مقاله پژوهشی

محاسبات ابری سبز با کاهش مصرف انرژی در مهاجرت زنده اولویت‌دار سرویس‌ها

محمد رستمی و سلمان گلی بیدگلی

چکیده: امروزه رشد سریع تقاضا برای استفاده از منابع محاسباتی ابری، موجب افزایش مصرف انرژی در مراکز داده شده است. محاسبات ابری سبز برای جلوگیری از افزایش مصرف انرژی مراکز داده بیان شده است. یکی از راهکارهای کاهش مصرف انرژی در سیستم‌های پردازش ابری، استفاده از روش‌های تجمیع سرویس‌ها است. روش‌های تجمیع موجود با مهاجرت‌های غیر ضروری، عدم تعادل بار کاری میزبان‌ها و نادیده‌گرفتن ارتباط بین سرویس‌ها ممکن است باعث کاهش کیفیت سرویس و افزایش مصرف انرژی شود. لذا در این تحقیق با مهاجرت‌دادن سرویس‌های ضروری بر اساس اولویت (شامل تعداد فرزندان، سطح و هزینه ارتباطی هر سرویس)، از میزبان‌هایی که بارکاری خیلی زیاد و یا خیلی کم دارند (که موجب مصرف انرژی زیادی می‌شوند) به میزبان‌هایی که حاوی سرویس‌های همکار هستند، بهره‌وری منابع موجود مرکز داده بهبود یافته و مصرف انرژی کاهش می‌یابد. مهاجرت زنده سرویس‌ها بر اساس اولویت و به حداقل رساندن تعداد مهاجرت‌ها با هدف کاهش مصرف انرژی، کاهش زمان پاسخ و افزایش کارایی سیستم می‌باشد. با بررسی و مقایسه روش پیشنهادی با روش‌های دیگر، بهبود ۱۰٫۳۷ درصدی در کاهش مصرف انرژی در مراکز داده ابری، ۱۱٫۱۸ درصدی در کاهش تعداد مهاجرت سرویس‌ها و ۱٫٤٦ درصدی در افزایش تعداد میزبان‌هایی که خاموش شده‌اند، مشاهده می‌شود.

***کلیدواژه:* رایانش ابری، سرویس‌های همکار، کاهش مصرف انرژی، مهاجرت زنده سرویس‌ها.**

۱- مقدمه

رایانش ابری یکی از فناوری‌های نوظهور است که فناوری اطلاعات را با تخصیص منابع محاسباتی بر اساس تقاضا متحول نموده است. با توجه به افزایش هزینه سخت‌افزاری و منابع محدود ذخیره‌سازی داده‌ها، طراحان شبکه به فکر ایجاد مراکز داده ابری افتاده‌اند. مراکز داده ابری در بستر شبکه امکاناتی برای کاربران فراهم می‌کنند تا با توجه به درخواست‌های آنها، منابعی مورد نیاز و متناسب با هزینه‌ای که می‌پردازند به صورت پویا در اختیار آنها قرار دهند. همین امر موجب گسترش مفهوم رایانش ابری در ایجاد مراکز داده با مقیاس بزرگ در سراسر جهان شده است. مراکز داده در محاسبات ابری مقدار زیادی از انرژی الکتریکی توسط تجهیزات و سخت‌افزارهای کامپیوتری، سیستم‌های خنک‌کننده، روشنایی، تهویه و ذخیره‌سازی (دیسک‌ها) را مصرف می‌کنند و این به نوبه خود منجر به بالارفتن دما و افزایش هزینه‌های عملیاتی و تولید دی اکسید کربن در محیط پیرامون می‌شود. کاهش انتشار دی اکسید کربن در محیط اطراف و محدود بودن ذخایر منابع انرژی نیاز به تکنیک‌هایی برای کاهش مصرف انرژی در مراکز داده بیش از پیش پر رنگ می‌شود. محاسبات ابری سبز باعث کاهش هزینه‌های عملیاتی (به عنوان مثال کاهش قابلیت اعتماد، طول عمر منابع سخت‌افزاری، افزایش بهای خدمات و بار زیاد سیستم خنک‌کننده) و اثرات زیست‌محیطی می‌گردد. همچنین محاسبات ابری سبز قصد دارد تا با طراحی الگوریتم‌ها و استراتژی‌هایی برای کاهش مصرف انرژی از منابع به شکلی مؤثرتر و مقرون به صرفه‌تر از لحاظ انرژی، در مراکز داده استفاده کند.

در مراکز داده ابر، تعداد زیادی از ماشین‌های فیزیکی از طریق یک شبکه متصل می‌شوند. این ماشین‌های فیزیکی به تعدادی از ماشین‌های مجازی که در حال اجرای انواع برنامه‌های کاربردی می‌باشند، تخصیص داده می‌شوند. هدف از این تخصیص، کمینه‌کردن مصرف انرژی و افزایش بهره‌وری است. هر دستگاه فیزیکی منابع سخت‌افزاری خاص خود را دارد (به عنوان مثال پردازنده، حافظه و پهنای باند شبکه). ظرفیت منابع ماشین‌های فیزیکی و درخواست (نیاز) منابع ماشین‌های مجازی به طور کلی متفاوت است و بنابراین در تخصیص ماشین‌های مجازی به ماشین‌های فیزیکی باید محدودیت‌های سخت‌افزاری هر ماشین فیزیکی را در نظر گرفت [۱]. یک سناریو می‌تواند به این صورت باشد که N ماشین مجازی[1] به M ماشین فیزیکی با هدف به حداقل رساندن مصرف انرژی و بهبود بهره‌وری منابع تخصیص یابد.

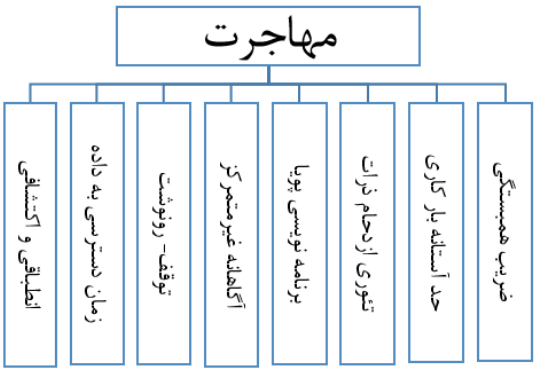
همان طور که بیان شد یکی از روش‌های افزایش بهره‌وری در محیط ابر استفاده از تکنیک‌های مجازی‌سازی است که امکان اشتراک منابع محدود فیزیکی را بین ماشین‌های مجازی مراکز داده ابری فراهم می‌کند. در این راستا، یکی از تکنیک‌های اصلی در مجازی‌سازی، استفاده از مهاجرت (انتقال) سرویس است که به سرویس اجازه انتقال به ماشین مجازی دیگری را می‌دهد. عمل مهاجرت با اهداف گوناگونی از جمله توازن و تقسیم بار، تحمل‌پذیری در برابر خرابی، مدیریت انرژی، کاهش زمان پاسخ و افزایش کیفیت سرویس و نگهداری سرورها انجام می‌شود. مهاجرت از دید سیستم عامل مهمان، برنامه‌های کاربردی در حال اجرا و کاربران راه دور ماشین مجازی مخفی است [۲]. سرویس‌هایی که به صورت مداوم نیاز به ارتباط جهت تبادل پیام دارند، سرویس‌های همکار نام برده می‌شوند. در واقع سرویس‌های همکار با مهاجرت‌نمودن در کنار هم برای رسیدن به یک هدف مشترک فعالیت می‌کنند.

این مقاله در تاریخ ۱۰ فروردین ماه ۱۳۹۹ دریافت و در تاریخ ۱۳ دی ماه ۱۳۹۹ بازنگری شد.

محمد رستمی، دانشکده مهندسی کامپیوتر، دانشگاه کاشان، کاشان، ایران، (email: mrostami@grad.kashanu.ac.ir).

سلمان گلی بیدگلی (نویسنده مسئول)، دانشکده مهندسی کامپیوتر، دانشگاه کاشان، کاشان، ایران، (email: salmangoli@kashanu.ac.ir).

1. Virtual Machine



شکل ۱: نمودار دسته‌بندی روش‌های مهاجرت.

در مراکز داده بزرگ سرویس‌های مختلفی وجود دارد که توسط سرورهای متفاوتی ارائه می‌شوند. ارتباطات سرویس‌ها بین میزبان‌های مختلف باعث هدررفتن منابع و افزایش مصرف انرژی می‌شود و بنابراین چنانچه بتوان سرویس‌هایی که با هم ارتباط زیادی دارند (سرویس‌های همکار) در یک میزبان قرار داد، هزینه ارتباطی (به عنوان مثال پهنای باند و هزینه رد و بدل نمودن داده) بین سرویس‌های همکار کاهش یافته، تعداد مهاجرت سرویس‌ها به حداقل رسیده و در مصرف انرژی نیز صرفه‌جویی می‌شود. اولویت‌دهی به سرویس‌های ماشین‌های مجازی و انتخاب سرویس مناسب برای مهاجرت از موارد مطروح روش پیشنهادی در افزایش محلی‌سازی ارتباطات سرویس‌ها، افزایش کارایی، کاهش تعداد مهاجرت‌ها، کاهش زمان پاسخ و به دنبال آن کاهش مصرف انرژی مراکز داده ابری می‌باشد (اولویت با مقدار عددی کم به عنوان اولویت بالا در نظر گرفته می‌شود).

بیشتر تحقیقات پیشین از روش‌های حداقل زمان مهاجرت، سیاست انتخاب تصادفی و حداقل زمان بهره‌وری استفاده نموده‌اند [۳] تا [۶] که در طی آن پارامترهای مصرف انرژی، تعداد مهاجرت‌ها و تعداد میزبان‌هایی که خاموش شده‌اند مورد مقایسه و بررسی قرار گرفته است که نتایج نشان‌دهنده عملکرد بهتری از لحاظ کاهش تأخیر و مصرف انرژی (حفظ کیفیت سرویس) می‌باشد. روش پیشنهادی در این مقاله، ضمن تضمین رعایت توافق‌نامه سطح سرویس (SLA)[1]، سعی در افزایش رضایت کاربر داشته است. همچنین زمان پاسخ را به حداقل رسانده و باعث افزایش بهره‌وری منابع و کارایی سیستم نسبت به روش‌های قبل می‌شود. بنابراین از مهم‌ترین دستاوردهای این مقاله می‌توان به موارد زیر اشاره کرد:

- ارائه روش اولویت‌دار برای محاسبه اولویت سرویس‌ها
- ارائه روشی برای تجمیع سرویس‌های همکار
- مهاجرت سرویس‌ها با رویکرد کاهش مصرف انرژی
- به حداقل رساندن تعداد مهاجرت سرویس‌ها

در ادامه، در بخش دوم پژوهش‌های پیشین مورد بررسی قرار گرفته و در بخش سوم روش پیشنهادی تشریح شده است. بخش چهارم به ارزیابی و تفسیر نتایج اختصاص دارد و نهایتاً مقاله در بخش پنجم با نتیجه‌گیری و پیشنهادهایی برای آینده به اتمام می‌رسد.

۲- مروری بر پژوهش‌های پیشین

در سال‌های اخیر پژوهش‌های بسیاری در رابطه با مهاجرت ماشین مجازی و تخصیص منابع در راستای کاهش مصرف انرژی انجام گرفته است. نمودار دسته‌بندی روش‌های مهاجرت در شکل ۱ نشان داده شده است. در ادامه به برخی از مهم‌ترین روش‌ها اشاره می‌شود.

در [۲]، روش مهاجرت ماشین مجازی به نام روش توقف- رونوشت بیان شده که در آن ماشین مجازی به منظور جلوگیری از تغییر محتوای حافظه کاری ماشین مجازی، پیش از انتقال، حافظه آن متوقف شده و پس از قرارگیری در ماشین فیزیکی مقصد، دوباره اجرا می‌شود. پس از اتمام کار رونوشت منابع لازم از ماشین مقصد در اختیار ماشین مجازی قرار گرفته و ماشین مجازی به اجرای خود در ماشین مذکور ادامه می‌دهد. از مزایای این روش سادگی پیاده‌سازی است و از معایبش می‌توان به عدم در دسترس بودن ماشین مجازی در هنگام انتقال اشاره نمود که می‌تواند بر روی کیفیت سرویس تأثیر منفی بگذارد.

در [۳]، یک روش انطباقی و اکتشافی برای مهاجرت ماشین‌های مجازی و خاموش‌کردن ماشین‌های فیزیکی غیر ضروری با استفاده از تاریخچه مصرف منابع توسط ماشین مجازی ارائه شده است. الگوریتم ارائه‌شده به صورت قابل توجهی مصرف انرژی را کاهش می‌دهد، در حالی که این کاهش مصرف انرژی باعث نقض SLA نمی‌شود.

در [۷]، ماشین‌های مجازی که کمترین ضریب همبستگی را دارا می‌باشند، در میان میزبان‌ها مهاجرت داده می‌شوند. معیارهای همبستگی شامل درصد استفاده از پردازنده، حافظه و پهنای باند است. ضریب همبستگی برای نشان‌دادن میزان ارتباط بین ماشین مجازی انتخاب شده و میزبان هدف می‌باشد. اگر ضریب همبستگی بیشتر باشد تأثیر بیشتری بر روی کارایی ماشین‌های مجازی دیگر دارد. با مهاجرت ماشین مجازی با حداقل ضریب همبستگی به میزبان هدف، از کاهش کارایی ماشین‌های مجازی دیگر جلوگیری می‌نماید. در این تحقیق علاوه بر نگرش جدی به پردازنده در تخصیص منابع از پارامترهای مصرف انرژی، زمان مهاجرت ماشین مجازی و نقض SLA در تخصیص و جابه‌جایی ماشین مجازی استفاده شده است. مصرف انرژی یک ماشین فیزیکی با استفاده از مصرف پردازنده تعیین می‌شود.

در [۸]، با الگوبرداری از شبکه واقعی، الگوریتمی برای تلفیق ماشین‌های مجازی معرفی شده است. در این تخصیص پارامتر رک[2] (مکان نگهداری سرورها) را در جابه‌جایی ماشین‌های مجازی استفاده می‌کنند تا جایگذاری بهتری انجام دهند. هر رک دارای چندین سرور است و هر سرور دارای یک واحد برق اختصاصی و فن خنک‌کننده می‌باشد. تعداد خنک‌کننده‌ها و ساختار شبکه مراکز داده ماشین‌های فیزیکی در هنگام تلفیق ماشین‌های مجازی در نظر گرفته می‌شود تا رک‌ها و روترهای کمتری بدون این که SLA نقض شود، به کار روند. در نتیجه می‌توان تجهیزات مسیریابی و خنک‌کننده را برای کاهش مصرف انرژی خاموش نمود که باعث بهبود مصرف انرژی سرورها، تجهیزات شبکه و سیستم‌های خنک‌کننده می‌شود.

در [۹]، با توجه به تقاضای گسترده منابع ابری، استفاده بهینه از منابع به هدف اصلی تأمین‌کننده ابر تبدیل شده است. برای تحقق این هدف، یک روش تعادل بار نیاز است که منابع ابر را به درستی توزیع کند زیرا عدم تعادل بار می‌تواند منجر به افزایش مصرف انرژی گردد. یک روش مهاجرت ماشین مجازی برای تعادل بار با رویکرد حد آستانه استفاده شده است. مهاجرت ماشین مجازی به دلیل بار اضافی موقتی یکی از موارد مهم در ابر است که عملکرد سرویس‌های ابری را به شدت تحت تأثیر قرار می‌دهد. برای جلوگیری از مهاجرت‌های غیر ضروری، از یک روش

1. Service Level Agreement

2. Rack

پیش‌بینی کننده استفاده می‌شود که بار را قبل از شروع مهاجرت‌ها پیش‌بینی کند. نتایج نشان می‌دهند که این روش باعث کاهش تعداد مهاجرت‌ها شده است.

در [۱۰]، یک الگوریتم اکتشافی[1] برای روش تئوری بهینه‌سازی ازدحام ذرات معرفی شده است. با تجزیه و تحلیل در مورد سرور فیزیکی مرکز داده و ویژگی‌های ماشین مجازی، توابع محدودیت، بهینه‌سازی استفاده از منابع و تعداد مهاجرت ماشین مجازی به دست آورده شده است. سپس یک ماشین مجازی بر اساس استراتژی‌های استقرار عامل بهینه‌سازی ازدحام ذرات چندمنظوره پیشنهاد شد که می‌تواند ضمن بهبود استفاده از منابع، تعداد مهاجرت ماشین مجازی را کاهش دهد. الگوریتم تخصیص منابع ماشین مجازی ترکیبی از درخواست‌های کاربر و منابع ماشین مجازی مرکز داده است.

مرجع [۱۱]، یک استراتژی جایگذاری ماشین‌های مجازی را بر اساس ویژگی‌های بار کاری پیشنهاد می‌دهد که در آن ماشین‌های مجازی به دسته‌های مختلف دسته‌بندی شده‌اند. در این تحقیق ماشین‌های مجازی که به طور هم‌زمان بارکاری یکسانی دارند در دسته‌های مختلف قرار می‌گیرند و منابع به ماشین‌های مجازی قرار داده شده در هر گروه مطابق با درخواست منبع اختصاص داده می‌شود (متناسب با اوج استفاده از منابع). طی چندین مرحله تکرار این روش ماشین‌های مجازی در بهترین میزبان قرار می‌گیرند و همچنین ماشین‌های فیزیکی کمتری استفاده می‌شود. این رویکرد کیفیت سرویس را تضمین کرده و استفاده از منابع را بهبود می‌بخشد.

در [۱۲]، به بررسی تجمیع‌نمودن کل ماشین‌های مجازی موجود در مرکز داده بر روی حداقل میزبان پرداخته شده است که تا حد ممکن از روشن‌ماندن میزبان‌های دیگر خودداری کنند و بدین ترتیب میزبان‌های بلااستفاده به حالت خاموشی می‌روند. با این روش، میزبان کمتری روشن می‌ماند و به تبع آن انرژی مصرفی میزبان و مرکز داده کاهش پیدا می‌کند. در این روش تمام میزبان‌ها در حالت بهره‌وری کامل می‌باشند و تعادل میان میزبان‌ها برقرار نیست.

در [۱۳]، یک الگوریتم تخصیص منابع پویا در مرکز داده ابری ارائه شده است. این الگوریتم با پیش‌بینی حجم کاری آینده از سربار بارکاری جلوگیری و به بهبود مصرف انرژی و بهره‌وری مرکز داده کمک می‌نماید.

در [۱۴]، یک روش جایگذاری پویای ماشین مجازی در مرکز داده ابری ارائه شده است. با این روش تعداد میزبان‌های کمتری فعال است تا کاهش مصرف انرژی و افزایش بهره‌وری وجود داشته باشد. این روش جایگذاری بر پایه ریاضیات بنا شده و از استدلال قوی برخوردار است. در این روش از فاکتورهای مهم دیگری که در مصرف انرژی تأثیرگذار می‌باشند نیز استفاده شده تا مصرف انرژی را تا حد ممکن کاهش دهد.

در [۱۵]، یک روش جدید برای مهاجرت ماشین‌های مجازی به یک ماشین فیزیکی و به حداقل رساندن تعداد ماشین‌های مجازی فعال ارائه شده است. این کار با استفاده از یک الگوریتم برنامه‌نویسی پویا انجام می‌شود که مناسب‌ترین ماشین مجازی برای مهاجرت را از یک ماشین فیزیکی که بارکاری آن از مرز امن گذشته است انتخاب نماید. در این مقاله ابتدا فرمول ناهمواری معرفی می‌شود. با استفاده از ناهمواری، اندازه حافظه و دانه‌دانه بودن ماشین مجازی، هزینه مهاجرت هر ماشین مجازی در ماشین فیزیکی‌ای که بارکاری آن از مرز امن گذشته است تخمین زده می‌شود. سپس دو الگوریتم که با استفاده از ناهمواری بهترین ماشین مجازی و بهترین مقصد را برای مهاجرت انتخاب می‌کنند، ارائه می‌شود. در این مقاله شناسایی ماشین‌های فیزیکی که بارکاری آنها از حد امن گذشته است با مشاهده مداوم ماشین‌های فیزیکی انجام می‌شود.

در [۱۶]، روش‌های Trace-based برای پیش‌بینی بارکاری و یکپارچه‌سازی سرورها به کار گرفته شده است که در آن روش‌های مانیتورکردن آنلاین برای تشخیص این که چه زمانی به حد آستانه یک سرور رسیده شده یا چه زمانی باید سرورها یکپارچه گردد به کار گرفته می‌شود. اما ارتباط سرویس‌های درون ماشین‌های مجازی در آن در نظر گرفته نشده است.

در [۱۷]، مجموعه‌ای از micro benchmarkها به کار گرفته شده تا بتوان قبل از انجام مهاجرت، میزان سربار را تشخیص داد تا به صورت تصادفی مهاجرت بر روی یک سرور با میزان منابع نامناسب انجام داده نشود. این کار در زمان آغازین استقرار ماشین مجازی می‌تواند مفید باشد.

در [۱۸]، ثابت شده که دو ماشین مجازی که با هم در ارتباط هستند اگر در یک سرور قرار بگیرند تا ۹۰٪ زمان انتقال اطلاعات بین آنها کاهش می‌یابد. این کار نه تنها زمان پاسخ را کاهش می‌دهد، بلکه ترافیک شبکه را نیز کاهش می‌دهد. همچنین در این مقاله این ادعا شده که کاهش بار شبکه باعث افزایش بهره‌وری از منابع دیگر مانند پردازنده می‌شود. در این مقاله یک روش آگاهانه غیر متمرکز برای مهاجرت که ارتباطات ماشین‌های مجازی را هم در نظر گرفته و سعی کرده تا سربار آنها را با الگوریتم bartering کاهش دهد ارائه شده است.

در [۱۹]، یک روش جعبه سیاه برای تشخیص ارتباط گروهی از ماشین‌های مجازی به نام Net-Cohort بیان شده است. روش انجام کار به این صورت است که به جای مشاهده ترافیک تمام ماشین‌های مجازی درون شبکه، با ساخت یک ماتریس $N \times N$ ارتباط درون شبکه تفسیر می‌شود. با استفاده از الگوریتم‌های سلسله‌مراتبی خوشه‌بندی، ماشین‌های مجازی بر اساس ارتباطی که با هم دارند گروه‌بندی می‌شوند.

در [۲۰]، یک الگوریتم جهت مهاجرت ماشین‌های مجازی در رایانش ابری بیان شده است. به این صورت که در رایانش ابری، داده‌های یک برنامه می‌توانند در بلوک‌های مختلف به صورت توزیع‌شده ذخیره شده باشند و بلاک‌ها ممکن است به صورت فیزیکی از هم فاصله داشته باشند. همچنین در این مقاله سرعت شبکه بین ماشین‌های مجازی در یک ماتریس وجود دارد و مدت زمانی که طول می‌کشد تا یک ماشین مجازی به داده‌های خود دسترسی داشته باشد در ماتریس زمان دسترسی نگهداری می‌شود. جهت مهاجرت یک ماشین مجازی به یک میزبان دیگر ابتدا زمان دستیابی داده‌های ماشین مجازی در تمامی میزبان‌های موجود محاسبه می‌شود و نهایتاً ماشین مجازی به میزبانی مهاجرت می‌کند که کمترین زمان دسترسی به داده‌هایش را در آن میزبان داشته و همچنین میزبان منابع کافی را برای ماشین مجازی جدید داشته باشد.

در [۲۱]، ماشین‌های مجازی به صورت هوشمند با راهنمایی‌هایی که کاربر انجام می‌دهد جایگذاری می‌شوند. راهنمایی‌ها پیشنهاد می‌کنند که ماشین‌های مجازی به چه صورتی چیده شوند. این روش معمولاً مقدار زیادی از راهنمایی‌ها را به علت منابع فیزیکی موجود نادیده می‌گیرد و همچنین کاربر ممکن است راهنمایی‌هایی را فراهم کند که با زیرساخت رایانش ابری سازگار نباشد.

در [۲۲]، یک سیستم اختصاص منبع به صورت پویا ارائه شده که تلاش می‌کند با پرهیز از عبور بارکاری ماشین‌های فیزیکی از نقطه امن تعداد ماشین‌های فیزیکی فعال را نیز کاهش دهد. برای شناسایی ماشین‌های فیزیکی که بارکاری آنها از نقطه امن عبور نموده است، آنها

1. Heuristic

به صورت دوره‌ای بارکاری تمامی مرکز داده را مشاهده می‌کنند. همچنین یک الگوریتم پیش‌بینی بارکاری برای برنامه‌ها ارائه شده که مشخص می‌کند برنامه‌ها در آینده به چه میزان منابع احتیاج دارند و در جایگذاری ماشین‌های مجازی از این الگوریتم استفاده شده است.

در [۲۳]، مشکل زمان‌بندی درخواست‌ها در برنامه‌های وب چندلایه‌ای در سیستم‌های مجازی نامتقارن مورد مطالعه قرار گرفته است. در این مقاله هدف این است مصرف انرژی تا زمانی که کارایی کم نشود کاهش یابد. زمانی که منابع تثبیت شدند، کاهش کارایی را در منابعی که به شدت از آنها بهره گرفته می‌شود بررسی می‌نماید. یک روش اکتشافی که برای مسایل چندبعدی bin packing استفاده می‌شود برای تثبیت بارکاری در نظر گرفته شده است. روش ارائه‌شده با بارکاری در تعامل و از برنامه‌ها مستقل است.

در [۲۴]، یک روش برای جایگذاری ماشین‌های مجازی در محیط‌های مجازی‌شده غیر همگن برای بهینه‌سازی مصرف انرژی ارائه شده است. پارامترهای حداقل، حداکثر و میزان زمانی که یک پردازنده در ماشین مجازی Xen به صورت اشتراکی توسط ماشین‌های مجازی دیگر استفاده می‌شود، تغییر داده شده است. این روش تنها برای محیط‌های تجاری مناسب است، زیرا از یک SLA قاطع و سخت پشتیبانی نمی‌کند و به اولویت برنامه‌ها نیاز دارد تا بتواند پارامترهای اشتراکی را تعریف کند. محدودیت دیگری که در این روش وجود دارد این است که جایگذاری ماشین‌های مجازی در زمان اجرا ممکن نیست و جایگذاری به صورت ایستا است.

در [۲۵]، مهاجرت زنده ماشین‌های مجازی (VMs) یکی از ویژگی‌های اساسی مجازی‌سازی بیان شده که امکان مهاجرت ماشین‌های مجازی از یک مکان به مکان دیگر را بدون توقف ماشین‌های مجازی فراهم می‌آورد. این فرایند برای مراکز داده مزایای بسیاری از قبیل تعادل بار، نگهداری آنلاین، مدیریت انرژی و تحمل خطا را دارد. در این مقاله درک بهتری از مهاجرت زنده ماشین‌های مجازی و رویکردهای اصلی آن ارائه شده است. به طور خاص، به بررسی تکنیک‌های بهینه‌سازی اختصاص داده شده به توسعه مهاجرت زنده ماشین مجازی با توجه به مهاجرت حافظه تمرکز می‌کند. ذخیره‌سازی تقریباً در مراکز داده مشترک است. مهاجرت حافظه، اصلی‌ترین مسأله مهاجرت زنده ماشین مجازی به دلیل استفاده مکرر صفحات حافظه ماشین مجازی در هنگام مهاجرت است.

در [۲۶]، یک روش ترکیبی برای محاسبات ابری سبز جهت بهبود مصرف انرژی الکتریکی در مراکز داده پیشنهاد شده است که بر اساس زمان‌بندی درخواست و رویکردهای ترکیب سرورها می‌باشد. این روش قبل از زمان‌بندی، درخواست‌های (وظایف) مشتریان را با توجه به زمان سرویس و مصرف برق آنها مرتب می‌کند. در این روش سه الگوریتم مورد استفاده قرار گرفته است: ۱) الگوریتم زمان‌بندی که در هنگام تصمیم‌گیری جهت اختصاص درخواست به ماشین مجازی، مصرف برق را در نظر می‌گیرد، ۲) الگوریتم ترکیب، سرورهای کم‌بار را برای خواب، سرورهای پربار را برای مهاجرت ماشین‌های مجازی و سرورهایی که ماشین‌های مجازی مهاجرت‌شده را دریافت می‌کنند، تعیین می‌کند و ۳) الگوریتم مهاجرت که برای انتقال ماشین‌های مجازی به سرورهای جدید است. روش ارائه‌شده به کاهش مصرف انرژی، میانگین زمان اجرا و توان عملیاتی می‌پردازد.

مرجع [۲۷]، جهت محاسبات ابری سبز روی طرح تخصیص منابع مؤثر به درخواست‌های سرویس کاربران ابری متمرکز است که با به کارگیری دو لایه مدیریت منابع ابر و مدیریت سبز در جهت حفظ کیفیت سرویس می‌باشد. لایه مدیریت ابر امکان‌پذیرترین منابع قابل اجرا را که با درخواست سرویس مطابقت دارند لیست می‌کند که شامل پارامترهایی مانند بار کار فعلی، طول صف و فاصله بین درخواست‌کننده سرویس و سرور ابری می‌باشد. لایه مدیریت سبز با استفاده از الگوریتم فرااکتشافی مناسب‌ترین و بهینه‌ترین منبع را برای درخواست سرویس انتخاب می‌کند. با انتخاب بهینه منابع، متوسط زمان پاسخ سرویس با کمترین مصرف انرژی کاهش می‌یابد.

در [۲۸]، روشی جهت مهاجرت ماشین مجازی بر اساس الگوریتم کرم شب‌تاب با آگاهی از انرژی در محاسبات ابری پیشنهاد شده است. در این روش، بر اساس الگوریتم کرم شب‌تاب ماشین‌های مجازی از سرورهای با بارکاری بالا به میزبان فیزیکی فعالی که کمترین میزان بارکاری را دارد، مهاجرت داده می‌شوند، به طوری که کارایی و انرژی مؤثر مراکز داده را حفظ می‌نماید. روش مهاجرت ماشین مجازی برای محاسبات ابری مبتنی بر رفتار چشمک‌زن کرم شب‌تاب است. این روش تلاش می‌کند تا ماشین مجازی را از یک گره فعال به یک گره فعال دیگر که انرژی کمتری مصرف می‌کند، مهاجرت دهد.

در [۲۹]، چارچوب زمان‌بندی آگاه از انرژی برای مراکز داده ابری ارائه گردیده که از روش ترکیبی مبتنی بر الگوریتم‌های کلونی زنبور عسل و ازدحام ذرات استفاده شده است. برای کاهش گره‌های فعال و خاموش‌نمودن گره‌های بلااستفاده، مهاجرت و قرارگیری ماشین‌های مجازی بر روی گره‌های فیزیکی انجام می‌گیرد. مزیت روش، یافتن راه حل بهتر به دلیل ویژگی متعادل‌سازی جستجو و بهره‌وری است. روش ارائه‌شده زمان اجرا و مصرف انرژی را کاهش داده و موجب عدم نقض SLA می‌گردد.

محاسبات ابری مبتنی بر فناوری مجازی‌سازی است. هر مرکز داده ابری از سرورهای ناهمگن تشکیل شده و شامل ماشین‌های مجازی مختلف با مشخصات و نمونه‌هایی می‌باشد. مفهوم مجازی‌سازی در محیط ابری با تکنیک‌های انتقال ماشین مجازی بیان شده است [۳۰]. جدول ۱ به بررسی روش‌های پیشین می‌پردازد.

روش پیشنهادی در این مقاله بر اساس شناسایی میزبان‌هایی با مصرف انرژی بیش از حد و همچنین بسیار کم (مناسب برای خاموش‌شدن) اقدام به انتخاب و مهاجرت ماشین‌های مجازی می‌نماید. هدف، انتقال سرویس‌ها از یک میزبان به دیگری با استفاده از مهاجرت زنده می‌باشد. مهاجرت زنده این امکان را می‌دهد که بعد از مهاجرت، سیستم بدون شروع از ابتدا، از همان جا که دچار وقفه شد کارش را از سر بگیرد. سرویس‌هایی که انتخاب شدند به میزبان جدیدی که حاوی سرویس‌های همکار است، برای ادامه اجرای خود نگاشت می‌شوند و این نگاشت باعث می‌شود هزینه ارتباطی بین سرویس‌های همکار که همان سرویس‌هایی بودند که مدام نیاز داشتند برای تبادل پیام با هم ارتباط داشته باشند صفر شود و در نتیجه این فرایند منجر به کاهش مصرف انرژی مرکز داده می‌شود.

۳- روش پیشنهادی

ساختار مراکز داده در الگوریتم ارائه‌شده به این صورت است که اطلاعات مربوط به ارتباطات بین سرویس‌ها در مرکز داده به صورت محلی ذخیره می‌شود. مرکز داده اطلاعات مربوط به ارتباطاتی که سرویس‌هایش با دیگر سرویس‌ها دارند را در یک پایگاه داده و یا با به کارگیری یکی دیگر از روش‌های مرسوم ذخیره و بازیابی اطلاعات نگهداری می‌کند. در نتیجه این اقدامات یک آگاهی از وضعیت ارتباطات

جدول ۱: بررسی روش‌های پیشین.

منبع	روش	دسته مهاجرت	هدف	پارامترها	سال
[۲۶]	چارچوب ترکیبی	الگوریتم اکتشافی	بهبود مصرف انرژی	زمان‌بندی درخواست و ترکیب سرورها	۲۰۲۰
[۲۷]	لایه‌بندی	الگوریتم فرااکتشافی	کاهش زمان پاسخ و مصرف انرژی	مدیریت منابع ابر و مدیریت سبز	۲۰۲۰
[۹]	پیش‌بینی بارکاری ماشین مجازی	حد آستانه بارکاری	کاهش تعداد مهاجرت	تعادل بار	۲۰۱۹
[۲۹]	چارچوب زمان‌بندی ترکیبی	کلونی زنبور عسل و ازدحام ذرات	کاهش مصرف انرژی	مهاجرت و قرارگیری ماشین مجازی	۲۰۱۹
[۲۵]	بررسی تکنیک‌های مهاجرت ماشین‌های مجازی	مهاجرت صفحات حافظه	کاهش زمان مهاجرت	تعادل بار، مدیریت انرژی و تحمل خطا	۲۰۱۸
[۲۸]	الگوریتم کرم شب‌تاب	مبتنی بر رفتار چشمک‌زن کرم شب‌تاب	بهینه‌سازی مصرف انرژی	تعداد مهاجرت‌ها	۲۰۱۶
[۷]	مهاجرت ماشین‌های مجازی دارای کمترین ضریب	ضریب همبستگی	کاهش تعداد مهاجرت‌ها	درصد استفاده از پردازنده، حافظه و پهنای باند	۲۰۱۵
[۸]	تجمیع ماشین‌های مجازی	الگوریتم اکتشافی	کاهش مصرف انرژی	رک	۲۰۱۵
[۱۳]	تخصیص منابع پویا	الگوریتم skewness	کاهش مصرف انرژی	پیش‌بینی بار آینده و تعادل بار	۲۰۱۵
[۱۵]	مهاجرت زنده و ادغام پویای ماشین مجازی	الگوریتم برنامه‌نویسی پویا	کاهش تعداد ماشین‌های مجازی	فرمول ناهمواری	۲۰۱۵
[۱۴]	جایگذاری پویای ماشین مجازی	ماتریس احتمال نرمال	کاهش مصرف انرژی	بر پایه ریاضیات آماری	۲۰۱۴
[۱۰]	تئوری بهینه‌سازی ازدحام ذرات	الگوریتم اکتشافی	کاهش تعداد مهاجرت	منابع ماشین مجازی	۲۰۱۳
[۱۱]	دسته‌بندی ماشین‌های مجازی	الگوریتم اکتشافی	بهبود مصرف منابع	بارکاری	۲۰۱۲
[۱۹]	خوشه‌بندی ماشین‌های مجازی	تطبیقی و اکتشافی	کاهش ارتباطات	جعبه سیاه Net-Cohort	۲۰۱۲
[۲۲]	جایگذاری برنامه‌ها	تطبیقی و اکتشافی	کاهش مصرف انرژی	پیش‌بینی بارکاری تعادل بار	۲۰۱۲
[۳]	تلفیق ماشین‌های مجازی	تطبیقی و اکتشافی	کاهش مصرف انرژی	تاریخچه مصرف منابع	۲۰۱۲
[۱۸]	مهاجرت ماشین‌های مجازی	آگاهانه غیر متمرکز	کاهش زمان پاسخ و ترافیک شبکه	ارتباطات ماشین‌های مجازی	۲۰۱۰
[۱۲]	تجمیع ماشین‌های مجازی روی حداقل میزبان	الگوریتم اکتشافی	کاهش مصرف انرژی	بارکاری	۲۰۱۰
[۲۰]	مهاجرت ماشین‌های مجازی	الگوریتم اکتشافی	کاهش تعداد مهاجرت	زمان دسترسی به داده‌ها	۲۰۱۰
[۲۴]	جایگذاری ماشین مجازی در محیط غیر همگن	الگوریتم اکتشافی	کاهش مصرف انرژی	میزان زمان استفاده مشترک پردازنده	۲۰۰۹
[۱۷]	تخمین منابع مورد نیاز جهت جایگذاری	–	کاهش سربار	Micro benchmark و رگرسیون	۲۰۰۸
[۲۳]	تلفیق بارهای کاری	الگوریتم اکتشافی	کاهش مصرف انرژی	bin packing چندبعدی	۲۰۰۸
[۱۶]	مدیریت ظرفیت منابع مبتنی بر ردیابی	پیش‌بینی بارکاری و یکپارچه‌سازی سرورها	مدیریت منابع	ظرفیت منبع، بارکاری سرور	۲۰۰۷
روش پیشنهادی	تجمیع سرویس‌های همکار	اولویت‌دهی به سرویس‌ها	کاهش مصرف انرژی و تعداد مهاجرت	سرویس‌های همکار، بارکاری	۲۰۲۱

سرویس‌ها به دست می‌آید که می‌تواند در کاندیداکردن سرویس‌ها برای مهاجرت بسیار کارامد باشد. زمانی که انرژی مصرفی یک سرور در مرکز داده بیشتر از حد آستانه (پرباری یا کم‌باری) شود، با کمک این اطلاعات یک سرویس مناسب برای مهاجرت انتخاب می‌شود و در یک مقصد مناسب قرار می‌گیرد که باعث کاهش تعداد مهاجرت‌های غیر ضروری سرویس‌ها نیز می‌شود.

در حالت کلی مسأله تخصیص سرویس‌ها به سرورها را می‌توان به سه زیرمسأله تقسیم نمود. زیرمسأله اول این است که کدام سرویس مهاجرت کند و زیرمسأله دوم به این که چه زمانی باید سرویس مهاجرت کند می‌پردازد. زیرمسأله سوم به پرسش سرویس به کجا مهاجرت کند باید پاسخ دهد. تمرکز اصلی این تحقیق بر روی حل زیرمسأله اول و سوم می‌باشد. برای حل مسأله اول سرویس با اولویت در نظر گرفته شده باید مهاجرت کند (بخش ۳-۱، رابطه (۸)) و برای حل زیرمسأله سوم سرویس به میزبان حاوی سرویس‌های همکار مهاجرت می‌کند. بدین گونه سرویس‌های همکار کنار هم قرار می‌گیرند و در نتیجه انرژی کمتری مصرف می‌کنند (بخش ۳-۲، رابطه (۹)). روش پیشنهادی دارای سه فاز می‌باشد:

- فاز اول: اولویت‌دهی و انتخاب سرویس مناسب برای مهاجرت
- فاز دوم: پیداکردن سرویس‌های همکار
- فاز سوم: انتخاب میزبان مناسب برای سرویس

ساختار روش پیشنهادی در شکل ۲ نشان داده شده است.

سرویس‌های درخواستی به صورت گراف‌های جهت‌داری دیده می‌شوند که می‌توان با استفاده از پارامترهای استخراج‌شده از گراف‌های جهت‌دار اولویت هر سرویس را مشخص نمود تا زمانی که قرار بر مهاجرت یک

جدول ۲: پارامترهای استخراج‌شده از گراف جهت‌دار.

توضیحات	پارامتر
سرویس	V_i
مقدار نرمال V_i	$Norm(V_i')$
حداقل و حداکثر مقدار پارامتر V	$\min(V)$ و $\max(V)$
سطح سرویس V_i	$Level(V_i)$
زمان اجرای سرویس V_i	$Execute\ Time(V_i)$
اولویت سرویس i در سرور j	$Priority(V_{ij})$
هزینه تبادل داده بین V_i و V_j	$Communication\ cost(V_i,V_j)$
رتبه همکاری سرویس V_i با V_j	$RankCooparator(V_i,V_j)$
زمان تخمینی شروع	EST *
تعداد دفعات ملاقات V_i با V_j	$sumMeet(V_i,V_j)$
مجموعه والدین V_i	$Predecessor(V_i)$
تخمین زمان پایان	EFT **
تعداد فرزندان سرویس V_i	$Count\ of\ Child(V_i)$

* Estimated Start Time

** Estimated Finish Time

سرویس بود، سرویس با اولویت پایین مهاجرت نماید. یک گراف جهت‌دار به صورت $(DAG)\ G=(V,E), v_i \in V, E_{i,j} \in E$ و شکل ۳ تعریف می‌شود. V به عنوان یک رأس در گراف، نشان‌دهنده یک سرویس است. E نیز به عنوان یک یال در گراف نشان‌دهنده وابستگی بین دو سرویس می‌باشد. هر رأس یک وزن دارد که زمان اجرای هر سرویس را نشان می‌دهد و وزن موجود روی هر یال نیز هزینه انتقال بین دو رأس i و j است.

روابطی که از این گراف استخراج شده است در جدول ۲ بیان گردیده و در روش پیشنهادی مورد استفاده قرار گرفته است. برای فهم بهتر مطالب مثالی در پیوست آورده شده است.

زمان تخمینی شروع (EST) از (۱) محاسبه می‌شود

$$EST(V_i)\begin{cases} 0 & \text{if } i=1 \\ EST(V_h)+ExecutionTime(V_h) \\ \quad +MAXCommunicationCost(V_h,V_i) & \\ \quad , \text{ otherwise} & \\ \text{where } V_h \text{ is the direct predecessors of } V_i \end{cases} \quad (1)$$

محاسبه EST به صورت بالا به پایین است (از ریشه به برگ) و EST گره ریشه صفر است.

با اجرای هر سرویس می‌توان EST آن را محاسبه کرد و خروجی این پیش‌پردازش سریع‌ترین زمانی است که همه سرویس‌های قبلی تکمیل شده باشند. طبق رابطه فوق محاسبه EST به صورت بازگشتی خواهد بود.

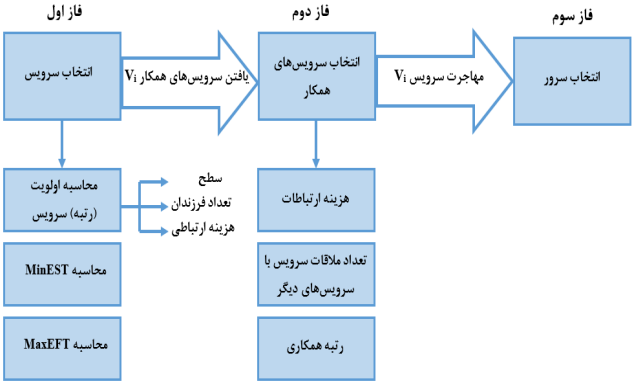
برای محاسبه EFT از (۲) استفاده می‌شود

$$EFT(V_i)=EST(V_i)+ExecuteTime(V_i) \quad (2)$$

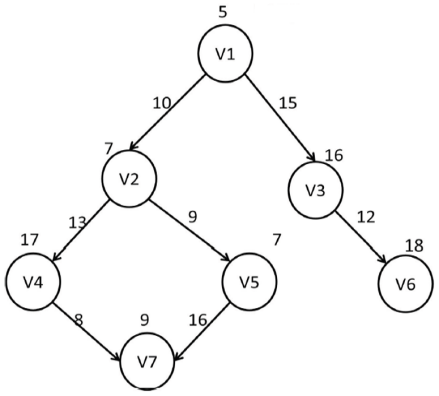
زمان تخمینی شروع با زمان اجرای وظیفه جمع می‌شود تا بتوان زمان پایان آن وظیفه را تخمین زد.

سرویسی که هیچ فرزندی نداشته باشد برگ نامیده می‌شود. سطح[1] سرویس از (۳) محاسبه می‌شود

1. Level



شکل ۲: ساختار روش پیشنهادی.



شکل ۳: ساختار گراف جهت‌دار.

$$\begin{aligned} &if\ \ predecessors(V_i)=0\ then\ level=2^n\ \ ,\ n=0 \\ &if\ \ predecessors(V_i)\neq 0\ then\ level=2^n \\ &\quad ,\ n=n(V_h)+1 \\ &where\ V_h\ is\ the\ predecessors\ of\ V_i \end{aligned} \quad (3)$$

به طوری که $Predecessors(V_i)$ یک مجموعه از وظایف بلافصل قبلی V_i یا به عبارتی بهتر مجموعه پدران است.

۳-۱ فاز اول: اولویت‌دهی و انتخاب سرویس مناسب برای مهاجرت

سرویس‌ها به صورت گراف‌های جهت‌دار دیده می‌شوند که می‌توان با استفاده از پارامترهای استخراج‌شده از گراف‌های جهت‌دار که با استفاده از آن اولویت هر سرویس مشخص می‌شود تا زمانی که قرار بر مهاجرت یک سرویس بود، سرویسی که عدد متناظر با اولویت آن پایین است مهاجرت یابد. برای نرمال‌سازی مقادیر پارامترها از (۴) استفاده می‌شود

$$Norm(V_i')=\frac{V_i-\min(V)}{\max(V)-\min(V)} \quad (4)$$

برای تمام سرویس‌ها، متوسط تعداد فرزندان از (۵)، متوسط سطح از (۶) و متوسط هزینه ارتباطی هر سرویس از (۷) در هر گراف جهت‌دار محاسبه می‌شود، سپس هر یک از این پارامترها را نرمال نموده و متوسط هر کدام از پارامترها در تمام گراف جهت‌دار به دست می‌آید. در نهایت اولویت سرویس با استفاده از (۸) مشخص می‌شود

$$\begin{aligned} &AverageCountOfChild(V_i) \\ &=\sum_{i=1}^{n}\frac{Norm(CountOfChild(V_i))}{N} \\ &where\ N\ is\ Count\ of\ DAG \end{aligned} \quad (5)$$

$$AverageLevel(V_i) = \sum_{i=1}^{n} \frac{Norm(Level(V_i))}{N} \quad (۶)$$
$$where\ N\ is\ Count\ of\ DAG$$

$$AverageCommunicationCost(V_i, V_j) = \sum_{i=1}^{n} Norm(\sum_{j=1}^{m} \frac{CommunicationCost(V_i, V_j)}{N}) \quad (۷)$$
$$where\ n\ is\ a\ Count\ Number\ of\ DAGs\ and\ m\ is\ a\ Count\ Number\ of\ services$$

$$priority(V_{ij}) = AverageCountOfChild(V_{ij}) - AverageLevel(V_{ij}) + AverageCommunicationCost(V_{ij}) \quad (۸)$$
$$(i = 1,2,\ldots,n \text{ and } j = 1,2,\ldots,m)$$

در این رابطه $\min(V)$ و $\max(V)$ به ترتیب حداقل و حداکثر مقدار پارامتر V می‌باشند و مقدار V_i از V به مقدار V_i' نرمال می‌شود. بعد از به دست آوردن اولویت تمام سرویس‌ها با استفاده از (۸)، برای انتخاب سرویسی که هنگام پرباری یا کم‌باری سرور باید مهاجرت کند، سرویسی انتخاب می‌شود که عدد متناظر با اولویت مهاجرت آن دارای مقدار کمتری باشد.

با محاسبه رتبه، MinEST و MaxEFT هر سرویس V_i در سرور X به (CandidatedList) لیست سرویس‌هایی که قرار است مهاجرت پیدا کنند اضافه می‌شود. سپس لیست به صورت نزولی بر اساس مقدار رتبه[1]، مرتب و اولین سرویس با اولویت بالاتر به تابع $ServicePartner(V)$ ارسال می‌شود تا سرویس‌های همکار (V') مشخص گردد. تابع $SelectHost(V,V')$ میزبان مناسب برای سرویس V_i را یافته و سرویس را به آن میزبان اختصاص می‌دهد و در نهایت سرویس V_i از CandidatedList حذف می‌شود.

Input: List of DAG (Full), List of Execute Time on every Service, ServiceList, HostList
Output: A mapping of Servers to Host
Main (V, V')
```
01: {
02:    foreach serviceType (Vi) in HOST X
03:       {
04:          Computing Rank(Vi)
05:          Computing MinEST(Vi)
06:          Computing MaxEFT(Vi)
07:          Insert Vi to CandidatedList
08:       }
09:    sort CandidatedList
10:       Select first item (V) from CandidatedList
11:       [V'] = ServicePartner(V)
12:       SelectHOST(V, V') to Migration Service V
13:    remove V from CondidatedList
14: }
```

در تابع computing Rank برای محاسبه اولویت هر سرویس باید مجموع سطح، مجموع تعداد فرزندان و مجموع هزینه ارتباطی در تمام گراف‌های جهت‌دار به دست آورده شود، سپس برای به دست آوردن رتبه هر سرویس مجموع متوسط هر یک از این سه پارامتر محاسبه می‌شود. قابل ذکر است که هر کدام از این پارامترها ابتدا نرمال و سپس در مجموع محاسبه می‌شود.

Input: service V_i
Output: Rank(V_i)
Computing Rank (service V_i)
```
01: {
02:    foreach DAG
03:    {
04:       SumLevel(Vi) += NormLevel(Vi) in DAG j
05:       SumNumber of children (Vi) +=
             NormCountOfChildren(Vi) in DAG j
06:       SumCommunication cost (Vi) +=
          NormComputtingCommunicationCost(Vi) in DAG j
07:    }
08:    N = The Number of DAG
09:    Rank(Vi) = SumNumber of children (Vi)/N −
          SumLevel(Vi)/N + SumCommunication cost (Vi)/N
10:    return (Rank(Vi))
11: }
```

برای یافتن MinEST، برای هر گراف جهت‌دار مقدار EST سرویس V_i را محاسبه نموده و در نهایت از بین مقادیر به دست آمده کوچک‌ترین به عنوان $MinEST(V_i)$ می‌باشد.

Input: service V_i
Output: MinEST(V_i)
Computing MinEST (V_i)
```
01: {
02:    foreach DAG
03:    {
04:       if (i==1) then EST(Vi) = 0
05:       else
06:       {
07:          EST(Vi) = EST(Vh) + ExecuteTime(Vh)
                + MAXCommunicationCost(Vh, Vi)
08:       }
09:    }
10: return (MinEST(Vi))
11: }
```

برای محاسبه MaxEFT، برای هر گراف جهت‌دار مقدار EFT سرویس V_i محاسبه شده و از بین مقادیر به دست آمده بزرگ‌ترین به عنوان $MaxEFT(V_i)$ نشان داده می‌شود.

Input: service V_i
Output: MaxEFT(V_i)
Computing MaxEFT(V_i)
```
01: {
02:    foreach DAG
03:    {
```

1. Rank

04: $EFT(V_i) = EST(V_i) + ExcuteTime(V_i)$
05: }
06: **return** (MaxEFT(V_i))
07: }

۳-۲ فاز دوم: پیداکردن سرویس‌های همکار هر سرویس

در این فاز سرویس‌های همکار هر سرویس با استفاده از گراف‌های جهت‌دار یافت می‌شود، زیرا قصد بر آن است که سرویس‌ها به سروری که حاوی سرویس‌های همکار هستند انتقال داده شود، زیرا هنگامی که سرویس‌های همکار کنار هم قرار می‌گیرند منابع کمتری (به عنوان مثال هارد، حافظه، پهنای باند و پاور) مصرف می‌شود و در نتیجه زمان پاسخ، انرژی مصرفی و تعداد مهاجرت‌ها کاهش می‌یابد.

برای به دست آوردن سرویس‌های همکار (V_j) یک سرویس (V_i)، باید مجموع تعداد دفعاتی که سرویس با هر سرویس ملاقات داشته (به طور متوسط) و همین طور مجموع هزینه‌های ارتباطی سرویس با سرویس‌هایی که در ارتباط است محاسبه شود (در پایگاه داده و به طور متمرکز). سرویس‌هایی که این مقدارها برای آنها بیشتر است به عنوان سرویس همکار شناخته می‌شود. مقادیر مجموع هزینه ارتباطات سرویس و مجموع تعداد دفعات ملاقات سرویس نرمال شده و در نهایت با استفاده از (۹) رتبه همکاری هر سرویس با سرویس‌های دیگر محاسبه می‌شود

$$RankCooparator(V_i,V_j) = Norm(sumCommunicationCost(V_i,V_j)) + Norm(sumMeet(V_i,V_j)) \quad (۹)$$

Input: service V_i
Output: service V // ServicePartner (service v)
ServicePartner (service V)
01: {
02: **foreach** service j
03: {
04: communication cost (service V, service j)=0
05: Meet (service V, service j)=0
06: RankCooparator (service V, service j)=0
07: }
08: **foreach** DAG i
09: {
10: **if** service V and service j in workflow i have communication
11: {
12: $communication\ cost(service\ V, service\ j) += communication\ cost(service\ V, service\ j\ inDAGi)$
13: $Meet(service\ V, service\ j) += 1$
14: }
15: $RankCooparator(service\ V, service\ j) = Norm(communicationcost(service\ V, service\ j)) + Norm(Meet(service\ V, service\ j))$
16: insert service j to list
17: }

۳-۳ فاز سوم: انتخاب سرور مناسب برای نگاشت سرویس

در این فاز باید سروری مناسب برای نگاشت سرویس منتخب یافت شود. در روش پیشنهادی ابتدا از لیست سرورها، سرورهایی که در زمان EST و EFT آن سرویس، در دسترس نیستند (بر اساس زمان‌بندی قبلی) از لیست سرورهای کاندیدا حذف می‌شوند. سپس از لیست سرورهای کاندیدای باقیمانده سرورهایی که بیش از حد آستانه انرژی مصرف می‌کنند نیز حذف می‌شوند. حال باید از لیست سرورهای کاندیدا سروری انتخاب شود که شامل سرویس‌های همکار باشد. چنانچه سرورهایی یافت شدند که به یک میزان حاوی سرویس همکار بودند، سروری انتخاب می‌شود که در آن مجموع اولویت سرویس‌هایش کم باشد. با تمام این شرایط چنانچه هنوز سرویس به سروری نگاشت نشده باشد به اولین سرور کاندیدا نگاشت می‌شود.

Input: Service V, Service V'
Output: A mapping of Service V to Host y
SelectHost (Service V, Service V')
01: {
02: allocatedHost=null; maxCountV'=0; sumRank=0;
03: **foreach** host
04: {
05: **if** each host available Time between MinEST(V) and MaxEFT(V), insert to Candidate Host List
06: }
07: **foreach** Host y in Candidate Host List
08: {
09: **if** ($T_{Host\ y}$ > threshold)
10: {
11: **remove** Host y from Candidate Host List;
12: }
13: **else**
14: {
15: **if** (Host y have V')
16: {
17: **if** (maxcountV' < count V' in Host y)
18: {
19: allocatedHost=Host y
20: maxcountV'=count V' in Host y
21: }
22: **else if** (maxcountV'==count V' in Host y)
23: {
24: **if** (sumRank allocatedHost > SumRankHost y)
25: {
26: allocatedHost=Host y
27: }
18: **sort** the list with RankCooparator
19: **return** (first item from list)
20: }

28: }
29: }
30: **if** (allocatedHost !=null)
31: {
32: Migration V to allocatedHost;
33: SumRank of allocatedHost += Rank(V);
34: }
35: **else**
36: {
37: **sort** Candidate Host List with SumRank of Host (order by Descending)
38: Migration V to first Host y in Candidate Host List
39: SumRank of Host y + = Rank(V)
40: }
41: } //end if in line 13
42: } //end if in line 7
43: } //end Function

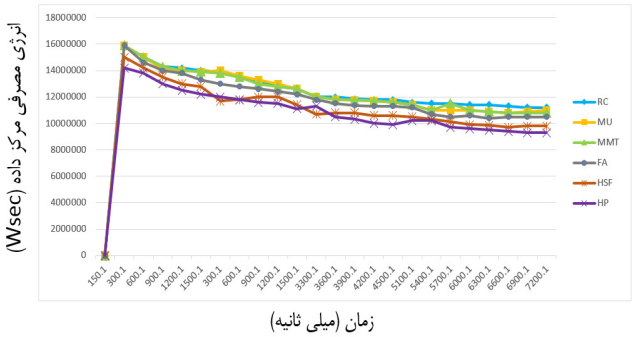
٤- ارزیابی روش پیشنهادی

شبیه‌سازی روش پیشنهادی توسط کلودسیم[1] انجام شده است. کلودسیم یک چارچوب قابل توسعه و متن باز را با مدلی یکپارچه به منظور آزمایش‌ها و شبیه‌سازی زیرساخت‌های محیط ابری و سرویس‌ها ارائه نموده است. جهت ارزیابی از داده‌های ارائه‌شده در پلنتلب[2] [۳۱] استفاده گردیده که یک مرکز داده حاوی ۸۰۰ میزبان فیزیکی و ۱۰۵۶ ماشین مجازی می‌باشد. در ابتدای شبیه‌سازی لیستی از میزبان‌ها و لیستی از سرویس‌ها و همچنین همکاری سرویس‌ها و گردش کار آنها به صورت تصادفی ساخته می‌شود. تعداد کل گردش کارها (ترتیب درخواست سرویس‌ها) به اندازه ۱۰ است. سپس سیاستی که باید سرویس را انتخاب کند و سیاستی که سرویس منتخب را به یک میزبان مناسب نگاشت نماید، ساخته می‌شود به گونه‌ای که SLA نیز در نظر گرفته شود. در طول زمان شبیه‌سازی الگوریتم، تشخیص سرویس‌های همکار و مهاجرت آنها در هر ۳۰۰ میلی‌ثانیه و یا زمانی که درخواست جدیدی از راه برسد اجرا می‌شود.

روش پیشنهادی (HP)[3] با ۵ الگوریتم زیر که از بهترین الگوریتم‌های موجود و مناسب با موضوع پژوهش می‌باشند، مقایسه شده است:

- حداقل زمان مهاجرت[4] [۵].
- سیاست انتخاب تصادفی[5] [۳].
- حداقل مصرف (استفاده)[6] [۶].
- الگوریتم کرم شب‌تاب[7] [۲۸].
- چارچوب زمان‌بندی ترکیبی[8] [۲۹].

1. CloudSim
2. PlanetLab
3. High Priority
4. Minimum Migration Time
5. Random Choice Policy
6. Minimum Utilization
7. Firefly Algorithm
8. Hybrid Scheduling Framework



شکل ۴: میزان انرژی مصرفی مرکز داده روش پیشنهادی با سایر سیاست‌ها.

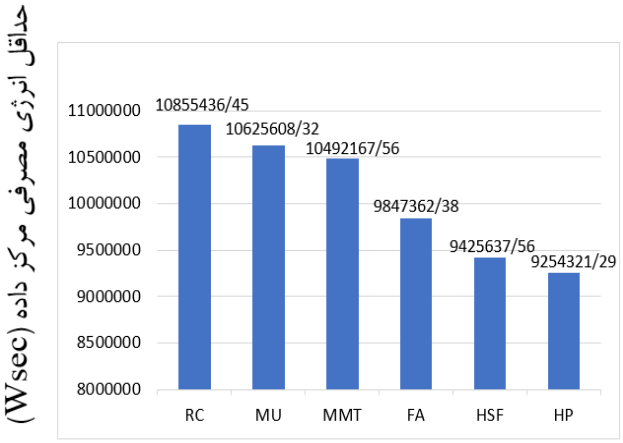
در سیاست MMT برای انتخاب یک سرویس، سرویسی انتخاب می‌شود که زمان تکمیل مهاجرت و اختصاص به میزبان در آن حداقل باشد. زمان مهاجرت با مقدار حافظه مورد استفاده ماشین مجازی تقسیم بر پهنای باند آزاد در دسترس میزبان j تخمین زده می‌شود. V_j مجموعه‌ای از ماشین‌های مجازی است که در حال حاضر به میزبان j اختصاص داده شده است. سیاست MMT یک ماشین مجازی را پیدا می‌کند که شرایط (۱۰) را برآورده نماید

$$\upsilon \in V_j \left| \forall_\alpha \in V_j \right. , \frac{RAM_u(\upsilon)}{NET_j} \le \frac{RAM_u(\alpha)}{NET_j} \qquad (۱۰)$$

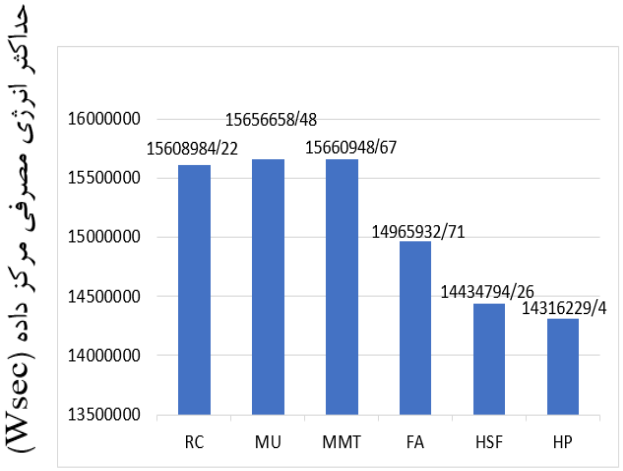
که $RAM_u(\alpha)$ مقدار حافظه است که در حال حاضر توسط $VM\alpha$ استفاده می‌شود و NET_j پهنای باند شبکه موجود برای میزبان j است. در سیاست RC برای انتخاب یک سرویس، یک سرویس به صورت تصادفی از روی متغیر تصادفی گسسته توزیع‌شده به صورت یکنواخت، $X_d = U(\cdot, \left|V_j\right|)$ را که مقادیر آن شامل مجموعه‌ای از ماشین‌های مجازی اختصاص داده شده به میزبان j می‌باشد انتخاب می‌کند. در سیاست MU برای انتخاب یک سرویس برای مهاجرت، سرویسی انتخاب می‌شود که نیاز به حداقل استفاده از پردازنده را نسبت به دیگر سرویس‌ها در میزبان دارد. در سیاست FA انتخاب ماشین مجازی و مهاجرت بر اساس الگوریتم کرم شب‌تاب می‌باشد. در سیاست HSF مهاجرت و قرارگیری ماشین مجازی بر اساس ترکیب الگوریتم‌های کلونی زنبور عسل و ازدحام ذرات است. در شکل ۴ نشان داده می‌شود که میزان انرژی مصرفی در روش پیشنهادی نسبت به روش‌های مورد مقایسه، در مرکز داده، انرژی کمتری مصرف کرده است (w: وات و sec: ثانیه).

در روش پیشنهادی، مهاجرت بر اساس اولویت و اختصاص به میزبان دارای سرویس‌های همکار می‌باشد که این عمل باعث کاهش تعداد مهاجرت‌ها می‌گردد و در نتیجه میزان مصرف انرژی کمتر از سیاست‌های RC (انتخاب سرویس تصادفی)، MU (حداقل استفاده پردازنده)، MMT (کمترین زمان مهاجرت)، FA (الگوریتم کرم شب‌تاب) و HSF (چارچوب زمان‌بندی ترکیبی) است. همچنین می‌توان بیان نمود روش پیشنهادی دارای حداقل و حداکثر انرژی مصرفی کمتری نسبت به سایر سیاست‌ها می‌باشد. در هر ۶ سیاست در بازه‌های زمانی که شبیه‌سازی صورت گرفته است حداقل انرژی مصرفی مرکز داده استخراج شده است. در شکل ۵، روش پیشنهادی کمترین حداقل انرژی مصرفی را دارد.

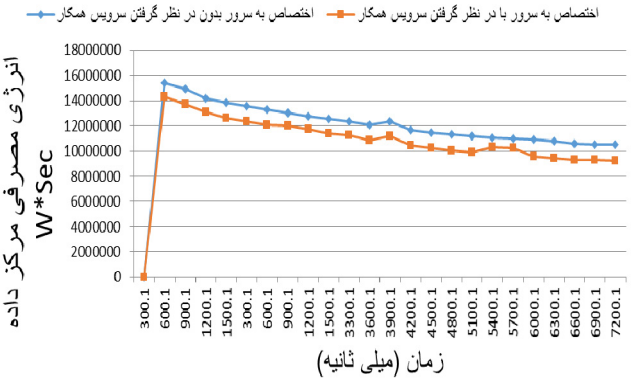
در هر ۶ سیاست در بازه‌های زمانی که شبیه‌سازی صورت گرفته است حداکثر انرژی مصرفی مرکز داده استخراج شده است. در شکل ۶، روش پیشنهادی کمترین حداکثر انرژی مصرفی را دارد که نشان‌دهنده این است که روش مهاجرت ارائه‌شده مبتنی بر اولویت و با توجه به کاهش تعداد مهاجرت‌ها دارای کاهش مصرف انرژی نسبت به ۵ سیاست دیگر می‌باشد.



شکل ۵: حداقل انرژی مصرفی مرکز داده در هر سیاست.



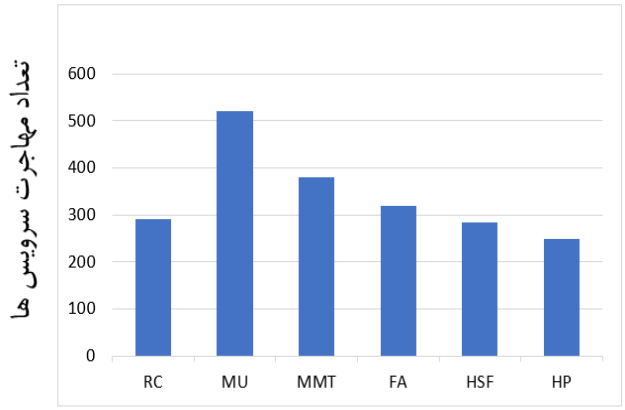
شکل ۶: حداکثر انرژی مصرفی مرکز داده در هر سیاست.



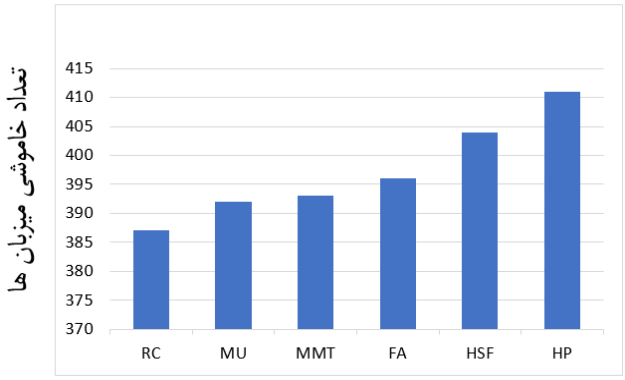
شکل۷ : نگاشت به میزبان با در نظر گرفتن سرویس‌های همکار و بدون در نظر گرفتن سرویس‌های همکار.

در شکل ۷، میزان انرژی مصرفی مرکز داده در حالت اختصاص سرویس منتخب به میزبان حاوی سرویس‌های همکار حدود ۸/۷۳٪ کمتر از حالتی است که سرویس منتخب برای مهاجرت را به میزبانی اختصاص دهیم که وجود سرویس‌های همکار در آن در نظر گرفته نشده است.

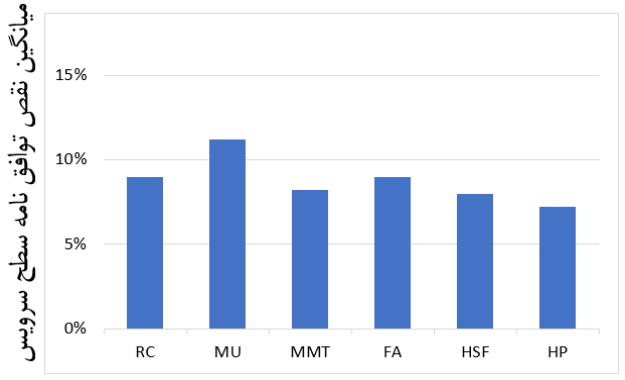
در شکل ۸ تعداد سرویس‌هایی که در طول شبیه‌سازی مهاجرت کردند در روش پیشنهادی (HP) کمتر است و این نشان می‌دهد که در سیاست روش پیشنهادی سرویس‌های کمتری نیاز به مهاجرت پیدا می‌کنند و از آنجایی که هرچه تعداد مهاجرت‌ها کمتر باشد مرکز داده انرژی کمتری مصرف می‌کند، بنابراین با توجه به نتیجه به دست آمده روش پیشنهادی منجر به کاهش مصرف انرژی مرکز داده می‌شود.



شکل ۸: تعداد مهاجرت سرویس‌ها در هر سیاست.



شکل ۹: تعداد خاموشی میزبان‌ها در هر سیاست.



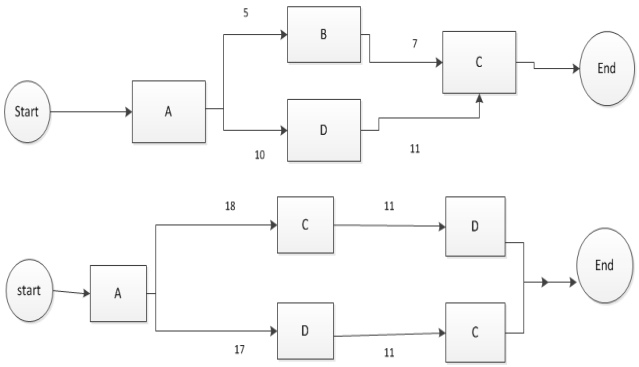
شکل ۱۰: میانگین نقض توافق‌نامه سطح سرویس (SLA) در هر سیاست.

در شکل ۹، در طول اجرای شبیه‌سازی، با قرار گرفتن سرویس‌های همکار در کنار یکدیگر تعداد میزبان‌های بیشتری خاموش می‌شوند و هرچه میزبان‌های بیشتری خاموش باشند انرژی مصرفی مرکز داده کاهش می‌یابد و بنابراین با توجه به نتیجه به دست آمده روش پیشنهادی منجر به کاهش مصرف انرژی مرکز داده می‌شود.

در شکل ۱۰ با قرارگرفتن سرویس‌های همکار در کنار یکدیگر، میانگین نقض توافق‌نامه سطح سرویس کاهش می‌یابد.

در جدول ۳ ارزیابی کیفی از نتایج سیاست‌های ارائه‌شده در این پژوهش جهت کاهش مصرف انرژی مراکز داده ابری مبتنی بر مهاجرت نشان داده شده است.

در روش پیشنهادی انتظار می‌رود سربار زیادی به مرکز داده ابر تحمیل شود اما به دلیل کاهش تعداد مهاجرت‌ها این سربار نسبت به سایر روش‌ها کمتر است. سربار ناشی از مهاجرت علاوه بر بهره‌وری و زمان مهاجرت به تعداد مهاجرت‌ها نیز بستگی دارد.



شکل ۱۱: مثالی از ساختار دو گراف جهت‌دار (DAG۱,DAG۲).

۵- نتیجه‌گیری

گسترش محاسبات ابری و افزایش منابع مرکز داده، موجب بالارفتن مصرف انرژی و به دنبال آن صرف هزینه‌های بیشتر برای فراهم‌کنندگان ابر شده است. تقاضاهای رو به رشد زیرساخت‌های ابری و استفاده از تجهیزات و سخت‌افزارهای کامپیوتری، سیستم‌های خنک‌کننده، روشنایی، تهویه و ذخیره‌سازی (دیسک‌ها) باعث افزایش مصرف انرژی و بالارفتن دما می‌شود که باعث افزایش تولید دی اکسید کربن و تولید گازهای گلخانه‌ای می‌گردد و این موجب بروز آلودگی‌های زیست‌محیطی است. این کربن توسط تجهیزات و سخت‌افزارهای کامپیوتری و تجهیزات جانبی تولید می‌شود. همچنین با بالارفتن مصرف انرژی، هزینه‌های عملیاتی شامل کاهش قابلیت اعتماد و طول عمر منابع سخت‌افزاری، افزایش بهای خدمات و بار زیاد سیستم خنک‌کننده افزایش می‌یابد. در نتیجه برای کاهش انتشار دی اکسید کربن در محیط اطراف و برای داشتن محاسبات سبز نیاز به تکنیک‌هایی برای کاهش مصرف انرژی در مراکز داده است.

در مراکز داده بزرگ تبادل پیام میان سرویس‌هایی که با هم ارتباطات زیادی دارند ولی در سرورهای مختلفی نگهداری می‌شوند سبب افزایش هزینه ارتباطی و تأخیر در پاسخ می‌شود. پژوهش‌های زیادی نشان داده‌اند که ارتباطات سرویس‌ها در بین سرورهای مختلف نه تنها باعث هدررفتن منابع ارتباطی می‌شوند، بلکه باعث مصرف بیهوده منابع دیگر مانند پردازنده نیز می‌شوند.

در این پژوهش تأکید بیشتر بر این است که افزایش محلی‌سازی سرویس‌ها با کنترل مهاجرت سرویس‌ها انجام شود. با مهاجرت سرویس‌ها بر اساس ارتباطات آن‌ها موجب در کنار هم قرار گرفتن سرویس‌های همکار و به دنبال آن کاهش تعداد مهاجرت‌ها و در نتیجه کاهش مصرف انرژی مراکز داده می‌شود. برای مراکز داده، کاهش هزینه‌های ارتباطی میان سرویس‌ها اهمیت و ضرورت زیادی دارد که این موضوع ارتباط مستقیمی با محل استقرار سرویس‌های مرتبط در مرکز داده دارد و استقرار صحیح سرویس‌ها در مرکز داده می‌تواند سبب کاهش مصرف انرژی مرکز داده شود.

در راستای بهبود کیفیت سرویس برای پیشنهادهای آینده می‌توان با استفاده از تاریخچه ارتباطات سرویس‌ها و با به کارگیری یکی از مدل‌های پیش‌بینی آینده بر اساس گذشته، سرویس‌هایی را که ممکن است در آینده با هم همکار شوند شناسایی نمود و به یکدیگر مهاجرت داد. همچنین زمان پایان همکاری سرویس‌ها را می‌توان به عنوان یکی از پارامترهای انتخاب سرویس برای مهاجرت در نظر گرفت. مهاجرت سرویس‌ها به نزدیکی مکان فیزیکی درخواست‌کننده سرویس از راهکارهایی است که موجب کاهش زمان پاسخ و مصرف انرژی و در نتیجه بهبود کیفیت

جدول ۳: ارزیابی کیفی از نتایج سیاست‌های کاهش مصرف انرژی مراکز داده ابری مبتنی بر مهاجرت.

پارامتر / روش	حداقل انرژی مصرفی	حداکثر انرژی مصرفی	تعداد مهاجرت سرویس	تعداد خاموشی میزبان
RC	بالا	متوسط	متوسط	پایین
MU	متوسط	متوسط	بالا	متوسط
MMT	متوسط	بالا	متوسط	متوسط
FA	متوسط	متوسط	بالا	متوسط
HSF	پایین	متوسط	پایین	متوسط
HP	پایین	پایین	پایین	بالا

سرویس می‌گردد.

پیوست

در شکل ۱۱، مثالی جهت تشریح روش پیشنهادی شامل چهار نوع سرویس A، B، C و D و دو گراف جهت‌دار که گره‌های آنها سرویس‌ها می‌باشند، بیان می‌گردد. زمان اجرای هر سرویس در جدول پ- ۱ و هزینه ارتباطی بین سرویس‌ها بر روی یال‌های گراف‌های جهت‌دار نشان داده شده است.

هر رأس یک وزن دارد که زمان اجرای هر سرویس را نشان می‌دهد و وزن موجود روی هر یال نیز هزینه انتقال بین دو رأس i و j است.

پارامترهای دو گراف جهت‌دار برای هر سرویس محاسبه و سپس نرمال می‌شود. مقادیر این پارامترها به همراه نرمال‌سازی در جداول پ- ۲، پ- ۳، پ- ۴ و پ- ۵ نشان داده شده است.

مقدار متوسط هر پارامتر از دو گراف جهت‌دار به دست می‌آید که در جدول پ- ۶ نشان داده شده است.

در نهایت اولویت هر سرویس جهت مهاجرت با توجه به رابطه موجود در متن، در جدول پ- ۷ مشخص می‌گردد.

بنابراین سرویس C کمترین رتبه را دارد و اگر در میزبانی که قرار است سرویس‌های آن مهاجرت پیدا کنند هر چهار سرویس موجود باشند باید به ترتیب ابتدا سرویس C، سرویس D، سرویس B و در نهایت سرویس A مهاجرت پیدا کند.

مینیمم EST و ماکسیمم EFT هر سرویس با توجه به رابطه‌هایی که برای این دو پارامتر ذکر شد، برای مثال در جدول پ- ۸ به دست می‌آید.

هزینه ارتباطی هر سرویس با سرویس‌های دیگر در گراف جهت‌دار شکل ۱۱ به صورت ماتریس‌های a و b خواهد بود

$$a=\begin{bmatrix} DAG۱ & A & B & C & D \\ A & ۰ & ۵ & ۰ & ۱۰ \\ B & ۰ & ۰ & ۷ & ۰ \\ C & ۰ & ۰ & ۰ & ۰ \\ D & ۰ & ۰ & ۱۱ & ۰ \end{bmatrix} \qquad (پ-۱)$$

$$b=\begin{bmatrix} DAG۲ & A & B & C & D \\ A & ۰ & ۰ & ۱۸ & ۱۷ \\ B & ۰ & ۰ & ۰ & ۰ \\ C & ۰ & ۰ & ۰ & ۱۱ \\ D & ۰ & ۰ & ۱۱ & ۰ \end{bmatrix}$$

مجموع هزینه‌های ارتباطی هر سرویس با سرویس‌های دیگر به صورت ماتریس c است

جدول پ-۱: زمان اجرای سرویس‌ها.

Service	A	B	C	D
ExecutionTime	۵	۲	۷	۳

جدول پ-۲: پارامترهای گراف جهت‌دار ۱.

Service	Level	Count of Child	Communication Cost
A	۱	۲	۲۵
B	۲	۱	۷
C	۴	۰	۰
D	۲	۱	۱۱

جدول پ-۳: نرمال پارامترهای گراف جهت‌دار ۱.

Service	Level	Count of Child	Communication Cost
A	۰	۱	۱
B	۰٫۲۵	۰٫۵	۰٫۲۸
C	۱	۰	۰
D	۰٫۲۵	۰٫۵	۰٫۴۴

جدول پ-۴: پارامترهای گراف جهت‌دار ۲.

Service	Level	Count of Child	Communication Cost
A	۱	۲	۳۵
B	۰	۰	۰
C	$(۲+۴)/۲=۳$	۰٫۵	۵٫۵
D	$(۲+۴)/۲=۳$	۰٫۵	۵٫۵

$$c=\begin{bmatrix} sum & A & B & C & D \\ A & ۰ & ۵ & ۱۸ & ۲۷ \\ B & ۰ & ۰ & ۷ & ۰ \\ C & ۰ & ۰ & ۰ & ۱۱ \\ D & ۰ & ۰ & ۲۲ & ۰ \end{bmatrix} \quad (پ-۲)$$

تعداد دفعات ملاقات هر سرویس با سرویس‌های دیگر برای هر گراف جهت‌دار به صورت ماتریس‌های d و e و مجموع ملاقات هر سرویس با سرویس‌های دیگر به صورت ماتریس f خواهد بود

$$d=\begin{bmatrix} DAG۱ & A & B & C & D \\ A & ۰ & ۱ & ۰ & ۱ \\ B & ۰ & ۰ & ۱ & ۰ \\ C & ۰ & ۰ & ۰ & ۰ \\ D & ۰ & ۰ & ۱ & ۰ \end{bmatrix}$$

$$e=\begin{bmatrix} DAG۲ & A & B & C & D \\ A & ۰ & ۰ & ۱ & ۱ \\ B & ۰ & ۰ & ۰ & ۰ \\ C & ۰ & ۰ & ۰ & ۱ \\ D & ۰ & ۰ & ۱ & ۰ \end{bmatrix} \quad (پ-۳)$$

$$f=\begin{bmatrix} sum & A & B & C & D \\ A & ۰ & ۱ & ۱ & ۲ \\ B & ۰ & ۰ & ۱ & ۰ \\ C & ۰ & ۰ & ۰ & ۱ \\ D & ۰ & ۰ & ۲ & ۰ \end{bmatrix}$$

جدول پ-۵: نرمال پارامترهای گراف جهت‌دار ۲.

Service	Level	Count of Child	Communication Cost
A	۰٫۳۳	۱	۱
B	۰	۰	۰
C	۱	۰٫۲۵	۰٫۱۶
D	۱	۰٫۲۵	۰٫۱۶

جدول پ-۶: متوسط هر پارامتر برای هر سرویس.

Service	Average Level	Average Count of Child	Average Communication Cost
A	۰٫۱۷	۱	۱
B	۰٫۱۳	۰٫۲۵	۰٫۱۴
C	۱	۰٫۱۳	۰٫۰۸
D	۰٫۶۳	۰٫۳۸	۰٫۳

جدول پ-۷: اولویت هر سرویس.

Service	Priority	Rank
A	۱٫۸۳	۱
B	۰٫۲۶	۲
C	-۰٫۷۹	۴
D	۰٫۰۵	۳

جدول پ-۸: محاسبه MinEST و MaxEFT سرویس‌ها.

			DAG۱		DAG۲	Result
	EST	EFT	EST	EFT	MinEST	MaxEFT
A	۰	۵	۰	۵	۰	۵
B	۲۰	۲۲	-	-	۲۰	۲۲
C	۳۳	۴۰	۲۳	۳۰	۲۳	۴۰
D	۱۵	۱۸	۲۲	۲۵	۱۵	۲۵

اکنون هر درایه از ماتریس‌های c و f نرمال می‌شود (ماتریس‌های g و h) و در نهایت رتبه همکاری هر سرویس با سرویس‌های دیگر به دست می‌آید (ماتریس i)

$$g+h=i \Rightarrow$$

$$\begin{bmatrix} & A & B & C & D \\ A & ۰ & ۰٫۵ & ۰٫۵ & ۱ \\ B & ۰ & ۰ & ۱ & ۰ \\ C & ۰ & ۰ & ۰ & ۱ \\ D & ۰ & ۰ & ۱ & ۰ \end{bmatrix} + \begin{bmatrix} & A & B & C & D \\ A & ۰ & ۰٫۱۹ & ۰٫۶۷ & ۱ \\ B & ۰ & ۰ & ۱ & ۰ \\ C & ۰ & ۰ & ۰ & ۱ \\ D & ۰ & ۰ & ۱ & ۰ \end{bmatrix} \quad (پ-۴)$$

$$= \begin{bmatrix} & A & B & C & D \\ A & ۰ & ۰٫۶۹ & ۱٫۱۷ & ۲ \\ B & ۰ & ۰ & ۲ & ۰ \\ C & ۰ & ۰ & ۰ & ۲ \\ D & ۰ & ۰ & ۲ & ۰ \end{bmatrix}$$

مراجع

[1] X. Ye, Y. Yin, and L. Lan, "Energy-efficient many-objective virtual machine placement optimization in a cloud computing environment," *IEEE Access*, vol. 5, pp. 16006-16020, 2017.

[2] P. Bezerra, G. Martins, R. Gomes, F. Cavalcante and A. Costa, "Evaluating live virtual machine migration overhead on client's application perspective," in *Proc. Int. Conf. on Information*

Networking, ICOIN'17, pp. 503-508, Da Nang, Vietnam, 11-13 Jan. 2017.
[3] A. Beloglazov and R. Buyya, "Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers," *Concurrency and Computation: Practice and Experience*, vol. 24, no. 13, pp. 1397-1420, Sept. 2012.
[4] P. G. J. Leelipushpam, J. Sharmila, "Live VM migration techniques in cloud environment - a survey," in *Proc. IEEE Conf. on Information & Communication Technologies*, pp. 408-413, Thuckalay, India, 11-12 Apr. 2013
[5] W. S. Cleveland, "Robust locally weighted regression and smoothing scatterplots," *J. of the American Statistical Association*, vol. 74, no. 368, pp. 829-836, Dec. 1979.
[6] A. Verma, G. Dasgupta, T. K. Nayak, P. De, and R. Kothari, "Server workload analysis for power minimization using consolidation," in *Proc. of the Conf. on USENIX Annual Technical Conf.*, pp. 1-28, San Diego, CA, USA, 14-19 Jun. 2009.
[7] X. Fu and C. Zhou, "Virtual machine selection and placement for dynamic consolidation in cloud computing environment," *Frontiers of Computer Science*, vol. 9, no. 2, pp. 322-330, Feb. 2015.
[8] S. Esfandiarpoor, A. Pahlavan, and M. Goudarzi, "Structure-aware online virtual machine consolidation for datacenter energy improvement in cloud computing," *Computers & Electrical Engineering*, vol. 42, no. 1, pp. 74-89, Feb. 2015.
[9] D. Patel, R. K. Gupta, and R. Pateriya, "Energy-aware prediction-based load balancing approach with VM migration for the cloud environment," *Data, Engineering and Applications,* vol. 2, no. 1, pp. 59-74, Apr. 2019.
[10] J. Gao and G. Tang, "Virtual machine placement strategy research," in *Proc. IEEE Int. Conf. on Cyber-Enabled Distributed Computing and Knowledge Discovery,* pp 294-297, Beijing, China, 10-12 Oct. 2013.
[11] J. Wan, F. Pan, and C. Jiang, "Placement strategy of virtual machines based on workload characteristics," in *Proc. IEEE 26th Int. Parallel and Distributed Processing Symp. Workshops & PhD Forum,* pp. 2140-2145, Shanghai, China, 21-25 May 2012.
[12] A. J. Younge, *et al.*, "Efficient resource management for cloud computing environments," in *Proc. IEEE Int. Conf. on Green Computing,* pp. 357-364, Chicago, IL, USA, 15-18 Aug. 2010.
[13] M. B. Nagpure, P. Dahiwale, and P. Marbate, "An efficient dynamic resource allocation strategy for VM environment in cloud," in *Proc. IEEE Int. Conf. on Pervasive Computing, ICPC'15,* 5 pp., Pune, India, 8-10 Jan. 2015.
[14] X. Zheng and Y. Cai, "Dynamic virtual machine placement for cloud computing environments," in *Proc. IEEE 43rd Int. Conf. on Parallel Processing Workshops,* pp. 121-128, Minneapolis, MN, USA, 9-12 Sept. 2014.
[15] E. Asyabi and M. Sharifi, "A new approach for dynamic virtual machine consolidation in cloud data centers," *International J. of Modern Education and Computer Science*, vol. 7, no. 4, pp. 61-66, Apr. 2015.
[16] D. Gmach, *et al.*, "Capacity management and demand prediction for next generation data centers," in *Proc. IEEE Int. Conf. on Web Services, ICWS'07,* pp. 43-50, Salt Lake City, UT, USA, 9-13 Jul. 2007.
[17] T. Wood, *et al.*, "Profiling and modeling resource usage of virtualized applications," in *Proc. ACM/IFIP/USENIX Int. Conf. on Distributed Systems Platforms and Open Distributed Processing,* pp. 366-387, Berlin, Germany, 2008.
[18] J. Sonnek, *et al.*, "Starling: minimizing communication overhead in virtualized computing platforms using decentralized affinity-aware migration," in *Proc. IEEE 39th Int. Conf. on Parallel Processing,* pp. 228-237, San Diego, CA, USA, 13-16 Sept. 2010.
[19] L. Hu, *et al.*, "Net-cohort: detecting and managing vm ensembles in virtualized data centers," in *Proc. of the 9th Int. Conf. on Autonomic Computing*, pp. 3-12, San Jose, CA, USA, 18-21 Sept. 2012.
[20] J. T. Piao and J. Yan, "A network-aware virtual machine placement and migration approach in cloud computing," in *Proc. IEEE 9th Int. Conf. on Grid and Cloud Computing,* pp. 87-92, Nanjing, China, 1-5 Nov. 2010.
[21] K. Tsakalozos, M. Roussopoulos, and A. Delis, "Hint-based execution of workloads in clouds with nefeli," *IEEE Trans. on Parallel and Distributed Systems*, vol. 24, no. 7, pp. 1331-1340, Jul. 2012.
[22] Z. Xiao, Q. Chen, and H. Luo, "Automatic scaling of internet applications for cloud computing services," *IEEE Trans. on Computers*, vol. 63, no. 5, pp. 1111-1123, Nov. 2012.
[23] S. Srikantaiah, A. Kansal, and F. Zhao, "Energy aware consolidation for cloud computing," in *Proc. of the Conf. on Power Aware Computing and Systems*, vol. 10, pp. 46-56, Berkeley, CA, USA, 7-10 Dec. 2008.
[24] M. Cardosa, M. R. Korupolu, and A. Singh, "Shares and utilities based power consolidation in virtualized server environments," in *Proc. IFIP/IEEE Int. Symp. on Integrated Network Management,* pp. 327-334, New York, NY, USA, 1-5 Jun. 2009.
[25] M. Noshy, A. Ibrahim, and H. A. Ali, "Optimization of live virtual machine migration in cloud computing: a survey and future directions," *J. of Network and Computer Applications*, vol. 110, no. 1, pp. 1-10, May 2018.
[26] A. Alarifi, *et al.*, "Energy-efficient hybrid framework for green cloud computing," *IEEE Access*, vol. 8, pp. 115356-115369, Jun. 2020.
[27] P. Geetha and C. R. R. Robin, "Power conserving resource allocation scheme with improved QoS to promote green cloud computing," *J. of Ambient Intelligence and Humanized Computing*, vol. 12, no. 1, pp. 7153–7164, Jul. 2020.
[28] N. J. Kansal and I. Chana, "Energy-aware virtual machine migration for cloud computing-a firefly optimization approach," *J. of Grid Computing*, vol. 14, no. 2, pp. 327-345, Feb. 2016.
[29] J. Meshkati and F. Safi-Esfahani, "Energy-aware resource utilization based on particle swarm optimization and artificial bee colony algorithms in cloud computing," *The J. of Supercomputing*, vol. 75, no. 5, pp. 2455-2496, May 2019.
[30] G. Singh, M. Malhotra, and A. Sharma, "A comprehensive study on virtual machine migration techniques of cloud computing," *Applications of Computing, Automation and Wireless Systems in Electrical Engineering,* vol. 553, no. 1, pp. 591-603, Jun. 2019.
[31] K. Park and V. S. Pai, "CoMon: a mostly-scalable monitoring system for PlanetLab," *ACM SIGOPS Operating Systems Review*, vol. 40, no. 1, pp. 65-74, Jun. 2006.

محمد رستمی در سال ۱۳۹۰ در رشته مهندسی کامپیوتر گرایش نرم‌افزار مدرک کارشناسی خود را از دانشگاه آزاد اسلامی واحد اصفهان دریافت و سپس در سال ۱۳۹۲ مدرک کارشناسی ارشد مهندسی کامپیوتر در گرایش نرم‌افزار را از دانشگاه اهواز اخذ نموده است. در حال حاضر دانشجوی دکتری دانشگاه کاشان در رشته مهندسی کامپیوتر گرایش نرم‌افزار و الگوریتم می‌باشد. زمینه‌های تحقیقاتی مورد علاقه ایشان اینترنت اشیاء، شبکه‌های نرم افزار محور و کیفیت سرویس است.

سلمان گلی بیدگلی مدرک کارشناسی خود را در رشته معماری کامپیوتر در سال ۱۳۸۷ از دانشگاه کاشان و مدارک کارشناسی ارشد و دکتری را در رشته مهندسی معماری کامپیوتر به ترتیب در سال‌های ۱۳۹۰ و ۱۳۹۶ از دانشگاه اصفهان اخذ نموده است. در حال حاضر استادیار گروه مهندسی کامپیوتر دانشگاه کاشان می‌باشد. ایشان در سال‌های اخیر نویسنده چندین مقاله فنی در مجلات و کنفرانس‌های شبکه‌های کامپیوتری و مخابرات بوده است. زمینه‌های پژوهشی مورد علاقه ایشان شبکه‌های بی‌سیم، قابلیت اطمینان، اینترنت اشیاء و بلاکچین است.