2nd. International Conference on
Information Technology, Communications and Telecommunications (irICT2016)
1-2 March 2016 - Iran, Tehran

Auto-Scaling of Cloud Computing Resources Based on Clustering of Instantaneous Request Patterns of Services in form of Off-line Computations

Yaghoob Siahmargooei, M. Kazem Akbari, S. Alireza Hashemi Golpayegani, Saeed Sharifian

Department of Computer Engineering and Information Technology, Amirkabir University of Technology
E-mail: {y.s, akbarif, sa.hashemi}@aut.ac.ir
The Electrical and Electronics Engineering Department, Amirkabir University of Technology
E-mail: sharifian_s@aut.ac.ir

Abstract. Scalability and flexibility in the allocation of resources has a special place in cloud computing. In this paper, we tried to propose a new approach for auto-scaling of resources in cloud computing. Firstly, was defined the general framework of the proposed model by specifying the main assumptions of the issue as well as identifying influential factors. Then, we began to cluster the requests using the TPC-W Benchmark and producing raw data. Next, using genetic algorithm and Bin Packing Problem, we calculated the optimal layout of the resources. After that, we proceeded to the implementation of the proposed model and validation of the scaling results. Finally, by evaluating and comparing the proposed approach with the other methods, we indicated that the proposed model has a better performance in terms of time complexity, efficiency in layout of resources, and total cost.

Keywords: cloud computing, resource management, scalability, auto-scaling, resource allocation, genetic algorithm.

مقیاس‌دهی خودکار منابع رایانش ابری بر اساس خوشه بندی الگوهای تقاضای لحظه‌ای سرویس‌ها بصورت محاسبات برون خط

[۱] یعقوب سیاه‌مرگویی، [۲] محمدکاظم اکبری، [۳] سیدعلیرضا هاشمی گلپایگانی، [۴] سعید شریفیان

[۱] دانشکده مهندسی کامپیوتر و فناوری اطلاعات، دانشگاه صنعتی امیرکبیر (پلی تکنیک تهران)، y.s@aut.ac.ir
[۲] دانشکده مهندسی کامپیوتر و فناوری اطلاعات، دانشگاه صنعتی امیرکبیر (پلی تکنیک تهران)، akbarif@aut.ac.ir
[۲] دانشکده مهندسی کامپیوتر و فناوری اطلاعات، دانشگاه صنعتی امیرکبیر (پلی تکنیک تهران)، sa.hashemi@aut.ac.ir
[۴] دانشکده مهندسی برق، دانشگاه صنعتی امیرکبیر (پلی تکنیک تهران)، sharifian_s@aut.ac.ir

چکیده

مقیاس‌پذیری و انعطاف در تخصیص منابع، جایگاه ویژه‌ای در رایانش ابری دارد. در این مقاله سعی کردیم، یک رویکرد جدید برای مقیاس‌دهی خودکار منابع در رایانش ابری ارائه نماییم. ابتدا با مشخص کردن مفروضات اصلی مسئله و شناخت عوامل تاثیرگذار، چارچوب کلی مدل پیشنهادی را تعریف کردیم. سپس با استفاده از محک TPC-W و تولید داده‌های اولیه اقدام به خوشه‌بندی درخواست‌ها نمودیم. در ادامه با استفاده از الگوریتم ژنتیک و مسئله کلاسیک بسته‌بندی، چیدمان بهینه منابع را محاسبه نمودیم. پس از آن اقدام به اجرای مدل پیشنهادی و صحت سنجی نتایج مقیاس‌دهی کردیم. درنهایت با ارزیابی و مقایسه رویکرد پیشنهادی با دیگر روش‌ها، نشان دادیم مدل پیشنهادی از نظر پیچیدگی زمانی، بهینگی در چیدمان منابع و هزینه نهایی علمکرد بهتری دارد.

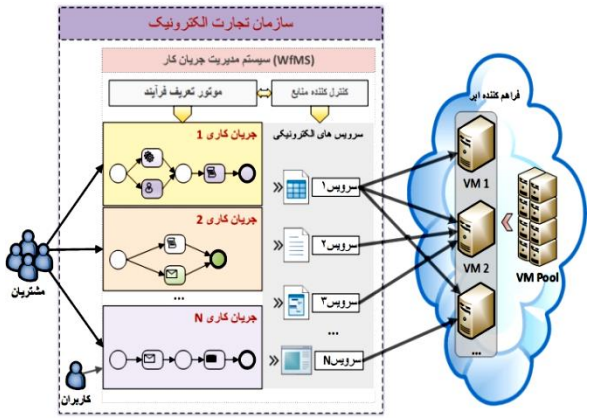
واژه های کلیدی : رایانش ابری، مدیریت منابع، مقیاس‌پذیری، مقیاس‌دهی خودکار، مقیاس‌دهی افقی، تخصیص منابع به سرویس، الگوریتم ژنتیک.

۱- مقدمه

امروزه در حوزه تجارت الکترونیک با افزایش مشتریان و به جهت کسب سود بیشتر و نیز افزایش سرعت ارائه کالا یا خدمت به مشتریان، نیاز به پردازش‌های بیشتر و سریع‌تر کاملاً مشهود است. رایانش ابری یکی از جدیدترین راه حل‌های ارائه شده برای پردازش‌های عظیم و دوام شرکت‌ها و سازمان‌ها در فضای رقابتی فناوری اطلاعات و تجارت الکترونیک است [1].

اکنون یک سازمان تجارت الکترونیک را در نظر بگیرید که بر اساس تحلیل‌های انجام شده و به دلیل روند رو به رشد مشتریان و نیازهای پردازشی خود و تلاقی بین نیازهای جدید مطرح شده در تجارت الکترونیک و نیز شعارها و دستاوردهای جدید رایانش ابری تصمیم به مهاجرت سرویس دهنده‌های خود به زیرساخت رایانش ابری داشته و در این راستا به‌منظور پاسخ به نیازهای پردازشی خود از فراهم کننده‌ی خدمات رایانش ابری، زیرساخت و منابع ابری اجاره کرده است. این سازمان با توجه به فرآیندها و جریان‌های کاری خود برای ارائه‌ی کالا و خدمات تجاری، نیاز به سرویس‌های مختلفی دارد که این سرویس‌ها به‌عنوان منابع اجرایی وظایف کاری در جریان‌های کاری سازمانی قرار دارند. حال مسئله این است که سازمان به چه میزان از منابع پردازشی و ماشین مجازی برای پاسخ به نیازهای پردازشی فرایندهای خود احتیاج دارد. سازمان از چه راهکاری استفاده کند تا بتواند با نزدیک کردن منابع پردازشی موجود خود با منابع پردازشی مورد نیازش هزینه‌ی اجاره‌ی منابع را کاهش دهد. بعلاوه امکان دارد، درخواست‌های سرویس سازمان دارای تغییرات شدیدی باشد که در این حالت کار بسیار پیچیده خواهد شد. چراکه با داشتن درخواست‌های متغیر، به منابع پردازشی متغیر در بازه‌های زمانی مختلف نیازمندیم. از جهتی دیگر شواهد حاکی از آن است که اکثر مشتریان خدمات رایانش ابری طبق اصل پارتو، در ۸۰٪ اوقات، تنها از ۲۰٪ منابع موجود خود استفاده می‌کنند که این مسئله نشان دهنده عدم توجه به مقدار منابع پردازشی مورد نیاز حتی در زمان اجاره خدمات رایانش ابری است [2].

همانطور که در شکل (۱) مشاهده می‌کنید، مسئله مورد بررسی دارای سه موجودیت اصلی یعنی سازمان تجارت الکترونیک، مشتریان آن سازمان و فراهم کننده خدمات رایانش ابری است. سازمان برای ارائه یک خدمت یا محصول، از طریق جریان کاری به مشتریان خود خدمات الکترونیکی ارائه می‌کند. این سازمان برای انجام پردازش‌های الکترونیکی خود از فراهم کننده خدمات رایانش ابری، زیرساخت رایانش ابری اجاره کرده است. زیرساخت اجاره شده شامل یک یا چند ماشین‌مجازی است که به‌عنوان عناصر اصلی پردازشی محسوب می‌شوند. هریک از ماشین‌های مجازی دارای منابع پردازشی مشخصی هستند. تعداد درخواست‌های ورودی به ماشین‌های مجازی و سرویس‌ها متغیر است و به ترافیک ورودی بستگی دارد. درخواست‌های مختلف از یک سرویس الکترونیکی مشخص، تقریباً به مقدار ثابتی از منابع پردازشی نیاز دارند. هریک از سرویس‌ها روی هریک از ماشین‌های مجازی، نصب و قابل اجرا می‌باشند. تخصیص سرویس‌ها به ماشین‌های مجازی روشن، با دو وضعیت تخصیص یا عدم تخصیص صورت می‌گیرد. تخمین اولیه‌ای از مدت زمان اجرای درخواست‌های هر سرویس در ماشین‌های مجازی وجود دارد. هزینه‌ی اجاره ماشین‌های مجازی به دو صورت پیش پرداخت و پرداخت لحظه‌ای است. زمان روشن و آماده به سرویس شدن ماشین‌های مجازی دارای توزیع آماری مشخص و بین ۲ تا ۵ دقیقه است.



شکل (۱) : نمای کلی مفروضات مسئله و مولفه‌های اصلی آن

۲- تحقیقات مرتبط

مقیاس‌دهی[1] از خصوصیات قابل توجه در رایانش ابری است. از این رو، مدل‌ها و الگوریتم‌های فراوانی برای مقیاس دهی منابع، زیرساخت و ماشین‌های مجازی در محیط ابر پیشنهاد شده است. هر یک از آن‌ها تلاش کرده‌اند الگوریتم‌های بهینه و کارایی ارائه دهند تا قابلیت‌های مقیاس پذیری در ابر را بهبود ببخشند. اگر بخواهیم دسته‌بندی کاملی از کارهای انجام‌شده در این حوزه داشته باشیم، روش‌های پیشنهادی را می‌توان به پنج دسته اصلی تقسیم کرد [3]:

۱) *سیاست‌های مبتنی بر آستانه به‌صورت ایستایی*؛ ۲) *یادگیری تقویتی*؛ ۳) *تئوری صف*؛ ۴) *تئوری کنترل*؛ ۵) *تحلیل دوره‌های زمانی*؛

دسته اول براساس آستانه‌های مشخص بصورت قانون محور عمل می‌کند[4]. اینگونه رویکردها در حال حاضر توسط بسیاری از فراهم کنندگان ابر مانند Amazon و RightScale مورد استفاده قرار می‌گیرند. دسته دوم، در واقع از یک نوع تصمیم گیری خودکار که منجر به مقیاس‌دهی خودکار می‌شود استفاده می‌کنند. بطوریکه بدون هیچ اولویت دانش، کارایی برنامه و سیاست‌های مورد استفاده را بررسی می‌کند. در روش‌های مبتنی بر تئوری صف، با استفاده از مسئله کلاسیک ریاضیات یعنی «صف» در تلاش برای تخمین سنجه‌های کارایی منابع مانند طول صف و زمان انتظار درخواست‌ها و استفاده از

[1] Scaling

صف‌های مختلف برای حل بهتر مسئله است. در روش‌های مورد استفاده از تئوری کنترل، تلاش شده است تا با استفاده از کنترل‌های مختلف صورت گرفته روی بخش‌های اصلی منابع ابری، مشکلات پیش آمده را با استفاده از روش‌های پیشگیرانه و یا فعال حل نماید. به‌عنوان آخرین دسته، تحلیل دوره‌های زمانی در حوزه‌های مختلفی مانند اقتصاد، مهندسی و بایوانفرماتیک مورد استفاده قرار می‌گیرد که سعی کرده است با استفاده از تغییرات صورت گرفته در بازه‌های زمانی و ترتیب‌های زمانی مختلف راه‌حل‌هایی جهت بهبود ارائه نماید. عیوب اصلی کارهای انجام شده در این حوزه مواردی چون، عدم توجه به فرآیندها و جریان‌های کاری سازمان و همچنین عدم پیش بینی وضعیت آینده درخواست‌ها، ناتوانی در پاسخگویی به تغییرات شدید درخواست‌ها، نیاز به مرتبه اجرایی بالا و سربار پردازشی فراوان، عدم تطابق با نیازهای تجارت الکترونیک مانند رضایت مشتری و زمان پاسخ کوتاه است. در جدول (۱) آخرین تحقیقات انجام شده در این زمینه را بصورت مقایسه ویژگی‌ها در کنار مدل پیشنهادی مشاهده می‌کنید.

جدول (۱) : مقایسه مدل پیشنهادی با روش‌های مشابه

منبع	دسته	داده واقعی	محیط اجرا واقعی	کاهش هزینه	تجارت الکترونیکی	پیچیدگی زمانی	کیفیت سرویس
[5]	ایستا	×	×	×	√	کم	√
[6]	ایستا	–	–	√	√	کم	√
[7]	تقویتی	×	×	√	×	زیاد	√
[8]	صف	√	√	√	×	کم	×
[9]	صف	×	√	×	√	زیاد	×
[10]	کنترل	√	×	×	×	زیاد	×
[11]	کنترل	√	√	√	×	زیاد	×
[12]	زمانی	√	–	×	×	-	×
[13]	زمانی	√	√	×	×	کم	×
مدل پیشنهادی		×	√	√	√	کم	√

۳- مدل پیشنهادی

اگر در طول زمان به وضعیت درخواست‌های سرویس در جریان‌های کاری توجه کنیم، این مطلب روشن است که درخواست‌ها دارای الگوهای خاصی خواهند بود. این نظم در اثر ثابت بودن احتمال انتقال سرویس‌ها پس از یکدیگر و قطعیت در نظر گرفته شده در روند حرکت درخواست‌ها در جریان‌های کاری ایجاد شده است. پس آنچه آشکار است، وجود یک نظم و الگوی خاص در تاریخچه درخواست سرویس‌ها در صورت ثابت بودن تعداد جریان‌های کاری، سرویس‌های الکترونیکی و احتمال اجرای سرویس‌ها و همچنین قطعیت در جریان‌های کاری سازمان است. چه بسا ممکن است تعداد درخواست سرویس‌ها در یک پنجره زمانی مشخص دقیقاً با تعداد درخواست سرویس‌ها در پنجره زمانی دیگر، مثلاً در گذشته یا آینده یکسان باشد. با این کار می‌توانیم مشخص کنیم درخواست‌های سرویس‌ها از چه شباهتی تبعیت می‌کنند و اینکه آیا می‌توانیم به تعداد دسته‌های مشخصی برسیم تا آن‌ها را درخواست پرتکرار بنامیم؟ اما نکته دیگر این است که ما چه نیازی به دسته‌بندی درخواست‌های سرویس‌ها در بازه‌های زمانی مختلف داریم؟ پاسخ این است که با در نظر گرفتن این حقیقت که هریک از سرویس‌ها منابع پردازشی مشخصی را مصرف می‌کنند، می‌توان نتیجه گرفت که مقدار منابع مورد نیاز برای پاسخ به تمام درخواست‌های سرویس در یک پنجره زمانی خاص با مقدار منابع مورد نیاز برای پاسخ به درخواست‌های سرویس در یک پنجره زمانی دیگر که در زمان گذشته یا آینده اتفاق می‌افتد درصورت مشابهت، یکسان خواهد بود. در همین راستا می‌توان نتیجه گرفت که تعداد ماشین‌های مجازی مورد نیاز در این دو پنجره زمانی مشابه، یکسان خواهد بود. در این صورت می‌توان با انجام عمل خوشه بندی روی درخواست‌های سرویس‌ها، درخواست‌های پرتکرار و در نتیجه‌ی آن مقدار منابع پردازشی و تعداد ماشین‌های مجازی آن درخواست‌ها را مشخص کرد.

رویکرد پیشنهادی برای حل این مسئله به این صورت است که اگر ما بتوانیم منابع پردازشی مورد نیاز که در اغلب زمان‌ها اتفاق می‌افتد را محاسبه نماییم، می‌توانیم در هر لحظه مشخص کنیم که درخواست سرویس فعلی با کدام‌یک از حالات پر تکرار شباهت دارد، تا به فراهم کننده خدمات رایانش ابری اعلام کنیم که در حال حاضر این تعداد از ماشین‌های مجازی و منابع پردازشی مورد نیاز است. اگر بتوانیم مقدار منابع مورد نیاز برای درخواست‌ها را در بدترین حالت ممکن در نظر بگیریم، آنگاه تفاوت‌های اندک میان درخواست واصله جدید با درخواست‌های پرتکرار قابل چشم پوشی خواهد بود. یکی دیگر از مسائل مهم این است که محاسبه‌ی مقدار منابع پردازشی مورد نیاز در زمان واقعی امری دشوار و غیرقابل انجام است. چراکه نمی‌توان دقیقاً در لحظه‌ای که درخواست‌های سرویس رسیده‌اند و آماده اجرا هستند، بدون هیچ تأخیری مشخص کنیم که مقدار منابع پردازشی مورد نیاز برای اجرای سرویس‌ها چه مقدار هستند. این کار مستلزم صرف زمان و انجام محاسبات دقیق فراوانی است که از حالت زمان واقعی خارج است. خود این مسئله نیز توجیهی برای انجام عمل خوشه بندی خواهد بود، چراکه با انجام خوشه بندی می‌توان محاسبات لازم را در زمان دیگری به انجام رساند و با ذخیره کردن نتایج در آینده از آن‌ها استفاده نمود [14].

پس به‌طور خلاصه رویکرد پیشنهادی این است که با خوشه بندی درخواست‌های سرویس‌ها در پنجره‌های زمانی، برای درخواست‌های پرتکرار مقدار منابع پردازشی مورد نیاز محاسبه گردد و بصورت یک جدول مراجعه و جستجو ذخیره سازی شود. حال در هر لحظه درخواست سرویس فعلی را با درخواست‌های پرتکرار بصورت برخط و با محاسبات ساده‌تر مقایسه نماییم. درخواست پرتکراری که بیشترین شباهت را به درخواست فعلی دارد انتخاب و مقدار منابع پردازشی مورد

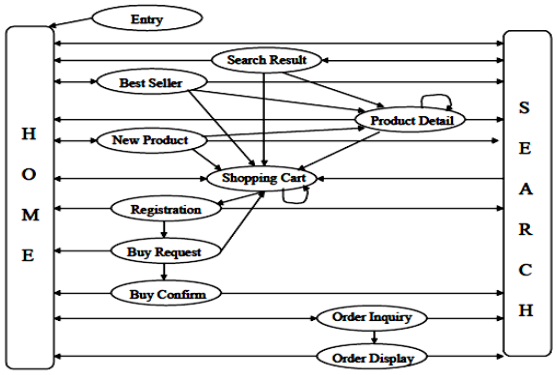
نیاز آن که قبلاً محاسبه شده است، انتخاب و اعلام گردد. حال باید به یک نکته اساسی توجه کرد، در صورتی می‌توانیم خوشه‌بندی مناسب و الگوهای پرتکرار خوبی داشته باشیم که الگوهایی که به‌عنوان تاریخچه برای خوشه بندی استفاده می‌کنیم، تمام فضای حالت ممکن را پوشش دهد و دارای تعدد زیادی باشد. چراکه هرچقدر تعداد الگوهای درخواست بیشتر باشد احتمال اینکه پوشش بهتری در فضای حالت داشته باشیم بیشتر خواهد بود. فضای حالتی که از آن صحبت می‌شود همان درخواست‌های متفاوت است که ممکن است توسط کاربران در زمانی خاص اتفاق بیفتد. مراحل درنظر گرفته شده برای حل این مسئله به این صورت بیان می‌شوند: (۱) استخراج درخواست‌های سرویس‌ها به‌عنوان اولین ورودی مدل؛ (۲) خوشه بندی درخواست‌های سرویس‌ها برای یافتن درخواست‌های پرتکرار؛ (۳) محاسبه منابع پردازشی مورد نیاز برای درخواست‌های پر تکرار به‌صورت برون خط؛ (۴) ذخیره سازی درخواست‌های پرتکرار و منابع مورد نیاز هریک به‌منظور رجوع به آن‌ها؛ (۵) مقایسه برخط برای یافتن بهترین درخواست پرتکرار مشابه با درخواست رسیده فعلی؛ (۶) بروز رسانی تدریجی درخواست‌های پرتکرار و خوشه بندی دوره‌ای آن‌ها برای افزایش دقت مدل؛

در شکل (۲) چارچوب کلی مدل پیشنهادی به‌همراه مؤلفه‌های اصلی و نحوه ارتباط هریک و تربیت اجرای هریک از مراحل قابل مشاهده است.

شکل (۲) : چارچوب کلی مدل پیشنهادی برای مقیاس‌دهی خودکار

۴- پیاده سازی و اجرا

بمنظور آغاز فرایند اجرا و در اختیار داشتن داده‌های اولیه ورودی به مدل، با کمک ابزارهای تولید درخواست سرویس کاربران و روش طراحی آزمایش‌ها، داده‌هایی نزدیک به داده‌های درخواست واقعی کاربران را تهیه کردیم. از میان ابزارهای مطرح در این زمینه، از محک[۲] TPC-W به سبب پشتیبانی از درخواست‌های تجارت الکترونیک و دارا بودن جریان کاری مشخص به همراه احتمال انتقال معین میان سرویس‌ها و همچنین بسترهای پیاده سازی مناسب، استفاده کردیم. همانطور که در شکل (۳) مشاهده می‌کنید، جریان کاری محک از ۱۴ سرویس اصلی تشکیل شده است. جهت اجرای محک از یک ماشین مجازی با سیستم عامل Linux استفاده شد که دارای قدرت پردازشی 3×2.5GHz و مقدار 3.5 گیگابایت حافظه اصلی می‌باشد. این محک با زبان java و به‌صورت Servlet روی سرویس دهنده Tomcat نصب گردیده است. پایگاه داده مورد استفاده در این محک MySQL می‌باشد.

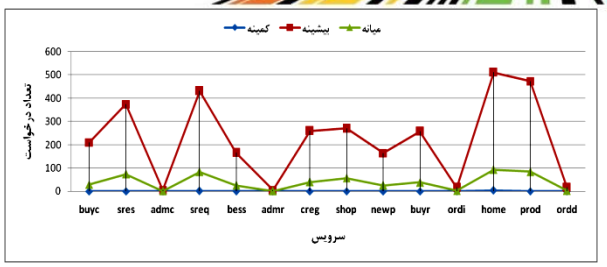


شکل (۳) : نمودار جریان کاری بستر آزمایشی محک TPC-W

اکنون نیاز به تولید درخواست اجرای هریک از این سرویس‌ها در بازه‌های زمانی مشخص داریم. برای رسیدن به تنوع درخواست‌های سرویس مورد نظر و همچنین شبیه سازی شرایط بحرانی مانند مواقعی که درخواست‌های انبوهی در یک پنجره زمانی ایجاد می‌شود، با کمک روش طراحی آزمایش‌ها از ۲۸ تکرار[۳] متفاوت استفاده نمودیم. هر یک از این تکرارها دارای مشخصه‌های متفاوتی هستند. این مشخصه‌ها از موارد تعیین کننده در نحوه عملکرد این محک هستند، چرا که تنوع درخواست‌ها را تضمین می‌کنند. بر اساس این مشخصه‌ها، کل زمان اجرای شبیه سازی ۸ ساعت به طول انجامید. لازم به ذکر است که طول پنجره زمانی مورد استفاده در این مقاله مقدار یک دقیقه در نظر گرفته شده است. خروجی حاصل از محک، فایل ثبت رویداد سرویس دهنده Tomcat می‌باشد که در آن هریک از درخواست‌های سرویس توسط کاربران مختلف که در واقع مرورگرهای شبیه سازی شده هستند و صفحات درخواستی آن‌ها در محک را ثبت می‌کند. در نهایت خروجی حاصل پس از تمیز شدن، تعداد ۵۰۰ بردار از مقدار درخواست سرویس‌ها در پنجره‌های زمانی مختلف خواهد بود. همانطور که در شکل (۴) مقادیر کمینه، میانگین و بیشینه برای تعداد درخواست‌های هریک از سرویس‌ها نمایش داده شده است. اختلاف میان تعداد درخواست‌های سرویس‌های مختلف به دلیل احتمال اجرای یک سرویس بر اساس جریان کاری می‌باشد.

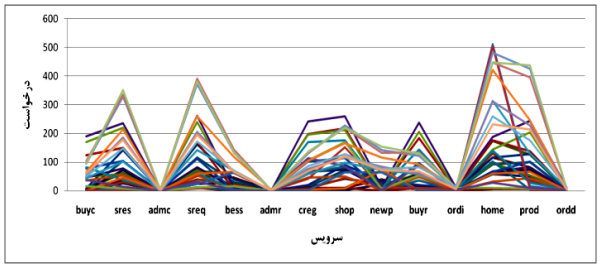
[2] Benchmark

[3] Iteration



شکل (۴) : نتایج اجرای محک TPC-W برای جمع آوری داده اولیه

همان‌طور که در راه حل پیشنهادی اشاره کردیم، به دلیل شباهت بسیاری از بردارهای درخواست سرویس‌ها با یکدیگر می‌توان از طریق عمل خوشه بندی، بردارهای پرتکرار را شناسایی نمود. در اینجا با استفاده از الگوریتم خوشه بندی K-Means و معیار شباهت اقلیدسی با کمک ابزار آماری Xlstat اقدام به خوشه بندی ۵۰۰ بردار تاریخچه درخواست نمودیم، با توجه به مشخص نبودن تعداد خوشه‌های بهینه با استفاده از تکنیک سطح برش[۴] و در نظر گرفتن نرخ همگرایی[۵] ۰,۵ توانستیم داده‌ها را در ۳۰ خوشه قرار دهیم. مراکز خوشه‌ها بعنوان بردارهای پرتکرار بصورت یکجا در شکل (۵) قابل مشاهده است. همانطور که مشخص است، این ۳۰ بردار گستره کاملی از ۵۰۰ بردار تاریخچه اولیه را پوشش داده‌اند.



شکل (۵) : بردارهای پرتکرار حاصل خوشه بندی درخواست‌ها

در مرحله بعد نیاز به یافتن منابع مورد نیاز برای درخواست‌های پرتکرار و تعداد ماشین‌های مجازی مورد نیاز آن‌ها داریم. ورودی این مرحله بردارهای درخواست پرتکرار به‌صورت مجزا هستند. هر بردار درخواست، متشکل از ۱۴ کمیت متفاوت است که این اعداد نشان دهنده‌ی تعداد درخواست‌های هریک از سرویس‌های چهارده‌گانه محک TPC-W در یک پنجره زمانی مشخص است. ورودی دیگر این مرحله، مشخصات ماشین‌های مجازی در دسترس از طرف فراهم کننده خدمات رایانش ابری است. جدول (۲) سه نوع ماشین مجازی با مشخصات سخت افزاری متفاوت، مطابق با نمونه‌های ماشین مجازی شرکت Amazon قابل مشاهده است.

جدول (۲) : نمونه ماشین‌های مجازی مدل بر اساس Amazon EC2

شناسه	ماشین مجازی	CPU (MIPS)	RAM (Mb)	هزینه اجاره بر ساعت
۱	t1.micro	1.7	660	$0.020
۲	c1.medium	7	1825	$0.145
۳	c3.large	9	3700	$0.150

جهت اجرای این مرحله از الگوریتم ژنتیک به‌وسیله کدهای نوشته شده در برنامه متلب استفاده شده است. مشخصه‌های مورد استفاده شامل ۵۰ تکرار برای تولید ۱۵۰۰ نسل با اندازه جمعیت ۱۰۰ که از نرخ جهش ۰,۵ و نرخ تقاطع ۰,۹ و نرخ همگرایی ۰,۰۱ بهره گرفته است. مقادیر تعیین شده برای مشخصه‌ها بر اساس روش تاگوچی تنظیم و انتخاب شده است. تابع برازش مورد استفاده نیز به‌صورت (۱) قابل مشاهده‌است. این مقادیر پس از محاسبه، نرمال گشته و در الگوریتم استفاده می‌شود. مشخصه‌های مورد استفاده برای ایجاد تابع برازش به همراه توضیحات در جدول (۳) آورده شده‌است.

$$Fitness\ \ Function = F_1 + F_2 + F_3 + F_4 + F_5 + F_6 \qquad (۱)$$

الگوریتم ژنتیک با استفاده از مسئله کلاسیک بسته‌بندی[۶] (تعمیم یافته مسئله کوله‌پشتی) به روش جستجوی فرا ابتکاری اقدام به یافتن تعداد بهینه ماشین‌های مجازی مورد نیاز، نوع هریک و همچنین سرویس‌های قابل اجرا روی ماشین‌ها می‌نماید. پس از آن نتایج به‌همراه بردارهای پرتکرار متناظر، در جدولی تحت عنوان جدول مراجعه و جستجو ذخیره خواهد شد.

جدول (۳) : مشخصه‌های تابع برازش در الگوریتم ژنتیک

مشخصه	توضیح	تأثیر
F_1	مقدار منابع در دسترس و آزاد هر یک از ماشین‌های مجازی	مثبت
F_2	نسبت کل منابع مورد نیاز سرویس‌ها به تعداد ماشین‌های مجازی	مثبت
F_3	تعداد سرویس‌های تخصیص داده شده به ماشین‌های مجازی	مثبت
F_4	انحراف از معیار مقدار مصرف ماشین‌های مجازی از منبع پردازنده	منفی
F_5	انحراف از معیار مقدار مصرف ماشین‌های مجازی از منبع حافظه	منفی
F_6	اختلاف میزان درصد مصرفی از پردازنده به درصد مصرفی از حافظه	مثبت

۴-۱ بررسی صحت عملکرد مدل پیشنهادی

اکنون زمان صحت سنجی مدل پیشنهادی فرا رسیده است. همانطور که قبلا اشاره کردیم، با در اختیار داشتن جدول مراجعه و جستجو که شامل بردارهای درخواست پرتکرار و پیکره بندی منابع متناظر با آن است، می‌توانیم با انجام عمل مقایسه، منابع مورد نیاز درخواستهای جدید را محاسبه و عمل مقیاس‌دهی خودکار را بصورت کامل به انجام رسانیم. در اینجا با استفاده از کلاس‌بندی و انتخاب یک معیار شباهت مناسب این‌کار صورت گرفته است. به جهت مقایسه بردار درخواست جدید با بردارهای نماینده موجود در جدول مراجعه و جستجو، نیازمند استفاده از یک روش مناسب و نظام‌مند هستیم. بمنظور بالابردن دقت خروجی مرحله کلاس بندی، از معیار شباهت ترکیبی متشکل از فاصله

[۴] Truncation Level

[۵] Convergence Rate

[۶] Bin Packing Problem

اقلیدسی و ضریب همبستگی[7] استفاده شده است. همان‌طور که در (۵) مشاهده می‌کنید فاصله اقلیدسی بین دو مجموعه X و Y به‌صورت d(X,Y) تعریف می‌شود. مقدار ضریب همبستگی برای دو مجموعه X و Y نیز به‌صورت (۴) تعریف می‌شود. در این رابطه x وy اعضای مجموعه مورد بررسی، $\bar{x}$ و $\bar{y}$ میانگین مجموعه‌های X و Y هستند. نحوه محاسبه این مقدار در (۲) و (۳) نشان داده شده است. مقادیر حداکثری برای این معیار نشان‌دهنده مقادیر بهتر قابل قبول و بهتری هستند. در اینجا V بردار ورودی و LKV بردارهای نماینده موجود در جدول مراجعه و جستجو هستند.

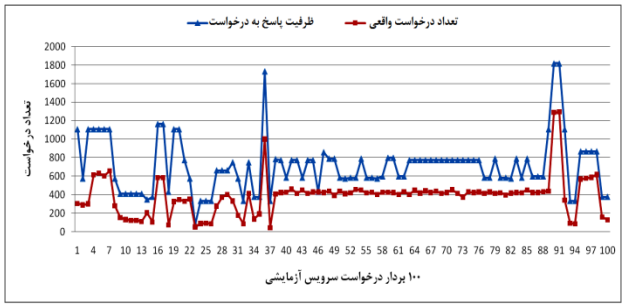
$$Max(SM_{lkv,v}) \quad (۲)$$

$$SM_{lkv,v} = Corr(V_t, LKV_k) + d(V_t, LKV_k) \quad (۳)$$

$$Corr(X,Y) = \frac{\sum_{i=1}^{n}((x_i - \bar{x})(y_i - \bar{y}))}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2 \sum_{i=1}^{n}(y_i - \bar{y})^2}} \quad (۴)$$

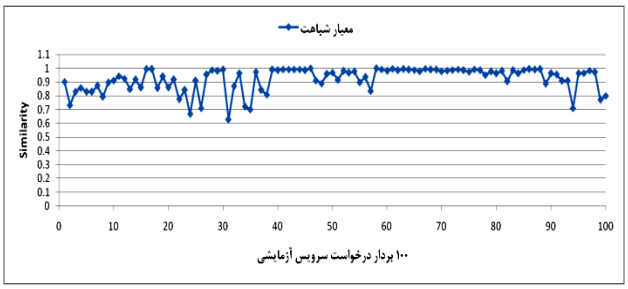
$$d(X,Y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \quad (۵)$$

حال برای بررسی صحت عملکرد مدل برای کلاس بندی و انتخاب بهترین بردار از جدول مراجعه و جستجو، با کمک محک TPC-W تعداد ۱۰۰ بردار درخواست جدید تولید نمودیم. این بردارها در واقع تعداد درخواست سرویس‌ها در پنجره‌های زمانی مختلف است. بمنظور اعتبار بخشیدن به این آزمون، پارامترهای متفاوت و جدیدی به شبیه ساز ترافیک در محک اختصاص دادیم تا الگوهای تقاضای تولیدی تفاوت چشمگیری با داده‌های اولیه ورودی مدل داشته باشند. فرآیند کار به این شکل است که هریک از بردارهای آزمون جدید به مؤلفه کلاس‌بندی الگو فرستاده می‌شود. این مؤلفه با بررسی میزان شباهت ($SM_{lkv,v}$) الگوی ورودی با بردارهای موجود در جدول مراجعه و جستجو شبیه‌ترین بردار نماینده را از جدول یافته و متناظر با آن منابع مورد نیاز و تعداد ماشین‌های مجازی و چیدمان سرویس‌ها را اعلام می‌کند. در شکل (۶) نسبت تعداد درخواست هریک از بردارهای آزمایشی به ظرفیت پاسخگویی منابع مؤلفه کلاس‌بندی آورده شده است. همانطور که مشاهده می‌کنید، برای اکثر بردارها منابع پیشنهادی قادر به پاسخگویی تعداد درخواست بیشتری از درخواست‌های ورودی هستند.



شکل (۶) : نسبت تعداد درخواست ورودی به ظرفیت پاسخگویی منابع

همانطور که روشن است بیشتر بودن ظرفیت منابع ماشین‌های مجازی برای پاسخ به درخواست‌ها در پنجره زمانی، باعث عدم بروز تاخیر یا خطا در اجرای سرویس‌ها خواهد شد. همچنین با بررسی مقادیر معیار شباهت برای بردارهای جدید آزمایشی و بردارهای نماینده انتخابی از جدول مراجعه و جستجو می‌توان بر صحت عملکرد مؤلفه کلاس بندی و به تبع آن عملکرد دقیق مدل مقیاس دهی صحه گذاشت. مقادیر نرمال شده (بین صفر و یک) معیار شباهت ($SM_{lkv,v}$) در شکل (۷) قابل مشاهده است.



شکل (۷) : مقادیر معیار شباهت برای بردارهای درخواست آزمایشی

۵- مقایسه و ارزیابی

در این بخش بمنظور ارزیابی مدل پیشنهادی و مقایسه آن با دیگر روش‌ها اقدام به پیاده سازی سه الگوریتم حریصانه[8] FF، BF و DFF نمودیم. هریک از این الگوریتم‌ها تعداد بهینه ماشین‌های مجازی و نیز چیدمان مناسب سرویس‌ها روی آن‌ها را برای ۳۰ بردار نماینده حاصل از خوشه‌بندی را محاسبه می‌کنند. پس از آن با مقایسه نتایج الگوریتم‌های حریصانه با مدل پیشنهادی از سه منظر پیچیدگی زمانی، تخصیص بهینه سرویس‌ها روی ماشین‌ها و هزینه اجاره ماشین‌های مجازی تحلیل می‌کنیم. هدف اثبات توانایی مدل در مقیاس‌دهی منابع موردنیاز سازمان در پنجره‌های زمانی مختلف است. درجدول (۴) مرتبه اجرایی الگوریتم‌های مختلف آورده شده است، همانطور که می‌دانیم پیچیدگی زمانی اجرای مدل پیشنهادی به اندازه جدول مراجعه و جستجو (تعداد سرویس‌ها× تعداد خوشه‌ها) وابسته است.

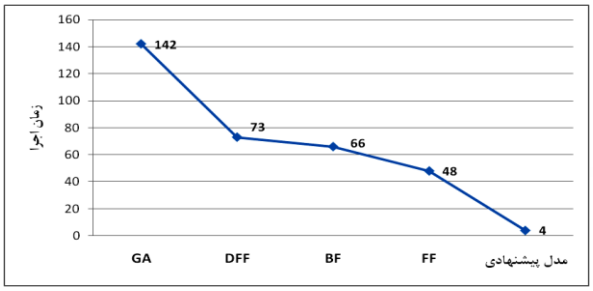
جدول (۴):مقایسه پیچیدگی زمانی مدل پیشنهادی با دیگر الگوریتم‌ها

[7] Correlation

[8] Greedy Algorithms (First Fit, Best Fit,…)

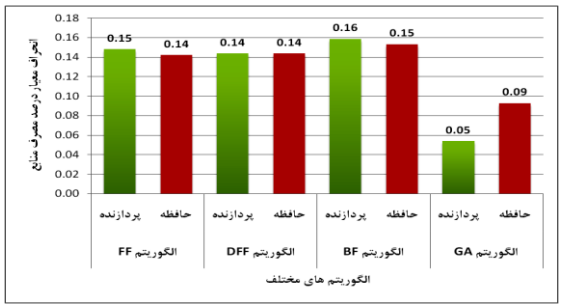
الگوریتم	پیچیدگی زمانی
FF	$O(N^2 logN)$
BF	$O(N^2 logN)$
DFF	$O(N^3)$
مدل پیشنهادی	O(k.s)

همانطور که در شکل (۸) مشاهده می‌کنید، زمان اجرای مدل پیشنهادی نیز در مقایسه با الگوریتم‌های حریصانه بسیار کمتر است. دلیل این مسئله، شکستن پیچیدگی مسئله به دو بخش برون خط و برخط است. به این صورت که محاسبات پیچیده در زمانی مناسب بصورت برون خط انجام شده‌اند.



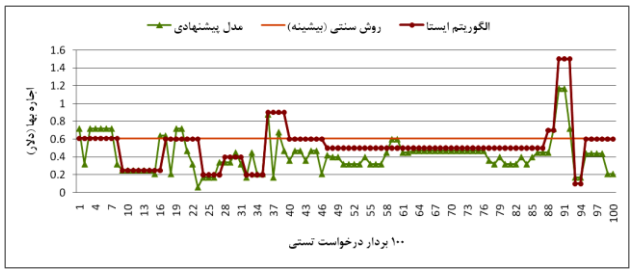
شکل (۸) : مقایسه زمان اجرای مدل پیشنهادی با دیگر الگوریتم‌ها

همچنین با بررسی انحراف معیار درصد مصرف منابع در ماشین‌های مجازی در الگوریتم‌های مورد بررسی مشخص شد، الگوریتم ژنتیک استفاده شده در مدل پیشنهادی بیشترین تعادل را در مصرف منابع ماشین‌های مجازی ایجاد کرده است. نتایج این مقایسه در شکل (۹) قابل مشاهده است.



شکل (۹) : مقایسه انحراف معیار درصد مصرف منابع در الگوریتم‌ها

در نهایت مدل مقیاس‌دهی پیشنهادی را با رویکرد مقیاس‌دهی در فراهم کنندگان مشهور مانند Amazon که از الگوریتم ایستا استفاده می‌کنند، مقایسه کردیم. همانطور که در شکل (۱۰) مشاهده می‌کنید میزان هزینه پرداختی برای مشتری خدمات زیر رایانش ابری برای هریک از ۱۰۰ بردار آزمایشی برای سه رویکرد الگوریتم ایستا، روش سنتی و مدل پیشنهادی با یکدیگر مقایسه گردیده‌اند. منظور از روش سنتی، درنظر گرفتن بدترین شرایط و تخصیص بیشترین منابع به درخواست‌ها می‌باشد. همانطور که واضح است روش سنتی در مقابل روش ایستا و مدل پیشنهادی نامناسب ترین روش محسوب می‌شود. هرچند الگوریتم ایستا منجر به صرفه جویی در هزینه اجاره منابع شده است، اما مدل پیشنهادی نتایج بسیار بهترین را ارائه کرده‌است.



شکل (۱۰) : مقایسه هزینه اجاره منابع زیرساخت رایانش ابری

۶- نتیجه گیری

در پایان بعنوان نتیجه گیری می‌توان گفت مدل پیشنهادی که با کمک الگوهای درخواست ورودی در طول زمان اقدام به مقیاس‌دهی خودکار ماشین‌های مجازی می‌کند، درصورت استفاده از سرویس‌های مشخص و جریان‌های کاری کارا در سازمان می‌توان ابزار مناسبی جهت کاهش هزینه اجاره منابع پردازشی و تطبیق به موقع با شرایط بحرانی باشد. همانطور که نشان دادیم، با کمک خوشه بندی الگوهای ورودی و محاسبه چیدمان بهینه منابع می‌توان درهر لحظه مقیاس‌دهی مطلوبی از نظر بهینگی، هزینه کمتر و چیدمان مناسب سرویس‌ها روی ماشین‌های مجازی ارائه نمود.

۷- مراجع

[1] Yeo, S., and H-HS L. "Using mathematical modeling in provisioning a heterogeneous cloud computing environment." **Computer** 44, no. 8 (2011): 55-62.
[2] Chen, Y., Tianyu W., and Jianxin L. "An efficient resource management system for on-line virtual cluster provision." In **Cloud Computing. CLOUD'09. IEEE International Conference on**, pp. 72-79. IEEE, 2009.
[3] Lorido-Botrán, T., José M., and Jose A. L. "Auto-scaling techniques for elastic applications in cloud environments." **Department of Computer Architecture and Technology, University of Basque Country, Tech. Rep. EHU-KAT-IK-09** 12 (2012).
[4] RightScale Cloud Management Available On. http://www.rightscale.com/, 2012.
[5] Han, R. et al. "Lightweight resource scaling for cloud applications." In **(CCGrid), 12th IEEE/ACM International Symposium on**, pp. 644-651. IEEE, 2012.
[6] Hasan, M. et al. "Integrated and autonomic cloud resource scaling." In **NOMS**, pp. 1327-1334. IEEE, 2012.
[7] Barrett, E. et al. "Applying reinforcement learning towards automating resource allocation and application scalability in the cloud." **Concurrency and Computation: Practice and Experience** 25, no. 12 (2013): 1656-1674.
[8] Tesauro, G., et al. "A hybrid reinforcement learning approach to autonomic resource allocation." **ICAC'06. IEEE International Conference on**. IEEE, 2006.

[9] Urgaonkar, B., et al. "Agile dynamic provisioning of multi-tier internet applications." **ACM Transactions on Autonomous and Adaptive Systems (TAAS***)* 3.1 (2008)

[10] Dutreilh, Xavier, et al. "Using reinforcement learning for autonomic resource allocation in clouds: towards a fully automated workflow." ***7th ICAS***, 2011.

[11] Villela, D. et al. "Provisioning servers in the application tier for e-commerce systems." **ACM (TOIT)** 7.1 (2007)

[12] Roy, N. et al. "Efficient autoscaling in the cloud using predictive models for workload forecasting." **(CLOUD), International Conference** *on*. IEEE, 2011.

[13] Islam, Sadeka, et al. "Empirical prediction models for adaptive resource provisioning in the cloud." **Future Generation Computer Systems** 28.1 (2012): 155-162.

[14] Siahmargooei, Y., et al. "Near-Optimal Virtual Machine Packing Based on Resource Requirement of Service Demands Using Pattern Clustering." **arXiv preprint arXiv:1406.7285** (2014).