موازنه‌گر نامتمرکز بار در محیط ابر با بهره‌گیری از سیاست تصمیم‌گیری چندشاخصه

شهرام جمالی[۱]، دانشیار، سمیرا حورعلی[۲]، مربی
۱- دانشیار - گروه آموزشی مهندسی کامپیوتر و فناوری اطلاعات - دانشگاه محقق اردبیلی - اردبیل - ایران - jamali@iust.ac.ir
۲- کارشناس ارشد معماری سیستم‌های کامپیوتری - گروه آموزشی مهندسی کامپیوتر و الکترونیک - مربی دانشگاه غیرانتفاعی- غیردولتی شاهرود - شاهرود - ایران - s.hourali68@gmail.com

چکیده: در محیط ابر، موازنه بار از طریق انتخاب ماشین مجازی مناسب از بین ماشین‌های مجازی موجود، جهت اجرای کار دریافت‌شده، صورت می‌گیرد. انتخاب ماشین مجازی مناسب برای انجام هر کار، تابع پارامترهای مختلفی است. در این مقاله با در نظر گرفتن تک‌تک پارامترها، مناسب‌ترین ماشین مجازی را برای کار موردنظر انتخاب می‌نمائیم. این انتخاب به‌صورت یک مسئله تصمیم‌گیری چندشاخصه تعریف می‌شود. ابتدا با در نظر گرفتن اهداف اساسی توازن بار، مسئله در قالب پارامترهای مؤثر در کارایی مدل می‌شود؛ سپس مدل فوق با استفاده از روش تاکسونومی غیرکلاسیک که از پرکاربردترین روش‌های تصمیم‌گیری چندشاخصه است، حل می‌شود. در این روش که TLB نامیده شده است، مطلوبیت هر ماشین مجازی با توجه به وزن اختصاص داده‌شده به معیارها و میزان اهمیت هر یک از معیارها برای کاربر که بر اساس شاخص آنتروپی تعیین می‌شود، محاسبه می‌گردد. درنهایت بهترین ماشین مجازی بر اساس ارزش اختصاص‌یافته انتخاب می‌شود. جهت بررسی کارایی روش پیشنهادی، شبیه‌سازی‌های گسترده‌ای در محیط CloudSim انجام شده است که نشان می‌دهد روش پیشنهادی نسبت به روش‌های FIFO، DLB، WRR و HBB-LB عملکرد بهتری دارد.

واژه‌های کلیدی: محاسبات ابری، موازنه بار، ماشین مجازی، روش تاکسونومی غیرکلاسیک، تکنیک آنتروپی.

Decentralized Load Balancer in Cloud Environment by using Multi Attribute Decision Making Policy

S. Jamali, Associated. Professor [1], S. Hourali, Instructor [2]

1- Faculty of Computer and Information Technology Engineering, University of Mohaghegh Ardabili, Ardabil, Iran, jamali@iust.ac.ir
2- Computer and Electronic Engineering Department, Non Benefit University of Shahrood, Shahrood, Iran, s.hourali68@gmail.com

Abstract: In the cloud environment, load balancing is done by choosing the appropriate VM (Virtual Machine) between existing VMs to execute the given task. Choosing the appropriate VM to do any task, is function of various parameters. In this paper, we choose the most appropriate VM to execute the task by considering each of criteria. Therefore, we consider this selection as a Multi Attribute Decision Making (MADM) problem. At first, we model the problem in terms of effective parameters in performance by considering the basic goals of the load balancing. Then, we solve the model by using the non-determenestic taxonomy method, which is one of the most widely used method for MADM problems. In this method which is called TLB, desirability of each VM is calculated according to the weights assigned to the criteria and the level of importance of each criterion for the user which is determined based on entropy method.Finally, the best VM is choosen based on its assigned value. To study the performance of the proposed approach, extensive simulations is carried out in CloudSim simulator which show that the proposed method has better performance compared to FIFO, DLB and WRR method.

Keywords: Cloud computing, load balancing, virtual machine, non-classical taxonomy method, entropy method.

تاریخ ارسال مقاله: ۱۳۹۴/۰۲/۰۴
تاریخ اصلاح مقاله: ۹۴/۰۳/۱۶ و ۹۴/۰۴/۲۰ و ۱۳۹۴/۰۶/۲۲
تاریخ پذیرش مقاله: ۱۳۹۴/۰۹/۱۷
نام نویسنده مسئول: شهرام جمالی
نشانی نویسنده مسئول: ایران - اردبیل - دانشگاه محقق اردبیلی - دانشکده فنی مهندسی

۱- مقدمه

متخصصـان و تحلیـل‌گران صـنعت کـامپیوتر و IT معتقدنـد، رشـد سیستم‌های مبتنی بر ابر در سال‌های آینده نه‌تنها متوقف نخواهد شد، بلکه شتاب بیش‌تری نیز خواهد گرفت [۱]. ویژگی‌هـایی از قبیـل هزینـه پایین، عملکرد راضی‌کننده و بـالابـودن تحمل‌پذیری خطا، شـبکه‌هـای کامپیوتری را بـه گزینـه مناسـبی بـرای تحقـق بسـیاری از کاربردهـای امروزی تبدیل کرده است [۲]. در این میان رایـانش ابـری مـدلی اسـت، برای فراهم کردن دسترسی آسان بر اساس تقاضای کاربر از طریق شبکه به مجموعه‌ای از منابع رایانشی قابل‌تغییر و پیکربندی (مثـل شـبکه‌هـا، سرورها، فضای ذخیره‌سازی، برنامه‌های کاربردی و سرویس‌ها) کـه ایـن دسترسی بتواند با کم‌ترین نیاز بـه مـدیریت منـابع و یـا نیـاز بـه دخالـت مستقیم فراهم‌کننده سرویس به سرعت فراهم شده یا آزاد گـردد. از دیـد زیرساخت نیز، ابر به نوعی از سیسـتم‌هـای مـوازی و توزیـع‌شـده گفتـه می‌شود که شامل مجموعه‌ای از کامپیوترهای مجازی به‌هم متصل است [۳].

درواقع، رایانش ابری بـه معنـای اسـتفاده اشـتراکی از برنامـه‌هـا و منابع است. ابرها انبـار بزرگـی از منـابع مجـازی هسـتند کـه بـه راحتـی قابل‌اسـتفاده و در دسـترس هسـتند (ماننـد سـخت‌افزار، پلتفـرم‌هـای توسعه‌یافته و/یا سرویس‌هـا) [۴]. ایـن منـابع مـی‌تواننـد بـه‌صـورت پویـا پیکربندی مجدد شوند تا بهره‌برداری مطلـوبی از منـابع را فـراهم کننـد. این انبار منابع معمولاً به‌وسیله یک مدل بنام "پرداخت به اندازه مصرف" توسـط ارائـه‌دهنـده زیرسـاخت بـر اسـاس توافـق‌هـای سـطح سـرویس بهره‌برداری می‌شود [۵]. ازآنجایی‌که منابع آزاد موجود در محیط ابر، در هر زمان در حال تغییر هستند، مسـئله زمـان‌بنـدی وظـایف امـر مهمـی است که تأثیر زیادی در کارایی محاسبات ابری دارد.

الگوریتم زمان‌بندی روشی است که به‌وسـیله آن وظـایف بـه منـابع موجود در مراکز داده تخصیص داده می‌شود. در محیط ابر، نیـاز اسـت منابع محاسباتی طوری زمان‌بندی شوند که هم ارائه‌دهندگان، حـداکثر استفاده را از منابع آن‌ها ببرند و هم کاربران برنامه‌های کاربردی موردنیاز خود را با کم‌ترین هزینه در اختیار بگیرند. بنابراین، زمـان‌بنـدی یکـی از مهم‌ترین مسائل در ابـر محسـوب مـی‌شـود. محـدودیت و مـوقتی بـودن منابع دو شرطی هستند که به زمـان‌بنـدی تحمیل‌شـده‌اند. بـرای مثـال ممکن است وظایف ترتیب اجرای مشخصی داشته باشند یا یک منبع در یک زمان فقط بتواند یک وظیفه را اجرا کنـد. در محـیط نـاهمگون، یـک تصمیم زمان‌بندی، پیچیده‌تر نیز می‌شود. اهمیـت مشـکل زمـان‌بنـدی باعث شده که تحقیقات وسیعی در این زمینه انجام شود.

انتخـاب یـک زمـان‌بنـد نامناسـب می‌توانـد باعـث ناکارآمـدی سخت‌افزار یا کند شدن برنامـه ابـر شـود [۶]. در برخـی مـوارد، انتخـاب نادرست الگوریتم باعث می‌شود، مسئله‌ای که چند ثانیـه زمـان می‌بـرد، در چندین ساعت حل شود؛ بنابراین یک زمان‌بند خوب بایـد در شـرایط مختلف رفتار مناسبی داشته باشد. استراتژی‌هـای زمـان‌بنـد وظیفـه بـر عـدالت یـا بهـره‌وری منـابع تمرکـز دارنـد کـه هزینـه، زمـان، فضـا، تـوان عملیاتی و کیفیت سرویس در محاسبات ابری را بهبود می‌دهند [۷].

در این مقاله زمان‌بندی کارها در محیط ناهمگن ابـر بـا اسـتفاده از روش تـاکسونـومی غیرکلاسـیک کـه یکـی از روش‌هـای پرکـاربرد در تصمیم‌گیری چندمعیاره است، انجـام مـی‌شـود. همچنـین سـعی شـده است، تمامی ویژگی‌های منابع و کارها کـه میـزان اهمیـت آن‌هـا توسـط تکنیک آنتروپی محاسـبه مـی‌شـود، بـرای اختصـاص منبـع مناسـب بـه کارهای ورودی به ابر در نظر گرفته شود. در ادامه این مقالـه در بخـش ۲ تعدادی از کارهای مطرح که در این زمینه انجام می‌شود، مـورد بررسـی قرار می‌گیرد، در بخش ۳ ساختار مدل در نظر گرفتـه‌شـده بـرای محـیط ناهمگن در نظر گرفته‌شده برای ابر و نحوه رتبـه‌بنـدی منـابع بـر اسـاس تکنیک آنتروپی و روش تصمیم‌گیری تاکسـونومی غیرکلاسـیک توصـیف مـی‌شـود. در بخـش ۴ روش پیشـنهادی موردبررسـی قـرار مـی‌گیـرد و آزمایش‌ها و نتایج در بخش ۵ ارائه می‌شود. فصل ۶ به جمع‌بندی مقالـه می‌پردازد.

۲- کارهای مرتبط

انواع متنوعی از الگوریتم‌هـای زمـان‌بنـدی در سیسـتم‌هـای توزیع‌شـده وجـود دارد. هـدف اصـلی الگـوریتم‌هـای زمـان‌بنـدی بـه دسـت آوردن عملکـرد محاسـباتی بـالا و بهتـرین تـوان عملیـاتی سیسـتم اسـت [۸]. الگوریتم‌های زمان‌بندی کار سنتی قادر به فـراهم کـردن زمـان‌بنـدی در محیط ابر نیستند، زیرا سربار هزینه دارند و لذا فراهم‌کننـدگان بـه‌سـمت استفاده از الگوریتم‌های اکتشافی یا ترکیبی می‌روند.

روش‌های اکتشافی با فرض‌های واقعی در مورد سیستم و گسـترده کردن دامنـه جسـتجو، ضـمن لحـاظ کـردن اطلاعـات سیسـتمی نظیـر پردازنده‌ها، بارکاری و غیـره تمـام راه‌حـل‌هـای ممکـن را یافتـه و جـواب نزدیک به بهینه را پیدا می‌کند. این نوع روش‌ها به دو دسته ایسـتا و پویـا تقسیم می‌شوند. روش اکتشافی ایستا هنگـامی اسـتفاده مـی‌شـود کـه مجموعه کامل از وظایف قبل از اجرا شـناخته شـوند. ایـن اسـتراتژی‌هـا تحت دو فرض اجرا می‌شوند. اول اینکه وظایف به‌طور همزمان می‌رسند و فرض دوم این است کـه ماشـین‌هـای موجـود، بعـد از هـر زمـان‌بنـدی وظیفه به‌روز می‌شوند.

در [۹، ۱۰] کارهای رسیده به محیط ابر بر اساس نیازهای متفاوتی که دارند ابتدا در کـلاس‌هـای مختلفـی طبقـه‌بنـدی مـی‌شـوند، سـپس کارهای هر کلاس را به صف مربوط به همان کلاس که پهنای بانـد و وزن مخصوص به خود را دارد وارد می‌شوند. درنهایـت کارهـا از صـف‌هـا وارد زمان‌بند می‌شوند این زمان‌بند بـر اسـاس الگـوریتم RounRobin عمـل می‌کند. این الگوریتم رفع مشکل گرسنگی[1] را تضمین می‌کند، چون بـر این باور است که سرویس‌های هر کلاس حداقل مقـدار پهنـای بانـد لازم برای انجام کارها را دارند. در [۱۱] رویکرد اولویت‌بنـدی وظـایف و تولیـد زمان‌بندی بر اساس اولویت ارائه شده است. این اولویت بر اسـاس زمـان اتمام وظیفه مورد انتظار بـر روی یـک منبـع تولیـد مـی‌شـود. ایـن روش

وظایف را در چند گروه وظایف مستقل تنظیم می‌کند. پس از آن این گروه‌ها مکرراً زمان‌بندی می‌شوند. هر تکرار مجموعه‌ای از وظایف مستقل نگاشت‌نشده را می‌گیرد و برای هر وظیفه، حداقل زمان تکمیل مورد انتظار[۲] (MECT) را تولید می‌کند. وظیفه‌ای که کوچک‌ترین مقدار MECT، بیش از تمام وظایف انتخاب‌شده به منابع مربوطه را دارد، در این تکرار اول زمان‌بندی می‌شود. این تا زمانی که تمام وظایف زمان بندی شوند، ادامه می‌یابد. هدف این الگوریتم، رسیدن به کم‌ترین پاسخ است و برای رسیدن به این هدف، ابتدا وظایفی با زمان تکمیل کم و سپس وظایفی با زمان بیش‌تر را زمان‌بندی می‌کند.

در [۱۲] یک زمان‌بند با نام FIFO ارائه شده است. این زمان‌بند یک کار جدید در صف را بر اساس زمان ورود آن نگه می‌دارد. کاری که در ابتدای لیست انتظار قرار دارد، برای اجرا انتخاب می‌شود. ازآنجایی‌که این زمان‌بند حداقل سربار را دارد، پیاده‌سازی آن بسیار آسان است، اما وظایف با زمان اجرای طولانی، توان عملیاتی ماشین‌ها را پایین می‌آورد.

در [۱۳] موازنه بار بر اساس جستجوی Tabu انجام شده است ، بر این روش یک جستجوی فضای راه‌حل است که ناحیه‌هایی از فضای راه‌حل را که قبلاً جستجو شده‌اند، یادداشت می‌کند تا در جستجوی بعدی حوالی این ناحیه‌ها را جستجو نکند. یک راه‌حل نگاشت، از همان نمایش کروموزوم در الگوریتم ژنتیک استفاده می‌کند. پیاده‌سازی جستجوی Tabu، با یک نگاشت تصادفی تولیدشده از یک توزیع یکنواخت به‌عنوان راه‌حل ابتدایی آغاز می‌گردد. برای دست‌کاری راه‌حل فعلی و حرکت در فضای راه‌حل یک گام کوتاه برداشته می‌شود. هدف از گام کوتاه پیدا کردن نزدیک‌ترین راه‌حل کمینه محلی در فضای راه‌حل است.

روش اکتشافی پویا زمانی که مجموعه وظایف و یا مجموعه ماشین‌ها ثابت نیستد، ضروری است. به‌عنوان‌مثال، همه وظایف به‌طور همزمان نرسند و یا برخی از ماشین‌های در فواصل زمانی به حالت خاموش بروند. روش اکتشافی پویا در دو حالت، برخط[۳] و دسته‌ای استفاده می‌شود. در حالت اول، زمانی که وظایف برسند ابتدا VMهای متناسب با وظایف روی سرورها ایجاد شده، سپس هر وظیفه به یک ماشین زمان‌بندی می‌شود. در حالت دوم، وظایف ابتدا در یک مجموعه جمع‌آوری شده و زمان‌بندی در زمان از پیش برنامه‌ریزی‌شده برای هر وظیفه از مجموعه به‌صورت تفکیکی و طبق VMهای موجود در سرورها، اجرا می‌شود. سپس برای مجموعه وظایف بعدی مجموعه VMها به‌روز شده و زمان‌بندی دوباره اجرا می‌شود.

در حالت اکتشافی برخط، هر وظیفه تنها یکبار زمان‌بندی شده و نتیجه زمان‌بندی نمی‌تواند تغییر کند. حالت اکتشافی برخط برای مواردی با نرخ ورود کم مناسب است.

روش ارائه‌شده در [۱۴] یک نوع استراتژی اکتشافی پویا است که هر وظیفه را به ماشینی که زودترین زمان اتمام را دارد تخصیص می‌دهد. این روش اکتشافی به‌عنوان یک معیار برای حالت برخط، استفاده می‌شود.

الگوریتم ارائه‌شده در [۱۵] نوع استراتژی پویا اکتشافی در حالت برخط است. این الگوریتم از هر دو روش اکتشافی MET و MCT به شیوه‌ای دوره‌ای بر اساس توزیع بار در سراسر ماشین‌ها استفاده می‌کند. الگوریتم MET، بهترین ماشین را انتخاب می‌کند اما وظایف زیادی را به آن تخصیص می‌دهد. اکتشافی MCT، بار را متوازن می‌کند، ولی در همان زمان آن وظیفه را به ماشین که دارای کم‌ترین زمان اجرا است، تخصیص نمی‌دهد. برای مثال، اگر وظایف به‌طور تصادفی برسند، ما می‌توانیم از MET برای رسیدن توازن بار به یک مقدار آستانه داده شده و سپس از MCT، برای ایجاد سطح بار در سراسر ماشین‌ها استفاده کنیم.

در حالت دسته‌ای، وظایف در برخی از لحظات از پیش تعریف‌شده زمان‌بندی می‌شوند. این حالت، روش‌های اکتشافی را قادر می‌سازد تا در مورد زمان اجرای واقعی تعداد زیادی از وظایف اطلاع داشته باشد.

روش ارائه‌شده در [۱۶] یک روش اکتشافی است، که در آن هر وظیفه بر اساس مقدار رأی، به منبع اختصاص داده می‌شود. مقدار رأی، تفاوت بین اولین و دومین زودترین زمان اتمام تعریف می‌شود. وظیفه با مقدار انتظار بیش‌تر به ماشینی با زودترین (کم‌ترین) زمان اتمام تخصیص داده می‌شود.

در [۱۷، ۱۸] محیط به‌صورت یک مدل G/S/M در نظر گرفته شده است که در آن، G تعداد خوشه‌هایی است که محیط توزیع‌شده را تشکیل داده‌اند، S تعداد مکان‌های[۴] محیط توزیع شده است و M تعداد عناصر محاسباتی یا گره‌های پردازشی این محیط است. زمان‌بندی کارها در یک درخت ۴ سطحی[۵] انجام می‌شود. در اولین سطح درخت، یک گره مجازی[۶] در ریشه درخت قرار دارد. که دو وظیفه را برعهده دارد: ۱) مدیریت اطلاعات بارکاری محیط توزیع‌شده، ۲) با دریافت کارها از کاربران، بر اساس نیاز هر کاربر و بار جاری محیط، تصمیم می‌گیرد که کارهای دریافتی باید به کدام قسمت فرستاده شوند. سطح دوم شامل G گره مجازی است، هر یک از گره‌ها به یک خوشه فیزیکی از محیط توزیع‌شده متصل هستند.

هر یک از این گره‌ها پاسخ‌گوی بار کاری یا درخواست‌های خوشه[۷] مربوط به خود هستند. در سطح سوم از درخت، S گره وجود دارد که هر کدام از این گره‌ها به مکان‌های فیزیکی تمام خوشه‌های محیط توزیع-شده متصل هستند. وظیفه اصلی این گره‌ها، مدیریت بارکاری گره‌های پردازشی است. در آخرین سطح از درخت M عنصر محاسباتی وجود دارد که هر کدام از این عناصر به خوشه‌ها و مکان‌های مربوط به خود متصل هستند.

الگوریتم اکتشافی جامعه مورچگان از رفتار اجتماعی مورچه‌ها الهام گرفته است. مورچه‌ها با اینکه نابینا هستند، ولی می‌توانند کوتاه‌ترین مسیر بین لانه و منبع غذایشان را پیدا کنند. مورچه‌ها مسیر بین غذا و لانه خود را در ابتدا به‌صورت کاملاً تصادفی انتخاب می‌کنند و کم‌کم پس از طی چندین دوره رفت و برگشت بین غذا و لانه و افزایش

اثر به جای‌مانده در مسیر کوتاه‌تر، درنهایت همگی در کوتاه‌ترین مسیر حرکت می‌کنند.

وقتی مورچه‌ای دنبال غذا می‌گردد، در طول مسیر حرکت، ماده شیمیایی به‌نام فرومون از خود بر جای می‌گذارند. با استفاده از این فرومون، کوتاه‌ترین مسیر پیدا می‌شود. از این مفهوم برای تخصیص کارها استفاده می‌شود. زمانی که یک منبع به کاری تخصیص و کامل می‌کند، مقدار فرومون آن افزایش می‌یابد. اگر منبعی نتواند کاری را تمام کند برای مجازات از فرومون آن کاسته می‌شود [۱۹]. طبق توضیحات ارائه‌شده در این بخش، روش پیشنهادی در این مقاله در رده روش‌های اکتشافی پویا از نوع دسته‌ای قرار می‌گیرد، در بخش بعدی به توضیح روش پیشنهادی پرداخته شده است.

۳- روش پیشنهادی

در این بخش ابتدا مدل مفروض برای محیط ابر و سپس چگونگی فرمول‌بندی مسئله، مورد اشاره قرار می‌گیرد و در ادامه مسئله فوق با استفاده از روش تاکسونومی غیرکلاسیک حل می‌شود.

۳-۱- مدل در نظر گرفته‌شده برای محیط ابر

فضای تعریف‌شده برای ابر محاسباتی شامل K خوشه پردازشی (سرویس) است. تعداد سرورهای فیزیکی n است که به‌صورت $R=\{R_1, R_2, \ldots, R_n\}$ می‌باشد. تعداد ماشین‌های مجازی (VMها) در هر سرور فیزیکی n است که به‌صورت:

$f \in [1,n]\ VM(R_f)=\{VM_1, VM_2 \ldots, VM_n\}$

نمایش داده می‌شود. هر کاربر نیز در محیط ابر دارای تعداد محدودی کار بوده و تمام کارهای ارائه‌شده به ابر عبارت است از $j=\{j_1, j_2, \ldots, j_z\}$ که در زمان T به ابر ارسال می‌شوند $(0 \le t_j < T)$. T بیانگر مدت‌زمان شبیه‌سازی است.

هر کار j_i در فضای ابر محاسباتی دارای ۵ مشخصه است که عبارت است از $J_i = (cp_i, m_i, IO_i, b_i, d_i, \mathrm{bw}_i)$ که cp_i برحسب MI است، نشان‌دهنده میزان بهره‌برداری کار از CPU است، m_i برحسب MB نشان‌دهنده میزان بهره‌برداری از حافظه، و IO_i برحسب MB نشان‌دهنده میزان بهره‌برداری از ورودی/خروجی است. سایر مشخصات به ترتیب عبارت است از بودجه تخصیص داده‌شده به کار موردنظر برحسب G$، سررسید اجرای کار برحسب واحد زمان (ثانیه) و پهنای باند موردنیاز برای انجام کار برحسب MB/S. مشخصات ذکرشده برای کار، توسط کاربر هنگام ارسال وظیفه موردنظر به ابر مشخص می‌شود.

برای هر گره پردازشی r_j (سرور فیزیکی یا VM) نیز در این محیط پنج مشخصه به‌صورت $r_j = (rcp_j, rm_j, rIO_j, rc_j, rn_j, \mathrm{bw}_j)$ شده است. rcp_j معرف قدرت پردازشی هر گره است که درواقع تعداد دستورات قابل اجرا توسط هر یک از اجزای پردازشی منبع برحسب میلیون در هر ثانیه است (MIPS). rm_j و rIO_j به ترتیب نشان‌دهنده میزان بهره‌برداری از حافظه و ورودی/ خروجی است، که بر مبنای مگابایت‌برثانیه محاسبه می‌شود (MB/s). مقدار rc_j قیمت منبع را برحسب G$ نشان می‌دهد، مقدار rn_j معرف میزان تأخیر (ترافیک) شبکه در دست‌یابی به گره پردازشی است که با واحد ثانیه اندازه‌گیری می‌شود، درنهایت bw_j پهنای باند گره پردازشی موردنظر را نشان می‌دهد که با واحد (MB/S) اندازه‌گیری می‌شود.

انتخاب VM مناسب برای انجام هر کار، تابع پارامترهای مختلفی مانند میزان منابع موردنیاز کار نظیر CPU، حافظه، حجم منابع در اختیار VMها، هزینه و سررسید VMها است. باید با در نظر گرفتن تک‌تک این معیار ها، مناسب‌ترین VM برای کار موردنظر انتخاب شود؛ بنابراین می‌توان این انتخاب را به‌صورت یک مسئله تصمیم‌گیری چندشاخصه در نظر گرفت.

در این مقاله گزینه‌ها، گره‌های پردازشی موجود در ابر هستند و شاخص‌های تصمیم‌گیری شامل هزینه پردازش، زمان لازم برای پردازش و سرعت پردازش است. با ورود هر کار به یک سرور، مدیر زیرساخت مجازی آن سرور ابتدا با استفاده از روش تصمیم‌گیری تاکسونومی غیرکلاسیک شروع به یافتن بهترین VM برای تخصیص به آن کار در خوشه مربوطه می‌کند و در صورت پر بودن تمام VMهای موجود در خوشه، سرورها با استفاده از روش تصمیم‌گیری فوق، اقدام به انتخاب بهترین سرور (از سایر خوشه‌ها) برای انتقال برخی VMهای خود به آن سرورمی‌کند تا ظرفیت جدیدی برای ایجاد VM به وجود آید.

۳-۲- فرمول‌بندی مسئله

برای اتخاذ تصمیم نیاز به فرموله کردن مسئله است که این کار توسط ماتریسی بنام P انجام می‌شود؛ بنابراین لازم است، تمام اطلاعات (کمی یا کیفی) مسئله بر روی ماتریس اولیه P پیاده شود. در شکل ۱ این ماتریس نمایش داده شده است.

شاخص / گزینه	X_1	X_2	....	X_n
A_1	R_{11}	R_{12}	...	R_{1n}
A_2	R_{21}	R_{22}	...	R_{2n}
.	.	.		.
.	.	.		.
.	.	.		.
A_m	R_{m1}	R_{m2}	...	R_{mn}

شکل ۱: ماتریس اولیه **P**

در شکل ۱ A_i نشان‌دهنده گزینه iام، X_j نشان‌دهنده شاخص jام و R_{ij} نشان‌دهنده ارزش شاخص jام برای گزینه iام است. تمام شاخص‌های در نظر گرفته‌شده در این مقاله کمّی می‌باشند، گزینه‌ها شامل VMها یا سرورها، و شاخص‌ها $rcp_j, rm_j, rIO_j, rc_j, rn_j, bw_j$ می‌باشند.

۳-۳- محاسبه ارزش (وزن) شاخص‌ها برای هر کار بر اساس تکنیک آنتروپی شانون

در این مقاله از روش آنتروپی [۲۰، ۲۱] برای محاسبه وزن هر یک ازشاخص‌ها استفاده شده است. با در نظر گرفتن ماتریس P در شکل ۱، به‌صورت زیر عمل می‌کنیم.

مقیاس اندازه‌گیری شاخص‌های کمی می‌توانند با یکدیگر متفاوت باشند (مانند هزینه، تأخیر و غیره)، به این دلیل انجام عملیات اصلی ریاضی قبل از بی‌مقیاس کردن یا یکسان‌سازی مقیاس‌ها مجاز نیست. بنابراین طبق رابطه (۱)، هر عنصر R_{ij} از ماتریس تصمیم‌گیری مفروض، بر نرم موجود از ستون jام (به ازای شاخص X_j) تقسیم می‌شود، یعنی:

$$N_{ij}=\frac{R_{ij}}{\sqrt{\sum_{i=1}^{m} R_{ij}^{2}}} \qquad (۱)$$

بدین طریق کلیه ستون‌های ماتریس مفروض دارای واحد طول مشابه شده و درنتیجه مقایسه کلی آن‌ها آسان می‌شود. این ماتریس در شکل ۲ نمایش داده شده است.

شاخص / گزینه	X_1	X_2	....	X_n
A_1	N_{11}	N_{12}	...	N_{1n}
A_2	N_{21}	N_{22}	...	N_{2n}
.	.	.		.
.	.	.		.
.	.	.		.
A_m	N_{m1}	N_{m2}	...	N_{mn}

شکل ۲: ماتریس تصمیم بی‌مقیاس‌شده **D**

برای E_j از مجموعه N_{ij}ها به ازای هر مشخصه خواهیم داشت:

$$E_j=-K\sum_{i=1}^{m}[N_{ij}\ln N_{ij}];\forall j \qquad (۲)$$

به‌طوری‌که $k=\frac{1}{\ln m}$ و لگاریتم در مبنای π است. عدم اطمینان یا درجه انحراف (d_j) از اطلاعات ایجادشده به‌ازای شاخص زام بدین قرار است:

$$d_j=1-E_j;\forall j \qquad (۳)$$

برای اوزان W'_j از شاخص‌های موجود خواهیم داشت:

$$W'_j=\frac{d_j}{\sum_{j=1}^{n} d_j};\forall j \qquad (۴)$$

سرانجام وزن نهایی شاخص‌ها براساس روش ترکیبی برای هر کار به‌صورت زیر محاسبه می‌شود.

همان‌طور که در بخش ۳-۱ اشاره شد، هرکار j_i در فضای ابر محاسباتی دارای ۶ مشخصه است، این مشخصات در قالب بردار J در شکل ۳ نمایش داده شده است. بر اساس میزان اهمیت نسبی‌ای که هر یک از شاخص‌ها برای کاربر دارند، وزنی به آن‌ها تلعق می‌گیرد، به‌طوری‌که مجموع آن‌ها برابر یک شود. این اهمیت نسبی بیانگر درجه ارجحیت هر شاخص نسبت به بقیه شاخص‌ها برای تصمیم‌گیری مورد نظر است. این وزن‌ها از طریق فرمول (۵) و از طریق تلفیق بردار J با بردار W' محاسبه می‌شوند.

شاخص / گزینه	X_1	X_2	X_3	X_4	X_5	X_6
J_i	P_1	P_2	P_3	P_4	P_5	P_6

شکل ۳: بردار **J** برای کار **i** ام

$$W_j=\frac{p_j w'_j}{\sum_{j=1}^{6} p_j w'_j},\ \sum_{j=1}^{k} w_j=1 \qquad (۵)$$

۴-۳- رتبه‌بندی نهایی گزینه‌ها بر اساس تکنیک تاکسونومی غیرکلاسیک

تاکسونومی‌ها فهرست سلسله مراتبی چندگانه‌ای از موضوعات و مقوله‌های موضوعی هستند که رابطه مفهومی بین موضوعات را بیان می‌کنند. به بیان دیگر، تاکسونومی‌ها علم طبقه‌بندی اشیاء هستند که شامل اصول کلی برای تقسیم‌بندی اشیاء و حقایق به رده‌های مختلف هستند و در آن‌ها هر رده اصلی به زیررده و زیررده‌ها به رده‌های فرعی‌تر تقسیم می‌شود. هدف اصلی تاکسونومی، نظام‌مند ساختن مجموعه‌ای از عناصر مختلف در یک ساختار سلسله مراتبی و کمک به بازیابی اطلاعات مرتبط است [۲۲]. در این مقاله از روش تاکسونومی غیرکلاسیک که جزء یکی از روش‌های تاکسونومی است، برای طبقه‌بندی VMها یا سرورها به‌صورت زیر استفاده می‌شود:

فرض کنید A مجموعه‌ای از گزینه‌ها است که باید طبقه‌بندی شوند و از میان آن‌ها انتخاب صورت گیرد. با فرض وجود K معیار مؤثر در تصمیم‌گیری، برای هر $A\in\alpha$ مقدار $f_j(\alpha)$ نشان‌دهنده ارزش معیار زام در گزینه a است. رتبه‌بندی در فازهای زیر انجام می‌شود:

فاز اول (رده‌بندی)

تشخیص گزینه‌های ناهمگن در ۳ مرحله پس از بی‌مقیاس‌سازی ماتریس تصمیم، ماتریس فواصل اقلیدسی بین گزینه‌ها (D') از طریق رابطه (۶) محاسبه می‌شود.

$$D'_{(K,l)}=[\sum(Z_{kj}-Z_{lj})^2]^{1/2} \qquad (۶)$$

پس از محاسبه فواصل اقلیدسی، کوچک‌ترین عدد هر سطر از ماتریس D' نشانگر کوتاه‌ترین فاصله بین گزینه‌ها (d_{i0}) است. با کمک d_{i0}های به‌دست‌آمده $\bar{d}$ (میانگین d_{i0}ها) و S_d (انحراف استاندارد d_{i0}ها) محاسبه می‌شود. و حدود اطمینان بالا (d^+) و پایین (d^-) نیز از طریق روابط (۷) و (۸) محاسبه می‌شود.

$$d^-=\bar{d}-2S_d \qquad (۷)$$

$$d^+=\bar{d}+2S_d \qquad (۸)$$

پس از محاسبه حدود اطمینان، گزینه‌ای که کوچک‌ترین فاصله اقلیدسی آن با سایر گزینه‌ها در حدود اطمینان تعریف‌شده قرار تگیرد خوشه مستقلی ایجاد می‌کند (ایجاد خوشه ناهمگن اول) و با خروج این گزینه به‌عنوان یک گزینه ناهمگن از ماتریس تصمیم‌گیری، دوباره وارد فاز اول می‌شویم.

درواقع مقادیر جدیدی برای میانگین، انحراف استاندارد و حدود اطمینان محاسبه می‌شود و مانند قبل گزینه‌هایی که در حدود اطمینان قرار نگیرند، خوشه مستقل تشکیل داده (ایجاد خوشه ناهمگن دوم) و به‌عنوان گزینه‌های ناهمگن از ماتریس تصمیم حذف می‌شوند. به همین ترتیب خوشه ناهمگن سوم نیز در مرحله بعد ایجاد می‌شود و گزینه‌های باقی‌مانده در ماتریس تصمیم برای طبقه‌بندی وارد فاز دوم می‌شوند.

فاز دوم (رتبه‌بندی)

دراین فاز ابتدا ماتریس بی‌مقیاس موزون (V) از طریق رابطه زیر و با استفاده از مقادیر محاسبه‌شده در روابط (۱) و (۵) به دست می‌آید.

$$V_{ij} = N_{ij} \times W_j \qquad (۹)$$

سپس راه‌حل‌های ایده‌آل مثبت (A^+) و راه‌حل‌های ایده‌آل منفی (A^-) طبق روابط (۱۰) و (۱۱) محاسبه می‌شود.

$$A^- = \{\mathrm{MinV}_{ij} \mid (J \in J^+), (\mathrm{MaxV}_{ij} \mid J \in J^-)\} = (V_1^-, V_2^-, ..., V_n^-) \qquad (۱۰)$$

$$A^+ = \{\mathrm{MaxV}_{ij} \mid (J \in J^+), (\mathrm{MinV}_{ij} \mid J \in J^-)\} = (V_1^+, V_2^+, ..., V_n^+) \qquad (۱۱)$$

پس از این مرحله، فاصله اقلیدسی هر یک از گزینه‌ها از راه‌حل ایده‌آل مثبت (d_i^+) و راه‌حل ایده‌آل منفی (d_i^-) با استفاده از روابط (۱۲) و (۱۳) محاسبه می‌شود.

$$d_i^- = [\sum_{j=1}^{n} (V_{ij} - V_j^-)^2]^{1/2} \qquad (۱۲)$$

$$d_i^+ = [\sum_{j=1}^{n} (V_{ij} - V_j^+)^2]^{1/2} \qquad (۱۳)$$

در مرحله بعد، شاخص نزدیکی نسبی از رابطه (۱۴)، استخراج می‌شود.

$$C_i^+ = \frac{d_i^-}{d_i^- + d_i^+} \qquad (۱۴)$$

نحوه تعریف شاخص نزدیکی نسبی (C_i^+) به‌گونه‌ای است که از نوع مثبت خواهد بود. حال زمان رتبه‌بندی گزینه‌ها است، هر گزینه‌ای که دارای بیش‌ترین مقدار C_i^+ باشد، در اولین رتبه قرار گرفته و بهترین گزینه برای انتخاب است و بقیه گزینه‌ها نیز به ترتیب اندازه C_i^+ در رتبه‌های پایین‌تر قرار می‌گیرند.

۴- الگوریتم پیشنهادی (TLB)

پیش از معرفی الگوریتم به‌کارگرفته‌شده در روش پیشنهادی لازم است، مشخصات آن و همچنین ویژگی‌های محیطی که برای ابر در نظر گرفته شده است، بر اساس مدل 3PCS [۲۳] بررسی شود. در این مقاله از روش موازنه‌گر بار نامتمرکز استفاده شده است که در آن هر سرور مسئول موازنه بار VMهای خود و سرورهای دیگر است. در هر سرور، مدیر زیرساخت مجازی مسئولیت تخصیص منابع به VMها و کارها را دارد، به همین منظور مقداری از منابع سرور همیشه به‌طور کامل در اختیار آن است و مدیر زیرساخت برای انجام محاسبات از آن منابع استفاده می‌کند. مدیر زیرساخت در هر سرور، دو جدول را تنظیم می‌کند که یکی جدول بار و دیگری جدول سرور است. جدول بار جدولی است که میزان بار منابع تمام VMهای ایجادشده بر روی آن سرور و همچنین میزان بار کلی خود سرور در آن قرار دارد. میزان بار هر یک از منابع یک گره پردازشی با استفاده از روابط (۱۵)، (۱۶)، (۱۷) و (۱۸) به دست می‌آید.

$$load\left(cpu_j\right) = \sum_{i=1}^{m} \left(\frac{job\_cpulength(i)^{(MI)}}{computingcapacity\left(\frac{MI}{\sec}\right)}\right) \qquad (۱۵)$$

$$load\left(Mem_j\right) = \sum_{i=1}^{m} \left(\frac{job\_Memlength(i)(MB)}{Memorybandwidth\left(\frac{MB}{\sec}\right)}\right) \qquad (۱۶)$$

$$load\left(IO_j\right) = \sum_{i=1}^{m} \left(\frac{job\_IOlength(i)(MB)}{IObandwidth\left(\frac{MB}{\sec}\right)}\right) \qquad (۱۷)$$

$$load\left(BW_j\right) = \sum_{i=1}^{m} \left(\frac{job\_BWlength(i)(MB)}{Bandwidth\left(\frac{MB}{\sec}\right)}\right) \qquad (۱۸)$$

که در آن j شماره گره پردازشی است. Job_cpulength، Job_Memlength، Job_Iolength و BW_IOlength به ترتیب بیانگر میزان بهره‌برداری هر کار موجود در صف منابع (CPU، حافظه، ورودی/ خروجی) از منابع و پهنای باند گره پردازشی است و برحسب MI، MB و MB می‌باشند. مخرج هر کدام از کسرهای روابط (۱۵)، (۱۶)، (۱۷) و (۱۸) نیز بیانگر سرعت منابع موجود است که برای CPU، حافظه، ورودی/ خروجی و پهنای باند به ترتیب MI/Sec، MB/Sec، MB/Sec و MB/Sec است. مجموع به‌دست‌آمده برای هر یک از منابع موجود بر میزان سرعت آن منبع تقسیم می‌شود، تا بارکاری موجود در صف آن منبع محاسبه شود. اگر این مقدار از مقدار آستانه (T) بیش‌تر باشد به معنای زیر بار بودن آن منبع است، در غیر این‌صورت منبع دارای بار سبک خواهد بود. اگر یکی از منابع یک گره پردازشی زیر بار باشد، به معنای زیر بار بودن کل آن گره است. این مفهوم در الگوریتم ۱ تعریف شده است:

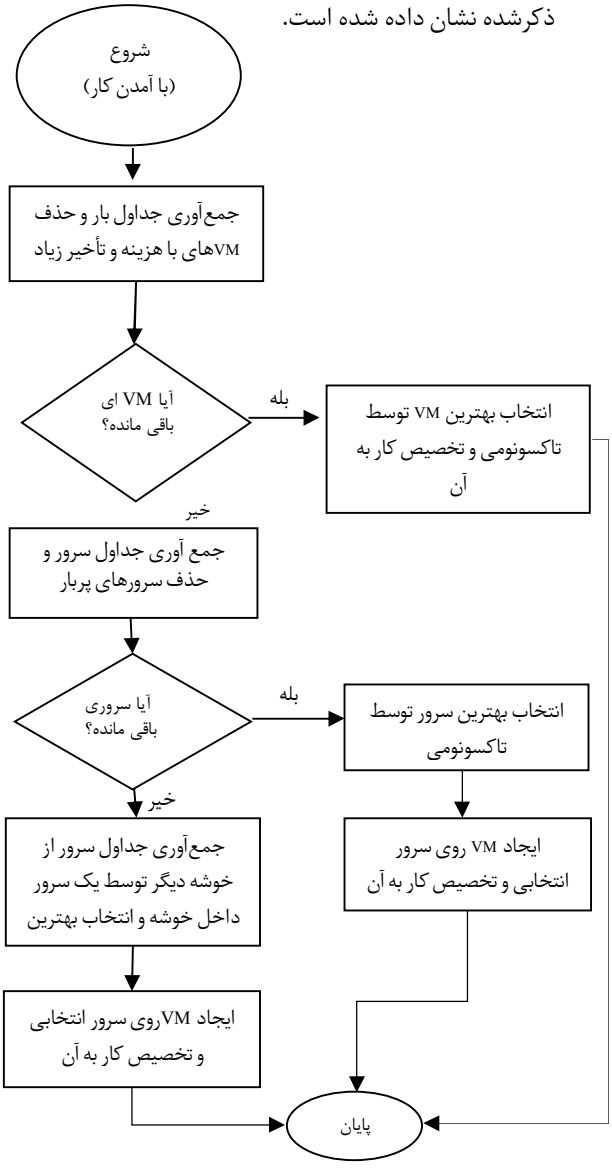
```
If  max(CPU, Mem, I/O, BW) ≥ T
   utilization Level is High
else
   utilization Level is Low
endif
```

الگوریتم ۱: شبه‌کد تعیین حالت گره پردازشی

جدول دوم جدول سرور است. در این جدول اطلاعات سرورهای ارائه‌کننده یک سرویس وجود دارد، با اضافه شدن هر سرور فیزیکی به ابر، این جدول به‌روز می‌شود، تا اطلاعات سرور اضافه‌شده که شامل تعداد سرویس‌هایی که توسط سرور ارائه می‌شود و قیمت سرور، است در میان سایر سرورها پخش شود. مراحل موازنه بار توسط الگوریتم TLB در دو سطح بیان می‌شود.

الف) سطح **VM**: زمانی که یک کار به سرور (سرور مبدأ) می‌رسد، مدیر زیرساخت مجازی سرور مبدأ با توجه به جدول سرور، فراخوانی را برای جمع‌آوری جداول بار سایر سرورهای درون خوشه‌ای می‌دهد. منظور از سرورهای درون خوشه‌ای، سرورهایی هستند که توسط تعدادی VM سرویس موردنظر را ارائه می‌دهند. سرورها با دریافت این فراخوان، ابتدا اطلاعات آن بخشی از جدول بار که VMهای ارائه‌دهنده سرویس در آن وجود دارد را ویرایش کرده، سپس برای سرور مبدأ ارسال می‌کنند. این ویرایش به این صورت است که تمام VMهایی که زیر بار باشند را از جدول حذف می‌کند و فقط آن دسته از VMها که بار سبک دارند (بار آن‌ها کم‌تر از T باشد) را به سرور مبدأ ارسال می‌کند. سرور مبدأ با دریافت تمامی جداول بار ویرایش‌شده و همچنین اندازه‌گیری میانگین زمان تأخیر برای هر سرور، ماتریس تصمیم‌گیری تاکسونومی غیرکلاسیک را تشکیل می‌دهد. سپس با توجه به مشخصه‌های میزان بهره‌برداری از CPU، حافظه و ورودی/ خروجی اقدام به تولید بردار وزن (رابطه (۷)) می‌کند و بهترین VM را برای تخصیص به کار ورودی مشخص می‌کند.

ب) سطح سرور فیزیکی: پس از ورود کار و فراخوانی سرور مبدأ مبنی بر جمع آوری جداول بار اگر هیچ VMای پیدا نشود، به‌عبارت دیگر اگر جداول بار رسیده به سرور مبدأ خالی باشند به‌معنای زیر بار بودن تمامی VMهای موجود در خوشه است. در این زمان کل خوشه زیر بار است. برای رفع مشکل باید ظرفیت‌های پردازشی (VM) جدید در خوشه ایجاد شود، تا قسمتی از بار خوشه به آن انتقال پیدا کند. برای این کار سرور مبدأ اقدام به بررسی میزان بار هر کدام از سرورها در جدول‌های باری دریافتی می‌کند. سرورهایی که دارای ظرفیت خالی باشند موظف به ایجاد VM می‌شوند. درصورتی‌که سروری برای ایجاد VM پیدا نشد سرورهای موجود در خوشه، اقدام به مهاجرت سایر VMهای خود (از سایر خوشه‌ها) به سرورهای آن خوشه‌ها می‌کنند. برای این کار، سرورها نیاز به انتخاب بهترین سرور برای مهاجرت سایر VMهای خود به آن را دارند؛ بنابراین با توجه به الگوریتم پیشنهادی اقدام به انتخاب بهترین سرور می‌کند. در این‌جا ماتریس تصمیم‌گیری شامل میزان بار هر کدام از منابع سرورها (CPU، حافظه، ورودی/ خروجی)، زمان تأخیر و قیمت هرکدام از آن‌ها است. با توجه به میزان بار هر کدام از منابع VM مهاجر، اقدام به وزن‌دهی به هرکدام از مشخصه‌های فوق می‌شود. بدین ترتیب سرورهای داخل خوشه ظرفیت جدید برای ایجاد VM داخل خوشه‌ای پیدا می‌کنند و با ایجاد آن می‌توانند از میزان بار موجود در خوشه بکاهند. در شکل ۴ موازنه بار در دو سطح ذکرشده نشان داده شده است.



شکل ۴: موازنه بار توسط الگوریتم پیشنهادی در دو سطح

۴-۱- مثال تشریحی

برای فهم بهتر روش پیشنهادی مثال زیر را بررسی می‌کنیم:

فرض می‌کنیم ۵ ماشین مجازی با مشخصات جدول ۱ داریم. ابتدا مقادیر این جدول بی‌مقیاس می‌شود. سپس فرض می‌کنیم بردار J برای کار موردنظر به‌صورت جدول ۲ باشد.

وزن محاسبه‌شده برای شاخص‌ها از طریق روش ذکرشده در قسمت ۳-۳ به صورت زیر است، به این صورت که ارزش یا وزن هزینه، تأخیر، میزان بهره‌برداری از I/O، حافظه و CPU و پهنای باند به ترتیب برابر ۰/۰۹، ۰/۱۷، ۰/۱۵، ۰/۲، ۰/۳۳، ۰/۰۷ است، و بردار وزن به‌صورت زیر است:

$W = \{۰/۰۹, ۰/۱۷, ۰/۱۵, ۰/۲, ۰/۳۳, ۰/۰۷\}$

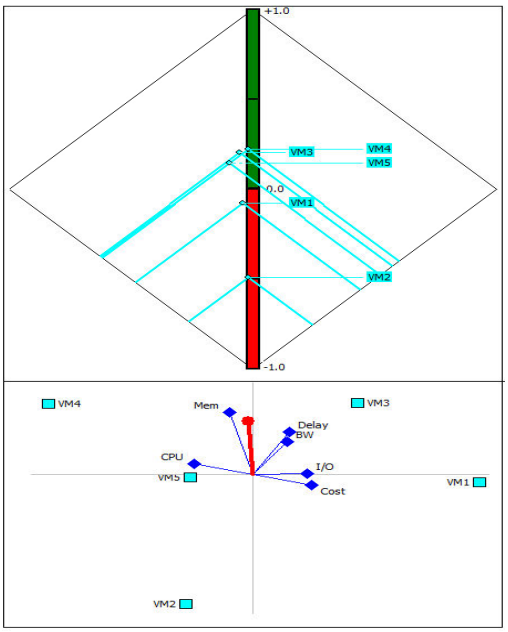
فاصله هر یک از VMها از شاخص‌ها در فضای سه‌بعدی در شکل ۵ نشان داده شده است. VMهای شماره ۲ و شماره ۴ به ترتیب بیش‌ترین و کم‌ترین فاصله را با شاخص‌ها دارند. رتبه‌بندی نهایی VMها بر اساس روش پیشنهادی به‌صورت شکل ۶ است. در قسمت بعد نشان می‌دهیم با شبیه‌سازی این روند در شبیه‌ساز Cloudsim به چه نتایجی دست یافته‌ایم.

جدول ۱: مشخصات **VM**ها

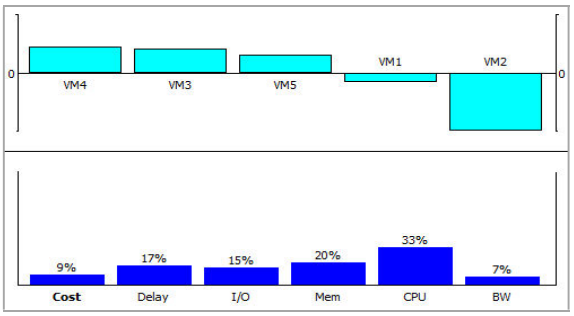
شاخص / گزینه	هزینه (دلار)	تأخیر (ثانیه)	میزان بهره‌برداری از ورودی/خروجی	میزان بهره‌برداری از حافظه	میزان بهره‌برداری از CPU	پهنای باند (MB\S)
VM1	۰/۸	۳	٪۶۵	٪۴۰	٪۳۵	۳۰۰
VM2	۰/۳	۱	٪۴۵	٪۲۵	٪۶۰	۱۰۰
VM3	۰/۷	۴	٪۵۰	٪۵۵	٪۴۵	۲۰۰
VM4	۰/۱	۲	٪۳۱	٪۶۰	٪۷۰	۲۰۰
VM5	۰/۲	۳	٪۵۰	٪۴۳	٪۶۵	۱۰۰

جدول ۲: بردار **J** برای کار **i**ام

شاخص / کار مورد نظر	میزان بهره‌برداری از CPU	میزان بهره‌برداری از حافظه	میزان بهره‌برداری از ورودی/خروجی	سررسید (ثانیه)	بودجه (دلار)	پهنای باند(MB\S)
j_i	۰/۰۹	۰/۱۷	۰/۱۵	۰/۲	۰/۳۳	۰/۰۷



شکل ۵: طبقه‌بندی **VM**ها طبق فواصل آن‌ها از شاخص‌ها برای مثال تشریحی



شکل ۶: رتبه‌بندی نهایی **VM**ها بر اساس روش تاکسونومی برای مثال تشریحی

۵- نتایج شبیه‌سازی

برای انجام شبیه‌سازی در محیط ابر نیاز به اطلاعاتی در مورد ظرفیت سرورهای فیزیکی و VMها است. برای تعیین این مقادیر از دو نوع مرجع استفاده شده است که نوع اول منابع موجود در مراجع [۲۴، ۲۵] است که در پردازش توزیع‌شده مورد استفاده قرار گرفته‌اند. نوع دوم، منابع مورد استفاده در ابزار CloudSim است.

این ابزار توسط آزمایشگاه CLOUDS[۸] در دانشگاه ملبورن[۹] توسعه داده شده و محیطی برای شبیه‌سازی پردازش ابری است. مثال‌های موجود در این ابزار نشان‌دهنده میزان قدرت پردازشی CPU، حافظه و ورودی / خروجی است که در جدول ۳ به آن‌ها اشاره شده است.

جدول ۳: مشخصات منابع

منبع	دستگاه	خصوصیات منبع: نوع منبع، سیستم عامل، تعداد اجزاء محاسباتی	نرخ سرعت اجزاء محاسباتی SPEC/MIPS	قیمت (G$/PE unit)
R1	M1	Compaq, AlphaSrever, CPU, OSFI, 4	۵۱۵	۸
R2	M2	Sun, Ultra, Solaris, 4	۳۷۷	۳
	M3	Sun, Ultra, Solaris, 4	۳۷۷	۳
	M4	Sun, Ultra, Solaris, 4	۳۷۷	۳
R3	M5	Intel, Pentiun/VC820, Linux, 2	۳۸۰	۳
R4	M6	SGI, Origin 3200, IRIX, 6	۴۱۰	۳
	M7	SGI, Origin 3200, IRIX, 16	۴۱۰	۳
R5	M8	SGI, Origin 3200, IRIX, 6	۴۱۰	۴
R6	M9	Intel, Pentiun/VC820, Linux, 2	۳۸۰	۱
R7	M10	SGI, Origin 3200, IRIX, 4	۴۱۰	۶
R8	M11	Sun, Ultra, Solaris, 8	۳۷۷	۳

در این شبیه‌سازی، ۵۰ عدد سرور در نظر گرفته شده است که با توجه به جدول ۴، ظرفیت منابع هر سرور با استفاده از توزیع نرمال و با مشخصات ذکرشده در جدول ۵ به دست آمده است.

جدول ۴: مشخصات منابع در ابزار **CloudSim**

شماره مثال	شماره میزبان (بسم)	مشخصات میزبان			مشخصات VM(ها) روی میزبان			قیمت میزبان ($G)
		CPU (MIPS)	Mem. (MB)	I/O (B/Sec)	CPU (MIPS)	Mem. (MB)	I/O (B/Sec)	
۱	۰	۱۰۰۰	۲۰۴۸	۱۰۰۰۰	۱۰۰۰	۵۱۲	۱۰۰۰	۳
۲	۰	۱۰۰۰	۲۰۴۸	۱۰۰۰۰	۲۵۰	۵۱۲	۱۰۰۰	۳
					۲۵۰	۵۱۲	۱۰۰۰	
۳	۰	۱۰۰۰	۲۰۴۸	۱۰۰۰۰	۲۵۰	۲۰۴۸	۱۰۰۰	۵
	۱	۱۰۰۰	۲۰۴۸	۱۰۰۰۰	۵۰۰	۲۰۴۸	۱۰۰۰	

جدول ۵: مشخصات سرورهای استفاده‌شده در شبیه‌سازی

تعداد سرورها	انتخاب نوع سرویس‌های ارائه شده توسط سرور	ظرفیت سرور				قیمت ($G)		تأخیر (Sec)	
		CPU (MIPS)		Mem. (MB)		I/O (B/Sec)			
		μ	σ۲	μ	σ۲	μ	σ۲	μ	σ۲
۵۰	تصادفی	۵۰۰۰	۵۰۰	۴۰	۱۰	۱۰۰۰۰	۱۰۰۰	۵	۲	۵۰۰	۱۰۰

تعداد سرویس‌های در نظرگرفته‌شده که توسط ابر ارائه می‌شوند برابر ۱۰ است، که هرکدام از این سرویس‌ها با یکدیگر متفاوت می‌باشند یعنی میزان مصرف منابع پردازشی هرکدام از آن‌ها با هم فرق دارد که این مستلزم در نظر گرفتن میزان متفاوت برای میانگین (μ) و انحراف‌معیار (σ۲) در توزیع نرمال برای تولید VMهای سرویس‌ها است [۲۶]. درنتیجه روی هر سرور، گروه VMهای متفاوت با منابع مختلف قرار خواهد گرفت که میزان میانگین و انحراف‌معیار هر سرویس نیز در جدول ۶ نشان داده شده است. بعد از توزیع VMها بر روی سرورها، به‌طور میانگین ۵۱۴ عدد VM به دست می‌آید.

میزان بهره‌برداری از منابع برای هر کار توسط تابع توزیع یکنواخت به دست آمده است. برای هر کار، میزان بهره‌برداری از CPU در بازه (۴۰-۲۰) (MIPS)، بهره‌برداری از حافظه در بازه (۰/۵-۰/۰۵) و بهره‌برداری از ورودی/خروجی در بازه (۸۰-۶۰) در نظر گرفته شده است. همان‌طور که از جدول ۶ پیداست، کارهای ارائه‌شده به محیط ابر نسبت به منابع VM دارای تناسبی در بازه (۰/۱۲۵-۰/۱) هستند. تعداد کل کارهای ارسالی به ابر نیز برابر ۱۰۰۰۰۰ در نظر گرفته شده است. میزان بودجه و سررسید هر کار توسط تابع توزیع نرمال به‌دست آمده که میانگین و انحراف‌معیار آن در جدول ۷ نشان داده شده است.

حال عملکرد الگوریتم پیشنهادی را با استفاده از نتایجی که از شبیه‌ساز Clousim به‌دست آوردیم، تحلیل می‌کنیم. با توسعه و تغییر کلاس‌های شبیه‌ساز Cloudsim الگوریتم پیشنهادی خود را شبیه‌سازی نمودیم.

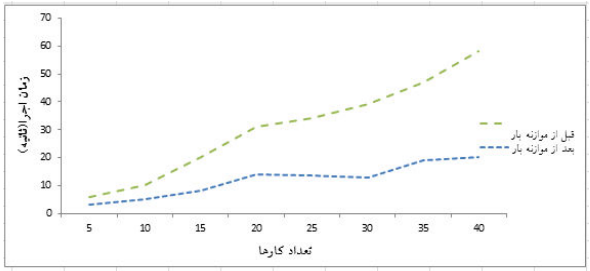
جدول ۶: مشخصات **VM**های استفاده شده در هر سرویس

سرویس	ظرفیت VM					
	CPU (MIPS)		Mem (MB/S)		I/O (B/S)	
	μ	σ۲	μ	σ۲	μ	σ۲
۱	۲۰۰	۵۰	۲	۰/۵	۵۰۰	۵۰
۲	۲۵۰	۴۰	۳	۰/۵	۴۰۰	۵۰
۳	۳۰۰	۵۰	۲	۰/۵	۶۰۰	۴۰
۴	۲۷۰	۳۰	۳	۰/۵	۵۰۰	۵۰
۵	۳۲۵	۲۰	۲	۰/۵	۴۵۰	۵۰
۶	۳۵۰	۱۰	۳	۰/۵	۵۵۰	۵۰
۷	۲۰۰	۶۰	۲	۰/۴	۴۷۰	۴۰
۸	۳۷۰	۲۰	۲	۰/۳	۵۰۰	۳۰
۹	۴۰۰	۱۰	۲	۰/۴	۵۵۰	۴۰
۱۰	۳۰۰	۲۰	۲	۰/۵	۶۰۰	۲۰

جدول ۷: مشخصات کارهای ارسالی به ابر

تعداد کارها	بودجه ($G)		سررسید (Sec)	
	μ	σ۲	μ	σ۲
۱۰۰۰۰۰	۵	۲	۶۰۰	۱۰۰

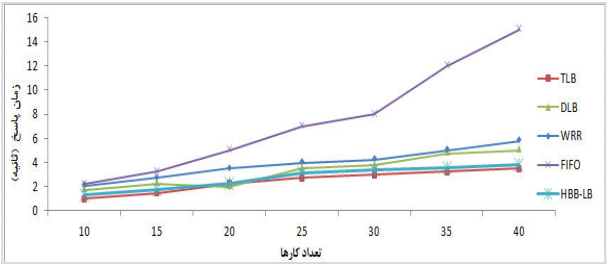
در شکل ۷ زمان اجرا، قبل و بعد از پیاده‌سازی روش پیشنهادی آورده شده است. محور X تعداد کارها و محور Y زمان اجرا نشان می‌دهد. همان‌طور که در این شکل می‌بینیم زمان اجرا پس از پیاده‌سازی روش پیشنهادی به‌طور قابل‌ملاحظه‌ای کاهش یافته است. همچنین قبل از اجرای موازنه بار با افزایش تعداد کارها زمان اجرا به‌میزان زیادی افزایش می‌یابد درحالی‌که پس از اجرای موازنه بار با افزایش تعداد کارها زمان اجرا رشد کمی دارد.



شکل ۷: زمان اجرا قبل و بعد از انجام موازنه بار با تاکسونومی

در شکل ۸ زمان پاسخ الگوریتم پیشنهادی در مقایسه با روش‌های WRR، FIFO، HBB-LB و DLB [۱۰، ۱۸، ۲۷، ۲۸، ۲۹] نشان داده شده است. در این شبیه‌سازی دنبال این فرضیه هستیم که افزایش بار

کاری موجب افزایش زمان پاسخ می‌شود. رشد غیرهماهنگ و سریع بار موجود در صف منابع در روش‌های WRR، FIFO و DLB باعث می‌شود که گره‌های پردازشی زیر فشار بار روند؛ بنابراین کارها برای اجرا باید مدت زیادی منتظر بمانند. روش پیشنهادی (TLB) و بعد از آن روش HBB-LB دارای کم‌ترین زمان پاسخ در بین الگوریتم‌های فوق هستند، زیرا در روش پیشنهادی، میزان بهره‌برداری کار از منابع در نظر گرفته شده است که باعث می‌شود، یک کار به مناسب‌ترین منبع محاسباتی تخصیص یابد؛ بنابراین بار منابع محاسباتی موجود در گره‌های پردازشی به‌طور هماهنگ رشد کرده و دیرتر به حالت زیر فشار بار می‌رسند؛ در نتیجه همان‌طور که در شکل ۸ مشاهده می کنیم، زمان انتظار و به‌تبع آن زمان پاسخ کارها تا حد زیادی کاهش می‌یابد.



شکل ۸: زمان پاسخ روش‌های **TLB**، **FIFO**، **DLB**، **WRR** و **HBB-LB**

برای مقایسه عملکرد روش TLB در مقایسه با بقیه روش‌ها معیار انحراف بار در رابطه (۱۹) معرفی شده است که آن با α نمایش داده شده است [۲۹ و ۳۰].

$$L_{VM_i,t} = \frac{N(T,t)}{S\left(VM_i,t\right)} \qquad (۱۹)$$

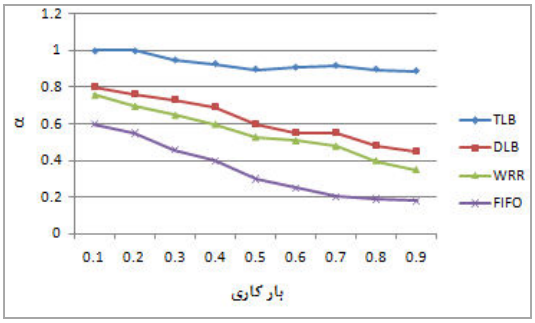
در رابطه بالا $N(T,t)$ تعداد کارهای موجود در صف VM_i مربوطه در زمان t و $S(VM_i,t)$ نرخ سرویس‌دهی VM_i در زمان t است؛ بنابراین α به‌صورت رابطه (۲۰) محاسبه می‌شود [۳۰].

$$\alpha = \frac{L_{VM_i,t_0} - L_{VM_i,t}}{L_{VM_i,t_0}} \qquad (۲۰)$$

که در آن $LVM_{i,t0}$ بار اولیه VM_iدر لحظه t_0 و $LVM_{i,t}$ بار این VM در لحظه فعلی (t) است. این پارامتر، معیار خوبی برای تحلیل تأثیر الگوریتم موازنه بار بر روی سیستم و میزان مهاجرت VMها است. مقدار α در بازه (۱-۰) تغییر می‌کند. اگر α برابر یک باشد، به این معنی است که گره پردازشی موردنظر در لحظه t تمام کارهای موجود در صف خود را اجرا نموده است و نرخ سرویس‌دهی بالایی دارد.

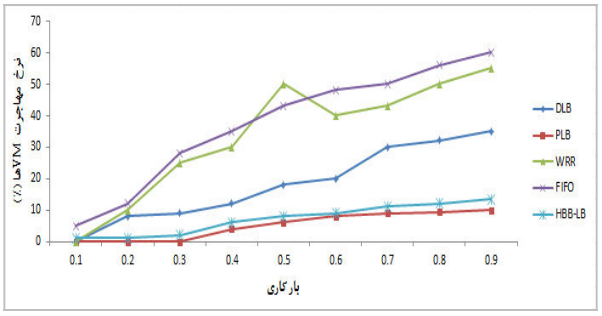
در شکل ۹ میزان α به ازای تغییر بار کاری نشان داده شده است. مفهوم بار کاری عبارت است از نسبت کل بار ارسال شده به ابر به کارهایی که در طول دوره شبیه‌سازی امکان اجرای آن‌ها را داشته است. فرضیه‌ای که در این بخش در نظر گرفته شده است، این است که افزایش بار کاری موجب افزایش شاخص α می‌شود. همان‌طور که در شکل مشاهده می‌شود، روش پیشنهادی دارای بیش‌ترین و روش FIFO دارای کم‌ترین مقدار α است. در نمودار روش پیشنهادی مقدار α همواره بیشینه است، زیرا در این روش همواره مناسب‌ترین VM برای کار موردنظر انتخاب می‌شود و کار موردنظر در حداقل زمان ممکن اجرا می‌شود.

بنابراین همیشه VMها دارای صف‌های کوتاه یا بدون صف هستند و صف مربوطه در مدت‌زمان کمی خالی می‌شود. روش HBB-LB رفتاری نزدیک به روش پیشنهادی دارد. اما در روش‌های DLB و FIFO و WRR ازآنجایی‌که همه شاخص‌ها برای انتخاب VM موردنظر لحاظ نمی‌شوند، کارهایی که در صف VMها قرار می‌گیرند تناسب زیادی با توانایی پردازشی VMها ندارند، بنابراین صف مربوط به VMها دیرتر خالی می‌شود و مقدار α تنها زمانی که بار کاری سبک است، تا حدی مطلوب است.



شکل ۹: شاخص **α** برای روش‌های **TLB**، **FIFO**، **DLB**، **WRR**

در برخی موارد خاص، افزایش شدید بار بعضی از VMها در سیستم به‌خاطر دسترسی‌های مکرر به آن‌ها باعث نامتعادل شدن کل سیستم می‌شود و در این حالت باید چند VM به سرورهای دیگر مهاجرت کنند و نمی‌توان هزینه این مهاجرت‌ها را نادیده گرفت. بیشینه بودن مقدار α باعث حداقل شدن نرخ مهاجرت می‌گردد. همان‌طور که در شکل ۱۰ مشاهده می‌کنیم، میانگین نرخ مهاجرت VMها در روش DLB به طرز قابل‌توجهی نسبت به روش‌های دیگر کم‌تر است.

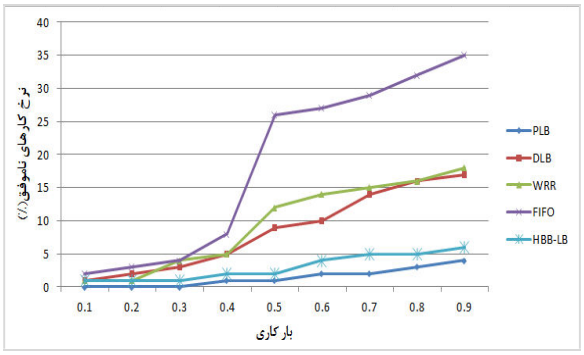


شکل ۱۰: نرخ مهاجرت **VM**ها برای روش‌های **PLB**، **FIFO**، **DLB** ، **WRR** و **HBB-LB**

در شکل ۱۱ نرخ عدم موفقیت کارها برای ۴ الگوریتم موردنظر نشان داده شده است. فرضیه ما در این بخش این بوده است که افزایش بار کاری بر نرخ عدم موفقیت کارها تأثیر می‌گذارد. همان‌طور که می‌بینیم در بارهای کم، این نرخ برای هر پنج الگوریتم کم و تقریباً برابر است، ولی وقتی بار بیش از ۰/۵ می‌شود، این نرخ برای روش‌های

FIFO، HBB-LB، WRR و DLB علی‌الخصوص روش FIFO به‌شدت افزایش یافته است.

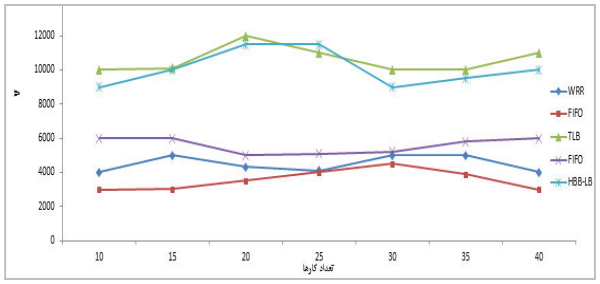
یکی از دلایل این موضوع به دلیل این است که VMهای موجود زیر فشار بار رفتند و FIFO مجبور به ایجاد VMهای جدید برای انجام دادن پردازش‌های خود است که این امر باعث از بین رفتن کارهای بعدی شده است. دلیل دیگر این است که در FIFO فقط زمان ورود کارها به ابر در نظر گرفته می‌شود؛ بنابراین کارها به سرعت به منابع نامتناسب با نیازهایشان فرستاده می‌شوند و با افزایش بار کاری VMها به سرعت پر می‌شوند، بنابراین کارهای بعدی ارسالی به ابر از بین خواهند رفت. برای الگوریتم WRR نیز وضع تقریباً به‌صورت مشابه است. در روش پیشنهادی چون تمام معیارها برای فرستادن کار مربوطه به گره پردازشی موردنظر، در نظر گرفته شده است، نرخ کارهای ناموفق کاهش چشم‌گیری دارد.



شکل ۱۱: نرخ عدم موفقیت کارها برای روش‌های **TLB**، **FIFO**، **DLB**، **WRR** و **HBB-LB**

ووود و همکارانش در [۳۱] یک معیار جامع برای موازنه بار ارائه دادند که به‌صورت رابطه ۲۱ است، که در آن CPU_U، Mem_U، I/O_U و BW_U به ترتیب میانگین بهره‌وری CPU، حافظه، ورودی / خروجی و پهنای باند شبکه در طول دوره شبیه‌سازی است. بالاترین مقدار V، نشان دهنده بالاترین مقدار بهره‌وری است. همان‌طور که در شکل ۱۲ مشخص است روش پیشنهادی دارای بالاترین مقدار V نسبت به روش‌های دیگر است.

$$V = \frac{1}{(1-CPU_U)(1-Mem_U)(1-I/O_U)(1-BW_U)} \qquad (۲۱)$$



شکل ۱۲: شاخص **V** برای روش‌های **TLB**، **FIFO**، **DLB**، **WRR** و **HBB-LB**

۶- نتیجه

در این مقاله، یک روش موازنه بار نامتمرکز بر پایه روش تصمیم‌گیری تاکسونومی غیرکلاسیک و روش وزن‌دهی آنتروپی برای محیط ناهمگن ابر ارائه شد. در این الگوریتم، انتخاب VM مناسب برای کار موردنظر بر اساس تمام معیارهای کمی و کیفی و بر پایه تمام نیازهای کاربر انجام شد. روش پیشنهادی با روش‌های دیگر مقایسه شد و نشان داده شد که این روش با کم‌ترین زمان بیکاری VMها محیط‌های ناهمگن را به‌خوبی مدیریت می‌کند و دارای کم‌ترین نرخ مهاجرت VMها و کم‌ترین نرخ خرابی کارها است و در کم‌ترین زمان ممکن وظایف را اجرا می‌کند. همچنین چون هر سرور، مسئول موازنه بار VMهای خود و سرورهای دیگر است، مشکل روش‌های متمرکز و از کار افتادن کل سیستم با یک نقطه از خرابی، کاملاً مرتفع شده است.

مراجع

[۱] شهرام جمالی، سپیده ملک‌تاجی و مرتضی آنالویی «مکان‌یابی ماشین‌های مجازی با استفاده از الگوریتم رقابت استعماری»، مجله مهندسی برق دانشگاه تبریز، جلد ۴۶ شماره ۱ صفحه ۶۲-۵۳.

[۲] سیدهادی اقدسی و مقصود عباس‌پور، «الگوریتم توزیع‌شده جهت فراهم آوردن پوشش چندجانبه از هدف در شبکه‌های حسگر بصری»، مجله مهندسی برق دانشگاه تبریز، جلد ۴۲، شماره ۲، صفحه ۶۳-۵۳.

[3] B. Rajkumar, Ch. Yeoa and S. Venugopala, "Market Oriented Cloud Computing: Vision, Hype, and Reality for Delivering IT Services as Computing Utilities," in *High Performance Computing and Communications, 10th IEEE International Conference on*, pp. 5-13, 2008.

[4] J. G. Aguado, J. M. Alcaraz Calero and W. D. Villanueva, "IaaSMon: Framework for Monitoring Cloud Computing Datacenters," *Jounal of Grid Computing*, vol. 14, no. 2, pp. 283–297, 2016.

[5] L. M Vaquero, L. R. Merino, J. Caceres and M. Lindner, "A Break in the Clouds: Towards a Cloud Definition," *ACM SIGCOMM Computer Communication Review*, vol. 39, no. 1, pp. 50-55, 2009.

[6] M. Jose, A. Calero and J.G. Aguado, "Comparative analysis of architectures for monitoring cloud computing infrastructures," *Future Generation Computer Systems*, vol. 47, pp.16-30, 2015.

[7] S. T. Maguluri, R. Srikant and L. Ying, "Heavy traffic optimal resource allocation algorithms for cloud computing clusters," *Performance Evaluation*, vol.81, pp. 20-39, 2014.

[8] A. Singh, D. Juneja and M. Malhotra, "Autonomous Agent Based Load Balancing Algorithm in Cloud Computing," *Procedia Computer Science*, vol. 45, pp. 832-841, 2015.

[9] K. Sunny, Kh. Shivani, "Analysis of different Scheduling Algorithms under Cloud Computing," *International Journal of Computer Science and Information Technologies*, vol. 5, no. 2, pp.2592-2595, 2012.

[10] B. Yagoubi and Y. Slimani, "Dynamic load balancing strategy for grid computing, transactions on engineering," *Computing and Technology*, pp. 260–265, 2006.

learning-innovative approach," *International Journal of Computer Applications*, vol. 8, no. 10, pp. 975–8887, 2010.
[28] L. D. Dhinesh Babu, P. Venkata Krishna, "Honey bee behavior inspired load balancing of tasks in cloud computing environ- ments," Applied Soft Computing, vol. 13, no. 5, pp. 2292–2303, 2013.
[29] P. P. Bonissone and K. S. Decker, "Selecting uncertainly calculi and granularity: An experiment in trading off precision and complexity," in *kanal and lemmer*, pp. 217-247, 1986.
[30] H. Guowei, W. Gongyi and Ch. Zhi, "A Fair Load Balancing Algorithm for Hypercube-Based DHT Networks," *APWeb/WAIM 2007, LNCS 4505*, pp. 116–126, 2007.
[31] T. Wood, P. Shenoy, A. Venkataramani and M. Yousif, "Black-box and gray-box strategies for virtual machine migration," *Proceedings of symposium on networked systems design and implementation (NSDI)*, pp. 1-14, 2007.

زیرنویس‌ها

۱ Starvation
۲ Minimum Expected Completion Times(MECT)
۳ Online
۴ Sites
۵ Four -Level Tree
۶ Virtual Node
۷ Cluster
۸ The Cloud computing and Distributed Systems (CLOUDS)
۹ Melbourne

[11] M. Maheswaran, S. Ali, H. J. Siegel. Hensgen and D. Freund, "Dynamic Matching and Scheduling of a Class of Independent Tasks onto Heterogeneous Computing Systems," in *Proceedings of the 8th Heterogeneous Computing Workshop*, pp. 30-44, 1999.
[12] M. Randles, A. Taleb-Bendiab and D. Lamb, "Scalable self governance using service communities as ambients," in *Proceedings of the IEEE Workshop on Software and Services Maintenance and Management (SSMM 2009) within the 4th IEEE Congress on Services*, pp. 813-820, 2009.
[13] I. DeFalco, R. DeBalio, E. Tarantino and R. Vaccaro, "Improving Search by Incorporating Evolution Principles in Parallel Tabu Search," *IEEE Conference on Evolutionary Computation*, vol. 2, pp. 823-828, 1994.
[14] M. Shoukat, M. Maheswaran, H. Siegel, D. Hensgen and R. Freund, "Dynamic Mapping of a Class of Independent Tasks onto Heterogeneous Computing Systems," *Journal of Parallel and Distributed Computing*, pp. 107–131, 2010.
[15] A. Singh, M. Korupolu and D. Mohapatra, "Server-storage virtualization: Integration and load balancing in data centers," *Proceedings of ACM/IEEE Conference on Supercomputing*, pp. 1-12, 2008.
[16] A. S. Shoukat, *Robust Resource Allocation in Dynamic Distributed Heterogeneous Computing Systems*, Ph.D. Thesis in School of Electrical and Computer Engineering, Purdue University, 2012.
[17] B. Yagoubi and Y. Slimani, "Dynamic load balancing strategy for grid computing," *Transactions on Engineering, Computing and Technology*, pp. 260–265, 2013.
[18] B. Yagoubi and Y. Slimani, "Task load balancing strategy for grid computing," *Journal of Computer Science*, vol. 3, no. 3, pp.186–194, 2007.
[19] Y. Adil, A. Abdul-hanan and M. Suliman, "Scheduling Jobs on Grid Computing Using Firefly Algorithm," *Journal of Theoretical and Applied Information Technology*, vol. 33, no. 2, pp. 155-164, 2011.
[20] J. J. Wang, Ch. F. Zhang, Y-Y. Jing and G-Zh. Zheng, "Using the fuzzy multi-criteria model to select the optimal cool storage system for air conditioning," *Energy and Buildings 40*, pp. 2059–2066, 2008.
[21] Z. Zhi-hong, Y. Yi and S. Jing-nan, "Entropy method for determination of weight of evaluating in fuzzy synthetic evaluation for water quality assessment," *Journal of environ mental science*, vol. 18, pp. 1020-1023, 2006.
[22] E. E, Karsaka, M. Dursun, "Taxonomy and review of non-deterministic analytical methods for supplier selection," *International Journal of Computer Integrated Manufacturing*, pp. 263-286, 2015.
[23] T. D. Braun, et al., "Characterizing Resource Allocation Heuristics for Heterogeneous Computing Systems," *Proceeding of Advances in Computers in Parallel, Distributed, and Pervasive Computing*, pp. 91-128, 2005.
[24] C. Dumitrescu and I. Foster, "Gangsim: A simulator for grid scheduling studies," *Proceedings of the IEEE International Symposium on Cluster Computing and the Grid*, pp. 26-34, 2005.
[25] A. Sulistio and R. Buyya, "A grid simulation infrastructure supporting advance reservation," *Proceedings of the 16th International Conference on Parallel and Distributed Computing Systems*, pp. 1-7, 2004.
[26] A. Nahir, A. Orda and D. Raz, "Distributed Oblivious Load Balancing Using Prioritized Job Replication," *Proceeding of Network and Service Management (CNSM)*, pp. 55-63, 2012.
[27] A. Revar, M. Andhariya, D. Sutariya and M. Bhavsar, "Load balancing in grid environment using machine