مکان‌یابی ماشین‌های مجازی با استفاده از الگوریتم رقابت استعماری

شهرام جمالی[۱]، دانشیار؛ سپیده ملک‌تاجی[۲]، دانشجوی کارشناسی ارشد؛ مرتضی آنالویی[۳]، دانشیار

۱- دانشکده فنی‌مهندسی - دانشگاه محقق اردبیلی - اردبیل - ایران - jamali@iust.ac.ir
۲- دانشکده فنی‌مهندسی - دانشگاه محقق اردبیلی - اردبیل - ایران - sepideh.malektaji@gmail.com
۳- دانشکده مهندسی کامپیوتر - دانشگاه علم و صنعت - تهران - ایران - analoui@iust.ac.ir

چکیده: مجازی‌سازی و استفاده از ماشین‌های مجازی اساس تکنولوژی پردازش ابری است. ماشین‌های مجازی پس از مکان‌یابی، بر روی ماشین‌های فیزیکی منتخب اجرا می‌شوند. منظور از مکان‌یابی، انتخاب میزبان مناسب برای ماشین‌های مجازی موجود است. مکان‌یابی ماشین‌های مجازی در میزان مصرف انرژی و جلوگیری از هدر رفتن منابع در بسترهای سخت‌افزاری، نقش اساسی دارند. از طرفی، توسعه روزافزون سیستم‌های ابری فرآیند مکان‌یابی ماشین‌های مجازی را پیچیده‌تر ساخته است. در پژوهش حاضر با در نظر گرفتن دو هدف کاهش مصرف انرژی و کاهش اتلاف منابع، مکان‌یابی کارا ارائه شده است. در روش پیشنهادی، ضمن استفاده از روش متداول تبدیل مسئله مکان‌یابی به مسئله بهینه‌سازی، از الگوریتم نوظهور رقابت استعماری استفاده شده و با معرفی مفهومی جدید بنام نماینده، روند تولید جواب‌های جدید و بررسی فضای جستجو هدفمند شده است. نتایج شبیه‌سازی نشان می‌دهد، انتخاب الگوریتم رقابت استعماری در حل مسئله مکان‌یابی ماشین‌های مجازی، با معرفی و استفاده از مفهوم نماینده، همراه با موفقیت بوده و الگوریتم پیشنهادی در مقایسه با الگوریتم‌هایی چون GGA و FFD پاسخ‌های قابل قبولی را ارائه می‌کند.

واژه‌های کلیدی: پردازش ابری، مکان‌یابی ماشین‌های مجازی، الگوریتم رقابت استعماری، بهینه‌سازی مصرف انرژی.

Virtual Machine Placement using Imperialist Competitive Algorithm

Shahram Jamali[1], Associate Professor; Sepideh Malektaji[2], MSc Student; Morteza Analoui[3], Associate Professor

1- Faculty of Engineering, University of Mohaghegh Ardabili, Ardabil, Iran, Email: jamali@iust.ac.ir
2- Faculty of Engineering, University of Mohaghegh Ardabil, Ardabil, Iran, Email: jamali@iust.ac.ir
3- Faculty of Computer Engineering, University of Science and Technology, Tehran, Iran, Email: analoui@iust.ac.ir

Abstract: The main enabling technology for cloud computing is the use of virtual machines. After making the decision of their placement on hosts, they will be set to run. This process is called virtual machine placement. This process has a great importance on energy consumption and resource wastage avoidance. On the other hand, the growing complexity of cloud infrastructure compounds the problem. In this article, the problem of virtual machine placement is transformed to an optimization problem. The goal is minimizing energy consumption and maximizing the profit of placement, simultaneously. A newly emerged optimization method, called Imperialist Competitive Algorithm is applied in this paper. In addition, a unique method for generating new solutions based on already discovered ones, proposed. Finally the success of the proposed algorithm is confirmed by simulation results and its evaluation is compared with GGA and FFD algorithm.

Keywords: Cloud computing, Virtual machine placement, Imperialist competitive algorithm, Energy consumption.

تاریخ ارسال مقاله: ۱۳۹۳/۰۳/۰۵
تاریخ بازنگری مقاله: ۱۳۹۳/۰۸/۱۴ و ۱۳۹۳/۱۰/۲۰
تاریخ پذیرش مقاله: ۱۳۹۳/۱۱/۳۰
نام و نام خانوادگی نویسنده مسئول: شهرام جمالی
نشانی نویسنده مسئول: ایران - اردبیل - انتهای خیابان دانشگاه - دانشگاه محقق اردبیلی - دانشکده فنی‌مهندسی.

۱- مقدمه

متخصصان و تحلیل‌گران صنعت کامپیوتر و IT معتقدند، رشد سیستم‌های مبتنی‌بر ابر در سال‌های آینده نه‌تنها متوقف نخواهد شد، بلکه شتاب بیشتری نیز خواهد گرفت. این سیستم‌ها امکان آن را فراهم می‌سازند تا سازمان‌ها به‌جای خرید چندین سرور برای تأمین نیازهای محاسباتی، برنامه‌های خود را بر روی سرورهای ابری اجرا کرده و تنها به‌میزان استفاده خود هزینه پرداخت کنند و هرگاه با افزایش تقاضا مواجه شدند، به‌میزان نیاز، منابع بیشتری را در خدمت بگیرند. آنچه این امکان را در بستر ابر فراهم می‌سازد، تکنولوژی مجازی‌سازی است که توسط آن، اشتراک پویای منابع سخت‌افزاری برای برنامه‌های مختلف ممکن می‌شود. همچنین با استفاده از تکنولوژی مجازی‌سازی، می‌توان بر روی یک ماشین فیزیکی، چندین پلتفرم[1] با سطح دسترسی و کیفیت اجرایی متفاوت ایجاد کرد. هریک از این پلتفرم‌ها را یک ماشین مجازی (VM)[2] می‌نامیم. هر ماشین مجازی برای اجرا نیاز به یک ماشین فیزیکی دارد تا نیازهای سخت‌افزاری آن همچون حافظه و CPU را تأمین کند. فرآیند یافتن ماشین فیزیکی با قابلیت تأمین منابع مورد نیاز برای اجرای یک ماشین مجازی را، فرآیند مکان‌یابی آن ماشین مجازی می‌نامیم.

این‌که کدام ماشین‌های مجازی برای اجرا بر روی یک هاست در نظر گرفته شوند، در مصرف انرژی و نرخ بهره‌وری آن ماشین فیزیکی، نقش کلیدی دارند. از همین‌رو، مسئله مکان‌یابی برای صاحبان بسترهای سخت‌افزاری بسیار حائز اهمیت است. از طرفی با روند رشد مراکز داده[3] مدرن، مسئله مکان‌یابی ماشین‌های مجازی، تبدیل به چالشی اساسی شده است. در حقیقت از لحاظ سطح دشواری این مسئله در دسته مسائل NP Hard قرار می‌گیرد [۱] و برای حل آن روش‌ها و رویکردهای متفاوتی اتخاذ شده است. استفاده از برنامه‌ریزی خطی [۲]، الگوریتم‌های تکاملی مانند ژنتیک [۳] و روش‌های بسته‌بندی[3] [۴] برخی از این روش‌ها می‌باشند.

در این مقاله، با هدف کاهش مصرف انرژی و کاهش میزان اتلاف منابع در بسترهای سخت‌افزاری، مکان‌یابی ماشین‌های مجازی به‌صورت یک مسئله بهینه‌سازی در نظر گرفته شده است. برای حل آن از الگوریتم بهینه‌سازی تکاملی رقابت استعماری (ICA)[4] [۵] استفاده شده است. این الگوریتم نوظهور در سال ۲۰۰۷ توسط اسماعیل آتشپز گرگری و کارو لوکاس معرفی شده و تاکنون در حل مسائل بهینه‌سازی در زمینه‌های مختلف، موفق عمل کرده است. نمونه‌هایی از به‌کارگیری این الگوریتم در مراجع [۱۰ - ۶] قابل بررسی است. این الگوریتم در حل مسائل بهینه‌سازی بسیار کارا بوده و با سرعت بالایی در فضای جستجو به نقاط بهینه همگرا می‌شود. ازاین‌رو، الگوریتم رقابت استعماری برای حل مسئله مکان‌یابی ماشین‌های مجازی، برای نخستین بار به‌کار گرفته شده است. همچنین لازم به ذکر است که رویکرد اتخاذشده در این پژوهش با فرض ثابت بودن شرایط ماشین‌های مجازی در یک بازه زمانی مشخص صورت گرفته است. مسئله مکان‌یابی با این فرض در مقالات، معمولاً با عنوان مکان‌یابی اولیه مطرح است.

در بخش ۲ برخی از کارهای انجام‌شده در زمینه مکان‌یابی ماشین‌های مجازی در ساختارهای ابری را مرور کرده، نقاط ضعف و قدرت هریک به‌طور خلاصه بررسی می‌شود. همچنین گام‌های اصلی الگوریتم رقابت استعماری نیز در این بخش توضیح داده شده است. در بخش ۳ با کمک روابط ریاضی، تابع هدف و محدودیت‌های منظور شده برای آن معرفی می‌شوند. بدنه اصلی این مقاله و مراحل مکان‌یابی ماشین‌های مجازی با استفاده از الگوریتم رقابت استعماری نیز، در این بخش به‌طور کامل شرح داده شده است. بخش ۴، نتایج شبیه‌سازی الگوریتم و مقایسه آن با الگوریتم‌های GGA و FFD [۱۱، ۱۲] را نشان می‌دهد. در بخش ۵ نتیجه‌گیری مقاله و بحث در مورد کارهای آینده قرار داده شده است.

۲- راهکارهای گذشته و مفاهیم پایه

در این بخش ابتدا مروری بر روش‌های موجود در حل مسئله مکان‌یابی ماشین‌های مجازی خواهیم داشت. رویکردهای متفاوت و مطرح در این زمینه را بررسی کرده و کاستی‌ها و ویژگی‌های هریک شرح داده می‌شود. سپس مراحل مختلف الگوریتم تکاملی رقابت استعماری به‌کار گرفته‌شده در این پژوهش، بررسی می‌شود

۲-۱- پیشینه تحقیق و کارهای مرتبط

مکان‌یابی ماشین‌های مجازی به‌دلیل افزایش پیچیدگی سیستم‌های ابری اهمیت ویژه‌ای دارد. مرجع‌هایی از جمله [۱، ۲] به توضیح ضرورت این مسئله به‌طور جامع پرداخته و تأثیر آن را در میزان مصرف انرژی و اتلاف منابع، نشان می‌دهد. به‌طورکلی در حل مسئله مکان‌یابی اهداف مختلفی را می‌توان در نظر گرفت. کاهش میزان توان مصرفی و تعداد ماشین‌های فیزیکی به‌کاررفته، کاهش میزان منابع به‌هدررفته و افزایش سود کلی سیستم، برخی از این نمونه‌ها هستند. در بسیاری از تحقیقات انجام‌شده، کاهش تعداد ماشین‌های فیزیکی به‌عنوان هدف اصلی مکان‌یابی در نظر گرفته شده است. برای مثال، در مرجع [۳] روش برنامه‌ریزی خطی برای یافتن مکان‌یابی بهینه به‌کار گرفته شده است. در مراجع [۱۳] و [۱۴] نیز مسئله طوری تغییر شکل یافته که تابع هدف، تابعی درجه‌یک از متغیرهای مسئله باشد. همچنین، محدودیت‌هایی از قبیل عدم هم‌نشینی دو ماشین مجازی خاص در یک ماشین فیزیکی در نظر گرفته شده است. علاوه بر این، اهدافی چون کاهش تعداد مهاجرت ماشین‌های مجازی در دو مرجع [۱۳] و [۱۴] لحاظ شده است. اشکال راه‌حل‌های مکان‌یابی با استفاده از توابع خطی، ساده‌سازی بدون در نظر گرفتن شرایط خاص پایگاه‌های ابری و محدود کردن مسئله است. به‌طوری‌که ممکن است جواب‌های نهایی در عمل چندان کارا نباشند.

مسئله مکان‌یابی ماشین‌های مجازی در بسیاری از تحقیقات انجام‌شده به‌صورت نسخه‌ای از مسئله بسته‌بندی که خود از رده مسائل NP hard است [۱۵]، فرموله می‌شود. این رویکرد ابتدا توسط Ajiro و Tanaka در مرجع [۱۶] پیشنهاد شد و روش اول برازش نزولی (FFD) برای حل آن اتخاذ گردید. با همین رویکرد روش‌های ابتکاری متنوعی دیگری نیز ارائه شده‌اند [۲۲-۱۷]. برای مثال در [۱۷] سیستمی بنام pMapper طراحی شده است که در آن دو هدف هزینه‌ها و کاهش توان مصرفی مدنظر قرار گرفته و سیستم با بهره‌گیری از روش متداول FFD در مسائل بسته‌بندی، مکان‌یابی ماشین‌های مجازی را مدیریت می‌کند.

با تغییر شکل مسئله مکان‌یابی به‌صورت مسئله بهینه‌سازی که روشی متداول در این حوزه است، الگوریتم‌های تکاملی به‌طور گسترده به‌کار گرفته شده‌اند. در مرجع [۳] روش بهینه‌سازی با استفاده از نسخه خاصی از الگوریتم ژنتیک بنام ژنتیک گروهی، (معرفی‌شده در [۱۱]) به‌کار رفته است. روش مطرح‌شده برای تولید پاسخ‌های جدید در مرجع [۳] چندان کارا نبوده و سربار بالایی را به الگوریتم، تحمیل می‌کند. همچنین احتمال به دام افتادن در جواب‌های بهینه محلی وجود دارد.

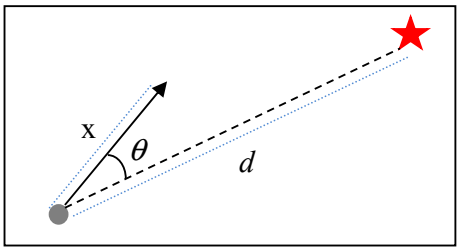
همان‌طور که ذکر شد، در تغییر شکل مسئله به‌صورت بهینه‌سازی، می‌توان بیش از یک هدف را مدنظر قرار داد. در مرجع [۸] دو هدف کاهش توان مصرفی و میزان هدر رفتن منابع در نظر گرفته شده است. در این روش، مکان‌یابی به شکل یک مسئله بهینه‌سازی ترکیبی معرفی شده و با استفاده از الگوریتم کلونی مورچگان [۲۰] روش حلی برای آن ارائه گشته است. این مقاله، با روشی هوشمند فضای جستجوی مسئله را بررسی کرده و با مفهوم بهینه‌سازی چند شاخصه جواب‌های قابل قبولی را گزارش کرده است.

در پژوهش حاضر نیز، مسئله مکان‌یابی به‌صورت مسئله بهینه‌سازی، با ترکیب دو هدف کاهش مصرف انرژی و کاهش میزان منابع به‌هدررفته، فرموله شده و با استفاده از الگوریتم تکاملی رقابت استعماری برای مسئله مکان‌یابی در ابعاد مختلف، پاسخ‌های مناسبی را از فضای جستجو استخراج می‌کند. در بخش بعد به معرفی الگوریتم رقابت استعماری پرداخته و گام‌های آن را بررسی خواهیم کرد.

۲-۲- الگوریتم رقابت استعماری

الگوریتم رقابت استعماری (ICA)، الگوریتمی نوظهور در حیطه محاسبات تکاملی و برگرفته از روند تکامل جوامع انسانی است. همانند سایر الگوریتم‌های تکاملی همچون ژنتیک، این الگوریتم نیز، با تعدادی جمعیت اولیه تصادفی که هرکدام از آن‌ها یک کشور نامیده می‌شوند، شروع می‌شود. هر کشور دارای مشخصاتی است که مکان آن را در فضای جستجو مشخص می‌کند. هر کشور در حقیقت به‌عنوان نقطه‌ای از این فضا تعریف می‌شود. پس از امتیازدهی، کشورها از لحاظ میزان برازش مرتب شده و تعداد مشخصی از بهترین‌ها، به‌عنوان کشورهای استعمارگر و سایر کشورها به‌عنوان مستعمره در نظر گرفته می‌شوند. در الگوریتم ICA به‌جای اصطلاح میزان برازش، از اصطلاح قدرت استفاده می‌شود. بر همین اساس قدرتمندترین کشورها استعمارگر و ضعیف‌ترها به‌عنوان مستعمره معرفی می‌شوند. کشورهای مستعمره، به‌طور تصادفی بین استعمارگران تقسیم می‌شوند. هرچه میزان قدرت کشور استعمارگر بیشتر باشد، تعداد کشورهای مستعمره بیشتری را به خود اختصاص می‌دهد.

حرکت در فضای جستجو به این شکل است که هر کشور مستعمره، تحت تأثیر قدرت کشور استعمارگر خود و هم‌راستا با او، به‌طور تصادفی حرکت می‌کند. شکل ۱، با نمایش کشور استعمارگر با نماد ستاره و کشور مستعمره با نماد دایره این حرکت را نشان می‌دهد.



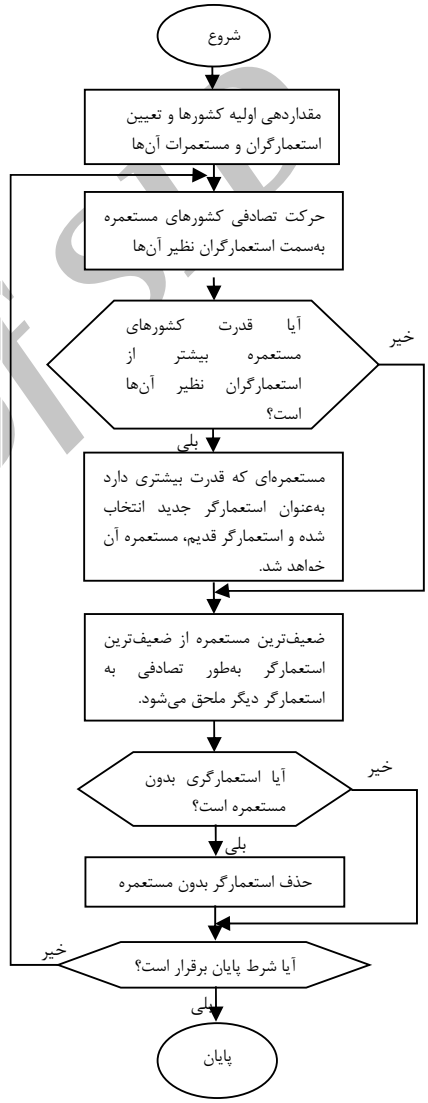
شکل ۱: نحوه حرکت کشورها در فضای جستجو بر اساس الگوریتم رقابت استعماری [۹]

همان‌طور که در شکل ۱ دیده می‌شود، فاصله مستقیم بین فاصله استعمارگر و مستعمره‌اش برابر با d در نظر گرفته شده است. حرکت مستعمره به‌سمت استعمارگر، نه به‌طور مستقیم و بر روی این خط، بلکه با زاویه‌ای برابر با θ و به‌اندازه x واحد انجام می‌شود. مقدار زاویه انحراف و اندازه حرکت به‌طور یکنواخت، در بازه‌های تعیین‌شده تصادفی مشخص می‌گردند. مقادیر θ و x ازجمله پارامترهای الگوریتم ICA است که بنا به ماهیت مسئله بهینه‌سازی مفهوم و مقادیر مختلفی را به خود می‌گیرد. در طی اجرای الگوریتم، اگر در نتیجه حرکت کشورها، یک کشور مستعمره قدرت بیشتری از استعمارگر نظیر خود پیدا کند، جای این دو عوض خواهد شد. به‌عبارت‌دیگر در مراحل بعدی اجرای الگوریتم، تمام کشورهای مستعمره و ازجمله استعمارگر سابق، به استعمارگر جدید تعلق خواهند گرفت و حرکت این مستعمرات به‌سمت استعمارگر جدید خواهد بود.

همچنین برای رهایی از جواب‌های بهینه محلی با نرخی مشخص کشورهای مستعمره دچار دگرگونی می‌شوند تا با تغییر مکان آن‌ها، مختصات متفاوتی از فضای جستجو نیز با هدف یافتن جوابی بهتر، بررسی شود. این مفهوم در نسخه‌های اولیه الگوریتم ICA انقلاب نامیده می‌شود.

هر مرحله تکرار الگوریتم ICA را یک دهه می‌نامند که در طی آن رقابتی استعماری بین استعمارگران برقرار است. در هر دهه کم قدرت‌ترین استعمارگر، ضعیف‌ترین کشور مستعمره خود را از دست خواهد داد. سایر کشورهای استعمارگر بر سر مالکیت این مستعمره ضعیف با یکدیگر به رقابت می‌پردازند. احتمال انتساب این مستعمره به هریک از استعمارگران متناسب با میزان قدرت استعمارگران خواهد

بود. در نهایت یک استعمارگر، با از دست دادن تمامی مستعمرات خود، به‌صورت یک مستعمره درآمده و تحت سلطه یک استعمارگر دیگر درخواهد آمد. به همین‌ترتیب، مراحل الگوریتم ICA ادامه می‌یابد تا بالاخره تنها یک استعمارگر باقی‌مانده و مالکیت تمام کشورها را در اختیار گیرد. همچنین شروط مختلفی، مانند اجرای تعداد تکرار از پیش تعیین‌شده‌ای از الگوریتم را، می‌توان به‌عنوان شروط پایان الگوریتم در نظر گرفت. شکل ۲ فلوچارت اجرای الگوریتم را مطابق با توضیحات فوق نشان می‌دهد.



شکل ۲: فلوچارت الگوریتم رقابت استعماری [۵]

۳- مکان‌یابی ماشین‌های مجازی با استفاده از الگوریتم رقابت استعماری

در این بخش به بررسی روش پیشنهادی حل مسئله مکان‌یابی ماشین‌های مجازی با استفاده از الگوریتم رقابت استعماری پرداخته می‌شود. در ابتدا، تابع هدف طراحی‌شده برای این مسئله را که ترکیب خطی دو هدف کاهش مصرف انرژی و کاهش سطح منابع به‌هدررفته است، معرفی می‌شود. محدودیت‌های لازم در خصوص پاسخ‌های احتمالی مسئله را برشمرده می‌شود. در کنار جزئیات مدل‌سازی، نحوه به‌کارگیری الگوریتم رقابت استعماری را در این مسئله بررسی می‌گردد. همچنین روشی هدفمند برای تولید پاسخ‌های جدید در قالب مفاهیم مطرح در الگوریتم رقابت استعماری و حرکت در فضای جستجو، ارائه شده است. نتایج شبیه‌سازی نشان می‌دهد این روش، در کنار استفاده از الگوریتم استعماری توانسته است، راهکاری کارا برای حل مسئله مکان‌یابی ماشین‌های مجازی ارائه دهد.

۳-۱- تابع هدف و محدودیت‌ها

روش مکان‌یابی با استفاده از الگوریتم رقابت استعماری به‌کارگیری توابع هدف متفاوتی را ممکن می‌سازد. همچنین محدودیت‌های معرفی‌شده در این بخش، قابلیت تغییر و توسعه دارند. برای سادگی، تنها دو هدف کلیدی کاهش مصرف انرژی و کاهش میزان منابع به‌هدررفته مد نظر قرار داده می‌شود. هرچه تعداد ماشین‌های فیزیکی به‌کار گرفته‌شده در یک مکان‌یابی پیشنهادشده کمتر باشد، مصرف انرژی کاهش می‌یابد، چرا که ماشین‌های فیزیکی بی‌استفاده را می‌توان به حالت Low Power برده، توان مصرفی استاتیک را حذف کرد، همچنین میزان تولید CO_2 توسط سرورها را در این حالت می‌توان کاهش داد.

در رابطه با هدف دوم یعنی کاهش سطح اتلاف منابع، مطابق با تابع معرفی‌شده در مرجع [۳]، نرخ منابع استفاده‌شده به ظرفیت هر ماشین فیزیکی، مدنظر قرار می‌گیرد. هر چه مدیریت منابع یک ماشین فیزیکی مناسب‌تر انجام شده باشد، منابع کمتری به هدر رفته و ماشین‌های مجازی بیشتری برای اجرا می‌توانند بر روی آن قرار گیرند. به همین‌ترتیب میزان سود به‌ازای هر ماشین فیزیکی بالاتر خواهد بود. در رابطه ریاضی تابع هدف، از نمادها و متغیرهای مطرح‌شده در جدول ۱ استفاده شده است. این تابع مطابق رابطه (۱) است.

مطابق با محدودیت رابطه (۲)، هر ماشین مجازی فقط به یک ماشین فیزیکی می‌تواند متناظر شود. محدودیت رابطه (۳) اطمینان حاصل می‌سازد که با تناظر ماشین مجازی jبه ماشین فیزیکی i، مقدار Y_i برابر با یک خواهد شد. در محدودیت رابطه (۴) این مسئله بیان شده است که اگر و تنها اگر به تعداد یک یا بیشتر ماشین مجازی به ماشین فیزیکی iاُم متناظر شود، مقدار Y_i برابر یک خواهد شد. محدودیت‌های روابط (۵) و (۶) بیانگر آن هستند که هر ماشین فیزیکی باید منابع بیشتری از مجموع نیازهای حافظه و CPU ماشین‌های مجازی متناظرشده به آن را در اختیار داشته باشد تا این ماشین‌های مجازی در هنگام اجرا بر روی ماشین فیزیکی متناظر، با مشکل کمبود منابع روبرو نشوند. رابطه (۱)، در مقالات بهینه‌سازی با عنوان تابع هدف، تابع ارزش و یا تابع برازش مطرح است. در الگوریتم رقابت استعماری همان‌طور که در بخش ۲-۲ مطرح شده است،

مقداری که این تابع به هر پاسخ (کشور) متناظر می‌کند بیانگر عکس قدرت آن کشور است. همچنین تعیین این‌که کدام کشورها استعمارگر و کدام‌ها مستعمره هستند بر اساس قدرت آن‌ها مشخص می‌شود. در مجموع، هدف به‌دست آوردن مقدار مناسب برای متغیر باینری x_{ij} است، به‌طوری‌که رابطه (۱) کمینه شده و محدودیت‌های روابط (۲) تا (۶) رعایت شوند.

Minimize:

$$\sum_i Y_i - K \times \sqrt{\sum_i \left(\left(\frac{\sum_j \text{Mem\_usage}_j \times x_{ij}}{\text{Mem\_cap}_i} \right)^2 + \left(\frac{\sum_j \text{CPU\_usage}_j \times x_{ij}}{\text{CPU\_cap}_i} \right)^2 \right)} \quad (۱)$$

Subject to:

$$\sum_i x_{ij} = 1 \qquad \forall j \quad (۲)$$

$$Y_i \geq x_{ij} \qquad \forall i,j \quad (۳)$$

$$\sum_i x_{ij} \geq Y_i \qquad \forall j \quad (۴)$$

$$Y_i \times \text{CPU\_cap}_i \geq \sum_j \left(x_{ij} \times \text{CPU\_usage}_j \right) \qquad \forall i,j \quad (۵)$$

$$Y_i \times \text{Mem\_cap}_i \geq \sum_j \left(x_{ij} \times \text{Mem\_usage}_j \right) \qquad \forall i,j \quad (۶)$$

جدول ۱: پارامترها، متغیرها و نمادهای به‌کاررفته در تابع هدف

نمادها	
j	اندیس ماشین‌های مجازی
i	اندیس ماشین‌های فیزیکی
متغیرها	
x_{ij}	متغیر باینری، برابر یک اگر ماشین مجازی j به ماشین فیزیکی i متناظر شده باشد
Y_i	متغیر باینری، برابر یک اگر ماشین فیزیکی i استفاده شده باشد
پارامترها	
Mem_cap_i	ظرفیت اولیه حافظه برای ماشین فیزیکی i
CPU_cap_i	ظرفیت اولیه CPU برای ماشین فیزیکی i
Mem_usage_j	حافظه مورد درخواست از سوی ماشین مجازی j
CPU_usage_j	CPU مورد درخواست از سوی ماشین مجازی j
K	پارامتر اولویت‌دهی

۳-۲- روش پیشنهادی

در این بخش روش به‌کارگیری الگوریتم رقابت استعماری در مسئله مکان‌یابی ماشین‌های مجازی را بررسی کرده، بدنه اصلی روش پیشنهادی را ارائه می‌دهیم. جزئیات مدل‌سازی مسئله، در قالب مفاهیم مطرح‌شده در این الگوریتم بررسی می‌شود. همچنین روشی هدفمند برای تولید پاسخ‌های جدید و حرکت در فضای جستجو، ارائه شده است. نتایج شبیه‌سازی نشان می‌دهد این روش، در کنار استفاده از الگوریتم استعماری توانسته است، راهکاری کارا برای حل مسئله مکان‌یابی ماشین‌های مجازی ارائه دهد.

۳-۲-۱- شیوه نمایش پاسخ‌ها

شیوه کد کردن پاسخ‌ها در این مقاله تا حدودی مشابه روش ارائه‌شده در [۳] است. برای توصیف این روش، شکل ۳ را که نمایش یک پاسخ پیشنهادی برای مکان‌یابی ۵ ماشین مجازی است، در نظر بگیرید.

Vm#1	Vm#2	Vm#3	Vm#4	Vm#5
A	C	B	A	B

شکل ۳: یک پاسخ برای مکان‌یابی ماشین‌های مجازی ۱ تا ۵

همان‌طور که در شکل ۳ دیده می‌شود هر پاسخ، آرایه‌ای است که تعداد دریچه‌های آن برابر با تعداد ماشین‌های مجازی مورد درخواست برای مکان‌یابی است. اندیس هر عنصر این آرایه، شماره ماشین مجازی و مقدار متناظر شده به آن، مشخصه ماشین فیزیکی است که برای اجرای آن ماشین مجازی در نظر گرفته شده است. برای مثال در شکل ۳، ماشین فیزیکی A برای اجرای دو ماشین مجازی ۱ و ۴، ماشین فیزیکی C برای اجرای ماشین مجازی ۲ و همچنین ماشین مجازی ۳ و ۵ به ماشین فیزیکی B متناظر گردیده است.

مطابق با الگوریتم ICA، هر پاسخ را یک کشور می‌نامیم. در الگوریتم پیشنهادی مکان‌یابی ماشین‌های مجازی مبتنی بر الگوریتم ICA، شیوه نوینی برای حرکت کشورها و تولید پاسخ‌های جدید معرفی شده است. برای توصیف این شیوه مفهوم جدیدی بنام "نماینده" به الگوریتم ICA افزوده گردیده است. در ادامه این مفهوم معرفی و روش نمایش آن ارائه می‌شود.

۳-۲-۲- مفهوم نماینده و شیوه نمایش آن

همان‌طور که در بخش قبل مطرح شد، هر پاسخ، آرایه‌ای است که اندازه آن برابر با تعداد ماشین‌های مجازی مورد درخواست برای مکان‌یابی است. با توجه به تابع هدف معرفی‌شده در بخش ۳-۱، آنچه برای برازش یک پاسخ نیاز است، تعداد ماشین‌های فیزیکی استفاده‌شده و نرخ بهره‌وری هر ماشین فیزیکی، در آن پاسخ است. ازاین‌رو ساختاری با هدف گردآوری این اطلاعات معرفی می‌شود.

نماینده هر پاسخ، آرایه‌ای است که اندازه آن برابر با تعداد ماشین‌های فیزیکی مجازی به‌کار گرفته‌شده در آن پاسخ است. مقدار هر عنصر، مشخصه هریک از ماشین‌های فیزیکی به‌کار گرفته‌شده است. اندیسی که این ماشین فیزیکی در آن قرار گرفته است، برای مقایسه نرخ بهره‌وری آن ماشین فیزیکی به‌کار می‌رود.

برای توضیح بهتر، پاسخ نمایش داده‌شده در شکل ۳ را بار دیگر در نظر بگیرید. در مجموع، این پاسخ از سه ماشین فیزیکی متفاوت C، A و B استفاده می‌کند. فرض کنید نرخ بهره‌وری برای این سه ماشین فیزیکی به‌ترتیب برابر ۷۰، ۴۰ و ۳۰ درصد است. با توجه به این توضیحات نماینده این پاسخ با شکل ۴ نمایش داده شده است.

C 1	A 2	B 3

شکل ۴: نماینده پاسخ نمایش داده‌شده در شکل ۳

اعداد ۱، ۲ و ۳ در شکل ۴، اندیس عناصر آرایه هستند. با توجه به نرخ بهره‌وری، هر ماشین فیزیکی در خانه مناسبی از آرایه نماینده مستقر شده است. به‌دلیل آن‌که ماشین فیزیکی C در مقایسه با دو ماشین فیزیکی دیگر نرخ بهره‌وری بالاتری را دارد، در خانه با اندیس کوچک‌تر قرار گرفته است. به همین‌ترتیب هر چه مقدار محاسبه‌شده این نرخ برای یک ماشین فیزیکی کمتر می‌شود، ماشین فیزیکی مورد بحث در خانه‌ای با اندیس بالاتر در آرایه نماینده قرار می‌گیرد. به این‌ترتیب از روی نماینده یک پاسخ، می‌توان فهمید که تمرکز اصلی پاسخ متناظر آن، بر روی کدام ماشین‌های فیزیکی است. در ادامه از مفهوم نماینده برای حرکت یک کشور مستعمره (پاسخ ضعیف‌تر) به‌سمت استعمارگر متناظرش (پاسخ قوی‌تر) استفاده می‌کنیم.

۳-۲-۳- تولید پاسخ جدید

این بخش، متناظر با مرحله حرکت مستعمره به‌سمت استعمارگر در الگوریتم ICA و یا مرحله Crossover یا تولید ژن‌های جدید در الگوریتم ژنتیک است. در ادامه، با معرفی روشی هدفمند با استفاده از مفهوم نماینده، به تولید پاسخ‌های جدید می‌پردازیم.

چالش اساسی در پیاده‌سازی حرکت مستعمره به‌سمت استعمارگر در الگوریتم ICA، تعریف زاویه انحراف θ و اندازه حرکت x است که در مقالاتی همچون مرجع [۹] نیز به آن اشاره شده است. در ادامه ضمن معرفی روش تولید پاسخ جدید، تعریفی خاص از این پارامترها در الگوریتم پیشنهادی ارائه می‌گردد.

دو پاسخ S۱ و S۲ را در نظر بگیرید. مطابق بخش ۴-۱ این دو پاسخ، آرایه‌هایی با طول یکسان هستند. فرض کنید، S۱ یک مستعمره و S۲ استعمارگر متناظر آن است. منظور از تولید پاسخ جدید در حقیقت، حرکت S۱ به‌سمت S۲ و در نتیجه تغییر مشخصات پاسخ S۱ می‌باشد. نمایندگان این دو پاسخ را $Pres_{S1}$ و $Pres_{s2}$ می‌نامیم. ماشین فیزیکی واقع‌شده در آخرین خانه $Pres_{S1}$ را last_host و ماشین فیزیکی موجود در اولین خانه $Pres_{s2}$ را first_host نام‌گذاری می‌گردد. در پاسخ S۱ ماشین‌های مجازی را که برای اجرا روی last_host در نظر گرفته شده‌اند، می‌یابیم. از این مجموعه ماشین‌های مجازی به تعداد x واحد (x پارامتر شبیه‌سازی) از آن‌ها را به‌طور تصادفی انتخاب کرده و برای اجرا به first_host، منتسب می‌کنیم. این گام را به تعداد θ (پارامتر شبیه‌سازی) بار تکرار می‌کنیم. پارامترهای θ و x در واقع کنترل‌کننده حرکت پاسخ مستعمره به‌سوی استعمارگر هستند که مناسب‌ترین مقدار برای هریک از آن‌ها طی آزمایش‌های شبیه‌سازی مقدماتی تعیین می‌شود.

مطابق با توضیحات ارائه‌شده در بخش ۲ قدرت پاسخ‌ها در هر بار تکرار الگوریتم، محاسبه شده و بر همین اساس رقابت بین استعمارگران شکل می‌گیرد تا در نهایت با توجه به یکی از شروط پایانی، الگوریتم خاتمه یافته و جواب نهایی گزارش شود.

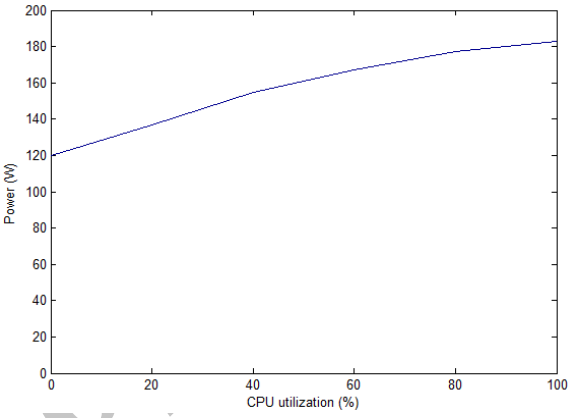
در بخش پیوست‌ها، شبه‌کدهای الگوریتم مکان‌یابی ماشین‌های مجازی با استفاده از رقابت استعماری، در سه بخش مجزا گردآوری شده است. در هریک از این بخش‌ها، عملکرد الگوریتم در مرحله تولید جواب‌های اولیه (پیوست ۱)، بدنه الگوریتم رقابت استعماری (پیوست ۲) و حرکت کشور مستعمره به‌سمت استعمارگر (پیوست ۳)، توضیح داده شده است.

۳-۲-۴- عملگر انقلاب

همان‌طور که در توضیح گام‌های الگوریتم رقابت استعماری توضیح داده شد، عملگری با نام "عملگر انقلاب"، در روند اجرای الگوریتم با هدف رهایی از جواب‌های بهینه محلی، طراحی شده است. این عملگر معادل عملگر "جهش" در الگوریتم ژنتیک است و با تغییرات ناگهانی در مکان کشورهای مستعمره پیاده‌سازی می‌شود. در الگوریتم پیشنهادی در این پژوهش نیز این عملگر در نظر گرفته شده است. در آغاز هر دهه، هر پاسخ قبل از ورود به رقابت، با احتمال مشخصی دچار تغییر می‌شود و ماشین‌های مجازی در ساختار پاسخ مورد بحث، به‌صورت تصادفی به ماشین فیزیکی دیگری منتسب می‌گردند. محدودیت‌ها برای پاسخ جدید بررسی می‌شود. در صورت رعایت این محدودیت‌ها پاسخ جدید پذیرفته شده و سپس مقدار تابع هدف برای پاسخ دگرگون‌شده محاسبه می‌شود. در غیر این‌صورت پاسخ مورد بحث، بار دیگر به‌صورت تصادفی تغییر داده می‌شود. در نهایت با انتخاب نرخ مناسبی برای عملگر انقلاب، می‌توان عملکرد الگوریتم را بهبود بخشید. در پیاده‌سازی‌های انجام‌شده در این پژوهش، نرخ ۰/۰۷ برای عملگر انقلاب در نظر گرفته شده است.

۴- نتایج شبیه‌سازی

الگوریتم پیشنهادی برای حل مسئله مکان‌یابی با استفاده از الگوریتم ICA، به زبان جاوا نوشته شده و در محیط شبیه‌ساز ابری Cloudsim [۲۸] پیاده‌سازی نهایی شده است. مطالعات اخیر [۲۲]، نشان می‌دهد که میزان توان و انرژی مصرفی یک ماشین فیزیکی، به‌سادگی از روی نرخ بهره‌وری آن قابل محاسبه است چرا که رابطه بین توان مصرفی و نرخ بهره‌وری در یک ماشین فیزیکی، رابطه‌ای خطی است. شکل ۵، این رابطه خطی را بین میزان توان مصرفی و نرخ بهره‌وری یک ماشین فیزیکی نشان می‌دهد.



شکل ۵: رابطه بین توان مصرفی و نرخ بهره‌وری یک ماشین فیزیکی [۲۲]

همان‌طور که در شکل ۵ مشخص است، یک ماشین فیزیکی روشن با نرخ بهره‌وری نزدیک به صفر، همچنان میزان مشخصی توان مصرف می‌کند (نزدیک به ۱۲۰ وات). لذا هر چه تعداد ماشین‌های فیزیکی به‌کاررفته در یک پاسخ برای مکان‌یابی، کمتر و نرخ بهره‌وری آن‌ها بیشتر باشد، میزان مصرف توان و انرژی کاهش می‌یابد. به همین دلیل بسیاری از مقالات مانند [۱۲]، هدف اصلی الگوریتم مکان‌یابی پیشنهادی خود را کاهش تعداد ماشین‌های فیزیکی به‌کاررفته در نظر می‌گیرند.

در این پژوهش علاوه بر در نظر گرفتن این پارامتر به‌عنوان هدف و یک معیار ارزیابی، میزان اتلاف منابع و همچنین میزان مصرف انرژی نیز در هر ماشین فیزیکی مورد توجه قرار داده شده است.

به‌منظور مقایسه عملکرد الگوریتم ارائه‌شده در این پژوهش با روش‌های مشابه، از روابط مطرح در مقالات، برای محاسبه توان مصرفی و همچنین میزان منابع به‌هدررفته، استفاده شده است.

برای محاسبه میزان توان مصرفی یک ماشین فیزیکی، از رابطه (۷) معرفی‌شده در مرجع [۲۰] استفاده شده است.

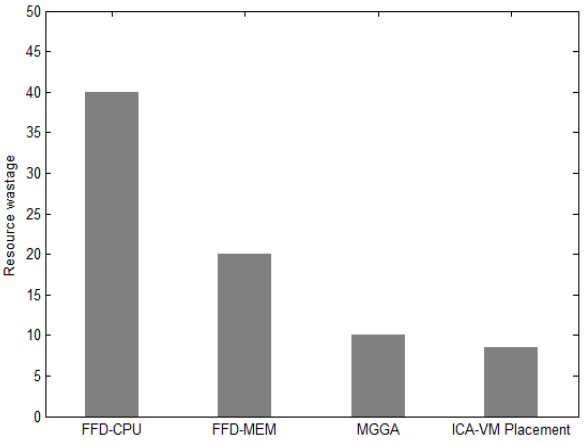
$$P_j = \begin{cases} \left(P_j^{\text{busy}} \quad P_j^{\text{idle}}\right) \times U_j^P + P_j^{\text{idle}}, & U_j^p > 0 \\ 0 \quad , \text{otherwise} \end{cases} \tag{۷}$$

که در آن، P_j^{idle} و P_j^{busy} متوسط توان مصرفی ماشین فیزیکی مورد نظر، به‌ترتیب در حالت بهره‌وری صفر و ۱۰۰٪ است که این مقدار برابر ۱۲۰ و ۱۸۰ وات در نظر گرفته شده است و U_j^p نرخ بهره‌وری ماشین فیزیکی j است. همچنین میزان اتلاف منابع برای یک ماشین فیزیکی با استفاده از رابطه (۸)، معرفی‌شده در مرجع [۱۲]، محاسبه می‌شود.

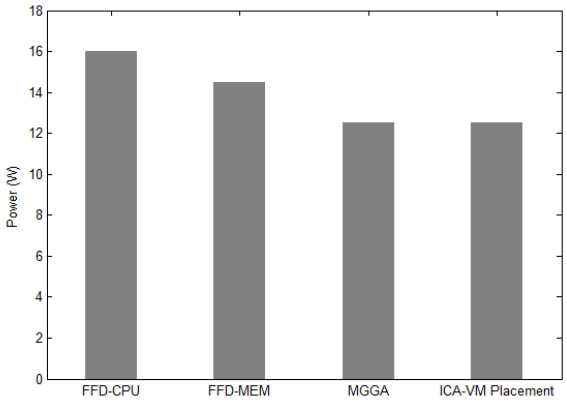
$$W = \sum_{j \neq k} \left(R_j \quad R_k\right) \tag{۸}$$

که در آن، W میزان اتلاف کل منابع، R_j و R_k نسبت منابع استفاده‌نشده به مجموع منابع یک ماشین فیزیکی، در دو بعد j (CPU) و k (Memory) است. برای ارزیابی کارایی در دو جنبه توان مصرفی و میزان اتلاف منابع، الگوریتم مکان‌یابی ماشین‌های مجازی با استفاده از الگوریتم رقابت استعماری، بر روی سیستمی با پردازنده دو هسته‌ای Intel با قدرت پردازشی ۲/۰۲ GHz و ۳GB حافظه موقت شبیه‌سازی شده است. لازم به‌ذکر است، مقدار پارامترهای الگوریتم ICA، در نتیجه نهایی نقش اساسی دارند. لذا، با انجام یک سری آزمایش‌های مقدماتی مقدار مناسب برای این پارامترها محاسبه شده است. الگوریتم مکان‌یابی با بهره‌گیری از رقابت استعماری در سناریویی با تعداد ۱۲۰ و ۲۵۰ ماشین مجازی با درخواست‌های حافظه و CPU تصادفی به‌ترتیب در بازه‌های [۱۰۰- ۱۰] و [۱۰ - ۱] مشابه با مرجع [۱۱]، ارزیابی شده است. همچنین ظرفیت ماشین‌های فیزیکی از لحاظ حافظه، مقدار ثابت ۱۵۰ و از لحاظ CPU، ۱۰ در نظر گرفته شد. نتایج گزارش‌شده در جدول ۲، موفقیت این الگوریتم را در مقایسه با دو الگوریتم ژنتیک گروهی [۱۱] و روش فرا ابتکاری دومرحله‌ای [۱۲] نشان می‌دهد. همچنین با استفاده از روابط (۷) و (۸) نمودارهای میله‌ای شکل‌های ۶ و ۷ برای مقایسه عملکرد الگوریتم پیشنهادی از لحاظ اتلاف منابع و توان مصرفی در مقایسه با دو الگوریتم FFD و MGGA تهیه شده است. MGGA [۱۷] نسخه‌ای موفق از به‌کارگیری الگوریتم ژنتیک در مکان‌یابی ماشین‌های مجازی است. نتایج شبیه‌سازی نشان می‌دهد، انتخاب الگوریتم ICA و نحوه استفاده از آن، در کنار روش اختصاصی مطرح در بخش ۳ برای تولید جواب‌های جدید، همراه با موفقیت بوده و نتایجی قابل مقایسه با روش‌های GGA و FFD [۱۱، ۱۲] حاصل شده است. همچنین از لحاظ زمانی به‌دلیل ماهیت الگوریتم رقابت استعماری و روش خاص تولید پاسخ‌های جدید، الگوریتم مکان‌یابی پیشنهادی، در زمان قابل قبولی به جواب‌های نزدیک به بهینه همگرا می‌شود. این زمان برای مکان‌یابی ۱۲۰ ماشین مجازی در حدود ۱۲ ثانیه و برای ۲۵۰ ماشین مجازی، ۳۷ ثانیه است. همچنین لازم به‌ذکر است که اطلاعات جدول ۲، گزارش بهترین اجراهای الگوریتم‌های مختلف از بین چندین اجرای پی‌درپی است. با این حال اجراهای متعدد مکان‌یابی ماشین‌های مجازی با استفاده از الگوریتم رقابت استعماری حاکی از آن است که الگوریتم، عملکرد نسبتاً ثابتی داشته و تنها در ۱۰٪ اجراها، در نتایج گزارش‌شده مکان‌یابی، تفاوت قابل توجهی دیده می‌شود. برای اطمینان از عملکرد یکنواخت الگوریتم، آزمون آماری ویلکاکوس روی دو دسته از نتایج اجراهای مختلف، انجام گردید و بر اساس آن فرض مربوط به معنی‌دار بودن اختلاف بین نتایج در این دو دسته، رد گردید.

همان‌طور که در بخش ۳ بررسی شد، روش مکان‌یابی با استفاده از الگوریتم ICA، با معرفی روشی جدید، با لحاظ کردن مستقیم بخشی از محاسبات تابع هدف، به تولید جواب‌های جدید می‌پردازد. نتایج شبیه‌سازی نشان می‌دهد، این شیوه می‌تواند جستجویی هدفمند را ایجاد کرده و فضای جستجو مسئله را به‌صورت کارا بررسی کند. در حقیقت در این رویکرد، از بخشی از تابع هدف به‌عنوان راهنما برای هدایت پاسخ‌های جدید استفاده می‌شود. تولید پاسخ از شیوه کاملاً تصادفی که در الگوریتم‌های دیگر مکان‌یابی استفاده‌شده، فاصله گرفته و به‌صورت هدفمند و ساختاریافته به‌سمت تولید پاسخ‌های بهتر پیش می‌رود. همچنین با استفاده از مفهوم انقلاب در الگوریتم ICA، که به‌صورت تحولات ناگهانی در مشخصات جواب، پیاده‌سازی شده است؛ از به دام افتادن در جواب‌های بهینه محلی پیشگیری به‌عمل آمده است. علاوه بر این، الگوریتم ICA با ارائه متغیرهایی چون θ و x امکان کنترل بر روی سرعت همگرایی را فراهم کرده است به‌طوری‌که می‌توان بین جستجو برای جواب‌های جدید و اتکا به جواب‌های یافته‌شده توازن مناسبی را برقرار ساخت.



شکل ۶: مقایسه اتلاف منابع یک ماشین فیزیکی در الگوریتم‌های مکان‌یابی مختلف



شکل ۷: مقایسه توان مصرفی یک ماشین فیزیکی در الگوریتم‌های مکان‌یابی مختلف

۵- نتیجه‌گیری و کارهای آینده

در این مقاله با پیشنهاد استفاده از الگوریتم بهینه‌سازی رقابت استعماری، روشی خودکار برای مکان‌یابی کارا برای مجموعه‌ای از ماشین‌های مجازی بر روی ماشین‌های فیزیکی ارائه شده است. مکان‌یابی ماشین‌های مجازی با استفاده از الگوریتم رقابت استعماری، با مدنظر قرار دادن اهداف کاهش مصرف انرژی و اتلاف منابع، با در نظر گرفتن محدودیت‌ها در ظرفیت این منابع، به جستجوی جواب می‌پردازد. در این روش با بهره‌گیری از مفهوم جدیدی بنام "نماینده"، فضای جستجو به دنبال پاسخ مناسب مورد بررسی قرار می‌گیرد. این ساختار قابلیت اعمال توابع هدف متفاوت و حتی متناقض را دارد. علاوه بر این، در نسخه‌های آتی آن می‌توان محدودیت‌هایی مانند قواعد هم‌نشینی ماشین‌های مجازی که به‌دلیل خاصیت بسترهای ابری مطرح است را نیز به‌کار گرفت. از جمله دیگر مسائل پیش رو بررسی و پیاده‌سازی مدل پویای مهاجرت ماشین‌های مجازی و ارائه روش جستجوی ترکیبی، با ادغام دو الگوریتم رقابت استعماری و الگوریتم ژنتیک در این ساختار است.

جدول ۲: مقایسه بهترین اجرای الگوریتم‌های مکان‌یابی برای ۱۲۰ و ۲۵۰ ماشین مجازی

الگوریتم مکان‌یابی با استفاده از رقابت استعماری		الگوریتم GGA		الگوریتم FFD		بهترین پاسخ	تعداد ماشین‌های مجازی	شماره اجرا
زمان (ثانیه)	تعداد ماشین‌های فیزیکی استفاده‌شده	زمان (ثانیه)	تعداد ماشین‌های فیزیکی استفاده‌شده	زمان (ثانیه)	تعداد ماشین‌های فیزیکی استفاده‌شده			
۱۳/۳	۵۰	۱۵/۲	۴۸	۰/۰۱۵	۴۹	۴۸	۱۲۰	۱
۱۲/۶	۴۹	۰/۰	۴۹	۰/۰۱۵	۵۰	۴۹	۱۲۰	۲
۱۵/۴	۴۸	۵/۸	۴۶	۰/۰۱۵	۴۷	۴۶	۱۲۰	۳
۴۵/۵	۹۹	۲۵۶/۷	۹۹	۰/۰۳۱	۱۰۱	۹۹	۲۵۰	۴
۳۲/۴	۱۰۲	۴۷/۴	۱۰۰	۰/۰۳۱	۱۰۲	۱۰۰	۲۵۰	۵
۳۵/۲۷	۱۰۲	۲۲۳/۸	۱۰۲	۰/۰۳۲	۱۰۴	۱۰۲	۲۵۰	۶

پیوست‌ها

Input: Set of VMs and hosts with their associated resource demand and the initial resource capacity of Hosts, Set of parameters.
Output: Initial population (Set of valid countries)

```
/*generating a set of valid solutions*/
1.  Set values of parameter:
a.  Pop_num (number of initial population)
b.  VM_num (number of VMs required to be placed)
2.  Initialize remaining capacity of each host as its associated Initial resource capacity.
3.  Repeat
4.      For Vm=1 to VM_num do
5.          Repeat
6.          Sort host list in random order
7.          Introduce a new host from host list
8.          Until (the remaining capacity of new host can provide enough resources for the VM)
9.      End For
10. Update remaining capacity of new host
11. Until Pop_num countries generate
```

پیوست ۱: شبه‌کد تولید جمعیت اولیه

Input: Initial population, Set of parameters
Output: Best found solution (country)

```
/*finding best solution*/
1. For each country in Initial population do
2.      Calculate power of country according to Eq. (1-4)
3. End For
4. Sort countries in descending order, according to their associative power
5. Set Imp_num of most powerful countries as Imperialists
6. Set remaining countries as Colonies
7. Distribute Colonies amongst Imperialists in respect to Imperialist's power
8. Repeat
9.      For each colony of each Imperialist do
10.         Assimilate the colony to its associative imperialist
11.         Calculate the power of assimilated colony
12.         If power of assimilated colony>power of associative imperialist Then
13.             set assimilated colony as new imperialist
14.             set imperialist as new colony
15.         End If
16.     End For
17. Calculate weakest colony of weakest Imperialist
18. Set the possession of weakest colony to other imperialist, who has the most likelihood to possess it.
19. If an imperialist with zero colony exists Then
20.     Eleminate the imperialist
21. End If
22. Until Decade=Decade_num
23. Calculate most powerful imperialist
24. Return most powerful imperialist
```

پیوست ۲: شبه‌کد الگوریتم رقابت استعماری

Input: a colony and its respective imperialist
Output: Assimilated colony

```
1.  For colony and its imperialist do
2.      set an array of distinct used host in colony or imperialist (colony_array, Imperialist_array)
3.      Calculate utilization of each used host
4.      Sort elements of each array in descending order in respect to each host's utilization
5.  End For
6.  Repeat
7.  For x=1to predefined value of x do
8.      Set the VM's associated to last host in sorted colony_array to the first host in sorted Imperialist_array.
9.  End For
10. Until the pre-defined number of iteration reached
11. Return assimilated colony
```

پیوست ۳: شبه‌کد حرکت مستعمره به‌سمت استعمارگر

مراجع

[1] Y. Gao, H. Guan, Z. Qi, Y. Hou, and L. Lio, "A multi-objective ant colony system algorithm for virtual machine placement in cloud computing," *Journal of Computer and System Sciences*, vol. 79, no. 8, pp. 1230-1242, 2013.

[2] S. Chaisiri, B. Lee, and D. Niyato, "Optimal virtual machine placement across multiple cloud providers," *Proceedings of the IEEE Asia-Pacific Services Computing Conference*, pp. 103-110, 2009.

[3] S. Agrawal, S. Bose, and S. Sundarrajan, "Grouping genetic algorithm for solving the server consolidation problem with conflicts," *Proceedings of the first ACM/SIGEVO Summit on Genetic and Evolutionary Computation*, pp. 1-8, 2009.

[4] S. Srikantaiah, A. Kansal, and F. Zhao, "Energy aware consolidation for cloud computing," *Proceedings of Hot Power'08 Workshop on Power Aware Computing and Systems*, 2008.

[5] E. Atashpaz-Gargari, and C. Lucas, "Imperialist competitive algorithm: an algorithm for optimization inspired by imperialistic competition," *IEEE Congress on Evolutionary Computation*, pp. 4661-4667, 2007.

[6] M. Morshed, and A. Asgharpour, "Hybrid imperialist competitive-sequential quadratic programming (HIC-SQP) algorithm for solving economic load dispatch with incorporating stochastic wind power: a comparative study on heuristic optimization techniques," *Journal of Energy Conversion and Management*, vol. 84, pp. 30-40, 2014.

[7] S. Shamshirband, A. Amini, N. Anuar, M. Kiah, T. Wah, and S. Furnell, "D-FICCA: a density-based fuzzy imperialist competitive clustering algorithm for intrusion detection in wireless sensor networks," *Proceeding of Journal of measurement*, vol. 10, 2014.

[8] A. Nourmohammadi, M. Zandieh, and R. Tavakkoli Moghaddam, "An imperialist competitive algorithm for multi-objective U-type assembly line design," *Journal of Computational Science*, vol. 4, no. 5, pp. 393-400, 2013.

[9] Y. Zhang, Y. Wang, and Peng, "Improved imperialist competitive algorithm for constrained optimization," *Proceedings of International Forum on Computer Science-technology and Applications*, Chongqing, vol. 1, pp. 204-207, 2009.

[10] M. Goldansaz, F. Jolali, and A. Zahedi Anaraki, "A hybrid imperialist competitive algorithm for minimizing makespan in a multi-processor open shop," *Applied Mathematical Modelling*, vol. 37, no. 23, pp. 9603-9616, 2013.

[11] E. Falkenauer, "A hybrid grouping genetic for bin packing algorithm," *Journal of Heuristics*, vol. 2, pp. 5-30, 2004.

[12] R. Gupta, S.K. Bose, and S. Sundarrajan., "A two stage heuristic algorithm for solving the server consolidation problem with item-item and bin-item incompatibility constraints," *Proceedings of IEEE International Conference on Service Computing*, vol. 2, pp. 39-46, 2008.

[13] M. Bichler, T. Setzer, and B. Speitkamp, "Capacity planning for virtualized servers," *Workshop on Information Technologies and Systems*, pp. 1-6, 2006.

[14] B. Speitkamp, and M. Bichler, "A mathematical programming approach for server consolidation problems in virtualized data centers," *IEEE Trans. Services Comput.*, vol. 3, no. 4, pp. 266-278, 2010.

[15] H. Duan, and L. Huang, "Imperialist competitive algorithm optimized artificial neural networks for UCAV global path planning", *Neurocomputing*, vol. 125, 11, pp. 166-171, 2014.

[16] A. Verma, P. Ahuja, and A. Neogi, "pMapper: power and migration cost aware application placement in virtualized systems," *Proceedings of the 9th ACM/IFIP/USENIX International Conference on Middleware*, pp. 243-264, 2008.

[17] D.V. Veldhuizen, *Multiobjective Evolutionary Algorithms: Classifications, Analyses, and New Innovations,* PhD thesis, Grad. School of Eng. of the Air Force Institute of Technology, Air University, 1999.

[18] E. Feller, L. Rilling, and C. Morin, "Energy-aware ant colony based workload placement in clouds," *Proceedings of the IEEE/ACM International Conference on Grid Computing (GRID)*, pp. 26-33, 2011.

[19] B. Li, J. Li, J. Huai, T. Wo, Q. Li, and L. Zhong, "Enacloud: an energy-saving application live placement approach for cloud computing environments," *Proceedings of the IEEE International Conference on Cloud Computing*, pp. 17-24, 2009.

[20] M. Dorigo, and L.M. Gambardella, "Ant colony system: a cooperative learning approach to the traveling salesman problem," *IEEE Transactions on Evolutionary Computation*, vol. 1, pp. 53-66, 1997.

[21] J. Békési, G. Galambos, and H. Kellerer, "A 5/4 linear time bin packing algorithm," *J. Comput. System Sci.*, vol. 60, no. 1, pp. 145-160, 2000.

[22] X. Fan, W. Weber, and L. Barroso, "Power provisioning for warehouse-sized computer," *Proceedings of the 34th Annual International Symposium on Computer Architecture*, pp. 13-23, 2007.

[23] M. Cardosa, M. Korupolu, and A. Singh, "Shares and utilities based power consolidation in virtualized server environments," *Proceedings of IFIP/IEEE Integrated Network Management (IM'09)*, pp. 327-334, 2009.

[24] B. Li, J. Li, J. Huai, T. Wo, Q. Li, and L. Zhong, "Enacloud: an energy-saving application live placement approach for cloud computing environments," *Proceedings of the IEEE International Conference on Cloud Computing*, pp. 17-24, 2009.

[25] Y. Ajiro, and A. Tanaka, "Improving packing algorithms for server consolidation," *Proceedings of the International Conference for the Computer Measurement Group (CMG)*, Computer Measurement Group, 2007.

[26] M.J. Sannella, *Constraint Satisfaction and Debugging for Interactive User Interfaces*, Ph.D. thesis, University of Washington, Seattle, WA, 1994.

[27] B. Li, J. Li, J. Huai, T. Wo, Q. Li, and L. Zhong, "EnaCloud: an energy-saving application live placement approach for cloud computing environments," *Proceedings of the IEEE International Conference on Cloud Computing*, pp. 17-24, 2009.

[28] R.N. Calheiros, R. Ranjan, A. Beloglazov, C.A.F.D. Rose, and R. Buyya, "CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," *Software: Practice and Experience (SPE)*, vol. 41, no. 1, pp. 23-50, 2011.

زیرنویس‌ها

[1] Platform

[2] Virtual Machines

[3] Packing Problems

[4] Imperialist Competitive Algorithm