Journal of Soft Computing and Information Technology (JSCIT)
Babol Noshirvani University of Technology, Babol, Iran
Journal Homepage: jscit.nit.ac.ir
Volume 12, Number 4, Winter 2023, pp. 39-51
Received: 03/25/2023, Revised: 03/18/2024, Accepted: 05/02/2024

A New Method To Increase Load balance In A Cloud Computing Environment Based On An Improved Genetic Algorithm

Wahab Aminiazar[1*], Rasoul Farahi[2] and Fatemeh Dashti[3]

1*- Department of Computer Engineering and Information Technology, Faculty of Electrical and Computer Engineering, Islamic Azad University Mahabad Branch, Mahabad, Iran.
2- Department of Computer Engineering and Information Technology, Faculty of Electrical and Computer Engineering, Islamic Azad University Mahabad Branch, Mahabad, Iran.,
3- Tabriz Electrictiy Distribution Company, Tabriz, Iran.
;
[1*]Aminiazar@iau-mahabad.ac.ir, [2] Stu.rasoul-farahi@iaut.ac.ir, and [3]Fatemehdashti1995@gmail.com

Corresponding author's address: Wahab Aminiazar, Faculty of Electrical and Computer Engineering, Azad University Mahabad Branch, Mahabad, Iran.

Abstract- With the high potential that cloud computing has for remote data storage and processing, Cloud computing has introduced a new computing technology for storing and processing data. In cloud service environments, virtual machines from multiple organizations are placed on a physical server, which increase the efficiency of virtualization. In such a scalability infrastructure, we have some bottlenecks and congestion in our cloud without managing traffic. So, we need the solutions to balance the load and distribute tasks among other processing servers. We present an advanced solution based on optimized genetic algorithm for scheduling and balancing load in cloud infrastructure in this paper. We add some parameters to the algorithm to check resource status before scheduling. In fact, the proposed technique prevents overload or underload on the servers by optimally assigning tasks to processing servers and the tasks of high-load and congested servers are transferred to another server using live migration of the virtual machine, in order to increase the load balance of the cloud infrastructure At the end, the proposed solution is evaluated through the Cloudsim simulator and by testing the volume of more than a thousand virtual machines on the Plant Lab data. The results of the simulation results show that the proposed solution has been able to violate the service level agreement compared to the AMUT and EQVS methods on average 46%, the energy consumption criterion on average by 18 % and averages number of virtual machine migrations improved by 24%.

Keywords- Cloud computing, Load balance, Task scheduling, Genetic Algorithm.

مجله علمی پژوهشی رایانش نرم و فناوری اطلاعات
دانشگاه صنعتی نوشیروانی بابل
صفحه مجله: jscit.nit.ac.ir
جلد ۱۲، شماره ۴، زمستان ۱۴۰۲ ، صفحه ۳۹-۵۱
دریافت: ۱۴۰۲/۰۱/۰۵، بازنگری: ۱۴۰۲/۱۲/۲۸، پذیرش: ۱۴۰۳/۰۲/۱۲

ارائه یک روش جدید جهت افزایش توازن بار در محیط رایانش ابری مبتنی بر الگوریتم ژنتیک بهبود یافته

وهاب‌امینی‌آذر*[۱]، رسول فرحی[۲]، فاطمه دشتی[۳]
*۱- دانشکده مهندسی برق و کامپیوتر، دانشگاه آزاد اسلامی واحد مهاباد، مهاباد، ایران.
۲- دانشکده مهندسی برق و کامپیوتر، دانشگاه آزاد اسلامی واحد مهاباد، مهاباد، ایران.
۳- شرکت توزیع نیروی برق تبریز، تبریز، ایران.
[1*]Aminiazar@iau-mahabad.ac.ir, [2] Stu.rasoul-farahi@iaut.ac.ir, and [3]Fatemehdashti1995@gmail.com

* نشانی نویسنده مسئول: وهاب امینی آذر، مهاباد، خیابان دانشگاه، دانشگاه آزاد اسلامی واحد مهاباد، دانشکده مهندسی برق و کامپیوتر.

چکیده- با پتانسیل بالایی که رایانش ابری برای ذخیره‌سازی و پردازش داده‌ها، از راه دور دارد، این فناوری نمونه‌ای جدید از محاسبات را ارائه داده است. در محیط‌های ارائه‌دهنده خدمات ابری، ماشین‌های مجازی از چندین سازمان روی همان سرور فیزیکی قرار داده شده‌اند که باعث می‌شود بازده مجازی‌سازی به حداکثر میزان خود برسد. در چنین زیرساخت مقیاس پذیری، بدون فراهم کردن راهکارهائی جهت مدیریت بارهای ترافیکی و پخش آن‌ها بین سایر گره‌ها، می‌تواند سبب ایجاد گلوگاه و ازدحام شود. درنتیجه نیاز به راهکارهایی جهت متوازن نمودن بار و پخش آن در بین سایر گره های پردازشی است. در این مقاله یک راهکار پیشرفته مبتنی بر الگوریتم ژنتیک بهینه یافته به منظور زمان‌بندی و ایجاد توازن بار در زیرساخت ابری ارائه می‌شود، بر همین اساس پارامترهای بررسی منبع به الگوریتم اضافه شده است تا قبل از زمان‌بندی، وضعیت منابع نیز موردبررسی قرار گیرد. درواقع تکنیک پیشنهادی از طریق تخصیص بهینه وظایف به سرورهای پردازشی، از افزایش بیش از حد یا کم باری در سرورها جلوگیری بعمل می‌آورد. همچنین سرورهائی که دارای بار زیاد هستند و بعبارتی دچار ازدحام شده‌اند، با استفاده از مهاجرت ماشین مجازی، وظایف آن‌ها به یک سرور دیگر انتقال داده می‌شوند تا از این طریق زمینه سازی افزایش توازن بار در زیرساخت ابری فراهم شود. در پایان راهکار پیشنهادی از طریق شبیه‌ساز کلودسیم و با آزمایش بر روی حجم‌کار بیش از هزار ماشین مجازی بر روی داده‌ها PlantLab مورد ارزیابی قرار داده شده است. نتایج حاصل از شبیه‌سازی نشان می‌دهد که راهکار پیشنهادی توانسته است که معیار نقض توافق‌نامه سطح سرویس را در مقایسه با روش‌های AMUT و EQVS به طور میانگین به میزان ۴۶ درصد، معیار مصرف انرژی را به طور میانگین به میزان ۱۸ درصد و معیار تعداد مهاجرت‌های ماشین مجازی را به طور میانگین به میزان ۲۴ درصد بهبود دهد.

واژه‌های کلیدی: رایانش ابری، توازن‌بار، زمان‌بندی وظایف، الگوریتم ژنتیک.

۱- مقدمه

زمان‌بندی وظایف[۱] یک موضوع بسیار مهم در محیط رایانش ابری[۲] است [۱] و هدف اصلی آن زمان‌بندی بهینه وظایف برای کاربران و استفاده همزمان از تمام توان سیستم ابری است [۲]. تعادل بار[۳]، کیفیت خدمات، اصل اقتصادی و زمانی، بهره برداری بهینه، قدرت سیستم و به حداقل رساندن زمان اتمام کار با کمترین منابع موثر از دیگر اهداف می‌باشد [۳]. رایانش ابری به مجموعه‌ای از سیستم‌ها و برنامه‌هایی اطلاق می‌شود که از منابع توزیع شده برای انجام یک کار غیرمتمرکز استفاده می‌کنند [۴]. هدف از متعادل‌سازی بار،

توزیع یکنواخت بار کار در سراسر ابر است. تعادل بار استفاده از منابع را بهبود می‌بخشد، زمان پاسخ را کاهش می‌دهد و با مکانیسم توزیع بار از اضافه بار در سیستم جلوگیری می‌کند [۵]. رویکرد متعادل‌سازی بار عبارت است از یافتن یک نقشه‌برداری مناسب از وظایف روی پردازنده‌های موجود در سیستم به گونه‌ای که میزان مساوی از وظایف بر روی هر پردازنده انجام شود [۶ - ۸]. در حال حاضر روش‌های متداول برای زمانبندی و تعادل‌بار در محیط رایانش ابری با استفاده از روش‌های سنتی مانند روش‌های فراابتکاری[۴] مانند الگوریتم کلونی مورچه‌ها[۵] و الگوریتم ژنتیک[۶] انجام می‌شود. روش‌های فوق غالباً بر بهینه‌سازی یک هدف متمرکز هستند و به طور همزمان چندین هدف را برآورد نمی‌کنند. در این میان برخی از الگوریتم‌های فرا ابتکاری نیاز به پارامترهای زیادی جهت تنظیم دارند و سرعت همگرایی آن‌ها پایین است [۳]. روش‌های مختلفی مبتنی بر الگوریتم‌های فراابتکاری برای حل مسئله تعادل‌بار و زمانبندی وجود دارند که هر کدام از آن‌ها کارایی های مختلفی و پیچیدگی‌های متفاوتی دارند که در این مقاله یک الگوریتم ژنتیک بهبودیافته مورد مطالعه و بررسی قرار داده می‌شود. از آنجایی که مسئله تعادل‌بار و زمانبندی در محیط رایانش ابری یک مسئله NP[۷] سخت است. بدین جهت به کارگیری الگوریتم تکاملی چندهدفه در محیط رایانش ابری می‌تواند کارایی آن را به طور گسترده بهبود دهد. این الگوریتم برای غلبه بر عیب اصلی الگوریتم ژنتیک یعنی فقدان جستجوی محلی، ارائه شده است و تفاوت اصلی آن با الگوریتم ژنتیک در عملگر انتخاب می‌باشد. در رویکرد ارائه شده ابتدا سعی می‌شود که بر اساس تابع سودمندی چند هدفه تعریف شده و در نظر گرفتن بردارهای جهش و تقاطع بر اساس بهره‌وری از منابع و انرژی مصرفی قصد داریم با استفاده بهینه از منابع و بهبود بهره‌وری در کاهش نقص توافق نامه سطح سرویس، تعداد مهاجرت ها و مصرف انرژی تأثیر گذار باشیم تا خدمات ابری مورد توجه مشتریان قرار گیرد.

این مقاله از هشت بخش تشکیل شده است. در بخش بعدی، خلاصه‌ای از روش‌های قبلی زمانبندی و متعادل سازی بار در رایانش ابری را بررسی می‌کنیم. در بخش سوم، جزئیات الگوریتم پیشنهادی را ارائه می کنیم. نتایج شبیه سازی و ارزیابی روش پیشنهادی در بخش هفتم شرح داده شده است و در نهایت بخش هشتم به نتیجه‌گیری و کارهای آتی می‌پردازد.

۲- کارهای مرتبط

میرشا و همکاران [۹] یک سازوکار توازن بار با الگوریتم دودویی مبتنی بر جایا پیشنهاد داده‌اند که سیستم را متعادل کرده و یک نگاشت مناسب از وظایف بر روی VM[8]ها 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.

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

در مقاله [۱۰]، الگوریتم‌های مکاشفه‌ای MIN-MIN و MAX -MAX که با مجموعه‌ای از درخواست‌های منتظر اجرا شروع می‌شود، زمان اجرای هر درخواست روی هر منبع در یک ماتریس محاسبه می‌شود. در این ماتریس، برای هر درخواست، منبعی که آن را در کمترین زمان مورد پردازش قرار می‌دهد، انتخاب و به مجموعه راه حل اضافه می‌شود و برای کاهش زمان اجرا، الگوریتم کارها را به ماشین‌های مجازی بررسی شده تخصیص می‌دهد. مزیت این الگوریتم تخصیص وظایف به بهترین ماشین مجازی است که باعث بهبود زمان اجرای وظایف می‌گردد. از معایب آن، توزیع بار نامتعادل بین ماشین‌های مجازی است.

لیم و همکاران [۱۱] یک الگوریتم توازن بار برای ابر لبه‌ای پیشنهاد کرده‌اند که به نحو موثری وظایف تخلیه شده از نقطه کانونی به سرویس‌دهنده نزدیک لبه 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 سرویس دهنده‌های لبه‌ای است.

طلعت و همکاران [۱۲] یک استراتژی توازن‌بار و بهینه‌سازی

(LBOS)پیشنهاد داده‌اند که از متد تخصیص منابع مبتنی بر یادگیری تقویت شده و الگوریتم ژنتیک استفاده می‌کند. LBOS پیوسته ترافیک شبکه را پایش کرده و 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 IOT ساده و کارا در سیستم‌های بلادرنگ در رایانش مه مانند سیستم‌های مراقبت درمان می‌باشد. سیستم مه پیشنهادی از سه لایه: ۱- لایه IoT ۲- لایه مه ۳- لایه ابر ماموریت تشکیل شده است. 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 [۱۳] نیز از تابع لیاپانوف برای به منظور بهینه‌سازی تعادل صف‌های منابع سیستم در بلندمدت استفاده شده که در واقع به نوعی عملیات توازن‌بار در سیستم انجام شده است، لذا پیشنهاد می‌شود استفاده از تابع لیاپانوف برای توازن بار مورد بررسی بیشتر قرار گیرد.

در [۱۴] الگوریتم جست وجوی فاخته[۹] به همراه الگوریتم کرم شب تاب[۱۰] (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 روشی بر توازن بار جهت بهینه نمودن مصرف انرژی شبکه با استفاده از الگوی توان فرکانسی تمرکز شده است. همچنین الگوریتم تعدیل بار و مدیریت انرژی توان را در شرایطی که حداکثر حجم‌کاری در مرکز داده برای سرویس‌دهی موجود است ارائه شده است. این الگوریتم در بازه‌های زمانی معینی با مانیتور کردن میزان بهره‌وری میزبان‌های سرویس‌دهنده از میزبان‌هایی که بار اضافی دارند کاسته و بار آن‌ها را به میزبان‌های با بار کم انتقال می‌دهد. در این تحقیق جهت ارزیابی نتایج از شبیه‌ساز Cloudsim و Matlab استفاده شده است. همچنین به موضوع مدیریت منابع مجازی سازی شده مراکز داده رایانش ابر از منظر تعدیل بار رسیده به مرکز بین منابع مختلف موجود روش‌های مدیریت و بهبود مصرف توان الکتریکی کلی مرکز داده رایانش ابر و ارائه سرویس در آن پرداخته شده است [۱۵].

در روشی با استفاده از تغییر در تابع هدف مبتنی بر خوشه‌بندی رویکردی برای برقراری توازن بار و افزایش طول عمر شبکه ارائه شده است. در این روش دو رتبه برای هر گره محاسبه شده است که رتبه اول مشخص کننده خوشه ها و سرخوشه‌ها است و رتبه دوم جهت انتخاب والد هر سرخوشه به منظور ارسال بسته‌ها به سمت مقصد است متریک‌های استفاده شده در محاسبه رتبه‌ها شامل تعداد انتقال مورد انتظار، تعداد گام انرژی باقیمانده و تعداد فرزندان است جهت بررسی عملکرد روش پیشنهادی این روش با شبیه ساز Cooja 03 و در شبکه‌ای با توپولوژی تصادفی با چگالی‌های کم و متوسط ۲۰، ۴۰ و ۶۰ گره مورد شبیه‌سازی قرار گرفته است. در نهایت هدف روش پیشنهادی برقراری توازن‌بار، کاهش مصرف انرژی و افزایش طول عمر شبکه است [۱۶].

در روشی با طراحی یک مدل چند هدفه و در نظر گرفتن حداکثر زمان اجرای وظایف بر روی ماشین‌ها و در کنار آن میانگین سودمندی ماشین‌ها یک راهکار جدید ارائه شده است که توسط الگوریتم‌های تکاملی چند هدفه برای یافتن پاسخ بهینه بررسی و شبیه سازی شده است. در روش ارائه شده، دو سری الگوریتم تکاملی تک هدفه و چندهدفه برای بررسی تاثیر طراحی دو هدف سودمندی و حداکثر زمان آزمایش شده‌اند. در آزمایش‌های صورت گرفته هر دو هدف به طور جداگانه یا با ترکیب خطی لحاظ شده‌اند تا بهتر بتوان به کارآمدی الگوریتم‌های تکاملی و مدل طراحی شده پرداخت. در بررسی‌های بخش نتایج این مقاله مشخص شد که تعریف دو هدف و استفاده از الگوریتم چندهدفه باعث بهبود چشم‌گیر سودمندی و البته شده کاهش حداکثر زمان انجام وظایف شده است. به طوری که استفاده از چند هدف در مدل‌های بررسی حداقل باعث ۵ درصد افزایش سودمندی با تابع هدف یکسان و بیش از ۱۰ درصد توابع هدف متفاوت شده است. همچنین این افزایش سودمندی به هیچ وجه باعث افزایش زمان کل اجرای سیستم در مقایسه با بهترین روش‌های متداول نشده است [۱۷].

در روشی دیگر یک رویکرد تکرار و مهاجرت داده به صورت پویا برای محیط ابری ارائه شده است. ایده اصلی روش پیشنهادی رسیدگی همزمان به مسائل مربوط به انتخاب، زمان محل و تعداد تکرار با توجه به مجموعه‌ای کامل از پارامترهای متعدد می‌باشد با در نظر گرفتن یک بازه زمانی ثابت و تقسیم زمان کل ورود درخواست‌ها به ابر به این بازه ها از پارامترهای متعددی نظیر تعداد تکرارهای فایل و میانگین زمان پاسخ گویی و واریانس تعداد وظیفه‌ها در بازه زمانی، گذشته در روش تعیین حد آستانه بهینه استفاده شده است. همچنین در روش تعیین مکان تکرار از پارامترهایی مانند واریانس تعداد وظیفه‌های درون سرور و فاصله سرور تا وظیفه‌هایی که به یک فایل نیاز دارند استفاده شده است. به علاوه، مسئله کمترین فاصله بین دو سرور با استفاده از الگوریتم دایجسترا به یک مسئله گراف تبدیل شده است. در روش انتخاب تکرار و تعیین تعداد تکرار نیز از پارامترهای مختلف و جدیدی استفاده شده است. پارامترهایی که مقدار آن‌ها از بازه زمانی گذشته محاسبه می‌شوند برای پیش‌بینی وضعیت آن‌ها در بازه زمانی آینده می‌باشند. مجموع ارزیابی انجام شده نشان می‌دهد که این روش در مقایسه با روش‌های، قبل در کمترین مقدار بیش از ۲۵ درصد بهبود زمان پاسخگویی و بیش از ۹ درصد بهبود توازن بار در سرورها را داشته است. [۱۸].

در روشی دیگر با توجه به بار کاری پویا در محیط رایانش ابری حد آستانه‌ی تطبیقی برای میزان بهره وری میزبان‌های فیزیکی در نظر گرفته شده است که بر اساس تاریخچه‌ی سیستم و روش‌های آماری تنظیم می‌شود، در روش پیشنهادی برای هر میزبان دو عامل هوشمند اختصاص داده شده است که به صورت موازی با عامل‌های میزبان‌های دیگر کار می‌کنند و حالت میزبان فیزیکی که روی آن مستقر هستند را تشخیص داده و ماشین‌های مجازی مناسب را برای مهاجرت انتخاب کرده و از میزبان جدا می‌کنند. در نهایت با خاموش کردن به موقع منابع فیزیکی کم بار و جلوگیری از سرریز شدن منابع به کاهش مصرف انرژی اقدام شده است؛ محققین برای تحقق این هدف دو رویکرد در نظر گرفته اند؛ در رویکرد اول از الگوریتم بهینه‌سازی ازدحام ذرات (PSO)[11] برای تخصیص اولیه میزبان‌های فیزیکی و مهاجرت ماشین‌های مجازی بهره برده و تابع برازندگی آن را بر پایه ی انرژی مصرفی منابع طراحی کرده‌اند در رویکرد دوم به جای الگوریتم بهینه‌سازی ازدحام ذرات از الگوریتم کلاسیک PABFD برای تخصیص میزبان‌های فیزیکی و مهاجرت ماشین‌های مجازی استفاده کرده‌اند بررسی نتایج حاصل از شبیه‌سازی کاهش مصرف انرژی در رویکرد اول را نسبت به رویکرد دوم نشان می‌دهد و در کل هر دو رویکرد نسبت به الگوریتم‌های پیشین میزان مصرف انرژی را به اندازه قابل قبولی کاهش داده و از نقض شدن قانون سطح سرویس از قبیل هزینه سرویس، کیفیت سرویس مورد انتظار و غیره جلوگیری کرده‌اند [۱۹].

۳- روش پیشنهادی

هدف اصلی از ایجاد توازن، یافتن نگاشتی مناسب از کارها بر روی پردازنده‌های موجود در سیستم است به طوری که در هر پردازنده مقدار تقریباً مساوی از کارها اجرا گردد تا زمان اجرای کلی به کمترین مقدار خود برسد. برای این منظور در رایانش ابری از متوازن کننده بار استفاده می‌شود. مهم‌ترین وظیفه این مولفه، دریافت کارهای کاربران و توزیع آن‌ها بر روی سرورها و میزبان‌های متفاوت می‌باشد، به گونه‌ای که دسترسی به داده‌ها تا حد امکان محلی شده و محاسبات نیز بصورت متوازن بر روی سرورها قرار گیرند. این عمل زمان پاسخ را حداقل، بهره‌وری منابع را افزایش و کارائی سیستم را بالا می‌برد. معمولاً متوازن‌کننده بار بصورت نرم‌افزاری بوده و در آن اطلاعات مربوط به هر کدام از منابع و سرورها موجود می‌باشد و تعیین می‌کند که درخواست مورد نظر به کدام سرور منتقل و اجرا شود [۲۰]. جهت تعادل بار معیارهای مختلفی از جمله میزان سربار، توان عملیاتی، کارایی، بهره‌برداری از منابع، زمان پاسخ، تحمل‌پذیری خطا و زمان مهاجرت وجود دارد که هر الگوریتم بسته به نوع آن، نسبت به بهبود این معیارها اقدام می‌نماید. در ادامه مراحل روش پیشنهادی به تفکیک شرح داده می‌شود.

۴- مدل کردن ماشین مجازی

با توجه به آنکه تمامی درخواست‌های مربوط به سرویس‌ها نهایتاً در ماشین‌های مجازی اجرا می‌شود، لازم است یک مدل بهینه جهت جایابی و برقراری توازن در ماشین‌مجازی ارایه شود [۲۱]. برای این منظور مجموعه کلیه ماشین‌های فیزیکی بصورت $P=\{p1,p2,\dots pn\}$ نشان داده می‌شود که N تعداد ماشین‌های فیزیکی و $p_i(1\le i\le N)$ ماشین‌فیزیکی i است. مجموعه VMهای ماشین فیزیکی P_i بصورت $V_i=\{V_{i1},V_{i2},V_{i3},\dots V_{imi}\}$ نشان داده می‌شود که در آن m_i تعداد VMها بر روی ماشین فیزیکی i می‌باشد. همچنین از $S=\{S_1,S_2,\dots S_N\}$ برای نمایش مجموعه راه‌حل‌های نگاشت ماشین‌مجازی V بعد از اینکه V بر روی یک ماشین‌فیزیکی قرار گرفت، استفاده می‌شود. در واقع منظور از S_i ، راه حل نگاشت ماشین‌مجازی V بر روی ماشین‌فیزیکی P_i می‌باشد.

۵- پردازش جریان کاری

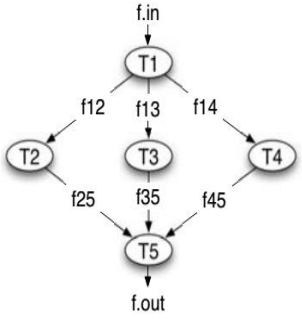
جریان کاری یک برنامه بوسیله یک گراف غیرچرخشی جهت‌دار بصورت $G=(V,E)$ نشان داده می‌شود که $V=\{T_1,T_2,\dots,T_n\}$

مجموعه وظایف و E بیانگر وابستگی های داده ای بین این وظایف می‌باشد [۲۲]. همچنین $f_{j,k}= (Tj, Tk) \in E$ داده فراهم شده توسط T_i و استفاده شده بوسیله T_k را نشان می‌دهد. با در نظرگرفتن اینکه مجموعه‌ای از مراکز ذخیره‌سازی $S=\{1,\ldots,i\}$، مجموعه‌ای از مراکز پردازشی و همچنین مجموعه‌ای از وظایف $T=\{1,\ldots,k\}$ وجود داشته باشد. فرض می‌گردد، میانگین زمان محاسباتی وظیفه T_k بر روی منبع پردازشی PC_i برای یک میزان مشخص ورودی مشخص است و هزینه محاسباتی یک وظیفه بر روی یک میزبان پردازشی نسبت معکوس دارد با زمانی که جهت محاسبات در آن منبع نیاز است. همچنین فرض می‌شود هزینه دسترسی به واحد داده $d_{i,j}$ از منبع i به منبع j نیز از قبل مشخص است. هزینه دسترسی بوسیله ارائه دهنده سرویس به شکل ثابت درآمده است. این هزینه انتقال می‌تواند بر اساس پهنای باند بین مراکز محاسبه شود. ضمناً هزینه انتقال واحد داده بین مراکز بر حسب ثانیه درنظر گرفته شده است. و همچنین فرض می‌شود که این هزینه‌ها غیر منفی و متقارن هستند، بطوریکه روابط زیر در آنها صدق می‌کند:

$$j \in N \;\; i, \;\; d_{i,j}=d_{j,i} \qquad (۱)$$

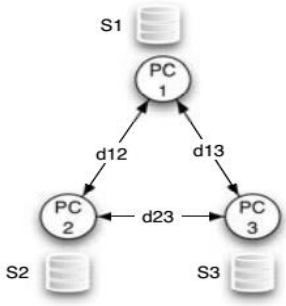
$$i, j, k \in N \;\; d_{i,j} + d_{j,k} \geq d_{i,k} \qquad (۲)$$

بعنوان نمونه، در شکل (۱) ساختار یک جریان کاری با ۵ وظیفه در قالب گره‌های یک درخت نشان داده شده است. وابستگی بین گره‌ها (وظایف) از طریق یال‌ها نشان داده شده‌اند. گره (وظیفه) ریشه، ممکن است یک فایل ورودی داشته باشد و گره پایانی ممکن است فایل نهایی را تولید کند. هر گره داده خروجی را بعد از اینکه به پایان رسید تولید می‌کند ($f_{12},f_{13},\ldots.f_{45}$). این داده‌ها بوسیله فرزندان گره استفاده می‌شوند. مقادیر عددی برای این داده‌ها وزن لبه ($e_{k1,k2}$) بین دو گره $k1 \in T$ و $k2 \in T$ است.



شکل ۱: یک نمونه جریان کاری [۲۰]

همچنین شکل (۲) سه منبع پردازش (PC1,PC2,PC3) متصل بوسیله پهنای‌باند مختلف و دارای واحد ذخیره‌سازی (S1,S2,S3) را نشان می‌دهد. در اینجا هدف تخصیص منابع پردازش به وظایف جریان‌کاری بصورتی است که هزینه پردازش به حداقل رسد، یا به عبارت دیگر: یافتن یک نمونه نگاشت وظیفه - منبع M بطوریکه هنگام تخمین هزینه کل استفاده هر منبع پردازش PC_j، حداکثر هزینه بین کلیه منابع پردازشی، کمترین مقدار ممکن باشد.



شکل ۲: گره‌های پردازشی (PC) و فضای ذخیره‌سازی (S) [۲۰]

بر همین‌اساس فرض می‌شود، $C_{exe}(M)_j$ مجموع هزینه کل وظایف واگذار شده به منبع پردازشی PC_j در رابطه (۳) است. این مقدار توسط مجموع وزن‌های کلیه وظایف واگذار شده به هر منبع در نگاشت M بدست می‌آید. $C_{tx}(M)_j$ نیز هزینه دسترسی کل بین وظایف واگذار شده به منبع پردازش PC_j وظایفی که به آن منبع در نگاشت M واگذار نشده‌اند، در رابطه (۴) نظر گرفته می‌شود. این مقدار برابر است با؛ اندازه فایل خروجی یک وظیفه $k_1 \in k$ به وظیفه $k_2 \in k$ و هزینه انتقال از منبعی که k_1 نگاشت شده است $M(k_1)$ به منبعی که k_2 نگاشت شده است $M(k_2)$، میانگین هزینه انتقال واحد داده بین دو منبع داده شده $dM(k_1),M(k_2)$ است. هزینه انتقال فقط هنگامیکه وابستگی داده‌ای بین وظایف وجود داشته باشد؛ برای دو یا چند وظیفه اجرایی بر روی یک منبع یکسان، صفر است.

$$C_{exe}(M)_j = \sum_k \omega_{jj} \qquad \forall M(k) = j \qquad (۳)$$

$$C_{tx}(M)_j = \sum_{k1 \in T} \sum_{k2 \in T} d_{M(k1),M(k2)}$$

$$\forall M(k) = j \; and \; M(k2) \neq j \qquad (۴)$$

$$C_{total}(M)_j = C_{exe}(M)_j + C_{tx}(M)_j \qquad (۵)$$

$$Cost(M) = \max(C_{total}(M)_j) \;\; \forall j \in P \qquad (۶)$$

$$Minimize(Cost(M) \;\; \forall M) \qquad (۷)$$

رابطه (۶) در واقع تضمین می‌کند که همه وظایف به یک منبع پردازشی واگذار نمی‌شوند. به حداکثر رسانیدن هزینه اولیه، باعث توزیع وظایف بین همه منابع می‌شود. رابطه (۷) باعث می‌شود که هزینه نهایی حداقل شود تا هزینه کل حتی بعد از توزیع اولیه نیز کمینه باشد. به منظور یک واگذاری داده شده M، هزینه کل

$C_{total}(M)_j$ برای یک منبع پردازشی PC_j مجموع هزینه اجرا و هزینه دسترسی خواهد بود (رابطه ۵). به عبارت دیگر با استفاده از رابطه (۷)، هنگام برآورد هزینه کل برای تمامی منابع، بیشینه هزینه برای منابع به حداقل رسیده است.

۶- کاربرد الگوریتم ژنتیک در روش پیشنهادی

درواقع در اکثر الگوریتم‌های پیشین در زمینه تعادل بار، فقط پارامترهایی از قبیل زمان اجرای وظایف، هزینه انتقال وظیفه از کاربر به ارائه‌دهنده سرویس و یا هزینه اجرای وظیفه، بهبود داده شده است. هر چند این الگوریتم‌ها سودمندی منابع ابری مانند کاهش مصرف انرژی را افزایش داده است، با این حال ممکن است در هر یک از این راهکارها یک یا چند پارامتر،کارائی بسیار پایینی داشته باشد. در این مقاله با استفاده از الگوریتم ژنتیک که یک روش فرا اکتشافی است، الگوریتمی کارا ارائه می‌شود که با توجه به وضعیت جاری سیستم و همچنین پس از محاسبه منابع موردنیاز برای ماشین‌های مجازی، بهترین ماشین جهت اجرای وظایف انتخاب شود که علاوه بر ایجاد تعادل بار نیاز به مهاجرت از یک سرور به سرور دیگر نیز حداقل شود و درنتیجه بهره‌وری سیستم بالا رود. همچنین در صورت نیاز از تکنیک مهاجرت زنده ماشین مجازی به منظور انتقال بار از یک سرور پربار به سروری کم بار، استفاده می‌شود. درواقع هدف اصلی این راهکار، ایجاد توازن بار و درنتیجه بهینه‌سازی پارامترهایی مانند تعداد مهاجرت، توافقنامه سطح سرویس و مصرف انرژی می باشد.

۷- فرموله‌سازی راهکار

۷-۱- تعیین ساختار کروموزوم

قدم اول در حل مسئله زمانبندی و توازن به کمک الگوریتم ژنتیک، تعیین ساختار و رمزگذاری کروموزم[۱۲] است. همانطور که می‌دانیم، هر کروموزم در الگوریتم ژنتیک، معادل یک وضعیت از حالات ممکن برای فضای حالت مسئله است. روش‌های زیادی برای رمزگذاری کروموزم‌ها وجود دارد که با توجه به ماهیت مسئله انتخاب می‌شوند. مسئله جایابی بهینه اشیاء براساس معیارهای سازگار، شبیه پیداکردن وزن‌ها در شبکه عصبی است اما با این تفاوت که در این راهکار از الگوریتم ژنتیک بجای شبکه‌های عصبی استفاده می‌شود. بر همین اساس از کدگذاری مقدار استفاده می‌کنیم. در این روش هر کروموزم از تعدادی مقدار (از هر نوع) تشکیل شده است. مقادیر ما وزن‌های W_1 و W_2 خواهند بود، که بر اساس پارامترهای زیر محاسبه می‌شود:.

۷-۲- بررسی سرورها

از آنجاکه منابع اصلی سرورهای پردازشی از جمله پردازنده نباید هیچ‌وقت به حداکثر ظرفیت خود برسند در نتیجه نیاز به یکسری راهکار است که از ایجاد سربار در منابع پردازشی جلوگیری به عمل آید [۲۲]، بر همین اساس به منظور مدل کردن میزان مصرف منابع از راهکار ارایه شده در [۲۱] استفاده می‌شود. این مدل بر این اساس ایجاد شده که بهره‌وری پردازنده دستگاه‌ها در واقع یک رابطه خطی با میزان مصرف انرژی دارد. بعبارتی دیگر برای یک وظیفه بخصوص اطلاعات مربوط به زمان پردازش آن و همچنین میزان بهره‌وری پردازنده؛ جهت محاسبه میزان مصرف انرژی برای آن وظیفه کافی است.

برای منبعی مانند r_i در هر زمان داده شده، میزان بهره‌وری بصورت زیر تعریف می‌شود:

$$U_i = \sum_{j=1}^{n} u_{i,j} \qquad (۸)$$

که در رابطه فوق، n تعداد وظایفی است که در آن لحظه در حال اجراست و $u_{i,j}$ میزان منبع استفاده شده توسط وظیفه t_j می‌باشد. بر همین اساس میزان مصرف انرژی (E_j) منبع r_j در هر لحظه بصورت زیر محاسبه می‌شود [۲۳]:

$$E_j = (P_{max} - P_{min}) \times U_i + P_{min} \qquad (۹)$$

که در این رابطه p_{max} میزان مصرف انرژی در حداکثر بارکاری می باشد (یعنی در حالت بهره‌وری ۱۰۰درصد) و همچنین p_{min} حداقل میزان مصرف انرژی در حالت فعال بودن سرور مورد نظر می‌باشد (یا بعبارتی در میزان بهره‌وری ۱ درصد).

۷-۳- بار ماشین فیزیکی

بار یک ماشین فیزیکی معمولاً از طریق جمع بار ماشین‌های مجازی در حال اجرا بر روی ماشین‌فیزیکی بدست می‌آید. فرض می‌کنیم بهترین محدوده زمانی ثبت شده با استفاده از داده‌های گذشته T است. محدوده زمانی T توسط مانیتور کردن تاریخچه داده‌های گذشته بدست می‌آید. بر اساس قانون تغییر بار ماشین فیزیکی می‌توان زمان T را به n محدوده زمانی تقسیم کنیم. به این ترتیب T به صورت زیر تعریف می‌شود [۲۳]:

$$T=[(t_1-t_0), (t_2-t_1), (t_3-t_2),\ldots.(t_n-t_{n-1})] \qquad (۱۰)$$

فرض شود بار VMها[۶] در هر دوره زمانی ثابت باشد به این ترتیب می‌توان بار ماشین‌مجازی شماره i در محدوده زمانی k را به صورت V(i,k) تعریف کرد. در نتیجه می‌توان نتیجه گرفت در سیکل T بار میانگین ماشین‌مجازی، V_i در ماشین P_i برابر فرمول زیر است:

$$\overline{V_i(i,T)}=\frac{1}{T}\sum_{k=1}^{n} V(i,k) * (t_k - t_{k-1}) \quad (۱۱)$$

با توجه به ساختار سیستم، بار یک ماشین فیزیکی می‌تواند بوسیله جمع بارهای ماشین‌مجازی در حال اجرا بدست آید. و از آنجا می‌توان نتیجه گرفت، بار ماشین فیزیکی P_i برابر با فرمول زیر است. با توجه به دو پارامتر تعریف شده فوق، تابع هزینه جهت مدل کردن مصرف منابع و باند شبکه به صورت زیر می‌باشد:

W_1= Minimize

$\sum_{k=1}^{n} E_j$

W_2= Minimize $\sum_{k=1}^{n} V_i$

بنابراین کروموزم i به صورت زیر نمایش می‌دهیم:

$$C_i = (w_{i1}, w_{i2}) , i = 1, 2, \dots, pop^{(s)}$$

۷-۴- جمعیت s[13]

جمعیت را با pop و اندازه جمعیت را با $pop^{(s)}$ نمایش می‌دهیم. اعضای تشکیل دهنده جمعیت در این مسئله، C_iها ($\forall i$=1, 2 , …,) $pop^{(s)}$ هستند. برای ایجاد جمعیت اولیه از اختصاص اعداد تصادفی اعشاری در بازه [1 .. 0] بدون در نظرگیری معیار خاصی به C_iها استفاده می‌کنیم.

۷-۵- تابع ارزیابی[۱۴]

در ادامه تابعی ایجاد می‌کنیم که بتوانیم توسط آن به هر C_i، عددی متناسب با میزان خوب بودن آن، اختصاص دهیم. این تابع را Func می‌نامیم که مقدار بازگشتی آن برای کروموزم C_i، برابر f_i است. هرچه مقدار f_i بیشتر باشد، احتمال مشارکت C_i برای تولید نسل بعدی افزایش می‌یابد. با استفاده از تابع ارزیابی، برای هر کروموزم C_i، ارزش V_{ij} محاسبه می‌شود. سپس بر اساس این ارزش برای همه اشیاء یک جایابی بهینه ایجاد می‌شود.

$$VmCost(U_j) = CTotal(M)_j + E_j$$

$$Func = Max\left(VmCost(U_j)\right) \quad (۱۲)$$

۷-۶- عملگرهای الگوریتم ژنتیک برای مسئله جایابی

۷-۶-۱- انتخاب[۱۵]

برای اعمال عملگرهای ژنتیک برروی راه‌حل‌ها (C_iها)، باید ابتدا C_iهایی به عنوان والد انتخاب شوند تا از روی این راه‌حل‌ها، راه‌حل‌های فرزند به وجود آیند. برای انتخاب راه‌حل‌های والد در الگوریتم ژنتیک انتخاب‌شده از روش انتخاب مسابقه‌ای استفاده می‌شود [۲۴]. این روش به دلیل کارایی و سادگی پیاده‌سازی، یکی از پرکاربردترین روش‌های انتخابی در الگوریتم ژنتیک است. در این روش n نمونه از یک مجموعه بزرگ بصورت تصادفی انتخاب شده و نمونه‌های انتخاب شده با یکدیگر مقایسه می‌شوند؛ سپس نمونه‌ای که دارای بالاترین میزان شایستگی است برنده شده و در ایجاد نسل بعدی مشارکت داده می شود. تعداد نمونه هایی که در هر بار رقابت شرکت داده می‌شوند بستگی به اندازه مسابقه دارد و معمولا مقدار (۲) فرض شده که به آن مسابقه باینری نیز می‌گویند.

با توجه به اینکه تنوع گرایی می تواند سرعت همگرایی را کاهش دهد این روش در واقع تنوع گراست و این روش این شانس را به همه نمونه‌ها می‌دهد که انتخاب شوند. است. در شکل (۳) شبه کد این الگوریتم نشان داده شده است.

Input: *Size of tournament population, SIZE and population, Pop*
Output: *an individual chromosome*

```
1.  procedure SELECTION
2.  bestScore ←0
3.  bestInd ← null
4.  for i = 1 to SIZE  do
5.  randomly select an individual chromosome
    from Pop, newInd
6.  calculate its fitness score, newScore
7.  if bestScore ≤ newScore then
8.  bestScore ← newScore
9.  bestInd ← newInd
10. end if
11. end for
12. return bestInd
13. end procedur
```

شکل ۳: شبه کد عملگر انتخاب روش پیشنهادی

۷-۶-۲- ترکیب[۱۶]

ترکیب با تعویض ژن‌ها، بین دو کروموزم انجام می‌گیرد و هر کروموزم خصوصیاتی از خود را به فرزندان می‌دهد. تعداد انتخاب کروموزم‌ها از pop به عنوان والد طبق پارامتر احتمال ترکیب Pc انجام می‌گیرد. کروموزم‌هایی که دارای f_i بیشتری هستند، شانس بیشتری برای آمیزش و ترکیب دارند. در این قسمت از ترکیب حسابی استفاده می‌کنیم. اگر از نسل فعلی، دو کروموزم C_i و C_j به عنوان والد انتخاب شوند، طبق آمیزش حسابی از آن‌ها دو فرزند ch_1 و ch_2 به صورت معادله‌های (۱۳) و (۱۴) بوجود می‌آید.

$$ch_1 = \alpha\, C_i + (1 - \alpha) C_j \quad (۱۳)$$

$$ch_2 = \alpha\, C_j + (1-\alpha) C_i \qquad (۱۴)$$

پارامتر α یک مقدار در بازه [1.. 0] می‌باشد که در هر ترکیب می‌تواند مقدار مختلفی داشته باشد. فرزندان بدست‌آمده از این بخش را در جمعیت pop2 قرار می‌دهیم. در شکل (۴) شبه کد این عملگر نشان داده شده است.

```
Input: Two parent chromosomes: ch1 and ch2
Output: One offspring chromosome: Cnew
    1: procedure CROSSOVER
    2:     for i = 1 to |v|        do
    3:         randomly create a value
    between 0 and 1, r;
    4:         if r ≤ Arithmetic Rate then
    5:             set gene i of Cnew as gene
    i of ch1
    6:         else
    7:             set gene i of Cnew as gene
    i of ch2
    8:         end if
    9:     end for
10: end procedure
```

شکل ۴: شبه کد عملگر ترکیب روش پیشنهادی

۷-۶-۳- جهش[11]

مهمترین وظیفه جهش اجتناب از همگرایی به بهینه محلی است. در این قسمت تعدادی از فرزندان بدست‌آمده از مرحله ترکیب، بر طبق پارامتر احتمال جهش Pm انتخاب شده و جهش می‌یابند. با توجه به آنکه در کدگذاری کروموزوم‌ها از روش مقدار اعشاری استفاده می‌شود، پس برای جهش باید از روش جهش حقیقی استفاده کنیم. اگر C_i با ژن‌های w_{i1} و w_{i2} تحت عمل جهش حقیقی انجام گیرد، آنگاه یکی از ژن‌ها (یکی از ژن‌های w_{i1} و w_{i2} به صورت تصادفی انتخاب می‌شود) توسط یک عدد اعشاری تصادفی با توزیع یکنواخت پیوسته در محدوده [1 .. 0] جایگزین می‌شود. C_iهای جهش‌یافته در pop2 جایگزین می‌شوند. در شکل (۵) شبه کد این عملگر نشان داده شده است. در ادامه به ترتیب در شکل (۶) و شکل (۷) فلوچارت روش پیشنهادی شبکه کد چارچوب پیشنهادی نشان داده شده است.

۷-۶-۴- ساختن نسل بعدی

در این مرحله باید از جواب‌های بدست‌آمده از مرحله ترکیب و جهش، جواب‌های برتر را به عنوان والدهای برتر نسل بعدی انتخاب کنیم و بقیه را دور بریزیم. برای این کار از روش انتخاب حالت پایدار استفاده می‌شود. به طوری که با توجه به پارامتر احتمال جایگزینی Rs بدترین اعضای (کمترین f_i) جمعیت کنونی pop توسط بهترین فرزندان (بیشترین f_i) موجود در جمیت pop2 جایگزین می‌شوند.

$$\mathrm{Pm} = \frac{1}{pop^{(s)}} \qquad (۱۵)$$

- **اندازه جمعیت:** اندازه جمعیت خیلی بزرگ معمولاً برخلاف انتظار کارایی الگوریتم ژنتیک را بهبود نمی‌دهد. در بسیاری از *تحقیق‌ها نشان داده شده است که بهترین سایز جمعیت به* سایز رشته‌های رمزگذاری‌شده (کروموزم) بستگی دارد. با توجه به اینکه روش رمزگذاری کروموزم‌های ما به صورت حقیقی، بهترین سایز جمعیت ۲۰ الی ۳۰ کروموزم گزارش شده است [۲۴].
- **درصد جایگزینی:** این پارامتر احتمال جایگزینی اعضای با کمترین f_i جمعیت کنونی pop، توسط فرزندان دارای بیشترین f_i موجود در جمیت pop2 بیان می‌کند. این پارامترا به صورت Rs بیان می‌کنیم.

```
Input: One chromosome: Ch1
Output: One mutated chromosome: Ch2
1. procedure MUTATION
2. Ch2← Ch1
3. randomly select a virtual machine v;
   where 1≤ v ≤ |v|
4. randomly select a physical machine
   p; where 1≤ P ≤ |P|
5. remove v from its current physical
   machine
6. add v to p
   return Ch2
end procedure
```

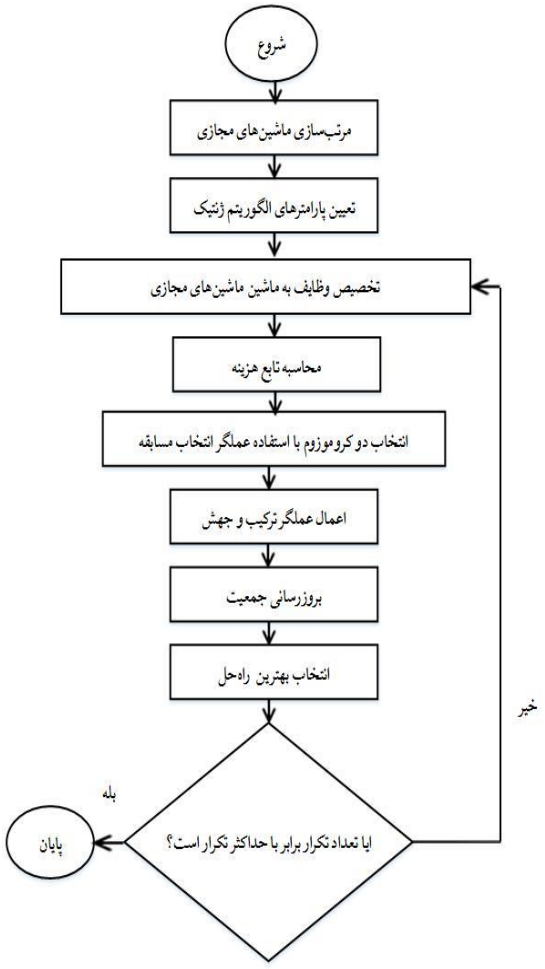
شکل ۵: شبه کد عملگر جهش روش پیشنهادی

۸- شبیه‌سازی روش پیشنهادی

ارزیابی الگوریتم پیشنهادی در زیرساخت مراکز داده ابری در مقیاس بزرگ و بر روی یک زیرساخت واقعی بسیار دشوار می‌باشد. در نتیجه برای اطمینان از تکرار آزمایش‌ها شبیه‌سازها به عنوان راهی برای ارزیابی عملکرد پیشنهادی انتخاب شده‌اند ابزار کلودسیم به عنوان یک بستر شبیه‌سازی انتخاب شده است [۲۵]. زیرا یک چارچوب شبیه‌سازی مدرن می‌باشد که برای محاسبات ابری طراحی شده است. این ابزار برخلاف سایر ابزارهای شبیه‌سازی، امکان مدل‌سازی محیط‌های مجازی پشتیبانی از تأمین منابع درخواستی و مدیریت آنها را فراهم می‌کند این شبیه‌ساز برای برنامه‌های آگاه از انرژی نیز توسعه یافته است جدا از مدل‌سازی مصرف انرژی قابلیت شبیه‌سازی برنامه‌های کاربردی با حجم کاری پویا نیز گنجانده شده است. برای ارزیابی روش پیشنهادی از حجم کاری دنیای واقعی برای ارزیابی بهتر استفاده کرده‌ایم حجم کاری استفاده شده، حجم کاری واقعی

می‌باشد که توسط داده‌های Planet Lab ایجاد شده است. در نتیجه در ابزار کلودسیم ما یک مرکز داده شبیه به ۸۰۰ گره فیزیکی ناهمگن را شبیه‌سازی کرده‌ایم که نیمی از آن‌ها سرورهای HP ProLiant ML110 G می‌باشند و نیمی دیگر سرورهای ProLiant ML 110 5G می‌باشند. مشخصات این سرورها بر اساس [۲۶] در جدول (۱) آورده شده است. هر دسته از سرورها دارای سرعت پردازش میزان حافظه، تعداد هسته و پهنای باند متفاوتی می‌باشند. مشخصات انواع ماشین‌های مجازی استفاده شده بر اساس نمونه‌های واقعی آمازون EC2 می‌باشد [۲۷]. با این تفاوت که همه ماشین‌های مجازی تک هسته‌ای هستند زیرا داده‌های حجم کاری مورد استفاده برای شبیه‌سازی از ماشین‌های مجازی تک هسته ای استفاده می‌کنند در ابتدا ماشین‌های مجازی با توجه به منابع مورد نیاز تخصیص داده می‌شوند با این، حال در طول عمر شبیه‌سازی ماشین‌های مجازی با توجه به حجم کار از منابع کمتری استفاده می‌کنند و فرصت‌هایی برای تجمیع ماشین‌های مجازی ایجاد می‌شود مشخصات انواع ماشین‌های مجازی استفاده شده در جدول (۲) آورده شده است. ماشین‌های مجازی از نظر سرعت پردازش و میزان حافظه متفاوت می‌باشند همانطور که در جدول (۲) نشان داده شده است.

حجم‌کاری استفاده شده حجم کاری واقعی می‌باشد که توسط داده‌های Planet Lab در ده روز مختلف دریافت شده است [۲۸].



شکل ۶: فلوچارت چارجوب روش پیشنهادی

```
Input: Size of jobs, PM resources
Output: Mapping of Jobs to VMs
1: procedure THE GA
       generate a population of POPSIZE number of random individuals, POP ;
2:     while THE TERMINATION CONDITION is not true do
3:         for each individual i in POP do4:     calculate its fitness value f (i)
5:         end for
6:         for each Individual i in POP do
7:             invoke the Selection Operation, that is using tournament selectiontechnique to select another individual to pair
8:         end for
9:         for each pair of parents do
10:            use Arithmetic Crossover Operation to produce an offspring
11:        end for
12:        for each offspring do
13:            apply Mutation Operation according to mutation rate
14:        end for
15:        find the best individual among offsprings, newBest
16:        if newBest is better than current best individual then
17:            replace current best individual with newBest
18:        end if
19:    end while
       return best individual
end
procedure
```

شکل ۷: شبه کد چارجوب روش پیشنهادی

این داده‌ها از داده‌های استفاده از پردازنده توسط بیش از ۱۰۰۰ ماشین مجازی در یک زمان از سرورهای واقع در بیش از ۵۰۰ مکان آورده شده است برای این منظور به طور تصادفی ۱۰ روز از بین جریان کار جمع‌آوری شده در ماه‌های مارس و آوریل ۲۰۱۱ انتخاب شده است. مشخصات داده‌ها در جدول (۳) نمایش داده شده است.

الگوریتم پیشنهادی و روش مورد مقایسه با استفاده از محیط توسعه نت بینز در نرم افزار کلودسیم ورژن ۳ کد نویسی شده‌اند و در سیستم عامل ویندوز ۳۲ بیتی و با حافظه 16G اجرا گردیده است.

جدول ۱: مشخصات میزبان‌های فیزیکی در شبیه‌سازی

ماشین مجازی	large–Extra instance	Small instance	Micro instance
سرعت پردازش (MIPS)	۲۰۰۰	۱۰۰۰	۵۰۰
حافظه (MB)	۳۷۵۰	۱۷۰۰	۶۱۳
پهنای باند (Gbit/s)	۱	۱	۱
سایز (GB)	۲.۵	۲.۵	۲.۵

جدول ۲: مشخصات ماشین‌های مجازی در شبیه‌سازی

میزبان فیزیکی	سرعت پردازش (MIPS)	حافظه (MB)	تعداد هسته	پهنای باند (Gbit/s)	سایز (GB)
HP ProLiant ML110 G4	۱۸۶۰	۴۰۹۶	۲	۱	۱
HP ProLiant ML110 G5	۲۶۶۰	۴۰۹۶	۲	۱	۱

جدول ۳: مشخصات داده‌های جریان کار

ردیف	تاریخ	تعداد ماشین‌های مجازی	Median	St. Dev
۱	۲۰۱۱/۰۳/۰۳	۱۰۵۲	٪۶	٪۱۷/۹
۲	۲۰۱۱/۰۳/۰۶	۸۹۸	٪۵	٪۱۶/۸۳
۳	۲۰۱۱/۰۳/۰۹	۱۰۶۱	٪۴	٪۱۵/۵۷
۴	۲۰۱۱/۰۳/۲۲	۱۵۱۶	٪۵	٪۱۲/۷۸
۵	۲۰۱۱/۰۳/۲۵	۱۰۷۸	٪۶	٪۱۴/۱۴
۶	۲۰۱۱/۰۴/۰۳	۱۴۶۳	٪۶	٪۱۶/۵۵
۷	۲۰۱۱/۰۴/۰۹	۱۳۵۸	٪۶	٪۱۵/۰۹
۸	۲۰۱۱/۰۴/۱۱	۱۲۲۳	٪۶	٪۱۵/۰۷
۹	۲۰۱۱/۰۴/۱۲	۱۰۵۴	٪۶	۱۵٪/۱۵
۱۰	۲۰۱۱/۰۴/۲۰	۱۰۳۳	٪۴	۱۵٪/۲۱

۸-۱- ارزیابی روش پیشنهادی

معیارها و پارامترهای کیفیت سرویسی که در روش پیشنهادی در نظر گرفته شده است انرژی ،مصرفی میزان تخطی کلی از توافقنامه سطح سرویس (SLA)[۱۲] و تعداد مهاجرت‌ها می‌باشند. کارهایی که جهت مقایسه در نظر گرفته شده است راهکار AMUT در مقاله [۲۹] و راهکار EQVS در مقاله [۳۰] می‌باشند. علت انتخاب این روش‌ها سازگاری بیشتر روش‌های مورد مقایسه با محیط شبیه‌سازی و استفاده از الگوریتم‌های فرااکتشافی و غیر فرااکتشافی می‌باشد. ایده روش پیشنهاد شده در این مقاله، رویکردی جدید مبتنی بر الگوریتم تکاملی تفاضلی برای حل مساله توزان بار فضای ابری می‌باشد که در رویکرد ارائه شده سعی می‌شود بر اساس تابع سودمندی چندهدفه تعریف شده و در نظر گرفتن بردارهای جهش و تقاطع بر اساس بهره وری از منابع و انرژی مصرفی، قصد داریم با استفاده بهینه از منابع و بهبود بهره‌وری در کاهش انرژی مصرفی، تعداد مهاجرت‌ها و نقض کیفیت سرویس تأثیرگذار باشیم، برای بررسی روش پیشنهادی و روش‌های مورد مقایسه از شبیه‌ساز کلودسیم استفاده شده است. در ادامه نمودارهای حاصل از شبیه‌سازی بر روی داده‌های PlanetLab در شکل‌های (۸) تا (۱۱) برای پارامترهای کیفیت سرویس مورد نظر آورده شده است. در ابتدا و در شکل (۸) میزان تخطی کلی از توافقنامه سطح سرویس (SLA) نشان داده شده است. سپس در شکل (۱۰) میزان کاهش SLA در اثر انجام عملیات مهاجرت نشان داده شده است.

۸-۱-۱- بررسی نقض توافق‌نامه سطح سرویس

در محیط ابری برای در نظر گرفتن کیفیت سرویس، معیاری به نام توافق‌نامه سطح سرویس(SLA) در نظر گرفته شده است. نقض توافق‌نامه سطح سرویس در محیط محاسبات ابری بسیار نامطلوب است و هزینه‌های سنگینی دارد برای این منظور دو معیار برای اندازه‌گیری سطح نقض-توافقنامه سطح سرویس در نظر گرفته می‌شود [۳۰].

SLATAHT[۱۳]: نقض توافق‌نامه سطح سرویس میزبان (فعال): زمانی که ماشین‌های فیزیکی فعال هستند و بهره ۱۰۰٪ پردازنده را تجربه می‌کنند و دچار برباری می‌شوند.

$$SlATAH = \frac{1}{N}\sum_{i=1}^{N}\frac{Ts_i}{Ta_i} \qquad (۱۶)$$

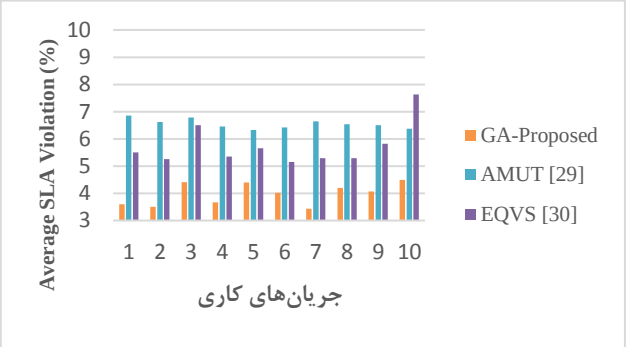
$$PDM = \frac{1}{M}\sum_{i=1}^{N}\frac{Cd_J}{Cr_J} \qquad (۱۷)$$

PDM[۱۴](نقض کارایی با توجه به مهاجرت): کاهش کارایی کل یا مهاجرت ماشین‌های مجازی N تعداد منابع محاسباتی Ts_i زمان

کل در زمانی که ماشین فیزیکی‌ام بهره‌ی ۱۰۰٪ را متحمل می‌شود که نقض کیفیت سرویس را ایجاد می‌کند. Ta_i کل ماشین فیزیکی iام که در حالت فعال است، M تعداد ماشین‌های مجازی Cd_j نقض کارایی ماشین مجازی iام که یا مهاجرت ایجاد می‌شود را برآورد می‌کند. Cr_j کل ظرفیت پردازنده درخواست شده توسط ماشین مجازی iام در زمان اجرای آن است. در آزمایشات Cd_j با ۱۰ درصد از بهره پردازنده در MIPS در طول همه مهاجرت‌های ماشین مجازی iام برآورد شده است. از آنجا که هر دو معیار SLATAH و PDM به طور مستقل و با اهمیت برابر سطح توافقنامه سطح سرویس نقض شده را تعیین می‌کنند. بنابراین یک معیار ترکیبی که شامل هر دو نقض کارایی یا توجه به درماشین فیزیکی پربار و با توجه به مهاجرت ماشین مجازی است در نظر گرفته می‌شود [۳۰].

$$SLAV = SLATAH * PDM \qquad (۱۸)$$

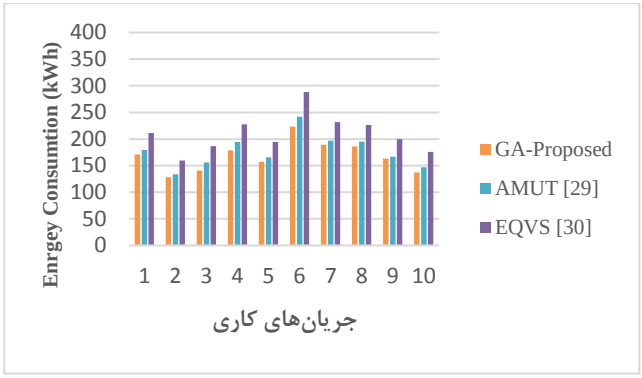
همانطور که در شکل (۸) قابل مشاهده است در راهکار پیشنهادی با بکارگیری تابع هدف چندمعیاره توانستیم نسبت به روش AMUT به میزان ۶۲ درصد و نسبت به روش EQVS به میزان ۳۰ درصد در کاهش درصد تخطی از توافق نامه سطح سرویس تاثیرگذار باشیم.



شکل ۸: درصد تخطی از توافق نامه سطح سرویس

۸-۱-۲- بررسی انرژی مصرفی

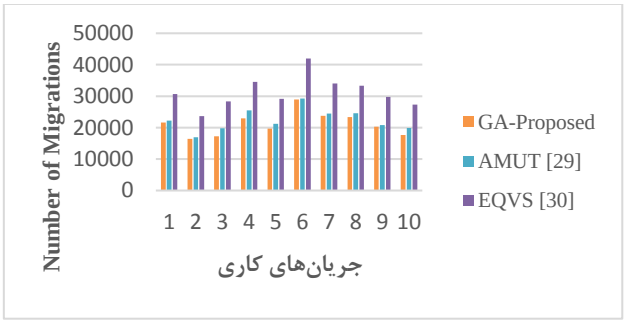
همانطور که در شکل (۹) ملاحظه می‌گردد، میزان مصرف انرژی در راهکار پیشنهادی به نسبت سه راهکار AMUT و EQVS تا حد زیادی کاهش پیدا کرده که این امر به دلیل چندهدفه بودن تابع برازش، زمانبندی‌های مناسب و همچنین پخش نمودن متوازن بار بین سرورهای پردازشی و خاموش نمودن سرورهای بیکار، ایجاد شده است؛ بطوریکه نسبت به زمانبندی مبتنی بر AMUT به میزان ۸ درصد و نسبت به زمانبندی مبتنی بر EQVS به میزان ۲۸ درصد کاهش مصرف انرژی بوجود آمده است. با توجه به آنکه در هر بار عملیات زمانبندی میزان مصرف منابع و انرژی مورد بررسی قرار می‌گیرد این کاهش انرژی قابل پیش‌بینی بود.



شکل ۹: میزان مصرف انرژی

۸-۱-۳- بررسی تعداد مهاجرت‌های ماشین مجازی

در شکل (۱۰) میزان تنزل SLA در اثر عملیات مهاجرت نشان داده شده است. با توجه به اینکه عملیات مهاجرت می‌تواند بر روی ترافیک شبکه تاثیرگذار باشد و همچنین توافقی که در اجرای دستورات می‌تواند داشته باشد، بر همین اساس بررسی کردن این پارامتر نیز لازم دیده شد. همانطورکه ملاحظه می‌شود راهکار پیشنهادی کمترین تنزل را داشته است زیرا کلیه مهاجرت‌ها بهینه و با توجه به نیاز زیرساخت بوده و در نتیجه کمترین کاهش کیفیت سرویس ایجاد شده است. در واقع پارامترهای کیفیت سرویس در قرارداد سطح سرویس مشخص می‌کنند که کیفیت ارایه شده تا چه اندازه مناسب بوده است. مشتریان از طریق این قرارداد از سطح کیفیت خدمات مورد ارایه اطمینان حاصل می‌کنند. همانطور که در شکل (۱۰) قابل مشاهده است راهکار پیشنهادی با بکارگیری تابع هدف چندمعیاره توانسته است در مقایسه با دو راهکار AMUT و EQVS تعداد مهاجرت‌ها را به ترتیب ۴ درصد و ۴۵ درصد کاهش دهد و از مهاجرت‌های اضافی جلوگیری نماید.



شکل ۱۰: تعداد مهاجرت ماشین‌های مجازی

۹- نتیجه‌گیری

پتانسیل بالایی که رایانش ابری برای ذخیره‌سازی و پردازش داده‌ها، از راه دور دارد، این فناوری نمونه‌ای جدید از محاسبات را ارایه داده است. اخیراً بسیاری از شرکت‌های بزرگ، بجای استفاده از منابع

داخلی، آن‌ها را به فضاهای ابری برون سپاری نموده‌اند. لذا کاربران در هر زمان و در هر نقطه از دنیا می‌توانند به داده‌ها و اطلاعات خود دسترسی داشته باشند و همچنین کاربران نیاز به سیستم‌های سخت‌افزاری با قابلیت پردازش و ظرفیت ذخیره سازی بالا ندارند. زیرا تمامی اعمال رایانشی و ذخیره سازی در بخش ارائه‌دهندگان خدمات ابری و توسط سرورهای مجهز و پیشرفته انجام می‌گیرد. در محیط‌های ارائه دهنده‌ی خدمات ابری، ماشین‌های مجازی از چندین سازمان روی همان سرور فیزیکی قرار داده شده‌اند که باعث می‌شود بازده مجازی‌سازی به حداکثر میزان خود برسد. در چنین زیرساخت مقیاس‌پذیری، بدون فراهم کردن راهکارهائی جهت مدیریت بارهای ترافیکی و پخش آن‌ها بین سایر گره‌ها، می تواند سبب ایجاد گلوگاه و ازدحام شود که در نتیجه آن کارائی می‌تواند به شدت پائین بیاید. بر همین اساس نیاز به راهکارهائی جهت متوازن نمودن بار و پخش آن در بین سایر گره‌های پردازشی است. در واقع از طریق توازن بار بهینه، درخواست‌ها بصورت پویا و متوازن در میان تمامی گره‌ها، با حصول اطمینان از تخصیص عادلانه و کارآمد هر منبع رایانشی، توزیع خواهد شد. این فرایند، منجر به افزایش رضایت‌مندی کاربر و بهره‌برداری بالا از منابع می‌شود؛ که نهایتاً افزایش کارایی را به همراه خواهد داشت. در واقع هدف اصلی از ایجاد توازن، یافتن نگاشتی مناسب از کارها بر روی پردازنده های موجود در سیستم است به طوری که در هر پردازنده مقدار تقریباً مساوی از کارها اجرا گردد تا زمان اجرای کلی به کمترین مقدار خود برسد. برای این منظور در رایانش ابری از متوازن کننده بار استفاده می شود. مهمترین وظیفه این مولفه، دریافت کارهای کاربران و توزیع آنها بر روی سرورها و میزبان‌های متفاوت می‌باشد، به گونه‌ای که دسترسی به داده‌ها تا حد امکان محلی شده و محاسبات نیز بصورت متوازن بر روی سرورها قرار گیرند. این عمل زمان پاسخ را حداقل، بهره‌وری منابع را افزایش و کارائی سیستم را بالا می‌برد. معمولاً متوازن‌کننده بار بصورت نرم‌افزاری بوده و در آن اطلاعات مربوط به هر کدام از منابع و سرورها موجود می‌باشد و تعیین می‌کند که درخواست مورد نظر به کدام سرور منتقل و اجرا شود. جهت تعادل بار معیارهای مختلفی از جمله میزان سربار، توان عملیاتی، کارایی، بهره برداری از منابع، زمان پاسخ، تحمل‌پذیری خطا و زمان مهاجرت وجود دارد که هر الگوریتم بسته به نوع آن، نسبت به بهبود این معیارها اقدام می‌نماید. بر همین اساس در این مقاله یک راهکار پیشرفته مبتنی بر الگوریتم ژنتیک بهینه یافته به منظور زمانبندی و ایجاد توازن بار در زیرساخت ابری ارایه شد، بطوریکه بتوان زمان اجرای درخواست کاربران را بصورت قابل توجهی در مراکز ابری حداقل نمود. تکنیک پیشنهادی از طریق تخصیص بهینه وظایف به سرورهای پردازشی، از افزایش بیش از حد یا کم باری در سرورها جلوگیری بعمل می آورد. همچنین سرورهائی که دارای بار زیاد هستند و بعبارتی دچار ازدحام شده‌اند، با استفاده از مهاجرت زنده ماشین مجازی، وظایف آن‌ها به یک سرور دیگر انتقال داده شود تا از این طریق زمینه‌سازی افزایش توازن بار در زیرساخت ابری فراهم شود و در نتیجه زمان اجرای وظایف به میزان قابل توجهی کاهش یابد. در انتها جهت ارزیابی، راهکار پیشنهادی در محیط شبیه‌ساز کلودسیم نسخه ۳ پیاده‌سازی و شبیه‌سازی شد و **بر** روی حجم‌کار بیش از هزار ماشین مجازی بر روی داده‌ها PlantLab مورد ارزیابی قرار داده شد. نتایج حاصل از شبیه‌سازی نشان می‌دهد که راهکار پیشنهادی توانسته است که معیار نقض توافق‌نامه سطح سرویس را در مقایسه با روش‌های AMUT و EQVS را به طور میانگین به میزان ۴۶ درصد، معیار مصرف انرژی را به طور میانگین به میزان ۱۸ درصد و معیار تعداد مهاجرت‌های ماشین مجازی را به طور میانگین به میزان ۲۴ درصد بهبود دهد.

جهت بهینه‌تر نمودن راهکار پیشنهادی می‌توان از دو عامل جهت زمانبندی و توازن‌بار استفاده نمود. به اینصورت که در اولین مرحله وظایف آماده جهت اجرا، به کمک الگوریتم ژنتیک بر روی سرورهای پردازشی زمانبندی می‌شوند در این مرحله ابتدا میزان منابع آزاد سرورها از طریق پارامترهائی که به الگوریتم ژنتیک اضافه شده است، محاسبه و بر اساس آن، وظایف به سرورهای کاندید، جهت اجرا ارسال می‌شوند. سپس در فاز بعدی پس از زمانبندی و شروع پردازش وظایف، این سرورها بصورت دوره‌ای و با بهره‌گیری از الگوریتم ژنتیک مورد بررسی قرار می‌گیرد تا در صورتیکه در اثر بروز خطا (سخت‌افزاری یا نرم‌افزاری) دچار ترافیک زیاد و یا ازدحام شده باشند، مشخص و به کمک تکنیک مهاجرت، وظایف آن‌ها به سایر سرورهای کاندید انتقال داده شود. با بهره‌گیری از این سیستم دوعاملی در واقع علاوه بر کاهش زمان اجرای وظایف، با ایجاد یک نوع توازن بار بهینه، می‌توان قابلیت اطمینان در زیرساخت ابری را نیز افزایش داد و در نتیجه زمینه‌سازی بهبود توافقنامه سطح سرویس که یکی از پارامترهای مهم جهت تضمین کیفیت و رضایتمندی کاربر می‌باشد، نیز فراهم گردد.

مراجع

[1] Sharma, N., S. Tyagi, and S. Atri, A Comparative Analysis of Min-Min and Max-Min Algorithms based on the Makespan Parameter. International Journal of Advanced Research in Computer Science, 2017. 8(3).

[2] Abd Elaziz, M., et al., Task scheduling in cloud computing based on hybrid moth search algorithm and differential evolution. Knowledge-Based Systems, 2019. 169: p. 39-52.

[3] Chaudhry, S.A., et al., An improved anonymous authentication scheme for distributed mobile cloud computing services. Cluster Computing, 2019. **22**(1): p. 1595-1609.

[4] Mishra, S.K., B. Sahoo, and P.P. Parida, Load balancing in cloud computing: a big picture. Journal of King Saud University-Computer and Information Sciences, 2020. 32(2): p. 149-158.

[5] Kumar, P. and R. Kumar, Issues and challenges of load balancing techniques in cloud computing: A survey. ACM Computing Surveys (CSUR), 2019. **51**(6): p. 1-35.

[6] Ala'Anzy, M. and M. Othman, Load balancing and server consolidation in cloud computing environments: a meta-study. IEEE Access, 2019. **7**: p. 141868-141887.

[7] Priya, V., C.S. Kumar, and R. Kannan, Resource scheduling algorithm with load balancing for cloud service provisioning. Applied Soft Computing, 2019. **76**: p. 416-424.

[8] Pradhan, P., P.K. Behera, and B. Ray, Modified round robin algorithm for resource allocation in cloud computing. Procedia Computer Science, 2016. **85**: p. 878-890.

[9] Mishra K, Pati J, Majhi SK. A dynamic load scheduling in IaaS cloud using binary JAYA algorithm. Journal of King Saud University-Computer and Information Sciences. 2022 Sep 1;34(8):4914-30.

[10] Gulbaz R, Siddiqui AB, Anjum N, Alotaibi AA, Althobaiti T, Ramzan N. Balancer genetic algorithm—A novel task scheduling optimization approach in cloud computing. Applied Sciences. 2021 Jul 6;11(14):6244.

[11] Kashikolaei SM, Hosseinabadi AA, Saemi B, Shareh MB, Sangaiah AK, Bian GB. An enhancement of task scheduling in cloud computing based on imperialist competitive algorithm and firefly algorithm. The Journal of Supercomputing. 2020 Aug;76:6302-29.

[12] Lim J, Lee D. A load balancing algorithm for mobile devices in edge cloud computing environments. Electronics. 2020 Apr 23;9(4):686.

[13] Talaat FM, Saraya MS, Saleh AI, Ali HA, Ali SH. A load balancing and optimization strategy (LBOS) using reinforcement learning in fog computing environment. Journal of Ambient Intelligence and Humanized Computing. 2020 Nov;11:4951-66.

[14] Bi S, Huang L, Wang H, Zhang YJ. Lyapunov-guided deep reinforcement learning for stable online computation offloading in mobile-edge computing networks. IEEE Transactions on Wireless Communications. 2021 Jun 9;20(11):7519-37.

[15] Kumar KP, Ragunathan T, Vasumathi D, Prasad PK. An efficient load balancing technique based on cuckoo search and firefly algorithm in cloud. Algorithms. 2020 May 1;423:422-32.

[16] Srivastava S, Saxena S, Buyya R, Kumar M, Shankar A, Bhushan B. CGP: Cluster-based gossip protocol for dynamic resource environment in cloud. Simulation Modelling Practice and Theory. 2021 Apr 1;108:102275.

[17] Shukri SE, Al-Sayyed R, Hudaib A, Mirjalili S. Enhanced multi-verse optimizer for task scheduling in cloud computing environments. Expert Systems with Applications. 2021 Apr 15;168:114230.

[18] Mahapatra B, Turuk AK, Panda SK, Patra SK. Utilization-aware VB migration strategy for inter-BBU load balancing in 5G cloud radio access networks. Computer Networks. 2020 Nov 9;181:107507.

[19] Pradhan A, Bisoy SK. A novel load balancing technique for cloud computing platform based on PSO. Journal of King Saud University-Computer and Information Sciences. 2022 Jul 1;34(7):3988-95.

[20] Pandey, S., et al. A particle swarm optimization-based heuristic for scheduling workflow applications in cloud computing environments. in 2010 24th IEEE international conference on advanced information networking and applications. 2010. IEEE.

[21] Shi, C., et al. Ultra-low latency cloud-fog computing for industrial internet of things. in 2018 IEEE Wireless Communications and Networking Conference (WCNC). 2018. IEEE.

[22] Skarlat, O., et al., Optimized IoT service placement in the fog. Service Oriented Computing and Applications, 2017. **11**(4): p. 427-443.

[23] Fan, X., W.-D. Weber, and L.A. Barroso, Power provisioning for a warehouse-sized computer. ACM SIGARCH computer architecture news, 2007. **35**(2): p. 13-23.

[24] Blickle, T. and L. Thiele, A comparison of selection schemes used in evolutionary algorithms. Evolutionary Computation, 1996. **4**(4): p. 361-394.

[25] Neves Calheiros R, Ranjan R, Beloglazov A, De Rose CA, Buyya R. CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. Software: Practice and Experience. 2011:23-50.

[26] tandard Performance EvaluationCorporation. https://www.spec.org/power_ssj2008/results/

[27] Amazon EC2 Instance Types .

https://aws.amazon.com/ec2/instance- ypes/

[28] Park K, Pai VS. CoMon: a mostly-scalable monitoring system for PlanetLab. ACM SIGOPS Operating Systems Review. 2006 Jan 1;40(1):65-74.

[29] Wang J, Gu H, Yu J, Song Y, He X, Song Y. Research on virtual machine consolidation strategy based on combined prediction and energy-aware in cloud computing platform. Journal of Cloud Computing. 2022 Sep 24;11(1):50.

[30] Tarafdar A, Debnath M, Khatua S, Das RK. Energy and quality of service-aware virtual machine consolidation in a cloud data center. The Journal of Supercomputing. 2020 Nov;76:9095-126.

پاورقی‌ها:

[1] Task Scheduling
[2] Cloud Computing
[3] Load Balancing
[4] Metaheuristic algorithms
[5] Ant Colony Algorithm (ACO)
[6] Genetic algorithm
[7] Non Deterministic Polynomia
[8] Virtual machines
[9] Cuckoo Optimization Algorithm
[10] Firefly Algorithm
[11] Particle Swarm Optimization (PSO)
[12] Encoding Chromosome
[13] Population s
[14] Fitness Function
[15] Selection
[16] Mutation