DOR :20.1001.1.2538161.1399.5.4.5.2

تاریخ دریافت مقاله: ۹۷/۰۵/۰۱
تاریخ پذیرش مقاله: ۹۹/۰۳/۲۴

ارائه رویکردی نوین مبتنی بر یادگیری بیزی برای تامین کارآمد منابع برنامه‌های کاربردی در محیط‌های ابری

مصطفی قبائی آرانی*
گروه مهندسی کامپیوتر، واحد قم، دانشگاه آزاد اسلامی، قم، ایران
پست الکترونیکی: m.ghobaei@qom-iau.ac.ir

سمانه کربلایی مهدی
گروه مهندسی کامپیوتر، واحد محلات، دانشگاه آزاد اسلامی، مرکزی، ایران
پست الکترونیکی: samaneh.karbalaie@gmail.com

چکیده

تامین منابع برنامه‌های کاربردی چندلایه در محیط‌های ابری با یکسری چالش‌ها روبروست که شامل کسر تامین، اضافه تامین و نوسان است. در این مقاله برای مرتفع کردن چالش‌های مطرح‌شده همچنین بهینه‌سازی زمان‌بندی درخواست‌های کاربران و پاسخ‌دهی به آن‌ها، تقلیل مشکل تخطی از سرویس، به ارائه رویکردی بهبود یافته مبتنی بر یادگیری ماشین با بهره‌گیری از حلقه "MAPE"[1] می‌پردازیم. در مرحلهٔ تحلیل این حلقه از مدل رگرسیون خطی(LRM) و در مرحلهٔ برنامه‌ریزی، از روش مبتنی بر نظریهٔ بیز به منظور بهینه نمودن اقدامات استفاده شده است. سپس رویکرد پیشنهادی خود را تحت بار کاری واقعی FIFA با روش‌های Stat-RA و DPM-RA مقایسه نموده‌ایم که راهکار ارائه شده نسبت به راه‌کارهای پیشین، منجر به افزایش تعداد ماشین‌های مجازی به میزان ۱۰ درصد با بهبود نرخ مقیاس‌بندی، کاهش ۸ درصدی میانگین بهره‌وری، کاهش ۳ درصدی زمان پاسخ‌دهی، در نتیجه کاهش ۵ درصدی هزینه تمام شده و افزایش ۱ درصدی سود حاصل شده است.

واژه‌های کلیدی: رایانش ابری، برنامه‌های چندلایه، مقیاس‌پذیری، رگرسیون خطی[2]، نظریهٔ بیز[3].

مقدمه

گسترش اینترنت و فراگیری استفاده از آن، تاثیر شگرفی بر زندگی بشر داشته و خدماتی که از طریق اینترنت می‌توان ارائه داد دائما در حال تغییر است.

با پیشرفت فناوری اطلاعات نیاز به انجام کارهای محاسباتی در همه جا و همه مکان به وجود آمده است. همچنین نیاز به این هست که افراد بتوانند کارهای محاسباتی سنگین خود را بدون داشتن سخت افزار و نرم‌افزارهای گران، از طریق خدماتی انجام دهند. رایانش ابری آخرین پاسخ فناوری به این نیازها بوده است [۱].

* نویسندهٔ مسئول
1- Monitor, Analyze, Plan, Execute_Knowledge (MAPE-K)
2- Linear Regression
3- Bayes theorem

رایانش ابری، باید به‌صورت خودکار منابع مورد نیاز به کاربران را تخصیص دهد و اگر کاربری مقداری بیشتر از منابع را خواستار باشد، باید آن منابع بدون آن که سرویس کاربر با مشکل مواجه شود به مشتری تخصیص یابد و در صورتی که منابع مورد استفاده بیش از حد مورد نیاز باشد باید منابع اضافی به‌صورت موقت خاموش شده تا در صورت نیاز مجددا مورد استفاده قرار گیرند. این مفاهیم در دنیای رایانش ابری تحت عنوان «مقیاس‌پذیری»[4] و یا گاهی اوقات «مقیاس‌بندی»[5] شناخته می‌شود. تعاریف گوناگونی برای مقیاس‌پذیری وجود دارد که به چند مورد آن اشاره می‌کنیم [۲].

از آنجائی‌که برنامه‌های کاربردی، مخصوصا برنامه‌های کاربردی تحت وب، دارای الگوهای حجم کاری منظمی نیستند، بنابراین عملیات مقیاس‌بندی (افزایش یا کاهش مقیاس) باید صورت بیدرنگ و با حداقل دخالت انسان انجام گیرد تا منابع در اسرع وقت برای برنامه‌های کاربردی تامین شود. به این نوع مقیاس‌بندی کردن منابع، که با کمترین دخالت انسان و به‌صورت خودکار انجام می‌شود، مقیاس‌بندی خودکار[6] گفته می‌شود. مقیاس‌بندی خودکار، راهبرد اصلی در راستای تامین پویای منابع برای برنامه‌های کاربردی ابری است. فرایند مقیاس‌بندی، توسط عاملی به نام Auto-Scaler انجام می‌شود [۳]. Auto-Scaler، مسئول تصمیم‌گیری در مورد اعمال مقیاس‌بندی، بدون تعامل و مداخله یک مدیر انسانی است. هدف Auto-Scaler، تطبیق پویای منابع انتساب داده شده به برنامه‌های کاربردی بر اساس بارکاری ورودی است و باید توازنی بین برآورده کردن اهداف SLA و حداقل‌سازی هزینه، برقرار نماید. هر Auto-Scaler، با چالش‌های زیر روبرو خواهد شد:

کسر تامین[7]: در این مورد، برنامه کاربردی، منابع کافی برای پردازش همه درخواست‌های ورودی با توجه به محدودیت‌های زمانی اعمال شده توسط SLA، را ندارد. در مورد ترافیک‌های ناگهانی، کسر تامین منابع برای برنامه کاربردی ممکن است منجر به تخطی از SLA شود.

اضافه تامین[8]: در این مورد، برنامه کاربردی منابع بیشتر از نیاز برای بر آورده کردن توافقات SLA دارد. اگر چه این حالت، برای برآورده کردن اهداف SLA مناسب است امّا به دلیل وجود منابع بیکار، مشتری باید هزینه‌های غیر ضروری پرداخت نماید.

نوسان[9]: این مورد، ترکیبی از دو پدیده نامطلوب کسر تامین و اضافه تامین است. در محیط ابری، وقتی با وجود اضافه تامین منابع برای یک برنامه کاربردی، تخطی از SLA رخ دهد، به این پدیده، نوسان گفته می‌شود شکل(۱).

در این مقاله به ارایه رویکردی نوین برای تامین کارامد منابع برنامه‌های کاربردی در محیط ابری پرداخته‌ایم. روش پیشنهادی برای تامین منابع بر روی ساختار MAPE که شامل چهار مرحلهٔ پایش، تحلیل، برنامه‌ریزی و اجرا است [۵] ارایه شده که در مرحلهٔ تحلیل از «مدل رگرسیون خطی(LRM)»[10] و در مرحلهٔ برنامه‌ریزی از «نظریهٔ بیز»[11] استفاده کرده‌ایم.

سهم‌های عمده این تحقیق عبارتند از:

- ارائه چارچوبی مبتنی بر حلقه MAPE برای تامین خودکار منابع برای برنامه‌های کاربردی چند لایه
- ارائه فرمول بندی برای مسئله تامین خودکار منابع برای برنامه‌های کاربردی چند لایه
- شبیه‌سازی رویکرد پیشنهادی تحت بارکاری واقعی

مسئله‌ای که در خور توجه است، تامین منابع ابری به‌صورت بهینه می‌باشد تا تمام کاربران بتوانند از هر منبعی بر اساس میزان مورد نیازشان استفاده کنند. در صورتی که این منابع به‌صورت بهینه برای کاربران تامین و مدیریت شوند شاهد تغییرات قابل توجهی در تعداد ماشین‌های مجازی، میانگین بهره‌وری، زمان پاسخ‌دهی،

4- Scalability
5- Scaling
6- Auto-Scaling
7- Under-Provisioning
8- Over-Provisioning
9- Oscillation
10- Linear regression model
11- Queueing theory

هزینه و سود حاصله خواهیم بود.

ادامه این مقاله بدین صورت ســازماندهی شده است: بخش دوم به کارهــای مربوطه اختصاص دارد، در بخش سوم رویکرد پیشنهادی را که شامل چارچوپ پیشنهادی و الگوریتم پیشنهادی است را تشریح خواهیم نمود و در بخش چهارم، به شبیه‌سازی و ارزیابی کارایی روش پیشنهادی خواهیــم پرداخــت و نهایتا بخش پنجم بــه نتیجه‌گیری و پیشنهادها اختصاص دارد.

۲-کارهای مربوطه

تحقیقات متنوع و زیادی در ســال‌های اخیر در رابطه با تامین منابع برای برنامه‌های کاربردی چندلایه صورت گرفته است، بیشــتر کارهای موجود براساس تکنیک‌هایی اســت که در خصوص مدیریت منابع اســتفاده شده است که هر یک از روش‌های موجود مزایا و معایبی دارد که بر تصمیم‌گیری در مورد انتخاب تکنیک، تاثیر خواهد گذاشت. در این بخــش مروری بر تحقیقات اخیــر در زمینه تامین منابع برنامه‌های کاربردی چندلایه خواهیم داشت.

جینگ‌بــی و همکارانــش در ســال ۲۰۱۳، به مســئله تخصیص منابع مجازی‌شــده برای مراکــز داده ابری در مقیاس بزرگ « CDC»[۱۲]ها [٦] پرداختند. آن‌ها یک معماری مدیریت خودمختار را بر اســاس سازوکار مجازی‌سازی برای برنامه‌های وب چندلایه در CDCها به منظور افزایش تخصیص منابع ارائه دادند که این معماری می‌تواند اجرای موثر محیط مجازی شــدهٔ خدمات کاربــردی «VASE»[۱۳] و نیازمندی‌های کارخواه‌های SLA را تضمین کند سپس یک مدل صف ترکیبــی انعطاف‌پذیر ارائه دادند و به‌طور کامل به زمان پاسخ، توان عملیاتی، هزینه و درآمد مشخص شده SLA پرداختند. پس از آن یک مسئله بهینه‌سازی غیرخطی محدود شده را بیان کردند.

از الگوریتم مکاشفه‌ای با هدف حداکثر کردن سود کل «CIP»[۱۴]ها جهت حل مسئله استفاده کردند که این الگوریتم بر اســاس تغییرات پویای حجم کار منابع مجازی شده را تخصیص می‌دهد. آن‌ها تاثیر راهبرد تخصیص پویای منبع را بر اساس ماشین‌های مجازی «DVM-RA»[۱۵] با دو راهبرد «Stat-RA»[۱٦] و «DPM-RA»[۱۷] مقایسه کردند.

تامین پویای منبع یک تکنیک چالشی برای نیازمندی‌های SLA از برنامه‌هــای کاربــردی چندلایه در ابــر مبتنی بر مجازی‌سازی است. در همین راستا Heng Wu و همکارانش در ســال ۲۰۱۳[۷] به ارائه یک روش آگاه از سود با نظریهٔ کنترل بازخورد پرداختند که طبق روش ارائه شــده، آن‌ها هزینه تامین منبع در مقایسه با روش هزینه آشکار تا ۳۰٪ کاهش می‌یابد و به‌طور موثر می‌تواند موارد نقض SLA را در مقایسه با روش آگاه از هزینه کاهش دهد.

تغییــرات درخواســت‌ها از ســوی مســتاجران و استفاده‌کنندگان متفاوت است بنابراین انعطاف‌پذیری برای آن‌ها افزایش می‌یابد. آن‌ها یک مدل تخریب کارایی را مبتنی بر یادگیری ماشین جهت محاسبه هزینه انتقال ایجاد کردند سپس یک تابع ســود و هزینه را برای همه پیکربندی‌های مجدد ارائه کردند. در نتیجه حداقل هزینه با بیشترین سود را به‌دست آوردند.

Bhuvan Urgaonkar و همکاران[۸]، تکنیکی پویا برای برنامه‌های چندلایه اینترنت را ارائه نمودند که شــامل یک مدل صــف انعطاف‌پذیر برای تعییــن چگونگی تخصیص منابع به هر لایه برنامــه کاربردی و ترکیبی از روش‌های پیش‌بینی و واکنش‌پذیر جهت تعیین زمان به منظور تامین منابع در مقیاس‌های زمانی کوچک و بزرگ است. با توجه به روشــی که آن‌ها ارائه دادند اهداف زمان پاســخ حفظ می‌شــود و سربار کاهش می‌یابد همچنین سربار تعویض کارسازها از چندین دقیقه به کمتر از یک ثانیه می‌رسد.

باتوجه به این که تغییرات حجم کار پویاست و تغییرات بارکاری در زمان‌های مختلف متفاوت است، بنابراین تکنیک پیشنهادی آن‌ها مناسب است. همچنین سازوکارهای تامین

12- Cloud Data Center
13- Virtualised Application Service Environment
14- Cloud Infrastructure Provider

15- Dynamically Resourse Allocation Strategy Based On Virtual Machine
16- Static Resource Allocation Strategy
17- Dynamically Resource Allocation Strategy Based On Physical Machine

پیش‌بینی و واکنشی به‌عنوان معماری مرکز داده بر اساس پایش ماشین مجازی ارائه شده است.

کیوان رحیم‌زاده و همکاران در سال ۲۰۱۵[۹]، یک مدل تحلیلی بر اساس"QN"[۱۸] به منظور تخمین معیارهای کیفیت سرویس برای برنامه‌های کاربردی چندلایه در یک مرکز داده مجازی ارائه نمودند. همچنین یک متدولوژی جهت اندازه‌گیری ویژگی‌های "VMTA"[۱۹] از درخواست‌ها بین لایه‌ها و معیارهای عملکرد برنامه‌های کاربردی ارائه نمودند سپس آزمایش‌های مختلفی را به منظور ارزیابی عملکرد تلفیقی لایه‌های VMTAها، در خصوص حجم کار تدریجی و انفجاری انجام دادند.

نتایج کارهای آن‌ها نشان می‌دهد که قابلیت و ویژگی‌های لایه‌ها مستقیما بر عملکرد کلی VMTAها تاثیر می‌گذارد و مغایرت درخواست‌ها برای داشتن منابع نرم‌افزار و سخت‌افزار موجب افزایش VM مسدود شده می‌گردد که سبب کاهش عملکرد VMTA خصوصا، در زمان حجم کار بالا می‌گردد. استفاده از این تکنیک موجب کاهش هزینه، امنیت سرویس‌های مجازی و تضمین اهداف SLA گردیده است.

مارتا بلتران در سال ۲۰۱۵[۱۰]، جهت تامین منابع برنامه‌های کاربردی چندلایه، تکنیکی را ارائه داده است که در آن تامین برنامه به‌صورت خودکار و مجازی صورت می‌پذیرد. تمام تصمیم‌گیری‌ها نیز به‌صورت کاملا پویا و بر اساس شرایط و بدون دخالت انسان انجام می‌شود. تکنیکی که توسط وی ارائه گردیده است، AutoMap نام دارد.

این تکنیک توانایی تطابق با SLA در هزینه‌های اضافی را با توجه به سربارها دارد. از مقیاس‌گذاری افقی و عمودی نیز استفاده شده است. این تکنیک برای تعیین مقدار منابع مجازی مورد نیاز در هر لایه از برنامه کاربردی، مصرف منابع را به حداقل می‌رساند و توانایی تعامل با انواع برنامه‌های کاربردی مختلف را دارد، وی از چارچوب CloudSim جهت پیاده‌سازی تکنیک پیشنهادی خود استفاده نمود.

Sukhpal Singh و همکاران در سال ۲۰۱۵ [۱۱]، به بررسی تامین منابع مناسب در حجم کارهای ابری وابسته به نیازمندی‌های QoS پرداختند. پارامترهای QoS بر اساس تکنیک تامین منبع به منظور تامین موثر منابع مورد نیاز است. آن‌ها به تحلیل و طبقه‌بندی بر اساس الگوهای رایج و تامین حجم کار ابر قبل از برنامه‌ریزی واقعی پرداختند. از آنجایی که پیچیدگی مدیریت منابع در ابرها روز به روز در حال افزایش است، یک تکنیک موثر به منظور مدیریت منابع نیاز است.

این تکنیک‌ها به شناسایی حجم کار ابر، تحلیل حجم کار ابر به منظور یافتن نیازمندی‌های QoS، طبقه‌بندی حجم کار و تامین منابع حجم کار بدون نقض SLA می‌پردازند. آن‌ها خوشه‌بندی حجم کار را از طریق k-means مبتنی بر الگوریتم خوشه‌بندی انجام دادند و از نرم‌افزار شبیه‌سازی CloudSim به منظور شبیه‌سازی استفاده کردند و در نهایت توانستند زمان صف‌بندی، زمان اجرا و هزینه را کاهش دهند و رضایت QoS و رضایت مشتری را تامین کنند.

رویکرد ارائه شده توسط سمانه کربلایی مهدی و مصطفی قبائی آرانی در [۱۲] دارای سه مرحله پایش، تحلیل و اجراست که در مرحله تحلیل از سیستم استنتاج عصبی–مرحلهٔ‌ی تطبیق‌پذیر استفاده می‌شود. این سیستم از قدرت آموزش شبکهٔ عصبی و مزیت زبانی سیستم‌های مرحلهٔ‌ی به منظور تحلیل فرآیندهای پیچیده به‌صورت بسیار قدرتمند عمل می‌کند. سپس رویکرد پیشنهادی را تحت بارهای کاری واقعی و مصنوعی با دو روش "ADRP_Fuzzy"[۲۰] و "ADRP_AL"[۲۱] مورد ارزیابی و مقایسه قرار داده‌اند. نتایج به‌دست آمده نشان می‌دهد که رویکرد پیشنهادی موجب کاهش زمان پاسخگویی و افزایش بهره‌وری می‌گردد.

18- Queuing Network
19-Virtualized Multi-Tier Application
20- Automatic Dynamic Resource Provisioning Fuzzy
21- Automatic Dynamic Resource Provisioning Automata Learning

ریحانهٔ خرسند و همکاران در سال ۲۰۱۸ [۱۳] تکنیکی جهت تأمین منابع ترکیبی برای برنامه‌های چندتایی ارائه داده‌اند که مبتنی بر ترکیبی از مفهوم محاسبات خودمختار، مدل پیش‌بینی رگرسیون بردار پشتیبانی "SVR" [۲۲] و رویکرد فرآیند سلسله مراتبی تحلیلی مرحلهٔی "FAHP" [۲۳] است چارچوبی که ارائه شده بر اساس حلقه کنترل MAPE-k برای تأمین منابع خودمختار برنامه‌های کاربردی چندلایه در محیط‌های محاسبات ابری است. اثربخشی این رویکرد پیشنهادی تحت بار کار واقعی و مصنوعی مورد بررسی شده و نتایج تجربی نشان می‌دهد که راه حل پیشنهادی از لحاظ اصولی فراتر است از ماشین‌های مجازی اختصاص یافته، زمان پاسخ و هزینه در مقایسه با دیگر رویکردهاست.

محمدصادق اصلانپور و همکاران در سال ۲۰۱۷ [۱٤] به ارائه مقیاس خودکار برنامه‌های وب در ابرها با توجه به هزینه رویکرد آگاهانه پرداختند که در این روش مجری حرفه‌ای اهمیت و اثربخشی این مرحله از چرخه کنترل را نشان می‌دهد. برخلاف معمول که جریان، راه حل پیشنهادی دستورات مقیاس پایین را از طریق انتخاب آگاهانه ماشین‌های مجازی مازاد اجرا می‌کند. علاوه بر این، با آن ویژگی‌های جدید، ماشین‌های مجازی مازاد در حداکثر مدت صورت‌حساب خود در قرنطینه نگهداشته‌اند تا هزینه به حداکثر برسد نتایج شبیه‌سازی نشان می‌دهد که مجری پیشنهادی ضمن افزایش کارایی، هزینه اجاره ماشین‌های مجازی را به میزان ۷ درصد کاهش داده است.

جینگ‌جینگ گو و همکاران در سال ۲۰۱۹ [۱۵] روندی را به منظور تأمین منابع تقاضا بر اساس تخمین بار و هزینه خدمات در محیط ابری ارائه نموده‌اند در این روش تقاضا برای منابع باید از قبل تخمین زده شود. برای این منظور، یک تخمین بارمدل بر اساس مدل "ARIMA" [۲٤] و شبکه عصبی "BP" [۲۵] ارائه شده است. این مدل می‌تواند بار را تخمین بزند و با توجه به داده‌های پیشینه، خطای تخمین را کاهش می‌دهد. قبل از انتشار منابع گره، کاربر برای اطمینان از عدم گم شدن اطلاعات کاربر، گره‌ها باید به گره‌های دیگر منتقل شوند. در این روش هنگام انتخاب هدف مهاجرت، سه معیار تعادل بار، زمان مهاجرت مصرف و هزینه مهاجرت خوشه در نظر گرفته شده است. با مقایسه نتایج تجربی، روش‌های پیشنهادی می‌توانند به‌طور موثر هزینه‌های خدمات را کاهش داده و خوشه را در حالت تعادل بار قرار دهد.

مطالعه تحقیقات فعلی نشان می‌دهد که کسر تامین، اضافه تامین و نوسان در تامین منابع برنامه‌های کاربردی چندلایه به‌عنوان مسئله‌ای چالش برانگیز مطرح بوده است زیرا منابع باید به‌صورت بهینه تامین شوند تا تمام کاربران بتوانند از هر منبعی بر اساس میزان مورد نیازشان استفاده کنند. رویکردهای قبلی، هر کدام مزایا و معایبی دارند که آن‌ها را به‌طور مختصر بیان کرده‌ایم. بنابراین جهت بهینه‌سازی زمان‌بندی درخواست‌های کاربران و پاسخ‌دهی به آن‌ها، همچنین تقلیل مشکل تخطی از سرویس، در روش پیشنهادی این مقاله در مرحلهٔ تحلیل حلقه MAPE-k از مدل رگرسیون خطی(LRM) و در مرحلهٔ برنامه‌ریزی، از روش مبتنی بر نظریهٔ بیز به منظور بهینه نمودن اقدامات استفاده شده است که بعد از ارزیابی و مقایسه با ارائه جداول و نمودارهای همه جانبه به تشریح قابلیت‌ها و بهینگی روش پیشنهادی می‌پردازیم و شاهد افزایش تعداد ماشین‌های مجازی با بهبود نرخ مقیاس‌بندی، کاهش میانگین بهره‌وری، کاهش زمان پاسخ‌دهی، در نتیجه کاهش هزینه تمام شده و افزایش سود حاصل خواهیم بود.

۳- دانش پیش زمینه

رایانش ابری، امکان دسترسی به محدوده وسیعی از منابع مجازی (مانند سخت‌افزار، نرم‌افزار، سرویس‌ها و...) را فراهم می‌آورد که این منابع می‌توانند به‌صورت پویا و با حجم متغییر به کاربران خود سرویس دهند، یعنی هر کاربر به اندازه‌ای که از آن منابع نیاز دارد، از آن‌ها استفاده

22- Support vector machines
23- Fuzzy Analytic Hierarchy Process
24- Auto Regressive Integrated Moving Average
25- Back Propagation Netwok

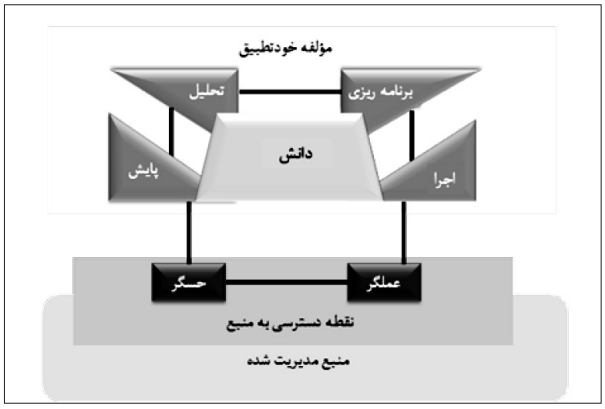
می‌کند که این کار به استفاده بهینه از منابع منجر می‌شود. رایانش ابـری، باید به‌صورت خودکار منابع مورد نیاز به کاربـران را تخصیص دهد و اگر کاربری مقداری بیشـتر از منابع را خواسـتار باشـد، باید آن منابـع بدون آن که سرویس کاربر با مشکل مواجه شود به مشتری تخصیص یابد و در صورتی که منابع مورد استفاده بیش از حد مورد نیاز باشد باید منابع اضافی به‌صورت موقت خاموش شده تـا در صورت نیاز مجددا مورد اسـتفاده قرار گیرند. این مفاهیـم در رایانش ابری تحت عنـوان مقیاس‌پذیری و یا گاهی اوقات مقیاس‌بندی شناخته می‌شود.

مقیاس‌پذیری یکی از بهترین راه‌های افزایش بهره‌وری و بهبود عملکرد سیستم‌های ابری می‌باشد و توسعه‌دهندگان ابر معتقدند که باید برای کاربران به مقدار نیاز منابع فراهم شود و از طرفی، کاربران نیز علاقه‌مند به پرداخت هزینه به مقدار استفاده شده (پرداخت-به ازای- استفاده)[۲۶] هستند. باید به این نکته توجه داشت که اکثر عناصری که در حوزه فنـاوری اطلاعات مطرح هسـتند، امکان تغییـر مقیاس را دارا می‌باشـند امّا برخی از مهم‌ترین این عناصر عبارتند از: مقـدار پردازنده اختصاص یافته، مقـدار حافظه RAM در دسترس، مقدار فضای دیسک سخت، تعداد برنامه‌های اجرا شده، مقدار پهنای باند ارتباطی، تعداد کاربران نهایی، تعداد درگاه‌های ارتباطی، تعـداد مجوزهای نرم‌افزاری و غیره. سـازوکار تامین پویای منابـع در رایانش ابری، به برنامه‌ها این امکان را می‌دهد تا در مقیاس‌های بالا و پایین اجرا شوند و بدین ترتیب عملکرد و پایداری اقتصادی آن‌ها متعـادل می‌گردد. یکی از چالش‌های مهم در رایانش ابری، دسـترس‌پذیری و مدیریت منابع می‌باشـد. نیاز هست که سیسـتم‌های ابری، قوی و قابل دسترس باشند تا مدیریت به راحتی صورت گیرد. چرا که بیشـتر تجارت‌ها، توسط شرکت‌های مختلف وابسته به خدمات ارائه شده است [۲].

۳-۱- چرخه خودمختار MAPE-K

یک سیسـتم خودتطبیق سیسـتمی اسـت کـه بدون

26- Pay-Per-Use



شکل ۱: چرخه MAPE-K [۱۶]

دخالـت کاربر بتواند بر روی مدیریـت معماری، عملکرد، کارایـی و کنترل خرابی خود تصمیـم گیری کند. در واقع در یک سیستم خودتطبیق معمولاً از سیستم‌های پشتیبانی تصمیم استفاده می‌شود. در یک سیستم خودتطبیق چرخه چهارگانـه MAPK-E وجـود دارد کـه عبارتنـد از پایش سیسـتم، تحلیل سیسـتم، برنامه‌ریزی و اجرا که همه این موارد در کنار یک پایگاه دانش انجام می‌شوند. این چرخه در شـکل (۱). قابل مشاهده می باشد [۱۶]. این مدل بیشتر و بیشـتر برای برقـراری ارتباط بیـن جنبه‌های معماری سیسـتم‌های خودمختار استفاده می‌شود. به همین ترتیب این راهی روشـن برای شناسایی و طبقه‌بندی بسیاری از کارهایی است که دراین زمینه در حال انجام است.

بنابراین این بخش، چرخه MAPE-K را با جزئیات بیشتر معرفی می‌کند و پس از آن هر یک از اجزای سازنده آن را به نوبه خود در نظر می‌گیرد. ما بعد از آن همان کار را در نظر گرفته و از نقطه نظر قواعد خودمختاری به بررسی آن می‌پردازیم.

در چرخه MAPE-K یک عامل هوشـمند، محیط خود از طریق حسـگرها درک نمـوده و از این ادراک برای تعیین اقدامات به منظور اجرا بر روی محیط استفاده می‌کند.

در چرخه MAPE-K مؤلفه خودمختار که بلوک سازنده یک سیستم خودمختار اسـت از مدیر خودمختار و مؤلفه مدیریت شـده متشـکل می‌باشـد. عنصر مدیریت شـده نشـان دهنده هر منبع نرم‌افزاری یا سـخت‌افزاری اسـت

که رفتار خودمختـــاری را از طریق قـــرار گرفتن در کنار مدیـــر خودمختار ارائه می‌دهد. بنابرایـــن، عنصر مدیریت شـــده می‌تواند برای مثال یک کارساز وب و یا پایگاه داده، یکی از اجزای نرمافزاری خـــاص در یک برنامه (به‌عنوان مثال، بهینه‌ســـاز پرس‌وجو در پایگاه داده)، سیستم عامل، یک خوشـــه از ماشـــین‌ها در یک محیط شـــبکه، یک پشته از دیســـک‌های سخت، یک شبکه ســـیمی یا بی‌سیم، CPU، چاپگر، و غیره باشد. مدیر خودمختار نیز مؤلفه نرم‌افزاری اســـت که توسط مدیران و با اســـتفاده از اهداف سطح بالا پیکربندی شـــده اســـت، در هر دو مدل، کارســـاز، ابزاری بـــرای جمع‌آوری اطلاعـــات مرتبط و تأثیرگـــذار بوده که برای یک ســـرویس‌دهنده وب، این می‌تواند شـــامل زمان پاســـخ به درخواست مشتری، استفاده از شبکه و مصرف دیســـک، CPU و مصرف حافظه باشـــد همچنین مســـئول اعمال تغییراتی از قبیل اضافه کردن یا حذف کارســـاز به/ از یک خوشه کارساز وب بزرگ بوده یا شامل پارامترهای پیکربندی در یک کارســـاز وب کوچک در مؤلفه مورد نظر باشد.

چهار مرحلهٔ اصلی رایانـــش خودمختار پایش، تحلیل، برنامه‌ریزی و اجراست. که در ادامه مقاله آن‌ها را تشریح می‌کنیم.

۳-۱-۱ پایش

پایش شـــامل اتخاذ خواص محیط (فیزیکی یا مجازی، به‌عنوان مثال، یک شـــبکه)، که برای چهار خواص سیستم ذکر شـــده (خود-X) دارای اهمیت هستند می‌باشد. مؤلفه نرم‌افزاری و یا ســـخت‌افزاری مورد اســـتفاده برای انجام پایش، حســـگر نام دارد. به‌عنوان مثال، زمان تأخیر شبکه و پهنای باند عملکرد کارساز وب را اندازه‌گیری می‌کند، در حالی که نمایه‌ســـازی پایگاه داده و بهینه‌سازی پرس‌وجو زمان پاســـخ "DBMS" [۲۷] را تحت تاثیر قـــرار می‌دهند، که می‌توان آن‌هـــا را تحت پایش قـــرار داد. مدیر خودمختار برای تشـــخیص خرابی یا عملکرد بهینه عنصر خودمختار به اطلاعات پایش شده مناسب و تغییرات مناسب تأثیرگذار نیاز دارد. انواع خواص پایش شـــده، و حســـگرهای مورد اســـتفاده، اغلب برنامه کاربردی خاصی هستند، و عوامل مؤثر مورد استفاده برای اجرای تغییرات در عنصر مدیریت شده نیز برنامه کاربردی خاص هستند.

۳-۱-۲ تحلیل

مرحلهٔ تحلیل، با پردازش معیارهای جمع‌آوری شده از سیستم پایش، ســـر و کار دارد و با پردازش این معیارها، داده‌هایی در مـــورد وضعیت بهره‌وری فعلی سیســـتم و پیش‌بینی‌هایـــی از نیازهای آتی، به‌دســـت مـــی‌آورد تا در صورت نیاز به مختار، پاســـخ مناســـبی به تحریکات داده شود. بعضی از رایانش‌های خودمختار هیچ نوع پیش‌بینی انجام نمی‌دهند و فقط به وضعیت جاری سیســـتم پاســـخ می‌دهند که به آن‌ها واکنشـــی می‌گوییـــم، امّا برخی دیگر از این محاســـبات از تکنیک‌های پیچیده‌ای برای پیش‌بینی تقاضاهای آتی به منظـــور تنظیم منابع به اندازه پیش‌بینی شـــده، اســـتفاده می‌کنند که به آن‌ها پیش‌کنشی می‌گوییم. پیش‌بینی مهم است زیرا همیشه یک تاخیر زمانی از لحظه اجرا شـــدن عمـــل رایانش خودمختار (مثـــلا اضافه کردن ماشین مجازی) تا مؤثر واقع شدن عمل (به دلیل راه‌اندازی مجدد سیستم، برای عملیاتی شدن یک ماشین مجازی چند دقیقه زمان لازم اســـت) وجود دارد. سیستم‌های واکنشی، قادر به مقیاس‌بندی ترافیک‌های ناگهانی بار کاری ورودی نیستند و به همین دلیل از رویکردهای پیشکنشی، به منظور برخورد با نوسانات تقاضا و تامین پیشاپیش منابع استفاده می‌شود.

۳-۱-۳- برنامه‌ریزی

این مرحله، هسته اصلی رایانش خودمختار است که در وسیع‌ترین مفهوم، شامل اتخاذ داده‌های پایش از حسگرها بـــرای تولید یک ســـری از تغییرات بـــرای تاثیر بر عنصر مدیریت شده می‌باشد تا در جهت بهبود عملیات ارائه شده، تغییراتی اعمال شود.

اگر اعمال این روش به شـــیوه‌ای بـــدون حالت (که در آن مدیر خودمختـــار هیچ اطلاعاتی در مورد حالت عنصر

27- Data-base Management System

مدیریت شـــده را نگه نمی‌دارد و برای تصمیم‌گیری مبنی براین که آیا بر یک طـــرح خودمختاری اثر گذارند یا خیر، صرفاً متکی بر اطلاعات حســـگر جریان می‌باشـــد) بسیار محدود است. در واقع، برای مدیر خودمختار به مراتب بهتر است که اطلاعات را بر روی حالت عنصر مدیریت شده که می‌تواند به تدریج از طریق داده‌های حســـگر به روز شود، نگه دارد تا با توجه به وضعیت سیســـتم و سایر اطلاعات موجود، تصمیم بگیرد چه راهبرد بایستی اتخاذ شود.

با توجه بیشـــتر به موضوع اطلاعـــات حالتی که مدیر خودمختار باید در مورد عنصر مدیریت شـــده حفظ کند، تحقیقـــات زیادی یـــک رویکرد کامل‌تر بـــرای مدیریت یک سیســـتم (که رویکرد مدل‌محور معماری نام دارد) که در آن نوعی از مدل کل سیســـتم مدیریت شـــده توسط مدیر خودمختار ایجاد می‌شـــود، را مورد بررســـی قرار دادند. ایـــن معمولاً یک مدل معماری نامیده می‌شـــود که نشـــان دهنده رفتار سیســـتم مدیریت شده، الزامات آن، و احتمالاً هدف آن می‌باشـــد. مدل همچنین ممکن است نشان دهنده برخی از جنبه‌های محیط عملیاتی باشد که در آن عناصر مدیریت شده قرار دارند که در آن محیط عملیاتی می‌تواند به‌عنوان هر خاصیت قابل مشاهده (توسط حسگرها) درک شود که به‌عنوان مثال می‌تواند اجرا، ورودی کاربر نهایی، دستگاه‌های سخت افزاری و خصوصیات اتصال به شبکه را تحت تاثیر قرار دهد.

مدل از طریق داده‌های حســـگر به‌روز شده و به همین دلیل در سیستم مدیریت شده برای برنامه‌ریزی سازگاری مورد استفاده قرار می‌گیرد. یک مزیت بزرگ رویکرد مبتنی بر مدل معماری برای برنامه‌ریزی این است که با این فرض که مدل به درســـتی منعکس کننده سیســـتم مدیریت شده باشد، مدل معماری را می‌توان برای تأیید این که یکپارچگی سیستم هنگام اعمال یک سازگاری حفظ می‌شود، استفاده نمـــود. به این معنا که می‌توانیم تضمین کنیم که پس از آن که ســـازگاری برنامه‌ریزی شـــده اجرا شـــود، سیستم به درستی به کار خود ادامه خواهد داد. این به‌خاطر این است که تغییرات برنامه‌ریزی شـــده اول به مدل اعمال می‌شود، که حالت سیســـتم ناشی از ســـازگاری (از جمله هر گونه نقض محدودیت و یا الزامات سیستم ارائه شده در مدل) را نشان خواهد داد. اگر حالت جدید سیستم قابل قبول باشد، این طرح پس از آن می‌تواند بر روی سیستم مدیریت شده واقعی تاثیر گذارد، بنابراین اطمینان می‌دهد که مدل و پیاده سازی نسبت به یکدیگر سازگار هستند.

هنگامـــی که یک تغییر در سیســـتم مدیریت شـــده رخ می‌دهد و این تغییر به مدل اعمال می‌شود، همیشه یک تاخیر بین زمان این دو کار وجود دارد. در واقع، در صورتی که تاخیر به اندازه کافی بالا بوده و سیســـتم نیز مکرر تغییر نماید، ممکن اســـت یک طرح سازگاری ایجاد شده و برای اجرا ارسال گردد.

۳-۱-۴ اجرا

این فرایند مسئول اجرای تصمیمات اتخاذ شده توسط مرحلهٔ برنامه‌ریزی اســـت. به عبارتی دیگـــر، این مرحله، مرحله‌ای ســـر راســـت و مستقیم اســـت که از طریق API ارائه‌دهنده ابری، پیاده‌سازی می‌شود و پیچیدگی‌های ذاتی و واقعی را از دید مشتری مخفی می‌کند [۵].

۳-۲ یادگیری بیزی

استدلال بیز روشی احتمالی برای استنتاج ارائه می‌کند. این روش بر اســـاس این فرض اســـت که کمیت‌های مورد نظر از توزیع‌های احتمـــال پیروی می‌کنند و تصمیم‌گیری بهینه را می‌توان با اســـتدلال بر ایـــن توزیع‌های احتمال و داده‌های مشـــاهده شـــده انجام داد. اهمیـــت این روش در یادگیری ماشـــین این است که روشـــی کمی برای ارزیابی مـــدارک فرضیه‌هـــا ارائه می‌کند. اســـتدلال بیزی اســـاس اگوریتم‌هـــای یادگیری که بـــا اســـتفاده از احتمالات کار می‌کنند است. همچنین استدلال بیز چارچوبی برای بررسی عملیات دیگر الگوریتم‌هایی که از احتمالات استفاده نمی‌کنند ایجاد می‌کند.

روش‌هـــای یادگیری بیـــزی به دو دلیل بـــه مطالعه ما در مورد یادگیری ماشـــین مربوط می شـــود. اول این که

الگوریتم های یادگیری بیـزی احتمال صریح هر فرضیه، مثل دسـته‌بندی‌کنندهٔ سـاده بیز را محاسبه می‌کنند. این نوع روش‌ها از پر کاربردترین روش‌ها در بعضی مسـائل یادگیری هستند [۱۷].

۳-۲-۱ ویژگی‌های یادگیری بیزی

آشنایی پایه‌ای با روش‌های بیزی برای درک بسیاری از الگوریتم‌هـای یادگیری ماشـین اهمیت بسـزایی دارد. ویژگی‌های روش‌هـای یادگیری بیزی شـامل این موارد می‌باشد:

- هـر نمونه آموزشـی می‌تواند احتمـال تخمینی این که فرضیه درسـت اسـت را کم یا زیاد کند. این حقیقت باعث می‌شـود که روش‌های بیزی نسـبت به الگوریتم‌هایی که کامـلاً فرضیه را بـا نمونه‌های غیرسـازگار رد می‌کنند انعطاف‌پذیرتر باشند.
- می‌توان از دانش قبلی به همراه داده‌ای مشـاهده شـده برای تعیین احتمال نهایی درسـتی فرضیه‌ها استفاده کرد. در یادگیـری بیزی، دانـش قبلی با (۱) احتمـال اولیه هر فرضیه و (۲) احتمال روی داده‌های تعیین شـده برای هر فرضیه ممکن تعیین می‌شود.
- روش‌های بیـزی می‌توانند بـرای فرضیه‌ها احتمالاتی راپیش‌بینی کنند (برای مثال، فرضیه این بیمار ذات‌الریه با احتمال ۹۳٪ کاملا بهبود خواهد یافت).
- نمونه‌هـای جدیـد را می‌توان با ترکیـب پیش‌بینی‌های چندین فرضیه، (هر کدام با وزن احتمال‌شـان) دسته‌بندی کرد.

حتی هنگامی که اثبات می‌شـود کـه روش‌های بیزی محاسـباتی غیرقابل پیش‌بینی انجـام می‌دهند، با این حال معیار اسـتانداردی برای دیگـر روش‌های عملی یادگیری مطرح می‌کنند [۱۸].

۳-۳ برنامه‌های کاربردی چند لایه

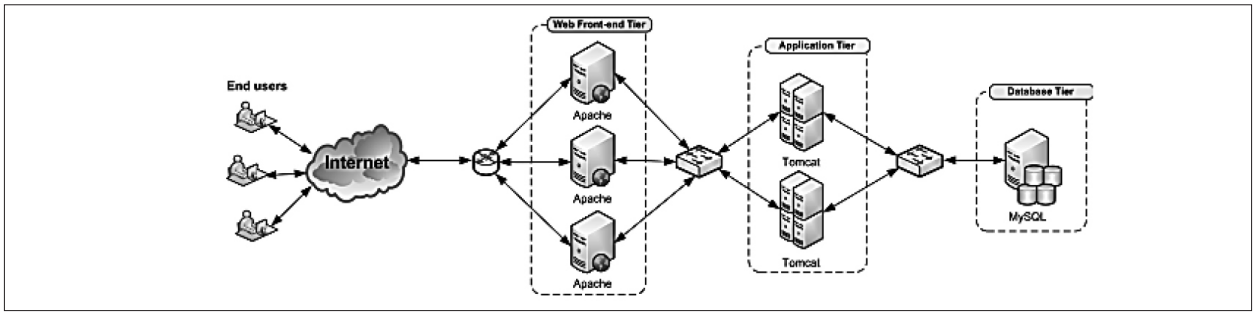
اخیراً پیشرفت‌های بزرگی در معماری سازمانی تحت وب صـورت گرفته اسـت. بخصـوص در رایانش ابری که ارائه‌دهنـدگان خدمات را قادر می‌کند خدمات مبتنی بر معماری وب سـنتی را متوقف کنند. ترتیـب تکامل چنین برنامه‌های کاربردی (از معماری رایانهٔ بزرگ تا کارخواه/ کارساز سنتی و سپس پیاده‌سازی مجازی فعلی) گویای این است که راه‌اندازی سیستم مستقل نه تنها هزینه‌های کلی را افزایش می‌دهد بلکه کاهش مقیاس‌پذیری، دسترس‌پذیری و کاهش تحمل خطای سرویس را به همراه خواهد داشت. امروزه ارائه‌دهندگان خدمات عمدتاً سرویس‌های مبتنی بر وب، برنامه‌های کاربردی قدیمی و سیستم‌های پایگاه داده را روی مجموعه‌ای از کارسازها مستقر می‌کنند.

معمـاری تک‌لایه سـاختار نسـبتا سـاده‌ای دارد و راه‌اندازی آن آسـان است، امّا اغلب وبگاه‌های امروزی از معماری چندلایه اسـتفاده می‌کنند. این معماری، پردازش برنامه کاربردی را به چندین لایه تقسـیم می‌کند. هر لایه، قابلیت‌های مشخصی را ارائه می‌کند. مزیت چنین معماری این است که می‌تواند سطح بالایی از مقیاس‌پذیری و قابلیت اطمینان را فراهم کند [۱۹]. علاوه بر این، معماری چندلایه برای بهینه‌سازی هزینه‌های مالی در فرایند گردش کار نیز مفید است. با این وجود، به دلیل وابستگی لایه‌ها، تخصیص منابع میان آن‌ها مشکل است.

بنابراین ارائه‌دهندگان خدمات برای وبگاه‌های تجارت الکترونیکی مثل eBay و سیستم‌های اطلاعاتی مبتنی بر وب مانند ویکی‌پدیا، از معماری چند لایه استفاده می‌کنند.

معماری چند لایه شـامل لایه‌های مختلف است که هر لایه میزبان کارسـازهایی با عملکرد مشابه است. هر لایه یک سرویس از پیش تعریف شده برای جایگزینی و دریافت خدمات از لایه قبلی ارائه می‌کند. واضح اسـت که اولین و آخرین لایه، قبل و جایگزین ندارند.

اغلـب برنامه‌های کاربردی چنـد لایه تحت وب، از یک معماری سـه‌لایه اسـتفاده می‌کنند. سـه لایه شامل لایه نمایش، لایه نرم‌افزار (منطق تجاری) و لایه دسترسـی به داده به ترتیب به‌صورت کارسـاز وب، کارسـاز نرم‌افزار (برنامـه کاربردی) و کارسـاز پایگاه داده، پیاده‌سـازی شده‌اند.



شکل ۲: معماری برنامه کاربردی۳- لایه[۹]

شکل (۲) یک برنامه کاربردی با یک پیکربندی دلخواه از کارسازها را در یک مرکز داده نشان می‌دهد. زنجیره‌ای از لایه‌ها ممکن است شامل متعادل‌کننده بار، پروکسی کارسازها، دروازه‌ها یا کارسازهای حافظهٔ نهان برای بهبود عملکرد نرم‌افزار و امنیت باشند. این مؤلفه‌ها در شکل به‌صورت اختصاری نشان داده شده‌اند.

اولین لایه که لایه نمایش نامیده می‌شود، شامل کارسازهای وب است. این لایه هر چیزی را که در سمت مشتری، داخل مرورگرهای وب ارائه می‌شود، نمایش می‌دهد. برای کارساز وب، سه وظیفه اصلی وجود دارد [۲۰]: (۱) پذیرفتن/ رد کردن درخواست‌های مشتریان و درخواست‌های سرویس‌های ایستای تحت وب؛ (۲) عبور دادن درخواست‌ها به سمت کارساز نرم‌افزار؛ (۳) دریافت پاسخ از کارساز نرم‌افزار و فرستادن آن‌ها برای مشتریان. کارساز آپاچی[۲۸] و کارساز اطلاعات اینترنت مایکروسافت (IIS) نمونه‌هایی از این نوع کارسازها هستند.

دومین لایه که لایه نرم‌افزار نامیده می‌شود، شامل کارسازهای نرم‌افزار است. پردازش منطق تجاری، در این لایه اجرا می‌شود. همچنین سه وظیفه این لایه عبارت است از: (۱) دریافت درخواست‌ها از کارساز وب؛ (۲) جستجوی اطلاعات در پایگاه داده و پردازش اطلاعات؛ (۳) فرستادن اطلاعات پردازش شده به کارساز وب. کارسازهای نرم‌افزار اغلب از آپاچی تامکت[۲۹] و کارساز نرم‌افزار سیستم سان‌جاوا[۳۰] استفاده می‌کنند.

آخرین لایه که لایه داده نامیده می‌شود، شامل کارسازهای پایگاه داده است. این لایه، پردازش پایگاه داده و دسترسی به داده‌ها را مدیریت می‌کند. لایه کارساز پایگاه داده، برای ذخیره و بازیابی اطلاعات یک وبگاه (مانند حساب کاربری، فهرست گزارش‌ها و سفارش‌های مشتری) استفاده می‌شود.

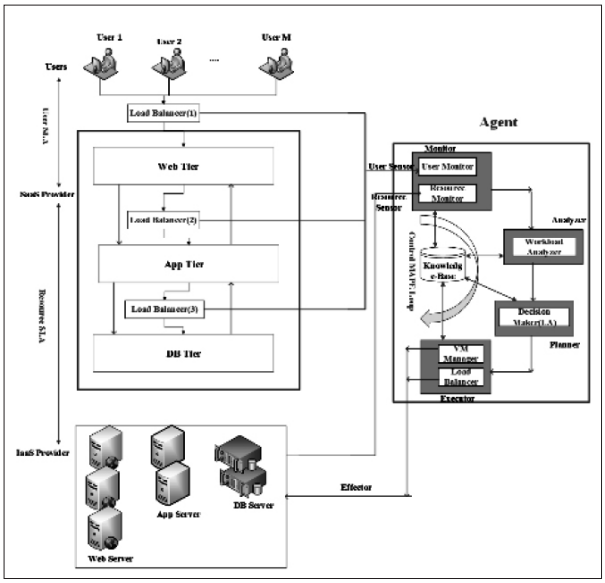
بر اساس قابلیت‌های لایه‌ها، تقاضای منابع از هر لایه به لایه دیگر متفاوت است. به‌عنوان مثال لایه وب روی CPU و لایه پایگاه داده روی I/O متمرکز است. هر لایه نرم‌افزارها و سخت‌افزارهای مختلفی دارد و قادر است تعداد مشخصی از درخواست‌ها را سرویس‌دهی کند. با توجه به قابلیت‌های مختلف و تقاضا برای منابع، برنامه‌های کاربردی چند لایه با چالش‌هایی مانند وابستگی عملکرد و جابه‌جایی تنگنا بین لایه‌ها مواجه است. ارسال درخواست بیشتر در یک لایه ممکن است آن را مستعد تنگنا و مشکلات عملکرد کند. تخریب عملکرد یک لایه ممکن است باعث ایجاد تنگنا شود که عملکرد برنامه را به‌طور کلی مورد تهدید قرار می‌دهد. (مثل اثر دومینو). در چنین شرایطی برخی لایه‌ها ممکن است استفاده از منابع لایه‌های بالا را تجربه کنند؛ در حالی که لایه‌های دیگر مورد استفاده قرارگرفته‌اند و در انتظار دریافت درخواست لایه‌های دیگرند. در نهایت عملکرد نرمافزار به‌طور کلی تخریب خواهد شد. بنابراین طراحی یک برنامه کاربردی متعادل‌کننده بار مناسب می‌تواند از تخریب بیشتر جلوگیری کند [۹]. در این بخش قصد داریم رویکرد پیشنهادی خود را با جزییات بیشتر توضیح دهیم، روشی برای تامین منابع بر روی ساختار MAPE-K با استفاده از ترکیب مدل رگرسیون خطی(LRM) و نظریهٔ بیز

28-Apache Server
29-Apache Tomcat
30-Sun Java

با توجه به این که این ساختار شامل چهار مرحلهٔ پایش، تحلیل، برنامه‌ریزی و اجرا است، در روش پیشنهادی تمامی مرحله‌ها عملکرد خاصی دارند. در ادامه این بخش ابتدا چارچوب مبتنی بر این حلقه را شرح می‌دهیم و در نهایت الگوریتم پیشنهادی در این خصوص را ارائه می‌دهیم.

٤-١ چهارچوب پیشنهادی

در این بخش، یک چهارچوب تأمین منبع برای برنامه‌های کاربردی چند لایه بر اساس حلقه کنترلی MAPE-K، برای پیاده‌سازی رویکرد پیشنهادی ارائه خواهیم داد. همان‌طور که در شکل (۲) نشان داده شده است، چهارچوب کلی شامل سه مولفه اصلی به نام تامین کننده "SaaS" [۳۱]، تامین کننده "IaaS"[۳۲] و مولفه عامل خودمختار مبتنی بر حلقه کنترلی MAPE-K می‌باشد. تامین کننده SaaS به‌عنوان مالک برنامه کاربردی سه لایه برای ارائه سرویس به کاربران نهایی می‌باشد از آنجائی که ممکن است تعداد درخواست‌های ورودی به یک لایه از ظرفیت میزان ماشین‌های مجازی تامین شده برای آن لایه، بیشتر باشد، صفی از درخواست‌ها قبل از ورود به هر لایه در قالب متوازن کننده بار[۳۳] تشکیل می‌شود. کاربران نهایی، درخواست‌هایشان را برای استفاده از برنامه کاربردی ابری، به تامین کننده SaaS ارائه می‌کنند. تامین‌کننده SaaS، برای میزبانی برنامه کاربردی‌اش، یا از منابع مراکز داده داخلی خودش استفاده می‌کند یا این که منابع مورد نیاز برای میزبانی برنامه کاربردی را از یک تامین کننده IaaS مانند Amazon EC2 اجاره می‌کند (که فرض ما در این تحقیق بر همین اساس است). تامین کنندگان IaaS، منابع نامحدودی در قالب ماشین‌های مجازی (کارساز) در رده‌های مختلف (مثلا سه نوع ماشین مجازی Small, Medium, Large) برای اجرای درخواست‌های لایه‌های مختلف برنامه کاربردی چند لایه به تامین کنندگان SaaS ارائه می‌دهد. مولفه عامل خودمختار مسئول مدیریت



شکل ۳: چارچوب تأمین منابع برای برنامه‌های کاربردی چندلایه بر اساس حلقه کنترلی MAPE

تامین منابع برای برنامه کاربردی چند لایه تامین کننده SaaS بر اساس حلقه کنترلی MAPE-K می‌باشد که از طریق دو زیرمولفه پایش کاربر و پایش منبع درخواست‌های کاربران و وضعیت منابع را به ترتیب پایش کرده و از طریق دو زیرمولفه متوازن کننده بار و مدیر ماشین مجازی میزان منابع مناسب را به درخواست‌ها تخصیص می‌دهد. علاوه بر این، دو نوع SLA به نام‌های SLA کاربر و SLA منبع وجود دارد. SLA کاربر، قراردادی بین تامین کننده SaaS و کاربر نهایی است و شامل فقره‌هایی مانند: مهلت اتمام، بودجه، نرخ جریمه و طول درخواست می‌باشد. SLA منبع، قراردادی بین تامین کننده SaaS و تامین کننده IaaS می‌باشد و شامل فقره‌هایی مانند: قیمت ماشین مجازی، نوع ماشین مجازی، سرعت پردازش، زمان سرویس و سرعت انتقال داده می‌باشد. در این پژوهش، برای محاسبه تخطی از SLA بر روی تعامل بین کاربر و تامین‌کننده SaaS (SLA) کاربر متمرکز می‌شویم.

همان‌طور که در شکل (۳) نشان داده شده است، مولفه‌های اصلی چارچوب تأمین منابع مبتنی بر حلقه کنترل MAPE-K عبارتند از:

- **پایش**[۳۴]: مولفه پایش، مسئول جمع‌آوری معیارهای

31- Software as a Service
32- Infrastructure as a Service
33- Load Balancer
34- Monitor

مرتبط با منابع و کاربران است. مولفه پایش شامل دو زیرمولفه پایش کاربر و پایش منبع می‌باشد. مولفه پایش منبع، مسئول جمع‌آوری اطلاعات در مورد منابع محاسباتی، ذخیره‌سازی و شبکه (مانند بهره‌وری پردازنده، میزان حافظه مصرف‌شده، متوسط زمان سرویس و ترافیک شبکه) از تامین کننده IaaS می‌باشد. مولفه پایش کاربر، مسئول جمع‌آوری اطلاعات در مورد بارکاری واگذار شده توسط کاربران (مانند نرخ درخواست‌های ورودی، نوع درخواست‌ها، اندازه درخواست‌ها، درخواست‌های پردازش شده،درخواست‌های‌متوقف‌شده‌و...)ازسه‌صف‌درخواست (Load Balancer1, Load Balancer3, Load Balancer2) می‌باشد. این اطلاعات پایش شده توسط دو زیرمولفه پایش کاربر و پایش منبع، جمع‌آوری و یکپارچه شده و برای استفاده توسط سایر مراحل در پایگاه دانش ذخیره می‌شود.

● **تحلیل‌گر**[35]: مولفه تحلیل‌گر، مسئول پردازش اطلاعات جمع‌آوری شده از مولفه پایش می‌باشد. داده‌های به‌دست آمده توسط زیرمولفه‌های پایش، توسط مولفه تحلیل‌گر برای بررسی اعمال خودمختاری مورد نیاز برای تضمین سطح کیفیت سرویس، بررسی می‌شوند. مولفه تحلیل‌گر بارکاری، از تکنیک‌های پیش‌بینی برای پیش‌بینی تعداد درخواست‌ها و یا بهره‌وری پردازنده برای مقابله با نوسانات بارکاری استفاده می‌کند.علاوه بر این، تحلیل‌گر زمان پاسخ مشتری‌های مختلف را نیز دریافت می‌کند.

● **برنامه‌ریز**[36]: مولفه برنامه‌ریز، هسته اصلی چارچوب تامین منابع است. این مولفه تعیین می‌کند که چه موقع و چه تعداد ماشین مجازی باید به سرویس‌های ابری مختلف برای پیدا کردن یک مصالحه بین میزان هزینه و میزان تخطی از SLA تخصیص یابد. در این مقاله از تکنیک یادگیری بیزی به‌عنوان تصمیم گیرنده در این مولفه استفاده می‌کنیم که از نتایج پیش‌بینی مولفه تحلیل‌گر برای حذف یا اضافه کردن ماشین مجازی استفاده می‌کنند.

35- Analyzer
36- Planner

● **اجرا کننده**[37]: مولفه اجرا کننده شامل دو زیرمولفه متوازن کننده بار و مدیر ماشین مجازی است. زیرمولفه متوازن کننده بار، تمام درخواست‌های ورودی از کاربران را دریافت کرده و آن‌ها را بین ماشین‌های مجازی مناسب بر طبق سیاست‌های توازن بار (مانند Round Robin, Random,...) توزیع می‌کند. به دلیل این که زیرمولفه مدیر ماشین مجازی، به‌طور مستقیم با ماشین‌های مجازی تامین کنندگان IaaS در ارتباط است، این مولفه، مسئول اجرای واقعی اعمال (Scale up/Scale down) تصمیم گرفته شده توسط مولفه برنامه ریز می‌باشد.

٤-٢- الگوریتم پیشنهادی

در این بخش، الگوریتم تامین خودکار منبع برای لایه‌های مختلف برنامه کاربردی ابری را با جزئیات بیشتری تشریح می‌کنیم و به بیان الگوریتم اصلی می‌پردازیم که چهار تابع MAPE-K را فراخوانی می‌کند. پس از استقرار ارائه دهنده SaaS با لایه‌های مختلف ابر در چهارچوب ابر، الگوریتم ارائه شده تا زمانی که هیچ یک از لایه‌های ابر در ارائه دهنده SaaS در حال اجرا نباشد، اجرا می‌شود. این الگوریتم حلقه کنترل MAPE-K را دنبال می‌کند و ماشین‌های مجازی را مدیریت می‌نماید. این الگوریتم، ابتدا تعدادی از ماشین‌های مجازی مناسب را بر اساس تاریخچه تجربی موجود با استفاده از پیش‌بینی تکنیک‌ها برای هر لایه از مخزن VMها توسط ارائه دهنده IaaS راه‌اندازی می‌کند. برای هر لایه ابر Li توسط ارائه دهنده SaaS در Δt پیشنهاد شده است.حلقه کنترل MAPE-K همانند آنچه در الگوریتم (۱) نشان داده شده است به‌طور منظم در بازه‌های زمانی خاص اجرا می‌شود و با مقدار بهینه منابع بر اساس حجم کار لایه‌های ابر منطبق می‌گردد. در این الگوریتم حلقه کنترل MAPE-K شامل مراحل پایش، تحلیل، برنامه‌ریزی و اجرا می‌باشد. ابتدا پایش سیستم شروع به جمع‌آوری اطلاعات درباره کاربر و وضعیت منابع می‌کند سپس تحلیل‌گر از اطلاعات به‌دست آمده به منظور تخمین منبع

37- Executor

اضافی و نیازمندی‌ها در این خصوص استفاده می‌نماید. سپس در مرحلهٔ برنامه‌ریزی، اقدامات لازم در خصوص منابع صورت می‌گیرد یعنی در صورت نیاز ماشین مجازی اضافه یا حذف می‌شود. در نهایت، ارائه دهنده اقدامات درخواست شده توسط برنامه‌ریز را برای مرحله بعد اجرا می‌کند.

Algorithm1: Pseudocode for Automatic resource provisioning

```
1:  Initialization: boots an appropriate number of VMs for cloud layers
2:  while (the system is running and in the beginning of interval Δt) do
3:  Begin
4:  for (every cloud layer Li in multi layer cloud application at the interval Δt ) do
5:  Begin
6:  Monitoring (Li)
7:  Analysis (Li)
8:  Planning (Li)
9:  Execution ( )
10: end for
11: end while
```

٤-٢-١ مرحلهٔ پایش

مرحلهٔ پایش، مرحله‌ای است که معیارهای مرتبط با کاربران و منابع، پایش شده‌اند. الگوریتم (۲) به منظور عملیات پایش در پژوهش مورد نظر مورد استفاده قرار گرفته است. در بخش کاربر، برای هر لایه ابر Li که توسط یک ارائه‌دهنده SaaS پیشنهاد شده است، پایش کاربر به معیارهایی از تعداد VMها که باید از ارائه‌دهنده IaaS اجاره شوند و توسط ارائه دهنده SaaS برای لایه ابر Li در بازه Δt ام و تعداد درخواست‌ها برای لایه Li در بازه Δt ام اجرا شود را پایش می‌کند. در بخش منابع، پایش منبع، مصرف پردازنده VMهایی که به لایه Li در Δt تخصیص یافته‌اند، تعداد VM ها و بارکاری را در Δt پایش می‌کند. این اطلاعات برای استفاده در سایر مراحل در پایگاه دانش ذخیره می‌شود.

Algorithm2: Pseudocode for Monitoring phase

```
1: Begin
2: for (every cloud layer Li in multi layer cloud application at the interval Δt) do
3: Begin
4: Monitor( Wi(Δt)); /* workload monitoring*/
5: Monitor (Numi(Δt)); /*Resource Monitoring*/
6: Monitor (Ui(Δt)); /*Monitoring CPU utilization of Layer Li*/
7: end for
8: end
```

۴-۲-۲ مرحلهٔ تحلیل

در این مرحله، معیارهای به‌دست آمده از مرحلهٔ پایش مستقیما به منظور پیش‌بینی میزان بهره‌وری پردازنده برای لایه ابر Li در بازه Δ(t+1) با استفاده از تکنیک‌های پیش‌بینی پردازش می‌شوند. بنابراین تحلیلی را انجام می‌دهیم که بر اساس آن در مرحلهٔ برنامه‌ریزی، اقدام به تصمیم‌گیری می‌نماییم. لذا تعیین می‌کنیم که برای بازه Δ(t+1)، تغییرات تعداد ماشین‌های مجازی چگونه خواهد بود. این تصمیم‌گیری بر اساس یک معیار[۳۸] خاص در مرحلهٔ برنامه‌ریزی صورت می‌گیرد. برای مثال می‌توان بهره‌وری پردازنده و یا تعداد درخواست‌ها را ملاک قرار داد. در اینجا بهره‌وری پردازنده را به دلیل دقت بالایی که دارد معیاری برای پیش‌بینی بارکاری در بازه زمانی آتی قرار دادیم.

در ابتدا الگوریتم برای تک تک لایه‌های برنامه کاربردی اجرا می‌شود. سپس در خط ٤ تا ٦ تمامی متغیرهای ارزیابی مقداردهی اولیه می‌شوند. سپس تعداد درخواست‌های وارد شده به لایه iام را با یک سری زمانی تجمیع می‌کنیم. این سری زمانی برای تخمین تعداد درخواست‌ها در بازه زمانی بعدی به کار می‌رود. همچنین میانهٔ تعداد میلیون دستورالعمل[۳۹]های هر درخواست را برای لایه iام در مشاهدات n بازهٔ زمانی گذشته تا زمان کنونی را می‌یابیم. در خط ۱۱ با تحلیل‌گر حجم کار از یک مدل رگرسیون خطی (LRM) به منظور پیش‌بینی تعداد درخواست‌ها در بازه زمانی بعدی استفاده می‌کند. رابطه (۱) مدل رگرسیون خطی برای لایه ابر Li در tΔ را بیان می‌کند:

$$Y_{t+1} = \beta_1 + \beta_2 \times X_t \qquad \text{رابطه (۱)}$$

در رابطه (۲)، t نمونه مشاهده شده است و Yt حجم کار همچون تعداد درخواست‌ها برای لایه ابر Li است و Xt مقدار زمان واقعی در لحظه‌ای است که هر نمونه گرفته شده است. فرض می‌کنیم n تعداد کل نمونه‌های مشاهده شده است. بنابراین β1,β2 با استفاده از حل معادله خطی رگرسیون بر اساس رابطه(۲) تعیین می‌شوند:

38- Metric
39- Million Instructions Per Second (MIPS)

رابطه (۲)

$$\beta_1 = \frac{\sum X_t^2 \sum Y_t - \sum X_t \sum X_t Y_t}{n\sum X_t^2 - (\sum X_t)^2}$$

$$\beta_2 = \frac{n\sum X_t Y_t - \sum X_t \sum Y_t}{n\sum X_t^2 - (\sum X_t)^2}$$

خط ۱۲ مجموع میانه به‌دست آمده از تعداد میلیون دستورالعمل‌ها را بر n بازه زمانی تقسیم می‌کند تا میانه برای هر مقطع زمانی به‌دست آید. بر اساس حجم تخمینی هر درخواست (تعداد میلیون دستورالعمل هر درخواست) و تعداد درخواست‌های تخمینی در بازه زمانی آتی، اقدام به محاسبه حجم کلی تعداد میلیون دستورالعمل‌های دریافتی در بازه زمانی بعدی می‌کنیم(خط۱۳). در مرحله ۱٤ تخمینی از میزان بهره‌وری در بازه زمانی بعدی را با تقسیم حجم درخواست تخمینی بر ظرفیت کلی منابع لایه iام به‌دست می‌آوریم. بدین ترتیب میزان بهره‌وری تخمینی همه لایه‌ها را محاسبه و به‌عنوان خروجی الگوریتم مرحلهٔ تحلیل برمی‌گردانیم.

Algorithm3:Pseudocode for Analyze phase

```
1:  Input RequestArrivalHistory
    Output predictionMap
2:  for (every cloud layer Li in multi layer cloud application at the interval Δt) do
3:  begin
4:  request_numi=0
5:  requestMipsi =0
6:  totalMips=0;
7:  for t= current_time-n to current_time do
8:    requestsNumi,t= Num_of_request_of _layer_i(t)
9:  RequestMedianMipsi +=Median_Requests_MIPS(Li,t)
10: end
11: predictedRequestNumberi=Predict number of requests based on local regression
    according to the request numbers in the last n time intervals
12: RequestMedianMipsi= RequestMedianMipsi /n
13: predictedMipsUsagei= predictedRequestNumberi* RequestMedianMipsi
14: predictedUtilizationi=
    predictedMipsUsagei/(Number_Of_VMi*VM_MIPSi*Time_Interval)
15: : predictionMap.add(predictedUtilizationi)
16: End
        return predictionMap
```

۴-۲-۳ مرحلهٔ برنامه‌ریزی

در اینجا، یک روش مبتنی بر یادگیری بیزی را برای به‌دست آوردن اقدام بهینه پیاده‌سازی می‌کنیم. ابتدا احتمالات افزایش،کاهش و یا عدم تغییر مقیاس‌بندی را تعیین می‌کنیم. در صورتی‌که نیاز به منبع بیشتری داشته باشیم مقیاس بالا انجام می‌گیرد. زمانی نیاز به منابع بیشتر احساس می‌شود که میانگین بهره‌وری ماشین‌های مجازی فعلی از حد معینی بالاتر باشد. حال با استفاده از روش پیشنهادی مبتنی بر یادگیری ماشین که بر اصل تئوری بیز و الگوریتم یادگیری جامع (فراگیر)[40] استوار است، اقدام به تصمیم‌گیری برای مقیاس‌بندی بالا یا پایین بسته به شرایط بهره‌وری پردازنده، در بارکاری آتی می‌نماییم:

رابطه (۳) $P_{ScaleUp=}(\text{Predicted Utilization} - 0.8) * 0.5$

رابطه (٤) $P_{ScaleDown=}(0.5 - \text{Predicted Utilization}) * 2$

رابطه(۵) $P_{Nothing=}1 - P_{ScalUp} - P_{ScaleDown}$

مطابق رابطه(۳)،چنانچه بهره‌وری بیش از ۸۰ درصد باشد، ScaleUp انجام می‌شود. تخمین بهره‌وری پردازنده در مرحلهٔ قبل توسط مدل رگرسیون خطی انجام شد. برای رابطه (٤) نیز فرض کردیم ماشین‌های مجازی که بهره‌وری آن‌ها کمتر از ۵۰ درصد است نیاز به ScaleDown دارند. رابطه (۵) نیز حاصل تفریق احتمالات مقیاس بالا و پایین از احتمال ۱۰۰ درصد می‌باشد.

با استفاده از قضیه بیز احتمال اولیه فرضیه‌ها را به‌دست می‌آوریم که حاصل آن احتمال یک نوع مقیاس‌بندی در صورت مطلوب بودن برای هر لایه خواهد بود. این احتمال برای هر سه لایه و بار کاری سطح وب، برنامه کاربردی و پایگاه داده محاسبه می‌شود. مفروضات در این بخش بدین شکل می‌باشد:

$$P(\text{Scale Up}|+)=\{0.9,0.9,0.87\}$$
$$P(\text{Scale Down}|+)=\{0.97,0.95,0.93\}$$
$$P(\text{Nothing}|+)=\{0.8,0.75,0.7\}$$

در بیان ساده‌تر در هر بازهٔ زمانی برای هر لایه یکی از سه حالت ScaleDown، ScaleUp و یا Nothing اعمال می شود. بنابراین مطابق رابطه(٦) احتمال شرطی برای هر حالت به‌دست می‌آید:

رابطه (٦) $p(h|D) = \frac{p(h|D)P(h)}{p(D)}$

رابطه(۷) $P(+|\text{Scale Up})=P(\text{ScaleUp}|+)P(+)/P(\text{ScaleUp})$

رابطه(۸) $P(+|\text{Scale Down})=P(\text{ScaleDown}|+)P(+)/P(\text{Scal}$

رابطه(۹) $P(+|\text{Nothing})=P(\text{Nothing}|+)P(+)/P(\text{Nothing})$

در روابط سه‌گانهٔ فوق احتمالات مطلوبیت مقیاس‌بندی‌های مختلف به‌دست می آید. همچنین با توجه

40- Brute-Force MAP Learning

به روابط بـــالا، احتمال عـــدم مطلوبیـــت مقیاس‌بندی نیز به‌صورت ذیل به‌دست می‌آید:

رابطه(۱۰) $P(-|\text{Scale Up})=1-P(\text{Scale Up}\ |+)P(+)/P(\text{Scale Up})$

رابطه(۱۱) $P(-|\text{Scale Down})=1-P(\text{Scale Down}\ |+)P(+)/P(\text{Scale Down})$

رابطه(۱۲) $P(-|\text{Nothing})=1-P(\text{Nothing}|+)P(+)/P(\text{Nothing})$

در انتهـــا مطابق الگوریتـــم ٤ و روش یادگیری جامع (فراگیر)، ماکزیمم احتمالات محاسبه می‌شود.

Algorithm4: Pseudocode for planning phase

```
1:  Input predictionMap
    Output Actions
2   for (every cloud layer Li in multi layer cloud application at the interval Δt) do
3:  begin
4:  P(Scale Up)=( predictedUtilizationi -0.8)*5
5:  P(Scale Down)= (0.5- predictedUtilizationi)*2
6:  P(Nothing)=1- P(Scale Up)- P(Scale Down)
7:  P(+|Scale Up)=P(ScaleUp|+)P(+)/P(ScaleUp)
8:  P(+|ScaleDown)=P(ScaleDown|+)P(+)/P(ScaleDown)
9:  P(+|Nothing)=P(Nothing|+)P(+)/P(Nothing)
10: if P(+|Scale Up) > P(+|Scale Down) and P(+|ScaleUp)> P(+|Nothing)
11:  Scale up a VM I the Layer Li
12  else if P(+|ScaleDown) > P(+|Scale Up) and P(+|ScaleDown)> P(+|Nothing)
13: Scale Down a VM in the Layer Li
14: else
15: Do nothing!
```

۴-۲-۴- مرحلهٔ اجرا

در ایـــن مرحله، اقدامات یعنی مقیاس بـــالا یا پایین که در مرحلهٔ برنامه‌ریزی تعیین شـــده‌اند، توســـط مدیر VM به منظـــور درک موثر تامین منبع اجرا می‌شـــود و تعامل بین SLA و حداقل هزینه به‌دســـت می‌آیـــد. در الگوریتم ۵ نشان داده شده اســـت که متعادل کننده بار درخواست‌ها را برای لایه Li در بازه زمانی به های مناســـب بر اساس سیاست‌های متعادل‌سازی بار توزیع می‌کند.

Algorithm5: Pseudocode for Execute phase

```
1:  Begin
2:  Execute adaption actions (); /*scale up / scale down by VM manager*/
3:  Dispatch requests to VMs(); /* by Load balancer*/
4:  Sort requests based on deadline
5:  Allocation of requests on VMs of layer I based on BFD algorithm
6:  end
```

در ایـــن مرحلـــه، اعمـــال انطباقـــی(Scale Up/Down) حاصـــل از مرحلهٔ برنامه‌ریزی،بـــرای تحقق کارایی تامین منابـــع و همچنین رســـیدن به توافق ســـطح ســـرویس و کاهش هزینه، توســـط مدیر ماشین مجازی اجرا می‌شوند. درخواســـت‌های وارد شـــده در یک بازه زمانی، به شـــکل حریصانه[41] اجرا می‌شـــوند. در اینجا تعیین می‌کنیم که هر درخواست مشخص روی کدام ماشین مجازی قرار گیرد. لذا ترتیب درخواســـت‌ها از اهمیت بالایی برخوردار است. بنابراین درخواست‌ها برحســـب «ضرب‌الاجل زمانی»[42] و به‌صورت صعودی مرتب می‌شـــوند تا از تخطی از توافق ســـطح ســـرویس پیشـــگیری به عمل آید. در طرف مقابل ماشـــین‌های مجازی به‌صورت نزولی و بر اســـاس میزان بهره‌وری مرتب می‌شـــوند. در نتیجه ظرفیت ماشـــین‌های مجازی با ظرفیت کم‌تر، تکمیل شده و ماشین‌های مجازی با کمترین میزان بهره‌وری حذف می‌شوند. بنابراین تعداد ماشین‌های مجازی کاهش می‌یابد.

۵- شبیه‌سازی روش پیشنهادی و ارزیابی کارایی

در ایـــن بخـــش نتایج حاصـــل از پیاده‌ســـازی روش پیشـــنهادی برای تامیـــن منابع با اســـتفاده از ترکیب مدل رگرســـیون خطی و نظریهٔ بیز را بر اســـاس ســـه معیار هزینه، زمان پاسخگویی و میزان بهره‌وری مورد ارزیابی قرار گرفته اســـت. برای شبیه‌ســـازی از ابزار CloudSim استفاده شـــده و روش پیشنهادی را تحت بارکاری واقعی FIFA World Cup [۲۱] با روش‌های Stat-RA و DPM-RA با درنظر گرفتن معیارهای مختلف مورد ارزیابی قرار گرفته اســـت. در ادامه ارزیابی کارایی را در سه بخش، تنظیمات شبیه‌سازی، معیارهای کارایی و ارزیابی نتایج با جزئیات بیشتر ارایه شده است.

۵-۱-تنظیمات شبیه سازی

برای شبیه‌ســـازی دقیق و قابل توســـعه ابـــر از ابزار CloudSim استفاده کرده‌ایم. بدین دلیل که ابزار CloudSim با زبان جاوا نوشـــته شده است، محیط توسعه NetBeans به‌عنوان محیط برنامه‌نویسی جاوا انتخاب شده است.

به منظور شبیه‌سازی راهکار پیشنهادی ما یک برنامه ۳ لایه ابری را در نظر گرفته‌ایم که هر لایه دارای ویژگی‌های متفاوتی نســـبت به سایر لایه‌هاست. به‌عنوان مثال هر لایه نوع ماشـــین مجـــازی مجزا و نرخ ورود درخواســـت‌های متفاوت را داراســـت. جدول(۱) مشـــخصات کلی لایه‌های

41- Greedy

42- DeadLine

جدول ۱: مشخصات لایه‌های مختلف برنامه ابری

Layer	Avg. Request Arrival Rate	VM Type	Number of Startup VMs
Web	100%	Large	5
Application	70%	Mediun	5
Database	40%	Small	5

مختلف معماری سه لایه‌ای ابری را نشان می‌دهد.

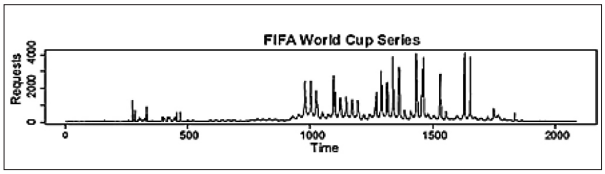
نرخ ورود درخواست‌ها در لایه‌های مختلف متفاوت است. به‌طور دقیق تر تعداد درخواست‌های وارد شده در لایه اول مطابق آنچه از بارکاری (FIFA) خوانده می‌شود، می‌باشد امّا تعداد درخواست‌های لایه دوم و سوم به ترتیب به‌طور میانگین برابر با ۷۰ و ٤٠ درصد درخواست‌های لایه اول می‌باشد.

همان‌گونه که دیده می‌شود، ماشین مجازی بزرگ به لایه اول (وب) که بیشترین حجم کار را داراست تخصیص داده شده است در حالی که ماشین مجازی با اندازه متوسط به لایه دوم و در نهایت کوچک‌ترین و کم قدرت ترین نوع ماشین مجازی به لایه آخر (پایگاه داده) که کمترین حجم درخواست‌ها را دریافت خواهد کرد اختصاص یافته است. هر چند تعداد ماشین‌های مجازی اولیه هر سه لایه یکسان (برابر با ٥) است امّا به دلیل تفاوت ظرفیت، قدرت و ماهیت ماشین‌های مجازی به‌کار گرفته شده در لایه‌های مختلف، قدرت محاسباتی لایه‌ها از همان ابتدا متفاوت خواهد بود. مشخصات ماشین‌های مجازی مختلف استفاده شده در این سه لایه را در جدول (۲) به‌طور کامل بیان کرده‌ایم.

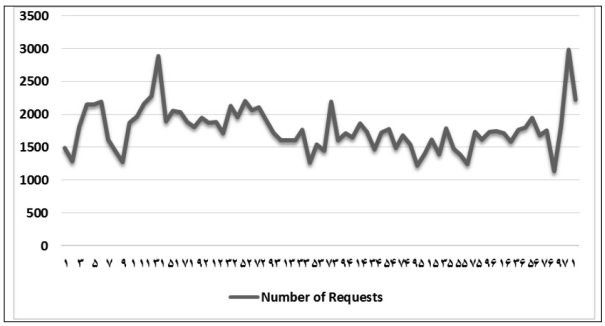
در این پژوهش به منظور ارزیابی کارایی راهکار پیشنهادی بار FIFA استفاده شده است. در واقع، این بار کاری، حجم درخواست‌های کاربران اینترنتی در محور زمان به این وبگاه مشهور را نشان می‌دهد. داده‌های FIFA شامل ۷ روز کاری و ۱۲۷۸٥٦ درخواست است. شبیه‌سازی در ۲٤ ساعت از ساعات پر درخواست است. نمودار(۱) نمایی از ساختار مجموعه داده واقعی را نشان می‌دهد. ما در این شبیه‌سازی به ترتیب ٦ ساعت از بارکاری به‌صورت ۷۲ بازه زمانی ٥ دقیقه‌ای استفاده کرده‌ایم.

جدول ۲: مشخصات ماشین‌های مجازی مختلف به‌کار گرفته شده

VM Type	Large	Medium	Small
CPU (MIPS)	3500	3000	2500
RAM (GB)	4	2	1
Price runtime($/hour)	1.28	0.64	0.32
VM Initialization($)	1.2	0.6	0.3



نمودار ۱: تعداد تقاضاهای بارهای کاری در محور زمان



نمودار ۲: بار کاری FIFA استفاده شده

۵-۲ معیارهای کارآیی

در این بخش به بررسی کارایی راهکارهای مختلف بر اساس معیارهای مختلف در لایه وب خواهیم پرداخت. تعداد ماشین‌های مجازی، بهره‌وری، زمان پاسخ، هزینه‌ها و میزان سود معیارهای اساسی مورد بررسی ما در این بخش می‌باشند. بدین منظور ما نتایج و تحلیل ارزیابی معیارهای مختلف را برای بار کاری FIFA ارائه داده‌ایم.

ما معیارهای زیر را برای مقایسه رویکرد پیشنهادی با سایر روش‌ها به‌کار گرفته و بدین صورت تعریف می‌کنیم:

الف- بهره‌وری[43]: بهره‌وری پردازنده برای ماشین‌های مجازی تخصیص داده شده به سرویس ابری S_i در بازه زمانی Δt، بر اساس تقسیم میزان MIPS تخصیص داده شده ماشین‌های مجازی برای اجرای درخواست‌های سرویس ابری S_i بر کل میزان MIPS ای که به‌طور بالقوه توسط ماشین‌های مجازی برای اجرای درخواست‌های

43- Utilization

سـرویس ابری در بازه زمانی قابل ارائه اسـت، تعریف میشود و بر اساس رابطه (۱۳) محاسبه می‌شود:

رابطه(۱۳) $$U_{S_i}(\Delta t)=\frac{Allocated\ MIPS\ (\Delta t)}{Total\ MIPS\ (\Delta t)}$$

ب- ماشین مجازی تخصیص داده شده[44]: این معیار به‌صورت تعداد ماشین‌های مجازی تخصیص داده شده به سرویس ابری در بازه زمانی تعریف می‌شود.

پ-تخطی از SLA[45]: تخطی از SLA برای سرویس ابری S_i در بـازه زمانی Δt، به‌صورت اختلاف بین زمان پاسـخ تعیین شـده در SLA کاربر و زمان پاسـخ واقعی (تجربه شـده) برای درخواست c سرویس ابری S_i در بازه زمانی Δt تعریف می‌شـود و بر اساس رابطه (۱۳) و رابطه (۱٤) محاسبه می‌شود:

رابطه (۱۳) $$SLA\ Violation_{S_i}(\Delta t)=\frac{1}{C}\sum_{c=1}^{C}SLA\ Violation(request_c)$$

رابطه (۱٤) $$SLA\ Violation(request_c)=\begin{cases}FT_c-DL_c & FT_c>DL_c\\ DL_c-AT_c & otherwise\\ 0 & \end{cases}$$

که AT_c زمان ورود درخواسـت c کاربر به سیسـتم، FT_c زمان اتمام درخواسـت c کاربـر و DL_c مهلت اتمام برای درخواست c کاربر می‌باشد.

ت- هزینـه کل[46]: این معیار به‌عنوان هزینه کل تحمیل شده به تامین کننده SaaS برای پردازش همه درخواست‌های سـرویس ابری در بازه زمانی تعریف می‌شود و شامل هزینه ماشـین مجازی و هزینه جریمه است که بر اساس رابطه (۱۵) محاسبه می‌شود:

رابطه (۱۵) $$Total\ \ Cost_{S_i}(\Delta t)=VM\ \ Cost_{S_i}(\Delta t)+Penalty\ \ Cost_{S_i}(\Delta t)$$

ث- سود[47]: این معیار به‌عنوان سود کسب شده توسط تامیـن کننده SaaS بـرای پردازش همه درخواسـت‌های سـرویس ابری در بازه زمانی تعریف شده که بر اساس رابطه (۱٦) محاسبه می‌شود:

رابطه(۱٦) $$Profit_{S_i}(\Delta t)=Budget_{S_i}(\Delta t)-Total\ \ Cost_{S_i}(\Delta t)$$

رابطه (۱۷) $$Budget_{S_i}(\Delta t)=\sum_{k=1}^{K}Budget(request_k)$$

کـه k معرف تعداد درخواسـت‌های به اتمام رسـیده سـرویس ابـری S_i در بازه زمانی Δt می‌باشـد. بنابراین، بودجه کل سـرویس ابری S_i در بازه زمانی برابر مجموع بودجه همه درخواست‌های به اتمام رسیده برای سرویس ابری S_i در بازه زمانی Δt می‌باشد.

در ابتدا مشاهده می‌شود که درصد میانگین بهره‌وری پردازنده در روش پیشنهادی و در بازه زمانی ۱ تا ٤ بیش از صد درصد می‌باشـد. این یعنی تعداد درخواسـت‌های رسیده به لایهٔ وب در روش ما بیش از ظرفیت منابع موجود می‌باشد. همین مسئله باعث افزایش زمان پاسخ نمودار (۵) به این ترتیـب که بازه‌های زمانـی کوچک‌تری در اختیار برنامـه ها برای اجرا قرار می‌گیـرد. از آنجا که بالا رفتن زمان پاسـخ موجب پرداخت جریمه می‌شود، لذا به هزینه کلی نیز افزوده می‌شود نمودار(٦). هزینه کلی ما ترکیبی از هزینهٔ اجرای ماشین‌های مجازی، راه‌اندازی آن‌ها و هزینهٔ جریمه می‌باشـد. با نگاهی به نمـودار(۷) می‌توان دریافت که قسـمتی از هزینهٔ کلی مربوط به راه‌اندازی ماشین‌های مجازی می‌باشد و از آنجا که تعداد ماشین‌های مجازی ما نمودار(۳) تا بـازهٔ زمانی ۱٦ به‌طور میانگین کاهش یافته، میزان سود افت چشـمگیری را نسبت به روش‌های دیگر نشان نمی‌دهد.

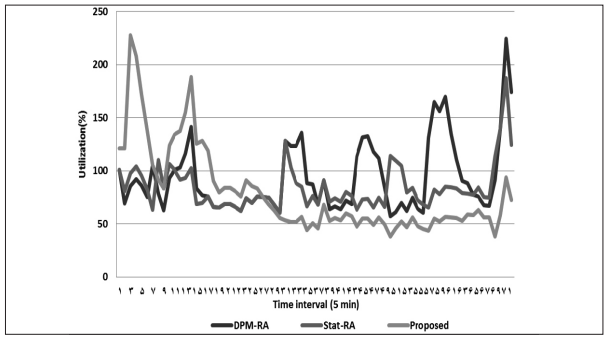
نکتـه حائز اهمیت در روش پیشـنهادی این اسـت که در ابتدای کار به جای افزایش تعداد ماشـین‌های مجازی با کاهش روبرو می‌شـویم. این یک اشـتباه است! از آنجا که روش پیشـنهادی بر مبنای احتمالات عمل می‌کند و ما تخمینی از گذشته نداریم، به همین دلیل معمولا در لایه‌های مختلف و در ابتدای کار ممکن اسـت با تخمین غلط روبرو شویم.

۵-۳ ارزیابی نتایج

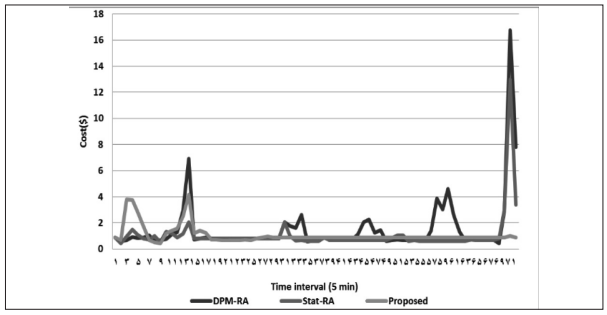
۵-۳-۱ ارزیابی معیارهای لایه برنامه کاربردی

در این بخش به بررسـی کارایی روش‌های مختلف بر اساس معیارهای مختلف در لایه برنامه کاربردی خواهیم پرداخت. تعداد ماشین‌های مجازی، بهره‌وری، زمان پاسخ، هزینه‌ها و میزان سـود معیارهای اساسی مورد بررسی

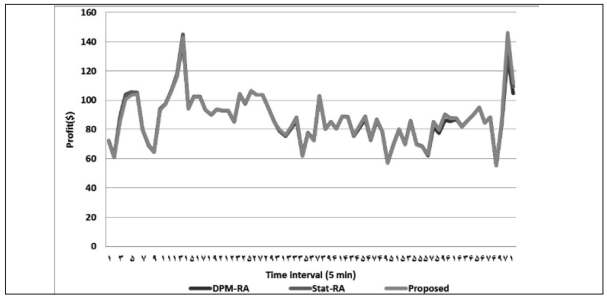
44- VM allocated
45- SLA violation
46- Total Cost
47- Profit



نمودار ۳: تعداد ماشین‌های مجازی لایه وب



نمودار ۵: میزان زمان پاسخ لایه وب

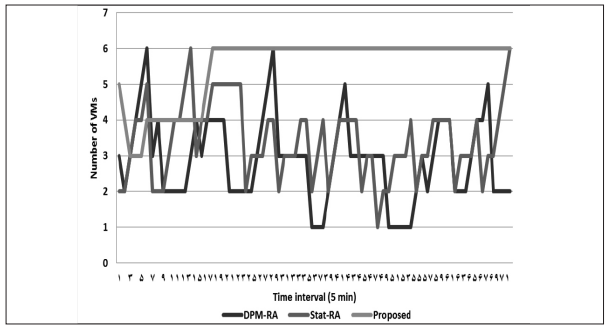


نمودار ۷: میزان سود لایه وب

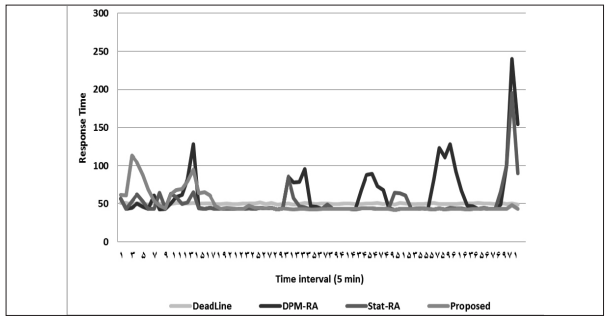
ما در این بخش می‌باشند. بدین منظور ما نتایج و تحلیل ارزیابی معیارهای مختلف را برای بار کاری FIFA ارائه داده‌ایم.

برای بار کاری FIFA تعداد ماشین‌های مجازی لایه دوم نسبت به لایه وب کمی بیشتر است. این اتفاق به‌دلیل افزایش تقاضاهای دریافتی لایه برنامه کاربردی نسبت به لایه وب افتاده است. به علاوه، سیاست تخصیص تعداد ماشین‌های مجازی مورد استفاده در لایه برنامه کاربردی نیز همانند آنچه در لایه وب اتفاق افتاده تبعیت می‌کند.

کاهش تعداد ماشین‌های مجازی نسبت به دو راهکار دیگر در بازه زمانی ۱ الی ۲۶، موجب افزایش سهمگین بهره‌وری در نمودار (۹) شده است. همین امر زمان پاسخگویی به



نمودار ۴: میزان بهره‌وری لایه وب

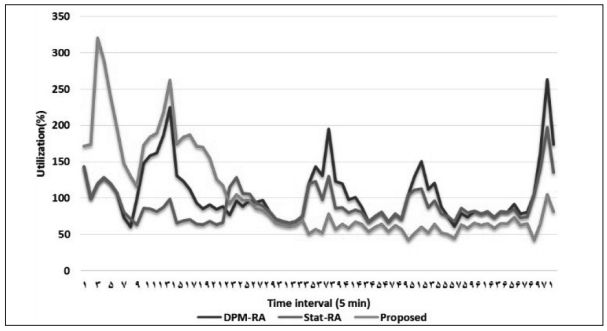


نمودار ۶: میزان هزینه ماشین مجازی لایه وب

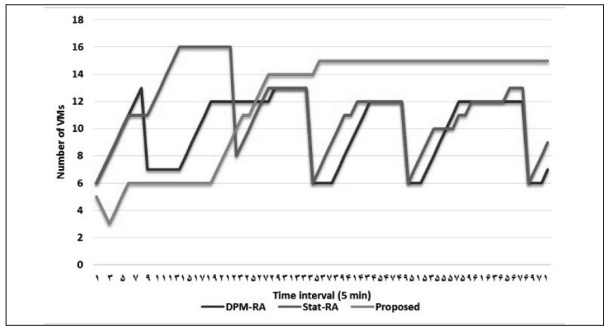
درخواست‌ها را در بازه زمانی ذکر شده طبق نمودار (۱۰) افزایش می‌دهد که به دلیل تخطی از توافق سطح سرویس منجر به پرداخت جریمه می‌گردد. باید توجه داشت که تعداد ماشین‌های مجازی زیاد همواره نامطلوب نیست. چراکه این اتفاق موجب کاهش بهره‌وری و در نتیجه زمان پاسخ به درخواست می‌شود. کاهش زمان پاسخ موجب کاهش هزینه کلی می‌گردد طبق نمودار (۱۱). که در نهایت بهبود سود را حاصل می‌شود طبق نمودار (۱۲).

۵-۳-۲ ارزیابی معیارهای پایگاه داده

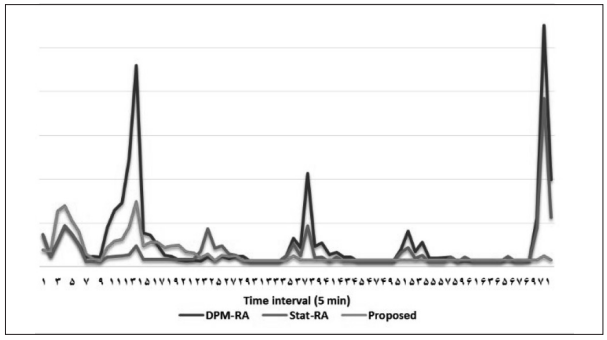
در این بخش اقدام به ارزیابی معیارهای راهکار ارائه شده برای لایه پایگاه داده تحت اجرای بارکاری FIFA خواهیم کرد پیشتر نیز گفته شد که تعداد درخواست‌های لایه پایگاه داده نه تنها کاهش یافته و بلکه می‌تواند متغیرتر از لایه‌های بالاتر باشد (انحراف معیار بیشتر). این قاعده منجر به عملکردی نسبی متفاوت راهکارها نسبت به دو لایه بالایی خواهد شد. نمودار (۱۳) نمودار تعداد ماشین‌های مجازی تخصیص یافته شده راهکارهای مختلف در لایه پایگاه داده را نشان می‌دهد. همان‌گونه که مشخص شده است، به دلیل کاهش تقاضاها در این لایه، به‌طور کلی



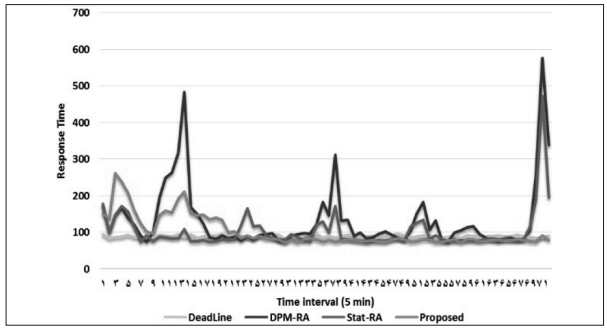
نمودار ۸: تعداد ماشین‌های مجازی لایه برنامه کاربردی



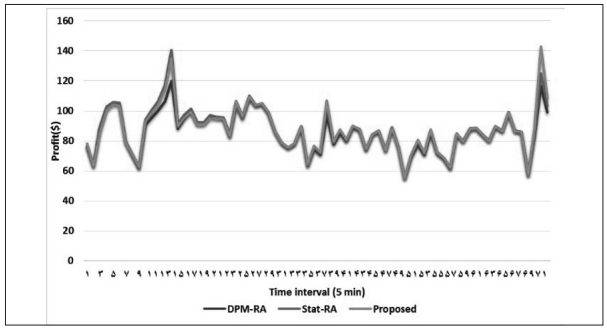
نمودار ۹: میزان بهره‌وری لایه برنامه کاربردی



نمودار ۱۰: میزان زمان پاسخ لایه برنامه کاربردی



نمودار ۱۱: میزان هزینه ماشین مجازی لایه برنامه کاربردی



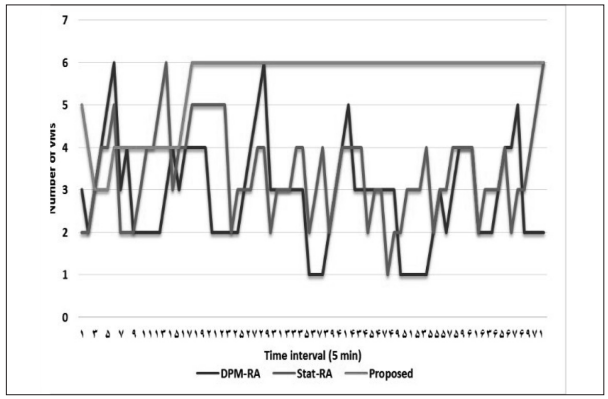
نمودار ۱۲: میزان سود لایه برنامه کاربردی

کمتریـــن تعداد ماشـــین‌های مجازی مورد اســـتفاده قرار گرفته اســـت. از آنجایی که تعداد درخواست‌های دریافتی در بازه‌هـــای زمانی مختلـــف در این لایه تابعـــی از تعداد درخواست‌های دریافتی در لایه‌های بالاتر است، روند ورود درخواست‌ها در این لایه کمی ناهمگون‌تر از لایه‌های قبلی اســـت.با این حال ماشـــین یادگیر به خوبی توانسته میزان ScaleUp/Down ها را در سطح معینی ثابت نگه دارد.

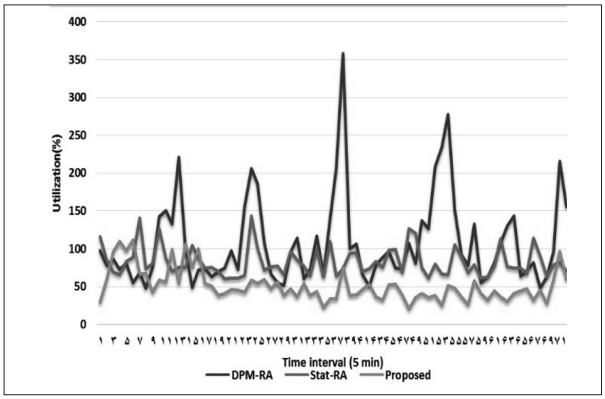
نمودار (۱٤) بهره‌وری ماشین‌های مجازی در لایه پایگاه داده را نشـــان می‌دهد. همان‌گونه که مشخص شده است، کاهش نامتوازن تقاضاها در این لایه، راهکار پیشـــنهادی را برخلاف ســـایر روش‌ها با چالش جدی روبرو نساخته اســـت. لذا، بهره‌وری روش ماشین یادگیر برای لایه پایگاه داده نیـــز همانند هر دو لایه بالاتر خود به شـــکل بهینه‌ای حول حالت تمام بهره‌ور درهر سه بار کاری به شکلی کاملا بهینه قرار دارد.

نمودار (۱۵) میزان زمان پاســـخ درخواست‌ها در لایه پایگاه داده برای بار کاری را نشـــان می‌دهد. همان‌گونه که مشخص شده است، کاهش نا همگون تقاضاها در این لایه، زمان پاسخ راهکار پیشنهادی را با چالش جدی مواجه نکرده است. از آنجا که تعداد درخواست‌های دریافتی در بازه‌های زمانی مختلف در این لایه تابعی از تعداد درخواســـت‌های دریافتی در لایه‌های بالاتر است، روند ورود درخواست‌ها در این لایه کمی نامتوازن‌تر از لایه‌های قبلی است. هرچند، زمان پاسخ روش ماشین یادگیر برای لایه پایگاه داده نیز همانند هر دو لایه بالاتر خود به ماشین‌های مجازی سعی در رعایت مهلت زمانی درخواست‌های مشتریان را دارد که نه تنها باعث کاهش هزینه ماشین‌های مجازی و بلکه باعث عدم جلوگیری از افزایش هزینه جریمه می‌شود.

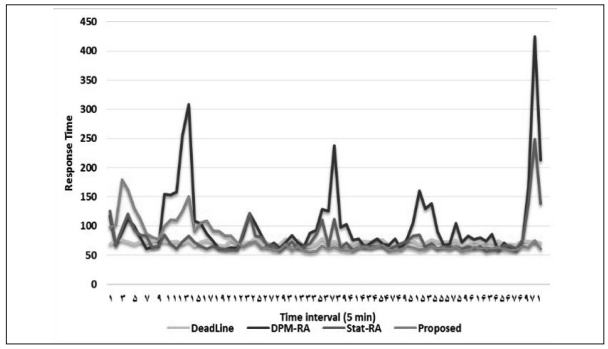
نمودار (۱٦) میـــزان هزینه در لایه پایگاه داده برای بار کاری FIFA را نشـــان می‌دهد. راهکار ماشـــین یادگیر سعی



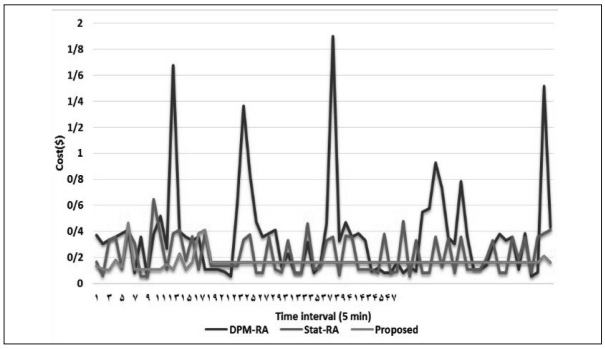
نمودار ۱۳: تعداد ماشین‌های مجازی لایه پایگاه داده



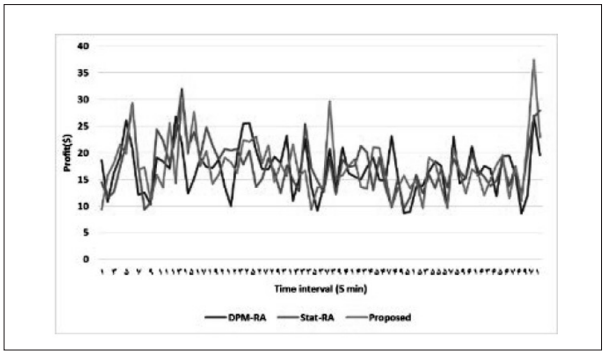
نمودار۱۴: میزان بهره‌وری لایه پایگاه داده



نمودار ۱۵: میزان زمان پاسخ لایه پایگاه داده



نمودار ۱۶: میزان هزینه ماشین مجازی لایه پایگاه داده



نمودار ۱۷: میزان سود لایه پایگاه داده

بر کاهش و ثبیت هزینه کلی دارد که نمودار مربوطه بیانگر موفقیت راهکار پیشنهادی نسبت به روش‌های دیگر می‌باشد.

نمودار (۱۷) میزان سود در لایه پایگاه داده را نشان می‌دهد. همان‌گونه که مشخص شده است. میزان سود روش ماشین یادگیر برای لایه پایگاه داده بهبود محسوسی (به دلیل انحراف معیار درخواست‌هایش) پیدا کرده است.

ارزیابی شاخص های هزینه و سود لایه‌های مختلف

در این بخش به بررسی هزینه‌های راه‌اندازی و اجرای ماشین‌های مجازی، جریمه و میزان سود کلی خواهیم پرداخت.

مطابق جدول (۳)، هزینه اجرای ماشین مجازی در روش پیشنهادی بیش از روش‌های دیگر می‌باشد که علت آن افزایش ماشین‌های مجازی توسط ماشین یادگیر می‌باشد. با این حال، هزینه راه‌اندازی راهکار پیشنهادی، به طرز قابل توجهی کمتر از دو روش دیگر شده است که علت آن را باید در کاهش چشمگیر مقیاس‌بندی بالا و پایین جست. همچنین مقدار جریمه نسبت به دو روش دیگر دارای بهبود می‌باشد که همین مهم روی هزینه کل و سود حاصل تاثیر مثبتی گذاشته است.

ارزیابی میانگین نتایج معیارهای کارایی هر سه لایه

همان‌گونه که در جدول (٤) نشان داده شده است، میانگن تعداد ماشین‌های مجازی مورد استفاده راهکار ارائه شده در بار کاری مورد استفاده بهبود(کاهش) یافته است.

در نتیجه استفاده از تعداد ماشین‌های مجازی بیشتر، بهره‌وری ماشین‌های مجازی راهکار ارائه شده نسبت به دیگر روش‌ها کاهش یافته است. طبق جدول (۵).

از آنجا که تعداد ماشین‌های مجازی استفاده شده جهت

جدول ۳: هزینه و سود حاصل شده برای راهکارهای مختلف در همه لایه‌ها

روش	هزینه زمان اجرا VM	هزینه راه‌اندازی VM	هزینه کل VM	نرخ جریمه	هزینه جریمه	بودجه درخواستی	هزینه کل	سود کل
MAPE-ML	۱۰۹,۳۲	۷,۴۲	۱۱۶,۷۴	۰,۱۶۷	۸۳,۱۸	۱۴۰۶۹,۰۳	۱۹۹,۹۳	۱۳۸۶۹
DPM-RA	۸۶,۲۶	۲۳	۱۰۹,۲۸	۰,۴۵۷	۲۲۵,۷۹	۱۴۰۳۶,۹۲	۳۳۵,۰۸	۱۳۷۰۱
Stat-RA	۹۵,۰۴	۲۴,۹۷	۱۲۰,۰۲	۰,۱۷۸	۸۸,۴۹	۱۴۰۷۰,۶۴	۲۰۸,۵۲	۱۳۸۶۲

جدول ۴: میانگین تعداد ماشین‌های مجازی مورد استفاده راهکارهای مختلف

روش	MAPE-ML	DPM-RA	Stat-RA
FIFA	۹,۱	۶,۹	۷,۷۳

سرویس‌دهی به درخواست‌های مشتریان در راهکار ارائه شده بیش از ماشین‌های مجازی سایر راهکارهاست، منجر به کاهش بهره‌وری و تعداد درخواست‌های تخصیص یافته به هر ماشین مجازی می‌شود. در نتیجهٔ کاهش تعداد درخواست‌های تخصیص یافته به هر ماشین مجازی، میانگین زمان پاسخ گویی به درخواست‌ها نیز نسبت به سایر راهکارها کاهش داشته است جدول (٦).بنابراین راهکار پیشنهادی با در پیش گرفتن سیاست کاهش زمان پاسخ، به بهبود سود می‌انجامد.

بر اساس جدول (۷)، اگرچه به نظر می‌رسد با افزایش تعداد ماشین‌های مجازی راهکار ارائه شده متحمل هزینه بیشتری می‌شود امّا با پاسخی قابل قبول با میانگین زمان پاسخی مطلوب در مقایسه با سایر روش‌ها و جلوگیری از جریمه‌های غیر معقول، منجر به کاهش هزینه کلی فراهم کننده ابر می‌شود. در نتیجه، راهکار ارائه شده در هر بازه زمانی باعث بهبود قابل توجهی در هزینه‌ها شده است.

در نتیجه آنچه برای سایر معیارها بیان شد، به‌ویژه با کاهش محسوس هزینه راهکار ارائه شده، میانگین میزان سود در هر بازه زمانی حاصل راهکار پیشنهادی نسبت به سایر روش‌ها در بارکاری و درنتیجه بهبود یافته است که در جدول (۸) ارائه شده است.

۶- نتایج و پیشنهادها

در این مقاله به ارائه رویکردی نوین برای تامین کارآمد منابع در محیط‌های ابری و همچنین شبیه‌سازی این روش

جدول ۵: میانگین بهره‌وری ماشین‌های مجازی مورد استفاده راهکارهای مختلف

روش	MAPE-ML	DPM-RA	Stat-RA
FIFA	۷۷,۴۳	۱۰۲,۳	۸۵,۶۸

جدول ۶: میانگین زمان پاسخ درخواست در راهکارهای مختلف

روش	MAPE-ML	DPM-RA	Stat-RA
FIFA	۱۷۶,۹	۲۲۴,۵	۲۰۹,۵

نوین تحت بار کاری واقعی FIFA و به ارزیابی در عواملی شامل، تعداد ماشین‌های مجازی، بهره‌وری، هزینه مجموع، زمان پاسخ و سود در لایه‌های مختلف پرداختیم و در نهایت روش پیشنهادی MAPE-ML با روش‌های DPM و Stat-RA مقایسه شد.

میانگین تعداد ماشین‌های مجازی مورد استفاده راهکار ارائه شده در بار کاری به‌صورت میانگین، روند افزایشی داشته که در نتیجه آن بهره‌وری پردازنده ماشین‌های مجازی و به تبع آن زمان پاسخگویی به درخواست کاربران کاهش چشمگیری یافته است. هر چند با افزایش تعداد ماشین‌های مجازی برای لایه‌های مختلف در راهکار پیشنهادی، به نظر می‌رسد که با افزایش هزینه روبرو می‌شویم، امّا بهبود قطعی زمان پاسخ و پیشگیری از جریمه‌های گزاف، ماشین یادگیر پیشنهادی را با هزینه کلی قابل قبول نسبت به راهکارهای دیگر روبرو می‌سازد. ماشین یادگیر با بهبود پلکانی عملکرد و سعی در ثابت نگه داشتن تعداد ماشین‌های مجازی، نسبت به افزایش سود حاصل اهتمام می‌ورزد.

همچنین راهکار ارائه شده به‌صورت میانگین در هر بازه زمانی باعث بهبود چشمگیری در هزینه‌ها شده است. از آنچه حاصل گردید میانگین حاصل از نتایج شبیه‌سازی،

جدول ۷: میانگین هزینه در هر بازه زمانی در راهکارهای مختلف

روش	MAPE-ML	DPM-RA	Stat-RA
FIFA	۰,۹۴	۱,۵۲	۰,۹۴

جدول۸: میانگین سود حاصل از استفاده از راهکارهای مختلف

روش	MAPE-ML	DPM-RA	Stat-RA
FIFA	۶۴,۲۰	۶۳,۴۳	۶۴,۱۷

نشـــان داد که راهکار ارائه شده در مقایسه با بهترین نتایج به‌دســـت آمده از راهکارهای پیشین، منجر به افزایش تعداد ماشـــین‌های مجـــازی به میـــزان ۱۰ درصد بـــا بهبود نرخ مقیاسبندی، کاهش ۸ درصدی میانگین بهره‌وری، کاهش ۳ درصدی زمان پاسخ‌دهی، در نتیجه کاهش ۵ درصدی هزینه تمام شده و افزایش ۱ درصدی سود حاصل شده است.

بـــا توجه به نتایج به‌دســـت آمـــده ازمقالـــه حاضر و برای ارزیابی بیشـــتر نتایج و تکمیل و توسعه این تحقیق، پیشنهادهای ذیل ارائه می‌گردد: بهینه نمودن بیشتر روش تامیـــن خـــودکار منابع در محیـــط رایانش ابـــر و اصلاح پارامترهـــا، تعمیم ایـــن تکنیک در مجموعه‌های رایانشـــی بزرگتـــر و پیچیده تر، ارزیابی و بهینه ســـازی معیارهای دیگر همچـــون حافظه و پهنای باند، اصـــلاح حلقه کنترلی MAPE-K در مرحلـــه تحلیل با اســـتفاده از روش ماشـــین بردار پشـــتیبان[۴۸]، ترکیـــب روش Stat-RA و MAPE-ML، ترکیب روش DVM-RA [۷] و MAPE-ML

مراجع:

[1]. Ashouraie, M. and Jafari Navimipour, N., 2015. "Priority-based task scheduling on heterogeneous resources in the Expert Cloud". Kybernetes, 44(10), pp.1455-1471.

[2]. Kupferman, J., Silverman, J., Jara, P. and Browne, J., 2009. "Scaling into the cloud". CS270-advanced.

[3]. Lorido-Botran, T., Miguel-Alonso, J. and Lozano, J.A., 2014. "A review of auto-scaling techniques for elastic applications in cloud environments". Journal of Grid Computing, 12(4),pp.559-592.

[4]. Jacob, B., Lanyon-Hogg, R., Nadgir, D.K. and Yassin, A.F., 2004. "A Practical Guide to the IBM Autonomic Computing Toolkit", IBM Redbooks.

[5].Yang, J., Liu, C., Shang, Y., Cheng, B., Mao, Z., Liu, C., Niu, L. and Chen, J., 2014. "A cost-aware auto-scaling approach using the workload prediction in service clouds". Information Systems Frontiers, 16(1), pp.7-18.

[6]. Bi, J., Yuan, H., Tie, M. and Tan, W., 2015. "SLA-based optimisation of virtualised resource for multi-tier web applications in cloud data centres". Enterprise Information Systems, 9(7), pp.743-767.

[7]. Wu, H., Zhang, W., Zhang, J., Wei, J. and Huang, T., 2013. "A benefit-aware on-demand provisioning approach for multi-tier applications in cloud computing". Frontiers of Computer Science, 7(4), pp.459-474.

[8]. Urgaonkar, B., Shenoy, P., Chandra, A., Goyal, P. and Wood, T., 2008. "Agile dynamic provisioning of multi-tier internet applications". ACM Transactions on Autonomous and Adaptive Systems (TAAS), 3(1), p.1.

[9]. RahimiZadeh, K., AnaLoui, M., Kabiri, P. and Javadi, B., 2015. "Performance modeling and analysis of virtualized multi-tier applications under dynamic workloads". Journal of Network and Computer Applications, 56, pp.166-187.

[10]. Beltrán, M., 2015. "Automatic provisioning of multi-tier applications in cloud computing environments". The Journal of Supercomputing, 71(6), pp.2221-2250.

[11]. Singh, S. and Chana, I., 2015. "Q-aware: Quality of service based cloud resource provisioning". Computers & Electrical Engineering, 47, pp.138-160.

[۱۲]. کربلائی مهدی ســمانه، قبائی آرانی مصطفــی، «رویکرد تامین منبع بهبود یافته برای برنامه‌های کاربردی چندلایه با اســتفاده ازسیســتم استنتاج عصبی-مرحله‌ای تطبیق پذیر در محیط رایانش ابری»، مجله علوم رایانشــی شماره ۱۳، تابستان ۱۳۹۸

[13]. Khorsand, R., Ghobaei-Arani, M., & Ramezanpour, M. (2018)." FAHP approach for autonomic resource provisioning of multitier applications in cloud computing environments". Software: Practice and Experience, 48(12), 2147-2173.

[14] Aslanpour, M. S., Ghobaei-Arani, M., & Toosi, A. N. (2017). "Auto-scaling web applications in clouds: A cost-aware approach". Journal of Network and Computer Applications, 95, 26-41.

[15] Guo, J., Li, C., Chen, Y., & Luo, Y. (2020). On-demand resource provision based on load estimation and service expenditure in edge cloud environment. Journal of Network and Computer Applications, 151, 102506.

[۱۶]. اسدی جواد، ناظمی اسلام،"بهبود معماری خودالتیامی در معماری سرویس گرا با استفاده از منطق مرحله‌ای" مجله علوم رایانشی شماره ۴، بهار ۱۳۹۶

[17] Li, W. and Tan, X., 2020." Locally Bayesian learning in networks". Theoretical Economics, 15(1), pp.239-278.

[18] Alpaydin, E., 2020. "Introduction to machine learning". MIT press.

[19] Barry, D.K., 2003. The Savvy Manager's Guide to Web Services and Service-Oriented Architectures.

[20] Liu, X., Heo, J., Sha, L. and Zhu, X., 2006, April. Adaptive control of multi-tiered web applications using queueing predictor. In Network Operations and Management Symposium, 2006. NOMS 2006. 10th IEEE/IFIP (pp. 106-114). IEEE.

[21]. http://ita.ee.lbl.gov/html/contrib/WorldCup.html

48- Support Vector Machine