نوع مقاله: پژوهشی

پذیرش مقاله: ۱۴۰۰/۰۹/۰۵
دریافت مقاله: ۱۴۰۰/۱۱/۱۷

ارائه یک روش مقیاس‌پذیری خودکار به منظور استفاده بهینه از منابع محاسباتی در محیط‌های رایانش ابری

نرگس‌سادات حسینی مونس
دانشجوی کارشناسی ارشد دانشکده کامپیوتر – دانشگاه علم و صنعت ایران – تهران – ایران
پست الکترونیکی: n_hoseini@comp.iust.ac.ir

مهرداد آشتیانی*
استادیار دانشکده کامپیوتر – دانشگاه علم و صنعت ایران–تهران–ایران
پست الکترونیکی:m_ashtiani@iust.ac.ir

چکیده

محاسبات ابری بسیاری از ارائه‌دهندگان برنامه‌ها را به سمت استقرار برنامه‌های تحت وب بر روی محیط‌های ابری جذب می‌کند. در مقایسه با روش‌های سنتی، رایانش ابری به کاهش هزینه‌ها کمک می‌کند. کاربران محیط‌های ابری می‌توانند در یک بازه زمانی مشخص، تعدادی ماشین مجازی را راه‌اندازی کنند و فقط به اندازه‌ای که از منابع استفاده می‌کنند هزینه پرداخت کنند، که این امر موجب کاهش هزینه‌های تهیه سخت افزارهای حقیقی و پیچیده می‌شود. از آن‌جایی که میزان درخواست‌ها از برنامه‌های وب در زمان‌های مختلف متفاوت است، تعیین مقدار مناسب منابع ابری مورد نیاز اغلب دشوار است و یکی از چالش‌های مهم در رایانش ابری خواهد بود. گرچه راه‌حل‌های مختلفی برای مدیریت تامین منابع ارائه شده‌است، برای مدیریت مؤثرتر سیستم‌های ذخیره‌سازی مبتنی بر ابر نیاز به روش‌های جدیدتری است. بر این اساس، این کار پژوهشی یک روش پویای مبتنی بر منطق فازی را برای بهبود عملکرد تامین منابع در سیستم‌های ذخیره‌سازی مبتنی بر ابر ارائه می‌کند. نتایج مقایسه رهیافت پیشنهادی با روش‌های دیگر نشان دهنده عملکرد مثبت این روش در کاهش زمان پاسخگویی به درخواست کاربران و مقدار CPU مصرفی در مرکز داده است.

واژه‌های کلیدی: مقیاس‌پذیری خودکار، رایانش ابری، استنباط فازی، مدل حلقهٔ OODA

مقدمه

رایانش ابری یک اصطلاح کلی برای هر نوع خدمتی است که شامل ارائه خدمات میزبانی شده از طریق اینترنت است. به عبارتی دیگر، رایانش ابری، در دسترس بودن منابع سیستم رایانه‌ای، به ویژه ذخیره داده(ذخیره‌سازی ابری) و قدرت محاسباتی، بدون مدیریت مستقیم کاربران تعریف شده است. رایانش ابری با ارائه منابع محاسباتی خود به‌عنوان خدمت، از درگیر کردن کاربران با موضوعاتی همچون خرید، مالکیت و نگهداری از مراکز داده و کارسازهای فیزیکی جلوگیری می‌کند که این موضوع باعث کاهش هزینه‌های کاربران می‌شود[۲].

خدمات رایانش ابری در حال حاضر طیف گسترده‌ای از خدمات را در بر می‌گیرد. این طیف، شامل خدماتی

* نویسندهٔ مسئول

مانند اصول اولیه ذخیره‌سازی، شبکه و قدرت محاسباتی تا پردازش زبان‌های طبیعـــی و هوش مصنوعی و به‌طور کلـــی هر خدمتی که نیاز به منابع ســـخت‌افزاری و فیزیکی دارد، می‌شود. رایانش ابری به دلایل زیادی از جمله صرفه جویـــی در هزینه، افزایش بهره‌وری، ســـرعت و کارایی و عملکرد و امنیت، گزینه‌ای محبوب برای شـــرکت‌ها و ارائه دهندگان برنامه‌ها است.

ایجـــاد برنامه‌های کاربـــردی ابری بـــا کیفیت بالا یک چالش مهم تحقیقاتی اســـت. در واقع، تخصیص مناســـب منابع در شرایطی که نیاز به اعمال خاصیت کشسانی برای مقیاس‌پذیری پویای منابع محاســـباتی بـــا توجه به تغییر تقاضا است، یک مســـئله مهم و چالش‌برانگیز خواهد بود. پیاده‌ســـازی یک سازوکار مناســـب تخصیص منابع برای به‌دســـت آوردن توازن میان هزینه و عملکرد بســـیار مهم است. نحوه تخصیص منابع باید به گونه‌ای باشد که در عین تامین منابع مورد نیاز برای اجرای درخواست‌های کاربران، مقدار منابـــع مصرفی و هزینه‌های موجود در حالت کمینه خود باشند و از هر هزینهٔ اضافی و غیرضروری خودداری شود. برای رسیدن به این هدف، تشخیص شرایط سیستم بسیار حیاتی است زیرا با تشخیص درست شرایط سیستم انتخاب بهترین تصمیم برای تامین منابع عملی می‌شـــود. باتوجه به امکان وجود نوســـان در بار ورودی برنامه‌های کاربردی، مطلوب اســـت که یک ســـازوکار خودکار برای انطباق میزان منابع اختصاص داده شده بر اساس نیازهای سیســـتم در هر زمان وجود داشته باشد[۳]. مقیاس‌پذیری خودکار یک تکنیک برای تنظیم پویای منابع اختصاص داده شـــده به برنامه‌های کاربردی با توجه به حجم‌کار ورودی اســـت. چالش اصلی در مقیاس‌پذیری خودکار، افزایش و یا کاهش منابع با توجه به نوســـانات در حجم‌کار ورودی است. روش‌های مقیاس‌پذیری خودکار بدون دخالت انسان و به‌صورت خودمختار کار می‌کنند[٤]. برای مدیریت موثر برنامه‌های کاربردی، الگوریتم مقیاس‌پذیری درکنار کمینه کردن هزینه باید به رویدادهای سیســـتم واکنش مناســـب نشـــان دهد و منابع مورد نیاز برای انجام وظایف سیستم را تامین کند. در مسئله مقیاس‌پذیری خودکار، تکنیک‌های تصمیم‌گیـــری برای تخصیص منابع به فرایندهای مختلف، به دو دســـته واکنشـــی[1] و فعال[2] تقســـیم می‌شوند. شیوه واکنشـــی به‌طور منظـــم رویدادهایی همچـــون میزان کار، میـــزان مصرف پردازنـــده و صف را مشـــاهده می‌کند و عملیـــات مقیاس‌پذیری را با توجه به ایـــن رویدادها انجام می‌دهد. اما در روش فعال ســـعی می‌شـــود با استفاده از اطلاعات گذشته ازدحام برنامه پیش‌بینی شود و براساس آن تصمیم‌گیری انجام شود. به دلیل سادگی و عملکرد قابل قبول روش‌های واکنشی، بیشتر سازمان‌ها و شرکت‌ها از این روش اســـتفاده می‌کنند. مشـــکل رویکرد واکنشی این اســـت که آســـتانه و تصمیمات مقیاس‌پذیری، باتوجه به حجم‌کار و شـــرایط سامانه تغییر نمی‌کند و ثابت است که این موضوع می‌تواند مانع انتخاب بهترین تصمیم در تامین منابع شـــود. یک روش برای حل این مشکل، اضافه کردن پویایی به رویکردهای واکنشی بوده به این صورت که در بازه‌های زمانی مشخص، شـــرایط سامانه مشخص شود و متناسب با شـــرایط سامانه در آن بازه زمانی مشخص، رویکرد واکنشی به تصمیم‌گیری بپردازد.

در ایـــن کار پژوهشـــی ما یک رویکـــرد مقیاس‌پذیری واکنشی را پیشنهاد می‌کنیم که مقیاس‌پذیری را بر اساس مشخصه‌های ســـامانه به‌صورت پویا انجام می‌دهد. برای این منظور، منطق فازی بســـیار کاربردی به نظر می‌رسد زیرا برخلاف روش‌های یادگیری ماشـــین برای ســـاختن مدل به داده و زمان آموزش طولانی نیاز ندارد و می‌تواند عملکرد مناســـبی داشـــته باشـــد. در روش پیشنهاد شده تلاش شده‌است با استفاده از برخی از ویژگی‌های سامانه مانند زمان پاســـخگویی به درخواست‌ها و مقدار مصرف پردازنده توســـط ماشـــین‌های مجازی، وضعیت سامانه مشـــخص شود و با توجه به وضعیت حال‌حاضر سامانه، متغیرهـــای مورد نیاز در تصمیم‌گیری برای مقیاس‌پذیری

1- Reactive
2- Proactive

مانند قوانین فازی، به‌صورت پویا و متناسب با وضعیت حال حاضراستفاده شوند. استفاده از رویکرد واکنشی در کنار پویایی اضافه شده به آن به ما کمک می‌کند که علاوه بر سرعت در تصمیم‌گیری، به دقت مناسب و قابل قبول متناسب با شرایط، دست پیدا کنیم.

در مسئله مورد بررسی، مرکز داده محیط رایانش ابری شامل تعدادی ماشین‌فیزیکی است. هر ماشین مجازی دارای تعدادی هسته پردازنده بوده و تمامی ماشین‌های مجازی درمرکز داده از نظر قدرت محاسباتی یکسان هستند. برای هر کاربر، تعدادی ماشین‌مجازی درنظر گرفته شده است که وظیفهٔ اجرای کارهای ارسالی از طرف کاربران را بر عهده دارند. کارهای ورودی از طرف کاربران از یکدیگر مستقل می‌باشند و منابع اختصاص داده شده به آن‌ها در طول اجرا ثابت می‌باشد. هر کار ارسالی از طرف کاربر دارای طول مشخصی می‌باشد که با توجه به قدرت محاسباتی منبع اختصاص داده شده به کار ورودی، می‌توان یک مهلت اجرای تخمینی برای هر وظیفه از طرف کاربر در نظر گرفت. روش پیشنهاد شده، یک روش مقیاس‌پذیری واکنشی فازی است که منطبق با مدل حلقه OODA، با در نظر گرفتن ویژگی‌های سیستم، مانند مدت زمان پاسخگویی به درخواست کاربران و میانگین مقدار پردازنده مصرف شده، بهترین قوانین فازی را از مجموعه قوانینی که از پیش برای حالت‌های مختلف سیستم تعریف شده‌است، انتخاب کرده و عملیات مقیاس‌پذیری را به‌صورت خودکار برای تامین منابع مورد نیاز کارهای درخواستی توسط کاربران که مستقل و غیرقابل پیش‌بینی هستند، انجام می‌دهد. قوانین تعریف شده می‌توانند به سادگی «اگر مدت زمان پاسخگویی زیاد بود، تعداد منابع محاسباتی افزایش پیدا کند»، باشد. روش پیشنهاد شده بدون نیاز به جمع‌آوری و ذخیره حجم داده زیاد، به دنبال کاهش متغیرهایی همچون مدت زمان پاسخگویی و میانگین قدرت محاسباتی مصرفی منابع است.

روش پیشنهادی از نظر عملکرد در مقایسه با دیگر روش‌های واکنشی معرفی شده، با توجه به وجود مؤلفهٔ پویایی، عملکرد بهتری دارد و از نظر قدرت محاسباتی و حافظه مورد نیاز برای اجرای این روش، در مقایسه با روش‌های فعال، کم‌هزینه و مناسب‌تر است. نوآوری و دستاوردهای این کار پژوهشی به‌طور خلاصه شامل موارد زیر است:

۱.خودکارسازی فرآیند مقیاس‌پذیری بدون دخالت انسان و در نتیجه دست‌یابی به تصمیم‌گیری بهینه برای بهبود مؤلفه‌های ارزیابی

۲.عدم نیاز به جمع‌آوری حجم بالای داده تاریخی.

۳.عدم نیاز به صرف زمان آموزش.

٤.و پویایی مضاعف در مراحل مختلف الگوریتم.

در ادامه پس از مقدمه اولیه گفته شده در این بخش، در بخش دوم به معرفی و توضیح ادبیات و مفاهیم اولیه مورد استفاده در این کار پژوهشی مانند مفاهیم مرتبط با مقیاس‌پذیری خودکار و توضیح عملکرد سیستم‌های استنباط فازی پرداخته می‌شود. در بخش سوم تعدادی از کارهای انجام شده و پژوهش‌های اخیر در حوزه مقیاس‌پذیری خودکار معرفی می‌شوند و نقاط مثبت و منفی هر کدام توضیح داده می‌شود. در بخش چهارم، جزئیات رهیافت پینشهادی تشریح می‌شود. در بخش پنجم، ارزیابی رهیافت پیشنهادی در مراحل متفاوت، مانند مرحله مقایسه با روش‌های دیگر و یا مرحله تحلیل حساسیت، انجام می‌شود و در بخش آخر، بخش ششم، به جمع‌بندی و نتیجه‌گیری از پژوهش انجام شده و ارائه پیشنهاد برای کارهای آینده، پرداخته می‌شود.

۲- مفاهیم پایه

در این بخش به‌صورت خلاصه به معرفی برخی از مفاهیم پایه‌ای برای درک بهتر این مقاله پرداخته می‌شود.

۲-۱- مقیاس‌پذیری خودکار

یک سامانه که از ویژگی کشسانی محیط ابری با اختصاص مقدار مناسب منابع به حجم کار برنامه‌ها،

با نظارت دستی اندک و یا بدون نظارت دستی استفاده می‌کنـد، می‌توانـد به‌عنوان یـک سـامانه مقیاس‌پذیر خودکار در نظر گرفته شود. مقیاس‌پذیری خودکار یک روش بـرای تنظیم پویای منابع محاسـباتی اختصاص داده شـده به برنامه‌های کاربردی با توجه به حجم‌کار ورودی است[۵]. چالش اصلی در مقیاس‌پذیری خودکار افزایـش و یـا کاهش منابع بـا توجه به نوسـانات در حجم‌کار ورودی خواهـد بود. روش‌های مقیاس‌پذیری خودکار بدون دخالت انسان و به‌صورت خودمختار کار می‌کنند[٤].

۲-۱-۱- انواع مقیاس‌پذیری

عملیـات مقیاس‌پذیری به دو دسـته واکنشـی و فعال تقسـیم می‌شـوند[٦]. زمان انجام عملیات مقیاس‌پذیری، نشـان دهنده روش اتخاذ شـده برای مقیاس‌پذیری است. در روش واکنشـی، پس از به وجود آمدن تغییرات در بار ورودی و تحلیل وضعیت فعلی سیسـتم، منابع اختصاص داده می‌شـوند در حالی‌کـه در روش فعـال، ابتدا سـعی می‌شـود تغییرات بار ورودی با اسـتفاده از داده‌هایی که از عملکرد سیسـتم در زمان‌های گذشته جمع‌آوری شده اسـت، پیش‌بینی شود و براساس این پیش‌بینی اختصاص منابع اتفاق می‌افتد. امروزه در بسیاری از روش‌های جدید پیشنهاد شده، ترکیبی از دو روش گفته شده برای رسیدن به نتایج بهتر استفاده می‌شود.

۲-۱-۲-سیاست‌های مقیاس‌پذیری

مقیاس‌پذیری می‌توانـد در دو حالت افقی و یا عمودی اتفـاق بیفتد[۷]. مقیاس‌پذیری افقی بـه معنای افزایش و یا کاهش تعداد منابع موجـود در مرکز داده و مقیاس‌پذیری عمـودی به معنـای افزایش و یا کاهـش ظرفیت و قدرت منابـع موجود در مرکز داده اسـت. ارائه‌دهندگان خدمات ابری اغلب از مقیاس‌پذیری افقی برای تامین خودکار منابع اسـتفاده می‌کنند به این صورت که یـک منبع قابل تنظیم در اختیـار کاربر قرار می‌گیرد کـه در آن کاربر می‌تواند ماشین‌های مجازی را با تعیین حافظه، هسته، پهنای باند و دیگر ویژگی‌های لازم پیکربندی کند و با توجه به نیاز خود، تعدادی ماشین‌مجازی اضافه و یا کم کند. در مقیاس‌پذیری عمودی، کاربر می‌تواند منابع ماشـین‌های مجازی همانند CPU، حافظـه و پهنای‌باند شـبکه را کـه در اختیار دارد، مجدداً پیکربندی کند.

۲-۱-۳-رویکردهای مقیاس‌پذیری

در روش‌های مقیاس‌پذیـری خودکار، رویکرد مقیاس نشـان‌دهنده متغیر یا متغیرهایی است که تمرکز روش بر روی بهبود آن اسـت. در مقیاس‌پذیری، چشم‌انداز روش می‌تواند بهبود یک یا چند عامل مانند هزینه، بهینه‌سـازی منابع و بهبود کیفیت خدمات باشد. مشخص کردن رویکرد مقیاس‌پذیـری به افراد کمک می‌کند تا روش مقیاس‌پذیری خودکار را با توجه به نیاز مشخص و خاص خود انتخاب کنند.

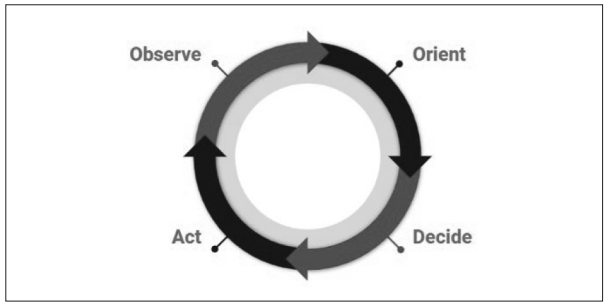
۲-۱-۴- معیارهای عملکرد

در روش‌های متفاوتی که وجود دارد، مهم‌ترین مرحله، ارزیابی مدل و روش پیشنهاد شده و بررسی پارامترهایی اسـت که روش در بهبود آن‌ها عملکرد مثبتی داشته است. پارامترها و معیارهای متنوعی مانند زمان پاسـخ، میزان استفاده از CPU و میزان درخواست ورودی برای بررسی عملکرد سیستم استفاده می‌شـوند. انتخاب روش مناسب مقیاس‌پذیری بـا توجه به معیارهـای عملکرد روش، یک مرحله ضـروری برای افزایش قابلیـت اطمینان کاربرانی اسـت که برنامه‌های خود را به محیط‌هـای رایانش ابری منتقل می‌کنند.

۲-۲- حلقه OODA

برای مدیریت موثـر برنامه‌های کاربـردی، الگوریتم مقیاس‌پذیری باید به رویدادهای سیسـتم واکنش مناسب نشـان دهد. باتوجه به این‌که فرایند مقیاس‌پذیری خودکار با مدل حلقه OODA (شـکل ۱) مطابقت دارد، در بسـیاری از الگوریتم‌هـا از ایـن مدل بـرای مقیاس‌پذیری خودکار استفاده می‌شود. این حلقه از چهار مرحله شامل مشاهده[۳]،

3- Observation



شکل ۱: مدل حلقهٔ OODA [۱]

جهت‌گیری[4]، تصمیم‌گیری[5] و عملیات[6] تشکیل شده‌است.

۲-۳- استنباط فازی

نظریه منطق فازی برای استفاده از دانش متخصص در قالب قوانین فازی برای تعیین یک اقدام مناسب و بهینه طراحی شده است. منطق فازی می‌تواند در هرس فضای حالت مسئله نیز تاثیرگذار باشد. استنباط فازی با مقایسه مقادیر واقعی و مقادیر مرجع عوامل موثر در سیستم، شرایط را کنترل می‌کند. منطق فازی برای استفاده در محیط‌های تصادفی و پویا با مجموعه‌ای از عدم قطعیت‌ها مانند شبکه‌های کامپیوتری، سیستم‌های توزیع شده و محیط‌های ابری مناسب است و برای حل مشکلات تصمیم‌گیری در حوزه مختلف مانند اینترنت اشیا، داده‌کاوی، رایانش ابری، شبکه‌های اجتماعی و پردازش تصویر، مورد استفاده قرار گرفته است. یک سیستم استنباط فازی با فازی‌سازی معیارهای ورودی کار می‌کند. تعداد معیارهای ورودی می‌توان یک، دو، سه و حتی بیشتر باشد. پس از دریافت اطلاعات ورودی، یک موتور استنباط به تصمیم‌گیری بر اساس داده‌های ورودی و پایگاه قاعده می‌پردازد. سیستم استنباط فازی از قسمت‌های پایگاه قاعده، واحد فازی‌سازی، تابع عضویت، واحد استنباط فازی و واحد خنثی‌سازی فازی تشکیل شده است[۳]. برای انجام استنباط فازی روش‌های متفاوتی از جمله دو روش ممدانی و سوگنو[7] وجود دارد که در این کار تحقیقاتی از روش ممدانی استفاده شده‌است[۸].

4- Orientation
5- Decision
6- Action
7- Sugeno

۳ - مروری بر کارهای مرتبط

تاکنون روش‌های متفاوتی در زمینه مقیاس‌پذیری خودکار پیشنهاد شده است. این روش‌ها در گذشته بیشتر با استفاده از روش‌های واکنشی مانند روش‌های مبتنی برآستانه و روش‌های تحلیل سری‌های زمانی، پیاده سازی می‌شدند اما امروزه بیشتر روش‌ها به تکنیک فعال روی آورده‌اند و اغلب با استفاده از روش‌های یادگیری ماشین، یادگیری تقویتی و روش‌های دیگر در زمینه هوش مصنوعی، پیاده‌سازی می‌شوند. مشکل روش‌های فعال نیاز آن‌ها به حجم زیاد داده، قدرت محاسباتی قابل قبول و زمان طولانی برای اجرا است. البته امروزه با پیشرفت فناوری، استفاده از این روش‌ها بسیار ساده و قابل قبول شده است. از طرفی مشکل اساسی روش‌های واکنشی، ایستا بودن این روش‌ها است. در روش‌های مقیاس‌پذیری خودکار به‌صورت واکنشی، الگوریتم، همواره اطلاعات محیط رایانش ابری را بررسی می‌کند و پس از بررسی اطلاعات به تصمیم‌گیری درباره تامین منابع می‌پردازد. برای پیاده‌سازی الگوریتم‌های واکنشی مقیاس‌پذیری خودکار از روش‌هایی مانند استنباط فازی[۱]، روش‌های مبتنی بر آستانه[۹] و روش‌های مبتنی بر تطبیق الگو[۱۰]، استفاده می‌شود.

روش‌های مبتنی بر آستانه از مهم‌ترین روش‌های مقیاس‌پذیری خودکار با استفاده از تکنیک واکنشی می‌باشند. در روش پیشنهاد شده توسط آقای Hung و همکارانشان[۹]، یک روش مقیاس‌پذیری خودکار مبتنی بر آستانه پیشنهاد شده است. در روش پیشنهاد شده، درخواست‌های کاربران توسط متعادل کننده بار میان ماشین‌های مجازی موجود در هر خوشه، توزیع می‌شوند. هر ماشین‌مجازی شامل تعدادی نشست[8]است و وظیفه اجرای کارهای ورودی بر عهده آن‌ها خواهد بود. در این روش، پارامتر مورد بررسی، تعداد نشست‌های فعال در هر ماشین‌مجازی است. درصورتی که تعداد نشست‌های فعال در یک ماشین‌مجازی از آستانه بالایی مشخص شده بیشتر

8- Session

باشد، الگوریتم، دستور افزایش تعداد ماشین‌های مجازی را اعلام می‌کند و این افزایش به اندازه یک ماشـــین‌مجازی خواهد بود. در شرایطی که تعداد نشست‌های فعال در یک ماشین‌مجازی از آستانه پایینی مشخص شده کمتر باشد، آن ماشـــین مجازی بیکار و بدون اســـتفاده اســـت. برای کاهش تعداد ماشین‌های مجازی در صورتی که بیش از دو ماشین‌مجازی در یک خوشه بیکار باشند، الگوریتم دستور تخریب و کاهش یکی از ماشـــین‌های مجازی را می‌دهد. با توجه به ســـادگی الگوریتم بیان شده، این الگوریتم از نظر اجرا مدت زمان کمی نیاز دارد اما این الگوریتم نیز همانند بســـیاری از روش‌های مبتنی بر آستانه دیگر، از مرزهای ایســـتا اســـتفاده کرده که این موضوع مانع از رسیدن به عملکرد مطلوب می‌شود. زیرا شرایط سیستم همواره ثابت نیســـت و به‌صورت پویا در تغییر بوده و با در نظر گرفتن یک آســـتانه ثابت، مهم‌ترین ویژگی سیســـتم‌های رایانش ابری که پویایی آن است را در نظر نگرفته‌ایم.

آقای Liao و همکاران ایشان [۱۴]نیز از یک روش مبتنی بر آســـتانه برای الگوریتم مقیاس‌پذیری خودکار استفاده کردنـــد که تفاوت اصلی این روش بـــا روش قبلی در پویا بودن مرزها و آســـتانه‌های آن اســـت. هدف اصلی روش ارائه شده، مقیاس‌پذیری خودکار برای خدمات وب آمازون اســـت. در این روش، متغیری که آستانه برای آن مشخص شده، مقدار اســـتفاده از منبع پردازنده مرکزی است. ایده اصلی در این روش، پویایی مرزهای آن است. در توضیح این روش آمده اســـت که زمانی که درصد استفاده از CPU بالا بوده و با بررســـی شـــرایط محیط متوجه افزایش بار کاری هر ماشین‌مجازی می‌شویم، احتمال این که در تامین منابع دچار کمبود شویم بالا می‌رود و به همین علت و برای جلوگیـــری از کمبود منابع، مرز بالایـــی با هر افزایش بار، مقداری کاهش پیدا می‌کند. به‌صورت مشـــابه، برای حالتی که با بررسی شـــرایط هر ماشین‌مجازی متوجه می‌شویم که بار ورودی در حال کاهش اســـت، احتمال تامین منابع بیش از حد مورد نیاز وجود داشـــته و در نتیجه، آســـتانهٔ بالایی با هر کاهش بار، مقداری افزایش پیدا می‌کند. مقدار تغییر آستانه، چه در کاهش و چه در افزایش به اندازه ۰/۵ درصد مقدار حال حاضر آستانه است.

برای آستانهٔ پایانی نیز تغییرات پویا صادق خواهد بود. تغییرات مرز پایینی مشابه مرز بالایی بوده به این صورت که در حالتی که در مرحله افزایش بار باشـــیم، مرز پایینی افزایش پیدا می‌کند و در صورتی که در مرحله کاهش بار باشیم، آستانه پایینی کاهش پیدا خواهد کرد. نکته‌ای که در مرز پایینی وجود دارد این اســـت که در شرایطی که بیش از ۵۰ درصد از ماشـــین‌های مجازی درحال انجام وظایف باشند، برای جلوگیری از نوسان در مقیاس‌پذیری و کاهش و افزایش متناوب منابع، مرز پایینی در کم‌ترین حالت خود قرار می‌گیرد. همان‌طور که مشـــخص است استفاده از این روش می‌تواند در بهبود عملکرد سیستم بسیار موثر باشد. زیرا با تعیین مرزهای پویا، الگوریتم، هم با ویژگی پویایی سیســـتم سازگار شـــده و عملکرد مناســـبی دارد و هم از نوسان‌های نامطلوب در مقیاس‌پذیری جلوگیری می‌کند.

روش‌هـــای مقیاس‌پذیری مبتنی بر اســـتنباط فازی، از منطق و سیستم استنباط فازی برای تصمیم‌گیری درمورد نحوه تامین منابع اســـتفاده می‌کننـــد. در این روش‌ها پس از جمع‌آوری داده مورد نیاز برای اســـتنباط فازی، مقادیر ورودی به یک سیســـتم اســـتنباط فازی داده می‌شـــود و خروجی مورد نظر تحویل گرفته می‌شـــود. در روش‌های متفاوت وابســـته به شـــرایط خاص هر مســـئله، خروجی سیســـتم اســـتباط فازی متفاوت اســـت. امـــا به‌طور کلی خروجـــی، نحوه مقیاس‌پذیری را مشـــخص می‌کند. مزیت اصلی این روش نســـبت به روش آستانه‌گذاری، پیچدگی بیشتر آن بوده که از پارامترهای بیشتر و پیچیده‌تری برای تصمیم‌گیری استفاده می‌کند.

روش پیشـــنهاد شده توسط Tseng و همکاران[۱۱] یک روش مبتنی بر استنباط فازی است. در این روش، تعدادی متغیر به‌عنوان ورودی در نظر گرفته شـــده و به سیســـتم اســـتنباط فازی داده می‌شـــود تا خروجی موردنیاز تولید

شود. برای استفاده از سیستم فازی ابتدا باید اصطلاحات فازی و توابع عضویت متغیرها مشخص شود. در این روش از تابع توزیع گاوسی برای توابع عضویت متغیرها استفاده شده است. نکته قابل توجهی که وجود دارد این است که تابع عضویت برای هر متغیر در شرایط مختلف متفاوت است. به این معنا که برای هر متغیر سه حالت زیاد، متوسط و کم وجود دارد که تابع عضویت هر متغیر متناسب با شرایطی که دارد تعریف و استفاده می‌شود. در این رهیافت، متغیرهای ورودی مورد بررسی مقدار CPU مصرفی، حافظه و شبکه و متغیر نهایی و خروجی، متغیر مقیاس است. برای هر کدام از متغیرهای خروجی و ورودی، سه اصطلاح فازی کم، متوسط و زیاد تعریف شده است. باتوجه به این‌که این روش از منطق فازی برای تصمیم‌گیری استفاده می‌کند، عملکرد قابل قبول و مناسبی دارد اما در نقطه مقابل نقاط ضعفی نیز برای این روش می‌توان در نظر گرفت از جمله این‌که پایگاه قاعده دچار مشکل ایستایی بوده ویک پایگاه قاعده با یک سری از قوانین در تمامی حالات استفاده می‌شوند که این موضوع می‌تواند در عملکرد روش اثر مخربی داشته باشد.

در کار پژوهشی انجام شده توسط آقای قبائی و همکاران ایشان [۸] نیز از سیستم استنباط فازی برای مقیاس‌پذیری استفاده شده‌است. این روش همانند رهیافت پیشنهادی ارائه شده، از مدل حلقهٔ OODA برای مدیریت بهتر استفاده می‌کند. سیستم مورد بررسی در این روش، دارای تعدادی گره‌داده بوده که وظیفه اجرای وظایف محول شده از طرف کاربران را بر عهده دارند. در این روش، متغیرهای ورودی به سیستم استنباط فازی، دو متغیر VPT به معنای تعداد بازدید از یک بلوک داده در یک بازه زمانی و AVT به معنای میانگین زمانی که برای بازدید یک بلوک داده صرف می‌شود، هستند. برای فازی‌سازی این دو متغیر، ابتدا دو مرز بالایی و پایینی برای هر کدام در نظر گرفته می‌شود و سپس فازی‌سازی آن‌ها به کمک سه اصطلاح فازی کم، میانه و زیاد انجام می‌شود. نقاط ابتدایی و انتهایی اصطلاح فازی میانه، دو آستانه بالایی و پایینی هر متغیر خواهند بود.

مرزهایی پایینی و بالایی هر متغیر با تغییر شرایط سیستم، تغییر می‌کند و تغییرات آن وابسته به تعداد دفعاتی است که سیستم دچار نقض توافقنامه در سطح سیستم شده باشد. زمانی یک وظیفه و درخواست دچار نقض توافقنامه در سطح سیستم می‌شود که مدت زمان اجرای آن وظیفه از مدت زمان مشخص‌شده برای اجرای آن وظیفه توسط کاربر، بیشتر شده باشد. این رهیافت، برای استنباط فازی خود از روش ممدانی استفاده کرده است. در مرحله آخر نیز با مشخص شدن مقدار متغیر خروجی، یک منبع محاسباتی به محیط رایانش اضافه یا کم می‌شود.

با بررسی این روش، مشخص می‌شود که این روش نسبت به روش قبل پویایی بیشتری دارد (مانند پویایی مرزها در اصطلاحات زبانی متغیرها و تغییر آن‌ها متناسب با توافقنامه سطح سیستم)، اما همچنان می‌توان به این روش پویایی بیشتری اضافه کرد. به‌طور مثال، مانند روش قبلی، به جای استفاده از یک پایگاه قاعده ثابت، پویایی به پایگاه قاعده نیز اضافه شود و یا تعداد منابعی که به محیط محاسباتی اضافه و یا کم می‌شوند نیز یک عدد ثابت نباشد و متناسب با شرایط سیستم تغییر کند.

در روش‌های مقیاس‌پذیری مبتنی بر تطبیق الگو[9]، الگوریتم مقیاس‌پذیری در ابتدا بدون هیچ‌اطلاعات اولیه درمورد سیستم، شروع به کار می‌کند. با گذر زمان، الگوریتم، اطلاعات موردنیاز در مورد نحوه تغییر بار ورودی و تغییرات متغیرهای موثر در مقیاس‌پذیری را جمع‌آوری می‌کند. پس از جمع‌آوری داده‌های مورد نیاز، الگوریتم به دنبال الگوهای مشترک و تکراری در تغییرات پارامترهای سیستم می‌گردد. برای پیدا کردن الگوهای مناسب و درست، معمولاً در ابتدا یک مجموعه نامزد به‌عنوان الگوهای تکرار شونده در سیستم، در نظر گرفته می‌شوند. سپس، با معیارهایی که از پیش تعریف شده است، درستی هر کدام از الگوها بررسی می‌شود و

9- Pattern matching

از مجموعه نامزد، وابسته به الگوریتم ارائه شده، یک یا چند الگو به‌عنوان الگوی تکراری در نظر گرفته می‌شوند. درنهایت، الگوریتم در شرایطی که متوجه وقوع این الگو شود، دستور متناسب با شرایطی که از وقوع این الگو پیش‌بینی می‌کند را صادر می‌کند.

در روش پیشنهاد شده توسط Nowick و دیگران، از روش تطبیق الگو برای مقیاس‌پذیری خودکار استفاده شده است[۱۰]. در الگوریتم پیشنهادی، از دو قسمت برای تخمین شرایط سیستم و درنتیجه تصمیم‌گیری در رابطه با نحوه مقیاس‌پذیری، استفاده شده است. در قسمت اول، الگوریتم برخی از فاکتورهای تاثیرگذار بر بار ورودی را مورد بررسی قرار می‌دهد. از جملهٔ این فاکتورها که در بار ورودی تاثیرگذارند می‌توان به، بازه زمانی روز، تعطیلات و روزهای آخر هفته اشاره کرد. قسمت دوم یک الگوریتم تطبیق الگوی دوفازی[۱۰] را پیاده‌سازی می‌کند.

در این روش، پس از جمع‌آوری داده، به تحلیل سری‌های زمانی پرداخته می‌شود. برای به‌دست آوردن تناوب در سری‌زمانی، از خودهمبستگی[۱۱] استفاده شده است که نشان‌دهنده همبستگی متقاطع یک سیگنال با خودش است. پس از به‌دست آوردن الگوهای متناوب، مرحلهٔ بعدی، یافتن الگوهای مشابه با الگوهای موجود در داده‌های تاریخی است. در حقیقت در این قسمت، الگوریتم می‌خواهد تعیین کند که در گذشته چه زمانی نوسانات و مقادیر حجم کار مشابه ثبت شده است. شباهت بر اساس مقدار حجم کار و همچنین نوسانات الگو (اعم از افزایش، کاهش یا پایدار بودن حجم کار) تعریف می‌شود. پس از به‌دست‌آوردن الگوهای متناوب، الگوریتم با استفاده از روش Relief، به ارزیابی الگوهای(ویژگی‌ها) به‌دست آمده می‌پردازد و سپس از این داده‌ها برای مرحلهٔ یادگیری ماشین استفاده می‌کند. در مرحله یادگیری، الگوریتم با استفاده از الگوهای متناوب ارزش‌گذاری شده، شروع به یادگیری عملکرد سیستم می‌کند تا بتواند با استفاده از شرایط حال حاضر سیستم، تخمین مناسبی از شرایطی که در آینده پیش‌خواهد آمد داشته باشد. در این رهیافت برای یادگیری ماشین از روش جنگل تصادفی[۱۲]، استفاده شده است. در نهایت نیز برای بهبود بیشتر عملکرد، یک معیار اعتماد برای خروجی‌های تخمینی الگوریتم، تعریف می‌شود.

با توجه به این‌که این روش از یادگیری ماشین‌استفاده می‌کند و از داده‌های تاریخی استفاده می‌شود، این سیستم عملکرد مناسبی دارد. زیرا با تحلیل گذشته سیستم، به تخمین درستی از اتفاقات پیش‌رو می‌رسد و به همین‌خاطر می‌تواند بهترین تصمیم‌گیری‌ها در مورد نحوه تامین‌منابع را انجام دهد. مشکلی که در این روش وجود دارد، نیازمندی این روش به جمع‌آوری حجم زیادی از داده و صرف وقت نسبتاً زیاد برای یادگیری است. همچنین، این روش از جنگل تصادفی برای آموزش استفاده می‌کند که دارای پیچیدگی زیادی بوده و به قدرت محاسباتی بالایی نیاز دارد. با توجه به دو مورد گفته شده، در استفاده از این روش باید با بررسی سیستم به این نتیجه رسید که جمع‌آوری حجم زیاد داده و صرف وقت و قدرت محاسباتی بالا برای آموزش مدل، توجیه منطقی دارد.

در روش‌های مقیاس‌پذیری مبتنی بر یادگیری، الگوریتم مقیاس‌پذیری همواره درحال جمع‌آوری داده درمورد عوامل موثر در عملکرد و وضعیت سامانه از نظر مصرف منابع و خدمات ارائه شده به کاربر است. پس از جمع‌آوری حجم مناسبی از داده، الگوریتم مقیاس‌پذیری شروع به یادگیری شرایط سامانه می‌کند به این معنی که الگوریتم بتواند با داشتن متغیرهای سامانه مانند، بار ورودی و شرایط منابع، تخمینی از شرایطی که در پیش روی محیط رایانش ابری است، داشته باشد و با استفاده از این تخمین، قبل از رسیدن به نیاز به افزایش یا کاهش منابع، دستور انجام عملیات مناسب صادر شود. داده‌های جمع‌آوری شده بسته به شرایط هر مسئله متفاوت خواهد بود. برای پیاده‌سازی الگوریتم یادگیری نیز از روش‌های متفاوتی

10- TPM
11- Autocorrelation
12- Random Forest

جدول ۱: مقایسه کارهای انجام شده

روش ارائه شده	نویسندگان	نوع	سیاست	تکنیک
Auto-scaling model for cloud computing system[9]	Hung, Che-Lun and Hu, Yu-Chen and Li, Kuan-Ching	واکنشی	عمودی	مبتنی بر آستانه
Auto-scaling Strategy for Amazon Web Services in Cloud Computing[14]	Liao, Wen-Hwa and Kuai, Ssu-Chi and Leau, Yu-Ren	واکنشی	عمودی	مبتنی بر آستانه
A Rapid Auto-Scaling Mechanism in Cloud Computing Environment[11]	Tseng, Chia-Wei and Tsai, Ming-Shiun and Yang, Yao-Tsung and Chou, Li-Der	واکنشی	عمودی	مبتنی بر استنباط فازی
An auto-scaling mechanism for cloud-based multimedia storage systems: A fuzzy-based elastic controller[8]	Mostafa Ghobaei-Arani, Maryam Rezaei, Alireza Souri	واکنشی	عمودی	مبتنی بر استباط فازی
AME-WPC: Advanced model for efficient workload prediction in the cloud[10]	Katja Cetinski and Matjaz B. Juric	فعال	عمودی	مبتنی بر تطبیق الگو
A reinforcement learning based auto-scaling approach for SaaS providers in dynamic cloud environment[12]	Wei, Yi and Kudenko, Daniel and Liu, Shijun and Pan, Li and Wu, Lei and Meng, Xiangxu	فعال	عمودی	مبتنی بر تطبیق الگو
A Comparison of Reinforcement Learning Techniques for Fuzzy Cloud Auto-Scaling[13]	Arabnejad, Hamid and Pahl, Claus and Jamshidi, Pooyan and Estrada, Giovani	ترکیبی	عمودی	مبتنی بر استباط فازی / یادگیری

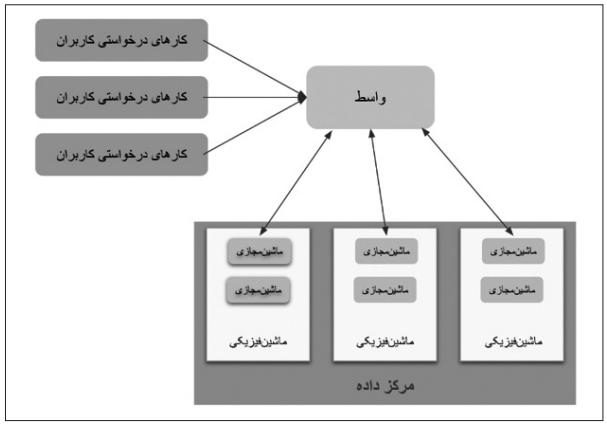
مانند Q-Learning، یادگیری تقویتی[۱۳] [۱۲]،یادگیری ماشین و شبکه‌های عصبی[٦] استفاده می‌شود. روش‌های مقیاس‌پذیری ترکیبی از ترکیب چندین روش با یکدیگر تولید می‌شوند. به‌طور مثال با ترکیب روش Q-Learning و منطق فازی می‌توان یک روش جدید برای مقیاس‌پذیری خودکار ایجاد کرد. نقطه مثبت روش‌های ترکیبی این است که در صورتی که روش‌های مناسب با یکدیگر ترکیب شوند، می‌توانند مکمل یکدیگر باشند و هر روش ایرادات روش دیگر را جبران کند. به‌طور مثال در کار تحقیقاتی انجام شده توسط خانم عرب‌نژاد و همکاران ایشان[۱۳]، یک روش fuzzy Q-Learning معرفی شده است. در این روش سعی شده است تا با ترکیب منطق فازی با یادگیری‌تقویتی، از طرفی مشکل ساده‌بودن و عدم استفاده از داده‌های محیط در روش فازی و از طرف دیگر، مشکل نیاز به ذخیره‌سازی حالات و اقدامات در روش‌های یادگیری تقویتی، مرتفع شود.

۴- رهیافت پیشنهادی

در رهیافت ارائه شده، الگوریتم مقیاس‌پذیری خودکار به دنبال پیدا کردن حالت بهینه اختصاص منابع بوده و برای رسیدن به این هدف از استنباط فازی استفاده می‌کند. همچنین، برای مدیریت بهتر از مدل حلقهٔ OODA در این روش استفاده شده است. رهیافت پیشنهادی از عواملی مانند میزان مصرف پردازنده و مدت زمان پاسخگویی محیط رایانش ابری برای تصمیم‌گیری در مورد نحوه تامین منابع استفاده می‌کند. تلاش رهیافت پیشنهادی این است که بتواند با تعیین مقدار مناسب منابع مورد نیاز، همزمان با انجام دادن درخواست‌های کاربران، از صرف هزینه‌های اضافه خودداری کند. روش پیشنهاد شده در این کار تحقیقاتی یک روش مقیاس‌پذیری پویا خودکار مبتنی بر منطق و استنباط فازی است که تلاش می‌کند متناسب با شرایط محیط رایانش ابری، تصمیمات و عملیات مقیاس‌پذیری را انجام دهد. محیط رایانش ابری در رهیافت پیشنهاد شده شامل قسمت‌های متفاوتی مانند، مرکز داده[۱٤]، ماشین‌های فیزیکی[۱٥]، ماشین‌های مجازی، واسط[۱٦] وکارهای درخواستی از طرف کاربران[۱۷] است. معماری کلی محیط رایانش ابری در شکل ۲ نشان داده شده است.

13- Reinforcement Learning

14- Data Center
15- Host
16- broker
17- Cloudlet



شکل ۲: معماری کلی محیط رایانش ابری

٤-١ مؤلفه‌های مورد بررسی در رهیافت پیشنهادی

همان‌طور که گفته شد، متغیرها و مؤلفه‌های متفاوتی در تصمیم‌گیری مقیاس‌پذیری درباره تامین منابع موثر هستند و متناسب با هر مسئله و شرایط خاص آن مسئله متفاوت خواهند بود. در این مسئله، دو مؤلفه میانگین مقدار پردازنده مصرف شده و تاخیر پاسخگویی به درخواست‌های کاربران، به‌عنوان دو مؤلفه مورد بررسی و موثر در تصمیم‌گیری انتخاب شده‌اند که در ادامه به توضیح هر کدام می‌پردازیم.

۴-۱-۱ میانگین مقدار پردازنده مصرف شده[18]

هر ماشین فیزیکی و ماشین‌مجازی در مرکز داده، دارای تعدادی هسته پردازشی و در نتیجه قدرت محاسباتی خاص خود هستند. هنگامی که کارهای درخواستی کاربران برای انجام به ماشین‌های مجازی سپرده می‌شود، هر ماشین مجازی با استفاده از منابع محاسباتی خود مانند پردازنده، تلاش می‌کند که در کم‌ترین زمان ممکن وظیفه خود را انجام دهد. هرچه بارکاری ورودی به مرکز داده و درنتیجه ماشین‌های فیزیکی و مجازی بیشتر باشد، نیاز به قدرت محاسباتی بیشتر بوده و در صورتی که تعداد منابع محاسباتی یا ماشین‌های مجازی ثابت باشند، هر ماشین‌مجازی با مصرف قدرت محاسباتی و مصرف پردازنده بیشتر، کارهای ورودی را کنترل می‌کند. با توجه به موضوع گفته شده، در رهیافت پیشنهادی مقدار میانگین پردازنده مصرفی به‌عنوان یکی از مؤلفه‌های موثر در مقیاس‌پذیری خودکار در نظر گرفته شده‌است. نحوه محاسبه این مؤلفه در فرمول ۱ قابل مشاهده است که در آن MCU میانگین پردازنده مصرف شده توسط تمامی ماشین‌های مجازی، CPU_{VM_i} مقدار پردازنده مصرف شده توسط ماشین‌مجازی i ام و |VM| تعداد ماشین‌های مجازی موجود در مرکز داده است.

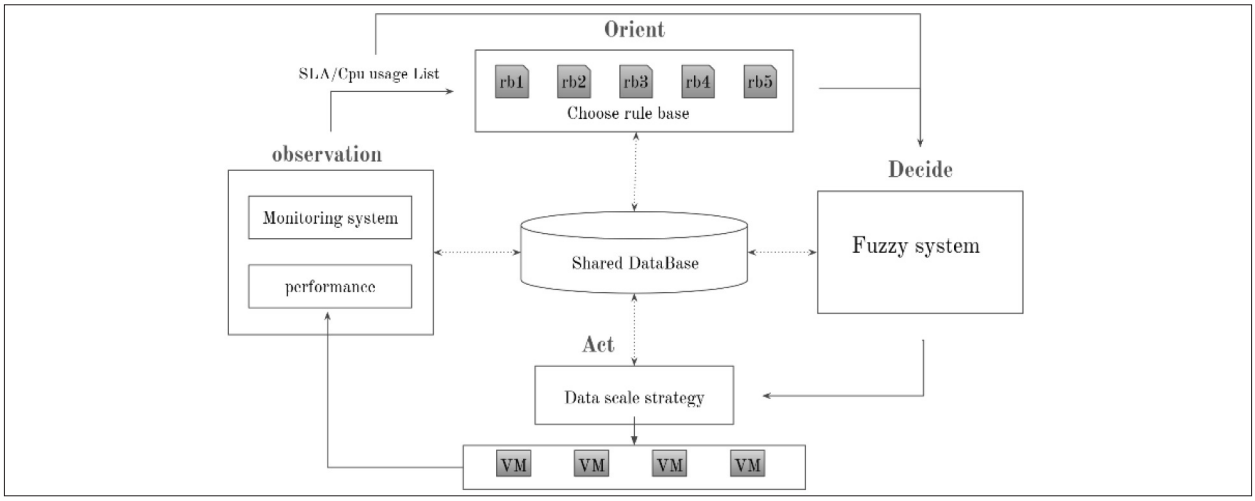
$$MeanCpuUsagt = MCU = \frac{\sum CPU_{VM_i}}{|VM|} \times 100 \quad (۱)$$

۴-۱-۲- تاخیر پاسخگویی به درخواست کاربران

منابع محاسباتی موجود در مرکز داده برای انجام درخواست‌های کاربران نیاز به مدت زمانی برای اجرای کامل درخواست‌ها دارند. مشخص است که هر چه مدت زمان اجرای درخواست‌ها کم‌تر باشد، عملکرد سیستم بهتر خواهد بود. نکته‌ای که قابل توجه است این است که با استفاده از برخی روش‌ها می‌توان زمان‌اجرای درخواست کاربران را کاهش داد اما با توجه به قدرت محاسباتی و حجم کار ورودی، مدت زمان پاسخگویی به درخواست کاربر، دارای یک کمینه بوده که مدت زمان پاسخگویی از آن کمینه، کم‌تر نمی‌شود. در مسئله مورد بررسی، کارهای درخواستی کاربران، دارای مهلت اجرای سخت[19] نبوده که در صورتی که سیستم نتواند آن را رعایت کند دچار مشکل شود. اما با توجه به حجم درخواست کاربران و قدرت محاسباتی منبع، کم‌ترین زمانی که یک منبع محاسباتی ایده‌آل می‌تواند برای اجرای درخواست نیاز داشته باشد، به‌عنوان مهلت اجرای نرم[20]، مشخص می‌شود. هرچه مدت زمان اجرای درخواست کاربر از این حد کمینه بیشتر شود، نشان دهنده پایین آمدن عملکرد سیستم و در نتیجه احتمال وجود کمبود منابع می‌باشد. هم‌چنین در شرایطی که هر درخواست در زمان کمینه خود اجرا شود، می‌توان این موضوع را برداشت کرد که سیستم در یک حالت ایده‌آل برای اجرای درخواست‌ها قرار دارد. برای محاسبه این مؤلفه مطابق با فرمول ۲ عمل می‌شود. در این فرمول

18- CPU

19- Hard deadline
20- Soft deadline



شکل ۳: معماری مدل حلقه OODA در رهیافت پیشنهادی

DeadLine، Responsetime، SLAT و |UserRequests| به ترتیب نشان‌دهنده نقض شدن یا نشدن توافقنامه سطح خدمات برای درخواست iام، مدت زمان اجرای درخواست iام، مهلت تعیین شده اجرای درخواست iام و تعداد درخواست‌های ورودی به مرکز داده هستند.

$$SLA_T = \begin{cases} 1 & Responsetime > DeadLine \\ 0 & otherwise \end{cases} \quad (۲)$$

$$SLA = \frac{\sum SLA_T}{|UserRequests|} \times 100$$

٤-۲ الگوریتم روش پیشنهادی

همان‌طور که گفته شد، در رهیافت پیشنهادی از مدل حلقه OODA که شامل چهار مرحله، مشاهده، جهت‌گیری، تصمیم‌گیری و عملیات است استفاده می‌شود. معماری کلی رهیافت پیشنهادی با توجه به حلقه OODA در شکل ۳ قابل مشاهده است.

۴-۲-۱ مرحله Observation

این مرحله، اولین مرحلهٔ اجرای الگوریتم مقیاس‌پذیری خودکار می‌باشد. در این مرحله اطلاعات اولیه درباره شرایط محیط رایانش ابری، برای استفاده در مراحل بعدی، جمع آوری می‌شود و در یک پایگاه داده مشترک ذخیره می‌شود.

۴-۲-۲- مرحله Orientation

این مرحله یکی از قسمت‌هایی است که پویایی مضاعف رهیافت پیشنهادی در آن پیاده‌سازی می‌شود.

الگوریتم ۱: شبه کد الگوریتم مرحله Observation

شبه کد الگوریتم مرحله Observation

```
Algorithm: Observation Phase
Input: List CPU, List ResponseTime, List DeadLine
Output: SLA, MCU
MCU = 0
SLA = 0
|For i = 1 to |VM
(MCU += CPU.get(i
(If ResponseTime.get(i) > DeadLine.get(i
++ SLA
End if
End for
|MCU = (MCU * 100) / |VM
|SLA = (SLA * 100) / |VM
Return MCU, SLA
```

وظیفه اصلی این مرحله، بررسی شرایط محیط با استفاده از مؤلفه‌های ضبط شده در پایگاه داده در مرحله قبل و سپس تصمیم‌گیری در مورد نحوه انتخاب پایگاه قاعده برای استفاده در استباط فازی در مراحل بعدی، است. ابتدا، الگوریتم حالت سامانه را از میان پنج حالت از پیش تعریف شده انتخاب می‌کند. این پنج حالت شامل تامین اضافی منابع در حالت شدید، تامین اضافی منابع، حالت عادی، کمبود در منابع و کمبود بیش از اندازه در منابع، تعریف شده‌اند. پس از تشخیص حالت سامانه، پایگاه قاعده متناسب با شرایط حال حاضر انتخاب می‌شود. انتخاب پایگاه قاعده با استفاده از دو مؤلفه مورد بررسی، MCU و SLA، به‌صورت زیر انجام می‌شود. حالت‌هایExOver

ExUnder، Under، over، و Normal بـــه ترتیـــب متناظر با حالت‌هـــای تامین اضافی منابع در حالت شـــدید، تامین اضافـــی منابع، کمبود در منابع، کمبود بیـــش از اندازه در منابع و حالت عادی هستند.

$$RuleBase = \begin{cases} ExOver\ SLA = 0\ and\ MCU \leq 15 \\ Over\ SLA = 0\ and\ 15 < MCU \leq 30 \\ Under\ 50 \leq SLA \leq 75\ and\ 30 < MCU \leq 60 \\ ExUnder\ 75 < SLA\ and\ 70 \leq MCU \\ Normal\ else \end{cases} \quad (۳)$$

در ادامـــه الگوریتم ایـــن مرحله از روش پیشـــنهادی مشاهده می‌شود

الگوریتم ۲: شبه کد الگوریتم مرحله Orientation

شبه کد الگوریتم مرحله Orientation

Algorithm: Orientation Phase
Input: SLA, MCU
Output: RuleBaseIndex
RuleBaseIndex = 3
If SLA = 0
If MCU <= 15
RuleBaseIndex = 1
End if
Else
RuleBaseIndex = 2
End else
End if
Else
If 30<= MCU <= 60 or 50 <= SLA <=75
RuleBaseIndex = 4
End if
Else if 70 <= MCU or 75 < SLA
RuleBaseIndex = 5
End else if
End else
Return RuleBaseIndex

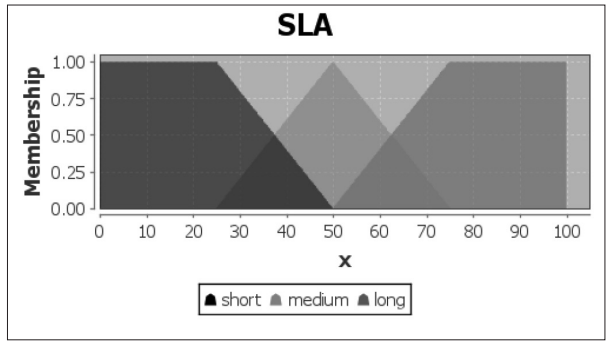
۴-۲-۳- مرحله Decision

پس از انتخاب پایگاه قاعده متناســـب با شرایط محیط رایانـــش ابری در مرحله قبل و جمع‌آوری مؤلفه‌های موثر در سیســـتم، در مرحله مشـــاهده، الگوریتم پیشنهادی در این مرحله به اســـتنباط فازی برای به‌دست آوردن بهترین عملیات مقیاس‌پذیـــری می‌پردازد. پـــس از دریافت مقدار مؤلفه‌ها، اولین مرحله، فازی‌سازی دو مؤلفه MCU و SLA اســـت. برای فازی‌سازی مؤلفه‌ها نیاز است تا اصطلاحات فازی هر کدام و تابع فازی‌ســـازی آن‌ها تعیین شود. نحوه فازی‌ســـازی هر دو متغیر مطابق با شکل‌های ٤ تا ۹ است. همان‌طور که مشخص است، برای فازی‌سازی مؤلفه‌ها از تابع فازی‌ســـازی مثلثی استفاده شده است و فازی‌سازی متناسب با پایگاه قاعده و شرایط محیط، متفاوت است.پس از انجام عملیات فازی سازی متغیرها، از پایگاه قاعده برای ترکیب قوانین موجود در پایگاه قاعده با متغیرهای ورودی فازی ســـازی شده، استفاده می‌شود. برای ترکیب ورودی با قوانین باید قدرت و مقدار درســـتی هـــر قانون با توجه به مقادیر ورودی به‌دســـت آید که در این مســـئله، باتوجه به این‌که از منطق استنباط فازی به روش ممدانی استفاده شده است، از روش کمینه برای تعیین درجه درستی قوانین اســـتفاده شده است. پس از محاسبه مقادیر خروجی برای هر قانون باید این مقادیر با یکدیگر ترکیب و جمع شـــوند و برای این کار از بیشـــینه استفاده می‌شود. پس از ترکیب خروجی‌ها، نیاز به خنثی‌ســـازی متغیر خروجی که همان مؤلفه مقیاس‌گذار می‌باشد، وجود دارد. با توجه به استفاده از روش ممدانـــی در این مرحلـــه، خروجی، مجموعه‌ای از ســـطوح زیر نمودارهای فازی بـــرای متغیر مقیاس‌پذیری در قوانیـــن می‌باشـــد، در نتیجه اســـتفاده از روش مرکز ثقل، منطقی‌ترین روش برای خنثی‌ســـازی متغیر خروجی می‌باشد. در مرحلهٔ آخر برای تصمیم‌گیری نهایی در رابطه بـــا نحوه مقیاس پذیـــری، با توجه به مقـــدار عددی متغیر مقیاس‌پذیـــری(scale) و فرمول ٤، دســـتور مقیاس‌پذیری در پایگاه داده مشـــترک ذخیره می‌شود. متغیر Increase به معنای افزایش در منابع، Decrease به معنای کاهش در منابع و NoChange به معنای عدم تغییر در منابع است.

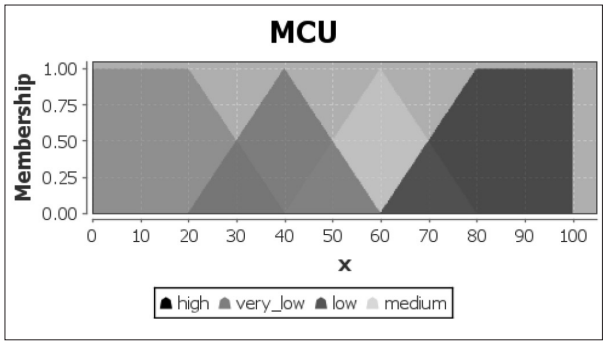
$$scale = \begin{cases} Increase\ scale \geq 60 \\ NoChange\ 50 < scale < 60 \\ Decrease\ scale \leq 50 \end{cases} \quad (٤)$$

الگوریتم ۳: شبه کد الگوریتم مرحله Decision

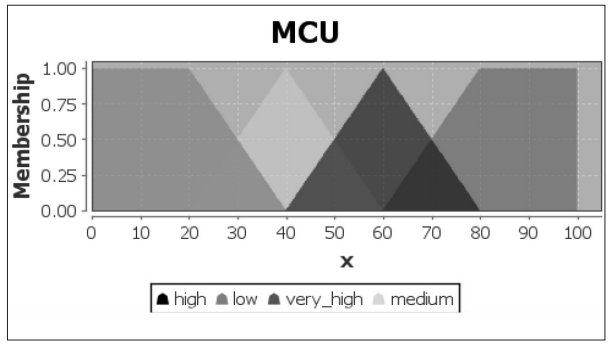
شبه کد الگوریتم مرحله Decision
Algorithm: Decision Phase Input: RuleBaseIndex, SLA, MCU Output: scale CurrentRuleBase = loadFromRuleBaseCollection(RuleBas (eIndex (FuzzyInference f = new FuzzyInference(CurrentRuleBase (f.setInputVariable(SLA, MCU ()scale = f.inference Return scale



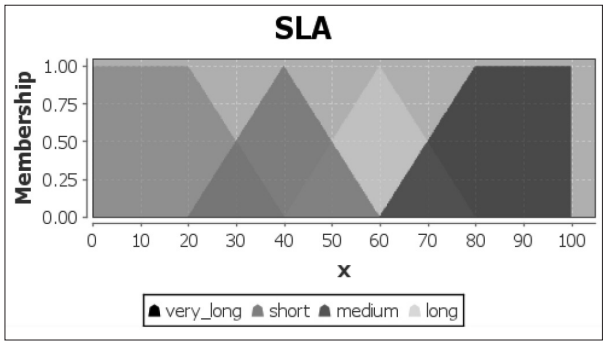
شکل۴: فازی سازی مؤلفه SLA



شکل ۵: فازی سازی مؤلفه MCU



شکل ۶: فازی سازی مؤلفه MCU در حالت کمبود منابع شدید



شکل ۷: فازی سازی مؤلفه SLA در حالت کمبود منابع شدید

۴-۲-۴- مرحله Action

قدم آخر در الگوریتم پیشنهادی، پیاده‌سازی دستور صادر شـــده در مرحله قبل توسط مؤلفه مقیاس‌پذیری خودکار اســـت. ایـــن مرحله نیز یکی دیگـــر از مراحلی اســـت که وظیفه پیاده‌سازی قســـمتی از پویایی اضافه شـــده به روش‌های فازی را بر عهـــده دارد. این مرحله در صورت صدور دستور افزایش در منابع موجود در مرکز داده، درخواست ایجاد ماشین‌مجازی را به واسط ارسال می‌کند. همان‌طور که گفته شد، تعداد ماشین‌های مجازی اضافه شـــده وابسته به شرایط محیط و پایگاه قاعده انتخاب شـــده، هســـتند. اگر پایگاه قاعده انتخاب شده متناســـب با شـــرایط کمبود شـــدید منابع باشد، الگوریتم در این مرحله درخواســـت اضافه شـــدن دو ماشین‌مجازی به مرکز داده موجود را به واسط ارسال می‌کند و در حالت‌های دیگر درخواســـت اضافه شـــدن تنها یک ماشـــین‌مجازی را ارســـال می‌کند. در دستور کاهش نیـــز پویایی در قدم مقیاس‌پذیـــری وجود دارد به این صورت که اگر پایگاه قاعده متناســـب با شرایط تامین اضافی منابع بیش از حد باشد، دو ماشین مجازی حذف می‌شوند و در حالت‌های دیگر یک ماشین‌مجازی حذف می‌شود.

الگوریتم۴: شبه کد الگوریتم مرحله Action

شبه کد الگوریتم مرحله Action

```
Algorithm: Action Phase
Input: scale, RuleBaseIndex, List ExecutingVms
Output: scaling action
step = 1
If RuleBaseIndex = 5 or RuleBaseIndex = 1
step = 2
End if
If 60 <= scale
For i = 1 to step
()AddNewVM
End for
End if
Else if scale <= 50
;(idleVm = ExecutingVms.get(0
|For i = 1 to |ExecutingVms
If ExecutingVms.get(i).IsIdle
(idleVm = ExecutingVms.get(i
End if
End for
(RemoveVM(idleVm
End Else id
```

۵- ارزیابی رهیافت پیشنهادی

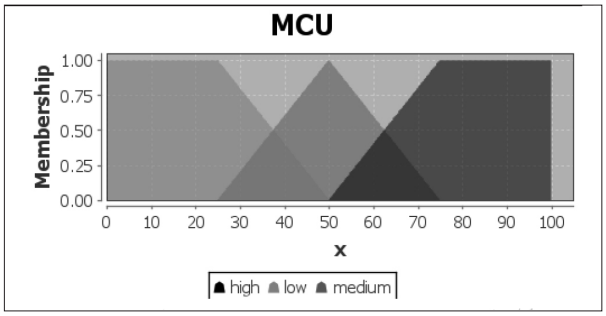
باتوجه به مشکلات پیاده‌سازی الگوریتم در محیط‌های واقعی رایانش ابری، در این کار پژوهشی از شبیه‌سازی برای ارزیابی روش پیشنهادی استفاده شده است. ارزیابی الگوریتم در این بخش در سه مرحله انجام می‌شود. مرحله اول تحلیل حساسیت رهیافت پیشنهادی است، به این صورت که با تغییر مقادیر مختلف در الگوریتم، نتیجه و عملکرد رهیافت پیشنهادی چگونه تغییر می‌کند. مرحله دوم، بررسی تاثیر مؤلفه‌های انتخابی ارزیابی است و مرحله سوم مقایسه رهیافت پیشنهادی با روش‌های پیشنهادی دیگر خواهد بود. در ادامه، ابتدا به معرفی ابزار شبیه‌ساز مورد استفاده و محیط آزمایش می‌پردازیم. سپس به ارزیابی رهیافت پیشنهادی مقیاس‌پذیری خودکار در سه مرحله گفته شده پرداخته می‌شود.

الگوریتم ۵: شبه کد الگوریتم مقیاس‌پذیری خودکار پیشنهادی

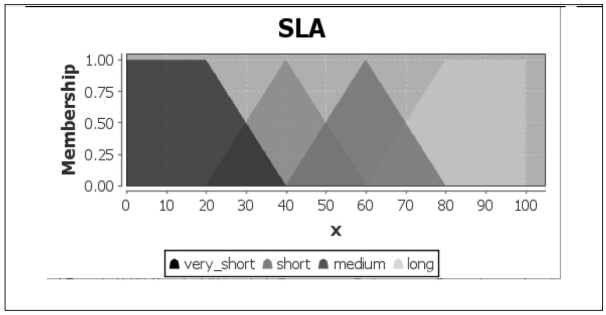
شبه کد الگوریتم مقیاس‌پذیری خودکار پیشنهادی

Algorithm: ProposedAutoscalingApproach
Input: MCU, SLA
Output: scale
While (TRUE) do
Begin While
|For i = 1 to |requests
(MainQueue.Add(requesti
End for
(Sorted_Req = Priority(MainQueue,DeadLine
(Observation.ADD(DB.Request,sorted_Req
(Observation.ADD(DB.VM,VMs
(requesti,VMs]=Orientation.Read(DB.Request,DB.VM]
(Rulebase = choose_rule_base(requests,VMs
(Orientation.ADD(DB.Rulebase,Rulebase
requests,VMs,rulebase] = Decision.Read(DB.Requests,DB.]
(VMs,DB.Rulebase
Startegy = Scale_Startegy_Fuzzy_
(logic(Requests,VMs,rulebase
(Decision.ADD(DB.Strategy,Strategy
(Req,command] = Action.Read(DB.Request,DB.Strategy]
(Action(Req,command
End While

طراحی و پیاده‌سازی رهیافت پیشنهادی مبتنی بر استنباط فازی با استفاده از زبان جاوا و در محیط شبیه‌سازی Cloud-Sim Plus انجام شده است. محیط انجام آزمایش شامل یک

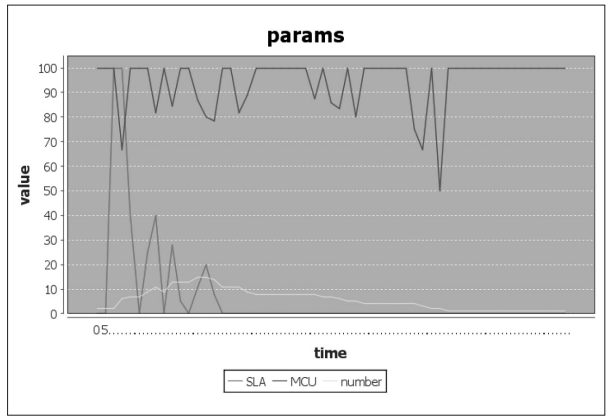


شکل ۸: فازی سازی مؤلفه MCU در حالت تامین اضافی منابع شدید



شکل ۹: فازی‌سازی مؤلفه SLA در حالت تامین اضافی منابع شدید

مرکز داده و تعدادی میزبان یا ماشین‌فیزیکی است. هر ماشین‌فیزیکی دارای منابع محاسباتی بوده که در اختیار ماشین‌های مجازی برای اجرای درخواست کاربران قرار می‌گیرد. همچنین برای پیاده‌سازی منطق‌فازی در روش پیشنهادی، از کتابخانه jFuzzyLogic استفاده شده است. محیط انجام آزمایش شامل یک مرکز داده و تعدادی میزبان یا ماشین‌فیزیکی است. هر ماشین‌فیزیکی دارای منابع محاسباتی است که در اختیار ماشین‌های مجازی برای اجرای درخواست کاربران قرار می‌گیرد. در مسئله‌های مختلف، بار ورودی و درخواست‌های کاربران می‌تواند از الگوی خاصی پیروی کند و یا این‌که به‌صورت تصادفی باشد. در ارزیابی انجام شده، بار ورودی توسط کاربران دارای هیچ الگوی مشخصی نبوده و به‌صورت تصادفی، درخواست‌های کاربران به واسط ارسال می‌شود. هر درخواست دارای اندازه مشخصی است که نشان دهنده تعداد دستورات مورد نیاز برای انجام این درخواست است. در ارزیابی انجام شده، طول و اندازه درخواست‌های کاربران به‌صورت تصادفی انتخاب می‌شوند. نحوه استفاده از منابع محاسباتی ماشین‌مجازی اختصاص داده شده به درخواست برای اجرا، که در ارزیابی انجام شده یک درخواست به‌طور کامل از



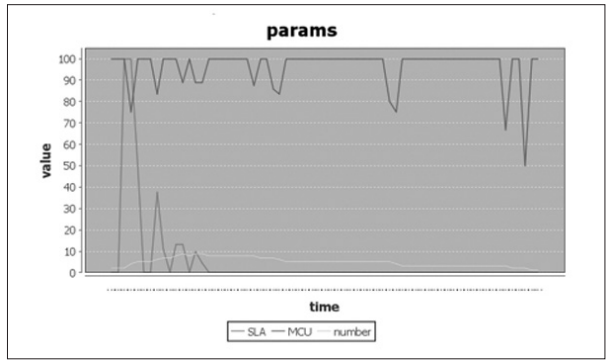
شکل ۱۰: خروجی رهیافت پیشنهادی

منابع محاسباتی ماشین‌مجازی استفاده می‌کند. در ارزیابی انجام شده، محیط رایانش ابری شامل تنها یک مرکز داده با تعدادی ماشـــین‌فیزیکی یا میزبان است. همچنین هر میزبان دارای تعدادی هســـته محاسباتی بوده که توانایی هر هسته اجرای ۱۰۰۰ میلیون دستور در ثانیه است. هرماشین‌فیزیکی تعدادی ماشـــین‌مجازی را میزبانی می‌کنـــد. وظیفه اجرای درخواست‌های کاربران بر عهده ماشین‌های مجازی است. هر ماشـــین مجـــازی منابع خود را از ماشـــین‌فیزیکی خود تامیـــن می‌کند و در نتیجه همانند ماشـــین‌های فیزیکی، یک ماشین‌مجازی دارای ویژگی‌های محاسباتی خواهد بود. هر ماشین‌مجازی دارای یک ظرفیت برای اجرای درخواست‌های کاربران بوده که ظرفیت تمامی ماشین‌های مجازی موجود در مرکز داده ۱۰۰۰میلیون دستور در ثانیه است. همچنین هر ماشین‌مجازی دارای دو هسته محاسباتی است.

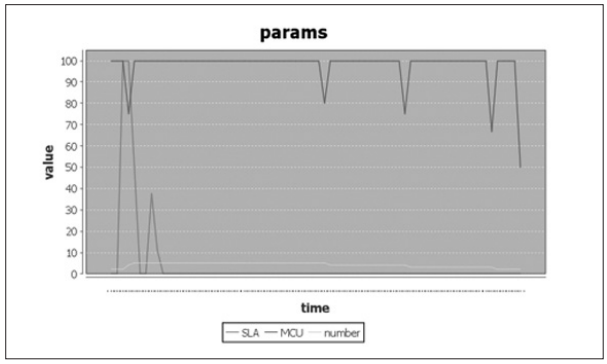
در روش پیشنهاد شده، الگوریتم سعی دارد با کم‌کردن مقدار پردازنده مصرف شـــده توســـط منابع محاسباتی و همچنین کمینه کردن مدت پاسخگویی به درخواست کاربران، به بهترین تصمیم در مسئله مقیاس‌پذیری در محیط رایانش ابری دســـت پیدا کند. همچنین، تعداد ماشـــین‌های مجازی استفاده شده برای انجام درخواست‌های کاربران نیز عامل موثری در ارزیابی کارایی روش پیشنهادی است. در نتیجه، سه مؤلفه SLA، MCU و تعداد ماشین‌های مجازی موجود در مرکز داده به‌عنوان شاخص‌های ارزیابی روش پیشنهادی در نظر گرفته شده‌اند. خروجی الگوریتم بر اساس این سه مؤلفه در شکل ۱۰ قابل مشاهده است.

الگوریتم در ابتدای اجرا از نظر مؤلفه تاخیر در پاسخگویی عملکرد مناســـبی ندارد و مقدار این مؤلفه تقریبا صد درصد می‌باشد اما با گذشـــت زمان و با اضافه کردن ماشین‌های مجازی بیشـــتر به مرکز داده، رفته رفته مقـــدار این تاخیر کاهش پیدا می‌کند. همچنین از نظر مؤلفه مقدار CPU مصرف شده، الگوریتم تلاش می‌کند تا مقدار CPU را در بیشینه خود، یعنی صد درصد، قرار ندهـــد و این موضوع نیز در نمودار قابل مشـــاهده است. از نظر تعداد ماشـــین‌های مجازی نیز الگوریتم پس از این‌که در مؤلفه تاخیر پاســـخگویی عملکرد نامناســـبی پیدا می‌کند، تصمیم به افزایش تعداد ماشین‌های مجازی می‌گیرد و پس از کاهش این تاخیر، تعداد ماشین‌های مجـــازی نیز کاهش پیدا می‌کند. نکته قابل توجه در خروجی رهیافت این اســـت که از آن‌جایی که روش پیشـــنهادی یک روش واکنشی می‌باشد، افزایش تعداد ماشین‌های مجازی با یک تاخیر انجام می‌شود به این معنی که هنگامی که الگوریتم با توجه به متغیر SLA متوجه تاخیر زیاد در پاســـخگویی به درخواست‌های کاربران می‌شود، دستور به افزایش در تعداد ماشین‌های مجازی می‌دهد اما همان‌طور که در نمودار قابل توجه اســـت این افزایش با مقداری تاخیر نسبت به نقطه اوج متغیر SLA انجام می‌شود.

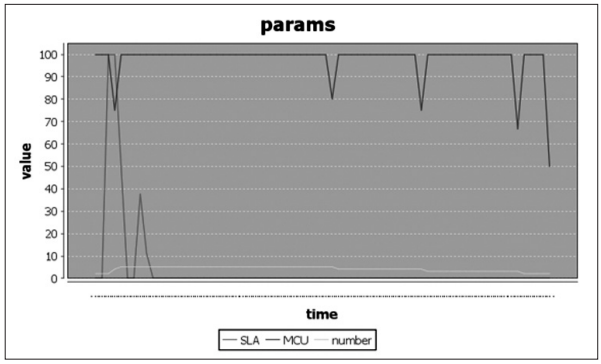
برای بررسی تاثیر پویایی‌های اضافه شده به الگوریتم، در دو مرحله پویایی اضافه شـــده بـــه الگوریتم در انتخاب پایگاه قاعده و در انتخاب قدم مقیاس‌پذیری را حذف می‌کنیم تا نتایج آن را بررســـی کنیم. برای ایجاد ایستایی در پایگاه قاعـــده، به جای انتخاب یک پایـــگاه قاعده از میان پنج پایگاه قاعده از پیش تعریف شـــده، پایگاه قاعده متناسب با شرایط عادی، که در تصمیم‌گیری هیچ سوگیری در افزایش یا کاهش منابع ندارد، به‌عنوان پایگاه قاعده استفاده می‌شود. نتایج این روش در شکل ۱۲ قابل مشاهده است. همان‌طور که مشخص اســـت، در روش ایســـتا، مؤلفه SLA نسبت به حالت پویا، در بازه‌های زمانی یکسان، دارای مقادیر بیشتری است. همچنین مقدار CPU مصرفی در مرکزداده نسبت به حالت پویا، مدت



شکل ۱۱: خروجی رهیافت پیشنهادی بدون قدم مقیاس‌پذیری پویا



شکل ۱۲: خروجی رهیافت پیشنهادی بدون پایگاه قاعده پویا



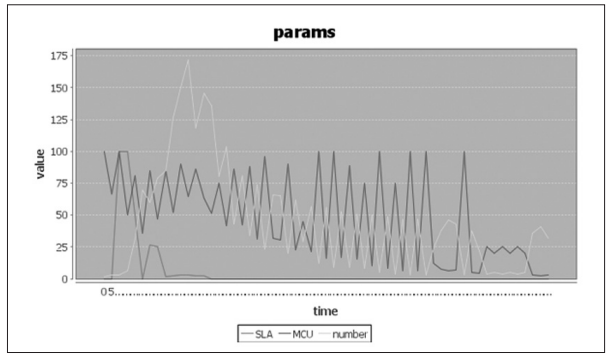
شکل ۱۳: خروجی روش مبتنی بر استنباط فازی

زمان بیشتری در حالت بیشینه خود، یعنی صد درصد خواهد بود. از نظر تعداد ماشین‌های مجازی نیز الگوریتم در حالت ایستا، تغییرات کم‌تری نسبت به حالت پویا دارد و به نظر می‌آید متناسب با شرایط تغییر نمی‌کند. با توجه به موارد گفته شده می‌توان نتیجه گرفت پویایی اضافه شده در مرحله انتخاب پایگاه قاعده، مؤلفه موثری در بهبود عملکرد رهیافت پیشنهادی است.

برای بررسی تاثیر قدم مقیاس‌پذیری پویا و ایجاد ایستایی در آن، به جای استفاده از تعداد قدم متغیر، در افزایش و یا کاهش منابع، تنها یک ماشین‌مجازی اضافه و یا کم می‌شود. نتایج خروجی این روش در شکل قابل مشاهده است. باتوجه به خروجی الگوریتم در حالت ایستای قدم‌های مقیاس‌پذیری، از نظر مقدار CPU مصرفی این حالت نیز مانند حالت قبل، مدت زمان بیشتری در حالت بیشینه خود است اما نسبت به حالت قبلی، این مدت زمان کم‌تر خواهد بود. این شرایط برای مؤلفه SLA نیز برقرار بوده به این معنی که حالت ایستای قدم‌های مقیاس‌پذیری نسبت به حالت پویا عملکرد نامناسب تری دارد. اما نسبت به حالت ایستای پایگاه قاعده، عملکرد آن بهتر است. علت عملکرد بهتر این روش نسبت به حالت ایستای پایگاه قاعده این است که تغییر ایجاد شده در این حالت بسیار جزئی بوده و تنها در سرعت عمل الگوریتم موثر است. از طرفی علت عملکرد بهتر روش پویا نسبت به این روش، به دلیل امکان ایجاد تغییرات گسترده‌تر در حالت‌هایی که وضعیت سیستم اضطراری است خواهد بود. باتوجه به موضوعات بررسی شده، تاثیر پویایی اضافه شده به الگوریتم در قدم‌های مقیاس‌پذیری واضح است. هر چند که این تاثیر به اندازه تاثیر مثبت پایگاه قاعده پویا نیست.

برای ارزیابی بهتر روش ارائه شده، رهیافت پیشنهادی با دو روش دیگر نیز مقایسه شده است. در قسمت اول به مقایسه این روش با یک روش مقیاس‌پذیری استنباط فازی می‌پردازیم. روش مقیاس‌پذیری خودکار مبتنی بر استنباط فازی، یک روش مشابه با رهیافت پیشنهادی است با این تفاوت که در این حالت الگوریتم تماماً ایستا بوده و تنها از یک پایگاه قاعده و یک قدم مقیاس‌پذیری در آن استفاده می‌شود.

برای ارزیابی روش مورد مقایسه، پایگاه قاعده همان پایگاه قاعده حالت عادی مورد استفاده در رهیافت پیشنهادی است و قدم مقیاس‌پذیری، یک ماشین‌مجازی است. خروجی روش مورد مقایسه و رهیافت پیشنهادی در شکل ۱۳ قابل مشاهده است. با مقایسه خروجی روش مورد مقایسه با روش پیشنهادی، مشخص است که مؤلفه MCU در روش استنباط فازی نسبت به رهیافت پیشنهادی، دارای مقادیر بیشتری بوده و در زمان‌های بیشتری در حالت بیشینهٔ است.



شکل ۱۴: خروجی روش مبتنی بر آستانه

مؤلفه تاخیر نیز در این روش نسبت به روش پیشنهادی، دارای مقادیر بیشتری است. از نظر تعداد ماشین‌های مجازی مورد استفاده نیز باتوجه به تغییرات بسیار کم در تعداد آن‌ها و با توجه به بالا بودن مقدار پردازنده مصرفی به نظر می‌آید که محیط رایانش ابری دچار کمبود در تامین منابع است. به‌طور کلی و با توجه به سه مؤلفه مورد بررسی برای ارزیابی الگوریتم، به نظر می‌آید که روش استنباط فازی نسبت به روش پیشنهاد شده دارای عملکرد ضعیف‌تری بوده و دچار کمبود در تامین منابع است. علت این موضوع این است که روش مورد مقایسه، هیچکدام از دو مرحله پویایی پیشنهاد شده، که هر دو باعث بهبود عملکرد الگوریتم می‌شوند، را ندارد و در نتیجه عملکرد ضعیف‌تری نسبت به رهیافت پیشنهادی از خود نشان می‌دهد.

روش دیگری که در ارزیابی مورد مقایسه قرار می‌گیرد یک روش مقیاس‌پذیری خودکار مبتنی بر آستانه و بر اساس روش ارائه شده توسط هانگ و همکارانشان[۹] است. این روش نیز در محیط شبیه‌سازی کاملاً مشابه با محیط ارزیابی رهیافت پیشنهادی، اجرا و مقایسه شده است. در این روش، در حالتی که مقدار میانگین CPU مصرفی از یک آستانه مشخص، که در اینجا ۷۰ درصد در نظر گرفته شده است، بیشتر باشد، درخواست اضافه شدن یک ماشین‌مجازی به واسط ارسال می‌شود. برای کاهش منابع نیز در صورتی که میانگین CPU مصرفی از ۳۰ درصد کم‌تر باشد و حداقل دو ماشین مجازی کاملاً بیکار باشند، دستور حذف یکی از ماشین‌های مجازی بیکار صادر می‌شود. نحوه عملکرد خروجی این الگوریتم در شکل ۱۴ نمایش داده شده‌است. همان‌طور که در شکل قابل مشاهده است، با توجه به این‌که این روش یک آستانه بالایی برای مقدار میانگین CPU مصرفی در نظر گرفته است، مقدار میانگین CPU در این روش نسبت به روش پیشنهادی کم‌تر بوده اما این روش در کاهش ماشین‌های مجازی بسیار محتاط عمل می‌کند. به همین دلیل تعداد ماشین‌های مجازی مورد استفاده در این روش نسبت به روش پیشنهادی بسیار بیشتر است و می‌توان گفت که در این روش، محیط رایانش دچار تامین بیش از اندازه منابع محاسباتی شده‌است. از نظر تاخیر در پاسخگویی به درخواست کاربران نیز با این‌که این روش نسبت به رهیافت پیشنهادی از تعداد بیشتری ماشین‌مجازی بهره می‌برد، اما تاخیر پاسخگویی در این روش تفاوت چندانی با رهیافت پیشنهادی ندارد.

با توجه به موارد گفته شده می‌توان نتیجه گرفت این روش از نظر مقدار CPU مورد استفاده در محیط رایانش ابری، نسبت به رهیافت پیشنهادی عملکرد بهتری دارد اما باید توجه داشت این موضوع با افزایش تعداد ماشین‌های مجازی و در نتیجه افزایش هزینه‌های تامین این ماشین‌های مجازی همراه است و از طرفی، کم شدن مقدار مصرف CPU تاثیری چندانی در کاهش تاخیر پاسخگویی به درخواست کاربران نداشته است. به بیانی دیگر می‌توان گفت در این روش محیط رایانش ابری دچار تامین بیش از اندازه منابع محاسباتی شده است.

۶- نتیجه‌گیری و کارهای آینده

در رهیافت پیشنهاد شده تلاش بر این بوده است که با اضافه کردن پویایی به روش‌های مقیاس‌پذیری خودکار مبتنی بر استنباط فازی، عملکرد بهتری در زمینه مقیاس‌پذیری حاصل شود. برای ایجاد پویایی در الگوریتم تلاش شده است که ابتدا شرایط محیط تشخیص داده شود به این معنی که محیط رایانش ابری در حال حاضر در حالت کمبود در تامین منابع یا در حالت تامین بیش از حد منابع است. هم‌چنین اگر به‌طور مثال محیط در حالت کمبود در تامین منابع باشد،

این کمبود به چه صورت است. پس از مشخص شدن حالت سیســـتم، الگوریتم برای اســـتنباط فازی از پایگاه قاعده‌ای استفاده می‌کند که تصمیم‌گیری‌های آن متناسب با شرایط به وجود آمده در محیط رایانش ابری است. همچنین، الگوریتم در هنگام پیاده‌سازی تصمیم مقیاس‌پذیری، با توجه به این که محیط رایانش ابری در حالت شـــدید یا حالت عادی باشد (چـــه در کمبود چه در تامین بیش از حد)، تصمیم به انتخاب قـــدم مقیاس‌پذیری می‌گیرد. در حالت‌های شـــدید، الگوریتم بـــرای مقیاس‌پذیری از قدم بزرگتری نســـبت به حالت‌های عادی اســـتفاده می‌کند. همان‌طور که در بخش ارزیابی گفته شد، نوآوری معرفی شده در رهیافت و پویایی اضافه شده به روش اســـتنباط فازی، در عملکرد رهیافت بســـیار موثر است و تصمیم‌گیری‌های بهینه مناسب و متناسب با محیط رایانش ابری در حال حاضر را عملی می‌ســـازد. باتوجه به این که روش پیشـــنهادی یک روش مقیاس‌پذیری مبتنی بر استنباط فازی اســـت، با تغییرات در مؤلفه‌ها و ورودی‌های سیســـتم، می‌توان عملکرد کلی را بهبود داد. به‌طور مثال در رهیافت پیشنهادی برای بررسی و شناخت شرایط محیط از دو مؤلفه MCU و SLA اســـتفاده شـــده است که برای بهبود عملکـــرد الگوریتم می‌توان مؤلفه‌های بیشـــتری مانند مقدار حافظه، پهنای‌باند شبکه و ویژگی زمانی حال حاضر(به‌طور مثال روزهای آخر هفته یا ساعات اوج درخواست کاربران) را برای بررســـی شرایط سیســـتم در نظر گرفت. همچنین در روش پیشـــنهاد شده، خروجی سیســـتم استنباط فازی یک متغیر اســـت که نشان دهنده نحوه مقیاس‌پذیری خواهد بود. برای بهبود بیشتر عملکرد، می‌توان خروجی سیستم را میان دو آستانه بالایی و پایینی برای یک مؤلفه یا چند مؤلفه مورد بررســـی، مانند CPU مصرفـــی، در نظر گرفت و پس از تعیین آســـتانه‌ها، در این روش مانند روش‌های مبتنی بر آستانه عمل کرد. با تغییر در سیستم استنباط فازی رهیافت پیشنهادی، مانند تغییر توابع عضویت و اصطلاحات فازی، افزایش تعداد متغیرهای ورودی سیستم‌استنباط فازی، تغییر تعداد پایگاه‌های قاعده و یا افزایش و تغییر قوانین موجود در پایگاه‌های قواعد موجود نیز عملکرد روش ارائه شده می‌تواند بهبود یابد.

مراجع

[1]R. Buyya, M. A. Rodriguez, A. N. Toosi, and J. Park, "Cost-efficient orchestration of containers in clouds: A vision, architectural elements, and future directions," 2018, vol. 1108: IOP Publishing, 1 ed., p. 012001.

[2]V. Persico, D. Grimaldi, A. Pescape, A. Salvi, and S. Santini, "A fuzzy approach based on heterogeneous metrics for scaling out public clouds," IEEE Transactions on Parallel and Distributed Systems, vol. 28, no. 8, pp. 2117-2130, 2017.

[3]F. Lombardi, A. Muti, L. Aniello, R. Baldoni, S. Bonomi, and L. Querzoni, "PASCAL: An architecture for proactive auto-scaling of distributed services," Future Generation Computer Systems, vol. 98, pp. 342-361, 2019.

[4]P. Singh, P. Gupta, K. Jyoti, and A. Nayyar, "Research on auto-scaling of web applications in cloud: survey, trends and future directions," Scalable Computing: Practice and Experience, vol. 20, no. 2, pp. 399-432, 2019.

[5]C. Qu, R. N. Calheiros, and R. Buyya, "A reliable and cost-efficient auto-scaling system for web applications using heterogeneous spot instances," Journal of Network and Computer Applications, vol. 65, pp. 167-180, 2016.

[6]«Cloud Autoscaling Explained." https://www.densify.com/articles/autoscaling», Last visite: 11/25/2021.

[7]«Mamdani and Sugeno Fuzzy Inference Systems." https://de.mathworks.com/help/fuzzy/types-of-fuzzy-inference-systems.html (accessed 24 October 2021).

[8]M. Ghobaei-Arani, M. Rezaei, and A. Souri, "An auto-scaling mechanism for cloud-based multimedia storage systems: a fuzzy-based elastic controller," Multimedia Tools and Applications, pp. 1-23, 2021.

[9]C.L. Hung, Y.C. Hu, and K.C. Li, "Auto-scaling model for cloud computing system," International Journal of Hybrid Information Technology, vol. 5, no. 2, pp. 181-186, 2012.

[10]K. Cetinski and M. B. Juric, "AME-WPC: Advanced model for efficient workload prediction in the cloud," Journal of Network and Computer Applications, vol. 55, pp. 191-201, 2015.

[11]C.W. Tseng, M.S. Tsai, Y.T. Yang, and L.D. Chou, "A Rapid Auto-Scaling Mechanism in Cloud Computing Environmen", Computing, vol.21, 2021.

[12]Y. Wei, D. Kudenko, S. Liu, L. Pan, L. Wu, and X. Meng, "A reinforcement learning based auto-scaling approach for SaaS providers in dynamic cloud environment," Mathematical Problems in Engineering, vol. 2019, 2019.

[13]H. Arabnejad, C. Pahl, P. Jamshidi, and G. Estrada, "A comparison of reinforcement learning techniques for fuzzy cloud auto-scaling," in Proceedings of the 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID), Madrid, Spain, 14-17 May 2017, pp. 64-73.

[14]W.H. Liao, S.C. Kuai, and Y.R. Leau, "Auto-scaling strategy for amazon web services in cloud computing," in Proceedings of the IEEE International Conference on Smart City/SocialCom/SustainCom (SmartCity), Chengdu, China, 19-21 December 2015, pp. 1059-1064.