نشریه علمی «نوآوری‌های فناوری اطلاعات و ارتباطات کاربردی»

سال اول، شماره ۲، تابستان ۱۴۰۰؛ ص ۳۳-۱۹

علمی – تخصصی

ارائه راهکاری جهت بهینه‌سازی در زمان و هزینه جایابی مؤلفه‌های نرم‌افزاری با استفاده از الگوریتم فرا اکتشافی چندهدفه در محیط ابری

مریم رضائی[۱]، مصطفی قبائی آرانی[۲*]

۱- گروه مهندسی کامپیوتر، واحد محلات، دانشگاه آزاد اسلامی، مرکزی، ایران ۲- گروه مهندسی کامپیوتر، واحد قم، دانشگاه آزاد اسلامی، قم، ایران

(دریافت: ۱۳۹۹/۱۲/۰۱، پذیرش: ۱۴۰۰/۰۵/۲۳)

چکیده

در دهه اخیر رایانش ابری مورد توجه بسیاری از ارائه‌دهندگان و استفاده‌کنندگان فناوری اطلاعات قرارگرفته است. یکی از مدل‌های پرکاربرد ارائه خدمات در حوزه رایانش ابری، مدل نرم‌افزار به‌عنوان خدمت بوده که معمولاً به‌صورت ترکیبی از مؤلفه‌های داده و برنامه ارائه می‌شوند. یکی از چالش‌های مهم در این حوزه، یافتن مکان بهینه برای مؤلفه‌های نرم‌افزاری بر روی زیرساخت‌های ابری است که در آن نرم‌افزار به‌عنوان خدمت بتواند بهترین عملکرد ممکن را داشته باشد. مسئله جایابی نرم‌افزار به‌عنوان خدمت به چالش تعیین اینکه کدام سرویس‌دهنده‌ها در مرکز داده ابر، بدون نقض محدودیت‌های نرم‌افزار به‌عنوان خدمت، می‌توانند میزبان کدام مؤلفه‌ها باشند اشاره دارد. در این مقاله، راهکار بهینه‌سازی چند هدفه با هدف کاهش هزینه و زمان اجرا جهت جایابی مؤلفه‌های در محیط‌های ابری را ارائه می‌دهیم. راهکار پیشنهادی خود را با استفاده از کتابخانه Cloudsim شبیه‌سازی کرده و در نهایت با دو الگوریتم ازدحام ذرات چندهدفه و فاخته مورد ارزیابی و مقایسه قرار دادیم. نتایج شبیه‌سازی نشان می‌دهد که راهکار پیشنهادی عملکرد بهتری نسبت به دو الگوریتم پایه داشته و موجب کاهش ۹/۴ درصدی زمان اجرای جایابی مؤلفه‌های نرم‌افزار به‌عنوان خدمت، کاهش ۹/۱ درصدی هزینه و افزایش ۷/۹ درصدی بهره‌وری می‌گردد.

کلیدواژه‌ها: رایانش ابری، مؤلفه‌های نرم‌افزاری ترکیبی، مدیریت منابع، جایابی SaaS، الگوریتم ژنتیک چند هدفه

۱- مقدمه

در دهه اخیر رایانش ابری موردپذیرش و استفاده بسیاری از مردم قرارگرفته است. منابع ابری و انواع مختلف خدمات ابری بر اساس تقاضا و برای دسترسی گسترده به شبکه کاربران آن، موجود می‌باشند، یکی از ویژگی‌های ابر انعطاف‌پذیری خدمات آن می‌باشد، یعنی می‌تواند متناسب با تقاضای کاربران باشند [۱] و کاربر می‌تواند از مدل پرداخت به ازای استفاده بهره‌مند گردد و کاربر طبق آن به‌جای پرداخت یک مقدار هزینه مشخص، تنها هزینه چیزی را که از ابر استفاده نموده را پرداخت نماید. خدمات رایانش ابر به سه دسته کلی طبقه‌بندی‌شده است. ۱- زیرساخت به‌عنوان خدمت[۱] (IaaS)، قابلیت ارائه و تأمین فضای زیرساختی برای ذخیره‌سازی، پردازش، شبکه و دیگر منابع ابری به‌اندازه‌ای که نرم‌افزار دلخواه بر روی آن مستقر شود و از آن‌ها استفاده کند، فراهم می‌کند. ۲- سکو به‌عنوان خدمت[۲] (PaaS) سکو به‌عنوان خدمت، قابلیتی است که چارچوبی برای محیط میزبانی و توسعه عملکردهای آماده اجرا موردنیاز شبکه‌ها، سرورها و ذخیره‌سازی ارائه می‌دهد [۲]. نرم‌افزار به‌عنوان خدمت[۳] (SaaS)، گونه‌ای از خدمات ارائه‌شده توسط محیط رایانش ابری است که با دیدگاهی جدید، به ارائه راهی مؤثر و هوشمندانه برای رفع نیاز تقاضامحور کاربر نهایی به منابع محاسباتی فراهم می‌کند [۳]. ما در این مقاله بر روی مشکلات حوزه SaaS تمرکز می‌کنیم، بسیاری از محققان به این امر اشاره دارند که SaaS قبل از رایانش ابری وجود داشته است [۴-۶] و تقاضای روزافزون برای استفاده از آن منجر شده است تا فروشندگان SaaS، آن را در محیط ابر گسترش دهند، زیرا ابر مقیاس‌پذیری موردنیاز SaaS را ارائه می‌دهد [۷]. این نوع از خدمات ابر، در مقایسه با خدمات دیگر ابر همچون PaaS و IaaS، در چند سال اخیر محبوبیت گسترده‌ای را به‌دست آورده است [۸ و ۹]. بررسی آمار نشان می‌دهد که درآمد اشتراک SaaS از تقریباً ۴۲ بیلیون دلار در سال ۲۰۱۲ به ۱۰۷ میلیارد دلار در سال ۲۰۱۶ افزایش داشته است [۹]. جهت انعطاف‌پذیری بیشتر SaaS، این نوع از خدمات را می‌توان به شکل

* رایانامه نویسنده مسئول: m.ghobaei@qom-iau.ac.ir

[1] Infrastructure-as-a-Service

[2] Platfrom -as- a-service

[3] Software- as -a- Service

مرکب ارائه داد. SaaS مرکب ترکیبی از زیرسیستم‌های مرتبط به یکدیگر می‌باشد که جهت تشکیل سیستم عملیاتی سطح بالا، به همدیگر وابسته می‌باشند [۱۰]. آن مؤلفه‌ها عموماً مؤلفه‌های برنامه (AC) می‌باشند که ممکن است به یک یا تعداد زیادی از بسته‌های داده نیاز به دسترسی داشته باشند، یا مؤلفه‌های داده (DC) ای هستند که همچنین به‌عنوان بخشی از SaaS مرکب در نظر گرفته می‌شوند. این شکل از SaaS ارائه داده‌شده در مدل مرکب باعث انعطاف‌پذیری بیشتر SaaS می‌گردد تا مناسب با تقاضای کاربران باشد، که این کار را از طریق ترکیب، جداسازی، و ترکیب دوباره مؤلفه‌ها در هنگام نیاز انجام می‌دهد. همچنین از طریق توانایی استفاده از مؤلفه SaaS در SaaS چندگانه دیگر، هنگامی که به عملکرد تجاری بخصوص آن نیاز است و بتوان آن را با مجموعه‌ای دیگر از مؤلفه‌ها "گردآوری" نمود، قابلیت استفاده مجدد را ممکن می‌سازد. هردو این مزیت‌ها می‌توانند منجر به صرفه‌جویی در هزینه و زمان برای ارائه‌دهندگان ابر شوند [۱۱و ۷]. با این‌حال، مدل مرکب جهت ارائه SaaS ممکن است باعث به وجود آمدن چندین چالش از نظر مدیریت منابع به ارائه‌دهنده ابر گردد. به‌عنوان مثال، یافتن راهکار جایابی برای SaaS مرکب که با کمینه‌سازی زمان اجرا و همچنین کاهش هزینه می‌تواند عملکرد آن را بهینه سازد.

این مسئله به‌عنوان مسئله جایابی SaaS (SPP)[1] شناخته‌شده و به‌عنوان مسئله مدیریت منابع رایانش ابری در نظر گرفته می‌شود. مدیریت منابع همیشه برای رایانش ابری چالش بوده است که این امر بخصوص در سال ۲۰۱۶ خود را بیشتر نشان داد، هنگامی‌که کمبود منابع و متخصصان به‌جای امنیت تبدیل به نگرانی اصلی برای ارائه‌دهندگان ابر شد [۸]، زیرا حجم بارکاری برای انتقال به ابر رو به افزایش بود. بسیاری از کارهای پژوهشی به چالش‌های مختلف مدیریت منابع در محیط ابر همراه با هوش محاسباتی (CI) و روش‌های بهینه‌سازی مختلف پرداخته‌اند [۱۱]. ما در این تحقیق با توجه به افزایش محبوبیت و استفاده از نرم‌افزار به‌عنوان خدمت توسط کاربران، مسئله مدیریت منابع ابر را مسئله اصلی تحقیق خود قرار می‌دهیم و با استفاده از الگوریتم‌های متفاوت به دنبال راه‌حل مناسبی برای حل مسئله جایابی مؤلفه‌های نرم‌افزار به‌عنوان خدمت می‌باشیم. با انجام شبیه‌سازی و مقایسه راهکار پیشنهادی با الگوریتم‌های دیگر که پیش‌تر استفاده‌شده بودند، این نتیجه حاصل می‌شود که SGA II[2] می‌تواند علاوه بر برآورده نمودن تقاضای کاربران، استفاده از منابع ابر را بهینه‌سازی کند. این الگوریتم برای جایابی اینکه کدام سرویس‌دهنده برای میزبانی کدام مؤلفه انتخاب شود، به‌گونه‌ای که تقاضای کاربران را برآورده کند و استفاده از منابع ابر را از طریق کاهش زمان اجرا و کاهش هزینه بهینه‌سازی کند و به یک جایابی مؤثر و کارا دست یابد، راهکار مناسبی می‌باشد. دستاوردهای اصلی این پژوهش به شرح زیر می‌باشد:

- ارزیابی کارایی الگوریتم پیشنهادی با دو الگوریتم MOPSO[3] و CSA[4] در دو سناریو متفاوت از نظر بهینه یودن در زمان اجرا و هزینه اجرایی
- ارائه الگوریتمی برای جایابی بهینه مؤلفه‌های SaaS مرکب در محیط ابری
- کاهش زمان اجرا برای جایابی بهینه مؤلفه‌های SaaS
- کاهش هزینه برای جایابی بهینه مؤلفه‌های SaaS
- افزایش بهره‌وری

ادامه این مقاله به‌صورت زیر سازمان‌دهی شده است. در بخش دوم به معرفی کارهای مربوطه که با الگوریتم‌های متفاوت برای جایابی بهینه انجام‌شده می‌پردازیم در بخش سوم راهکار پیشنهادی به‌وسیله الگوریتم NSGA II برای جایابی با اهداف بهینه‌سازی در زمان و هزینه ارائه می‌گردد. در بخش چهارم نتایج عملکرد جایابی با الگوریتم پیشنهادی در قالب نتایج شبیه‌سازی مورد ارزیابی قرارگرفته و با دو روش دیگر مقایسه می‌شود. درنهایت، در بخش آخر نتیجه‌گیری و پیشنهادها مطرح می‌گردد.

۲- کارهای مربوطه

اخیراً مسئله جایابی سرویس در چندین کار پژوهشی مورد بررسی قرارگرفته شده است که از بین آن‌ها چندین پژوهشگر رویکردهای هوشمند محاسباتی مختلف را جهت بهینه‌سازی عملکرد SaaS ارائه داده‌اند. محققان بهینه‌سازی را در دوشاخه جدا مورد بررسی قراردادند برخی محققان هدف بهینه‌سازی را کاهش هزینه‌های قراردادند که خواستن با کاهش هزینه‌های گسترش SaaS در سرورها و افزایش سود ارائه‌دهندگان ابر به آن بپردازند و برخی دیگر کاهش زمان اجرای کل را هدف بهینه‌سازی قرار دادند [۱۲]. کارها و آثار زیادی در زمینه جایابی SaaS با استفاده از الگوریتم‌های مختلف ارائه شده است که ما در اینجا به معرفی هریک از آن‌ها پرداخته و آن‌ها را شرح می‌دهیم.

۲-۱- راهکارهای جایابی SaaS با هدف بهینه‌سازی در زمان اجرا

Yusoh و Tang اولین کسانی بودند که مجموعه‌ای از پژوهش‌های جایابی را با مؤلفه‌های ناهمگن انجام دادند [۱۳-۱۵]. در اولین

[1] SaaS placement proplem
[2] None-dominated Sorting Algorithm
[3] Multi Particle Swarm Optimization
[4] Coco search algorithm

پژوهش الگوریتم ژنتیک (GA) مبتنی بر خطا با شروع تصادفی و تخمین[1] TETE را ارائه دادند که اگر راه‌حل تولیدی کار آیی نداشته و همچنین محدودیت‌های منابع را نیز نقض کند، الگوریتم آن‌ها به‌جای رهاسازی کامل، آن را "جریمه" کند. عیب این روش زمان اجرای طولانی بود. اولین پژوهش خود را با استفاده از رویکرد جدید مبتنی بر الگوریتم هم تکامل تعاونی (CCEA) بهبود بخشیدند. بررسی‌های مجدد نشان می‌دهند که CCEA موازی از نظر کیفیت راه‌حل‌های تولیدی و زمان محاسبه نسبت به GA و CCEA تکراری برتری دارد؛ اما مانند دو پژوهش پیشین، CCEA موازی تنها محدودیت‌های منابع را در نظر گرفته است.

Ni و همکاران [۱۶] الگوریتم ACO[2] را برای جایابی سرویس پیشنهاد دادند که از حرکت مورچه‌های مصنوعی و فرمون به جامانده در محیط جایابی انجام می‌شد. این روش از GA برتر بود اما با افزایش تعداد مؤلفه‌های SaaS به خوبی عمل نکرد و تفاوت‌های بین جایابی AC ها یا DC را در نظر نگرفتند.

Bowen و Shaochun [۱۷] از الگوریتم پیوندی انطباقی که GA را با گرم کردن شبیه‌سازی شده (SA) ترکیب می‌نماید استفاده کردند. نرخ جهش و هم‌گذری به گونه‌ای تنظیم شده‌اند تا بسته به نزدیکی مقادیر سازگاری راه‌حل‌های پیشنهادی تغییر کنند. طبق بررسی معرفی الگوریتم دوم باعث تسریع در بهینه‌سازی شده است.

Liu و همکاران [۱۸] دومین کسانی بودند که از رویکرد ACO استفاده نمودند که درحالی‌که ETET آن‌ها را به حداقل می‌رساند، مطلوبیت جایابی مؤلفه SaaS در یک ماشین بخصوص تحت تأثیر معادله درجه تطابق عملکرد (PMD) که خلاصه‌شده‌ی درجه نزدیکی بین نیازمندی منبع مؤلفه و عملکرد ماشین‌مجازی می‌باشد. این معادله PMD به به‌کارگیری مؤلفه فعلی در ماشین کمک می‌نماید که با قوی‌ترین عملکرد همراه است. عملکرد ACO نسبت به GA کلاسیک بهتر است، محققان تنها نیازمندی‌های منابع و مؤلفه‌های SaaS همگن ترمیم‌شده را در نظر گرفته‌اند.

Huang و Shen [۱۹]، مسئله جایابی را با هدف دوگانه (۱) کاهش هزینه‌های ارتباط بین کارها و (۲) افزایش همبستگی با ادغام دو گراف،[3] (SDG) و[4] (SCG) و تبدیل آن به مسئله kبخشی و حل آن با استفاده از الگوریتم حریصانه، مورد بررسی قرار دادند. مشکل این روش این بود که برای کاهش زمان اجرا به سرویس‌دهنده قوی نیاز بود که افزایش هزینه را به دنبال داشت و تفاوت بین جایابی DC و AC را نیز در نظر نگرفت.

Hajji و Mezni [۲۰] با تغییر در الگوریتم PSO، الگوریتم (PSO-CP) را ارائه دادند که در آن هر ذره مرکب در ازدحام نشان‌دهنده طرحی برای جایابی SaaS بود، ذرات مرکب جهت بررسی و ارزیابی هر فضا رفتار جمعی را اتخاذ می‌کنند (مانند مراکز داده) و از طریق همکاری با ذرات مرکب دیگر یا ذرات مستقل (مانند سرورها) ساختار خود را تنظیم می‌کنند. که نتایج آزمایش بر روی پیشنهاد آن‌ها نشان داد که PSO-CP نه‌تنها قابل‌اجرا در جایابی سرویس می‌باشد بلکه توانست راه‌حل‌های قابل قبولی را در فضای جستجو بزرگ را ارائه دهد و نسبت به GA و FFD[5] برتری داشت.

Mezni و همکارانش [۲۱] رویکرد (MS-SPP)[1] را برای حل مسئله جایابی بکار بردند و با افزایش امنیت که یکی از عوامل مؤثر بر کارایی می‌باشد به روشی برای جایابی SaaS که امنیت آگاه است دست یابند. آن‌ها یادگیری تعاونی به الگوریتم جایابی خود ترکیب کردند که اطلاعات بهترین سرویس‌دهنده‌های کاندید را برای جایابی کارآمدتر با توجه به ارزیابی خطرات امنیتی سرویس‌دهنده‌های میزبان بر اساس سطح آسیب‌پذیری، نمره امنیت برای هر میزبان و اجزای سازنده در نظر گرفتند تا آسیب‌پذیرترین سرویس‌دهنده‌ها شناسایی و در نهایت امن‌ترین سرویس‌دهنده‌ها با توجه به سطح عملکردی و زمان کلی اجرا انتخاب شوند؛ یعنی تابع ارزیابی بر اساس زمان کلی اجرا و ارزیابی امنیتی سرویس‌دهنده‌های نامزد را ارزیابی کند و آن‌هایی که با سطح امنیتی پایین‌تر هستند را کنار بگذارد. زمان اجرا در این روش در مقایسه با الگوریتم‌های FFD,GA,CP-PSO کمتر می‌باشد ولی زمان محاسبات با افزایش تعداد سرویس‌دهنده‌ها افزایش یافت.

۲-۲- راهکارهای جایابی SaaS با هدف بهینه‌سازی در هزینه با حداقل رساندن هزینه‌ها و افزایش سود

Kumar [۲۲] از GA مبتنی بر اصلاح جهت اصلاح راه‌حل‌ها و دستیابی به جمعیت دیگر و مناسب‌تر استفاده نمود. هر راه‌حل تولیدی که محدودیت‌های منابع را نقض کند از طریق تولید دوباره تصادفی بخشی از راه‌حل در فضای جستجو صحیح

[1] Total Execution Time
[2] Algorithm colony optimization
[3] Service Dependencies Graph
[4] Service Concurrence Graph
[5] First-Fit Decreasing

"اصلاح" می‌شد. در مقایسه این روش با (FFD) سود بیشتری به همراه داشت اما زمان محاسبه طولانی‌تر داشت و تفاوت بین جایابی AC و DC را در نظر نگرفته بود.

Bhardwaj [۲۳] رویکرد مبتنی بر (PSO) را پیشنهاد نمود. الگوریتم PSO به‌گونه‌ای شبیه به GA با یک جمعیت تصادفی آغاز شده و جهت تولید جمعیت بهتر تکامل می‌یابد اما در PSO هیچ اپراتور ژنتیکی اعمال نمی‌شود، زیرا این "ذرات"اند که جهت یافتن راه‌حل‌های بهینه در فضای جستجو پخش شده و با سرعت مشخص حرکت می‌کنند و پس از به‌روزرسانی سرعت و موقعیت‌شان، ذراتی که در جای جدید قرار گرفته‌اند، جمعیت جدیدی از راه‌حل‌های جایابی را ارائه می‌دهند و بر اساس مقادیر سازگاری جدیدشان، در صورت تغییر بهترین راه‌حل‌های محلی و کلی نیز به‌روزرسانی می‌شوند. PSO در مقایسه با GA کم‌هزینه‌تر و مقیاس‌پذیری بهتر داشت اما راجع به زمان پردازش الگوریتم هیچ بررسی تجربی وجود ندارد و تنها از سرویس‌دهنده‌ها و مؤلفه‌های همگن استفاده نموده است.

Altmann و Kashef [۲۴] یک الگوریتم بهینه‌سازی هزینه به نام COMBSP[1] برای تصمیم‌گیری در مورد جایابی خدمات در ابرها پیشنهاد دادند. آن‌ها مدل هزینه‌ای جامع که فاکتورهای هزینه و انواع ابرها را پوشش می‌داد، برای تصمیم‌گیری هزینه‌های خدمات با تخمین محاسبات هزینه بر اساس (فرمول هزینه ثابت، فرمول هزینه متغیر، فرمول هزینه کل) ارائه دادند که هزینه کلی همه‌ی گزینه‌های جایابی خدمات احتمالی را مقایسه و گزینه جایابی خدمات با حداقل هزینه را شناسایی می‌کند؛ که این الگوریتم هزینه انتقال اطلاعات بین ابرهای عمومی و هزینه استقرار خدمات که از طریق مهاجرت VM حاصل می‌شود. هزینه استقرار می‌تواند هزینه انتقال داده‌ها را جبران کند پس می‌توانیم بیان کنیم که افزایش تعداد مهاجرت خدمات می‌تواند تأثیر منفی در سود کل داشته باشد در این مورد به حداقل رساندن هزینه‌ها اجرا نخواهد شد.

W Chainbi ،E Sassi [۲۵] رویکردی مبتنی بر MPSO برای جایابی پیشنهاد دادند. نویسندگان بر اساس کارهای پیشین به این نتیجه رسیدند که pso در محیط‌های ایستا خوب عمل می‌کند و برای ایجاد نتایج قابل‌قبول در محیط‌های پویا مانند ابر باید سازگاری پیدا کند و این ناسازگاری به خاطر حافظه‌های منسوخ و فقدان تنوع در pso بود چون محیط ابر پویا است و درخواست‌ها و حجم منابع موردنیاز متغیر می‌بود آن‌ها دریافتند که رویکرد چندگانه در محیط‌های پویا بسیار سودمندتر خواهد بود و در فضای جستجو در چند منطقه نتایج امیدوارکننده‌تری ارائه می‌دهد و بر همین مبنا از الگوریتم MPSO استفاده کردند که البته این رویکرد را از لحاظ زمان اجرا با الگوریتم GA و از لحاظ هزینه با الگوریتم pso مقایسه کردند که به این نتیجه رسیدند که از هر دو کارآمدتر هست ولی مشکل این روش این بود که با افزایش تعداد سرویس‌دهنده‌های ابر هم هزینه و هم‌زمان اجرا افزایش پیدا می‌کرد.

۳- رویکرد پیشنهادی

در این مقاله راهکاری برای جایابی اجزای SaaS بر روی ابر با استفاده از الگوریتم ژنتیک چند هدفه ارائه می‌شود. در راه‌حل پیشنهادی الگوریتم شامل شش مرحله است، در روش پیشنهادی ما توابع هدف را زمان و هزینه بیان کردیم و هدف ما بهینه‌سازی در زمان و هزینه می‌باشد، تمامی فازها عملکرد خاصی دارند. به‌دلیل این‌که روش پیشنهادی بر پایه الگوریتم NSGA II می‌باشد در ادامه به‌صورت مشروح در مورد هر کدام از بخش‌ها توضیحاتی ارائه می‌گردد.

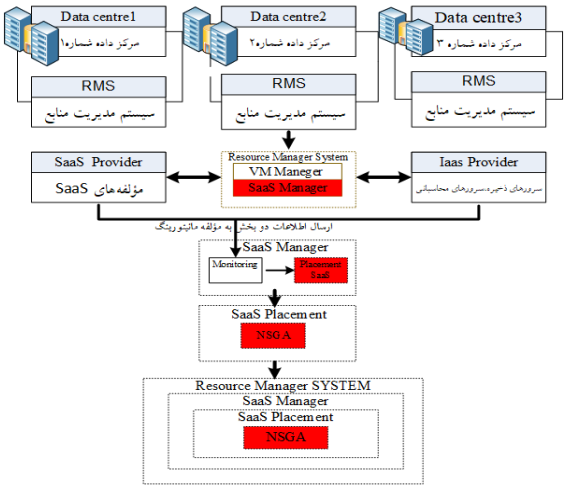
۳-۱- چارچوب پیشنهادی

ساختار معماری روش پیشنهادی بر اساس الگوریتم ژنتیک چند هدفه (NSGA II) دارای چند مرحله مدیریت بر روی منابع ابری می‌باشد. این مدیریت باید به‌گونه‌ای باشد که با رعایت SLA بتوانیم به حداقل زمان اجرا و حداقل هزینه در جایابی مؤلفه‌های SaaS بر روی محیط ابر دست‌یابیم، محیط ابری که برای اجرای SaaS به آن احتیاج داریم شامل چند مرکز داده ابری در مناطق مختلف می‌باشد که این مراکز شامل سرویس‌دهنده‌های ذخیره (SS)[2]، سرورهای محاسباتی (CS) هستند که هرکدام قابلیت منابع خاص خود را دارند و از طریق سوئیچ‌ها و مسیریاب‌ها به هم متصل می‌شوند ماشین‌های مجازی که همان VMها هستند بر روی این سرورها مستقر composite هایی که در اختیار SaaS providerها قرار دارند استفاده کنند SaaS composite ها شامل دو مؤلفه داده و برنامه هست و SaaS providerها که همان ارائه‌دهندگان SaaS می‌باشند شامل IaaS provider که همان زیرساخت ارائه‌دهنده برای اجرای SaaS می‌باشد و همان‌طور که ذکر کردیم این زیرساخت خود شامل سرویس‌دهنده ذخیره‌سازی و سرویس‌دهنده‌های محاسباتی هستند که سرویس‌دهنده‌ها در اختیار SaaS provider ها برای اجرای SaaS composite هایشان

[1] multi swarm SaaS Placement Program

قرار داده شده و تعدادی VM داریم که بر روی این سرویس‌دهنده‌ها مستقر شده‌اند پس به این نتیجه می‌رسیم که برای مدیریت منابع موجود بر روی سرورها احتیاج به یک مدیریت داریم تا بتواند اهداف ما را ارضا کند و همچنین SLA را رعایت کند و بتواند با حداقل زمان اجرا و حداقل هزینه با مدیریت خود بهترین جایابی ممکن مؤلفه‌های SaaS را بر روی سرورها انجام دهد، این مؤلفه سیستم مدیریت منابع (RMS) نام دارد که هر مرکز داده از طریق آن منابع مرتبطش را مدیریت می‌کند. در مرحله اول سیستم مدیریت منابع به دو بخش SaaS Manager و VM Manager تقسیم می‌شوند که SaaS Manager مسئول مدیریت بین مؤلفه‌های SaaS و زیرساخت ارائه‌دهنده ابر برای اجرای SaaS می‌باشد و VM Manager مسئول مدیریت ماشین‌های مجازی بر روی سرویس‌دهنده‌های فیزیکی هستند، تمرکز خود را بر روی SaaS Manager می‌گذاریم و به تفضیل به شرح مراحل مدیریتی آن می‌پردازیم. در مرحله دوم SaaS Compositeها به دو مؤلفه داده‌ها و برنامه‌های SaaS تقسیم می‌شوند، مؤلفه‌های برنامه باهم در ارتباط هستند و گاهی نیاز هست مؤلفه‌های برنامه با مؤلفه داده در ارتباط باشند، زیرساخت ارائه‌دهنده ابر شامل سرویس‌دهنده‌های ذخیره و سرویس‌دهنده‌های محاسباتی هست.

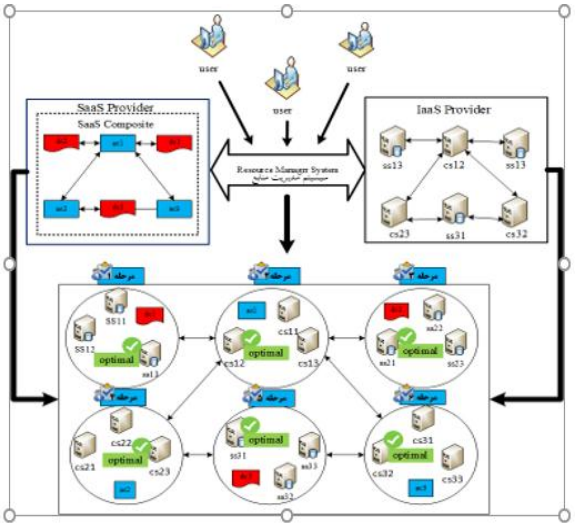
مؤلفه‌های برنامه بر روی سرویس‌دهنده‌ها محاسباتی و مؤلفه‌های داده بر روی سرویس‌دهنده‌های ذخیره قرار می‌گیرند پس نیاز هست سرورهای محاسباتی با همدیگر و گاهی با بعضی سرورهای ذخیره که مؤلفه داده موردنیاز بر روی آن مستقر هست در ارتباط باشند این انتخاب باید به‌گونه‌ای باشد که نزدیک‌ترین سرویس‌دهنده ذخیره به سرویس‌دهنده محاسباتی انتخاب شود تا به این وسیله زمان انتقال داده کاهش یابد و زمان اجرا به حداقل برسد پس برای این ارتباطات مدیریت لازم هست که SaaS Manager که در بین دو بخش SaaS providerها و IaaS provider قرارگرفته است این وظیفه را به عهده می‌گیرد و خود به دو زیر بخش Monitoring و SaaS placement تقسیم می‌شود که این نظارت با بخش مانیتورینگ انجام می‌شود و اطلاعات بین این دو بخش را نظارت می‌کند و به یک مؤلفه دیگر به نام SaaS placement که جایابیSaaS را که با الگوریتم ژنتیک چند هدفه (NSGA II) پیاده‌سازی شده می‌فرستد و آن مؤلفه به‌عنوان خروجی لیست جایابی به‌طوری‌که در زمان و هزینه بهینه اجرا می‌شود را ارائه می‌دهد. تفکیک مؤلفه‌های RMS را در شکل (۱) نشان دادیم.



شکل (۱): تفکیک مؤلفه‌های RMS

این مراحل سیستم مدیریت منابع را برای یافتن بهترین جایابی ممکن را با ذکر مثالی شرح می‌دهیم، زمانی که کاربری درخواستی را به ابر می‌فرستد تمام مراحل بالا انجام می‌شود با فرض این‌که SaaS composite شامل شش مؤلفه داده و برنامه که آن‌ها شامل مؤلفه‌های برنامه (ac_1,ac_2,ac_3) و مؤلفه‌های داده (dc_1,dc_2,dc_3) باشد برای انجام جایابی بهینه مراحلی انجام می‌شود مثلاً برای مؤلفه dc1 با بررسی ظرفیت منابع و ظرفیت منابع موردنیازش سه سرور ذخیره (ss_{11},ss_{12},ss_{13}) به‌عنوان کاندید در نظر گرفته می‌شود با محاسبه تابع برازش الگوریتم پیشنهادی خود به این به این نتیجه می‌رسیم که سرویس‌دهنده ذخیره ss_{13} از لحاظ زمان اجرا بهینه است پس این سرور ذخیره انتخاب می‌شود و مؤلفه داده dc_1 بر روی ماشین مجازی روی این سرویس‌دهنده ذخیره مستقر می‌شود. (۱) در جایابی بعدی مؤلفه‌های برنامه برای جایابی مؤلفه ac_1 با بررسی ظرفیت منابع و ظرفیت منابع موردنیازش سه سرویس‌دهنده محاسباتی (cs_{11},cs_{12},cs_{13}) به‌عنوان کاندید در نظر گرفته شده و از بین این سه سرویس‌دهنده محاسباتی با محاسبه تابع برازش از لحاظ زمان اجرا سرور cs12 بهینه است پس ac_1 بر روی ماشین مجازی روی این سرویس‌دهنده محاسباتی مستقر می‌شود (۲) در جایابی بعدی برای مؤلفه dc_2 با بررسی ظرفیت منابع و ظرفیت منابع موردنیازش سه سرویس‌دهنده ذخیره (ss_{21},ss_{22},ss_{23}) به‌عنوان کاندید در نظر گرفته می‌شود که بعد از محاسبه تابع برازش، سرویس‌دهنده ss_{21} از بین آن‌ها بهینه است پس dc_2 بر روی ماشین مجازی روی این سرویس‌دهنده ذخیره مستقر می‌شود (۳)، برای جایابی مؤلفه برنامه ac_2 با بررسی ظرفیت منابع و ظرفیت منابع موردنیازش سه سرویس‌دهنده محاسباتی (cs_{21},cs_{22},cs_{32}) به‌عنوان کاندید در نظر گرفته شده و با محاسبه تابع برازش سرویس‌دهنده cs_{23} بهینه است پس ac2 بر روی ماشین مجازی روی این سرویس‌دهنده محاسباتی مستقر می‌شود

(۴) در جایابی مؤلفه dc3 با بررسی ظرفیت منابع و ظرفیت منابع موردنیازش در بین سه سرویس‌دهنده ذخیره (SS_{31},SS_{32},SS_{33}) سرور ss_{31} بهینه است پس dc_3 بر روی ماشین مجازی روی این سرویس‌دهنده ذخیره انتخاب و مستقر می‌شود. (۵) در آخر برای مؤلفه ac_3 با بررسی ظرفیت منابع و ظرفیت منابع موردنیازش سه سرور محاسباتی (cs_{31},cs_{32},cs_{33}) در نظر گرفته شده و با محاسبه تابع برازش بین آن‌ها سرور cs_{32} بهینه است پس ac_2 بر روی ماشین مجازی روی این سرور محاسباتی مستقر می‌شود. (۶) تمام این مراحل جایابی را در شکل (۲) توضیح دادیم.



شکل (۲): چارچوب پیشنهادی

۳-۲- فرموله‌بندی مسئله

در این بخش متغیرها و فرمول‌های مورداستفاده در روش پیشنهادی مورد بررسی قرار می‌گیرد. جدول (۱) متغیرها و نمادهای مورداستفاده در روش پیشنهادی معرفی می‌گردد.

جدول (۱): متغیرها و تعاریف آن‌ها

نام	تعریف
DC	مرکز داده ابر
SS	سرور ذخیره
CS	سرور محاسباتی
E	لبه‌ی ارتباطی بین سرورها
N	تعداد سرور محاسباتی
VM_S	ماشین مجازی مستقر روی سرور
$VM_K.MIPS$	ظرفیت پردازش ماشین مجازی مستقر بر سرور محاسباتی
$VM_K.Dis$	ظرفیت ذخیره ماشین مجازی مستقر سرور محاسباتی
$VM_K.BW$	پهنای باند بین رئوس
$Latency_{S_{m+},S_{m-1}}$	تأخیر بین رئوس
S	مجموعه اجزای SaaS
AC	مجموعه مؤلفه‌های برنامه SaaS
DC	مجموعه مؤلفه‌های داده SaaS
$VM_K.MIPS$	نیازمندی ظرفیت ذخیره مؤلفه برنامه
$VM_K.Mem$	میزان خواندن نوشتن اجزا با یکدیگر
$VM_K.Disk$	گراف گردش کار
$Req_m.Data$	ارتباط مؤلفه برنامه با برنامه
DAG_s	ارتباط مؤلفه برنامه با داده
DAC	زمان اجرای کل
DDC	زمان پردازش
TET	زمان انتقال داده از سرور ذخیره به سرور محاسباتی
PT	ارتباط مؤلفه برنامه با برنامه
DTT	ارتباط مؤلفه برنامه با داده
$Cost$	هزینه
$VM_m.CostBW$	هزینه پهنای باند موردنیاز ماشین مجازی
$VM_m.CostMIPS$	هزینه پردازش موردنیاز ماشین مجازی
$VM_m.CostDisk$	هزینه ذخیره موردنیاز ماشین مجازی
$VM_m.CostMem$	هزینه حافظه موردنیاز ماشین مجازی
$Comp_{K_{MIPS}}$	مقدار سازگاری نهایی
$Comp_{K_{Mem}}$	مقدار سازگاری پردازنده
$Comp_{K_{Disk}}$	مقدار سازگاری ذخیره
$Comp_{K_{BW}}$	مقدار سازگاری پهنای باند
L	طول مسیر
K	تعداد درخواست‌ها

۳-۲-۱- مرکز داده

هر مرکز داده شامل تعدادی سرویس‌دهنده ذخیره و سرویس‌دهنده محاسباتی می‌باشد که از طریق سوئیچ‌ها و مسیریاب‌ها به هم متصل شده‌اند و ماشین‌های مجازی بر روی

این سرویس‌دهنده مستقر می‌باشند و هر کدام منابع خاص خود را دارند مثلاً سرویس‌دهنده محاسباتی شامل منابع پردازش، حافظه و ذخیره می‌باشند و سرویس‌دهنده ذخیره شامل منابع ذخیره هستند و ماشین‌های مجازی مستقر بر رویشان همین منابع را دارند، متغیر DC معرف مرکز داده ابر است که به‌صورت آرایه‌ای به‌صورت DC ={SSUCS,E } شناخته می‌شود؛ که سرور ذخیره با نماد SS، سرور محاسباتی با نماد CS و نشانگر ارتباط بین این دو سرور E می‌باشد. CS سرویس‌دهنده محاسباتی را با مجموعه CS={CS1,CS2,…,CSn} نمایش می‌دهیم که شامل n سرور محاسباتی می‌باشد، VM_S معرف ماشین‌های مجازی مستقر بر روی سرورهای محاسباتی می‌باشد که شامل تاپل (pci,memi,diski,VMCi) می‌باشد که pci معرف ظرفیت پردازش، memi معرف ظرفیت حافظه، diski معرف ظرفیت ذخیره، VMCi معرف مجموعه‌ای از ماشین‌های مجازی مستقر می‌باشد؛ و $VMC_i \subseteq VM$ می‌باشد. حال ماشین‌های مجازی مستقر بر روی سرورها را با نمادهای $(VM_K.MIPS,VM_K.Mem,VM_K.Disk)$ نمایش می‌دهیم، هر ماشین مجازی شامل ظرفیت پردازش که با نماد $VM_K.MIPS$ ،ظرفیت حافظه با نماد $VM_K.Mem$، ظرفیت ذخیره با نماد $VM_K.Disk$ نمایش داده می‌شود. حال به معرفی سرویس‌دهنده ذخیره و ویژگی‌هایش می‌پردازیم، SS سرویس‌دهنده ذخیره و با مجموعه SS={ss1,ss2,..,ssm} نمایش می‌دهیم که یعنی شامل m سرویس‌دهنده ذخیره بر روی مرکز داده ابر می‌باشد و ظرفیت سرویس‌دهنده ذخیره با نماد $disk_{ssi}$ نشان داده می‌شود که بیانگر ظرفیت ذخیره سرویس‌دهنده ذخیره می‌باشد که در محدوده یک تا m می‌باشد.، E لبه ارتباطی بین سرویس‌دهنده ذخیره و محاسبه در نظر گرفته می‌شود اگر ارتباطی بین رئوس باشد ما این رئوس گراف را با vi,vj نمایش می‌دهیم $B_{vi,vj}$ معرف پهنای باند بین رئوس و Lvi,vj معرف تأخیر بین رئوس می‌باشد .حال به معرفی اجزای SaaS می‌پردازیم. مجموعه اجزای SaaS را با نماد S نمایش می‌دهیم که شامل S=(AC,DC,DAC,DDC) می‌باشد.

۳-۲-۲- مؤلفه‌های SaaS

هر SaaS از مجموعه‌ای از برنامه‌ها و داده‌ها تشکیل شده است که هر کدام مقدار منابع خاص خودشان را نیاز دارند مثلاً مؤلفه‌های برنامه نیاز به منابعی برای پردازش، ذخیره، حافظه مختص به خود را دارند. مؤلفه‌های برنامه مستقر بر روی ماشین‌های مجازی که بر روی سرویس‌دهنده‌های محاسباتی قرار دارند با هم در ارتباط هستند و گاهی نیاز هست با مؤلفه‌های داده مستقر بر روی ماشین‌های مجازی که مستقر بر روی سرویس‌دهنده‌های ذخیره هستند در ارتباط باشند. AC={ac1,ac2,…,acx} معرف مجموعه‌ای از اجزای برنامه از یک تا x می‌باشد و هر مؤلفه برنامه شامل تاپل (Req.MIPS, Req.Mem, Req.Disk) می‌باشد که Req.MIPS معرف نیازمندی ظرفیت پردازش مؤلفه برنامه موردنظر Req.Mem معرف نیازمندی ظرفیت حافظه برای مؤلفه برنامه موردنظر، Req.Disk نیازمندی ظرفیت ذخیره برای مؤلفه برنامه موردنظر، Req.Data معرف میزان خواندن و نوشتن اجزای ارتباطی با یکدیگر است. DAGs بیانگر گراف گردش کار ما می‌باشد در اصل مجموعه وابستگی‌های بین مؤلفه‌های برنامه با همدیگر به نام DAC که به‌صورت DAC=AC*AC نمایش داده می‌شود و DDC=AC*DC که وابستگی یا ارتباط بین برنامه و داده را نمایش می‌دهد؛ که از طریق پهنای باند باهم در ارتباط هستند و نماد آن $VM_K.BW$ می‌باشد.

۳-۲-۳- محدودیت‌ها

در این جایابی سازگاری‌هایی لحاظ می‌کنیم سازگاری نهایی را با نماد $Comp_k$ معرفی می‌کنیم که باید برای جایابی مجموعه‌ای سازگاری‌ها رعایت شود و نباید آن‌ها را نقض کنیم و در صورت نقض این سازگاری‌ها عدد صفر و در صورت رعایتش عدد یک لحاظ می‌کنیم. این سازگاری در اصل معیاری برای مجموعه‌ای از سازگاری‌ها ازجمله سازگاری پردازشی، سازگاری حافظه، سازگاری ذخیره می‌باشد که بیان می‌کند ظرفیت پردازشی، ظرفیت حافظه، ظرفیت ذخیره ماشین مجازی باید بیشتر یا برابر میزان درخواست موردنیاز برای حافظه، ذخیره و پردازش باشد اگر کمتر باشد پس سازگاری نداریم پس عدد صفر می‌گذاریم که در معادله (۱) نشان دادیم. در نهایت برای محاسبه سازگاری کل همه‌ی سازگاری‌ها در هم ضرب می‌شوند و ماحصل یک عدد به‌دست می‌آید صفر یا یک اگر یک یعنی سازگاری داریم و عدد صفر یعنی ناسازگار پس جایابی انجام نمی‌شود که محاسبه را در معادله (۲) نشان دادیم.

$$Comp_{K_{MIPS}} = \begin{cases} 1 & VM_K.MIPS \geq Req.MIPS \\ 0 & Otherwise \end{cases} \quad (۱)$$

$$Comp_{K_{Mem}} = \begin{cases} 1 & VM_K.Mem \geq Req.Mem \\ 0 & Otherwise \end{cases}$$

$$Comp_{K_{Disk}} = \begin{cases} 1 & VM_K.Disk \geq Req.Disk \\ 0 & Otherwise \end{cases}$$

$$Comp_k = Comp_{K_{MIPS}} \times Comp_{K_{Mem}} \times Comp_{K_{Disk}} \times Comp_{K_{BW}} \quad (۲)$$

۳-۳- اهداف

هدف ما پیدا کردن بهترین جایابی ممکن برای مؤلفه‌های برنامه بر روی ماشین‌های مجازی مستقر بر روی سرویس‌دهنده‌های محاسباتی و مؤلفه‌های داده بر روی ماشین‌های مجازی مستقر بر روی سرویس‌دهنده ذخیره به‌طوری‌که زمان کلی اجرا و هزینه به حداقل برسد یعنی جایابی با هدف بهینه‌سازی در زمان و هزینه می‌باشد پس ما دو هدف کلی را دنبال می‌کنیم که در ذیل این اهداف را شرح می‌دهیم:

۳-۳-۱- هدف اول: زمان اجرای کل (TET)

برای محاسبه زمان اجرای کل باید مقادیر زمان انتقال داده (DTT) و زمان پردازش (PT) محاسبه گردد. در صورتی‌که تعداد درخواست‌ها K باشد محاسبات در مورد TET به‌صورت زیر خواهد بود.

$$DTT = \sum_{m=1}^{K} Comp_m \times \left(\frac{Req_m.Data \times 8}{BW_{S_{m+},S_{m-1}}} + Latency_{S_{m+},S_{m-1}}\right) \quad (۳)$$

در رابطه ۳ S_0 به‌عنوان محل شروع درخواست جایابی SaaS است. پس از محاسبه زمان انتقال، زمان پردازش محاسبه می‌شود.

$$PT = \sum_{m=1}^{K} Comp_m \times \left(\frac{Req_m.MIPS}{VM_m.MIPS} + Max(DTT)\right) \quad (۴)$$

در ادامه مسیری که برای اجرای کلیه درخواست‌ها باید طی شود با رابطه ۵ محاسبه می‌شود.

$$Path = set\ i \in \{1..k\} |\ Comp_i = 1 \quad (۵)$$

درنهایت با توجه مسیر طی شده، زمان اجرا به‌صورت زیر محاسبه می‌گردد.

$$TET = \sum_{m=1}^{L} PT \times Distance_{S_m,S_{m-1}} \quad (۶)$$

در رابطه (۶)، L طول مسیر یا تعداد سرورهای طی شده برای اجرا است.

۳-۳-۲- هدف دوم: هزینه (cost)

برای محاسبه هزینه از رابطه (۷) استفاده می‌شود.

$$Cost = \sum_{m=1}^{K} Comp_m \times ((DTT_m \times VM_m.CostBW) + (PT_m \times (VM_m.CostMIPS + VM_m.CostMe + VM_m.CostDisk)) \quad (۷)$$

۳-۴- الگوریتم پیشنهادی

در این بخش روش جایابی مؤلفه‌های SaaS در محیط رایانش ابری با استفاده از الگوریتم ژنتیک چند هدفه مورد بررسی قرار می‌گیرد. در روش پیشنهادی کروموزوم‌ها به‌عنوان اجزای SaaS ترکیبی در نظر گرفته می‌شود که هر کروموزوم شامل n مؤلفه برنامه و m مؤلفه داده می‌باشد که داخل هر کروموزوم شماره سرور به‌عنوان مشخصه‌ی کروموزوم لحاظ شده است که به‌صورت تصادفی تعدادی از کروموزوم‌ها انتخاب می‌شوند بعد با توجه به توابع هدف در نظر گرفته‌شده در الگوریتم مورد بررسی قرار می‌گیرند توابع ارزیابی کروموزوم‌ها را بررسی می‌کنند و کروموزومی که توابع ارزیاب آن‌ها را تأیید کند در جدول غلبه قرار می‌گیرند و بر اساس برتری به یکدیگر به‌صورت صعودی مرتب می‌شوند اگر دو کروموزوم تعداد غلبه یکسان داشته باشند در یک جبهه قرار می‌گیرند و بعد به‌صورت تصادفی یکی از آن دو به‌عنوان والد انتخاب می‌شود بعد از آن روی دو کروموزوم انتخاب‌شده عملیات تقاطع و جهش انجام می‌شود. الگوریتم ۱ شبه کد الگوریتم ژنتیک چند هدفه نامغلوب را نشان می‌دهد. بهترین تابع ارزیابی و بهترین جایابی در مرحله ۱ و ۲ آغازشده است. مرحله ۳ شامل جمعیت اولیه‌ای که به‌صورت تصادفی برای جایابی مؤلفه‌های داده و برنامه تولید شده است می‌باشد، بر اساس طرح کدگذاری استفاده‌شده این احتمال وجود دارد که ژن‌های نامعتبر تولید شده باشد در این وضعیت کروموزوم‌ها مرمت خواهند شد که در مرحله ۵ و ۶ این کار انجام می‌شود. مرحله ۷ تا ۱۱ مسئولیت ارزیابی هر یک از راه‌حل‌های کاندید را دارد که در آن بهترین جایابی و بهترین ارزیابی ذخیره می‌شود. در مرحله ۱۲ بر اساس توابع هدف الگوریتم جدول غلبه تشکیل و مرتب‌سازی نامغلوب صورت می‌گیرد در مرحله ۱۳ فاصله ازدحامی محاسبه و انجام می‌شود، در مراحل ۱۴ تا ۱۸ عملیات انتخاب، متقاطع و جهش انجام می‌شود. در ادامه به توضیح بیشتر روند اجرای الگوریتم می‌پردازیم.

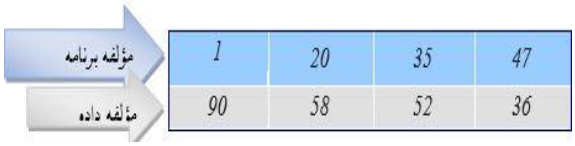
Algorithm 1: NSGA _Selection

۱. *best Fitness=0*
۲. best placement plan=0
۳. create and initial Population of Pop Size individuals that consist of solition for application components placement and data component placement
۴. while termination condition is not true do
۵. if X contains invalid gens then
۶. Repair(x)
۷. for x∈ population do
۸. Calculate X fitness value,F(TET),F(COST)penalized if $COMP_K$ SaaS resource requirement constraint
۹. If F(TET),F(COST)> **best Fitness then**
۱۰. Replace best fitness
۱۱. best Placement Plan=X

۱۲. Selection individuals from the Population based on fitness value sort
۱۳. Selection individuals from the Population based on crowding distance
۱۴. Selection individuals from the Population based on roulette wheel select
۱۵. Probabilistically apply the crossover operator to generate new individual
۱۶. Probabilistically select individuals for motion to replace the old
۱۷. Use the new individuals to replace the old individuals in the Population
۱۸. Output best Fitness
۱۹. Output best Placement Plan

۳-۴-۱- رمزگذاری ژنتیکی

یک کروموزوم در الگوریتم NSGA نشان‌دهنده SaaS ترکیبی، که شامل دو قسمت که m ژن مؤلفه داده و n ژن مؤلفه برنامه که باید داده‌ها بر روی سرویس‌دهنده ذخیره و برنامه‌ها بر روی سرویس‌دهنده محاسباتی جای بگیرند. در این رمزگذاری اطلاعات درون ژن‌ها شماره سرویس‌دهنده‌ها لحاظ شده‌اند. (اعداد که به‌عنوان شماره سرور در کدگذاری لحاظ کردیم به‌صورت فرضی و تصادفی است (شکل ۳)).



شکل (۳): یک نمونه از طرح رمزگذاری ژن و کروموزوم

۳-۴-۲- ارزیابی کروموزوم‌ها

با توجه به این‌که در روش پیشنهادی از الگوریتم ژنتیک چند هدفه استفاده می‌شود ما احتیاج به توابع هدف داریم عواملی مانند زمان اجرای کلی (TET) و هزینه (COST) یک مؤلفه در ماشین مجازی به‌عنوان توابع هدف در نظر گرفته می‌شوند و بر اساس مقدار آن‌ها جدول غلبه تشکیل می‌شود و بهترین انتخاب ممکن انجام شده و بهترین جایابی ارائه می‌شود؛ که در پیش‌تر نحوه محاسبه زمان اجرای کلی و هزینه را شرح دادیم.

۴- ارزیابی عملکرد و نتایج

در این بخش نتایج حاصل از پیاده‌سازی الگوریتم NSGA برای بهبود جایابی مؤلفه‌های SaaS مرکب در ابر با اهداف بهینه‌سازی در هزینه و زمان اجرای کل مورد بررسی قرار می‌گیرد. محیط مورداستفاده شبیه‌سازی NetBeans است. به‌منظور شبیه‌سازی دقیق و قابل توسعه ابر از ابزار Sim Cloud [۲۶] استفاده شده است. در ادامه به بررسی پارامترهای شبیه‌سازی، مورداستفاده قرارگرفته در این پژوهش و نتیجه شبیه‌سازی پرداخته می‌شود.

۴-۱- پارامترهای شبیه‌سازی

ساختار تمامی مراکز داده مورداستفاده در شبیه‌سازی در جدول مشخص شده است. ساختار Center Data مورداستفاده در شبیه‌سازی در جدول (۱) مشخص شده است.

جدول (۱): مشخصات مراکز داده

معماری	X64
سیستم‌عامل	Cloud Linux
مدیریت ماشین‌های مجازی	XEN

در Data Center ۱۰ گره یا میزبان وجود دارد. در جدول (۲) مشخصات میزبان یا ساختار فیزیکی به‌صورت جامع آمده است. به هر میزان سخت‌افزار میزبان قوی‌تر و از سطح بالاتری برخوردار باشد میزان هزینه دسترسی به منابع مربوط به VM موجود در میزبان افزایش می‌یابد.

جدول (۲): مشخصات میزبان (Host)

پهنای باند	حافظه اصلی (GB)	فرکانس (MIPS)	تعداد هسته
10 Gbit/s	۳۲	۴۰۹۶	۸

در جدول (۳) پارامترهای الگوریتم بهینه‌سازی NSGA را نمایش داده شده است.

جدول (۳): مشخصات الگوریتم بهینه‌سازی NSGA

تعداد جمعیت اولیه	تعداد نسل	نرخ جهش	نرخ ترکیب	شرط خاتمه
۵۰	۲۰۰	۰/۱	۰/۶	تعداد نسل

در جدول (۴ و ۵) به ترتیب مشخصات دو الگوریتم ازدحام ذرات چندهدفه و فاخته نشان داده شده است.

جدول (۴): مشخصات الگوریتم بهینه‌سازی MOPSO

تعداد جمعیت اولیه	تعداد نسل	ضریب جابه‌جایی	شرط خاتمه
۵۰	۲۰۰	۰/۶	تعداد نسل

جدول (۵): مشخصات الگوریتم بهینه‌سازی CSA

تعداد جمعیت اولیه	تعداد نسل	نرخ جهش	شرط خاتمه
۵۰	۲۰۰	۰/۷	تعداد نسل

جدول (۶) میزان تأخیر بین سه عامل اصلی در روش پیشنهادی و جدول (۷) مقادیر سخت‌افزار موردنیاز مؤلفه‌های SaaS و سرورها را نشان می‌دهد.

جدول (۶): میزان تأخیر در ارتباط بین سه عامل اصلی

ارتباط	میزان تأخیر (میلی‌ثانیه)
درخواست و سرورها	تصادفی توزیع نرمال بین ۵ تا ۲۰
سرورها با همدیگر	تصادفی توزیع نرمال بین ۳ تا ۱۰

جدول (۷): مقادیر سخت‌افزار موردنیاز مؤلفه‌های SaaS و سرورها

پردازنده	تصادفی بین ۵۰ تا ۲۰۰ MIPS
حافظه اصلی	تصادفی بین ۱۲۸ تا ۵۱۲ MBs
حافظه ذخیره‌سازی	تصادفی بین ۲ تا ۴ GB
پهنای باند	تصادفی بین ۲۰۰ تا ۵۰۰ Mbit/S
هزینه بابت مؤلفه سخت‌افزاری	تصادفی بین ۰/۱ تا ۰/۳ دلار

۴-۲- معیارهای کارآیی

شاخص‌های ارزیابی کل ساختار شامل هزینه و زمان اجرا است.

۴-۲-۱- هزینه

مجموع هزینه اختصاص ماشین مجازی و هزینه ناشی از جریمه به‌ازای رخداد تخطی از شرایط سرویس را میزان هزینه می‌گویند.

۴-۲-۲- زمان اجرا

زمان اجرا، تفاوت زمانی دقیق بین زمان درخواست جایابی و نیاز به اجرای SaaS و زمان تحویل SaaS می‌باشد.

۴-۲-۳- میزان تابع ارزش‌گذاری

مقدار حاصل‌شده از تابع ارزش‌گذاری در الگوریتم چندهدفه که متشکل از کاهش زمان اجرا و کاهش هزینه جایابی مؤلفه‌های SaaS می‌باشد.

۵- نتایج شبیه‌سازی

در این بخش نتایج شبیه‌سازی مورد بررسی قرار می‌گیرد و نتایج روش پیشنهادی با دو الگوریتم CSA و MOPSO مقایسه می‌شود.

۵-۱- ارزیابی و نتایج

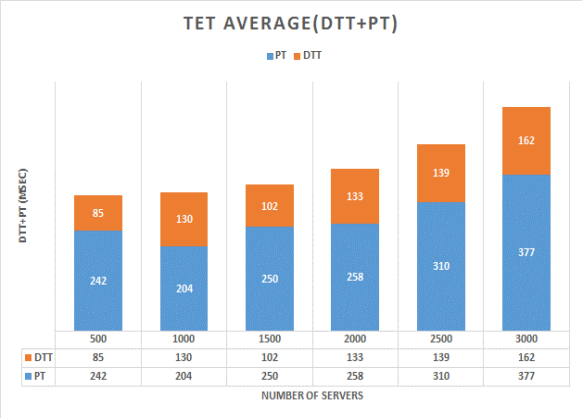
جهت ارزیابی روش پیشنهادی ۲ سناریو با ساختار جدول (۸) تشکیل شد. در شبیه‌سازی هر سناریو سه معیار مهم ارزیابی در سه روش مورد بررسی قرار می‌گیرد.

جدول (۸): سناریوهای مورد ارزیابی

سناریو	تعداد سرورها	تعداد مؤلفه‌های SaaS	هدف
یک	۵۰۰ تا ۳۰۰۰ (۲۵ درصد ذخیره‌سازی، ۷۵ درصد محاسباتی)	۴۰	اندازه‌گیری سه شاخص: مقدار تابع برازش، زمان پاسخگویی، مقایسه هزینه و میزان بهره‌وری منابع
دو	۳۰۰۰	۵۰ تا ۲۰۰ (۲۰ درصد DC و ۸۰ درصد AC)	

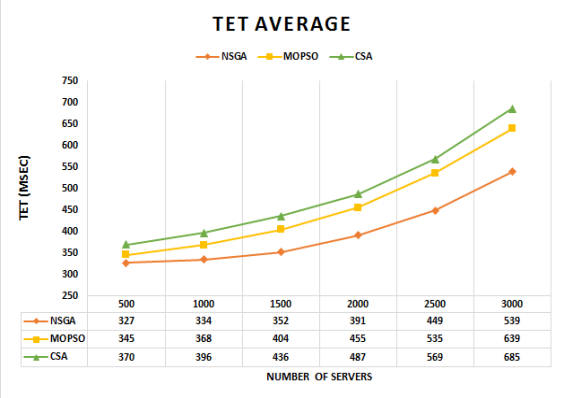
۵-۱-۱- سناریو اول

زمان اجرا به‌عنوان یک از مؤثرترین اهداف شرایط سرویس نقش مؤثری در تخصیص منبع دارد. نمودار (۱) مدت زمان دو شاخص DTT و PT را در زمان اجرا در روش پیشنهادی را نمایش می‌دهد.



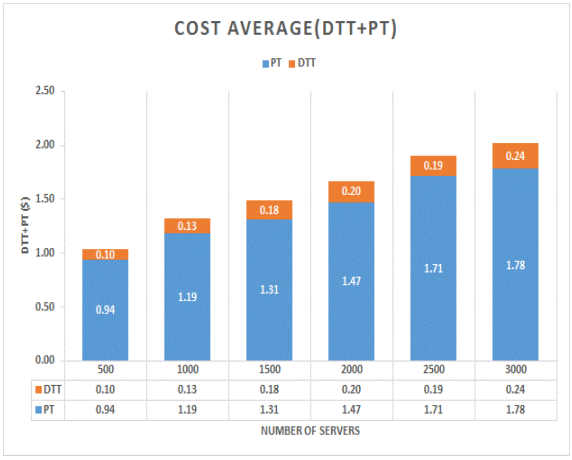
نمودار (۱): میانگین زمان انتقال و محاسبه در روش پیشنهادی

در صورتی‌که زمان اجرا موردنظر درخواست که با نام حد مجاز زمان پاسخگویی شناسایی می‌شود حاصل نشود، تخطی از شرایط پذیرش سرویس رخ داده است و کیفیت سرویس‌دهی از دید کاربر کاهش داشته است. در این بخش میانگین زمان پاسخگویی در روش پیشنهادی بررسی می‌شود و با دو الگوریتم CSA و MOPSO مقایسه می‌شود. نمودارهای (۲) میانگین زمان اجرا در طول شبیه‌سازی با تغییر سرورها از ۵۰۰ تا ۳۰۰۰ و تعداد ۴۰ مؤلفه SaaS را نمایش می‌دهد.



نمودار (۲): میانگین زمان اجرا در سناریوی اول

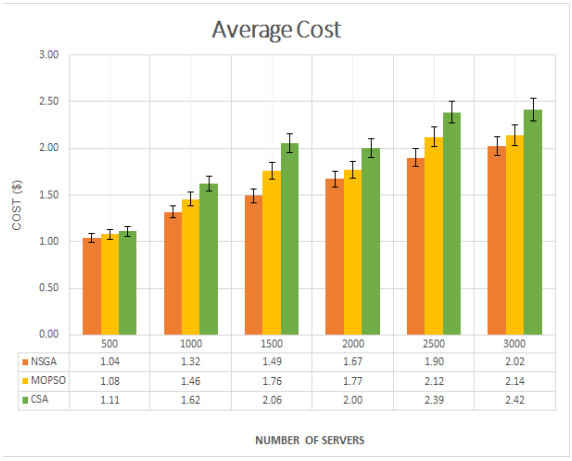
در نمودار (۲) میانگین زمان اجرا در روش پیشنهادی کمتر است. اصلی‌ترین شرط سرویس میزان هزینه مورد توافق کاربر است. در صورتی‌که هزینه سرویس‌دهی افزایش یابد، به‌طور مشخص سود سرویس‌دهنده کاهش و به همان میزان موجب نارضایتی کاربر از ساختار سرویس‌دهی می‌شود. در این بخش میانگین هزینه در روش پیشنهادی بررسی می‌شود و با دو الگوریتم CSA و MOPSO مقایسه می‌شود. نمودار (۳) میانگین هزینه را برای حالت انتقال (پهنای باند) و حالت محاسبه را به تفکیک نشان می‌دهد. نمودار (۴) میانگین هزینه را در شبیه‌سازی سناریو اول نمایش می‌دهد.



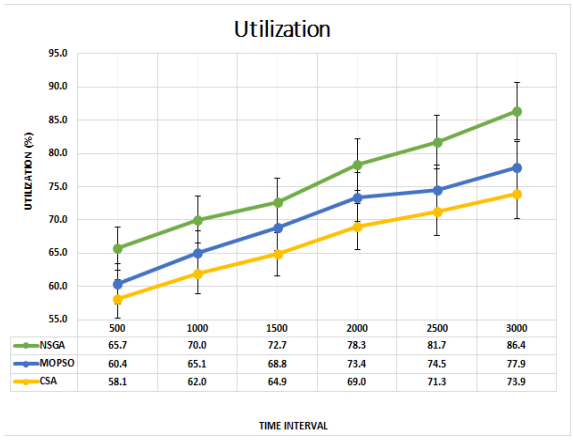
نمودار (۳): میانگین هزینه زمان انتقال و محاسبه

با توجه به نتایج، روش پیشنهادی کارآیی بهتری در مورد میانگین هزینه دارد. در الگوریتم پیشنهادی به‌دلیل در نظر گرفتن شرایط درخواست و قرار دادن هزینه به‌عنوان یکی از اهداف الگوریتم NSGA بیشترین دقت را در مورد هزینه تخصیص سرویس‌دهنده‌ها و جایابی SaaS اعمال می‌کند. استفاده از دو مرحله سختگیری در مورد زمان ارتباط و زمان محاسبه دقت در انتخاب و تخصیص سرویس‌دهنده را افزایش داده و به همین ترتیب هزینه موردنظر کاربران را برآورده نماید.

نمودار (۵) میزان بهره‌وری (میزان استفاده از منابع) را در سه روش مورد مقایسه نمایش می‌دهد.



نمودار (۴): میانگین هزینه در سناریو اول

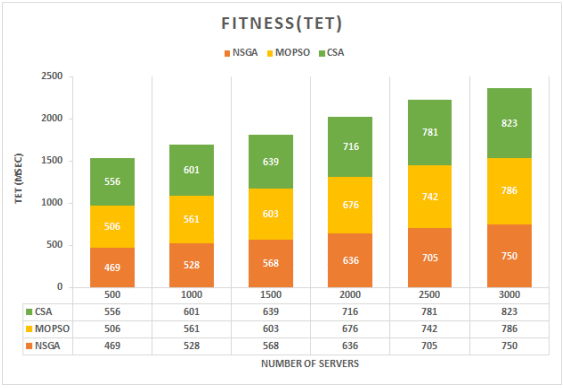


نمودار (۵): میانگین بهره‌وری از منابع در سناریو اول

به‌طور مشخص به‌دلیل تخصیص صحیح وظایف به مراکز داده، منابع پردازشگر با بهترین وضعیت در اختیار درخواست‌ها قرار می‌گیرد. البته ممکن است در تعداد درخواست بالا میزان بهره‌وری به آستانه یک برسد که دلیل آن تعداد درخواست‌های بالا است که نیاز به در اختیار گرفتن پردازنده دارند. استفاده از NSGA، تأثیر بسیار زیادی در کنترل مناسب درخواست‌ها و تصمیم‌گیری در مورد تخصیص مراکز داده شده است.

نمودار (۶ و ۷) میانگین میزان تابع برازش در مورد هدف هزینه و هدف زمان اجرا را نشان می‌دهد.

نمودار (۶): میانگین تابع برازش در مورد هزینه



نمودار (۷): میانگین تابع برازش در مورد زمان اجرا

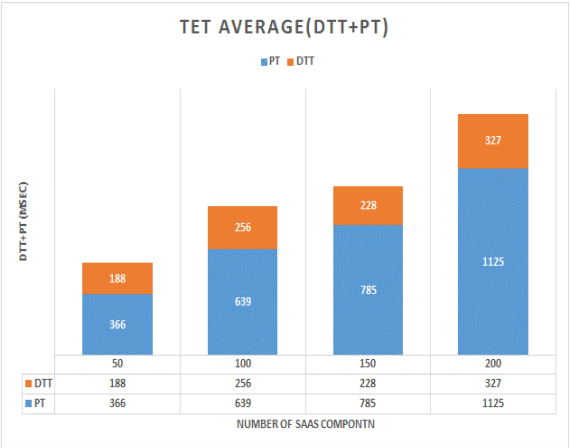
جدول (۹) میزان بهترین، میانگین و بدترین مقدار تابع برازش در تغییرات تعداد سرویس‌دهنده‌ها را در روش‌های مختلف نشان می‌دهد.

جدول (۹): جدول تغییرات تابع برازش

تعداد سرور		500		100		1500		2000		2500		3000	
زمان / هزینه	ه= هزینه، ز= زمان اجرا	ه	ز	ه	ز	ه	ز	ه	ز	ه	ز	ه	ز
NSGA	بهترین	۱/۰۶	۳۲۲	۱/۱۴	۴۱۲	۱/۱۸	۴۲۸	۱/۲۸	۵۱۲	۱/۳۹	۵۳۷	۱/۶۲	۵۹۲
	بدترین	۱/۶۸	۵۷۲	۱/۷۸	۶۰۳	۱/۹۲	۶۴۱	۱/۹۳	۶۷۸	۱/۸۸	۷۲۲	۲/۰۱	۸۱۲
	میانگین	۱/۵۲	۴۶۹	۱/۵۹	۵۲۸	۱/۶۳	۵۶۸	۱/۶۵	۶۳۶	۱/۶۷	۷۰۵	۱/۷۳	۷۵۰
MOPSO	بهترین	۱/۲۱	۳۷۲	۱/۲۹	۴۳۸	۱/۳۲	۴۶۱	۱/۴۰	۵۵۷	۱/۴۷	۵۶۹	۱/۶۷	۶۲۴
	بدترین	۱/۸۱	۵۸۴	۱/۹۰	۶۱۸	۱/۹۲	۶۵۸	۱/۸۹	۷۱۲	۲/۰۷	۷۸۲	۲/۲۱	۸۴۵
	میانگین	۱/۵۳	۵۰۶	۱/۶۷	۵۶۱	۱/۷۸	۶۰۳	۱/۶۷	۶۷۶	۱/۶۹	۷۴۲	۱/۷۸	۷۸۶
CSA	بهترین	۱/۲۹	۳۸۸	۱/۴۹	۴۵۱	۱/۴۹	۴۷۳	۱/۵۸	۵۹۲	۱/۶۶	۶۰۱	۱/۷۹	۶۸۹
	بدترین	۱/۸۴	۶۰۱	۱/۹۰	۶۲۷	۱/۹۸	۷۰۴	۱/۹۳	۷۷۹	۲/۱۶	۸۲۷	۲/۳۳	۹۱۹
	میانگین	۱/۶	۵۵۶	۱/۶۷	۶۰۱	۱/۷۸	۶۳۹	۱/۶۸	۷۱۶	۱/۷۱	۷۸۱	۱/۸۸	۸۲۳

۵-۱-۲- سناریو دوم

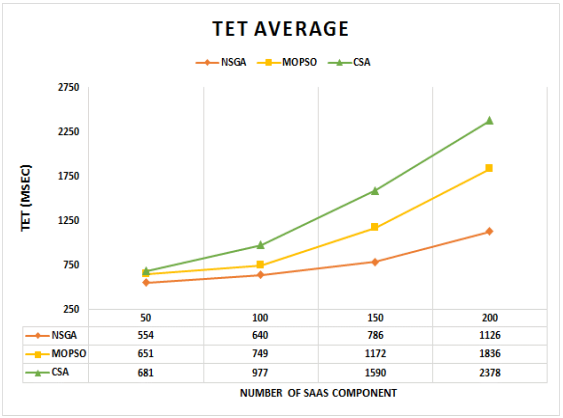
زمان اجرا به‌عنوان یکی از مؤثرترین اهداف شرایط سرویس یا SLA نقش مؤثری در تخصیص منبع دارد. نمودار (۸) مدت‌زمان دو شاخص DTT و PT را در زمان اجرا در روش پیشنهادی را نمایش می‌دهد. در صورتی‌که زمان اجرا موردنظر درخواست که با نام حد مجاز زمان پاسخگویی شناسایی می‌شود حاصل نشود، تخطی از شرایط پذیرش سرویس رخ داده است و کیفیت سرویس دهی از دید کاربر کاهش داشته است. در این بخش میانگین زمان پاسخگویی در روش پیشنهادی بررسی می‌شود و با دو الگوریتم CSA و MOPSO مقایسه می‌شود. نمودارهای (۸) میانگین زمان اجرا در طول شبیه‌سازی با تغییر مؤلفه‌های SaaS از ۵۰ تا ۲۰۰ و تعداد ۳۰۰۰ سرور را نمایش می‌دهد.



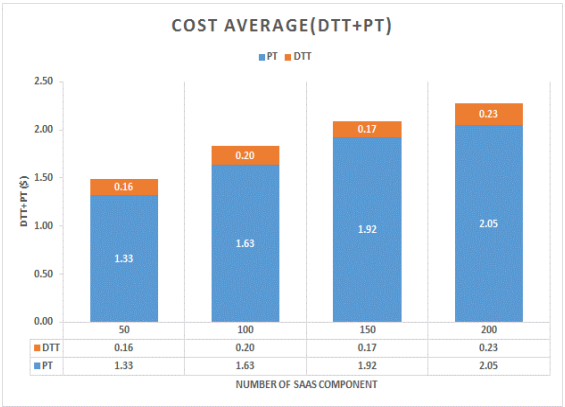
نمودار (۸): میانگین زمان انتقال و محاسبه در روش پیشنهادی

با توجه به نمودار (۹) میانگین زمان اجرا در روش پیشنهادی کمتر است.

اصلی‌ترین شرط سرویس میزان هزینه موردتوافق کاربر است. در صورتی‌که هزینه سرویس‌دهی افزایش یابد، به‌طور مشخص سود سرویس‌دهنده کاهش و به همان میزان موجب نارضایتی کاربر از ساختار سرویس‌دهی می‌شود. در این بخش میانگین هزینه در روش پیشنهادی بررسی می‌شود و با دو الگوریتم CSA و MOPSO مقایسه می‌شود. نمودار (۱۰) میانگین هزینه را برای حالت انتقال (پهنای باند) و حالت محاسبه را به تفکیک نشان می‌دهد. نمودار (۱۱) میانگین هزینه را در شبیه‌سازی سناریو دوم نمایش می‌دهد.



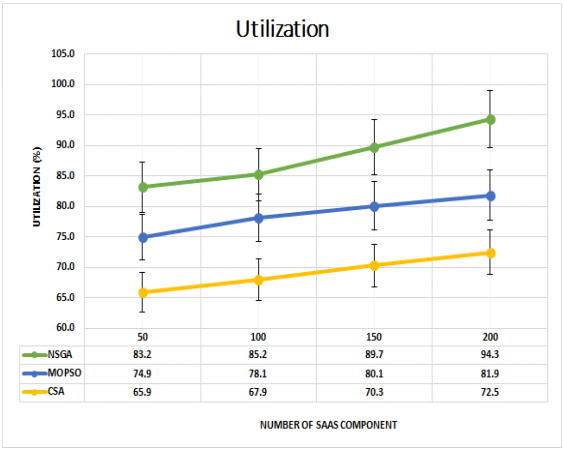
نمودار (۹): میانگین زمان اجرا در سناریوی دوم



نمودار (۱۰): میانگین هزینه زمان انتقال و محاسبه

با توجه به نتایج، روش پیشنهادی کارآیی بهتری در مورد میانگین هزینه دارد. در الگوریتم پیشنهادی به‌دلیل در نظر گرفتن شرایط درخواست و قرار دادن هزینه به‌عنوان یکی از اهداف الگوریتم NSGA بیشترین دقت را در مورد هزینه تخصیص سرورها و جایابی SaaS اعمال می‌کند. استفاده از دو مرحله سختگیری در مورد زمان ارتباط و زمان محاسبه دقت در انتخاب و تخصیص سرور را افزایش داده و به همین ترتیب هزینه موردنظر کاربران را برآورده نماید. نمودار (۱۲) میزان بهره‌وری (میزان استفاده از منابع) را در سه روش مورد مقایسه نمایش می‌دهد. استفاده از NSGA، تأثیر بسیار زیادی در کنترل مناسب درخواست‌ها و تصمیم‌گیری در مورد تخصیص مراکز داده شده است.

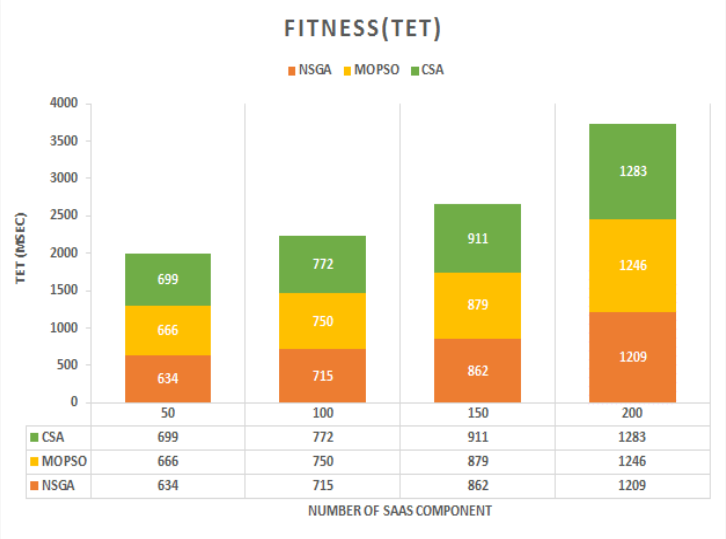
نمودار (۱۱): میانگین هزینه در سناریو دوم



نمودار (۱۲): میانگین بهره‌وری از منابع در سناریو دوم

نمودار (۱۳ و ۱۴) میانگین میزان تابع برازش در مورد هدف هزینه و هدف زمان اجرا را نشان می‌دهد.

نمودار (۱۳): میانگین تابع برازش در مورد هزینه



نمودار (۱۴): میانگین تابع برازش در مورد زمان اجرا

جدول (۱۰): جدول تغییرات تابع برازش

تعداد مؤلفه‌ها		۵۰		۱۰۰		۱۵۰		۲۰۰	
ه= هزینه، ز= زمان اجرا		ه	ز	ه	ز	ه	ز	ه	ز
NSGA	بهترین	۱/۴۶	۵۲۷	۱/۵۷	۵۸۱	۱/۷۴	۶۲۷	۱/۹۶	۱۱۵۸
	بدترین	۲/۰۷	۷۱۲	۲/۴۷	۷۷۱	۲/۵۱	۹۲۳	۲/۷۸	۱۳۲۸
	میانگین	۱/۸۶	۶۳۴	۲/۲۰	۷۱۵	۲/۲۷	۸۶۲	۲/۵۴	۱۲۰۹
MOPSO	بهترین	۱/۵۸	۵۴۹	۱/۷۸	۵۹۳	۱/۸۵	۶۹۸	۲/۰۷	۱۱۷۷
	بدترین	۲/۱۳	۷۲۸	۲/۸۷	۷۹۲	۲/۹۱	۹۴۸	۳/۱۸	۱۳۸۶
	میانگین	۲/۰۴	۶۶۶	۲/۳۷	۷۵۰	۲/۳۳	۸۷۹	۲/۵۷	۱۲۴۶
CSA	بهترین	۱/۷۲	۵۸۲	۱/۸۳	۶۱۰	۱/۹۲	۷۱۵	۲/۲۱	۱۱۹۲
	بدترین	۲/۲۱	۷۵۹	۲/۹۶	۸۱۴	۳/۰۸	۹۸۷	۳/۲۷	۱۴۰۵
	میانگین	۲/۱۱	۶۹۹	۲/۴۱	۷۷۲	۲/۴۷	۹۱۱	۲/۶۵	۱۲۸۳

۶- نتیجه‌گیری

در این مقاله الگوریتمی مبتنی بر ژنتیک چند هدفه NSGA برای جایابی مؤلفه‌های SaaS در محیط ابر با اهداف کاهش هزینه و کاهش زمان اجرا برای بهینه‌سازی جایابی ارائه شد، که در الگوریتم پیشنهادی مؤلفه‌های SaaS کروموزوم‌های ما در نظر گرفته شدند که هر کروموزوم شامل N ژن برنامه و M ژن داده که این ژن‌ها حاوی اطلاعات وراثتی برای انتقال به نسل بعد هستند می‌باشند. الگوریتم پیشنهادی شامل شماره سرورهای ابری می‌باشند که با تشکیل جدول غلبه و فاصله ازدحامی به بهترین انتخاب برای نسل بعد دست می‌یابیم که در مسئله جایابی در نهایت به بهترین جایابی ممکن می‌رسیم، با مقایسه و ارزیابی در دو سناریو متفاوت که یک‌بار تعداد سرویس‌دهنده‌ها متغیر و تعداد مؤلفه‌ها ثابت و یک‌بار تعداد سرویس‌دهنده‌ها ثابت و تعداد مؤلفه‌های SaaS متغیر جایابی را انجام دادیم و به این نتیجه رسیدیم که نسبت به دو الگوریتم دیگر برتری داشت و توانسته بود با کاهش زمان و هزینه به جایابی بهینه دست یابد. به‌منظور تکمیل و توسعه این تحقیق استفاده از روش‌های ترکیبی، به‌عنوان نمونه ترکیب برنامه‌نویسی پویا با الگوریتم‌های تکاملی دیگر، در مسئله‌های بزرگ درصورتی‌که تعداد فراهم‌کننده‌ها بالا باشد و به نسبت آن تعداد درخواست‌های زیادی نیز وجود داشته باشد. می‌توان با استفاده از برنامه‌نویسی پویا آن مسئله را به زیر مسئله‌ها شکست و هرکدام از آن‌ها را با الگوریتم‌های تکاملی به جواب بهینه رساند پیشنهاد می‌شود.

۷- مراجع

[1] P. Mell, G. Timothy, "The NIST definition of cloud computing", Technical report, National Institute of Standards and Technology, 2011.

[2] K. Chandrasekaran, "Essentials of cloud computing", CrC Press, 2014.

[3] A. Ghafouri, "Multi-tenant survey in cloud computing environments", Journal of Information Technology and Applied Communication Innovations, Vol.1, no. 1, 1398. (In Persian)

[4] S. Bhardwaj, "Service level agreement aware SaaS placement in cloud", (Doctoral dissertation), 2015.

[5] A. Kumar, "Placement of software-as-a-service components in cloud computing environment" (Doctoral dissertation), 2014.

[6] Z. Yusoh " Composite SaaS resource management in cloud computing using evolutionary computation. PhD thesis", Science and Engineering Faculty Queensland University of Technology Brisbane, Australia, 2013.

[7] K. Candan, Li. W-S, and T. Phan, M. Zhou "At the frontiers of information and software as services", In: New Frontiers in information and software as services, Springer, pp. 283–300, 2011.

[8] Right Scale IRight scale 2016 state of the cloud report. Technical report, RightScale Inc.2016.

[9] I. Statista, "Software as a service (SaaS)", subscription revenue from 2012 to 2016 by category (in billion u.s. dollars). http:// www.statista.com/statistics/468649/saas-software-subscriptionrevenue-by-category/. Accessed 22 March 2016.

[10] Cisco service-oriented network architecture: support and optimizesoaandweb2.0 applications. Technicalreport, Cisco Inc. 2008.

[11] G. TalbiE- ,G., BouvryP(2015)"A survey of evolutionary computation for resource management of processing incloud computing" [review article]. IEEE Comput Intell Mag 10(2):53–67. 2015.

[12] M. Rezaei and M. Ghobaei Arani, "An Approach Based on Multiobjective Genetic Algorithm (NSGA II) to placement Composite SaaS", 4the National Conference on Technology in Electrical and Computer Engineering, 2018. (In Persian)

[13] Z. Yusoh, and M. Tang, "A penalty-based genetic algorithm for the composite SaaS placement problem in the cloud", In IEEE Congress on Evolutionary Computation, pp. 1-8, IEEE. 2010.

[14] Z. Yusoh, and M. Tang, "A cooperative coevolutionary algorithm for the composite SaaS placement problem in the cloud", In International Conference on Neural Information Processing Springer, Berlin, Heidelberg, pp. 618-625, 2010.

[15] M. Yusoh, and Z. Izzah, "Composite SaaS resource management in cloud computing using evolutionary computation (Doctoral dissertation", Queensland University of Technology, 2013.

[16] Z.W. Ni, X.F. Pan, and Z. Wu, "An ant colony optimization for the composite SaaS placement problem in the cloud", In Applied mechanics and materials Trans Tech Publications, Vol. 130, pp. 3062-3067, 2012.

[17] Y. Bowen, and W. Shaochun, "An adaptive simulated annealing genetic algorithm for the data placement problem in SaaS." In 2012 International Conference on Industrial Control and Electronics Engineering, pp. 1037-1043, IEEE, 2012.

[18] Z. Liu, Z. Hu, and L. Jonepun, "Research on Composite SaaS Placement Problem Based on Ant Colony Optimization Algorithm with Performance Matching Degree Strategy", Journal of Digital Information Management, Vol. 12, no. 4, 2014.

[19] K. Huang, and B. Shen, "Service deployment strategies for efficient execution of composite SaaS applications on cloud platform", Journal of Systems and Software, Vol. 107, pp. 127-14, 2015.

[20] M. ajji, and H. Mezni, "A composite particle swarm optimization approach for the composite saas placement in cloud environment", Soft Computing, Vol. 22, no. 12, pp. 4025-4045, 2018.

[21] H. Mezni, M. Sellami, and J. Kouki, "Security-aware SaaS placement using swarm intelligence. Journal of Software", Evolution and Process, Vol. 30, no. 8, p. 1932, 2018.

[22] A. Kumar, "Placement of software-as-a-service components in cloud computing environment", (Doctoral dissertation), 2014.

[23] S. Bhardwaj, "Service level agreement aware SaaS placement in cloud", (Doctoral dissertation), 2015.

[24] J. Altmann. and M. Kashef, "Cost model based service placement in federated hybrid clouds", Future Generation Computer Systems, Vol. 41, pp. 79-90, 2014.

[25] W. Chainbi, And E. Sassi, "A multiswarm for composite SaaS placement optimization based on PSO. Software", Practice and Experience, Vol. 48, no. 10, pp.1847-1864. 2018.

[26] N. Calheiros, N. Rodrigo, R. Ranjan, A. Beloglazov, César AF De Rose, and B. Rajkumar "CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms", Software: Practice and experience, Vol. 41, no. 1, pp. 23-50, 2011.

Journal of "Innovations of Applied Information and Communication Technologies"
Vol. 1, No. 2, 2021 (Serial No. 2)

The Presentation of a Solution to Optimize the Time and Cost of Software Component Placement Using a Multi-Objective Meta-Exploratory Algorithm in the Cloud Environment

M. Rezaei, M. Ghobaei Arani*

Islamic Azad University of Qom

Abstract

In the last decade, cloud computing has attracted the attention of many IT providers and users. One of the most widely used models of providing services in the field of cloud computing is the "software as a service" or SaaS model, which is usually provided as a combination of data and application components. One of the major challenges in this area is finding the optimal location for the software components on the cloud infrastructure where the software as a service can perform at its best. The problem of locating software as a service, addresses the challenge of determining which components in the cloud data center can host which components without violating the limitations of the software as a service. In this paper, we have presented a multi-objective optimization solution with the aim of reducing costs and execution time for locating components in cloud environments. We have simulated our proposed solution using the Cloudsim library and finally evaluated and compared it with two multi-objective and cuckoo search algorithms. The simulation results show that the proposed solution performs better than the two basic algorithms, reducing the implementation time of the "software as a service" components and the costs by 9.4% and 9.1% respectively, and increasing the productivity by 7.9%.

Keywords: Cloud Computing, Hybrid Software Components, Resource Management, SaaS Placement, Multi Objective Genetic Algorithm

* Corresponding author E-mail: m.ghobaei@qom-iau.ac.ir