراابتکاری در متوازن سازی بار در محیط ابر پرداخته‌ایم. پس از معرفی روش‌های توازن بار فراابتکاری مختلف، آنها را براساس معیارهای مذکور باهم مقایسه کرده و به مزایا و معایب هر کدام پرداخته‌ایم.

الگوریتم‌های کلونی مورچه، کلونی مورچه مصنوعی، کلونی زنبور، کلونی زنبور مصنوعی، جست‌وجوی غذای زنبور عسل، ازدحام ذرات، ازدحام گربه‌ها، تبرید شبیه‌سازی شده، الگوریتم ژنتیک، جست‌وجوی ممنوعه، الگوریتم دسته ماهی‌ها و الگوریتم‌های ترکیبی در این کار بررسی شده‌اند.

واژگان کلیدی: رایانش ابری، توازن بار، روش‌های فراابتکاری، بیش‌باری ، کم‌باری

۱. مقدمه

با رشد سریع فناوری اطلاعات، رایانش ابری به عنوان جایگزین فناوری‌های رایانش سنتی ظهور کرد تا کاربران بتوانند با پرداخت هزینه، در هر مکان و هر زمانی از خدمات رایانشی استفاده کنند. این فناوری به کاربران امکان دسترسی به مجموعه‌ای از منابع رایانش (سرویس‌دهنده‌ها، حافظه‌ها، شبکه‌ها و برنامه‌ها) را می‌دهد. شرکت‌های زیادی مانند سرویس وب آمازون[۱] (AWS)، مایکروسافت آژور[۲]، گوگل، ابر آی.بی.ام[۳] و آمازون جزء ارائه دهندگان بزرگ خدمات ابر به شمار می‌آیند [1]. هدف اولیه ابر استفاده موثر از منابع توزیع شده با هدف رسیدن به بازدهی و کارایی بالا می‌باشد. این امر به ابر توانایی حل مسائلی را می‌دهد که نیاز به توان رایانشی بالایی دارند. همچنین امکان توزیع منابع در

نویسنده مسئول: مهدی مرسلی mehdi_morsali@qiau.ac.ir

[۱] Amazon Web Service

[۲] Microsoft Azure

[۳] IBM Cloud

تمام جهان و اجرای وظایف بر روی مراکز داده مختلف را فراهم می‌آورد[2].

رایانش ابری را می‌توان به دو صورت دسته‌بندی کرد: براساس مکان یا براساس خدمات ارائه شده. براساس مکان، یک ابر می‌تواند عمومی، خصوصی، ترکیبی یا انجمنی باشد. خدمات ابر عمومی با اجازه شرکت ارائه دهنده خدمات، در هر زمان و هر مکان و بر روی هر زیرساختی و برای هر کسی در دسترس هستند. ابرهای عمومی در برابر حمله‌های مختلف بسیار آسیب‌پذیر هستند، اما از نظر هزینه به صرفه‌اند. ابرهای خصوصی به طور انحصاری برای کاربران یا سازمان‌های خاصی در دسترس هستند. بالاترین سطح امنیت و کنترل دسترسی را دارند، البته هزینه‌هایشان نیز بالاتر است. ابرهای ترکیبی همانطور که از نامشان پیداست ترکیبی از ابرهای عمومی و خصوصی هستند و براساس نیازهای سازمان برای اهداف مختلفی مورد استفاده قرار می‌گیرند. ابرهای انجمنی از یک زیرساخت عمومی تشکیل شده‌اند که به وسیله سازمان‌هایی مورد استفاده قرار می‌گیرند که مدیریت و داده مشترک دارند. ابرها براساس خدماتی که ارائه می‌دهند، به دسته‌های زیرساخت به عنوان سرویس[۱] (IaaS)، پلتفرم به عنوان سرویس[۲] (PaaS) یا نرم‌افزار به عنوان سرویس[۳] (SaaS) تقسیم می‌شوند. در IaaS ابر منابع اولیه فناوری اطلاعات مانند ویژگی‌های شبکه، کامپیوترها، انعطاف‌پذیریِ کنترل بر روی منابع رایانشی ارائه می‌کند. PaaS معمولا سازمان را از زیرساخت پایه بی‌نیاز می‌کند و اجازه می‌دهد تا سازمان روی گسترش برنامه‌ها تمرکز کند. SaaS به کاربران اجازه می‌دهد تا به جای فکر به زیرساخت و سرویس‌ها بر روی یک نرم‌افزار خاص تمرکز کند. رایانش ابری در کنار این سرویس‌ها، سرویس‌هایی مانند پایگاه داده به عنوان سرویس[۴] (DaaS)، سیستم خبره به عنوان سرویس[۵] (EaaS)، محل ذخیره‌سازی به عنوان سرویس[۶] (SaaS)، شبکه به عنوان سرویس[۷] (NaaS)، امنیت به عنوان سرویس[۸] (SECaaS) ارائه می‌کند [3].

ما در این کار پژوهشی، چالش‌های اصلی رایانش ابری را معرفی کرده و سپس به توازن بار در محیط ابر خواهیم پرداخت. هدف اصلی این کار پژوهشی پرداختن به کاربرد روش‌های جست‌وجوی فراابتکاری در بهینه‌سازی بار در رایانش ابری می‌باشد. برای رسیدن به این هدف نخست به برخی چالش‌های رایانش ابری خواهیم پرداخت، سپس مروری بر رویکردهای توازن بار و معیارهای سنجش توازن بار خواهیم داشت و در نهایت به مرور روش‌های توازن بار مبتنی بر جست‌وجوی فراابتکاری خواهیم پرداخت و مروری بر شکاف‌های موجود میان روش‌های ارائه شده و نیاز به روش‌های نوین ارائه خواهیم کرد.

۱-۱- انگیزه پژوهش

مقاله‌های مروری نسبتا زیادی در ارتباط با توازن بار و زمان‌بندی در ابر و کاربرد روش‌های مبتنی بر یادگیری ماشین و همچنین کاربرد الگوریتم‌های فراابتکاری در رایانش ابری منتشر شده‌اند.

برای نمونه آرونرانی و همکاران[۹] مروری بر استراتژی‌های زمان‌بندی ارائه کرده‌اند و به برخی مزایا و معایب پرداخته‌اند. تمرکز این کار بر روی الگوریتم‌های جست‌وجوی فراابتکاری نیست ولی با این وجود به دلیل تعدد کاربرد الگوریتم‌های فراابتکاری، بخش عمده‌ای از کار به الگوریتم‌های ژنتیک، ازدحام ذرات و کلونی مورچه‌ها اختصاص داده شده است [4]. در این کار معیارهای مقایسه مدونی برای مقایسه استراتژی‌های زمان‌بندی ارائه نشده است و فقط به مرور ادبیات پرداخته شده است. کومار و همکاران[۱۰] یک مرور سیستماتیک روی الگوریتم‌های زمان‌بندی در رایانش ابری انجام داده‌اند و معیارهایی برای مقایسه الگوریتم‌های زمان‌بندی معرفی کرده‌اند و مروری کوتاه بر ابزارهای شبیه‌سازی ابر ارائه نموده‌اند. [5]. نقطه قوت این کار ارائه فلوچارت‌های ساده و گویای الگوریتم‌های فراابتکاری پرکاربرد می‌باشد. یانگ[۱۱] و رحمانی مروری بر سازوکارهای زمان‌بندی در رایانش مه ارائه کرده‌اند. در این مقاله مروری روش‌های زمان‌بندی به دسته ابتکاری و فراابتکاری دسته‌بندی و معرفی شده‌اند، اما به جزئیات و چگونگی کارکرد روش‌ها پرداخته نشده است [6]. موهان سینگ و همکاران[۱۲]، در کاری ارزشمند، یک رده‌بندی نسبتا کامل از روش‌های جست‌وجوی فراابتکاری و کاربرد آنها در زمان‌بندی ابر و مه، ارائه کرده‌اند [7]. کونجانگ و لینا[۱۳] یک مرور سیستماتیک بر کاربرد رویکردهای فراابتکاری در زمان‌بندی ابرهای زیرساخت به عنوان سرویس ارائه کرده‌اند که به مزایای الگوریتم‌های فراابتکاری، چالش‌های این الگوریتم‌ها پرداخته‌اند. این کار از نظر بیان نظری مزایا و معایب و دسته‌بندی الگوریتم‌های فراابتکاری بسیار جامع

[۱] Infrastructure as a Service
[۲] Platform as a Service
[۳] Software as a Service
[۴] Database as a Service
[۵] Expert system as a Service
[۶] Storage as a Service
[۷] Network as a service
[۸] Security as a Service

[۹] Arunarani et al.
[۱۰] Kumar et al.
[۱۱] Yang.Z
[۱۲] Singh et al.
[۱۳] J. Kok Konjaang and Lina. Xu

است، اما به خود الگوریتم‌ها و نحوه عملکرد آنها اشاره‌ای نکرده است [8].

۱-۲- نوآوری‌ها

- مطالعه تحلیلی روش‌های توازن بار مبتنی بر الگوریتم‌های فراابتکاری در طول بازه تحقیق
- بررسی مزایا و معایب روش‌های فراابتکاری ارائه شده در منابع
- شناسایی معیارهای سنجش توازن بار و مقایسه روش‌های موجود بر اساس معیارهای شناسایی شده
- ارائه پیشنهادهایی برای مطالعه در زمینه کاربرد روش‌های فراابتکاری در توازن بار در رایانش ابری

۱-۳- روش انجام پژوهش

مسئله توازن بار با استفاده از متدلوژی چارچوب سازنده عمومی[۱] (CGF) مورد بررسی قرار گرفت. با مطالعه منابع مرتبط با موضوع روش‌های فراابتکاری در توازن بار در محیط ابر، از یک ساختار درختی برای رده‌بندی روش‌های فراابتکاری و کاربرد آنها در توازن بار در منابع استفاده شد. علاوه بر این مطالعه منابع مرتبط، براساس مرور سیستماتیک منابع و تمرکز بر روی کاربرد الگوریتم‌های فراابتکاری در توازن بار ابر انجام شد. حوزه کاری محدود به بهینه‌سازی توازن بار بوده و به موضوع‌های دیگری مانند بهینگی مصرف انرژی و چیزهایی از این دست پرداخته نشده است. منابع مطالعه شده محدود به بازه زمانی ۲۰۱۰ تا فوریه ۲۰۲۲ می‌باشد. برای اینکه سرعت پژوهش بیشتر باشد و نتایج مطالعه به روزتر باشند، تمرکز کار بیشتر بر روی منابع منتشر شده پس از سال ۲۰۱۶ می‌باشد.

برای یافتن منابع پایگاه داده‌های IEEE Xplore، ACM Digital Library، Springer Link، Web of Science، Google Scholar و منابع دیگر با کلیدواژه‌هایی همچون Cloud load balancing، Task Scheduling، Workflow Scheduling، Resource Allocation، Single and Multi-objective scheduling techniques به همراه واژگان Meta-heuristics، Meta-heuristic optimization مورد جست‌وجو قرار گرفتند و کلیدواژه‌های به دست آمده جدید دوباره جست‌وجو شدند. منابع به دست آمده براساس عنوان، چکیده مقاله‌ها و در انتها براساس مطالعه متن پالایش شدند. مراحل روش جست‌وجو در تصویر ۱ نمایش داده شده است.

[۱] Constructive Generic Framework

۱-۴- شناسایی پرسش‌های اصلی

پرسش ۱: چرا متوازن سازی بار در ابر اهمیت دارد؟

پرسش ۲: معیار سنجش توازن بار چیست؟ آیا توازن بار یک مسئله با تک هدف است یا چندین هدف دارد؟

پرسش ۳: روش‌های پرکاربرد برای متوازن سازی بار کدامند؟

پرسش ۴: استفاده از الگوریتم‌های فراابتکاری چه مزایا و معایبی نسبت به روش‌های معمول دارد؟

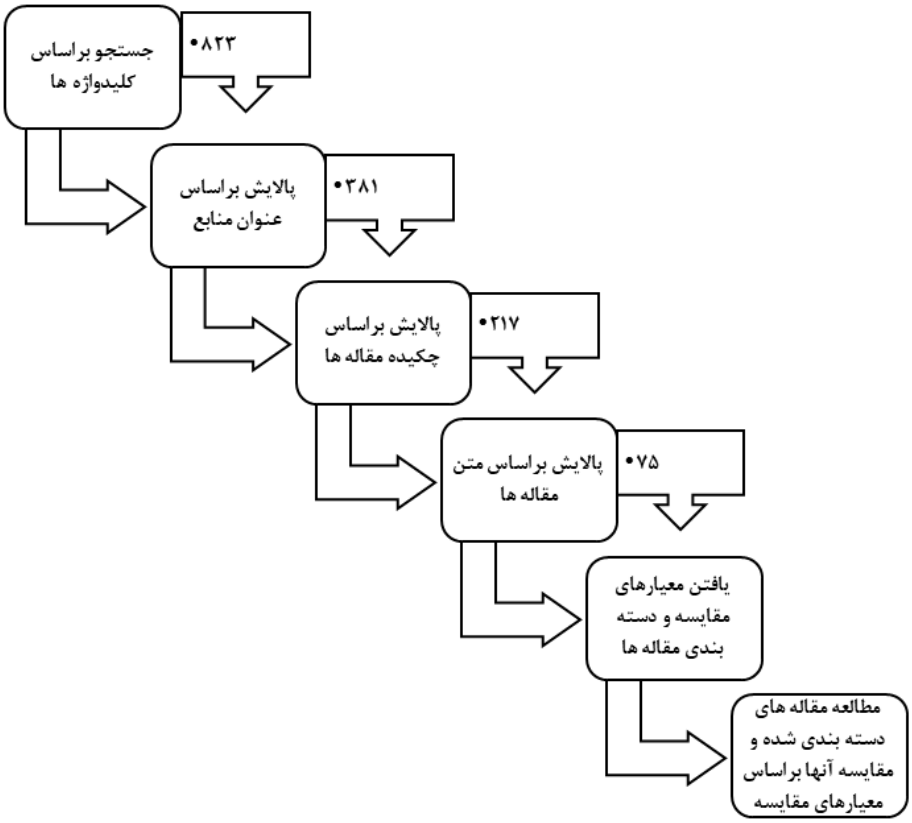
۲. روش‌های توازن بار در منابع مطالعه شده

برای پاسخ به پرسش سوم، در این بخش به مرور استراتژی‌های توازن بار در حالت کلی پرداخته‌ایم. البته تکنیک‌های زیادی در این گروه وجود دارد، مانند راند روبین[۲]، الگوریتم تصادفی‌سازی شده[۳]، الگوریتم حد آستانه[۴]، OLB، OLB+LBMM، Min-Min، Max-Min، الگوریتم اجرای جریان پخش برابر[۵]، توازن بار متوقف کننده[۶]، الگوریتم تپه نوردی تصادفی[۷]، پیوستن به صف خالی که مزایا و معایب آنها در جدول ۱ نمایش داده شده است.

۲,۱. Round Robin (RR)

یک الگوریتم توازن بار ثابت است که وضعیت بار پیشین گره را در تخصیص بار کنونی در نظر نمی‌گیرد و از زمان‌بندی RR برای تخصیص کارها استفاده می‌کند. این الگوریتم گره نخست را به طور تصادفی انتخاب می‌کند و سپس کارها را به ترتیب RR به گره‌ها تخصیص می‌دهد. این الگوریتم برای رایانش ابری مناسب نیست، زیرا ممکن است بار برخی گره‌ها بسیار سنگین شود و برخی بیکار بمانند. دلیل این امر آن است که زمان اجرای هیچکدام از فرایندها پیش از اجرا معلوم نیست [9].

[۲] Round Robin
[۳] Randomize Algorithm
[۴] Threshold Algorithm
[۵] Equally Spread Current Execution Algorithm
[۶] Throttled Load Balancing
[۷] Stochastic Hill Climbing



بنابراین الگوریتم RR وزن‌دار برای حل این مشکل ارائه شد. در این الگوریتم به هر گره یک وزن خاص، اختصاص داده می‌شود. هر گره براساس وزن اختصاص داده شده به آن تعداد مناسبی درخواست دریافت می‌کند. اگر وزن اختصاص داده شده به تمام گره‌ها یکسان باشد، آنگاه تمام گره‌ها ترافیک یکسان دریافت می‌کنند. در رایانش ابری تعیین دقیق زمان اجرا امکان‌پذیر نیست، بنابراین این الگوریتم توصیه نمی‌شود [10].

تصویر ۱- نمودار روش جستجو، شناسایی و دسته‌بندی منابع مربوط به کاربرد روش‌های فراابتکاری در رایانش ابری و تعداد منابع در هر گام از پژوهش

Min-Min.۲,۲

یک الگوریتم توازن بار ایستاست، بنابراین تمام اطلاعات مربوط به کار (job) از پیش مشخص است. برخی اصطلاحات مربوط به توازن بار در ادامه آمده است:

ETC[۱]: زمان مورد انتظار برای اجرای کار روی تمام گره‌ها در ماتریس ETC ذخیره می‌شود. اگر کار بر روی یک گره خاص قابل اجرا نباشد، مقدار مربوطه در ماتریس ETC بی‌نهایت در نظر گرفته می‌شود.

OLB[۲]: یک روش توازن بار فرصت طلبانه است که در آن هر کار با ترتیبی بدون تبعیض به ETC مربوطه روی گره‌ها تخصیص داده می‌شود. OLB زمان‌بندی توازن بار را فراهم می‌کند ولی دامنه پوشش ضعیفی دارد.

MET[۳]: الگوریتم با زمان اجرای کمینه. در این الگوریتم هر کار به گرهی که کمترین زمان اجرا را مطابق جدول ETC دارد تخصیص داده می‌شود، بدون اینکه به بار کنونی پردازنده توجه شود. MET تلاش می‌کند تا بهترین زوج کار-پردازنده را بیابد ولی چندان موفق نیست و موجب عدم توازن بار سرویس دهنده می‌شود.

MCT[۴]: الگوریتم با پایین‌ترین زمان کامل شدن. در این الگوریتم کارها براساس کوتاه‌ترین زمان برای کامل شدن به گره‌ها تخصیص داده می‌شوند. زمان کامل شدن با افزودن زمان مورد انتظار برای انجام آن کار با توجه به زمان آماده به کار گره محاسبه می‌شود. گره با کوتاه‌ترین زمان کامل شده برای انجام آن کار خاص انتخاب می‌شود. این الگوریتم در هر زمان تنها یک کار را در نظر می‌گیرد.

الگوریتم Min_Min با مجموعه‌ای از کارهای اختصاص داده نشده آغاز می‌شود. نخست کوتاه‌ترین زمان برای کامل شدن تمام کارها محاسبه می‌شود. کار با کوتاه‌ترین زمان انتخاب شده و سپس گرهی

[۱] Expected Time of Compute
[۲] Opportunistic Load Balancing
[۳] Minimum Execution Time
[۴] Minimum Complete Time

که کوتاه‌ترین زمان کامل شدن را برای تمام کارها دارد، انتخاب می‌شود. در پایان گره انتخاب شده و کار انتخاب شده تایید می‌شوند و زمان آماده به کار گره به‌هنگام‌سازی می‌شود. این فرایند تا هنگامی که تمام کارهای تخصیص داده نشده، تخصیص داده شوند، تکرار می‌شود. مزیت این الگوریتم این است که کار با کوتاه‌ترین زمان اجرا، انجام می‌شود. عیب این الگوریتم هم این است که برخی کارها ممکن است با گرسنگی مواجه شوند [11].

Max_Min .۲,۳

الگوریتم Max_Min مانند الگوریتم Min_Min است به جز اینکه تفاوت‌های زیر را دارد: پس از یافتن کوتاه‌ترین زمان کامل شدن برای کارها، بیشترین مقدار انتخاب می‌شود و ماشینی که کوتاه‌ترین زمان کامل شدن برای تمام کارها را دارد، انتخاب می‌شود. در پایان گره انتخاب شده و کار انتخاب شده تایید می‌شوند. سپس زمان آماده به کار گره با افزودن زمان اجرای وظیفه تخصیص داده شده به‌هنگام‌سازی می‌شود [11].

جدول ۱- روش‌های توازن بار و مزایا و معایب آنها

شماره منبع	سال چاپ	نوع روش استفاده شده	مزایا	معایب
[12]	2018	Central manager Algorithm [12]	• برای میزبان‌های متفاوت از کارهای پیچیده ایده‌آل است.	• دارای گلوگاه است، زیرا نیاز به هماهنگی عملیاتی بالایی دارد.
[13]	2019	Round Robin Algorithm [13]	• پرکاربرد است و راه‌اندازی آن آسان است. • با تعداد عملیات بیشتر از تعداد پردازنده‌ها به خوبی عمل می‌کند.	• در سرویس‌دهنده‌های با ورودی چندگانه بلااستفاده است و ممکن با بیش‌باری و خرابی روبرو شود. • امیدی به نتایج خوب نیست.
[9]	2020	Randomize Algorithm [14]	• ارائه گزینه‌ها سریع است. • پذیرش و سرعت بالایی دارد.	• در حالت‌های تحت فشار ممکن است بسیار کند باشد. • احتمال کمی وجود دارد که پاسخ‌های اشتباه دهد.
[13]	2019	Threshold Algorithm [13]	• مقدار آستانه‌ای ارتباط کمی با فرایندها دارد. • تخصیص‌های متنوعی به یک فرایند محلی داده می‌شود.	• اگر تمام فرایندها بیش از حد اجرا شوند، تمام رویه‌ها باید به صورت دستی تخصیص داده شوند.
[15]	2020	Opportunistic LB Algorithm [15]	• به سادگی از عهده کارهای اجباری همزمان برای گره برمی‌آید.	• وظایف به کندی برنامه‌ریزی می‌شوند، زیرا نمی‌تواند دوره زمانی اجرای کنونی گره‌ها را تعیین کند.
[16]	2015	OLB+LBMM[16]	• منابع با بازدهی بالایی مورد استفاده قرار می‌گیرند و رقابت کاری بهبود می‌یابد.	• کامل شدن و زمان اجرای وظایف گره‌ها در OLB در نظر گرفته نشده است. به Main دلیل زمان زیادی برای کامل شدن وظایف صرف می‌شود.
[17]	2019	Min-Min LB Algorithm [17]	• ساده و سریع است. • زمان انجام کلی را بهبود می‌بخشد.	• منجر به گرسنگی می‌شود.

[13], [18]	2019, 2019	Max-Min LB Algorithm[18], [13]	• کارایی بهتر از Min-Min • تعداد وظایف کوچ نسبت به وظایف طولانی بیشتر است.	• از گرسنگی رنج می‌برد.
[19]	2020	Equally Spread Current Execution Algorithm [19]	• تقویت زمان بارگذاری مرکز داده و زمان‌های پاسخ	• باید زمان پردازش را بهتر کند. • افزایش هزینه‌ها
[20]	2018	Central LB for VMs [20]	• افزایش کارایی کل فرایند	• سیستم‌های مقاوم در برابر خطا را در نظر نگرفته است.
[21]	2018	Throttled LB [21]	• بیشینه کردن استفاده از منابع • کاهش قابل توجه میانگین زمان اجرا	• در موقعیت بار کاری خاصی شبیه‌سازی نشده است. • محدودیت‌های زمانی را نمی‌یابد.
[22]	2013	Stochastic Hill Climbing [22]	• حمله به مسئله گلوگاه • توزیع بار کاری موثر سیستم	• راه حل نامناسب برای حل مسائل بهینه‌سازی
[23]	2018	Join Idle Queue [23]	• هزینه سربار اتصال پایین • مقیاس پذیری بالا • زمان پاسخ کوتاه	• پیچیدگی بالا • منابع همگون
[24]	2020	Least Connections [24]	با بررسی تعداد ارتباط‌های سرویس‌دهنده از بار بیش از حد روی یک سرویس‌دهنده پیشگیری می‌کند.	هنگام محاسبه تعداد ارتباط‌های موجود، نمی‌توان ظرفیت سرویس‌دهنده را در نظر گرفت.
[10]	2019	Weighted Round Robin [10]	می‌توان برای سرویس‌دهنده‌های با بار بالایی که ظرفیت بیشتری دارند، درخواست‌های بیشتری فرستاد.	تمام برآوردها نیاز به پیاده‌سازی این الگوریتم دارند و این اشکال بزرگی است. همچنین نیاز به برآورد شبکه‌های IP با اندازه‌های بسته مختلف دارد که انجام آن دشوار است.
[16]	2015	Source Hash [16]	کاربران پس قطع اتصال و اتصال مجدد به نشستی وصل می‌شوند که هنوز فعال است که باعث افزایش کارایی می‌شود.	• ISPها آدرس‌های IP پویا تولید می‌کنند، بنابراین نگهداری آنها دشوار است.
[25]	2020	Least Response Time [25]	ظرفیت سرویس‌دهنده، زمان پاسخ و تعداد اتصال‌های کنونی را در نظر می‌گیرد تا از بیش‌باری و شکست پیشگیری کند.	• از ماشین‌های مجازی با عملکرد ساده استفاده شده و ممکن است باعث ترافیک نابرابر شود. • برای برنامه‌های با نشست‌های مبتنی بر کوکی توصیه نمی‌شود.
[26]	2012	Least Bandwidth [26]	می‌توان برای سرویس‌دهنده‌های با بار بالایی که ظرفیت بیشتری دارند، درخواست‌های بیشتری فرستاد.	• نیاز به پهنای باند تقریبی درباره شبکه دارد که در شبکه‌های با اندازه بسته متغیر دشوار است. • ممکن است پهنای باند شبکه فرسوده شود.
[27]	2020	Adaptive LB [27]	تعصبی روی روش خاصی ندارد و بسته به شرایط بهترین روش توازن بار را انتخاب می‌کند.	• روی سیستم‌های موازی پیاده‌سازی نشده است. • سوئیچ کردن میان الگوریتم‌ها سربار دارد.

۲,۴. توازن بار Min_Min

LBMM یک الگوریتم توازن بار ثابت است. این الگوریتم توازن بار میان گره‌ها را با در نظر گرفتن این کار به عنوان یک مسئله زمان‌بندی پیاده‌سازی می‌کند. هدف اصلی این الگوریتم کاهش زمان صرف شده (زمان کل) است که به صورت بیشینه زمان کامل شدن تمام کارهای زمان‌بندی شده به منابع است. این الگوریتم گامهای زیر را برای زمان‌بندی کارها روی گره‌ها انجام می‌دهد.

- الگوریتم زمان‌بندی Min-Min را اجرا کن و زمان صرف شده را محاسبه می‌کند.
- گره با بیشترین زمان انجام را محاسبه می‌کند.
- کوتاه‌ترین زمان اجرای مربوط به گره را محاسبه می‌کند.
- زمان کامل شدن کار مربوطه برای تمام منابع محاسبه می‌شود.
- بیشینه زمان کامل شدن کار انتخاب شده و گامها تکرار می‌شوند.
- فرایند هنگامی متوقف می‌شود که تمام کارها و تمام گره‌ها تخصیص داده شوند.
- در سناریوهایی که در meta-task تعداد کارهای کوتاه بیشتر از کارهای بلند است، این الگوریتم کارایی بهتری دارد. این الگوریتم ناهمگونی پایین و بالای ماشین‌ها و ناهمگونی وظایف را در نظر نمی‌گیرد [11].

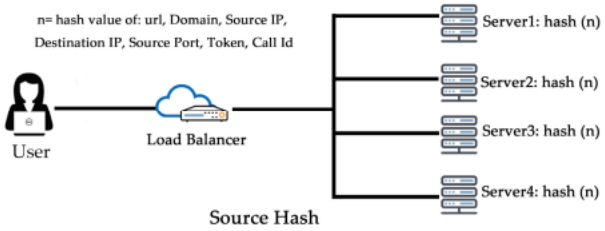
۲,۵. توازن بار Max-Min-Max

سی. وانگ و همکاران[۱] [28] یک الگوریتم زمان‌بندی دو مرحله‌ای پیشنهاد کرده‌اند که OLB را LBMM ترکیب می‌کند. الگوریتم زمان‌بندی OLB تمام گره‌ها را در وضعیت در حال کار نگه می‌دارد تا به هدف توازن بار برسد و از الگوریتم زمان‌بندی LBMM برای کمینه کردن زمان اجرای هر کدام از وظایف روی گره استفاده می‌شود و بنابراین زمان کامل شدن کلی کمینه می‌شود. این رویکرد ترکیبی شامل مراحل زیر است:

۱. میانگین زمان اجرای هر وظیفه فرعی (زیروظیفه) محاسبه می‌شود.
۲. اگر زمان مورد نیاز برای اجرای وظیفه فرعی کمتر یا مساوی میانگین زمان اجرا بود، آنگاه وظیفه فرعی را انجام می‌دهد تا به طور معمول به پایان برسد.
۳. اگر زمان مورد نیاز برای انجام وظیفه فرعی بیشتر از میانگین زمان اجرا بود، زمان اجرا برابر بینهایت در نظر گرفته می‌شود (زمان اجرا بسیار طولانی است و نمی‌تواند در نظر گرفته شود). دیگر گره‌هایی که اجرا می‌شوند، دوباره وارد سیستم خواهند شد تا در انجام وظایف فرعی مشارکت کنند.
۴. گامهای ۱ تا ۳ را تکرار می‌کند تا تمام کارهای فرعی کامل شوند.

۲,۶. درهم سازی[۲] منابع

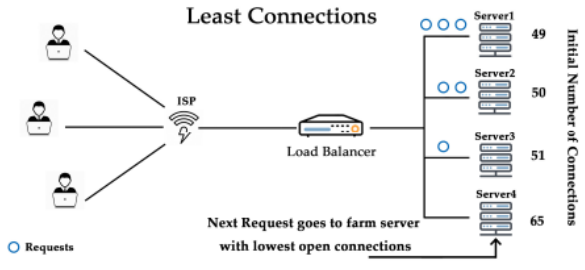
در این الگوریتم آدرس IP منبع (کاربر) دقیقا مشخص می‌کند که کدام سرویس‌دهنده به درخواست او پاسخ می‌دهد. تصویر ۲ این الگوریتم را نمایش می‌دهد. این الگوریتم هنگامی مورد استفاده قرار می‌گیرد که فراهم کننده برنامه می‌خواهد کاربر را به همان سرویس‌دهنده‌ای هدایت کند که آخرین درخواستش را انجام داده است. انجام این کار دشوار است زیرا ISPها از آدرس IP پویا استفاده می‌کنند [24] .



تصویر ۲- الگوریتم درهم ساز IP [24]

۲,۷. کمترین اتصال

همانطور که از نامش پیداست درخواست‌های کاربران را به سرویس‌دهنده‌ای منتقل می‌کند که کمترین تعداد اتصال‌های فعال را داشته باشد. در تصویر ۳ نمایش داده است که این الگوریتم چگونه درخواست‌های کاربران را میان سرویس‌دهنده‌های موجود توزیع می‌کند. این الگوریتم هنگامی که نشست‌های طولانی داشته باشیم، کاربرد بیشتری دارد. این الگوریتم برای کاربردهای با نشست‌های کوتاه مانند HTTP توصیه نمی‌شود اما در نشست‌های طولانی مانند LDAP، SQL و چیزهایی از این دست مفید است [24].



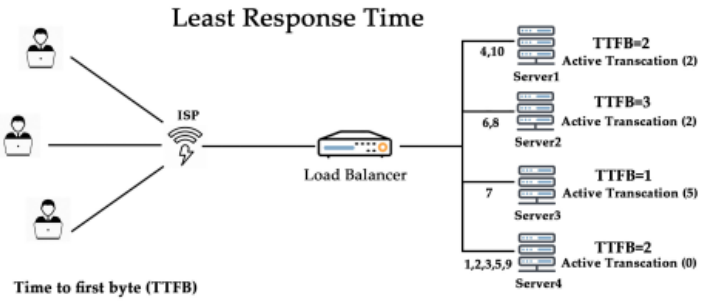
تصویر ۳- الگوریتم با کمترین خطوط ارتباطی [24]

[۱] Shu-Ching Wang et al.

[۲] Hashing

۲،۸.کوتاه‌ترین زمان پاسخ

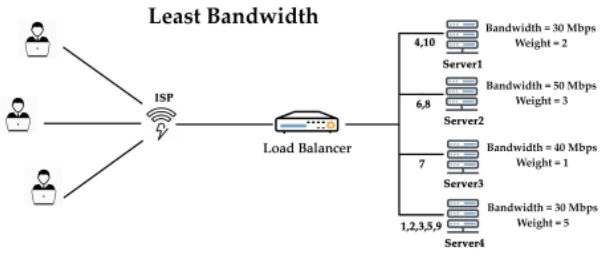
الگوریتم کوتاه‌ترین زمان پاسخ مطابق تصویر ۴ بازدیـد کننـدگان را به سرویس‌دهنده‌ای هدایت می‌کند که کمترین مقـدار اتصـال‌هـای فعال و کوتاه‌ترین زمان پاسخ را دارد. اتصال‌هایی استفاده مـی‌شـوند که دو پارامتر (کمترین اتصال‌های فعال و زمان پاسخ کوتاه) کمتری برای هر ماشین مجازی داشته باشند. این الگوریتم نیاز بـه ماشـین مجازی دارد. در صورتی که از ماشین‌های مجازی با عملکـرد سـاده استفاده شود، ممکن است ترافیک مسیر نامساوی مشاهده شود. این الگوریتم برای نشست‌های کاربردی مبتنی بر کوکی مناسب نیست.



تصویر ۴- کوتاه‌ترین زمان پاسخ [24]

۲،۹. کمترین پهنای باند

این الگوریتم کاربران را براساس مگابیت بر ثانیه (Mbps) پـردازش کـرده و مطـابق تصـویر ۵ درخواسـت‌هـای کـاربران را بـه سرویس‌دهنده‌های با کمترین نرخ بازدید کاربران ارسـال مـی‌کنـد. این الگوریتم هنگامی که تمام ماشین‌های مجازی در زیرساخت ابـر پهنای باند متفاوتی دارند، بسیار پرکاربرد اسـت. ایـن الگـوریتم بـه پهنای بانـد تقریبـی شـبکه نیـاز دارد کـه بـه دسـت آوردن آن در شبکه‌های با اندازه بسته‌های متنوع متفاوت است.



تصویر ۵- الگوریتم کمترین پهنای باند [24]

۳. معیارهای سنجش توازن بار

برای دستیابی به کارایی بهتر و مدیریت بهتـر منـابع، یـک متـوازن کننده بار باید دائما بار کـاری را میـان منـابع موجـود توزیـع کنـد. روش‌های توزیع بار متفاوت و معیارهای سنجش کارایی مختلفی در منابع توسط پژوهشگران برای مقایسه روش‌های مختلف توزیـع بـار در ابر ارائه شده‌اند تا بتوانند کارایی کلی سیسـتم را بهینـه کـرده و رضایت کاربران را افـزایش دهنـد[28]. در پاسـخ بـه پرسـش دوم برخـی از معیارهـای سـنجش تـوازن بـار مطـرح شـده در منـابع، عبارت‌اند از:

- **کارایی**[۱]: کارایی در محیط ابر، معیاری است که به صورت نسبت سرویس‌های مصرف شده در طول زمان به ظرفیت تقاضا تعریف می‌شود [29].
- **میانگین زمان پاسخ**: زمان کلی صرف شده برای انجـام درخواست ثبت شـده بـر روی سیسـتم پـس از پـردازش درخواست [24].
- **بازدهی**: مقدار کلی وظایف ثبت شده یا فرایندهای کامل شده در واحد زمان روی یک سیستم. بـازدهی بـالاتر بـه معنای کارایی بالاتر سیستم می‌باشد [11].
- **مقیاس پذیری (قابلیت گسـترش)**: توانـایی سیسـتم برای برآمدن از عهده توازن بار با افـزایش تعـداد گـره‌هـا [20].
- **تحمل خطا**: توانایی روش توازن بار برای ادامه یکنواخـت کـار در صـورت از کـار افتـادن گـره‌هـا یـا خطـوط ارتباطی[۲][30].
- **زمان مهاجرت**: زمان مهـاجرت[۳] بـه کـل زمـانی اطـلاق می‌شود که طول می‌کشد تا یک درخواست یا وظیفه را از یک ماشین با بیش‌باری به یک ماشین با کم‌بـاری انتقـال داده شود. هرچه زمان مهاجرت پـایین‌تـر باشـد، کـارایی سیستم ابر بالاتر است [30].
- **استفاده از منابع**: این ارزیابی برای اطمینان از اسـتفاده درست از تمام منابع موجود در سیستم ابر انجام می‌شود. استفاده بیشتر از منابع منجر به کاهش هزینه‌های کلـی و کاهش انرژی مصرفی سیستم ابر می‌شود [24].
- **درجه نامتوازن بودن**: میزان تغییرات میان ماشین‌هـای مجازی را توصیف می‌کند [31].
- **زمان صرف شده (زمان کـل)**[۴]: یکـی از معمـول‌تـرین معیارها برای سنجش کارایی زمان‌بندی در ابر می‌باشـد و به صورت زمـان اتمـام آخـرین کـار کامـل شـده تعریـف می‌شود. زمان کل کوچکتر نشان می‌دهـد کـه تخصـیص کارها به ماشین‌های مجازی به صورت مناسبی انجام شده است [32].

---

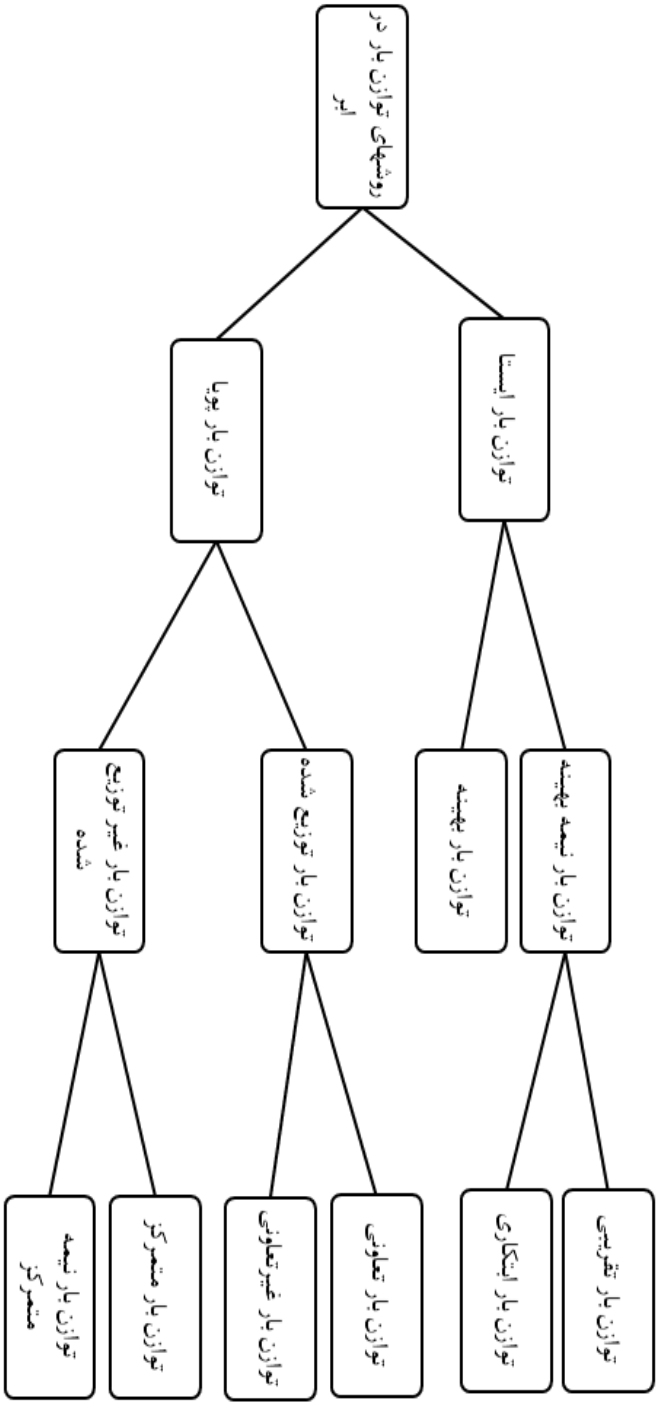
۱ Efficiency
۲ Links
۳ Migration time
۴ Make span

۴- توازن بار در ابر

پرسش نخست ما این است که چرا باید به توازن بار در ابر پرداخت؟ *پاسخ کوتاه این است* که هدف اصلی توازن بار پیشگیری از بیش‌باری یک سرویس‌دهنده و کم باری سرویس‌دهنده دیگر است، که بیش‌باری می‌تواند به افت کارایی و افزایش زمان پاسخ و در نهایت از کار افتادن سرویس‌دهنده منجر شود و کم باری هم باعث اتلاف منابع می‌شود [33]. ابر با ارائه مجموعه سرویس‌های متنوع نیاز دارد تا بتواند کیفیت این سرویس‌ها را با پایش سرویس‌های ارائه شده تضمین کند. ابر برای پاسخ به درخواست‌های کاربران با چالش‌های زیادی روبرو است که یکی از آنها توازن بار[۱] است [34]. زمان‌بندی یا تخصیص درخواست‌های کاربر – وظایف- یک مسئله بهینه‌سازی NP-hard است [35]. بنابراین برای ارائه سرویس‌های بهتر لازم است تا توازن بار در ابر (LBC) مورد مطالعه قرار گرفته و با بررسی چالش‌های آن از موقعیت‌هایی که باعث بار بیش از حد و بار کمتر از حد[۲] انتظار می‌شوند پیشگیری شود [2] و همچنین روش‌هایی برای توازن بار موثر ارائه گردد. برای توازن بار در ابر روش‌های گوناگونی ارائه شده‌اند که به دو دسته کلی روش‌های توازن بار ایستا و پویا تقسیم می‌شوند. روش‌های ایستا به دو دسته نیمه بهینه و بهینه تقسیم‌بندی می‌شوند. روش‌های پویا نیز به دو دسته توزیع شده و غیر توزیع شده تقسیم می‌شوند. در تصویر ۶ دسته‌بندی روش‌های توازن بار از این منظر به نمایش گذاشته شده‌اند.

الف) الگوریتم‌های ایستا: الگوریتم‌های ایستای توازن بار، بار را به طور متوازن میان سرویس‌دهنده‌ها تقسیم می‌کنند. در این سناریوها، ابر نیاز به یک دانش اولیه درباره اندازه سرویس‌دهنده، توان پردازش، حافظه، کارایی و نیازمندی‌های کاربران ابر دارد. در طول زمان اجرا نیازهای کاربران تغییر می‌کنند و الگوریتم‌های ایستا توانایی تطبیق با نیازهای در حال تغییر کاربران را ندارند. به عنوان نمونه‌هایی از این نوع الگوریتم‌ها می‌توان به الگوریتم‌هایی مانند راند روبین، راند روبین وزن‌دار و الگوریتم نخست کوتاه ترین کار[۳] (SJF) اشاره کرد.

ب) الگوریتم‌های پویا: الگوریتم‌های پویای توازن بار شرایط کنونی سرویس‌دهنده‌های ابر را روی شبکه بررسی کرده و براساس رویکرد به کار رفته سرویس‌دهنده مناسب با کمترین بار کاری را برای ارائه سرویس انتخاب می‌کنند. با توجه به تغییر نیازهای کاربران در هنگام اجرا، توازن بار در هنگام اجرا تغییر کرده و



الگوریتم توازن بار باید بتواند در هنگام اجرا با کاری روی سرویس‌دهنده‌ها را با توجه به شرایط موجود مدیریت کند [36].

تصویر ۶- دسته‌بندی استراتژی‌های توازن بار در سیستم‌های مختلف

۵. روش‌های فراابتکاری (الگوریتم‌های فراابتکاری)

در دهه اخیر روش‌های فراابتکاری، کارایی خود را در حل مسائل بهینه‌سازی زیادی در علم و صنعت نشان داده‌اند [37]. بهینه‌سازی فرایند تعیین متغیرهای تصمیم برای یک تابع است تا مقادیر آن را

---

[۱] Load Balancing
[۲] Overloading/underloading
[۳] Shortest Job First

کمینه یا بیشینه نماید. بیشتر مسائل دنیای واقعی محدودیت‌های غیرخطی، غیر محدب، پیچیده و با فضاهای تصمیم زیاد می‌باشند. بنابراین حل چنین مسائلی پیچیده و زمان‌بر است. چندین راه حل بهینه محلی برای حل چنین مسائلی وجود دارد که هیچکدام بهینگی کلی مسئله را تضمین نمی‌کنند.

برای حل چنین مسائلی روش‌های بهینه‌سازی فراابتکاری ارائه شده‌اند که قادر به حل این مسائل در تعداد دورهای محدود می‌باشند. الگوریتم‌های فراابتکاری از یک منظر به دو دسته کلی تقسیم می‌شوند:

- الگوریتم‌های مبتنی بر یک جواب
- الگوریتم‌های مبتنی بر جمعیت

در الگوریتم‌های مبتنی بر یک جواب، یک راه حل به طور تصادفی تولید شده و بهبود می‌یابد تا به جواب بهینه به دست آید ولی در الگوریتم‌های مبتنی بر جمعیت، به طور تصادفی مجموعه‌ای از جواب‌های تصادفی تولید می‌شوند و فضای جست‌وجوی داده شده و مقادیر راه حل، گام به گام به روز رسانی می‌شوند تا بهترین راه حل تولید شود. الگوریتم‌های مبتنی بر یک جواب ممکن است در دام جواب بهینه محلی بیفتند و نتوانند جواب بهینه کلی را بیابند، بنابراین امروزه بیشتر از الگوریتم‌های مبتنی بر جمعیت که توانایی ذاتی گریز از دام بهینگی محلی دارند، برای مسائل بهینه‌سازی استفاده می‌شود [38].

در این کار تعدادی از الگوریتم‌های مبتنی بر یک جواب و تعدادی از الگوریتم‌های مبتنی بر جمعیت پرکاربرد مطرح شده در منابع، مورد بررسی قرار گرفته‌اند:

- الگوریتم‌های فراابتکاری
  - الگوریتم‌های مبتنی بر یک جواب
    - تبرید شبیه‌سازی شده
    - جست‌وجوی ممنوعه
  - الگوریتم‌های مبتنی بر جمعیت
    - الگوریتم‌های تکاملی
      - الگوریتم جست‌وجوی فاخته
      - الگوریتم ژنتیک[1]
      - الگوریتم استراتژی عقاب
    - الگوریتم‌های ازدحامی
      - الگوریتم ازدحام ذرات
      - الگوریتم کلونی مورچه‌ها
      - الگوریتم جست‌وجوی زنبور عسل
      - الگوریتم زنبور عسل مصنوعی
      - الگوریتم عقاب
    - الگوریتم‌های مبتنی بر فیزیک
      - الگوریتم توازن فشار اسمزی
  - الگوریتم‌های ترکیبی
    - کلونی مورچه و زنبور عسل
    - ژنتیک و ازدحام ذرات
    - ...

در تصویر ۷ رده‌بندی پیشنهادی الگوریتم‌های جست‌وجوی فراابتکاری به کار رفته در ابر نمایش داده شده‌اند.

۵-۱- الگوریتم‌های مبتنی بر یک جواب

۵-۱-۱- تبرید شبیه‌سازی شده

تبرید شبیه‌سازی شده از تبرید مواد جامد مانند فلزات و شیشه الهام گرفته شده است که جامدات را گرم کرده و به آرامی سرد می‌کنند تا جنبش‌های درونی آن کمینه گردد. هدف این است که اندازه کریستال‌ها در حالت جامد به بزرگترین مقدار برسد. شبیه‌سازی تبرید رویکردی است که مسئله کمینه‌سازی یک تابع با تعداد بسیار زیادی متغیر را به یک مسئله مکانیک آماری کاهش می‌دهد. می‌توان تبرید تدریجی و آرام در این الگوریتم را به عنوان کاهش تدریجی احتمال انتخاب پاسخ‌های بدتر در هنگام جستجو در فضای پاسخ‌ها دانست [39]. در [32] از بهبود کارایی بهینه‌ساز شاهین هریس (HHO)[2] با استفاده از الگوریتم تبرید شبیه‌سازی شده به عنوان متدی جایگزین برای تخصیص کار در محیط ابر استفاده شده است.

۵-۱-۲- جست‌وجوی ممنوعه

الگوریتم جست‌وجوی ممنوعه یک الگوریتم بهینه‌سازی سراسری با هدف شبیه‌سازی عقل انسانی است و توانایی بهینه‌سازی بالایی دارد. هدف این الگوریتم مدیریت رویکردهای دیگر است تا در تله بهینه‌سازی محلی نیفتند و برای تخصیص منابع و دیگر مسائل بهینه‌سازی کاربرد دارد [30]. در [40] از یک معماری ارتباطی میان گره‌های ابر و مه استفاده شده است. هدف این معماری استفاده از مزایای توانایی‌های رایانشی گره‌های مه در لحظه زمان‌بندی وظایف و انتخاب گره‌های مشارکت کننده در اجرا می‌باشد. این معماری دارای سه لایه زیرساخت، مه و ابر می‌باشد. در این کار، از یک روش جست‌وجوی ممنوعه ساده برای توازن بار بهینه میان گره‌های مه و ابر استفاده کرده است و مسئله

[1] Genetic

[2] Harris's Hawks Optimizer

بهینه‌سازی دو هدفه (بیشینه کردن استفاده از لایه رایانش مه و کمینه کردن لایه رایانش ابر) را به یک مسئله بهینه‌سازی تک هدفه تبدیل نموده است.

۵-۲- الگوریتم‌های مبتنی بر جمعیت

۵-۲-۱- الگوریتم‌های تکاملی

۵-۲-۱-۱- الگوریتم جست‌وجوی فاخته (کوکو)

تکنیک جست‌وجوی فاخته یک الگوریتم فراابتکاری است که رفتار جنس فاخته را مدل‌سازی می‌کند. این الگوریتم با جابجایی متغیرها بهترین راه حل و کاراترین تعادل محلی را کشف می‌کند. مقادیر به دست آمده با این روش بهتر از بهینه‌سازی ازدحام ذرات می‌باشند. در [41] الگوریتم جست‌وجوی فاخته به همراه الگوریتم کرم شب‌تاب[۱] (CS-FA) برای توازن بار در محیط ابر پیشنهاد شده است. نخست ظرفیت و بار هر کدام از ماشین‌ها محاسبه می‌شود. اگر بار ماشین مجازی بیشتر از یک مقدار آستانه‌ای بود، الگوریتم توازن بار برای تخصیص وظایف استفاده می‌شود. این الگوریتم بهترین ماشین مجازی را برای تخصیص وظایف انتخاب کرده و وظایف ماشین‌های مجازی با بیش‌باری را به ماشین‌های با بار کم مهاجرت می‌دهد. این الگوریتم بیشتر از نامتوازن شدن بار در محیط ابر پیشگیری می‌کند. کارایی الگوریتم CS-FA با الگوریتم‌های موجود مقایسه شده و مشاهده شده است که در این الگوریتم تعداد مهاجرت‌ها به میزان قابل توجهی کمتر از روش‌های توازن بار دیگر می‌باشد. در [42] یک الگوریتم آگاه از بار مبتنی بر جست‌وجوی کوکو، با هدف کاهش زمان انجام کل و هزینه رایانش با داشتن محدودیت زمانی ارائه شده است. مدنی[۲] و همکاران، مسئله زمان‌بندی منابع را با در نظر گرفتن یک الگوریتم جست‌وجوی با چند هدف الهام گرفته از جست‌وجوی کوکو حل کرده‌اند [43].

۵-۲-۱-۲- الگوریتم ژنتیک

الگوریتم‌های ژنتیک دسته‌ای از الگوریتم‌های تکاملی هستند که روی سناریوهای با یک یا چند هدف اعمال می‌شوند. تصویر ۸ شکل کلی یک الگوریتم ژنتیک را بازنمایی می‌کند. در [44] یک الگوریتم ژنتیک تغییر یافته با بهینه‌سازی جست‌وجوی محلی[۳] (GA-LS) پیشنهاد شده است که طوری مدل‌سازی شده تا بتواند پارامترهای مصرف انرژی و مصرف حافظه را کاهش دهد. این الگوریتم برای مدیریت تخصیص و مهاجرت VMها مورد استفاده قرار گرفته است.

محیط ابر می‌تواند به عنوان کل جمعیت یا فضای پاسخ‌های کاندید در نظر گرفته شود. هر گره، کروموزومی است که باید براساس تابع برازش بهینه‌سازی شود. در این مورد مطالعاتی، تابع برازش یا تابع هدف می‌تواند مقدار حافظه یا توان استفاده شده باشد که در هر دو مورد هدف ما کمینه کردن آن است. آمیزش انجام شده یک آمیزش تک نقطه‌ای است که دو فرد جدید تولید شده‌اند.

معادله (۱) تابع برازش محلی را نمایش می‌دهد در حالی که معادله (۲) تابع برازش عمومی را بازنمایی می‌کند [14].

$$f\,t_{local}\,(i,j) = E\,(i,j) \times M\,(i,j) \qquad (۱)$$

$$f\,t_{global} = \sum_{j=0}^{n} f\,t_{local} \qquad (۲)$$

در (۱) $f\,t_{local}$ تابع برازش محلی زوج i و j، $E\,(i,j)$ انرژی و $M(i,j)$ حافظه مصرفی می‌باشد. جست‌وجوی محلی یک متد ابتکاری است که برای جست‌وجوی فضای راه حل مسئله به کار گرفته می‌شود و به طور محلی راه‌حل‌ها را تغییر داده یا به‌هنگام‌سازی می‌کند تا بهترین گزینه‌ها را برای عمل آمیزش بیابد. در این مورد بهترین گزینه‌ها، آنهایی هستند که بهترین ژنها یا بهترین برازش را از نظر مصرف انرژی و حافظه دارند.

در [30] الگوریتمی ارائه شده است که تلاش می‌کند بار روی ابر را با کاهش زمان وظیفه مورد نظر متعادل کند. الگوریتم مذکور یک الگوریتم تصادفی است که روی پردازش انتخاب طبیعی و ژنتیک تکیه کرده است. یک الگوریتم ژنتیک ساده از سه فرایند تشکیل شده است: ۱) در دسترس بودن[۴] ۲) ژنتیک و ۳) عملیات جایگزینی[۵]. در [45] یک الگوریتم زمان‌بندی وظیفه مبتنی بر الگوریتم ژنتیک برای تخصیص منابع به وظایف پیشنهاد شده است. این الگوریتم با ۱) بهبود الگوریتم ژنتیک با استفاده از جدول محدود نوآورانه ۲) در نظر گرفتن نیازهای سطح سرویس[۶] کاربران و ۳) کاهش زمان با محاسبه موازی مقدار برازش کروموزوم‌ها توانسته نرخ توازن بار وظایف ورودی را بهبود بخشد. در [46] یک چارچوب توازن بار بهینه چند هدفه برای بهبود استفاده از منابع و کاهش توان مصرفی مرکز داده پیشنهاد شده است. در این کار پژوهشی از یک عملگر انتخاب هدایت شده به وسیله مرتب‌سازی

---

۱ Firefly Algorithm
۲ Madni
۳ Genetic Algorithm with Local Search
۴ Availability
۵ Replacement
۶ Service-level agreement

ناغالب[۱] استفاده شده است که توانایی انتخاب مناسب‌ترین افراد از میان جمعیتی با چندین مقدار هزینه با در نظر گرفتن اهداف تخصیص ماشین‌های مجازی را دارد.

سویی[۲] و همکاران الگوریتم ژنتیکی پیشنهاد کرده‌اند که می‌تواند با دقت میزان استفاده از پردازنده در سرویس‌دهنده‌ها را پیش‌بینی کند که هدف آن حل مشکل تعداد خوشه‌ها و انتخاب مراکز خوشه اولیه در الگوریتم k-means می‌باشد. در این کار یک الگوریتم خوشه‌بندی k-means مبتنی بر min-max بهینه‌سازی شده ارائه شده است. این الگوریتم ماشین‌های مجازی با هزینه مهاجرت و ترافیک شبکه کمتر بر روی سرویس‌دهنده با بیش‌باری را می‌یابد. در این کار پژوهشی الگوریتم تکاملی تفاضلی سنتی بهبود یافته است و یک الگوریتم تکاملی تفاضلی با قابلیت تطبیق پیشنهاد شده است تا توانایی جست‌وجوی محلی را بهبود بخشد. ماشین مجازی یافته شده به وسیله الگوریتم خوشه‌بندی با کمترین هزینه، ترافیک شبکه و اختلال در کارایی به سرویس‌دهنده هدف مهاجرت داده می‌شود تا هدف توازن بار در مرکز داده ابر محقق شود [18].

در [47] یک الگوریتم توازن بار برای ابر لبه‌ای[۳] پیشنهاد شده است که به نحو موثری وظایف تخلیه شده از نقطه کانونی[۴] به سرویس‌دهنده نزدیک لبه[۵] را توزیع می‌کند. الگوریتم پیشنهادی براساس تئوری رنگ کردن گرافها و با پیاده‌سازی یک الگوریتم ژنتیک که پیچیدگی الگوریتم را کاهش می‌دهد، عمل می‌نماید. نتایج ارائه شده نشان می‌دهد که الگوریتم توازن بار پیشنهادی در این کار پژوهشی بهتر از تکنیک‌های پیشین است و نرخ استفاده از پردازنده ماشین‌های مجازی را افزایش می‌دهد که نشان دهنده بهره‌وری بالاتر سرویس‌دهنده‌های لبه‌ای است [48].

در [49] یک سازوکار توازن بار با الگوریتم دودویی مبتنی بر جایا[۶] پیشنهاد شده است که سیستم را متعادل کرده و یک نگاشت مناسب از وظایف بر روی VMها انجام می‌دهد. این تخصیص وظیفه-ماشین مجازی قابل مقایسه با ذرات و منبع غذایی است. این سازوکار بر روی مهاجرت وظایف از VMهای با بیش‌باری به VMهای با کم‌باری متمرکز شده است و تاثیر قابل توجهی بر روی کاهش زمان کل و زمان پاسخ دارد. جایا یک الگوریتم بهینه‌سازی ساده و قدرتمند است که برای مسائل بهینه‌سازی دارای محدودیت و بدون محدودیت پیشنهاد شده است. این الگوریتم براین اساس استوار است که راه حل به دست آمده برای یک مسئله باید به سمت بهترین راه حل برود و باید از بدترین راه حل اجتناب کند. این الگوریتم تنها نیاز به پارامترهای کنترل عمومی دارد و نیازی به پارامترهای کنترل مخصوص این الگوریتم ندارد [50].

در [19] یک استراتژی توازن بار و بهینه‌سازی[۷] (LBOS) پیشنهاد شده است که از متد تخصیص منابع مبتنی بر یادگیری تقویت شده[۸] و الگوریتم ژنتیک استفاده می‌کند. LBOS پیوسته ترافیک شبکه را پایش کرده، اطلاعاتی درباره بار هر سرویس‌دهنده جمع‌آوری نموده، درخواست‌های ورودی را مدیریت کرده و آنها را با استفاده از متد تخصیص پویای منابع، میان سرویس‌دهنده‌های در دسترس توزیع می‌کند. بنابراین حتی زمان اوج بار نیز کارایی را بهبود می‌دهد. LBOS یک سیستم ساده و کارا در سیستم‌های بلادرنگ[۹] در رایانش مه مانند سیستم‌های مراقبت درمان می‌باشد. سیستم IOT مه پیشنهادی از سه لایه تشکیل شده است که عبارت‌اند از: ۱) لایه IoT، ۲) لایه مه و ۳) لایه ابر. ماموریت لایه IoT پایش علائم بیمار است. لایه مه وظیفه مدیریت درخواست‌های ورودی و ارسال آنها به سرویس‌دهنده را دارد. لایه ابر مسئول مدیریت ارسال و دریافت داده‌ها به لایه مه است. هدف اصلی این سیستم دستیابی به تاخیر پایین است. لایه مه از دو ماژول عامل متوازن کننده بار (LBA) و تخصیص دهنده منابع (RA) تشکیل شده است. LBA نرم‌افزاری است که مسئول انتخاب سرویس‌دهنده مه مناسب برای درخواست‌های ورودی است. ماژول RA مبتنی بر الگوریتم RL است تا به توازن بار بالایی برای محیط دست یابد.

در [51] یک روش نوآورانه مبتنی بر الگوریتم ژنتیک برای زمان‌بندی وظایف ارائه شده است که زمان کل و توازن بار را بهبود می‌دهد. روش پیشنهادی با نام الگوریتم متوازن کننده ژنتیک[۱۰] (BGA) یک تابع برازش برای بهینه‌سازی سازی چند هدفه را فرمول بندی کرده است. نتایج ارائه شده نشان می‌دهند که BGA نسبت به زمان‌بندی های دیگر زمان کل، کارایی و بازدهی بهتری دارد.

---

[۱] Nondominated sorting
[۲] Sui
[۳] Edge Cloud
[۴] Hotspot
[۵] Edge
[۶] JAYA
[۷] Load Balancing and Optimization Strategy
[۸] Reinforcement learning
[۹] Real time
[۱۰] Balancer Genetic Algorithm

۵-۲-۱-۳- استراتژی عقاب

در [52] یک استراتژی عقاب نوآورانه با الگوریتم بهینه‌سازی چاه[۱] (ESWOA) ارائه شده تا کشف، بهره‌گیری مصرف سرویس‌ها در ابر آگاه از کیفیت را متعادل کند.

در این مقاله سه معیار ساده برای بررسی اولویت در نظر گرفته شده است [26]:

- First Come First Serve (FCFS): یک طرح زمان بندی است که در آن به درخواستی که زودتر ارائه شده زودتر پاسخ داده می‌شود.
- Smallest Job First (SJF): یک طرح زمان‌بندی وظیفه است که به فرایندهای کوچکتر، زودتر پردازش می‌شوند.
- Largest Job First (LJF): یک طرح زمان‌بندی وظیفه است که کارهای طولانی‌تر زودتر پردازش می‌شوند.

۵-۲-۲- الگوریتم‌های ازدحامی

۵-۲-۲-۱- بهینه‌سازی ازدحام ذرات (اجزاء)

یک الگوریتم بهینه‌سازی پیچیده زیستی است که رفتار جمعی حیوانات شبیه‌سازی می‌کند. در ابتدا این الگوریتم به منظور کشف الگوهای حاکم بر پرواز هم‌زمان پرندگان و تغییر مسیر ناگهانی آن‌ها و تغییر شکل بهینه دسته به کار گرفته شد. در PSO، ذرات در فضای جستجو جاری می‌شوند. تغییر مکان ذرات در فضای جست‌وجو تحت تأثیر تجربه و دانش خودشان و همسایگانشان است. بنابراین موقعیت توده‌های ذرات روی چگونگی جست‌وجوی یک ذره اثر می‌گذارد. نتیجه مدل‌سازی این رفتار اجتماعی فرایند جست‌وجویی است که ذرات به سمت نواحی موفق میل می‌کنند. ذرات از یکدیگر می‌آموزند و بر مبنای دانش بدست آمده به سمت بهترین همسایگان خود می‌روند اساس کار PSO بر این اصل استوار است که در هر لحظه، هر ذره مکان خود را در فضای جست‌وجو با توجه به بهترین مکانی که تاکنون در آن قرار گرفته‌است و بهترین مکانی که در کل همسایگی‌اش وجود دارد، تنظیم می‌کند [53].

میشرا[۲] و همکاران یک سازوکار توازن بار مبتنی بر الگوریتم زمان‌بندی JAYA ارائه کرده‌اند تا یک سیستم متوازن به دست آید و سپس نگاشت مناسب به VMها انجام شده است. در این پژوهش یک سازوکار توازن بار ارائه شده تا تعادل منابع را با توجه به بار آنها حفظ کند. این سازوکار بر روی مهاجرت وظایف از ماشین‌های با بار بیشینه به ماشین‌های با بار کمینه تمرکز کرده است و منجر به کاهش قابل توجه در زمان کل و زمان پاسخ شده است [49].

۵-۲-۲-۲- الگوریتم دسته ماهی‌ها

الگوریتم فراابتکاری دسته ماهی‌ها از رفتار دسته ماهی‌ها الهام گرفته شده است. این متد به رفتار هوشمندانه دسته ماهی‌ها پایبند است که برای یافتن غذا در یک نقطه تمرکز می‌کنند. در [54] از الگوریتم دسته ماهی‌های مصنوعی برای توازن بار و مهاجرت وظایف و کشف ماشین‌های با بیش‌باری در محیط ابر استفاده شده است. در [55] از یک الگوریتم دسته ماهی‌های مصنوعی تغییر یافته برای تخصیص کارای وظایف در محیط ابر استفاده شده و الگوریتم پیشنهادی در مقایسه با الگوریتم‌های موجود مانند الگوریتم ژنتیک و ازدحام ذرات از نظر هزینه اجرا و زمان کل، کارایی بهتری دارد.

۵-۲-۲-۳- الگوریتم بهینه‌سازی ازدحام گربه‌ها

یک الگوریتم زمان‌بندی هوشمند ابتکاری است که مبتنی بر رفتار اجتماعی خانواده گربه‌هاست. نتایج به دست آمده مقدار کلی انرژی مصرفی را بهینه‌سازی می‌کند. این الگوریتم همچنین یک تابع بهینه‌سازی زمان‌بندی منابع ارائه می‌کند که هزینه‌های زمان‌بندی را کمینه می‌کند. این الگوریتم با کاهش اندازه نمونه‌ها، در واقع یک بهینه‌سازی PSO می‌باشد. در [56] با استفاده از الگوریتم ازدحام گربه‌ها، کاهش قابل توجهی در مصرف انرژی و زمان اجرا به دست آمده است. البته آنچه در این کار جلب توجه می‌کند عدم مقایسه کار با دیگر کارهای مشابه در پارامترهایی مانند زمان کل، هزینه و چیزهای از این دست می‌باشد.

۵-۱-۲- جست‌وجوی ممنوعه

الگوریتم جست‌وجوی ممنوعه یک الگوریتم بهینه‌سازی سراسری با هدف شبیه‌سازی عقل انسانی است و توانایی بهینه‌سازی بالایی دارد. هدف این الگوریتم مدیریت رویکردهای دیگر است تا در تله بهینه‌سازی محلی نیفتند و برای تخصیص منابع و دیگر مسائل بهینه‌سازی کاربرد دارد [30]. در [40] از یک معماری ارتباطی میان گره‌های ابر و مه استفاده شده است. هدف این معماری استفاده از مزایای توانایی‌های رایانشی گره‌های مه در لحظه زمان‌بندی وظایف و انتخاب گره‌های مشارکت کننده در اجرا می‌باشد. این معماری دارای سه لایه زیرساخت، مه و ابر می‌باشد.

[۱] Eagle Strategy with Whale Optimization Algorithm
[۲] Mishra

در این کار، از یک روش جست‌وجوی ممنوعه ساده برای توازن بار بهینه میان گره‌های مه و ابر استفاده کرده است و مسئله بهینه‌سازی دو هدفه (بیشینه کردن استفاده از لایه رایانش مه و کمینه کردن لایه رایانش ابر) را به یک مسئله بهینه‌سازی تک هدفه تبدیل نموده است.

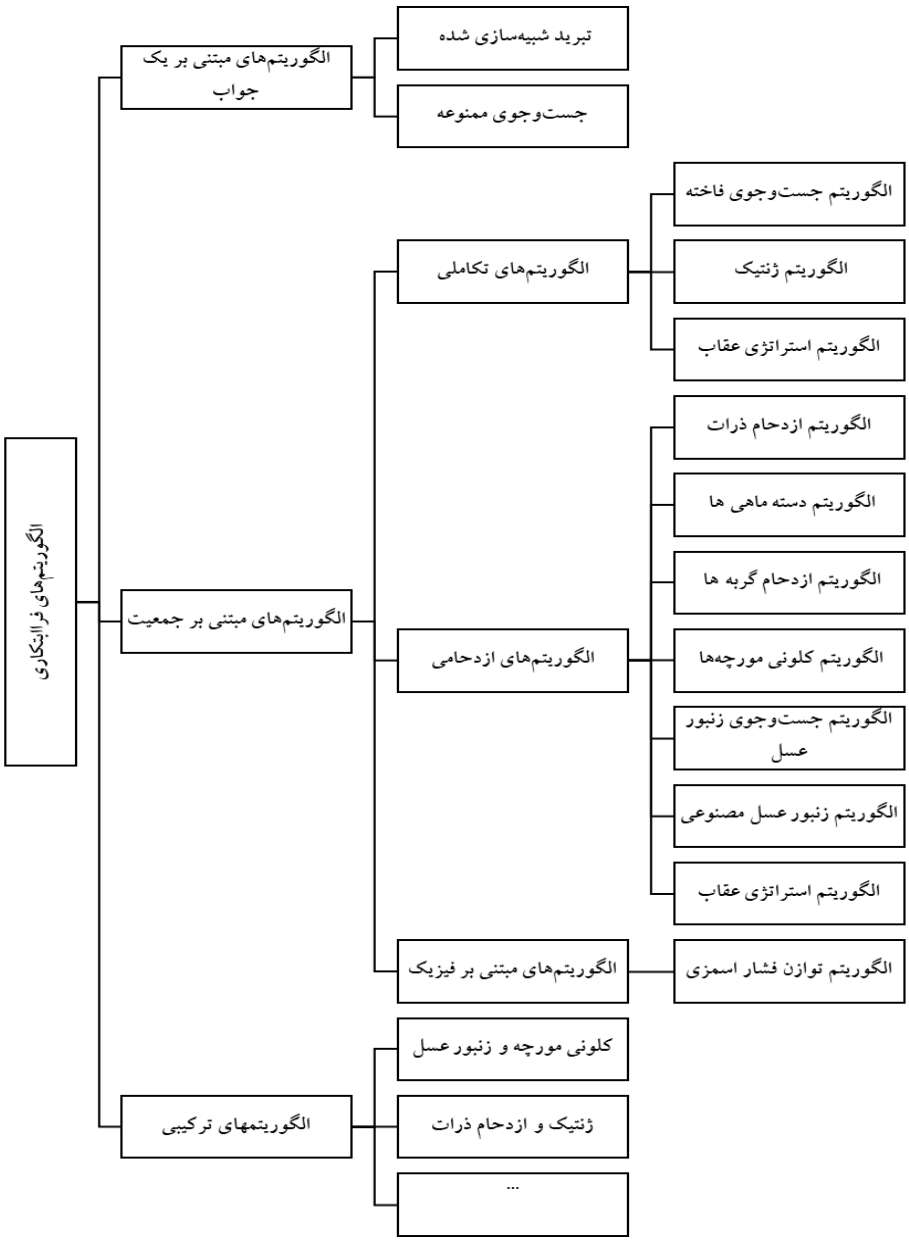
۵-۱-۲- جست‌وجوی ممنوعه

الگوریتم جست‌وجوی ممنوعه یک الگوریتم بهینه‌سازی سراسری با هدف شبیه‌سازی عقل انسانی است و توانایی بهینه‌سازی بالایی دارد. هدف این الگوریتم مدیریت رویکردهای دیگر است تا در تله بهینه‌سازی محلی نیفتند و برای تخصیص منابع و دیگر مسائل بهینه‌سازی کاربرد دارد [30]. در [40] از یک معماری ارتباطی میان گره‌های ابر و مه استفاده شده است. هدف این معماری استفاده از مزایای توانایی‌های رایانشی گره‌های مه در لحظه زمان‌بندی وظایف و انتخاب گره‌های مشارکت کننده در اجرا می‌باشد. این معماری دارای سه لایه زیرساخت، مه و ابر می‌باشد. در این کار، از یک روش جست‌وجوی ممنوعه ساده برای توازن بار بهینه میان گره‌های مه و ابر استفاده کرده است و مسئله بهینه‌سازی دو هدفه (بیشینه کردن استفاده از لایه رایانش مه و کمینه کردن لایه رایانش ابر) را به یک مسئله بهینه‌سازی تک هدفه تبدیل نموده است.

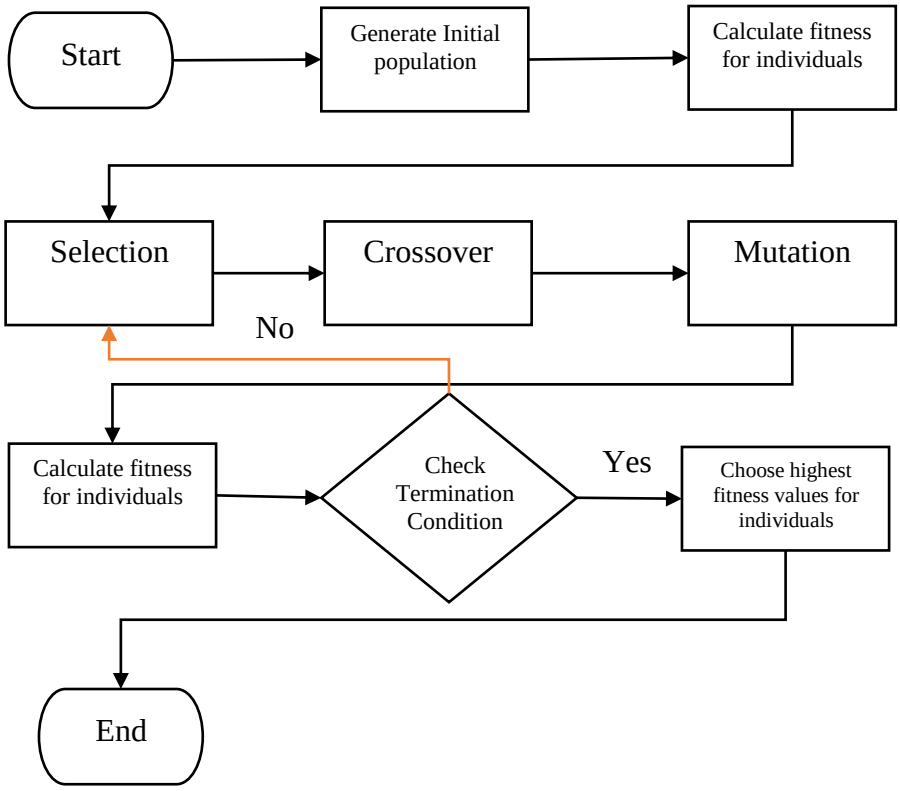
۵-۲-۲-۵- الگوریتم جست‌وجوی زنبور عسل

ایده اصلی پشت الگوریتم جست‌وجوی زنبور عسل از رفتار زنبورهای عسل الهام گرفته شده است: جوینده‌گان و برداشت کننده‌ها. زنبورهای جوینده به بیرون کندو می‌روند و منابع شهد را می‌یابند. پس از یافتن شهد به کندو برگشته و با انجام رقص حرکتی کیفیت و کمیت عسل قابل استخراج را توصیف می‌کنند. سپس برداشت کنندگان به بیرون رفته و عسل را از منابع استخراج می‌کنند. پس از جمع‌آوری شهد، به کندو برگشته و رقص حرکتی را انجام می‌دهند. این رقص نشان می‌دهد که چه مقدار شهد باقی مانده است. ام. راندلس[1] یک الگوریتم نامتمرکز مبتنی بر جست‌وجوی زنبور عسل را برای خودسازماندهی پیشنهاد داده است. در این الگوریتم سرویس‌دهنده‌ها به صورت سرویس‌دهنده مجازی گروه‌بندی می‌شوند و هر سرویس‌دهنده مجازی یک صف فرایند دارد. هر سرویس‌دهنده پس از پردازش یک درخواست از صف خود، سودی مانند سود نمایش داده شده با رقص حرکتی زنبورها را محاسبه می‌کند. اگر سود بالا باشد، سرویس‌دهنده باقی می‌ماند، در غیر این صورت به جست‌وجو ادامه می‌دهد. این الگوریتم نیاز دارد که هر گره یک صف جداگانه داشته باشد. این روش محاسبه سود، منجر به سربار اضافی می‌شود. اشکال این الگوریتم این است که هیچ بهبود قابل توجهی در کارایی نشان نمی‌دهد که به دلیل صف اضافی و سربار محاسباتی است.

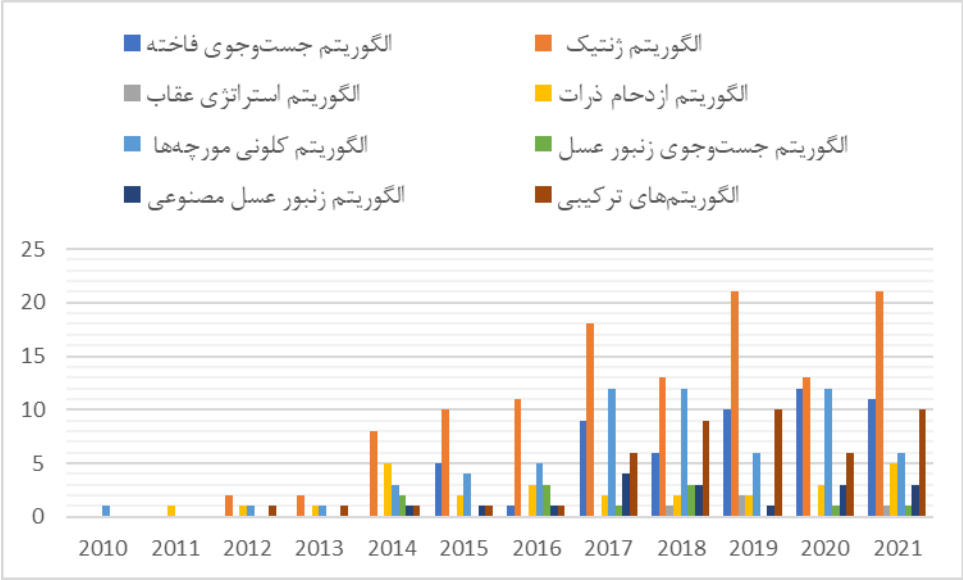
[1] M. Randles



تصویر ۷- رده‌بندی پیشنهادی الگوریتم‌های فراابتکاری استفاده شده در توازن بار در ابر



تصویر ۸- فرایند الگوریتم ژنتیک [47]



تصویر ۹- روند استفاده از الگوریتم‌های فراابتکاری برای متوازن سازی بار ابر

۵-۲-۲-۶- الگوریتم بهینه‌سازی کلونی زنبور

در [16] یک متد بهینه‌سازی الهام گرفته شده از فرایند انتخابِ گزینه برای توازن بار با استفاده از الگوریتم کلونی زنبور مصنوعی ارائه شده است. بهینه‌سازی کلونی زنبور یک چرخه انتخاب گزینه برای انتخاب بهترین گزینه جست‌وجوی غذا از میان فرصت‌های موجود ارائه می‌کند. فرایند انتخاب گزینه بستگی به ازدحام دارد.

در [16] الگوریتم HBF براساس روش جست‌وجوی غذای زنبور عسل پیشنهاد شده است. در اینجا یک کلونی از زنبورها وجود دارد که به دنبال تامین غذا هستند. زنبورها هنگام یافتن غذا یک رقص امضا گونه انجام می‌دهند که رقص حرکتی نامیده می‌شود. آنها با انجام رقص حرکتی به لانه بازمی‌گردند تا خبر یافتن غذا را اعلان کنند. این رقص شامل حرکاتی است که مربوط به مقدار غذا و فاصله آن تا لانه است.

۵-۲-۲-۷- کلونی زنبور مصنوعی[۱]

کلونی زنبور مصنوعی (ABC) یک الگوریتم ازدحام هوشمندانه است که از رفتار جست‌وجوی غذای زنبورهای عسل الهام گرفته است صورت تغییر یافته‌ای از بهینه‌سازی کلونی زنبور (BCO) می‌باشد که نخستین بار در سال ۲۰۰۱ پیشنهاد شد [62].

تانکا[۲] و همکاران در [63] یک الگوریتم فرا ابتکاری ازدحام هوشمند کلونی زنبور مصنوعی پیشنهاد کرده‌اند که روش‌های جست‌وجوی کلونی زنبور عسل را تقلید می‌کند. این الگوریتم از سه عنصر کلیدی تشکیل شده است: زنبور کارگر، زنبور عسل کاوشگر و زنبور پیشاهنگ. زنبورهای کارگر ویژگی‌های شهدهای موجود در منطقه رقص کندو را جمع‌آوری می‌کنند. پس از این تبادل دانش در کندو، زنبورهای کارگر به محل منبع شهد که به خاطر سپرده‌اند، برمی‌گردند و فرایند پیشین تکرار می‌شود. در [64] یک متد جست‌وجوی بهینه شده مبتنی بر جمعیت ارائه شده که در آن موجودیت‌هایی به نام منبع غذایی با گذشت زمان براساس زنبورهای مصنوعی و هدف زنبورها تغییر می‌کنند تا منابع با مقدار شهد بیشتر را شناسایی کنند. در [65] منطقه رقصی مانند کندو برای تبادل اطلاعات و مهارت‌ها پیشنهاد شده است.

در ادامه [64] روش زیر پیشنهاد شده است. فرض کنیم $VM = \{vm_1, vm_2, vm_3, \dots, vm_m\}$ متغیرهای استفاده شده در این مطالعه باشند که تعریف آنها در جدول زیر آمده است. فرض کنیم سیستم طوری تنظیم شده که بدون وقفه کار کند. پیش از اجرای الگوریتم ABC وظایف با الگوریتمی ابتکاری به سه دسته تقسیم می‌شوند. هنگامی که وظایف طبق آنچه در بالا گفته شد، به وسیله الگوریتم ABC پردازش می‌شوند، بهترین چیدمان کمترین زمان محاسباتی را خواهد داشت. سپس الگوریتم ABC وظایف را زمان‌بندی می‌کند تا با گامهای زیر به VMها دسترسی داشته باشند.:

۱- تعداد زنبورها (n) در جمعیت مشخص می‌شود. آنها به طور تصادفی برای جست‌وجوی منابع غذای مختلف (m) تخصیص داده می‌شوند که در واقع VMها را بازنمایی می‌کنند و مقدار برازش آنها محاسبه می‌شود.

مطابق با ویژگی‌های VM، یک مقدار پیش‌فرض متفاوت به هر کدام از منابع غذایی تخصیص داده می‌شود. زنبورها به سه دسته تقسیم می‌شوند. الف) زنبور دیدبان که موقعیت اولیه منابع غذایی را می‌یابد. ب) زنبور کارگر که سراغ منابع غذایی می‌رود و محاسبه مجدد کرده و مقدار برازش منابع غذایی را به‌هنگام‌سازی می‌کند. ج زنبور ناظر) که تصمیم می‌گیرد که کدام منبع، بهترین منبع غذایی است.

۲- زنبورهای کارگر اطراف منبع غذا را جست‌وجو می‌کنند. آنها اطلاعاتی درباره منبع غذا برای ناظر می‌آورند. ناظر مقادیر برازش را محاسبه می‌کند.

۳- زنبور کارگری که مالک بهترین منبع غذایی است، تبدیل به زنبور دیدبان می‌شود.

۴- عملکرد کلی الگوریتم HABC در [61] نمایش داده شده است. [61].

۵-۲-۳- الگوریتم‌های مبتنی بر فیزیک

الگوری‌تم‌های مبتنی بر فی‌زی‌ک از اصول فی‌زی‌کی حاکم بر طبی‌عت مانند نظری‌ه‌های نجوم، قوانی‌ن حرکت، نی‌روی جاذبه و چی‌زهای‌ی از ای‌ن دست الهام گرفته‌اند.

۵-۲-۳-۱- الگوریتم جست‌وجوی گرانشی[۳]

ی‌ک الگوری‌تم مبتنی بر جمعی‌ت است که از قانون جاذبه برای ی‌افتن بهتری‌ن مسی‌ر استفاده می‌کند. در [66] از ی‌ک روش ترکی‌بی مبتنی بر جست‌وجوی گرانشی برای کمی‌نه‌سازی زمان انجام کل و هزی‌نه استفاده شده است.

۵-۲-۳-۲- الگوریتم توازن بار اسمزی[۴]

در [16] یک مدل توازن بار اسمزی (OLB) وظایف را به ماشین‌های مجازی تخصیص می‌دهد، با این هدف که بار نهایی متوازن گردد. سیستم توازن بار کاملا نامتمرکز است و به وسیله عامل‌هایی مانند مورچه پشتیبانی می‌شود که مراکز داده روی سطح Chord هدایت می‌کنند. هر مرکز داده با یک فهرست تعامل کرده و می‌تواند یک یا تعداد بیشتر وظیفه را به طور همزمان با ویژگی‌های پیاده‌سازی مختلف اجرا کند.

۵-۲-۴- الگوریتم‌های ترکیبی

۵-۲-۴-۱- ترکیبی (کلونی مورچه، زنبور عسل با بازخورد پویا)

در [45] یک الگوریتم LB برای استفاده کارا از سرویس‌های رایانش ابری ارائه شده است. متد ارائه شده ترکیبی از کلونی مورچه، زنبور عسل و یک متد متوازن‌سازی مبتنی بر RR و الگوریتم ژنتیک موازی می‌باشد که وظایف را براساس اولویت زمان‌بندی می‌کند. هدف این الگوریتم بهبود نرخ توازن بار برای

---

[۱] Artificial Bee Colony
[۲] Thanka
[۳] Gravitational Search Algorithm
[۴] Osmosis

وظایف دریافتی و تکمیل وظایف در زمانی کوتاه‌تر می‌باشد که در سناریوهای مختلف با RR و یک متد ترکیبی دیگر مقایسه شده و نتایج بهتری به دست آورده است.

۵-۲-۴-۲- ترکیبی (الگوریتم ژنتیک، ازدحام ذرات)

منصرح و علی[۱] یک روش ترکیبی از الگوریتم ژنتیک و الگوریتم ازدحام ذرات GA-PSO برای زمان‌بندی جریان کار در رایانش ابری پیشنهاد کرده‌اند که منجر به توازن بار بیشتر در ابر می‌شود. روش پیشنهادی در دو فاز انجام شده است و در فاز نخست الگوریتم ژنتیک روی کل جمعیت و به تعداد n/2 دفعات مشخص شده اعمال می‌شود تا راه حل بهینه را از میان راه‌حل‌های موجود انتخاب کند که برای حل مسئله زمان‌بندی ضروری است. سپس PSO بر روی تمام جمعیت تولید شده در n/2 دورهای تعیین شده اعمال می‌شود که به وسیله الگوریتم GA تولید شده‌اند.

الگوریتم PSO بهترین و بدترین راه‌حل‌ها را در حافظه نگه می‌دارد، زیرا اینها می‌توانند برای همگرایی سریع راه‌حل‌ها در هنگامی که GA راه حل بد تولید می‌کند، مفید باشند. راه‌حل‌هایی در GA که راه حل زمان‌بندی مسئله ما را تعریف می‌کنند، به وسیله چندین کروموزوم بازنمایی می‌شوند که طول آنها برابر با تعداد کل جریان‌های کاری است. هر کروموزوم از چندین ژن تشکیل شده است که VMهای میزبان را بازنمایی می‌کنند. در هر دور GA کروموزوم‌ها را به سه عملگر انتخاب، آمیزش و جهش می‌فرستد. راه‌حل‌هایی که از الگوریتم GA بازگردانده شده‌اند، با تعداد دورهای باقی مانده به PSO داده می‌شوند تا راه حل بهینه را از میان راه‌حل‌های تولید شده به وسیله GA بیابد. در الگوریتم PSO راه‌حل‌ها Particle نامیده می‌شوند [67].

۵-۲-۴-۳- ترکیبی (الگوریتم شاهین هریس و بهینه‌سازی کبوتر)

آنی و رادامانی[۲] یک الگوریتم ترکیبی مبتنی بر شاهین هریس و بهینه‌سازی کبوتر برای ایجاد یک زمان‌بند متوازن کننده بار با کارایی بالا پیشنهاد کرده‌اند.

الگوریتم پیشنهادی روی بهینه‌سازی زمان پاسخ، میانگین زمان انتظار، زمان اجرا، تاخیر، زمان اجرا و بازدهی در محیط شبیه‌ساز Cloudsim تمرکز کرده است [68].

۶- پیشنهادها

- هدف توازن بار بهینه‌سازی اهدافی است که ممکن است باهم وابستگی، تداخل یا حتی تضاد داشته باشند. استفاده از الگوریتم‌های بهینه‌سازی جدید با چند هدف[۳] می‌تواند گام موثری در بهینه‌سازی بار کاری ابر باشد [71].
- با توجه به تغییرات مداوم بار کاری ابر و پیش‌بینی ناپذیری آن، استفاده از سیاست‌های بلند مدت مانند مدل تصمیم مارکوف[۴] (MDP) یا تابع تعادل لیاپانوف[۵] برای توازن بار کاری در بلند مدت، به جای استفاده از روش‌های بهینه‌سازی کوتاه مدت پیشنهاد می‌شود. برای نمونه، در [72] از مدل تصمیم مارکوف و یادگیری تقویتی برای بهینه‌سازی بارسپاری[۶] در لبه استفاده شده است که با توجه به اینکه بارسپاری را می‌توان نوعی توازن بار در نظر گرفت، می‌تواند برای بهینه‌سازی توازن بار مورد استفاده قرار گیرد. در [73] نیز از تابع لیاپانوف برای به منظور بهینه‌سازی تعادل صف‌های منابع سیستم در بلندمدت استفاده شده که در واقع به نوعی عملیات توازن بار در سیستم انجام شده است، لذا پیشنهاد می‌شود، استفاده از تابع لیاپانوف برای توازن بار مورد بررسی بیشتر قرار گیرد.
- با توجه به پیچیدگی فرایند بهینه‌سازی توازن بار و تعدد پارامترهایی که باید بهینه شوند، استفاده از متدهای یادگیری ماشین مانند یادگیری عمیق، یادگیری تقویتی و یادگیری تقویتی عمیق به همراه الگوریتم‌های فراابتکاری، به صورت ترکیبی می‌تواند مرتبه زمانی فرایند توازن بار را کاهش دهد [74] [75].
- رایانش ابری موبایل[۷] (MCC) و رایانش لبه‌ای موبایل[۸] (MEC) زمینه‌هایی هستند که چالش‌های باز زیادی در زمینه توازن بار، بهبود میزان استفاده از منابع، موازنه جریان کار و مصرف انرژی دارند. بیشتر مطالعات بررسی شده در این زمینه توجه کمتری به قابلیت جابجایی و مخصوصا سرعت جابجایی تجهیزات موبایل و پردازش استریم داشته‌اند. توازن بار و بارسپاری در شبکه‌های

---

[۱] Manasrah and Ali

[۲] G. Annie, A. S. Radhamani

[۳] Multi-objective

[۴] Markov Decision Process

[۵] Lyapunov

[۶] Offloading

[۷] Mobile Cloud Computing

[۸] Mobile Edge Computing

موبایل، اینترنت اشیاء و اینترنت خودروها، جای بحث و مطالعه بیشتر دارند.

- می‌توان تکنیک‌های توازن بار مطالعه شده در این کار را برای محیط مه و لبه و مخصوصا برای بهینه‌سازی بارسپاری، تخصیص منابع و توان مصرفی در محیط لبه گسترش داد که موضوعی باز برای مطالعه و پژوهش بیشتر می‌باشند.

جدول ۲- مقایسه الگوریتم‌های جست‌وجوی فراابتکاری (پاسخ پرسش چهارم پژوهش)

نوع روش استفاده شده	مزایا	معایب
الگوریتم کلونی مورچه [25]، [69]،[57]	• توازن بار بهتر، کارایی بالاتر، کاهش زمان عملیات • تاثیر بالایی دارد و می‌تواند در زمینه‌های مختلف مورد استفاده قرار گیرد. • می‌تواند به سرعت با دیگر الگوریتم‌های ابتکاری باهم مورد استفاده قرار گیرد. • توانایی بالا در جست‌وجوی راه‌حل‌ها	در محیط ابر واقعی آزمایش نشده است.
الگوریتم ژنتیک [46]، [19]	• کارایی بالای سیستم • کاستن زمان وظایف • استفاده بهتر از منابع	• بازدهی کم • نداشتن مقیاس پذیری (قابلیت گسترش)
جست‌وجوی غذای زنبور عسل [70]	• Makespan و زمان پاسخ پایین‌تر • عملکرد خوب با رشد پیچیدگی سیستم	• روی وظایفی که برپایه آن بنا شده‌اند عمل نمی‌کند • با افزایش اندازه ماشین، کارایی بالا نمی‌رود
الگوریتم کلونی زنبور مصنوعی [63]	زمان مهاجرت کمینه	• از تمام منابع استفاده نمی‌کند
ترکیبی (کلونی مورچه، زنبور عسل با بازخورد) [45]	• زمان پاسخ پایین‌تر، استفاده بهتر از منابع، مهاجرت وظایف کمتر	پیچیدگی بالا و مقیاس پذیری کم
کلونی مورچه و شبکه پیچیده [69]	• تخصیص وظایف به گره مربوطه • زمان‌بندی بیشینه کارها	• نداشتن زمان پاسخ مناسب • نداشتن زمان مهاجرت مناسب • نداشتن کارایی مناسب
الگوریتم توازن بار اسمزی [16]	• کارکرد خوب در ماشین‌های مجازی همگون و ناهمگون • مفید در تخصیص دوباره وظایف میان ماشین‌های مجازی متصل	• انجام یک وظیفه در هر لحظه • نامتمرکز
الگوریتم بهینه‌سازی کلونی زنبور [62]	• کمینه‌سازی زمان زمان‌بندی • کاهش حجم داده‌ها	در هنگام کمینه بودن زمان کلی، الگوریتم کارایی دارد.
توازن بار جست‌وجوی غذای زنبور عسل [41]	• افزایش زمان پاسخ برای ماشین‌های مجازی و کاهش زمان صرف شده (زمان کل)	بدون کم کردن عدم توازن بار روی ماشین‌های مجازی کار نمی‌کند
ترکیبی الگوریتم ژنتیک با ازدحام ذرات [67]	• توزیع عادلانه بار • کاهش هزینه پردازش و زمان کلی	• محدود بودن به یک مرکز داده • جریان کاری ایستا

۷- نتیجه‌گیری

در این کار تلاش شده تا مروری بر روش‌های توازن بار در محیط ابر ارائه شود، اما تمرکز اصلی بر روی روش‌های توازن بار متاثر از الگوریتم‌های فراابتکاری بوده است. با توجه به پویایی محیط ابر، افزایش رو به رشد تعداد کاربران ابر و نیاز به بهبود دائم کارایی ابرها و به تبع آن نیاز به توازن بار کارا در محیط ابر، استفاده از تکنیک‌های فراابتکاری می‌تواند کمک قابل توجهی به تسریع فرایند توازن بار بدون دخالت عامل انسانی داشته باشد.

الگوریتم‌های کلونی مورچه، کلونی مورچه مصنوعی، کلونی زنبور، کلونی زنبور مصنوعی، جست‌وجوی غذای زنبور عسل، ازدحام ذرات، ازدحام گربه‌ها، تبرید شبیه‌سازی شده، الگوریتم ژنتیک، جست‌وجوی ممنوعه، الگوریتم دسته ماهی‌ها و الگوریتم‌های ترکیبی برخی از موارد بررسی شده در این کار می‌باشند.

با بررسی تعداد کارهای مطالعاتی انجام شده در بازه زمانی پژوهش، مشاهده می‌شود که سیر کلی مطالعات مربوط به توازن بار در ابر، به سمت روش‌های مبتنی بر ازدحام، الگوریتم‌های ژنتیک می‌باشد و البته ناگفته نماند که تعداد قابل توجهی از مطالعات در زمینه توازن بار از یادگیری عمیق و یادگیری تقویتی و یادگیری تقویتی عمیق برای بارسپاری و توازن بار در ابر بهره برده‌اند که می‌تواند زمینه مناسبی برای ادامه پژوهش در آینده باشد.

در این کار معیارهای مختلفی برای سنجش توازن بار در ابر ارائه شد و کارهای پژوهشی انجام شده براساس این معیارها مورد مقایسه قرار گرفتند و مزایا و معایب هر کدام به طور مختصر در جدولی ارائه شد.

به دلیل تنوع در کاربرد ابرها و روش‌های پیاده‌سازی آنها، تعیین معیار برای سنجش کیفیت خدمات، منجر به یک مسئله بهینه‌سازی با چند هدف می‌شود که راه حل قطعی ندارد و نیاز به توسعه الگوریتم‌های توازن بار جدید یا ترکیب الگوریتم‌های موجود برای مصالحه و تعیین وزن اهداف دارد.

با توجه به مدل‌سازی و پیاده‌سازی روش‌های مختلف روی سیستم‌های مختلف و با استفاده از ابزارهای شبیه‌سازی متفاوت، پیاده‌سازی مستقلی انجام نشد و امکان سنجش میزان صحت ادعای نویسندگان وجود نداشت و ادعای نویسندگان ملاک مقایسه ما قرار گرفت.

در کارهای آینده به بررسی مدل‌های پیاده‌سازی ابر، ابزارهای شبیه‌سازی استاندارد و پیاده‌سازی روش‌های مورد مطالعه در منابع و مقایسه نتایج روش‌های مورد مطالعه، خواهیم پرداخت.

مراجع

[1] P. Kumar and R. Kumar, "Issues and challenges of load balancing techniques in cloud computing: A survey," *ACM Comput. Surv.*, vol. 51, no. 6, 2019, doi: 10.1145/3281010.

[2] A. S. Milani and N. J. Navimipour, "Load balancing mechanisms and techniques in the cloud environments: Systematic literature review and future trends," *Journal of Network and Computer Applications*, vol. 71. Academic Press, pp. 86–98, Aug. 01, 2016, doi: 10.1016/j.jnca.2016.06.003.

[3] R. de Monts *et al.*, "Defined Categories of Security as a Service," *Cloud Secur. Alliance –* , 2016, [Online]. Available: https://downloads.cloudsecurityalliance.org/assets/research/security-as-a-service/csa-categories-securities-prep.pdf.

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

[5] M. Kumar, S. C. Sharma, A. Goel, and S. P. Singh, "A comprehensive survey for scheduling techniques in cloud computing," *J. Netw. Comput. Appl.*, vol. 143, no. June, pp. 1–33, 2019, doi: 10.1016/j.jnca.2019.06.006.

[6] X. Yang and N. Rahmani, "Task scheduling mechanisms in fog computing: review, trends, and perspectives," *Kybernetes*, vol. 50, no. 1, pp. 22–38, 2021, doi: 10.1108/K-10-2019-0666.

[7] R. M. Singh, L. K. Awasthi, and G. Sikka, "Towards Metaheuristic Scheduling Techniques in Cloud and Fog: An Extensive Taxonomic Review," *ACM Comput. Surv.*, vol. 55, no. 3, pp. 1–43, 2023, doi: 10.1145/3494520.

[8] J. K. Konjaang and L. Xu, *Meta-heuristic Approaches for Effective Scheduling in Infrastructure as a Service Cloud: A Systematic Review*, vol. 29, no. 2. Springer US, 2021.

[9] S. Ramasubbareddy, T. Adityasaisrinivas, K. Govinda, S. S. Manivannan, and E. Swetha, "Analysis of load balancing algorithms using cloud analyst," *Int. J. Recent Technol. Eng.*, vol. 7, no. 6, pp. 684–687, 2019.

[10] K. D. Patel and T. M. Bhalodia, "An efficient dynamic load balancing algorithm for virtual machine in cloud computing," in *2019 International Conference on Intelligent Computing and Control Systems, ICCS 2019*, 2019, no. April, pp. 145–150, doi: 10.1109/ICCS45141.2019.9065292.

[11] G. P. P. Geethu and S. K. Vasudevan, "An in-depth analysis and study of Load balancing techniques in the cloud computing environment," *Procedia Comput. Sci.*, vol. 50, pp. 427–432, 2015, doi: 10.1016/j.procs.2015.04.009.

[12] A. R. Kumar, A. O. Salau, S. Gupta, and S. Arora, "A Survey of Machine Learning Methods for IoT and their Future Applications," *Amity J. Comput. Sci.*, vol. 2, no. 2, pp. 1–5, 2018, [Online]. Available: www.amity.edu.in/ajcs.

[13] J. Prassanna and N. Venkataraman, "Threshold Based Multi-Objective Memetic Optimized

[23] C. Wang, C. Feng, and J. Cheng, "Distributed Join-the-Idle-Queue for Low Latency Cloud Services," *IEEE/ACM Trans. Netw.*, vol. 26, no. 5, pp. 2309–2319, 2018, doi: 10.1109/TNET.2018.2869092.

[24] B. Alankar, G. Sharma, H. Kaur, R. Valverde, and V. Chang, "Experimental setup for investigating the efficient load balancing algorithms on virtual cloud," *Sensors (Switzerland)*, vol. 20, no. 24, pp. 1–26, 2020, doi: 10.3390/s20247342.

[25] K. Bhargavi, B. Sathish Babu, and J. Pitt, "Performance Modeling of Load Balancing Techniques in Cloud: Some of the Recent Competitive Swarm Artificial Intelligence-based," *J. Intell. Syst.*, vol. 30, no. 1, pp. 40–58, 2020, doi: 10.1515/jisys-2019-0084.

[26] C. Yaashuwanth *et al.*, "Performance Comparison of Priority Rule Scheduling Algorithms Using Different Inter Arrival Time Jobs in Grid Environment," *Int. J. Grid Distrib. Comput.*, vol. 6, no. 4, pp. 157–168, 2012.

[27] C. R. Anna Victoria Oikawa, V. Freitas, M. Castro, and L. L. Pilla, "Adaptive load balancing based on machine learning for iterative parallel applications," *Proc. - 2020 28th Euromicro Int. Conf. Parallel, Distrib. Network-Based Process. PDP 2020*, pp. 94–101, 2020, doi: 10.1109/PDP50117.2020.00021.

[28] C. Wang, B. Hu, S. Chen, D. Li, and B. Liu, "A Switch Migration-Based Decision-Making Scheme for Balancing Load in SDN," *IEEE Access*, vol. 5, no. c, pp. 4537–4544, 2017, doi: 10.1109/ACCESS.2017.2684188.

[29] S. Lehrig, H. Eikerling, and S. Becker, "Scalability, elasticity, and efficiency in cloud computing: A systematic literature review of definitions and metrics," *QoSA 2015 - Proc. 11th Int. ACM SIGSOFT Conf. Qual. Softw. Archit. Part CompArch 2015*, no. 1, pp. 83–92, 2015, doi: 10.1145/2737182.2737185.

[30] M. A. Shahid, N. Islam, M. M. Alam, M. M. Su'Ud, and S. Musa, "A Comprehensive Study of Load Balancing Approaches in the Cloud Computing Environment and a Novel Fault Tolerance Approach," *IEEE Access*, vol. 8, no. c, pp. 130500–130526, 2020, doi: 10.1109/ACCESS.2020.3009184.

[31] Y. Fahim *et al.*, "Load balancing in cloud computing using meta-heuristic algorithm," *J. Inf. Process. Syst.*, vol. 14, no. 3, pp. 569–589, 2018, doi: 10.3745/JIPS.01.0028.

[32] I. Attiya, M. Abd Elaziz, and S. Xiong, "Job Scheduling in Cloud Computing Using a Modified Harris Hawks Optimization and Simulated Annealing Algorithm," *Comput. Round Robin Scheduling for Resource Efficient Load Balancing in Cloud," *Mob. Networks Appl.*, vol. 24, no. 4, pp. 1214–1225, 2019, doi: 10.1007/s11036-019-01259-x.

[14] M. S. Viana, O. M. Junior, and R. C. Contreras, "A modified genetic algorithm with local search strategies and multi-crossover operator for job shop scheduling problem," *Sensors (Switzerland)*, vol. 20, no. 18, pp. 1–32, 2020, doi: 10.3390/s20185440.

[15] E.-G. Talbi, "Machine learning into metaheuristics: A survey and taxonomy of data-driven meta-heuristics," pp. 1–30, 2020, [Online]. Available: https://hal.inria.fr/hal-02745295.

[16] B. Mallikarjuna and P. Venkata Krishna, "OLB: A nature inspired approach for load balancing in cloud computing," *Cybern. Inf. Technol.*, vol. 15, no. 4, pp. 138–148, 2015, doi: 10.1515/cait-2015-0060.

[17] M. Aruna, D. Bhanu, and S. Karthik, "An improved load balanced metaheuristic scheduling in cloud," *Cluster Comput.*, vol. 22, no. March, pp. 10873–10881, 2019, doi: 10.1007/s10586-017-1213-9.

[18] X. Sui, D. Liu, L. Li, H. Wang, and H. Yang, "Virtual machine scheduling strategy based on machine learning algorithms for load balancing," *Eurasip J. Wirel. Commun. Netw.*, vol. 2019, no. 1, 2019, doi: 10.1186/s13638-019-1454-9.

[19] F. M. Talaat, M. S. Saraya, A. I. Saleh, H. A. Ali, and S. H. Ali, "A load balancing and optimization strategy (LBOS) using reinforcement learning in fog computing environment," *J. Ambient Intell. Humaniz. Comput.*, vol. 11, no. 11, pp. 4951–4966, 2020, doi: 10.1007/s12652-020-01768-8.

[20] M. O. Ahmad and R. Z. Khan, "Load balancing tools and techniques in cloud computing: A systematic review," *Adv. Intell. Syst. Comput.*, vol. 554, no. November 2017, pp. 181–195, 2018, doi: 10.1007/978-981-10-3773-3_18.

[21] G. Ramadhan, T. W. Purboyo, R. Latuconsina, and A. R. Robin, "Experimental Model for Load Balancing in Cloud Computing Using Throttled Algorithm," *Int. J. Appl. Eng. Res.*, vol. 13, no. 2, pp. 1139–1143, 2018, [Online]. Available: https://www.ripublication.com/ijaer18/ijaerv13n2_42.pdf.

[22] K. Dasgupta, B. Mandal, P. Dutta, J. K. Mandal, and S. Dam, "A Genetic Algorithm (GA) based Load Balancing Strategy for Cloud Computing," *Procedia Technol.*, vol. 10, pp. 340–347, 2013, doi: 10.1016/j.protcy.2013.12.369.

in cloud environment," *Smart Innov. Syst. Technol.*, vol. 105, pp. 319–326, 2019, doi: 10.1007/978-981-13-1927-3_34.

[45] M. Ashouraei, S. N. Khezr, R. Benlamri, and N. J. Navimipour, "A New SLA-Aware Load Balancing Method in the Cloud Using an Improved Parallel Task Scheduling Algorithm," *Proc. - 2018 IEEE 6th Int. Conf. Futur. Internet Things Cloud, FiCloud 2018*, pp. 71–76, 2018, doi: 10.1109/FiCloud.2018.00018.

[46] J. Kumar, A. K. Singh, and A. Mohan, "Resource-efficient load-balancing framework for cloud data center networks," *ETRI J.*, vol. 43, no. June 2019, pp. 53–63, 2020, doi: 10.4218/etrij.2019-0294.

[47] J. Lim and D. Lee, "A load balancing algorithm for mobile devices in edge cloud computing environments," *Electron.*, vol. 9, no. 4, pp. 1–13, 2020, doi: 10.3390/electronics9040686.

[48] M. Alam and Z. Ahmad Khan, "Issues and Challenges of Load Balancing Algorithm in Cloud Computing Environment," *Indian J. Sci. Technol.*, vol. 10, no. 25, pp. 1–12, 2017, doi: 10.17485/ijst/2017/v10i25/105688.

[49] K. Mishra, J. Pati, and S. Kumar Majhi, "A dynamic load scheduling in IaaS cloud using binary JAYA algorithm," *J. King Saud Univ. - Comput. Inf. Sci.*, 2020, doi: 10.1016/j.jksuci.2020.12.001.

[50] R. Venkata Rao, "Jaya: A simple and new optimization algorithm for solving constrained and unconstrained optimization problems," *Int. J. Ind. Eng. Comput.*, vol. 7, no. 1, pp. 19–34, 2016, doi: 10.5267/j.ijiec.2015.8.004.

[51] R. Gulbaz, A. B. Siddiqui, N. Anjum, A. A. Alotaibi, T. Althobaiti, and N. Ramzan, "Balancer genetic algorithm-a novel task scheduling optimization approach in cloud computing," *Appl. Sci.*, vol. 11, no. 14, 2021, doi: 10.3390/app11146244.

[52] S. K. Gavvala, C. Jatoth, G. R. Gangadharan, and R. Buyya, "QoS-aware cloud service composition using eagle strategy," *Futur. Gener. Comput. Syst.*, vol. 90, pp. 273–290, Jan. 2019, doi: 10.1016/j.future.2018.07.062.

[53] S. Pattnaik, J. Prakash Mishra, B. Kumar Sahoo, and B. Kumar Pattanayak, "Load Balancing in Cloud Computing Environment Using CloudSim," *Smart Innov. Syst. Technol.*, vol. 194, no. 01, pp. 197–205, 2021, doi: 10.1007/978-981-15-5971-6_22.

[54] M. G. Rani, A. Marimuthu, A. Kavitha, M. G. Rani, A. Marimuthu, and A. Kavitha, "Artificial Fish Swarm Load Balancing and Job Migration Task with Overloading Detection in Cloud Computing Environments," *Intell. Neurosci.*, vol. 2020, 2020, doi: 10.1155/2020/3504642.

[33] N. Djennane, R. Aoudjit, and S. Bouzefrane, "Energy-efficient algorithm for load balancing and VMs reassignment in data centers," *Proc. - 2018 IEEE 6th Int. Conf. Futur. Internet Things Cloud Work. W-FiCloud 2018*, pp. 225–230, 2018, doi: 10.1109/W-FiCloud.2018.00043.

[34] S. Afzal and G. Kavitha, "Load balancing in cloud computing – A hierarchical taxonomical classification," *J. Cloud Comput.*, vol. 8, no. 1, 2019, doi: 10.1186/s13677-019-0146-7.

[35] S. Kumar Mishra, B. Sahoo, and P. Parida, "Load balancing in cloud computing: A big picture q," 2018, doi: 10.1016/j.jksuci.2018.01.003.

[36] P. S. Chauhan and J. Patel, "Load Balancing in Cloud Computing Using Machine Learning Techniques," *JASC J. Appl. Sci. Comput.*, vol. VI, no. Iv, pp. 3533–3541, 2019.

[37] E.-G. Talbi, "Frontmatter Enhanced Reader.pdf." p. 25, 2009.

[38] G. Dhiman and A. Kaur, "Optimizing the design of airfoil and optical buffer problems using spotted hyena optimizer," *Designs*, vol. 2, no. 3, pp. 1–16, 2018, doi: 10.3390/designs2030028.

[39] V. Černý, "Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm," *J. Optim. Theory Appl.*, vol. 45, no. 1, pp. 41–51, 1985, doi: 10.1007/BF00940812.

[40] N. Téllez, M. Jimeno, A. Salazar, and E. D. Nino-Ruiz, "A Tabu search method for load balancing in fog computing," *Int. J. Artif. Intell.*, vol. 16, no. 2, pp. 106–135, 2018.

[41] K. P. Kumar, T. Ragunathan, D. Vasumathi, and P. K. Prasad, "An efficient load balancing technique based on cuckoo search and firefly algorithm in cloud," *Int. J. Intell. Eng. Syst.*, vol. 13, no. 3, pp. 422–432, 2020, doi: 10.22266/IJIES2020.0630.38.

[42] A. A. Alexander and D. L. Joseph, "An Efficient Resource Management for Prioritized Users in Cloud Environment Using Cuckoo Search Algorithm," *Procedia Technol.*, vol. 25, no. Raerest, pp. 341–348, 2016, doi: 10.1016/j.protcy.2016.08.116.

[43] S. H. H. Madni, M. S. A. Latiff, J. Ali, and S. M. Abdulhamid, "Multi-objective-Oriented Cuckoo Search Optimization-Based Resource Scheduling Algorithm for Clouds," *Arab. J. Sci. Eng.*, vol. 44, no. 4, pp. 3585–3602, 2019, doi: 10.1007/s13369-018-3602-7.

[44] S. Basu, G. Kannayaram, S. Ramasubbareddy, and C. Venkatasubbaiah, "Improved genetic algorithm for monitoring of virtual machines

February. 2018.

[66] A. Choudhary, I. Gupta, V. Singh, and P. K. Jana, "A GSA based hybrid algorithm for bi-objective workflow scheduling in cloud computing," *Futur. Gener. Comput. Syst.*, vol. 83, pp. 14–26, 2018, doi: 10.1016/j.future.2018.01.005.

[67] A. M. Manasrah and H. B. Ali, "Workflow Scheduling Using Hybrid GA-PSO Algorithm in Cloud Computing," *Wirel. Commun. Mob. Comput.*, vol. 2018, 2018, doi: 10.1155/2018/1934784.

[68] G. Annie Poornima Princess and A. S. Radhamani, "A Hybrid Meta-Heuristic for Optimal Load Balancing in Cloud Computing," *J. Grid Comput.*, vol. 19, no. 2, 2021, doi: 10.1007/s10723-021-09560-4.

[69] A. Kaur and B. Kaur, "Load balancing optimization based on hybrid Heuristic-Metaheuristic techniques in cloud environment," *J. King Saud Univ. - Comput. Inf. Sci.*, no. xxxx, 2019, doi: 10.1016/j.jksuci.2019.02.010.

[70] T. Davidović, D. Teodorović, and M. Šelmić, "Bee Colony Optimization part I: The algorithm overview," *Yugosl. J. Oper. Res.*, vol. 25, no. 1, pp. 33–56, 2015, doi: 10.2298/YJOR131011017D.

[71] A. Ghasemi and A. Toroghi Haghighat, "A multi-objective load balancing algorithm for virtual machine placement in cloud data centers based on machine learning," *Computing*, vol. 102, no. 9, pp. 2049–2072, 2020, doi: 10.1007/s00607-020-00813-w.

[72] C. Li *et al.*, "Dynamic Offloading for Multiuser Muti-CAP MEC Networks: A Deep Reinforcement Learning Approach," *IEEE Trans. Veh. Technol.*, vol. 70, no. 3, pp. 2922–2927, 2021, doi: 10.1109/TVT.2021.3058995.

[73] X. Li, S. Bi, L. Huang, H. Wang, and Y. J. A. Zhang, "Lyapunov-guided deep reinforcement learning for stable online computation offloading in mobile-edge computing networks," *J. Grid Comput.*, vol. 19, no. 3, pp. 7519–7537, 2021, doi: 10.1109/TWC.2021.3085319.

[74] J. Vijayaraj, D. Loganathan, P. Engineering, and P. Engineering, "An Improved Co-operative Load Balancing Algorithm with Machine learning techniques in Cloud Computing," vol. 11, no. 3, pp. 1927–1935, 2020.

[75] P. Wei *et al.*, "Reinforcement Learning-Empowered Mobile Edge Computing for 6G Edge Intelligence," 2022.

Int. Rev. Comput. Softw., vol. 9, no. 4, pp. 727–734, Apr. 2014, doi: 10.15866/IRECOS.V9I4.848.

[55] H. Krishnaveni and V. S. Janita, "Modified Artificial Fish Swarm Algorithm for Efficient Task Scheduling in Cloud Environment," *Int. J. Comput. Sci. Eng.*, vol. 7, no. 5, pp. 1363–1371, 2019, doi: 10.26438/ijcse/v7i5.13631371.

[56] K. Balaji, P. Sai Kiran, and M. Sunil Kumar, "An energy efficient load balancing on cloud computing using adaptive cat swarm optimization," *Mater. Today Proc.*, Jan. 2021, doi: 10.1016/J.MATPR.2020.11.106.

[57] P. Xu, G. He, Z. Li, and Z. Zhang, "An efficient load balancing algorithm for virtual machine allocation based on ant colony optimization," *Int. J. Distrib. Sens. Networks*, vol. 14, no. 12, 2018, doi: 10.1177/1550147718793799.

[58] T. Agarwal and A. Saxena, "A Review On Load Balancing Algorithm in Cloud Computing Using Restful Web Services," *Int. J. Comput. Sci. Eng.*, vol. 6, no. 7, pp. 704–707, 2018, doi: 10.26438/ijcse/v6i7.704707.

[59] K. Li, G. Xu, G. Zhao, Y. Dong, and D. Wang, "Cloud task scheduling based on load balancing ant colony optimization," *Proc. - 2011 6th Annu. ChinaGrid Conf. ChinaGrid 2011*, pp. 3–9, 2011, doi: 10.1109/ChinaGrid.2011.17.

[60] L. Wang and J. Shen, "Multi-Phase Ant Colony System for Multi-Party Data-Intensive Service Provision," *IEEE Trans. Serv. Comput.*, vol. 9, no. 2, pp. 264–278, 2016, doi: 10.1109/TSC.2014.2358213.

[61] L. Adhianto *et al.*, "HPCTOOLKIT: Tools for performance analysis of optimized parallel programs," *Concurr. Comput. Pract. Exp.*, vol. 22, no. 6, pp. 685–701, 2010, doi: 10.1002/cpe.

[62] B. Kruekaew and W. Kimpan, "Enhancing of artificial bee colony algorithm for virtual machine scheduling and load balancing problem in cloud computing," *Int. J. Comput. Intell. Syst.*, vol. 13, no. 1, pp. 496–510, 2020, doi: 10.2991/ijcis.d.200410.002.

[63] M. R. Thanka, P. Uma Maheswari, and E. B. Edwin, "An improved efficient: Artificial Bee Colony algorithm for security and QoS aware scheduling in cloud computing environment," *Cluster Computing.*, vol. 22, pp. 10905–10913, 2019, doi: 10.1007/s10586-017-1223-7.

[64] M. . B. S. UMA, Mrs. R., "OPTIMIZATION ALGORITHMS IN LOAD BALANCING: A STUDY," vol. XII, no. Iv, pp. 1–22, 2019.

[65] Mohammad Oqail Ahmad and Rafiqul Zaman Khan, *Advances in Computer and Computational Sciences*, vol. 554, no.