# به‌کارگیری الگوریتم جهش قورباغه در یافتن تخطی‌ها در کنترل تعهدات SLA در محاسبات ابری

زهرا رحیمی[۱]، فاطمه سعادت‌جو[۲]، علی محمد اسمعیلی زینی[۳]

[۱] دانشجوی کارشناسی ارشد ، دانشکده فنی و مهندسی، دانشگاه علم و هنر یزد، یزد، zahra.rahimi8686@gmail.com
[۲] استادیار، دانشکده فنی و مهندسی، دانشگاه علم و هنر یزد، یزد، saadatjou@sau.ac.ir
[۳] دانشکده فنی و مهندسی، دانشگاه علم و هنر یزد، یزد، esmailizaini@gmail.com

## چکیده

**محاسبات ابری شیوه‌ای از محاسبات کامپیوتری در فضایی است که قابلیت‌های مرتبط با فناوری اطلاعات به عنوان سرویس یا خدمات را به کاربر عرضه می‌کند. در کنار مزایای فراوانی که ابر در اختیار کاربرانش قرار می‌دهد مشکلات و خطراتی هم دارد. در صورت بروز خطا وجود توافقنامه سطح سرویس می‌تواند به کاربران ابر در اثبات ادعای خود علیه سرویس دهنده کمک کند. توافقنامه قراردادی بین سرویس دهنده ابر و مشتری است و تضمین کیفیت خدمات را از سوی سرویس دهنده ابر تضمین می‌کند که اگر رعایت نشود سرویس دهنده می بایست در قبال تخلفاتش جریمه پرداخت کند.**

**در این مقاله برای بررسی تخلف از توافقنامه، روشی پیشنهاد داده‌ایم که هدف آن یافتن بهترین بازه زمانی است که در آن بیشترین تخطی از توافقنامه، کشف شده است و در عین حال، این کشف موجب افزایش سربار زمانی تحمیلی به سرویس دهنده نیز نشده باشد. برای دستیابی به این هدف لازم است یک تعادل بین هزینه کشف تخطی و شمارش بیشترین تعداد تخطی‌ها صورت بگیرد. برای تشخیص تخطی، یک روش با هدف پیدا کردن بازه‌های زمانی پویا با تعداد بیشتری از SLA پیشنهاد شده است. برای انجام این کار چندین آزمایش با روش‌های مبتنی بر الگوریتم جهش قورباغه (SFLA)، و الگوریتم ژنتیک انجام شده است. نتایج نشان می دهد که الگوریتم مبتنی بر SFLA می‌تواند بهتر از الگوریتم ژنتیک عمل نماید.**

## کلمات کلیدی

**الگوریتم بهینه سازی جهش قورباغه، محاسبات ابری، تعهدات سطح سرویس**

## ۱- مقدمه

محاسبات ابری از ترکیب دو کلمه رایانش و ابر[۱] ایجاد شده است. کلمه ابر، از مفهوم شبکه‌های مخابراتی گرفته شده است. در اینجا استعاره از شبکه یا شبکه‌ای از شبکه‌های وسیع مانند اینترنت است. نرم‌افزارهای کاربردی و اطلاعات روی سرورها ذخیره می‌گردند و بر اساس تقاضا در اختیار کاربران قرار می‌گیرد [۱].

در محاسبات ابری[۲] یک ابر، گروهی از کامپیوترهای توزیع‌شده است که منابع و سرویس‌های محاسباتی را به‌صورت درخواستی برای کاربران راه دور در شبکه فراهم می‌کند. دریک ابر، خدمات و منابع برای کاربران به‌صورت اجاره‌ای فراهم می‌شود و کاربران می‌توانند منابع را با استفاده از راه‌حل‌های شبیه‌سازی رایگان و مؤثر مثل Xen hypervisor[۳] بدون خطر کنترل کنند. یکی از مزایای ابرها این است که ظرفیت‌های محاسباتی آماده‌شده برای کاربران نهایی، مؤثر و قابل‌انعطاف هستند [۲].

منابع محاسباتی بر اساس یک الگو توافقنامه سطح سرویس [۴]تعریف شده‌اند این الگو حاوی اطلاعاتی از جمله خصوصیات خدمات، نام و مقادیر آن است. بنا به تسهیل ایجاد این توافقنامه و مدیریت آن، الگوهای توافقنامه سطح سرویس تعریف‌شده‌اند. این الگوها فرمت‌های توافقنامه سطح سرویس متداول را ارائه می‌دهند و متشکل از عناصری مانند نام سازمان تجاری و مقادیر ویژگی هستند. پارامترهای توافقنامه سطح سرویس شامل پهنای باند ورودی و خروجی و پردازنده، حافظه است [۳].

بازه‌های زمانی ایستا بازه‌هایی هستندکه در محیط برنامه و در زمان اجرا دارای مقدار ثابتی بین دو زمان دلخواه می‌باشد و در طول اجرای برنامه تغییر نمی‌کند. برعکس بازه‌های زمانی پویا بازه‌هایی هستند که در محیط برنامه و در زمان اجرا مقدارشان بین دو بازه زمانی مداوم در حال تغییر است تا بازه مناسب انتخاب شود. بازه مناسب بازه‌ای است‌که بتواند تعداد تخطی‌های بیشتری را نسبت به بازه ایستا پیدا کند ضمن اینکه این کار از نظر زمانی مقیاس‌پذیر باشد. انتخاب بازه‌های ایستا باعث می‌شود تا بعضی از تعهدات طرفین نقض گردد [۴].

## ۲- کارهای مرتبط

Bonifice و همکاران ارائه خدمات را به صورت دینامیک، با استفاده از GRIA[۵] مورد بحث قرار داده‌اند. ارائه خدمات بر اساس توافقنامه های سطح سرویس توافق شده و مدیریت توافقنامه‌های سطح سرویس برای جلوگیری از تخطی انجام شده است. این روش تنها محیط‌های توری را در نظر می‌گیرد و نه ابری. به‌علاوه، نحوه نظارت بر معیارهای سطح پایین و نگاشت به توافقنامه سطح سرویس که سطح بالا را در زمان اجرا تقویت می کند، مورد بررسی قرار نمی‌گیرد[۵].

Emeakaroha و همکاران معماری زیرساخت تشخیص نقض توافقنامه Desvi را ارائه دادند که نقض توافقنامه را از طریق نظارت بر منابع پیچیده می‌سنجد. بر اساس درخواست کاربر، Desvi منابع محاسباتی را برای یک سرویس درخواست شده اختصاص و استقرار خود را در یک محیط مجازی ترتیب می‌دهد. منابع با استفاده از یک چارچوب جدید نگاشت معیارهای منابع سطح پایین به معیارهای توافقنامه سطح سرویس تعریف‌شده توسط کاربر نظارت می‌شوند. تشخیص نقض توافقنامه‌های ممکن به اهداف سطح خدمات از پیش تعریف‌شده و استفاده از پایگاه داده‌های دانش برای مدیریت و جلوگیری از چنین نقضی متکی است [۳].

Wood و همکاران یک سیستم به نام Sandpiper ارائه کردند که فرآیند نظارت و کشف نقاط حساس و نیز نگاشت و پیکربندی، ماشین‌مجازی را در زمان لازم انجام می‌دهد. این سیستم نظارت، یادآور اهداف اجتناب از تخطی توافقنامه سطح سرویس است. Sandpiper از آستانه‌هایی برای بررسی تخطی توافقنامه سطح سرویس استفاده می‌کند، اما این کار نگاشت معیارهای سطح پایین مانند CPU و حافظه را به پارامترهای توافقنامه سطح سرویس در سطوح بالا مانند زمان پاسخ در اجرای توافقنامه سطح سرویس را در نظر نمی‌گیرد[۷].

Commuzi و همکاران تعریف فرآیندی برای اجرای توافقنامه سطح سرویس بدون فریم ورک SOIV۶ و توافقنامه سطح سرویس پروژه اتحادیه امریکا تعریف کردند. این نویسندگان، یک معماری برای نظارت توافقنامه‌های سطح سرویس ارائه دادند که دو نیاز معرفی شده به وسیله تثبیت توافقنامه سطح سرویس را بیان می‌کند. یکی قابلیت دسترسی اطلاعات قدیمی برای ارزیابی پیشنهادهای توافقنامه سطح سرویس و دیگری ارزیابی قابلیت آن برای نظارت در مواردی که در یک توافقنامه سطح سرویس پیشنهاد می‌شود. اما آن‌ها نظارت بر معیارهای سطح پایین را مورد توجه قرار نداده و آن‌ها را به موارد در سطح بالا نگاشت ننمودند[۸].

Tordsson و همکاران روش مبادله ابری ارائه داده‌اند که قرارگیری زیرساخت‌های مجازی در سراسر ابرهای متعددی را بهینه‌سازی و استقرار و مدیریت اجزاء زیرساخت‌ها را در این ابرها جدا می‌کند. نتایج تجربی تأیید می‌کنند که استقرار چند ابر در مقایسه با استفاده از تنها یک ابر عملکرد بهتر و هزینه‌های پایین‌تری را فراهم می‌کند؛ اما کار آن‌ها از زمان‌بندی پویا پشتیبانی نمی‌کند [۹].

Sarvanan و همکاران برای کاهش نقض تعهدات توافقنامه سطح سرویس به‌عنوان یک نیاز جهت بالا بردن کیفیت سرویس و راضی کردن مشتریان تمرکز کرده‌اند. روش Bee-MMT استفاده‌شده در آن مقاله با توجه به کاهش مصرف برق باعث نقض بیشتر توافقنامه سطح سرویس نسبت به روش‌های دیگر شده است. لذا با استفاده از مدل کاهشی SALMON[۶] و ADA[۷] به کنترل خودکار و تجزیه‌وتحلیل توافقنامه سطح سرویس از محدودیت‌های ارضای مسئله در آن مقاله پرداخته‌شده است [۱۰].

سارا زنگنه و احمد فراهی یک مکانیزم هوشمند جهت یافتن بازه‌های زمانی پویا جهت نظارت و بهینه کردن تضاد منافع در معماری Desvi ارائه دادند. فرآیند اندازه‌گیری شامل نظارت بر همه ماشین‌های مجازی، پردازش داده‌های نظارت‌شده، نگاشت معیارهای سطح پایین به سطح بالا، توافقنامه سطح سرویس و ارزیابی تابع هدف است. رویکرد مقاله با توجه به مصرف منابع و معیارهای سطح توافقنامه در هر بار تکرار اجرا، تعداد اندازه‌گیری متفاوت اما نزدیک را نشان می‌دهد. هرچه بازه‌های نظارتی نزدیک‌تر باشند احتمال اینکه نقضی بروز کند و دیده نشود کمتر خواهد بود. این مسئله به‌خوبی از نتایج قابل استنباط است. هرچه بازه‌های نظارتی کوتاه‌تر باشد تعداد نقض توافقنامه‌ای که دیده نمی‌شوند کاهش می‌یابد، ولی سربار واردشده به سیستم افزایش می‌یابد[۱۱].

ناهید غفوری و فاطمه سعادت جو، با استفاده از الگوریتم رقابت استعماری تضاد منافع میان سربار سیستم و تخطی در محاسبات ابری جهت یافتن بازه های پویا در معماری Desvi ارائه دادند. برقراری تعادل نسبی بین زمانی که بازه‌ها بلند هستند و زمانی که بازه‌ها از بین بازه‌های کوتاه انتخاب می‌شوند از مزایا و عدم پیاده سازی در محیط کلودسیم از معایب آن مقاله به شمار می‌رود[۱۲].

## ۳- الگوریتم جهش قورباغه(SFLA)

الگوریتم جهش قورباغه[۱۱] اولین بار در سال ۲۰۰۳ توسط لنزی و یوسف ایجاد شد. این الگوریتم مشابه GA و PSO یک الگوریتم بهینه سازی مبتنی بر جمعیت است که می‌توان از آن برای حل مسائل بهینه سازی پیچیده‌ای که غیرخطی، تشخیص‌ناپذیر و چند وجهی هستند استفاده کرد.در این الگوریتم گروهی از قورباغه‌ها (مجموعه‌ای از جواب‌ها) به چندین زیرمجموعه تقسیم میشوند، که هر قورباغه فرهنگ مختص به خود را دارد و می‌تواند از فرهنگ‌ها یا ایده‌های قورباغه های دیگر در طول روند تکامل استفاده کند[۱۳] .

مراحل اجرای الگوریتم به شرح زیر است:

### ۳- ۱- تولید جمعیت اولیه

جمعیت اولیه (قورباغه‌ها) به صورت تصادفی بین بازه مساله با توجه به رابطه (۱) تولید می‌شود:

$$x_i = x_i^1 + \text{rand} \times (x_i^h - x_i^1) \quad (۱)$$

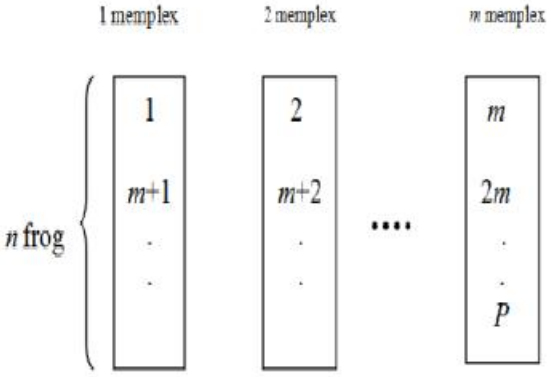
که در آن $x_i$ مقدار معادل هر عضو از جمعیت، $x_i^1$ کران پایین متغیر $x_i$، $x_i^h$ کران بالای متغیر $x_i$ و rand یک مقدار تصادفی بین ۰ الی ۱است.

هر قورباغه نمایانگر یک راه‌حل قابل قبول مساله بهینه‌سازی است که دارای یک مقدار شایستگی است. قورباغه‌ها بر اساس شایستگی‌شان به صورت نزولی مرتب می‌شوند و بر اساس روندی خاص به زیر مجموعه‌های مختلف تقسیم می‌شوند.

## ۳-۲- دسته‌بندی قورباغه‌ها

فرض کنید که جمعیت اولیه با p قورباغه تولید شده است. با توجه به شکل (۱)، p قورباغه‌ها به m مجموعه تقسیم می‌شوند. روند تقسیم‌بندی قورباغه‌ها بدین صورت است که قورباغه اول به مجموعه اول، قورباغه دوم به مجموعه دوم و قورباغه m به مجموعه mام و قورباغه m+1 به مجموعه اول تعلق دارند. این روند به صورت مشابه تا قورباغه آخر تکرار می‌شود. هر مجموعه m شامل n قورباغه است به طوری که رابطه (۲) نشان داده شده است:

$$P = m \times n \qquad (۲)$$




**شکل (۱): دسته‌بندی قورباغه‌ها به m زیرمجموعه[۱۳]**

## ۳-۳- مراحل جستجوی محلی الگوریتم SFLA

در هر مجموعه موقعیت قورباغه iام، بر اساس اختلاف بین قورباغه بهتر (با بهترین شایستگی $x_b$) و قورباغه بدتر (با بدترین شایستگی $x_w$) با استفاده از رابطه (۳) به دست می‌آید:

$$D(i) = \text{rand}() \times (x_b - x_w) \qquad (۳)$$

که rand() یک عدد تصادفی یکنواخت بین ۰و۱ است. موقعیت جدید قورباغه توسط رابطه (۴) به دست می‌آید که $D_{max}$ ماکزیمم تغییراتی است که در موقعیت قورباغه اعمال می‌گردد.

$$X_w = current\ position\ X_w + D_i \qquad (۴)$$
$$-D_{max} \le D_i \le D_{max}$$

اگر این تغییر موقعیت، قورباغه‌ای با شایستگی بهتر تولید کرد این قورباغه جایگزین قورباغه بدتر می‌شود. در غیر این صورت، قورباغه با بهترین شایستگی درکل جمعیت (بهینه فرامحلی) $X_g$ جایگزین $X_b$ در معادله (۳) شده و قورباغه جدیدی تولید می‌شود. در غیر این صورت، قورباغه‌ای جدید به صورت تصادفی تولید و جایگزین بدترین قورباغه می‌شود. روند تکامل در هر مجموعه تا *زمانی که* به معیار توقف برسد ادامه می‌یابد[ ۱۳].

## ٤- الگوریتم پیشنهادی

نظارت کافی و کشف تخطی از توافقنامه سطح سرویس امری مهم و چالش‌برانگیزی است [۶]. در این مقاله برای بررسی تخلف از توافقنامه سطح سرویس، روشی پیشنهاد داده که هدف آن یافتن بهترین بازه زمانی است که در آن بیشترین تخطی از توافقنامه سطح سرویس، کشف شده است و در عین حال، این کشف موجب افزایش سربار زمانی تحمیلی به سرویس دهنده نیز نشده باشد. برای دستیابی به این هدف لازم است یک تعادل بین هزینه کشف تخطی و شمارش بیشترین تعداد تخطی‌ها صورت بگیرد.

روش انجام کار این است که در ابتدا می‌بایست برای هرکدام از پارامترهای توافقنامه سطح سرویس که عبارتند از پردازنده، حافظه و پهنای باند، بار کاری بر مبنای بار کاری پیش‌فرض کتابخانه کلودسیم تولید کرد. کلودسیم داده‌های خود را از آزمایشگاه پلنت‌لب دریافت می‌کند که این داده‌ها ۵ دقیقه‌ای یک‌بار به‌صورت ایستا تخطی‌ها را بررسی می‌کند. این روش برای بررسی تخطی در بازه‌های پویا کاربرد ندارد برای بررسی تخطی‌ها به‌صورت پویا باید بین هر یک از این ۵ دقیقه‌ها یک سری بازه‌های تصادفی (m) تولید شود که تخطی‌ها را هر m دقیقه یک‌بار بررسی کند. سپس بر اساس این بار دستگاه تابع هزینه طبق رابطه (۵) توسط الگوریتم جهش قورباغه محاسبه و بررسی می‌شود که آیا هزینه قابل‌قبول هست یا نه؟ درصورتی‌که هزینه وارده قابل‌قبول باشد بازه انتخابی بدون تغییر می‌ماند و اگر هزینه وارده زیاد باشد بازه مناسب بعدی انتخاب می‌گردد.

$$C = \mu \times Cm + \Sigma_{\varphi \epsilon [CPU, Memory, Storage]}\ \alpha(\varphi) \times CV \qquad (۵)$$

در رابطه (۵) $\mu$ تعداد اندازه‌گیری‌ها، $Cm$ هزینه اندازه‌گیری، $\alpha(\varphi)$ تعداد تخطی‌های کشف نشده توافقنامه سطح سرویس و $CV$ هزینه از دست رفتن تخطی توافقنامه است.

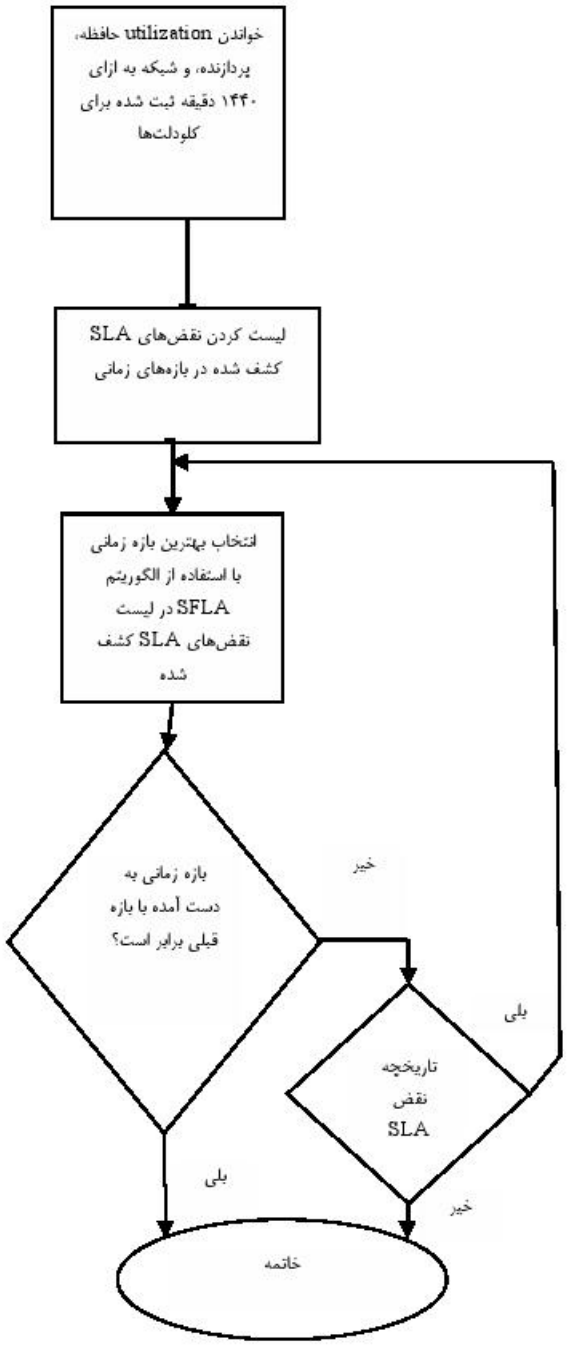
در روش پیشنهادی فرض شده است که مقدار بازه برحسب دقیقه باشد. این فرض یعنی از محاسبه برحسب ثانیه چشم‌پوشی می‌کنیم. چرا که به نظر می‌رسد که بار دستگاه تحمیل‌شده به سیستم برای بررسی ثانیه به ثانیه، زیاد خواهد شد و محاسبات برحسب دقیقه منطقی‌تر است

هر بار در این رویکرد با تغییر بارکاری منابع و یا توافقنامه بازه‌ها نیز به‌صورت خودکار تغییرخواهندکرد. در محیط کلودسیم قبل از اینکه تخطی‌های SLA محاسبه شود زمان شبیه‌سازی سیستم برحسب نانوثانیه تنظیم می‌شود و با اندازه‌گیری طول زمان اجرا در هر بازه، زمان شبیه‌سازی محاسبه می‌شود.

در این مقاله، بارکاری در یک شبانه روز یا ۱۴۴۰ دقیقه معادل یک فایل پلنت لب به‌صورت تصادفی تولید شده است. برای هر بار کاری میزان نیاز آن به حافظه، پهنای باند و پردازنده مشخص شده است.

بر اساس بهره‌وری محاسبه شده، تخطی‌های صورت گرفته در زمان اجرای بار کاری، در متغیر StateHistory ذخیره می‌گردد. از روی این متغیر، تخطی‌های SLA با الگوریتم SFLA تنظیم می‌شود و مشخص می‌گردد که بازه زمانی در وضعیت فعلی چه مقدار در نظر گرفته شود. هدف تابع برازش SFLA کاهش سربار زمانی تحمیل‌شده به سیستم و افزایش کشف تخطی‌های انجام شده و به دنبال آن تعداد تخطی کشف نشده کمتر، در بازه زمانی تعیین شده می‌باشد. تعادل بین این دو خواسته توسط الگوریتم SFLA تعیین می‌شود؛ بنابراین SFLA دو متغیر ورودی تخطی‌های SLAو بازه‌های زمانی متناظر با آن را دارد.

از آنجا که می‌بایست تعادلی بین مدت زمان کشف تخطی و تعداد تخطی کشف نشده برقرار گردد؛ با استفاده از الگوریتم جهش قورباغه، بازه زمانی انتخاب می‌شود که نه خیلی کم باشد آن‌قدر که هر تخطی از SLA را در نظر بگیرد و زمان پردازش زیاد شود و نه آن‌قدر زیاد باشد که خیلی از تخطی‌ها را پیدا نکند .شکل (۲) روندنمای الگوریتم پیشنهادی را نشان می‌دهد.




**شکل (۲): روندنمای الگوریتم پیشنهادی**

## ۵- تنظیم پارامترها و آزمایشات و نتایج ارزیابی

به منظور ارزیابی الگوریتم پیشنهادی با استفاده از الگوریتم SFLA جهت بهره‌وری از حافظه، پردازنده و پهنای باند از داده‌های آزمایشگاه پلنت‌لب ۱۴۴۰ داده به صورت تصادفی انتخاب گردید. این داده‌ها به این صورت است که اگر روز را ۲۴ ساعت در نظر بگیریم به ازای هر ساعت یعنی ۶۰ دقیقه در مجموع خواهیم داشت۱۴۴۰ =۲۴×۶۰ یعنی هر روز معادل ۱۴۴۰ دقیقه است. فرض می‌شود که بازه زمانی بر حسب دقیقه باشد. با این فرض از محاسبه برحسب ثانیه چشم پوشی می‌شود چرا که به نظر می‌رسد بارکاری تحمیل شده به سیستم برای بررسی ثانیه به ثانیه تغییرات، زیاد خواهد بود و محاسبه بر حسب دقیقه منطقی‌تر است.

در این مقاله محیط ابر در شبیه‌ساز کلودسیم نسخه ۳.۰.۳ ایجادشده است. این شبیه‌ساز در محیط نرم‌افزار NetBeans IDE 8.1 اجراشده است. در محیط شبیه‌سازی ۵۰ دستگاه فیزیکی طراحی و در هر دستگاه فیزیکی ۵۰ ماشین مجازی میزبانی شده است. با شروع کار، ماشین‌های مجازی بر روی میزبانان فیزیکی تخصیص داده می‌شوند. درنتیجه باعث ایجاد یک محیط مجازی ابر تا ۵۰ گره محاسباتی می‌شود که قادر به تأمین منابع لازم برای برنامه‌های کاربردی است. در جدول (۱) مشخصات سخت‌افزاری دستگاه‌های فیزیکی و ماشین‌های مجازی طراحی‌شده آورده شده است.

**جدول ۱: مشخصات ماشین‌های فیزیکی و مجازی**

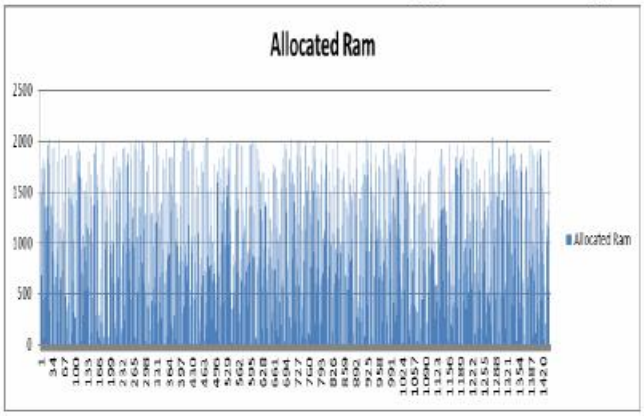
| | سیستم‌عامل | پردازنده | RAM |
|---|---|---|---|
| ماشین فیزیکی | Linux | ۴ هسته‌ای | ۴ گیگابایت |
| ماشین مجازی | Ubunta | ۱ هسته‌ای | ۱ گیگابایت |

برای بررسی میزان تخطی‌های توافقنامه سطح سرویس در الگوریتم SFLA ما فرض کرده‌ایم که درخواست کاربر برای حافظه حداقل ۷۰۰ مگابایت بوده است. همچنان که در شکل (۳) ملاحظه می‌کنید، در دقیقه‌های مختلف از یک روز مقادیر متفاوتی از حافظه توسط میزبان به کاربر اختصاص داده‌شده است. محور افقی زمان برحسب دقیقه و محور عمودی میزان حافظه اختصاص داده شده بر حسب مگابایت می‌باشد.

در شکل (۴) به ازای بازه‌های زمانی، تخطی‌های SLA محاسبه‌شده است. به‌عنوان‌مثال برای بازه زمانی ۴ دقیقه مقدار ۱۶۳ تخطی به‌دست‌آمده است. این مقادیر قبل از این‌که به الگوریتم SFLA داده شود به‌دست‌آمده است. هم چنین بر اساس شکل (۵) متوجه می‌شویم که زمان مصرفی برای کشف کردن این تعداد ۱۶۳ تایی SLA حدوداً برابر ۵۶۹۱۲۲۸ نانوثانیه می‌باشد. هدف SFLA به دست آوردن بازه زمانی است که در آن تخطی SLA بیشتر و بار زمانی اعمالی به سیستم کمتر شود. بعد از اجرای تمام باردستگاه، تعداد SLA کشف شده و بازه زمانی متناظر با هر باردستگاه به الگوریتم جهش قورباغه داده می‌شود. بعد از اجرای الگوریتم مشخص شد که تعادل در کشف تخطی در بازه دودقیقه صورت گرفته است.

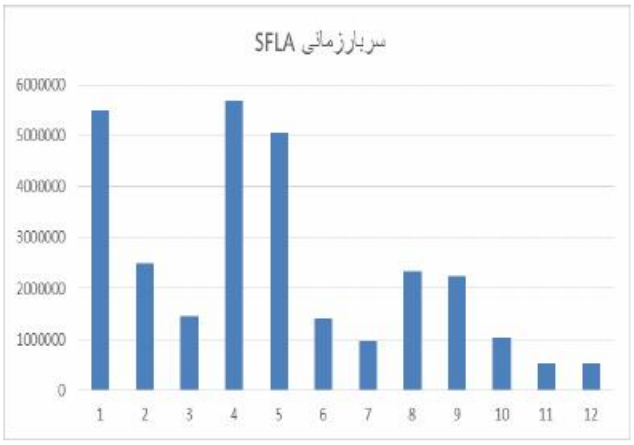
در شکل (۵) سرباز زمانی سیستم نشان داده شده است. در این شکل، محور افقی بازه زمانی و محور عمودی سرباز زمانی سیستم را نشان می‌دهد؛ که در بازه های متفاوت داری مقادیر متفاوتی است. با توجه به شکل (۴) در بازه زمانی دو ۱۷۰ تخطی کشف شده است و در این بازه سرباز زمانی ۲۴۹۴۶۸۰ نانو ثانیه است. هر چه طول بازه کمتر باشد، تعداد تخطی بیشتری پیدا می‌شود؛ اما برای این‌که به دفعات بیشتری به بررسی سیستم پرداخته شده، سرباز زمانی تحمیل‌شده به سیستم بیشتر شده است. هم چنان‌که در شکل (۵) ملاحظه می‌کنید، در بازه زمانی چهارم بیشترین سرباز زمانی را داریم.

جهت بررسی کارایی الگوریتم جهش قورباغه آزمایش دیگری با الگوریتم ژنتیک انجام شده است. برای انجام این آزمایش تمام کارهای مربوط به آزمایش قبلی تکرار شده است. لذا تابع هدف و بار کاری نیز تغییر نکرده است. شکل (۷) مقایسه بین روش مبتنی بر الگوریتم بهینه‌سازی جهش قورباغه و الگوریتم ژنتیک را نشان می‌دهد. نمودار میله‌ای آبی نشان دهنده مقادیر تابع برازش مربوط به الگوریتم ژنتیک و نمودار میله‌ای قرمز مربوط به تابع برازش الگوریتم جهش قورباغه می‌باشد. هدف دو روش الگوریتم جهش قورباغه و الگوریتم ژنتیک کاهش سربار زمانی تحمیل شده به سیستم و در عین حال کشف تعداد تخطی بیشتر می باشد. همانطور که شکل (۷) نشان می‌دهد الگوریتم جهش قورباغه در بازه زمانی دو دارای کمترین مقدار برازش ۱۲/۷۲ و الگوریتم ژنتیک در بازه زمانی ششم دارای کمترین مقدار برازش ۱۴/۸۰ است. در نتیجه الگوریتم جهش قورباغه در بازه زمانی کوتاه‌تری به جواب رسیده است و این الگوریتم توانسته تعادلی بین سربار سیستم و تعداد تخطی‌های کشف شده برقرار کند.

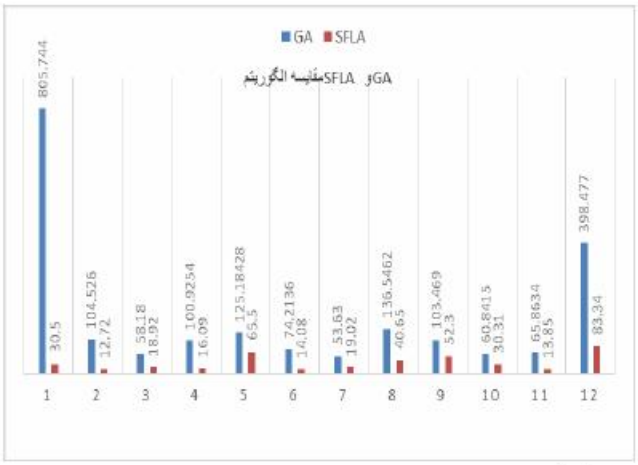



شکل (۳): حافظه اختصاص داده شده به میزبان


شکل (٤): تخطی‌های SLA کشف شده




شکل(٥): سربار زمانی تحمیل شده به سیستم




شکل(٦): مقایسه بین مقدار تابع برازش الگوریتم SFLA و الگوریتم GA

## ٦- نتیجه‌گیری و پیشنهادات آینده

کشف تخطی و سربار زمانی در محاسبات ابری چالشی است که باعث شده است کارهای زیادی در این زمینه انجام شود. کارهای گذشته در بازه‌های زمانی ایستا انجام شده است که باعث افزایش سربار زمانی سیستم و در نتیجه کاهش تعداد تخطی های کشف شده می‌شود. به منظور ایجاد تعادل بین سربار زمانی سیستم و کشف تخطی توافقنامه سطح سرویس رویکردی با استفاده از الگوریتم جهش قورباغه ارائه گردید.

برای ارزیابی روش پیشنهادی، الگوریتم فوق با الگوریتم ژنتیک با شرایط یکسان در یافتن بازه پویا مورد بررسی قرار گرفته است.

نتایج بدست‌آمده نشان می‌دهد که الگوریتم جهش قورباغه نسبت به الگوریتم ژنتیک در بازه زمانی کمتری به جواب رسیده است و توانسته تعادلی بین تخطی‌های کشف شده و سربار سیستم برقرار کند.

## مراجع

[۱] حامدی مسعود، (۱۳۸۹)، **رایانش ابری**، عصر فناوری اطلاعات اولین ماهنامه تخصصی IT در ایران، انتشارات دانشگاه امیرکبیر، شماره ۶۳، ص ۱۰۵-۱۰۲.

[2] Li, J, Qiu, M. Ming, Z, Quan, G, Qin, X, Gu, Z. Online optimization for scheduling preemptable tasks on IaaS cloud systems. Journal of Parallel and Distributed Computing, 666-677, 2012.

[3] Emeakaroha, V. C, Netto, M, A, S, Calheiros, R. N, Brandic, I, Buyya, R, De Rose, C. A. F "Towards autonomic detection of SLA violations in Cloud infrastructures" Future Generation Computer Systems,1017-1029 , 2012.

[4] White, M, Melvin, H, Schukat, M.The Impact of Dynamic Monitoring Interval Adjustment on Power Consumption in Virtualized Data Centers ,2014.

[5] Boniface, M, Phillips, S, Sanchez-Macian, S, Surridge, M. Dynamic service provisioning using GRIA SLAs, International Conference on Service-Oriented Computing, 105–124 , 2007.

[6] Wood, T, Shenoy, P, Venkataramani, A, Yousif, M. Sandpiper: Black-box and gray-box resource management for virtual machines.Computer Networks. 2923-2938, . 2009.

[7] Comuzzi, M, Kotsokalis, C, Spanoudkis, G, Yahyapour,R. Establishing and monitoring SLAs in complex service based systems, Proceedings of the 7th International Conference on Web Services, 45–64, 2009.

[8] Tordsson, J, Montero, R. S, Moreno-Vozmediano, R, Llorente, I.M. Cloud brokering mechanisms for optimized placement of virtual machines across multiple providers. Future Generation Computer Systems. 358-367,2012.

[9] Saravanan, S, Venkatachalam, V, Malligai, S.T. Optimization of SLA violation in cloud computing using artificial bee colony. Int. J. Adv. Eng, 410-414 ,2015.

[۱۰] زنگنه سارا، فراهی احمد، (۱۳۹۳)، رویکرد پیشنهادی یافتن هزینه بهینه کشف خودکار تخطی از توافقنامه سطح خدمات در معماری DeSVi با اختصاص بازه‌های پویا، همایش ملی مهندسی رایانه و مدیریت فناوری اطلاعات، تهران، شرکت علم و صنعت طلوع فرزین.

[11] Ghafuri, N, Saadatjoo, F.Find the Best Time Intervals in the Control of Service Level Agreement Commitments in Cloud Computing Using Colonial Competitive Algorithm. International Science and Investigation journal, 20-35,2015.

[۱۲] سیدی ایمان، مغفوری فرسنگی ملیحه، براتی محمد، نظام آبادی پورحسین، **یک رهیافت جدید برای مسائل چند مدی با استفاده از الگوریتم بهبود یافته جهش قورباغه**، (۱۳۹۰)، دوره ۲، شماره ۱، ص ۴۵-۵۶.

## زیرنویس‌ها

[1] Cloud
[2] Cloud computing
[3] Xen Is a Virtulization Platform
[4] Service Level Agreement
[5] GRIA Is a Service Oriented Architecture framework
[6] Frame Work
[7] Silicon On Insulator
[8] SALMON is the service responsible for monitoring the services QoS
[9] ADA is Agrement Document Analysis
[10] ADA is Agrement Document Analysis
[11] Shuffled frog leaping (SFL)