خدمات با در نظر گرفتن هزینه و عملکرد الگوریتم های بارگزاری متعادل در ابر

چکیده

تامین منابع مورد تقاضا باعث میشود ابر به عنوان یک فناوری پیشرو شناخته شود .تمام ارائه دهنده های خدمات ابری، منابع رایانشی مناسب با نوع رابط کاربری، نوع نمونه و سیاست قیمت گزاری خود را پیشنهاد میدهند.یک کارگزار خدمات مبتنی بر ابر، برای یافتن ارائه دهندگان خدمات مناسب واسطه گری میکند طوری که مبادله مناسب بین قیمت و عملکرد را در نظر میگیرد.از سوی دیگر، تعادل بارگزاری میان منابع ابری مختلف، تضمین استفاده کارآمد از زیر ساخت های فیزیکی را میدهد، و به طور همزمان زمان اجرا را به حداقل میرساند.این باعث میشود که کارگزاران خدمات و بارگزاری متعادل در بین مهم ترین مسائل سیستم های محاسبات ابری قرار بگیرد.هدف از این مقاله پیشنهاد سه الگوریتم مختلف بارگزاری متعادل ابری و یک الگوریتم بارگزاری متعادل است.شبیه سازی مبتنی بر استقرار تصدیق میکند که الگوریتم پیشنهادی ما هزینه ها را کاهش میدهد و به طور همزمان شاهد موفقیت در خدمات اجرایی هستیم.

1.مقدمه

یک ابر شامل یک مرکز داده بزرگ یا گروه هایی از مرکز داده های بزرگ است، که ممکن است در یک یا چند منطقه جغرافیایی، جایی که ابرها منابع محاسبات نامحدود را فراهم می کنند، واقع شده باشند، که این منطقه برای رفع تقاضای کاربر مورد اطمینان است.یک ابر میتواند توسط سرمایه گزاران، دولت ها و ارائه دهندگان خدمات، میزبانی شود(Bernstien et al.,2011,2009).محتوا، ذخیره سازی و محاسبات، قادر به ارائه خدمات در هرجایی سرتاسر شبکه هستند، که با عنوان "درون ابری" خوانده میشود(Bernstien and Vij,2010).در یک سناریو قابل

همکاری، ابر ها باید به منظور تبادل اطلاعات، قادر به تشخیص یکدیگر باشند(Vecchiola et al., 2011).زیر ساخت ابرها توسط سرویس هایی ارائه میشوند که نه تنها فقط استفاده میشوند، بلکه با کمک مجازی سازی نصب، مستقر و تکرارهم میشوند.این خدمات در روند کسب و کار های پیچیده اعمال میشود، که بیشتر توافقنامه سطح سرویس(SLAs) را پیچیده تر میکند.به عنوان مثال، به دلیل تغییر اجزاء، بار کاری، شرایط خارجی، سخت افزار و شکست های نرم افزار، SLA های ایجاد شده ممکن است نقض شوند.کاربر تکراری، در طی مذاکرات SLA و خدمات اجرا، با سیستم مبادرت میکند(که معمولا برای خرابی ها ضروری است)، که ممکن است چالشی برای یک ابر رایانه ای موفق باشد.Garg et al.(2014) تکنیک برنامه ریزی SLAرا که شامل مدیریت VM با حجم کاری متغیر است را پیشنهاد داد.

یک مورد استفاده ی از رویکرد "درون ابر"ی توسط Buya et al. (2009) کشف شد که شامل معاملات بازار از طریق کارگزاران است.در چنین موارد استفاده ای، نهاد کارگزاری یک میانجی بین استفاده کنندگان و ارائه دهندگان ابر به طور متقابل است، که در حمایت از سابقه انتخاب ارائه دهنده، بهتر است مطابق با نیاز های کاربر باشد.یکی دیگر از خدمات افزون کارگزاری، استقرار و مدیریت آسان خدمات فائوس بدون در نظر گرفتن ارائه دهنده منتخب، از طریق رابط یکنواخت است.ارزیابی کارگزار در آزمون دنیای واقعی وقت گیر و هزینه بر است، زیرا حجم زیادی از منبع ابری برای دستیابی به نتایج واقعی و قابل اعتماد، مورد نیاز است.یک روش امیدوارانه و به صرفه تر برای فرآیند ارزیابی کارگزار، یک برنامه محیط شبیه سازی است.

در این پژوهش، محققان تمرکز خود را روی تعادل بارگزاری در میان مراکز داده و ماشین های مجازی، گذاشته اند; محدودیت های کارآمد بارگزاری متعادل ابر، برای ایجاد و توسعه یک الگوریتم متعادل کننده بارگزاری جدید، انگیزه ما را برانگیخته میکند.الگوریتم های پیشنهادی، تمام پردازش ها و زمان های پاسخ را کاهش میدهد، تا آنجایی که وظایف به طور کارآمدی به منابع فیزیکی تخصیص داده شده باشد.الگوریتم پیشنهادی به جای الگوریتم های پیشنهادی قبلی ارائه شده بود.در این پژوهش، ابر افرینی و الگوریتم تعادل بارگزاری برای یک محیط رایانش ابری

پیشنهاد شده است.بحث در مورد نتایج شبیه سازی سه تا از الگوریتم های کارگزاری ابر و الگوریتم بارگزاری متعادل نیز ارائه شده است.

2.کار های مرتبط

در حال حاضر بسیاری از سازمان های استاندارد، روی تعیین استاندارد های معین برای محاسبات ابری کار میکنند.این استاندارد ها توسط سازمان های استاندارد عظیم ابر ارائه شده اند.در حال حاضر کاربران با چالشی برای انتخاب ابر مناسب برای برآورده شدن نیاز های خود مواجه هستند.استفاده از سرویس بارگزاری ابر متوسط برای پیدا کردن یک ارائه دهنده ی خاص، از الزامات آن ها جهت یک تحقیق امیدوارکننده است(Jrad et al., 2012).دو مولفه پایه و اساسی بارگزاری ابر، تامین منابع و برنامه ریزی آن است.(Van den Bosche et al.,(2010 یک رویکرد برنامه ریزی شده برای یک ابر ترکیبی پیشنهاد میدهد، که عملیات را از لحاظ مقیاس پذیری، کاهش هزینه و امکان پذیری آن اجرایی میکند.از طریق برنامه نویسی اعداد صحیح باینری، تحقیقات آن ها کاربران را در تصمیم گیری برای بخش های خودکار، پشتیبانی میکند.محققان همچمنین ادعا میکنند که ابتدا به مشکلات مدیریت منبع در ابر های ترکیبی با این تکنیک پرداخته اند.

Wickremasinghe et al.,(2010) ابزار آنالیز ابر را با بهره گیری از یک مطالعه موردی بر اساس یک برنامه شبکه اجتماعی، به منظور مدلسازی و ارزیابی مستقر در ابر، در دنیای واقعی پیشنهاد داده است.آنها در کارشان، شیوه ای را نشان میدهند که یک ابر کارگزار خدمات میتواند منابع سیستم را بر اساس برنامه های کاربردی در مکان های مختلف جغرافیایی، بهینه سازی کند.علاوه بر این، سه ماشین مجازی که با الگوریتم های بارگزاری متعادل کار میکند و دو الگوریتم بارگزاری ابر را برای آزمایش هایشان روی ابزار تحلیلگر ابر پیشنهاد دادند.

Rochwerger et al.,(2011) نشان داد که اگرچه کمبود تعاملات داخلی وجود دارد، اما بین محدودیت های دیگر، در فن آوری های فعلی، ابر های متعهد، آینده وسیعی را در بر میگیرند.همچنین این نشان میدهد که برنامه ریزی و تامین منابع کارآمد، کارکرد بارگزاری ابر را افزایش میدهد.

Ferreto et al.,(2011) روی مدیریت منابع کار کرد که راه حل های موجو برای تثبیت سرور را گسترش داد.راه حل پیشنهادی آنها راه حل محدودی را تعریف میکند که با ماشین مجازی با ظرفیت معین، مهاجرت کرده ولی ماشین مجازی با استفاده مداوم برای به حداقل رساندن استفاده از سرور فیزیکی، مهاجرت نمیکند. Ferreto et al., (2011) این رویکرد را تقویت پویا با کنترل مهاجرت نامید.نتایج آنها نشان میدهد که با سیستم کنترل مهاجرت، مراکز داده بسیار مفید میشوند.

Tordson et al.,(2012) یک معماری بارگزاری ابری برای مدیریت ماشین مجازی در محیط رایانش ابری مخلوط ارائه کرد.ائ همچنین یک برنامه جایگزاری الگوریتم بهینه سازی به همراه برنامه ی فرمولاسیون برنامه نویسی دو دویی ارائه کرد که باعث کاهش قیمت عملکرد ها می شوند.بستر نرم افزاری Aneka(آنکا) Vecchiola et al ., (2011) شامل مدیریت منابع تهیه شده از خوشه ها، شبکه ها و ابر هاست.محققان ثابت میکنند که آنکا از کیفیت خدمات پشتیبانی میکند، و از کنترل هدایت برنامه ها در ابر های مختلف آگاه است.نتایج تجربی کار آنها پیشنهاد میدهد که آنکا دارای توانایی اختصاص عاقلانه منابع به قصد کاهش زمان اجرای برنامه است.

Calheiros et al., (2012) یک معماری هماهنگ کننده ابر را پیشنهاد میدهد که در محیط های ابری خصوصی و عمومی کار میکند.این هماهنگ کننده ابری، کارگزاران و مراکز داده را در بازار InterCloud را ارائه میدهد، مسئول مدیریت مذاکرات منابع است و همچنین انتشارات را هم پیشنهاد میدهد.محققان راجع به تاثیر آن روی برنامه های قابل ارتجاع و اثربخشی معماری پیشنهاد شده روی سناریو های متعادل، بحث میکنند.

Quarati et al., (2013) الگوریتم بارگزاری ابر را، بر اساس انواع مختلف شرایط برنامه ریزی برای ابر های ترکیبی، ارائه میدهد.نتایج شبیه سازی کارشان نشان میدهد که این الگوریتم بارگزاری، درآمد بارگزاری را به حداکثر رسانده و باعث رضایت کاربران میشود.محققان مکانیزم های صرفه جویی انرژی را به منظور افزایش درآمد کارگزاران، به کار میبرند.Akhter and Othman (2014) شروطی برای ماشین مجازی در غالب یک روش مصرف انرژی برای مرکز داده پیشنهاد دادند، و استراتژی های مختلفی را ترکیب کردند (Akhtar and Othman,2016).

Kessaci et al., (2013) کاهش به الگوریتم ژنتیک چند هدفه را برای بارگزاری ابر(MOGACB) پیشنهاد دادند، که برای رضایتمندی مشتری و سود کارگزار نشانه گرفته شده است.آنها همچنین ذکر میکنند که رضایت مشتری بیشتر مربوط به زمان پاسخ است تا موارد دیگر.

هدف از بارگزاری متعادل ابر، افزایش سرعت اجرای برنامه هاست.مطالعات متعددی روی بارگزاری متعادل در چند سال گذشته انجام گرفته است.Fang et al., (2010) درباره دو وظیفه برنامه ریزی برای استفاده از منابع بالا بحث میکند.برنامه پیشنهادی و برنامه ریزی سطح وظایف ماشین مجازی توسط برنامه Cloudsim که یک چاچوب شبیه سازی tool kitsimulation است، اثبات شده است.به هرحال عواملی مانند پهنای باند و هزینه را در کارشان، در طول دوره ی زمانی توازن بار در نظر نگرفته اند.

Randless و همکاران در سال 2010 بر اساس یک نمونه گیری تصادفی برای یک رفتار قدیمی و تعادل بارگیری خوشه بندی های فعال، با الهام از زنبور عسل تکنیک بارگزاری متعادل را پیشنهاد دادند.الگوریتم پیشنهادی آنها برای یک محیط ابری همراه با گره های نا همگن گسترش یافت اما روی یک سیستم با مقیاس کوچک شبیه سازی شد.توازن بارگزاری بر اساس دور رابین و تورینگ توسط Wickremasinghe و همکارانش در سال 2010 ارائه شد.آنها روی رفتار برنامه ای مطالعه میکردند که توسط شبیه سازی در یک محیط رایانش ابری در مقیاس عظیم گسترده شده بود.به هر حال کارایی توازن بار میتوانست بیشتر افزایش یابد، به طوری که در الگوریتم پیشنهادی ما نشان داده شده است.

منابع توازن بار ماشین مجازی از یک الگوریتم ژنتیک که توسط Hu و همکارانش در سال 2010 ارائه شده، استفاده میکند.این الگوریتم به عدم توازن بار و مشکلات هزینه مهاجرت، در الگوریتم های قدیمی پیشنهاد شده در کار های قبلی، میپردازد.به هر حال، مکانیزم مدیریت و نظارت هنوز برای تغییرات پویای ماشین های مجازی مورد نیاز است.

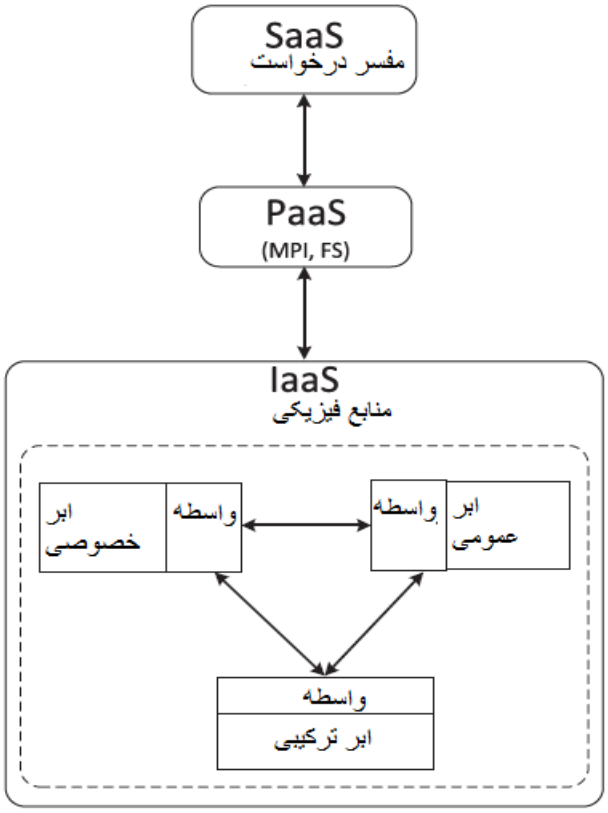
Wangetal در سال 2010 به طور فرصت طلبانه، بارگزاری متوازن و بارگزاری متوازن حداقل-حداقل را پیشنهاد میکند که الگوریتم های زمانبندی استاتیک برای سیستم بارگزاری متوازن است.این الگوریتم ها برای سه سطح از شبکه رایانش ابری گسترش یافته است.مدیر درخواست و مدیر خدمات و گره های سرویس، سه جزء از معماری در

چارچوب آنهاست.بارگزاری متوازن فرصت طلبانه تاش میکند تا کل سیستم را هنگام اجرا مشغول کند، زمانی که الگوریتم بارگزاری متوازن حداقل-حداقل زمان اجرا را به حداقل میرساند.

Maguluri و همکارانش در سال 2012 یک مدل تصادفی برای بارگزاری متوازن ارائه کردند که شامل محاسبات خوشه ها و تمرکز روی بارگزاری متوازن، در میان سرور هاست.آنها نشان میدهند که زمانبندی بهترین تناسب، بازده مطلوبی ندارد.بر اساس معیار های بهینه سازی مختلف، و محدودیت های کاربر، Lucas-Simarro و همکارانش در سال 2013 یک بارگزاری متوازن و الگوریتم زمانبندی برای ابر های مختلف ارائه دادند.ضمناکار آنها در الگوریتم پیشنهادیشان، هزینه بهره وری و بهبود کارایی را نشان میدهد.

یکی دیگر از الگوریتم های بارگزاری متوازنی که از زنبور عسل الهام گرفته شده، توسط Krishna در سال 2013 ارائه شده است.این الگوریتم، ماشین های مجازی در سرتاسر آن، به بارگزاری متوازن مجهز میکند و کارایی را به حداکثر میرساند و زمان وظایفی که در صف هستند را به حداقل میرساند که منجر به بهبود قابل توجهی برای زمان اجرای متوسط، میشود.این کار بیشتر تمرکز خود را به جای عوامل Qos، روی اولویت وظایف میگزارد.

اولین کار ما روی بارگزاری متعادل توسط Nahaand Othman در سال 2014 ارائه شده است.دور رابین و سرور وابستگی توسط Mahajan و همکارانش در سال 2013 ارائه شده است.نویسندگان عملکرد الگوریتم را توسط یک الگوریتم بارگزاری متعادل کارآمد، بهبود بخشیده اند.به هر حال، از آنجایی که تشخیص تاثیر بیشینه و کمینه شدن ساعات بارگزاری متوازن، بدون حداقل یک روز کاری، امکان پذیر نیست، در این نوع شبیه سازی ها، برنامه ای که برای یک ساعت بارگزاری میشود، ممکن است منجر به نتایج دقیقی نشود.بارگزاری متوازنی که از مدل پرواز آتشین استفاده میکند، در سال 2014 توسط Florence و Shanthi ارائه شد.شبیه سازی این مدل در یک محیط با مقیاس کوچک انجام شد.الگوریتم های وظایف ماشین مجازی مربوط به بازگزاری است را Domanal و Reddy در سال 2014 ارائه کردند.پاسخ الگوریتم را در یک مورد پویا و حجم کاری ترکیبی، مشخص کردند.Sun و همکارانش در سال 2014 یک تجزیه و تحلیل جامع روی یک رویکرد انتخابی از خدمات ابری انجام دادند و یک شکاف تحقیقاتی بالقوه در سرویس ابری انتخابی، یافتند.



شکل 1.مدل لایه ای سرویس ابری

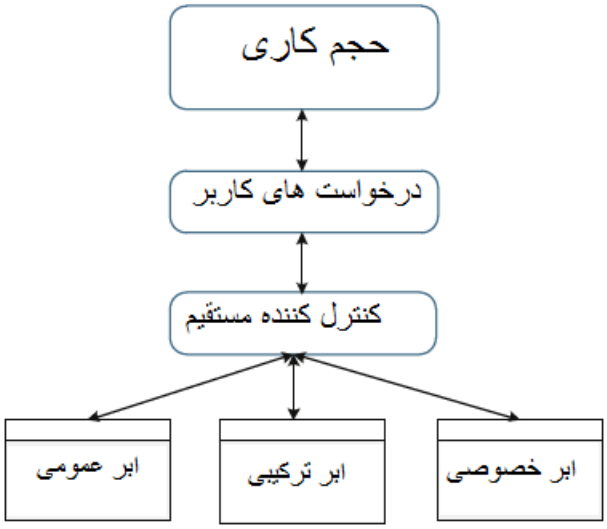
3.مدل های سرویس ابری

3-1.مدل های لایه ای سرویس های ابری

IaaS ، PaaS و SaaS سه تا از ساده ترین لایه های سرویس ابری است.شکل 1 مدل سرویس ابری را نشان میدهد.بالاترین لایه، لایه ی SaaS است که مرتبط با الزامات اجرای برنامه است.این لایه الزامات پلتفرم را برای برنامه ترجمه میکند و به لایه ی PaaS میفرستد.

سپس لایه ی PaaS یک کتابخانه گسترش یافته شامل محیط اجرای برنامه، مقیاس گزاری و جداسازی برنامه را آماده میکند.این لایه از طریق یک واسطه ی سنتی عمل میکند، و یک پل بین زیر ساخت و نیاز های برنامه ایجاد میکند.اما ساختار حقیقی مانند، ماشین های مجازی مورد نیاز را تعیین نمیکند.با این حال، نمای بالاتری از واحد های اجرایی مانند، موضوعات، فرآیند و وظایف را نشان نمیدهد.Microsoft Azure و Google App Engine نمونه هایی در دنیای واقعی از سرویس های ابری هستند.

لایه ی IaaS با تعدادی از فرآیند ها سر و کار دارد مانند اندازه دیسک، ماشین مجازی و اتصال شبکه.این لایه تامین منابع مجازی را تعیین میکند.کارگزاران هر ابر نگران این هستند که با در دسترس بودن منابع سیستم، و اطلاع از هزینه های منابع، و کارایی زمان اجرا، نسبت به کارگزاران دیگر چگونه است.الگوریتم پیشنهادی ما برای کارگزاران در لایه ی IaaS ارائه شده است.کارگزاری در لایه های SaaS و PaaS نسبت به معیار های متمایز اجراییشان، باهم تفاوت دارند.کارگزاری لایه ی SaaS بر اساس SLA و نیاز های کاربر است در حالی که کارگزاری لایه ی PaaS بر اساس زمان و پشتیبانی استقرار است.



شکل2.حجم کاری و جریان درخواست های منبع

3-2.حجم کاری و مدل منبع

طرح زدن برای بارگزاری یک کار واقعی، برای یک ابر، کار دشواری است.در سناریو های دنیای واقعی، بارگزاری کار IaaS به طور پویایی قابل تغییر و هیچ الگوی ثابتی ندارد.با این حال برای اعداف شبیه سازی، میتوانیم از ردیابی حجم کاری یا بارگزاری های تولید شده استفاده کنیم.منابع به عنوان تقاضا از حجم کاری فعلی ارائه شده است.شکل 2 مدل کار بارگزاری و مدل منابع را نشان میدهد.حجم کاری به سمت کنترل کننده مستقیم به عنوان یک درخواست کاربر، فرستاده میشود.بر اساس تقاضا، کارگزار با دیگر کارگزار های منابع برای منابع در دسترس ارتباط برقرار میکند.در میان منابع موجود،کارگزاران میفهمند که بهترین منابع آنهایی هستند که بر اساس محدودیت های

کاربران تعیین میشوند.هنگامی که کارگزاران منابع مناسب را پیدا میکنند، از طریق تخصیص و سیاست های تعیین شده، پردازش میشود.

4.انتقال داده و پردازش درخواست کاربر

تاخیر انتقال داده(Dt) به تاخیر شبکه(NI) و زمان انتقال اطلاعات(Dtr) بستگی دارد.تاخیر انتقال داده بر اساس میلی ثانیه (ms) اندازه گیری میشود.تاخیر شبکه در بین مناطق به عنوان یک ماتریس تاخیر بیان میشود، که یکسری داده فرضی است که از قاعده توزیع پواسون تبعیت میکند. زمان انتقال داده به اندازه داده های هر درخواست(RS) و فشار پهنای باند(Bp)، بستگی دارد.بنابراین زمان انتقال داده میتواند به عنوان (*) معنی شود.

$$D_t = N_l + \frac{R_s}{B_p} \qquad (1)$$

پهنای باند برای هر کاربر از کل پهنای باند در دسترس(T_b) است و درخواست همزمان کاربران بین دو منطقه (R_n) است.پهنای باند برای هر کاربر از فرمول زیر محاسبه میشود:

$$B_p = \frac{T_b}{R_n} \qquad (2)$$

کنترلر مرکز داده درخواست های پایه کاربر را دریافت کرده و آن ها را به زیر ابرهای کوچک تر تقسیم میکند. سپس درخواست های زیر ابرهای کوچک تر برای پردازش بیشتر از طریق متعادل ساز بار به VM ها واگذار میگردد. زمان پردازش مرکز داده بر حسب زمان دریافت اولین پاسخ از زیرابرها محاسبه میشود. بنابراین ، زمان پردازش مرکز داده برابر است با مجموع زمان مورد نیاز برای پردازش درخواست کاربر تا هنگامی که به VM واگذار میشود. زمان مورد نیاز برای پردازش درخواست یک کاربر را به عنوان زمان پاسخ دهی مرکز داده در نظر میگیرند. ما زمان پردازش مرکز داده و زمان پاسخ دهی آن را محاسبه کردیم.

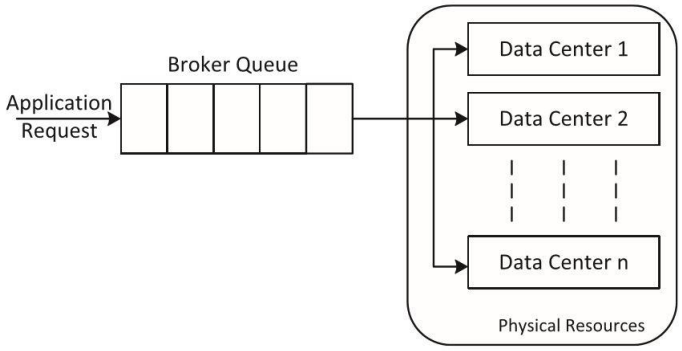
5 . الگوریتم پیشنهادی

5_1. توصیف الگوریتم پیشنهادی کارگزاری ابر

کارگزار ابر مسئولیت رسیدگی درخواست های کاربران و ارائه دهندگان منابع را بر عهده دارد. اولین اقدامات در زمینه کارگزاری ابر توسط Naha et al. (2015) صورت گرفت. برای اختصاص منابع بر اساس درخواست کاربر ، کارگزار باید ابتدا تمامی منابع در دسترس را شناسایی کند. کاربرهای مختلف باید برای تخصیص منابع دارای محدودیت های مختلفی باشند. از طرف دیگر ، هر ارائه دهنده SLA خود را نگهداری میکند. کارگزار ابر همواره بدون تخطی از SLA ، رضایت کاربران را در نظر میگیرد. پس از شناسایی منابع در دسترس ، کارگزار منابع مورد نیاز کاربر را بر اساس وظیفه آن تخصیص میدهد. عملکرد کارگزاری به منابع موثر شناسایی شده و تخصیص آن ها بستگی دارد. شکل3 سیستم صف را برای کارگزار پیشنهادی نشان میدهد.

5_1_1. کارگزاری با در نظر گرفتن هزینه

برای شناسایی منابع قابل دسترس کاربران از دو رویکرد متفاوت که هرکدام دارای یک الگوریتم منحصر به فرد هستند ، استفاده میکنیم. الگوریتم انتخاب منابع برحسب هزینه و بار در بخش الگوریتم1 مشخص شده است. در این الگوریتم regionalist شامل لیستی از مناطقی است که مراکز داده قرار دارند. متغیر dcCostLoadList مقادیر هزینه و بار را برای تخصیص های آتی نگهداری میکند. Sm مقدار کم هزینه ترین dcName را ذخیره میکند. الگوریتم های کارگزاری با در نظر گرفتن هزینه همواره کم هزینه ترین VM را انتخاب میکنند. همچنین در صورت یافتن یک VM با هزینه پایین تر ، آن را به لیست اضافه میکند.



شکل3. فرآیند صف کارگزار

```
   Input: userBase, regionalList
   Output: dcName
 1 Fetch region of the requested sender;
 2 if regionalList is not NULL then
 3 |  listSize ← Size(regionalList);
 4 |  if listSize is 1 then
 5 |  |  dcName ← regionalList.get(0);
 6 |  else
 7 |  |  dcCostLoadList ← regionalist sort by cost then by load;
 8 |  |  for all dcp in dcCostLoadList do
 9 |  |  |  if dcCostLoadList.get(sm) > dcCostLoadList.get(dcp)
   |  |  |   then
10 |  |  |  |  sm ← dcp;
11 |  |  |  end
12 |  |  end
13 |  |  dcName ← dcCostLoadList.get(sm);
14 |  end
15 end
16 Return dcName
```

الگوریتم1. الگوریتم شناسایی منابع (هزینه و بار)

این الگوریتم لیستی از مراکز داده را نگهداری میکند که محل قرارگیری مراکز داده را مشخص میکند. در صورتی که درخواست کاربری توسط کارگزار دریافت شود ، کارگزار ابتدا محل ارسال درخواست را از طریق لیست های مذکور بازیابی میکند. سپس با جستجو مراکز داده ای که در همان منطقه (محلی که درخواست ارسال شده) قراردارند را مشخص میکند. سپس این مراکز داده در آرایه ای و بر اساس تاخیر شبکه نگهداری میشوند.

تاخیر بین مشتری و مرکز داده را به عنوان تاخیر شبکه در نظر میگیریم. سپس اگر تنها یک مرکز داده در منطقه ی مورد نظر یافت شود ، دیگر مراکز داده که در آن منطقه قرار ندارند نیز مجددا بر اساس تاخیر شبکه به انتهای آرایه-ی مذکور اضافه میشوند. در گام نهایی ، اگر بیش از یک مرکز داده در آرایه قرار داشته باشند ، آن مرکزی انتخاب میشود که کمترین هزینه و کمترین بار را داشته باشد.

5_1_2. الگوریتم کارگزاری با در نظر گرفتن بار

این الگوریتم تمام درخواست ها را بین همه ی مراکز داده مورد در دسترس توزیع میکند. برای انتخاب منابع از تکنیک بارگزاری متعادل که در بخش الگوریتم2 توصیف شده ، بهره میبریم. عموما در صورتی که وظایف بین مراکز داده تقسیم شوند زمان پردازش کاهش میابد ، با این حال امکان افزایش هزینه های اجرایی نیز وجود دارد.

```
   Input: userBase, regionalList
   Output: dcName
 1 Fetch region of the requested sender;
 2 if regionalList is not NULL then
 3     listSize ← Size(regionalList);
 4     if listSize is 1 then
 5         dcName ← regionalList.get(0);
 6     else
 7         getRandomNumber(listSize) ;
 8     end
 9     dcName ← dcCostLoadList.get(sm);
10 end
11 Return dcName
```

الگوریتم2. شناسایی منابع

3_1_5. کارگزاری با اولویت بار بر هزینه

این الگوریتم هزینه های مراکز داده را در جدولی مجزا نگهداری میکند. در طول شبیه سازی ، لیست کم هزینه ترین مرکز داده را با لیستی دیگر که شامل منطقه ی ارسال درخواست است ، تجمیع میکند. زمانی که لیست کم هزینه ترین مراکز داده آماده شد ، تمامی ترافیک دریافتی را بین مراکز داده با کمترین هزینه توزیع میکند. بخش الگوریتم3 اجرای گام به گام این الگوریتم را نشان میدهد.

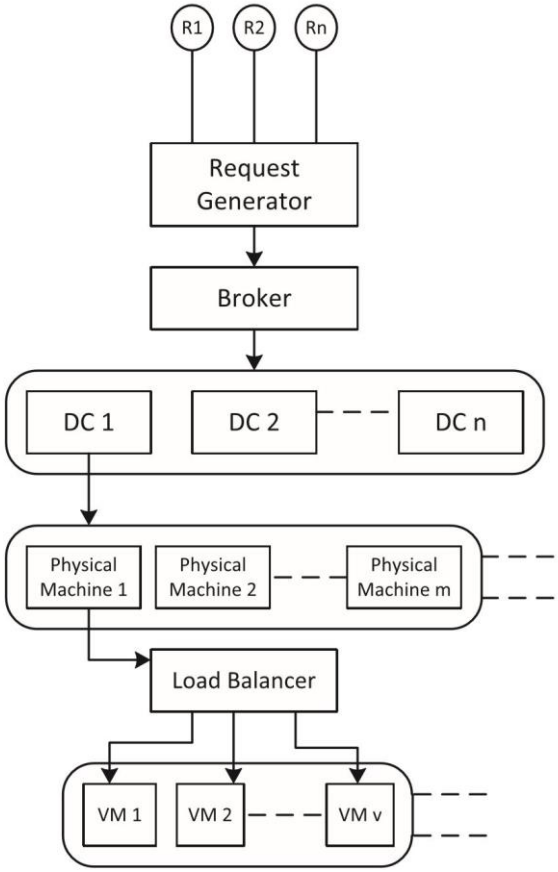
```
Input: userBase, regionalList, dcXCostList
Output: dcName
Fetch region of the requested sender;
if regionalList is not NULL then
    listSize ← Size(dcXCostList);
    if listSize is 1 then
        dcName ← regionalList.get(0);
    else
        getRandomNumber(listSize) ;
    end
    dcName ← dcXCostList.get(sm);
end
Return dcName
```

الگوریتم3. الگوریتم اولویت بار بر هزینه

2_5. روش های بارگزاری متعادل و الگوریتم پیشنهادی

برای بارگزاری متعادل میتوانیم از روش های ایستا و پویا استفاده کنیم. تنوع بالایی از بار در رایانش ابری مشاهده میشود. بنابراین ، الگوریتم های ایستا مناسب محیط های رایانش ابری نیستند. بارگزاری متعادل پویا به طور موفقیت آمیز در ابر به کار گرفته شده است و الگوریتم های پویا میتوانند به بار هایی متغیر در طول زمان رسیدگی کنند. الگوریتم بارگزاری متعادل مبتنی بر حالت (SBLB) پیشنهادی ما ، وظایف را به صورت پویا به هاست های بیکار واگذار میکند. همچنین وظایف را از صف وظایف به هاست های در دسترس واگذار میکند. با استفاده از این تکنیک از بارگیری بیش از حد یک هاست جلوگیری میشود. از طرف دیگر اما ، یک وظیفه نباید بیش از حد در صف وظایف باقی بماند.

زمانی که یک کاربر درخواستی را به سیستم ابر ارسال میکند ، کنترلر ابر درخواست برای پردازش بیشتر قبول میکند. کنترلر مرکز داده ، متعادل ساز بار و کارگزار خدمات منابع در دسترس را جستجو میکنند و وظیفه را بر اساس نوع درخوسات به یک VM متناسب واگذار میکنند. در هر مخزن مجازی منابع ، سطح متفاوتی از CPU ، منابع حافظه و پهنای باند شبکه در اختیار است. از طرف دیگر ، هر درخواست دریافتی از کاربر ملزومات و محدودیت های مختص به خود را دارد. بنابراین ، یافتن راه حل بهینه برای درخواست کاربران امری پیچیده محسوب میشود.



شکل4. روند کلی متعادل ساز بار VM

الگوریتم بارگزاری متعادل ، میزان بار در VM ها را متعادل میکند. زمانی که کارگزار یک مرکز داده را برای ارائه خدمت انتخاب میکند ، متعادل ساز بار VM بعدی بار را بین VM ها بر اساس چگونگی آن ها توزیع میکند. شکل4 روندکار کلی متعادل ساز بار را نشان میدهد. الگوریتم SBLB پیشنهادی ما دو جدول مجزا بر اساس حالت های VM ها را نگهداری میکند . هنگامی که یک VM به آستانه ی مصرفی خود میرسد ، حالت آن در جدول به " مشغول " تغییر می یابد. با این حال ، در صورتی که به آستانه ی مورد نظر نرسد در جدول با عنوان " در دسترس " قرار میگیرد.

کنترلر مرکز داده درخواست های کاربران را در صف قرار داده و آن ها ره به متعادل ساز بار ارسال میکند. سپس متعادل ساز بار VM های در دسترس را بر اساس درخواست کاربران از جدول مذکور بازیابی میکند. پس از

تخصیص درخواست به VM ، جدول نیز به روزرسانی میشود. در صورتی که کنترلر مرکز داده VM در دسترسی را پیدا نکند ، درخواست ها را در صف قرار میدهد تا زمانی که منابع دردسترس قرار گیرند. زمانی که امر پردازش در یک VM به پایان میرسد، متعادل ساز بار ، وظیفه ی دیگری را در اختیار VM قرار میدهد. مرکز داده به طور تصادفی منابع در دسترس را برای وظایفی که در انتظار به سر میبرد ، بررسی میکند. بخش الگوریتم4 شبه کد های این الگوریتم را نشان میدهد. در این الگوریتم ، vmSL لیست حالت های VM را نشان میدهد و vmSi اندیس حالت VM را نگهداری میکند. . لیست از VM های مشغول و در دسترس در متغیر vmAv و vmBu نگهداری میشوند.

```
   Input: vmSl
   Output: vmId
 1 if vmSl is NULL then
 2 |  vmSi ← getVmSl();
 3 else
 4 |  if (vmAv is NULL) or (vmBu is NULL) then
 5 |  |  for all vmSI do
 6 |  |  |  if i is Busy then
 7 |  |  |  |  Add in vmBu;
 8 |  |  |  else
 9 |  |  |  |  Add in vmAv
10 |  |  |  end
11 |  |  end
12 |  else
13 |  |  vmId ← getNextAvailableVm();
14 |  |  return vmId
15 |  end
16 end
17 if e.getId ← allocated then
18 |  vmBu.put(vmId, virtualMachineState.BUSY);
19 else if e.getId ← finished then
20 |  vmAv.put(vmId, virtualMachineState.AVAILABLE);
21 end
```

الگوریتم4. بارگزاری متعادل مبتنی بر حالت

6. راه اندازی و سناریو شبیه سازی

برای استنتاج شبیه سازی آزمایش ها از cloudAnalyst استفاده کردیم (wickremasinghe et al. 2010) . cloudAnalyst افزونه ای برای cloudSim است(calheiros el al 2011) . cloudSim یک چارچوب شبیه سازی مدرن برای رایانش ابری است که از مدل های مقیاس بزرگ در ابر پشتیبانی میکند. cloudSim از شبیه

سازی های مقیاس بزرگ ابر با مقدار کمی سربار (درمواردی بدون سربار) مصرف حافظه و زمان راه اندازی پشتیبانی میکند. برای تولید درخواست کاربر ، از آمار جهانی Facebook استفاده کردیم.

DC Physical machines configuration	
CPU speed	10,000 MIPS
CPU core	4
RAM	20 GB
Storage	1000 GB
Network bandwidth	10 Gbps
CPU architecture	x86
OS	Linux
Virtual machine monitor (VMM)	Xen
VM configuration	
CPU speed	1000 MIPS
RAM	1 GB
Storage	50 GB
Network bandwidth	100 Mbps

جدول1. پیکربندی ماشین

بر اساس آمار منتشر شده این وبسایت در 30 سپتامبر 2013 ، این وبسایت 1.19 میلیارد کاربر فعال در ماه از سراسر جهان داشته است. حدود 21.85٪ کل کاربران این شبکه از آمریکای شمالی بوده اند. ما اینطور در نظر گرفتیم که هر کاربر هر دو دقیقه یک درخواست جدید ایجاد میکند. کنترلر مرکز داده مسئولیت نگهداری اطلاعات حجم کار را بر عهده دارد. ساعت پیک در این منطقه بین ساعت 15 تا 17 GMT است.

مجموعا هفت مرکز داده را در دو منطقه ی متفاوت نمایش دادیم. منطقه ی اول جایی است که پایگاه کاربران قرار دارد. محل قرارگیری سایر مراکز داده نیز به طور تصادفی انتخاب شد. هر مرکز داده شبیه سازی شده شامل 5 ماشین فیزیکی و تعدادی ماشین مجازی است. با تغییر تعداد VM ها تا 15 عدد توانستیم 6 سناریوی مختلف طراحی کنیم.

ماشین های فیزیکی و پیکربندی VM در مرکز داده در جدول1 مشخص است. VM ها بر اساس سناریوی مورد استفاده ایجاد میشوند. کاربران و درخواست ها به ترتیب با عامل های 1000 و 100 گروه بندی میشوند. در برخی موارد ، سرعت پردازش به طور مساوی بین VM ها بر اساس سناریوی شبیه سازی تقسیم شد. حجم دستورات اجرایی به ازای هر دستور برابر 250 بایت است. هزینه ساعتی هر VM بر اساس حجم 1 گیگابایت در

هر ساعت بر اساس قیمت های وبسایت آمازون برابر با 0.10 دلار آمریکا است. ماتریس تاخیر و پهنای باند با استفاده از توزیع پوآسون تولید شده اند و تنها از مقادیر میانگین آن ها استفاده شده است. از همین تکنیک برای تولید سایز پایگاه کاربران و فاصله بین درخواست ها استفاده شد. جدول2 نام های DC را همراه با محل قرارگیری و هزینه بر حسب هر VM را برای مراکز داده نشان میدهد.

7. نتایج آزمایش و بحث پیرامون آن

ما الگوریتم های پیشنهادی خود را در محیط های ابری در مقیاس بزرگ شبیه سازی کردیم. زمان پردازش همواره برای الگوریتم با در نظر گرفتن بار (LA) کمتر است. الگوریتم با در نظر گرفتن هزینه (CA) در هزینه صرفه جویی میکند اما زمان پردازش بیشتری را صرف میکند. برای همه سناریو ها ، CA همیشه هزینه کمتری را نسبت به LA مصرف میکند. نتایج نشان میدهد که CA مرقون به صرفه تر است ؛ با این حال ، LA سریع ترین قدرت پردازشی را به ارمغان می آورد.

DC name	Region name	Region number	Cost per VM (US$/H)
DC1	North America	0	0.30
DC2	North America	0	0.38
DC3	North America	0	0.30
DC4	South America	1	0.15
DC5	South America	1	0.30
DC6	South America	1	0.15
DC7	South America	1	0.73

جدول2. محل مرکز داده و هزینه

در مقایسه با کارگزار خدمات بر اساس مجاورت خدمات (SPSB) که توسط wickremasinghe et al 2010 ارائه شده و توسط Naha and Brokering (2014) بازبینی شده ؛ به این نتیجه دست یافتیم که LA زمان پاسخ دهی کمتر با تعداد VM های کمتری را ارائه میکند و این در حالی است که CA همواره با زمان پردازش و پاسخ دهی بیشتری همراه بود. به علاوه ، الگوریتمی با در نظر گرفتن اولویت بار بر هزینه (LAOC) را توسعه دادیم که علاوه بر بهبود زمان پردازش و پاسخ دهی مقرون به صرفه نیز هست.

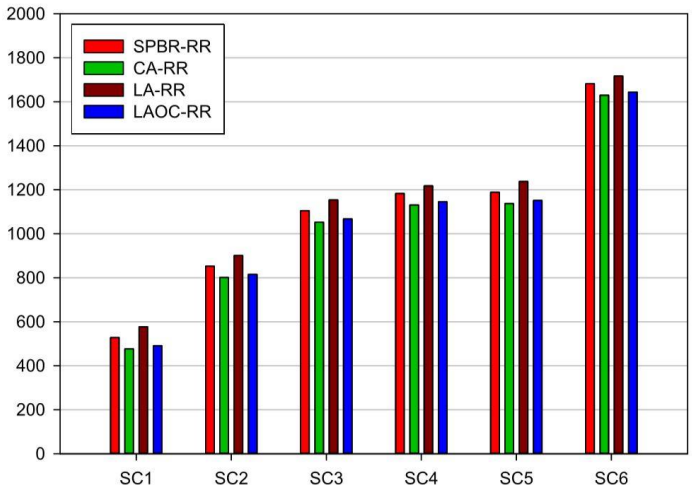
پژوهش ما به مکانیسم کارگزاری و الگوریتم انطباق با در نظر گرفتن SLA محدود است و نمیتوانیم معماری ارائه دهندگان خدمات ابری را دستخوش تغییر کنیم. یکی از محدودیت های شبیه ساز این است که قابلیت مدیریت و رسیدگی به شکست ها را ندارد. ترافیک داده کاربران با فرض فواصل زمانی 5 دقیقه ای تولید شده اند. با این حال ، در دنیای واقعی ترافیک کاربران به طور پیچیده تری تولید میشود. در ادامه تحلیلی از نتایج به صورت جزئی نشان داده شده است.

7_1. نتایج شبیه سازی کارگزاری ابر

الگوریتم پیشنهادی ما با در نظر گرفتن هزینه نسبت به سایر الگوریتم هایی که توسط wickremasinghe et al (2010) ارائه شده بودند ، مقرون به صرفه تر بود. با این که در این الگوریتم در هزینه ها صرفه جویی میشود اما عملکرد مربوط به زمان پردازش و پاسخ دهی دچار افت میشود. در ادامه ، الگوریتمی با در نظر گرفتن بار ، ارائه کردیم که باعث بهبود عملکرد مذکور میگردد. برای اینکه الگوریتم های پیشنهادی خود را مرقون به صرفه تر کنیم ، الگوریتم دیگری را با تلفیق الگوریتم های با درنظر گرفتن بار و هزینه توسعه دادیم که هم از لحاظ هزینه و هم از لحاظ عملکرد بهینه هستند. نتایج این سه الگوریتم در بخش های بعدی توضیح داده شده اند.

7_1_1. مجموع هزینه VM در الگوریتم کارگزاری پیشنهادی

به طور میانگین ، الگوریتم با در نظر گرفتن هزینه ، 5/5٪ از سایر الگوریتم های مرقون به صرفه تر است. الگوریتم کارگزاری مسیریابی بر اساس مجاورت خدمات (SPBR-RR) پیش از این ارائه شده بود. در این مقاله الگوریتم های کارگزاری با در نظر بار LA ، هزینه CA و با اولیت بار بر هزینه LAOC ارائه شده است. شکل5. مقایسه ای بین الگوریتم های ارائه شده در این مقاله و (SPBR-RR) را بر حسب هزینه نشان میدهد. زمانی که بر بار متمرکز شدیم ، هزینه ماشین های مجازی نیز افزایش یافت. در مقایسه با الگوریتم CA-RR هزینه کلی 11٪ نسبت به الگوریتم LA-RR افزایش داشته است. با این حال الگوریتم کارگزاری با در نظر گرفتن بار، زمان پردازش DC را بهبود میبخشد اما هزینه و زمان پاسخ دهی را تضعیف میکند.



شکل5. مجموع هزینه ماشین های مجازی در مقایسه با الگوریتم های کارگزاری

به علاوه ، الگوریتم LAOC را برای از بین بردن نقاط ضعف الگوریتم های توصیف شده ، توسعه دادیم. به این نتیجه رسیدیم که هزینه کلی VM در الگوریتم LA افزایش می یابد. اما الگوریتم LAOC پیشرفت چشم گیری را از نظر هزینه نشان میدهد. هزینه کلی VM در مقایسه با الگوریتم LA ، 8% کاهش داشته است.

7_1_2. زمان پردازش و پاسخ دهی الگوریتم های پیشنهادی

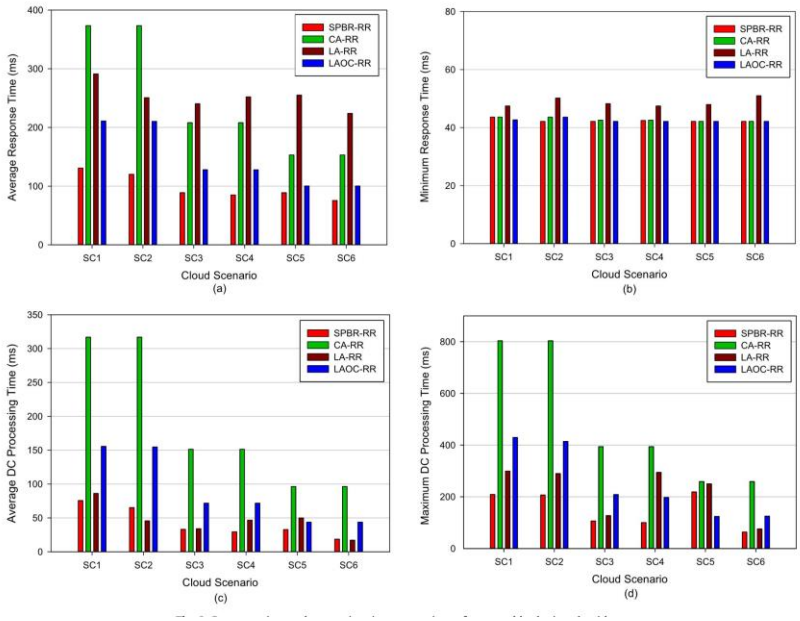
ما زمان پاسخ دهی و پردازشDC ها را برای الگوریتم های جدید و قدیمی محاسبه کردیم. شکل6 قسمت (a) و (B) مقایسه ی زمان پاسخ دهی حداقل و میانگین را نشان میدهد. شکل6 قسمت (c) و (d) زمان پردازش dc حداکثری و میانگین را بر اساس قسمت (a) شکل6 نشان میدهد. کاملا واضح است که متوسط زمان پاسخ دهی به طور چشم گیری افزایش داشته است ، این بدان معنی است که زمان پاسخ دهی با کاهش تعداد VM ها ، افزایش می یابد. با مقایسه الگوریتم های با در نظر گرفتن هزینه و بار ، به این نتیجه میرسیم که زمان پاسخ دهی برای سناریو های 1 و 2 کاهش یافته است.

با این حال ، مجموع زمان پاسخ دهی برای 4 سناریویِ باقی مانده افزایش می یابد. در سناریو های 1 و 2 ، 37٪ بهبود مشخص شد. از طرف دیگر ، مجموع زمان پاسخ دهی برای 4 سناریویِ دیگر 37٪ تنزل یافته است. برای الگوریتم LAOC کاهش واضحی در زمان پاسخ دهی مشاهده میشود. استفاده از این الگوریتم در سناریو های 3 ، 4 ، 5 بیشترین کاهش ها را در زمان پاسخ دهی نشان داد.

از طرف دیگر ، شبیه سازی با سناریو های 1 و 2 بهبود کمتری را نسبت به دیگر سناریو ها نشان داد. با این حال ، به طور متوسط زمان پاسخ دهی 43٪ برای همه ی سناریو ها نسبت به الگوریتم LB بهبود داشته است.

شکل6 (b) مقایسه بین دو الگوریتم را از نظر حداقل زمان پاسخ دهی نشان میدهد. همانطور که در شکل مشخص است ، حداقل زمان پاسخ دهی در الگوریتم های کارگزاری پیشنهادی ما با افزایش کوچکی همراه بوده است. با این حال ، در برخی سناریو ها به خصوص 1 ، 4 و 6 تفاوت آنچنانی در حداقل زمان پاسخ دهی برای SPBR-RR و CA-RR وجود ندارد. الگوریتم LA برای تمام 6 سناریو با افزایش حداقل زمان پاسخ دهی رو به رو است. بر اساس نتایج بدست آمده از شبیه سازی الگوریتم ها با در نظر گرفتن بار و هزینه ، حداقل زمان پاسخ دهی 14٪ درصد نسبت به الگوریتم با در نظر گرفتن هزینه افزایش داشته است. الگوریتم LAOC 13% بهبود در حداقل زمان پاسخ دهی در مقایسه با LA نشان داده است.

همانطور که در شکل6. (c) زمان پردازش متوسط DC در الگوریتم های CA-RR در مقایسه با الگوریتم SPBR-RR افزایش چشم گیری داشته است. برای بهبود عملکرد الگوریتم جدیدی با در نظر گرفتن بار طراحی کردیم که در مقایسه با الگوریتم با در نظر گرفتن هزینه ، منجر به بهبود شد.



شکل6. مقایسه زمان پاسخ دهی و پردازش در الگوریتم های کارگزاری

با توجه به شکل6. (c) الگوریتم با در نظر گرفتن بار ، متوسط زمان پردازش DC را در مقایسه با الگوریتم با در نظر گرفتن هزینه ، کاهش داد. به طور متوسط 73٪ کاهش در زمان پردازش DC مشاهده شد. زمان پردازش DC الگوریتم LAOC-RR در مقایسه با LA-RR تنها در سناریوی 5 کاهش داشت ، که بیانگر 6٪ بهبود است. برای سایر سناریو ها ، زمان پردازش DC الگوریتم LAOC-RR افزایش چشم گیری داشت. برای این الگوریتم در سناریو های ابری 1 و 2 زمان پردازش DC به طور شگفت انگیزی افزایش داشت.

شکل6. (d) تفاوت حداکثر زمان های پردازش DC را برای الگوریتم های جدید و قدیمی نشان میدهد. با توجه به شکل ، زمان پردازش Dc در الگوریتم SPBR-RR در مقایسه با CA-RR در تمام سناریو ها ، افزایش داشته است. به طور کلی این نتایج نشان میدهند که عملکرد با توجه به زمان پردازش و پاسخ دهی با افت رو به رو شده است. حداکثر زمان پردازش DC در الگوریتم LA در مقایسه با CA-RR کاهش داشته است. با این حال در سناریویی 5 ، اختلاف در حداکثر زمان پردازش DC بسیار ناچیز بوده است. حداکثر زمان پردازش DC در سناریو های 4 و 5 برای الگوریتم های LA-RR و LAOC-RR کاهش داشته است. برای این دو سناریو حداکثر زمان پردازش DC به طور متوسط با 112٪ کاهش همراه بوده است. با این حال ، حداکثر زمان پردازش DC در شبیه سازی دیگر سناریو ها افزایش داشته است.

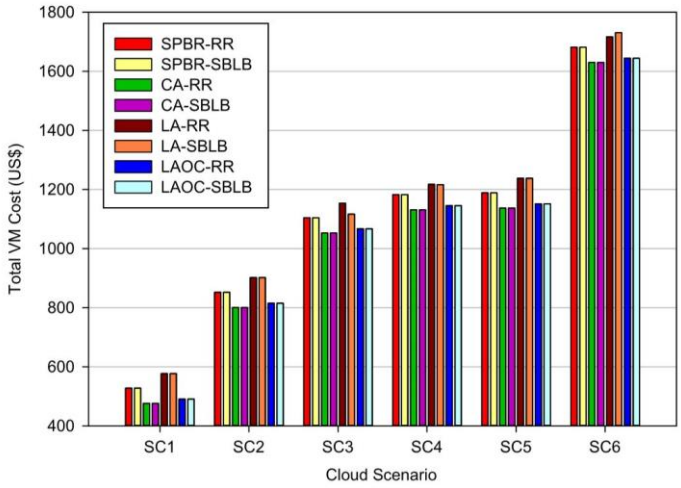
ما مجموع هزینه VM، متوسط زمان پاسخ دهی ، حداقل زمان پاسخ دهی ، متوسط زمان پردازش DC و حداکثر زمان پردازش DC را اندازه گیری کردیم. الگوریتم SBLB پیشنهادی ما در مقایسه با الگوریتم round robin عملکرد بسیار بهتری داشت. الگوریتم ما بدون افزایش در هزینه ، زمان پاسخ دهی و پردازش را بهبود بخشید. بنابراین ، استفاده از این الگوریتم از افزایش هزینه های اجرایی جلوگیری میکند. شکل7 مقایسه ای بین هزینه ی الگوریتم های جدید و قدیمی را نشان میدهد.

الگوریتم متعادل ساز بار SBLB زمان پاسخ دهی و زمان پردازش DC را بهبود میبخشد. در برخی موارد ، حداقل زمان پاسخ دهی در مقایسه با الگوریتم Round robin افقزایش داشته است. اما به طور کلی ، در الگوریتم بارگزاری متعادل پیشنهاد شده حداقل زمان پاسخ دهی بهبود یافته است. متوسط زمان پاسخ دهی و زمان پردازش DC به

طرز چشم گیری در این الگوریتم کاهش داشته اند. حداقل زمان پاسخ دهی و حداکثر زمان پردازش DC نیز با کمی بهبود همراه بوده اند. در بخش های بعدی نتایج آزمایش به طور جزئی توضیح داده شده اند.

7_2_1. متوسط زمان پاسخ دهی

الگوریتم بارگزاری متعادل SBLB متوسط زمان پاسخ دهی را در همه ترکیب های الگوریتم ها ، کاهش میدهد. با استفاده از این الگوریتم ، متوسط زمان پاسخ دهی 23٪ کاهش می یابد. الگوریتم بارگزاری متعادل SBLB همراه با الگوریتم کارگزاری با در نظر گرفتن هزینه ، متوسط زمان پاسخ دهی را حدود 37٪ کاهش دادند ، که در شکل8. (a) مشخص است. الگوریتم کارگزاری با در نظر گرفتن بار متوسط زمان پاسخ دهی را 8٪ کاهش داد و همچنین شبیه سازی ترکیب آن با سیاست کارگزاری SBPR متوسط زمان پاسخ دهی را 18٪ کمتر کرد. الگوریتم LAOC نیز متوسط زمان پاسخ دهی را با کاهش 29٪ مواجه کرد. الگوریتم های پیشنهادی را با الگوریتم های کارگزاری SPBR، LA ، CA و LAOC شبیه سازی کردیم. نتایج شبیه سازی نشان میدهند که الگوریتم SBLB متوسط زمان پاسخ دهی را در مقایسه با الگوریتم round robin بهبود بخشیده است.

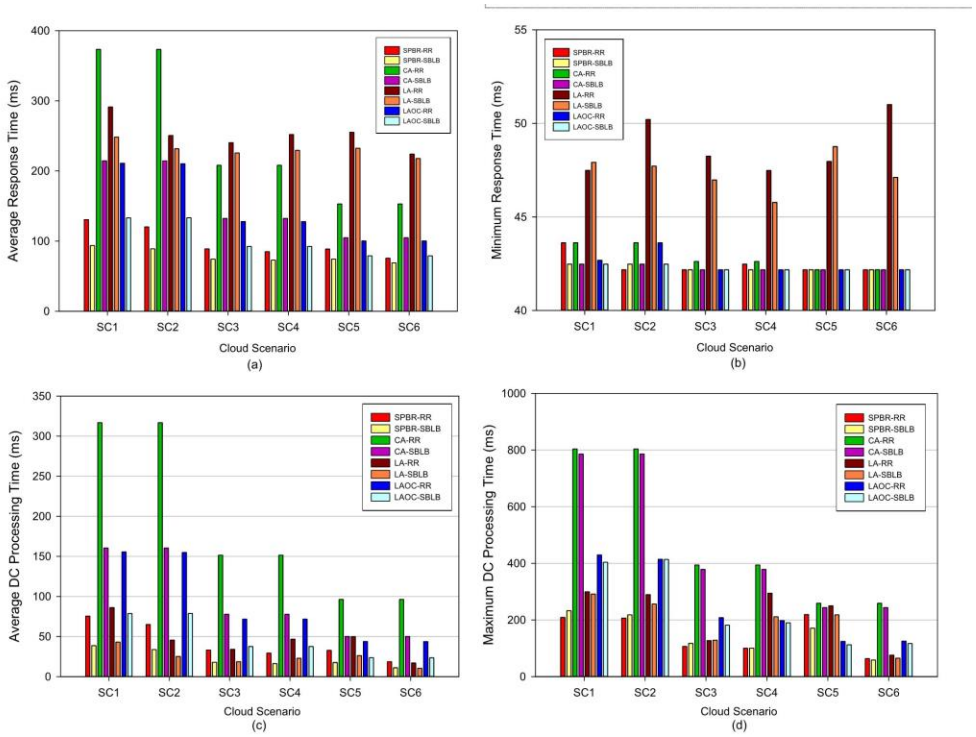


شکل7. مجموع هزینه ماشین مجازی در الگوریتم های بارگزاری متعادل

7_2_2. حداقل زمان پاسخ دهی

با استفاده از الگوریتم SBLB ، 1% بهبود در حداقل زمان پاسخ دهی مشاهده شد. برخی ترکیب های الگوریتم ها ، زمان پاسخ دهی مشابه یا کمی بیشتر داشتند. برای سناریو های 3 ، 5 و 6 الگوریتم های SPBR-SBLB و

SPBR-RR حداقل زمان پاسخ دهی مشابهی داشتند که در شکل8(b) مشخص است. با همین ترکیب الگوریتم ها ، حداقل زمان پاسخ دهی در سناریو 2 کمی افزایش داشت. حداقل زمان پاسخ دهی در سناریو های 1 و 4 با کاهش همراه بود. از طرف دیگر ، الگوریتم های CA-RR و CA-SBLB در سناریو های 5 و 6 نتایج مشابهی داشتند. حداقل زمان پاسخ دهی برای سایر سناریو های شبیه سازی کاهش داشت. این چهار سناریو در مقایسه با الگوریتم round robin با بهبود 1.8٪ همراه بود. به طور مشابه ، LAOC-RR و LAOC-SBLB در سناریو های 3،4،5 و 6 حداقل زمان پاسخ دهی یکسانی داشتند. با این حال در سناریو های 1 و 2در مقایسه با الگوریتم round robin با حداقل زمان پاسخ دهی کمتری همراه بودند. نتایج شبیه سازی برای LA-RR و LA-SBLB در سناریو های 1 و 5 نشاندهنده افزایش در حداقل زمان پاسخ دهی است. با این حال ، برای 4 سناریوی باقی مانده حداقل زمان پاسخ دهی با کاهش همراه بود. به طور میانگین 4.7٪ بهبود در این 4 سناریو مشاهده شد. بر اساس شکل8. (b) میتوان نتیجه گرفت که به استثنا برخی شرایط ، حداقل زمان پاسخ دهی با به کارگیری الگوریتم پیشنهادی این مقاله، بهبود می یابد.



شکل8. زمان پاسخ دهی و پردازش الگوریتم های بارگزاری متعادل

7_2_3. میانگین زمان پردازش DC

همانطور که در شکل8.(c) مشخص است ، نتایج نشان دهنده ی بهبود چشم گیر زمان پردازش DCاست. در مقایسه با الگوریتم round robin زمان پردازش DC با بهبود 47 درصدی مواجه شده است. به علاوه ، الگوریتم های SPBR-RR و SPBR-SBLB با بهبود 46 درصدی در زمان پردازش DC همراه بودند. با این حال ، برای سناریو 6 بهبود برابر با 41 درصد است و برای سایر سناریو ها نتایج مشابهی بدست آمد. به طور مشابه CA-RR و CA-SBLB با نتایج بسیار مشابهی همراه بودند. الگوریتم CA-SBLBدر مقایسه با CA-RR با کاهش 49 درصدی زمان پردازش DC همراه بود. نتایج شبیه سازی الگوریتم های LA-RR و LA-SBLB بهبود 39 تا 51 درصدی در زمان پردازش DC را نشان دادند. به طور کلی ، کاهش 46 درصدی در زمان پردازش DC برای الگوریتم های LA-RR و LA-SBLB مشاهده شد. از طرف دیگر ، الگوریتم های LOAC-RR و LOAC-SBLB نیز نتایج مشابهی داشتند. الگوریتم پیشنهادی ما دقیقا با 48٪ بهبود در زمان پردازش DC همراه بود. به طور کلی ، این نتایج نشان دهنده این است که الگوریتم بارگزاری متعادل SBLB سریع ترین پردازش ها را در مراکز داده ابری ارائه میدهد.

7_2_4. حداکثر زمان پردازش DC

به جز ترکیب الگوریتم های SPBR-RR و SPBR-SBLB ، سایر ترکیب ها با کاهش حداکثر زمان پردازش DC همراه بودند. شکل8 (d) نتایج اعمال ترکیب های مختلف الگوریتم ها را از نظر حداکثر زمان پردازش DC نشان میدهد. همانطور که در این شکل مشخص است. LA-RR و LA-SBLB کمترین میزان حداکثر زمان پردازش DC را نسبت به سه گروه دیگر نشان دادند. به طور متوسط حداکثر زمان پردازش DC 11% کاهش داشت. الگوریتم های SPBR-RR و SPBR-SBLB حداکثر زمان پردازش DC را در سناریو های 5 و 6 بهبود بخشیدند . با این حال ، در سناریو های 1 تا 4 این مقدار افزایش یافته بود.

نتایج شبیه سازی برای حداکثر زمان پردازش DC الگوریتم های CA-RR و CASBLB در شکل8(d) مشخص شده است. با توجه به این شکل ، متوسط بهبود 4 درصدی حداکثر زمان پردازش DC قابل مشاهده است. الگوریتم

های LOAC-RR و LOAC-SBLB حداکثر زمان پردازش DC را 7٪ بهبود بخشیدند. بیشترین میزان بهبود برای حداکثر زمان پردازش DC در سناریوی 3 مشاهده شد و مقدار بهبود برابر با 13٪ گزارش شده است و کمترین بهبود در سناریوی2 و برابر 0.24٪ است. به طور خلاصه ، این نتایج نشان دهنده این است که الگوریتم بارگزاری متعادل ارائه شده در این مقاله حداکثر زمان پردازش DC را کاهش داده است.

8.نوآوری های مهم

نوآوری این مقاله از دو جهت قابل بررسی است. اولین نوآوری ها ارائه الگوریتم های کارگزاری ابر با عناوین CA,LA و LAOC است. دومین نوآوری الگوریتم بارگزاری متعادل است که با عنوان SBLB اطلاق میشود. الگوریتم کارگزاری CAدر مقایسه با الگوریتم های موجود 5.5٪ در هزینه ها صرفه جویی میکند اما زمان پاسخ دهی و پردازش آن افزایش می یابد. از طرف دیگر ، الگوریتم LA با بهبود 37 درصدی در متوسط زمان پاسخ دهی و افزایش تعداد VMها و بهبود 73 درصدی در زمان پردازش DC همراه است. با این حال ، هزینه ها این الگوریتم در مقایسه با CA ، 11% بیشتر است. الگوریتم LAOC ترکیبی از دو الگوریتم ذکر شده است. هزینه و زمان پاسخ دهی و پردازش این الگوریتم بین مقادیر CA و LA است. همچنین الگوریتم بارگزاری متعادل SBLB زمان پردازش DC و زمان پاسخ دهی را بدون افزایش هزینه ، کاهش داد. به علاوه ، الگوریتم های کارگزاری پیشنهادی به کاربران ابر کمک میکنند تا ارائه دهندگان مورد نیاز خود را بر اساس محدودیت ها و صرفه جویی در هزینه انتخاب کنند. همچنین الگوریتم پیشنهاد شده میتواند به ارائه دهندگان خدمات برای مدیریت منابع کمک کند.

9. نتیجه

در سال های گذشته با پیشرفت های حاصل شده در زمینه رایانش ، تقاضا برای خدمات رایانش ابری نیز افزایش یافته است. در رایانش ابری ، کاربران میتوانند بر اساس نیاز های خود از منابع رایانشی استفاده کنند. رویکرد ابر به کاربران کمک میکند تا در هزینه های زیرساختIT صرفه جویی کنند. برای این منظور ، به دلیل نبود استانداردی خاص، ارائه دهندگان اینگونه خدمات ، پلتفرم رایانشی خود را مطابق میل خود میسازند. انتخاب یک ارائه دهنده

خدمات ابری مناسب از بین چندین ارائه دهنده امری چالش برانگیز برای کاربران است. یک کارگزار میتواند یک سرویس دهنده مناسب را بر اساس نیاز های کاربر و تحت توافق سطح خدمت به او معرفی کند. بارگزاری متعادل در ابر نیز یکی دیگر از موضوعات مورد تحقیق است. در رایانش ابری منابع رایانشی در صورت تقاضا در اختیار کاربر قرار میگیرند. ارائه دهندگان خدمات ابری باید تقاضای های زیادی را برای پاسخ دادن ، مدیریت کنند. تخصیص و مدیریت درخواست های کاربران برای استفاده از منابع فیزیکی امری بسیار چالش برانگیز است زیرا باید بار بین منابع سیستم به طور متعادل توزیع شود. بارگزاری متعادل به طور موثر میتواند هزینه های اجرایی را کاهش دهد و رضایت مشتریان را بهبود بخشد.

نتایج شبیه سازی نشان داد که برای صرفه جویی بیشتر در هزینه کارگزاری مستلزم زمان پاسخ دهی بیشتری است. با این حال ، سریع ترین پردازش نیز مستلزم هزینه های بیشتر است. بر اساس نیاز مشتریان ، کارگزار تصمیم میگیرد که فرآیند باید به صورت سریع و یا کم هزینه اجرا شود. نتایج این مطالعه نشان میدهد که یک تکنیک کارگزاری موثر میتواند در زمان و هزینه صرفه جویی کند. نتایج شبیه سازی عملکرد الگوریتم SBLB در مقایسه با سایر الگوریتم ها در محیط آزمایشی بسیار بهتر است. این مقاله نشان میدهد که الگوریتم بارگزاری متعادل میتواند زمان اجرا را برای کاربران و همچنین ارائه دهندگان کاهش دهد. کارگزار باید قابلیت مدیریت درخواست های متغیر کاربران و تخصیص منابع به صورت پویا بر اساس SLA را داشته باشد. حل کردن این مشکل به کارهای آینده موکول شد. یکی دیگر از محدودیت ها این است که آزمایش ها در محیط شبیه سازی شده صورت پذیرفته اند. به شدت توصیه میشود که کارگزاری ابری و بارگزاری متعادل در دنیای واقعی انجام شود.

References

Akhter, N., Othman, M., 2014. Energy Efficient Virtual Machine Provisioning in Cloud Data Centers. In: International Symposium on Telecommunication Technologies (ISTT), IEEE, Langkawi, Malaysia, pp. 282–286.

Akhter, N., Othman, M., 2016. Energy aware resource allocation of cloud data center: review and open issues. Clust. Comput., 1–20.
Bernstein, D., Vij, D., Diamond, S., 2011. An intercloud cloud computing economy-technology, governance, and market blueprints. In: 2011 Annual SRII Global Conference (SRII), IEEE, San Jose, CA, USA, pp. 293–299.
Bernstein, D., Ludvigson, E., Sankar, K., Diamond, S., Morrow, M., 2009. Blueprint for the intercloud-protocols and formats for cloud computing interoperability. In: 2009 ICIW'09 Fourth International Conference on Internet and Web Applications and Services, IEEE, Mestre, Italy, pp. 328–336.
Bernstein, D., Vij, D., 2010. Intercloud directory and exchange protocol detail using XMPP and RDF. In: 6th World Congress on Services (SERVICES-1), IEEE, Miami, Florida, USA, pp. 431–438.
Buyya, R., Yeo, C.S., Venugopal, S., Broberg, J., Brandic, I., 2009. Cloud computing and emerging IT platforms: vision, hype, and reality for delivering computing as the 5th utility. Future Gener. Comput. Syst. 25, 599–616.
Calheiros, R.N., Toosi, A.N., Vecchiola, C., Buyya, R., 2012. A coordinator for scaling elastic applications across multiple clouds. Future Gener. Comput. Syst. 28, 1350–1362.
Calheiros, R.N., Ranjan, R., Beloglazov, A., De Rose, C.A., Buyya, R., 2011. CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. Softw.: Pract. Exp. 41, 23–50.
Domanal, S.G., Reddy, G.R.M., 2014. Optimal load balancing in cloud computing by efficient utilization of virtual machines. In: 2014 Sixth International Conference on Communication Systems and Networks (COMSNETS), pp. 1–4.
Facebook User Statistics, 2014. ⟨http://newsroom.fb.com/Key-Facts⟩(accessed 06.12.13).
Fang, Y., Wang, F., Ge, J., 2010. A task scheduling algorithm based on load balancing in cloud computing. In: Web Information Systems and Mining, Springer, Sanya, China, pp. 271–277.
Ferreto, T.C., Netto, M.A., Calheiros, R.N., De Rose, C.A., 2011. Server consolidation with migration control for virtualized data centers. Future Gener. Comput. Syst. 27, 1027–1034.
Florence, A.P., Shanthi, V., 2014. A load balancing model using firefly algorithm in cloud computing. J. Comput. Sci. 10, 1156–1165.
Garg, S.K., Toosi, A.N., Gopalaiyengar, S.K., Buyya, R., 2014. SLA-based virtual machine management for heterogeneous workloads in a cloud datacenter. J. Netw. Comput. Appl. 45, 108–120.
Hu, J., Gu, J., Sun, G., Zhao, T., 2010. A scheduling strategy on load balancing of virtual machine resources in cloud computing environment. In: 2010 Third International Symposium on Parallel Architectures, Algorithms and Programming (PAAP), pp. 89–96.
Jrad, F., Tao, J., Streit, A., 2012. Simulation-based evaluation of an intercloud service broker. In: The Third International Conference on Cloud Computing, GRIDs, and Virtualization, Cloud Computing 2012, pp. 140–145.
Kessaci, Y., Melab, N., Talbi, E-G., 2013. A Pareto-based genetic algorithm for optimized assignment of vm requests on a cloud brokering environment. In: 2013 IEEE Congress on Evolutionary Computation (CEC), pp. 2496–2503.
Krishna, P.V., 2013. Honey bee behavior inspired load balancing of tasks in cloud computing environments. Appl. Soft Comput. 13, 2292–2303.
Lucas-Simarro, J.L., Moreno-Vozmediano, R., Montero, R.S., Llorente, I.M., 2013. Scheduling strategies for optimal service deployment across multiple clouds. Future Gener. Comput. Syst. 29, 1431–1441.
Maguluri, ST., Srikant, R., Ying, L., 2012. Stochastic models of load balancing and scheduling in cloud computing clusters. In: 2012 Proceedings IEEE INFOCOM, pp. 702–710.
Mahajan, K., Makroo, A., Dahiya, D., 2013. Round robin with server affinity: a VM load balancing algorithm for cloud based infrastructure. J. Inf. Process. Syst. 9, 379–394.
Naha, R.K., Othman, M., 2014. Brokering and load-balancing mechanism in the cloud-revisited. IETE Tech. Rev. 31, 271–276.
Naha, RK., Othman, M., 2014. Optimized Load Balancing for Efficient Resource Provisioning in the Cloud. In: International Symposium on Telecommunication Technologies (ISTT), IEEE, Langkawi, Malaysia, pp. 382–285.
Naha, R.K., Othman, M., Akhter, N., 2015. Evaluation of cloud brokering algorithms in cloud based data center. Far East J. Electron. Commun. 15, 85–98.
Quarati, A., Clematis, A., Galizia, A., DAgostino, D., 2013. Hybrid clouds brokering: business opportunities, QoS and energy-saving issues. Simul. Model. Pract. Theory 39, 121–134.
Randles, M., Lamb, D., Taleb-Bendiab, A., 2010. A comparative study into distributed load balancing algorithms for cloud computing. In: 2010 IEEE 24th International Conference on Advanced Information Networking and Applications Workshops (WAINA), pp. 551–556.
Rochwerger, B., Breitgand, D., Epstein, A., Hadas, D., Loy, I., Nagin, K., et al., 2011. Reservoir—when one cloud is not enough. IEEE Comput. 44, 44–51.
Sun, L., Dong, H., Hussain, F.K., Hussain, O.K., Chang, E., 2014. Cloud service selection: state-of-the-art and future research directions. J. Netw. Comput. Appl. 45, 134–150.
Tordsson, J., Montero, R.S., Moreno-Vozmediano, R., Llorente, I.M., 2012. Cloud brokering mechanisms for optimized placement of virtual machines across multiple providers. Future Gener. Comput. Syst. 28, 358–367.
Van den Bossche, R., Vanmechelen, K., Broeckhove, J., 2010. Cost-optimal scheduling in hybrid IaaS clouds for deadline constrained workloads. In: 2010 IEEE 3rd International Conference on Cloud Computing (CLOUD), pp. 228–235.
Vecchiola, C., Chu, X., Mattess, M., Buyya R, 2011. Aneka—integration of private and public clouds. In: Cloud Computing Principles and Paradigms, Wiley, Hoboken, pp. 251–274.
Wang, S-C., Yan, K-Q., Liao, W-P., Wang, S-S., 2010. Towards a load balancing in a three-level cloud computing network. In: 2010 3rd IEEE International Conference on Computer Science and Information Technology (ICCSIT), vol. 1, pp. 108–113.
Wickremasinghe, B., Calheiros, RN., Buyya, R., 2010. Cloudanalyst: a cloudsim-based visual modeller for analysing cloud computing environments and applications. In: 2010 24th IEEE International Conference on Advanced Information Networking and Applications (AINA), pp. 446–452.