Journal of Soft Computing and Information Technology (JSCIT)
Babol Noshirvani University of Technology, Babol, Iran
Journal Homepage: jscit.nit.ac.ir
Volume 11, Number 3, Autumn, pp. 92-105
Received: 05/08/2022, Revised: 09/30/2022, Accepted: 11/08/2022

# Hybrid Analytical Models based on Queueing Networks and Generalized Stochastic Petri Nets for Performance Analysis of Load Balancing in Cloud Systems

Ehsan Ataie[1*]
1*- Department of Computer Engineering, University of Mazandaran, Babolsar, Iran.
[1*] ataie@umz.ac.ir

Corresponding author's address: Ehsan Ataie, Department of Computer Engineering, Faculty of Engineering and Technology, University of Mazandaran, Babolar, Iran.

**Abstract-** In this paper, analytical models are presented for modeling cloud load balancing mechanism combing queueing networks (QNs) and generalized stochastic Petri nets (GSPNs). To this end, a hybrid model is proposed to model a cluster inside an Infrastructure-as-a-Service (IaaS) cloud data center in the first step. The model includes several real aspects of such environments, such as request queueing, virtual machine (VM) provisioning, VM servicing, and powering on and off of physical machines (PMs). Based on the presented cluster model, a hybrid monolithic model is proposed in the second step that encompasses requests arrival and load balancing component of the cloud. The proposed monolithic model supports heterogeneity of requests in terms of the arrival process and the number and execution time of requested VMs. To demonstrate the applicability of the hybrid monolithic model, several load balancing algorithms that can be applied to such a model are introduced and evaluated based on different performance metrics of interest.

**Keywords**- Analytical modeling, Queueing network, Petri network, Cloud computing, Load balancing.

مجله علمی پژوهشی رایانش نرم و فناوری اطلاعات
دانشگاه صنعتی نوشیروانی بابل
صفحه مجله: jscit.nit.ac.ir
جلد ۱۱، شماره ۳، پاییز ۱۴۰۱ ، صفحه ۹۲-۱۰۵
دریافت: ۱۴۰۱/۰۲/۱۸ ، بازنگری: ۱۴۰۱/۰۷/۰۸ ، پذیرش: ۱۴۰۱/۰۸/۱۷

# مدل‌های تحلیلی ترکیبی مبتنی بر شبکه‌های صف و شبکه‌های پتری تصادفی تعمیم‌یافته برای تحلیل کارایی موازنه بار در سیستم‌های ابری

احسان عطائی[1*]
*۱- گروه مهندسی کامپیوتر، دانشکده مهندسی و فناوری، دانشگاه مازندران، بابلسر، ایران.
[1*]ataie@umz.ac.ir

* نشانی نویسنده مسئول: احسان عطائی، بابلسر، دانشگاه مازندران، دانشکده مهندسی و فناوری، گروه مهندسی کامپیوتر.

**چکیده- در این مقاله، مدل‌های تحلیلی برای مدل‌سازی مکانیزم موازنه بار در ابر با استفاده از ترکیب شبکه‌های صف و شبکه‌های پتری تصادفی تعمیم‌یافته ارائه شده‌اند. بدین منظور، در گام اول یک مدل ترکیبی برای مدل‌سازی یک کلاستر در مرکز داده ابر زیرساخت-به-عنوان-سرویس پیشنهاد شده است. این مدل، جنبه‌های واقعی متعددی از سیستم‌های این‌چنینی، نظیر صف‌بندی درخواست‌ها، تدارک ماشین‌های مجازی، سرویس‌دهی به ماشین‌های مجازی، و روشن و خاموش کردن ماشین‌های فیزیکی را در بر می‌گیرد. در گام دوم، بر اساس مدل ارائه شده برای کلاستر، یک مدل یکنوای ترکیبی پیشنهاد شده است که ورود درخواست‌ها و مولفه موازنه بار ابر را شامل می‌شود. مدل یکنوای پیشنهادی از ناهمگونی درخواست‌ها از منظر فرایند ورود و نیز تعداد و زمان اجرای ماشین‌های مجازی مورد تقاضا پشتیبانی می‌نماید. برای نشان دادن کاربست‌پذیری مدل یکنوای ترکیبی، تعدادی الگوریتم موازنه بار که می‌توان آن‌ها را روی این مدل اعمال نمود، معرفی و بر اساس چندین معیار کارایی مطلوب، ارزیابی شده‌اند.**

**واژه‌های کلیدی:** مدل‌سازی تحلیلی، شبکه صف، شبکه پتری، رایانش ابری، موازنه بار.

## ۱- مقدمه

در طول حدود دو دهه‌ای که از پیدایش رایانش ابری می‌گذرد، این پارادایم محاسباتی همچنان در حال توسعه و تغییر بوده است. مدل زیرساخت-به-عنوان-سرویس[۱] (IaaS)، یکی از مدل‌های سرویس محبوب رایانش ابری است که در آن، درخواست کاربران برای منابع محاسباتی سطح پایین، عموما در قالب یک یا چند ماشین مجازی[۲] (VM) که بر روی ماشین‌های فیزیکی[۳] (PM) یکسان یا مختلف اجرا می‌شوند، به آنان اعطا می‌شود [۱]. ماشین‌های مجازی مورد درخواست کاربران، از منظر سیستم عامل، حافظه اصلی، فضای ذخیره، تعداد و نوع هسته‌های پردازشی و موارد دیگر می‌توانند متفاوت باشند. همچنین زمان لازم برای اجرای این ماشین‌ها نیز ممکن است نایکسان باشند [۲].

برای آنکه فراهم‌کنندگان سرویس‌های ابری بتوانند به توافقات سطح سرویس[۴] (SLA) منعقد شده با کاربران وفادار بمانند لازم است تا کیفیت سرویس‌های ارائه شده به مشتریان را مانیتور، اندازه‌گیری و مدیریت نمایند [۳]؛ در غیراینصورت، فراهم‌کننده ناچار خواهد بود تا بخشی قابل توجه از درآمد خود را به عنوان جریمه به مشتریانی که توافق سطح سرویس آنان نقض شده است بپردازد و به این ترتیب، شاهد کاهش سود خود خواهد بود [۴] از سوی دیگر، گزارش شده است که میانگین بهره‌وری منابع در

مراکز داده ابری کمتر از ۴۰ درصد است که به منزله بی‌کار بودن بسیاری از منابع در اغلب مواقع است [۵]. بنابراین، مدل‌سازی و ارزیابی دقیق کارایی منابع و سیستم ابر می‌تواند به فراهم‌کنندگان خدمات ابری کمک کند تا تاثیر سیاست‌های مختلف تخصیص و مدیریت منابع را بر عملکرد مراکز داده بسنجند، انتخاب درستی در این خصوص اتخاذ نمایند، و حداکثر سوددهی را محقق سازند.

اندازه‌گیری، مدل‌سازی تحلیلی و شبیه‌سازی، سه روش اصلی مدل‌سازی و ارزیابی کارایی سیستم‌های پیچیده کامپیوتری نظیر ابرهای زیرساخت-به-عنوان-سرویس هستند. روش اندازه‌گیری، مبتنی بر راه‌اندازیِ بستر واقعی یا آزمایشی، و مستلزم اجرای آزمون‌های مختلف با تغییر بارهای کاری و پیکربندی سیستم است و از این جهت، برای سیستمی با پیچیدگی ابر پرهزینه می‌نماید. استفاده از روش‌های تحلیلی با توجه به مدل‌سازی ریاضی و دقت بالای نتایج، نه تنها به فراهم‌کنندگان ابر امکان ارزیابی تحت شرایط مختلف را می‌دهد، بلکه از منظر محدودیت‌های هزینه‌ای و زمانی نیز ارزشمند و درخور توجه است. از سوی دیگر، سربار هزینه‌ای کم در روش شبیه‌سازی، از این جهت که امکان تکرار اجراهای شبیه‌سازی را در دفعات زیاد و با پیکربندی‌های مختلف سیستم ابری فراهم می‌کند، حائز اهمیت و نویدبخش است؛ بعلاوه، روش‌های مبتنی بر شبیه‌سازی، امکان تحلیل حساسیت، تحلیل‌های what-if و بررسی شرایط گلوگاه را نیز ممکن می‌سازند [۶, ۷].

در این مقاله یک معماری لایه‌ای از سیستم ابر زیرساخت-به-عنوان-سرویس با استفاده از روش‌های تحلیلی مدل‌سازی شده است. به این ترتیب که ابتدا یک مدل تحلیلی بر مبنای ترکیبی از روش‌های شبکه‌های صف و شبکه‌های پتری تصادفیِ تعمیم‌یافته برای یک کلاستر از مرکز داده ابری پیشنهاد شده است. سپس، مدل دیگری با همان فرمالیسم و بر مبنای مدل اول برای کل مرکز داده ابری ارائه گردیده است. در مدل‌های ارائه شده، جزئیات قابل ملاحظه‌ای از سیستم‌های ابری واقعی نظیر تفکیک مراحل تدارک و اجرای ماشین‌های مجازی، آگاهی از توان مصرفی با مدل‌سازی روشن و خاموش کردن ماشین‌های فیزیکی، و نیز ناهمگونی درخواست‌های کاربران از منظر فرایندهای متفاوت ورود و تعداد متفاوت ماشین‌های مجازیِ مورد تقاضا به ازای هر درخواست لحاظ گردیده است. با هدف نشان دادن کاربست‌پذیری مدل‌های پیشنهادی، چند الگوریتم مختلف موازنه بار ورودی میان کلاسترهای سیستم ابر پیشنهاد گردیده است؛ آنگاه، عملکرد الگوریتم‌های معرفی شده، با تعریف معیارهای مختلف کارایی بر مبنای مدل‌های ارائه شده - نظیر میانگین طول، توان عملیاتی و زمان انتظار درخواست‌ها در صف‌های سراسری و محلی، و نیز تعداد ماشین‌های مجازی آزاد و مشغول- مورد ارزیابی و مقایسه قرار گرفته است.

بصورت خلاصه، نوآوری‌های اصلی این مقاله شامل موارد زیر می‌باشد:

- ما یک مدل تحلیلی جدید برای کلاستری از یک ابر زیرساخت-به-عنوان-سرویس ارائه می‌کنیم که جزئیات بسیاری از این سیستم‌ها نظیر تفکیک مراحل تدارک و اجرای ماشین‌های مجازی، روشن و خاموش کردن ماشین‌های فیزیکی، و نیز ناهمگونی درخواست‌های کاربران را در بر می‌گیرد.
- مدل پیشنهادی، بر مبنای ترکیبی از دو روش تحلیلی شبکه‌های صف و شبکه‌های پتری تصادفی تعمیم‌یافته ارائه می‌شود.
- بر اساس یک رویکرد سلسله‌مراتبی و بر مبنای مدل اول، ما یک مدل تحلیلی دیگر برای کل مرکز داده ابری پیشنهاد می‌کنیم که سطح تجرید مناسبی از سیستم ارائه می‌نماید و امکان گنجاندن راهبردهای مدیریت منابع ابر نظیر الگوریتم‌های موازنه بار میان کلاسترها در سطح بالا را فراهم می‌نماید.
- با استفاده از مدل‌های ارائه شده و برای نشان دادن نحوه کاربست‌پذیری این مدل‌ها، الگوریتم‌های موازنه بار مختلفی پیشنهاد می‌شوند و بر اساس معیارهای مختلف کارایی با یکدیگر مورد مقایسه قرار می‌گیرند.

ادامه مقاله بصورت زیر سازمان یافته است: پژوهش‌های مرتبط در بخش ۲ مورد بررسی و مرور قرار می‌گیرند. در بخش ۳، مفاهیم پایه و پس‌زمینه لازم برای مطالعه این پژوهش معرفی و مرور می‌شوند. بخش ۴، توصیف معماری سیستم مورد بررسی را شامل می‌شود. به مدل‌های ارائه شده، الگوریتم‌های موازنه بار، و معیارهای کارایی در بخش ۵ پرداخته می‌شود. نتایج ارزیابی کارایی در بخش ۶ ارائه خواهند شد. نهایتا، بخش ۷ شامل نتیجه‌گیری و پیشنهاداتی برای ادامه این پژوهش خواهد بود.

## ۲- مروری بر پژوهش‌های پیشین

پژوهش‌های مختلفی وجود دارند که به مدل‌سازی تحلیلی کارایی در رایانش ابری پرداخته‌اند. نظر به تعدد این تحقیقات و محدودیت فضا، در این بخش به تعدادی از موارد به‌روز که نزدیکی بیشتری با مطالعه جاری دارند، به اختصار اشاره خواهد شد.

در [۶] به مدل‌سازی تحلیلی سلسله‌مراتبی برای مدل نمودن ابر

خصوصی مبتنی بر اپن‌استک پرداخته شده است. برای این منظور، از نمودارهای بلوکی قابلیت اعتماد[۵] برای مدل‌سازی جنبه‌هایی از سیستم و از زنجیره‌های مارکوف پیوسته‌زمان[۶] برای مدل‌سازی جنبه‌های دیگری از سیستم استفاده شده است. آنگاه این سیستم ابر در قالب نُه سناریوی مختلفِ بدون افزونگی و با افزونگی مورد مطالعه قرار گرفته و سناریوهای مختلف بر مبنای زمان دردسترس نبودن سیستم ابر و هزینه اجرای آن‌ها با یکدیگر مقایسه شده‌اند. شایان ذکر است که مدل‌های ارائه شده در پژوهش مذکور، صرفا مبتنی بر ابر اپن‌استک بوده و الزاما قابل اعمال بر سایر بسترهای راه‌اندازی ابرهای IaaS نمی‌باشند. همچنین در پژوهش مذکور از ترکیبی از روش‌های مبتنی و غیرمبتنی بر فضای حالت استفاده شده است؛ در حالیکه در این مقاله، دو فرمالیسم متفاوت، اما مبتنی بر فضای حالت جهت مدل‌سازی مورد استفاده قرار گرفته است. بعلاوه، بر خلاف مقاله جاری، بحث روش و خاموش کردن ماشین‌های مجازی در پژوهش مذکور مطرح نمی‌باشد.

در [۸] از ترکیبی از روش‌های تحلیلی مبتنی بر فضای حالت و غیرمبتنی بر فضای حالت برای تحلیلِ در دسترس بودن کانتینرها در یک سیستم ابری استفاده شده است. برای حل مدل‌های ارائه شده نیز از روش‌های تحلیلی و شبیه‌سازی کمک گرفته شده است. بعلاوه، یک بسته نرم‌افزاری متن‌باز در این مطالعه به منظور شبیه‌سازی و ارزیابی سیستم‌های مبتنی بر کانتیر توسعه داده شده است. در اینجا نیز بر خلاف روش استفاده شده در این مقاله، از یک تکنیک غیرمبتنی بر فضای حالت استفاده شده است؛ ضمن اینکه تمرکز بر کانتینرسازی و هدف، ارزیابی دسترس‌پذیری بوده است؛ در حالیکه در مقاله ما، تمرکز بر مجازی‌سازی و هدف، ارزیابی کارایی می‌باشد.

در [۹]، مدل‌های کارایی تحلیلی برای مدل‌سازی بسترهای محاسباتی بدون سرور[۷] ارائه شده است که پیچیدگی‌های مرتبط با خودمقیاس‌پذیری مبتنی بر متریک را در بر می‌گیرد. برای مدل‌سازی، از زنجیره‌های مارکوف استفاده شده است. محققین، تبعات پیکربندی‌های مختلف سیستم و بارهای کاری متفاوت را بر کارایی سیستم مورد مطالعه، بررسی نمودند. مدل‌های ارائه شده، با انجام آزمایشات گسترده در بستر یک محیط اجرای واقعی مورد اعتبارسنجی قرار گرفتند. بر خلاف تحقیق مذکور، روش آزمایش در این مقاله، اجرای شبیه‌سازی مدل‌های تحلیلی ارائه شده می‌باشد. همچنین، جزئیات مدل شده در این مقاله از یک سیستم ابری، با جزئیات مدل‌های پیشنهادی در پژوهش مذکور متفاوت است.

در [۱۰] مدلی برای بهبود خاصیت کشسانی برای تأمین منابع در شبکه‌های ابری ارائه شده است. مدیریت کشسانی در مدل ارائه شده، با استفاده از شبکه پتری رنگی و در قالب کنترل شبکه‌های صف انجام یافته است. به گونه‌ای که به ازای ورود هر درخواست یا ارائه سرویس در صف، حرکت افقی و به ازای نیاز به افزایش یا کاهش ماشین مجازی، حرکت عمودی در صف تعریف شده است. در مقام مقایسه، رویکرد پیشنهادی ما مبتنی بر شبکه‌های پتریِ تصادفی تعمیم یافته و ترکیب آن با ویژگی‌های نظریه صف می‌باشد.

در [۱۱] یک روش ترکیبی مبتنی بر مدل‌سازی تحلیلی و یادگیری ماشینی برای پیشبینی زمان اجرای برنامه‌های کلان‌داده‌ها (شامل MapReduce، Tez و Spark) در کلاسترهای سیستم ابری پیشنهاد شده است. برای مدل‌سازی تحلیلی، از نظریه صف استفاده شده است و مدل ارائه شده، به شیوه شبیه‌سازی اجرا شده است. نتایج ارزیابی در بستر واقعی ابر نشان از آن دارد که این روشِ ترکیبی نسبت به روش مبتنی بر یادگیری ماشینی صِرف و روش‌های ترکیبی دیگر، دقت بیشتر و هزینه کمتری دارد. اگرچه در رویکرد پیشنهادی در این مقاله از یادگیری ماشینی استفاده نشده است، اما استفاده از نظریه صف و شبیه‌سازی مدل ارائه شده در آن، مشابه کار ما می‌باشد.

در [۱۲] یک روش تصادفی برای ارزیابی دسترس‌پذیری در سیستم ابر زیرساخت-به-عنوان-سرویس ارائه شده است که در آن، خرابی ماشین‌های فیزیکی با مهاجرت ماشین‌ها بین سه مخزن با نرخ تعمیر متفاوت کاهش می‌یابد. آنگاه، نتایج حاصل از تحلیل عددی مدل‌های یکنوا و تقریبی بر حسب اندازه فضای حالت زنجیره مارکوف زیرین، تعداد ماشین‌های هر دسته، و زمان حل مساله با یکدیگر مقایسه شده‌اند. بر خلاف روش پیشنهادی در پژوهش مذکور، در این مقاله بحث خرابی ماشین‌های فیزیکی و دغدغه دسترس‌پذیری یا اتکاپذیری سیستم مطرح نیست. همچنین با وجود اتکا به تکنیک‌های تحلیلی، استفاده از ابزار مدل‌سازی مناسب در این مقاله، ما را از درگیر شدن در جزئیات زنجیره‌های مارکوف زیرین و حل آن‌ها بی‌نیاز ساخته است.

در [۱۳] از شبکه‌های پاداش تصادفی[۸] برای مدل‌سازی یکپارچه سیستم ابر استفاده شده است. برای غلبه بر محدودیت‌های مدل پیشنهادی از منظر مقیاس‌پذیری، از دو تکنیک تقریب‌زنی به نام‌های روش تا کردن[۹] و روش نقطه ثابت[۱۰] برای تعیین درصد ماشین‌های فیزیکی در دسترس، زمان پاسخ، و توان مصرفی ماشین‌های فیزیکی استفاده شده است. نتایج نشان می‌دهد که

مدل‌های تقریبی بدون فدا کردن دقت ارزیابی، فضای حالت بسیار کوچک‌تری را دارا هستند و دیرتر دچار مشکل انفجار فضای حالت می‌شوند. رویکرد لایه‌ای مدل‌سازی در پژوهش مذکور، مشابه رویکرد ارائه شده در مقاله ما می‌باشد. با این وجود، در حالیکه مدل‌های ارائه شده در پژوهش مذکور، مبتنی بر حل مدل تحلیلی و در نتیجه، از منظر مقیاس‌پذیری دچار مشکل هستند، روش و ابزار شبیه‌سازی مدل‌های ارائه شده در این مقاله، مساله عدم مقیاس‌پذیری را مرتفع می‌سازد.

در [۱۴] راهبردی برای مکان‌یابی ماشین‌های مجازی میزبان برنامه‌های کاربردی چندلایه‌ایِ مجازی‌شده پیشنهاد شده است. به این ترتیب که در ابتدا، رتبه‌بندی این ماشین‌های مجازی بر اساس تابع کاب-داگلاس انجام شده است. آنگاه، برنامه‌های کاربردی مورد نظر بر اساس معیارهای کارایی اولویت‌بندی می‌شوند. در نهایت، برنامه‌های کاربردی چندلایه‌ایِ مجازی‌شده با توجه به پارامترهای مختلفی نظیر نیازمندی منابع هر لایه و وضعیت کارایی ماشین‌های میزبان، مکان‌یابی می‌گردند. نتایج نشان می‌دهد که روش پیشنهادی، در موازنه بار، کاهش مصرف انرژی، و کاهش نقض سرویس موثر بوده است. به وضوح، معیارهای کارایی و روش ارزیابی این پژوهش با مقاله جاری متفاوت است. با این وجود، بحث مکان‌یابی ماشین‌های مجازی مورد اشاره در پژوهش مذکور، مشابهت‌هایی با مساله موازنه ماشین‌های مجازی مورد نیاز کاربران در کلاسترهای مرکز ابری دارد.

در [۱۵] زمانبندی موازنه بار و مدل‌سازی قابلیت اعتماد برای یک سیستم پردازش تراکنش در محیط محاسباتی حسب تقاضا انجام شده است. برای این منظور از شبکه‌های پتری رنگی استفاده شده است که قابلیت مدل‌سازی سلسله‌مراتبی آن، به‌کارگیری سطوح مختلف تجرید و استفاده از زیرمدل‌ها را میسر می‌سازد. آنگاه، مدل پیشنهادی برای بیان همروندی نیز گسترش یافته که منجر به بهبود نتایج قابلیت اطمینان شده است. در مقایسه با این مقاله، رویکرد پژوهش مذکور معطوف به کاربرد خاصی – یعنی سیستم پردازش تراکنش- است و بر خلاف این مقاله، تمرکز بر قابلیت اطمینان بوده است.

در [۱۶] یک رویکرد زمانبندی آگاه از سود و آگاه از تداخل ماشین‌های مجازی ارائه شده است که هدف آن، ادغام کردن بهینه این ماشین‌ها بر روی تعداد کمتری ماشین فیزیکی در محیط زیرساخت-به-عنوان-سرویس است. هدف، بررسی سود و هزینه ادغام از منظرهای مختلفی نظیر توان مصرفی، تداخل عملیاتی ماشین‌های مجازی، بهره‌وری منابع، توافق سطح سرویس، و تعداد صفحات حافظه منتقل شده حین فرایند مهاجرت زنده ماشین‌ها می‌باشد. بدین منظور، یک مدل برنامه‌نویسی پویای تصادفی و یک مساله بهینه‌سازی طراحی و ارزیابی شده است. بر خلاف رویکرد ارائه شده در پژوهش مذکور، بحث تداخل ماشین‌های مجازی در کار ما مطرح نیست. همچنین در مقاله ما بر خلاف پژوهش مذکور، مدیریت ابر بر اساس ساختار دو سطحی مدل شده است. اما از آنجا که رویکرد پیشنهادی در کار ما امکان اعمال سیاست‌های مختلف مدیریت منابع و موازنه بار کاری را فراهم می‌آورد، می‌توان از مدل‌های ارائه شده در این مقاله نیز همچون پژوهش مذکور، برای بهبود توافق سطح سرویس یا ادغام ماشین‌های مجازی با هدف کاهش توان مصرفی بهره برد.

به طور کلی و در مقایسه با روش‌های موجود، روش مدل‌سازی ارائه شده در این مقاله، دست طراحان مراکز داده را در مدل کردن و ارزیابی یک ساختار لایه‌ای از سیستم ابر IaaS که شامل تعدادی کلاستر در لایه پایین‌تر است، باز می‌گذارد و راهکاری برای غلبه بر پیچیدگی‌های مدل‌سازی جلوی پای طراح قرار می‌دهد.. علاوه بر آن، رویکرد پیشنهادی با وجود سادگی، جنبه‌های متعددی از سیستم‌های ابری را که غالبا در پژوهش‌های پیشین دیده نشده است، در بر می‌گیرد. همچنین بر خلاف سایر مطالعات انجام شده، مدل ترکیبی ارائه شده در این مقاله، مزایای دو فرمالیسم مهم مدل‌سازی تحلیلی یعنی شبکه‌های پتری و شبکه‌های صف را با یکدیگر ترکیب نموده و دست طراح مرکز ابری را برای گنجاندن قابلیت‌ها و جزئیات بیشتری در مدل و از جمله برای مدل کردن راهبردهای مدیریت منابع در سیستم ابر می‌گشاید.

## ۳- مفاهیم پایه

از آنجا که در این مقاله از فرمالیسم ترکیبیِ مبتنی بر شبکه‌های صف و شبکه‌های پتری تصادفی تعمیم‌یافته استفاده شده و نرم‌افزار مدل‌سازی JMT برای مدل‌سازی و اجرای مدل پیشنهادی مورد استفاده قرار گرفته است، در این بخش به مفاهیم و مبانی اولیه مرتبط با این دو شبکه و نیز نرم‌افزار JMT بصورت خلاصه پرداخته شده است. طبیعتا کسب اطلاعات بیشتر در مورد این مفاهیم، مستلزم رجوع به منابع دیگر است.

### ۳-۱- شبکه‌های صف

نظریه صف، یک تکنیک مهم برای مدل‌سازی تحلیلی تصادفی سیستم‌های کامپیوتری توزیع شده محسوب می‌شود که در آن‌ها، کارهای متعدد و منابع محدود وجود دارند، و میزان تقاضای

سرویس کارها و زمان‌های بین ورود کارها تصادفی فرض می‌شوند [۱۷]. در واقع، نظریه صف یک روش ریاضی است که به مطالعه رفتار صف‌ها در شبکه‌ها و سیستم‌های مبتنی بر صف می‌پردازد. با مدل‌سازی سیستم‌های توزیع شده بصورت شبکه‌ای از صف‌های مرتبط با یکدیگر، ارزیابی کارایی سیستم‌های موجود و پیشبینی عملکرد سیستم‌های در دست طراحی میسر می‌گردد.

### ۳-۲- شبکه‌های پتری تصادفی تعمیم‌یافته

شبکه‌های پتری، یکی از زبان‌های مدل‌سازی ریاضی و یک سیستم گذار حالت برای توصیف و ارزیابی همروندی و همگامی انواع مختلف سیستم‌ها و از جمله، سیستم‌های کامپیوتری توزیع شده هستند [۱۸]. در این شبکه‌ها، توکن‌ها با شلیک گذارها از یک مکان به مکان دیگر جابه‌جا می‌شوند. ارتباط بین گذارها و مکان‌ها با استفاده از مفهوم کمان مدل می‌شود. به لحاظ ریاضی، این شبکه‌ها با یک گراف دوبخشیِ جهت‌دار نمایش داده می‌شوند. برای درک بهتر، از نمادگذاری گرافیکی هم برای نشان دادن این شبکه‌ها استفاده می‌شود. یک شبکه پتری تصادفی تعمیم‌یافته، نوع خاصی از شبکه‌های پتری است که از یک سو در آن گذارهای زمان‌دار پس از یک مدت زمان تصادفی که با یک متغیر تصادفی قابل بیان است، شلیک می‌نمایند؛ از سوی دیگر، در چنین شبکه‌ای گذارها می‌توانند از دو نوع زمان‌دار یا آنی باشند. به این ترتیب که بر خلاف گذارهای زمان‌دار که مدت زمانی طول می‌کشد تا شلیک نمایند، گذارهای آنی، بلافاصله پس از مساعد بودن شرایط و فعال شدن، شلیک می‌کنند [۱۹].

### ۳-۳- پکیج نرم‌افزاری JMT

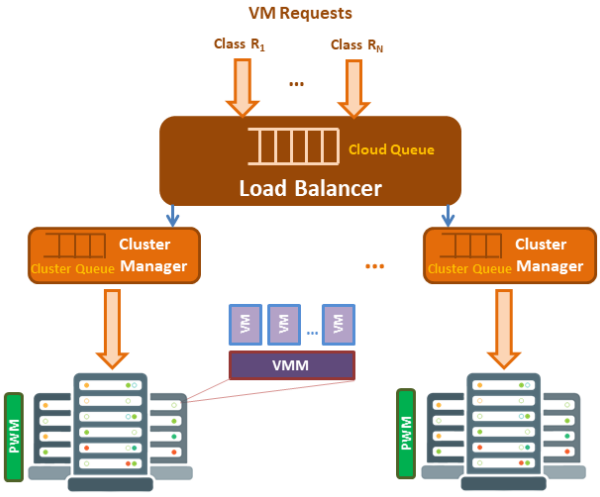
پکیج نرم‌افزاری JMT مجموعه‌ای از ابزارهای نرم‌افزاری مناسب برای ارزیابی کارایی، برنامه‌ریزی ظرفیت، و تبیین بار کاریِ سیستم‌های ارتباطی، توزیع شده و مبتنی بر شبکه است که توسط دانشگاه پلی‌تکنیک میلان و کالج سلطنتی لندن مدیریت و راهبری می‌شود [۲۰]. این مجموعه ابزار، امکان مدل‌سازی بر مبنای زنجیره‌های مارکوف، شبکه‌های صف، و شبکه‌های پتری را ارائه کرده است. همچنین امکان ارزیابی بر مبنای حل مدل‌های تحلیلی و یا انجام شبیه‌سازی (بسته به کاربرد و مدل‌های استفاده شده) در این ابزار وجود دارد. بعلاوه، فرمالیسم‌های مدل‌سازی آن، تعریف رنگ‌ها برای توکن‌های شبکه‌های پتری و نیز تعریف رده‌ها برای وظایف یا کارهای شبکه‌های صف را میسر نموده است.

## ۴- توصیف سیستم

برای آنکه بتوان یک مدل واقع‌گرایانه‌تر از سیستم‌های ابری را مورد بررسی قرار داد، یک معماری لایه‌ای را بعنوان معماری پایه در نظر می‌گیریم که در آن دو لایه مدیریتی توسط فراهم‌کننده سرویس ابری بصورت سلسله‌مراتبی مورد استفاده قرار گرفته‌اند. ساختارهای مشابهی نیز در دیگر منابع استفاده شده‌اند [۲۳-۲۱] . شکل ۱ این معماری پایه را نشان می‌دهد. درخواست‌های رسیده به مرکز داده، در صف اصلی درخواست‌های ورودی، موسوم به صف ابر، CLQ، با الگوریتم زمابندی اول-ورود-اول-خروج (FIFO) قرار می‌گیرند. سپس، متوازن‌کننده یا توزیع‌کننده بار، LB، بر اساس سیاست مشخص موازنه بار، هر یک از این درخواست‌ها را برای دریافت سرویس به یکی از کلاسترهای مرکز داده گسیل می‌نماید. این درخواست‌ها توسط مدیر کلاستر، CRM، دریافت و مورد رسیدگی قرار می‌گیرد. درخواست‌های دریافت شده توسط مدیر کلاستر، در صفی محلی به نام صف کلاستر، CRQ، جا داده می‌شوند. بنا بر سیاست موازنه بار، درخواست ممکن است به کلاستری با طول صف کلاستر کوتاه‌تر یا بلندتر، کلاستری با ماشین‌های مجازی آزاد بیشتر یا کمتر، کلاستری با تعداد بیشتر یا کمتر ماشین‌های فیزیکی خاموش و نظایر آن ارسال گردد. به درخواست‌های رسیده به مدیر کلاستر باید بنا به نیاز، ماشین(های) مجازی تخصیص داده شود. مدیر کلاستر تعیین می‌کند که روی کدام ماشین فیزیکی، ماشین یا ماشین‌ها مجازی مورد نیازِ درخواست باید آماده شده و تخصیص یابند. هایپروایزر ماشین فیزیکی میزبان، مسئول تخصیص منابع لازم برای اجرای ماشین‌های مجازی مورد درخواست می‌باشد. به جهت مطابقت بیشتر با دنیای واقع، در اینجا فرض می‌کنیم که هر درخواست IaaS می‌تواند از رده‌های مختلف و شامل تقاضای تعداد متفاوتی ماشین مجازی باشد. همچنین زمان سرویس یا اجرای مورد نیاز ماشین‌های مجازی مختلف نیز متفاوت فرض می‌شود.

زمانی که یک درخواست به CRM می‌رسد، اگر در مخزن منابع مجازی آن کلاستر، ماشین‌های مجازی آزاد برای تخصیص وجود داشته باشد (به عبارت دیگر حداقل یک ماشین فیزیکی با منابع آزاد کافی موجود باشد) درخواست بصورت آنی پاسخ داده می‌شود؛ در غیر اینصورت، باید در ابتدا به تعداد کافی ماشین فیزیکی خاموش، روشن و آماده به کار شوند تا بتوان ماشین‌های مجازی مورد نیاز را آماده کرد و تخصیص داد. از سوی دیگر، ماشین‌های فیزیکی که میزبان هیچ ماشین مجازی آماده یا در حال اجرایی نیستند را با هدف کاهش توان مصرفی می‌توان به مد استندبای یا خاموش برد. این کار توسط مولفه مدیریت توان، PWM، در آن

کلاستر انجام می‌شود.




**شکل ۱: معماری پایه سیستم ابر مورد مطالعه**

## ۵- مدل پیشنهادی

در این بخش، در ابتدا مدلی برای یک کلاستر از مرکز ابری در بخش ۱-۵ ارائه می‌گردد. سپس، بر مبنای مدل ارائه شده برای کلاستر ابری، مدل یکنوایی در بخش ۲-۵ برای کل مرکز داده ابری پیشنهاد خواهد شد. الگوریتم‌ها و سیاست‌های موازنه بار میان کلاسترهای مرکز داده در بخش ۳-۵ معرفی خواهند گردید. سپس معیارهای مطلوب کارایی در بخش ۴-۵ ارائه خواهند شد.

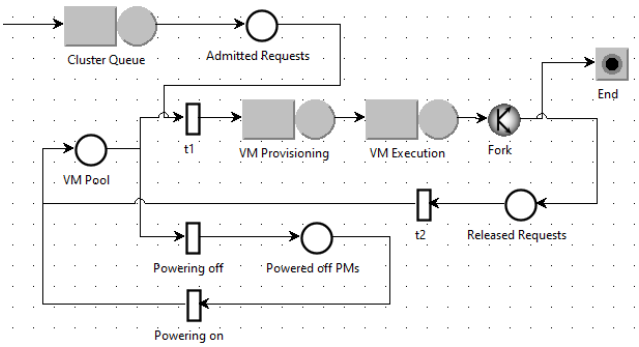
### ۵-۱- مدل پیشنهادی در سطح کلاستر

مدل ارائه شده برای یک کلاستر ابری، در شکل ۲ نشان داده شده است. درخواست‌های ارسال شده به کلاستر ابری، ابتدا در گره Cluster Queue که یک گره صف است، قرار می‌گیرند تا مورد رسیدگی قرار گیرند. ظرفیت این صف محدود و الگوریتم رسیدگی به درخواست‌ها در آن از نوع FIFO تعریف شده است. بنابراین چنانچه تعداد درخواست‌های در حال سرویس و در حال انتظار در این صف، از ظرفیت صف فراتر رود، درخواست‌های بعدی بلاک خواهند شد. زمانی که یک درخواست در این صف مورد رسیدگی قرار گیرد و از این سیستم خارج گردد، یک توکن در مکان Admitted Requests قرار می‌گیرد. تعداد توکن‌های موجود در این مکان، تعداد درخواست‌هایی را نشان می‌دهد که در کلاستر پذیرفته شده و در صورت وجود ماشین‌های مجازی آزاد به تعداد مورد نیاز، سرویس خود را دریافت خواهند نمود. مکان VM Pool متناظر با مخزن منابع مجازی آزاد در کلاستر است و تعداد توکن‌های موجود در آن، تعداد ماشین‌های مجازی آزاد در کلاستر را مشخص می‌نماید. گذار t1، یک گذار آنی است که در شبکه‌های پتری قابل تعریف است. این گذار به این صورت تعریف می‌شود که درصورتیکه حداقل یک توکن در هر دو مکان ورودی به این گذار، یعنی مکان‌های Admitted Requests و VM Pool باشد، فعال و شلیک می‌شود. با شلیک این گذار، یک توکن از هر یک از دو مکان فوق کسر، و یک توکن درخواست در صف VM Provisioning قرار داده می‌شود. البته بسته به نوع درخواست کاربر، می‌توان تعداد ماشین مجازی لازم برای اجابت آن درخواست را در گذار t1 بصورت متفاوتی کد نمود.

گره VM Provisioning یک گره صف است که فرایند آماده‌سازی ماشین یا ماشین‌های مجازی برای درخواست کاربر را مدل می‌نماید. پس از تدارک ماشین‌های مجازی لازم، نوبت به خدمت‌دهی یا اجرای ماشین‌های مجازی می‌رسد؛ در واقع با اجرای ماشین(های) مجازی، کاربر قادر به اجرای برنامه‌های مطلوب خود بر روی این ماشین(های) مجازی خواهد بود. این فرایند توسط گره صف VM Execution مدل‌سازی شده است. آهنگ اجرا یا میانگین مدت اجرای ماشین مجازی توسط کاربر را نیز می‌توان بر اساس نوع درخواست کاربر، بصورت متفاوتی تعریف نمود. پس از آنکه کاربر به میزان مورد نیاز ماشین یا ماشین‌های مجازی خود را اجرا نمود، لازم است که درخواست وی خاتمه یابد. عملکرد گره Fork بدین صورت است که درخواست کاربر را که اجرای آن به پایان رسیده است، تکثیر می‌نماید. یک نسخه از درخواست وارد گره چاهک با نام End می‌شود که خاتمه درخواست و خروج آن از سیستم را مدل می‌کند؛ نسخه دوم به عنوان توکنی وارد مکان Released Requests می‌گردد تا در ادامه، آزادسازی ماشین‌های مجازی اختصاص یافته به آن درخواست را منعکس نماید. کار گذار آنی t2 این است که متناسب با نوع درخواست‌های خاتمه یافته، آزادسازی یک یا تعداد بیشتری ماشین مجازی (به تعداد ماشین‌های مجازی تخصیص یافته به آن درخواست) و بازگشت آن‌ها به مخزن منابع مجازی کلاستر را مدل نماید. در واقع با شلیک این گذار، یک توکن از مکان Released Requests کسر و به تعداد ماشین‌های مجازی تخصیص یافته به آن درخواست، توکن در مکان VM Pool قرار می‌گیرد.

عملکرد مولفه مدیریت توان مصرفی کلاستر، با دو گذار زمان‌دار Powering off و Powering on نشان داده شده است. مکان Powered off PMs متناظر با ماشین‌های فیزیکی خاموش است و تعداد توکن‌های آن، تعداد ماشین‌های فیزیکی خاموش یا در حالت استندبای کلاستر را نشان می‌دهد. گذار Powering on رویداد روشن شدن یک ماشین فیزیکی را در کلاستر مدل می‌نماید. در یک سیستم واقعی، با روشن شدن یک ماشین فیزیکی، باید به

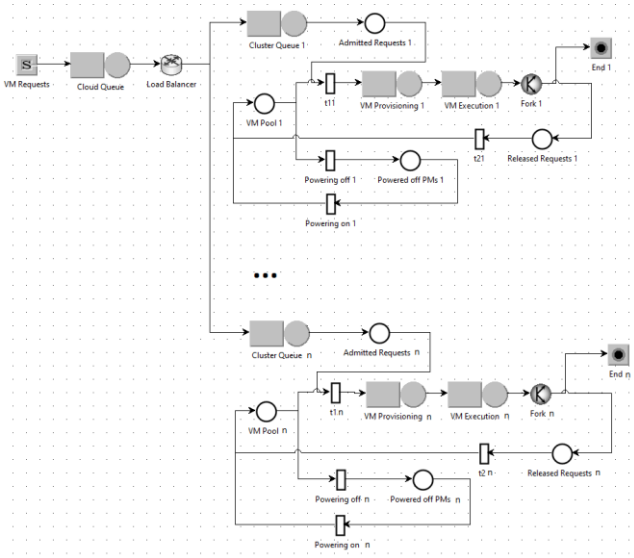
تعداد L ماشین مجازی به مخزن منابع مجازی کلاستر اضافه گردد که L ضریب تسهیم[11] می‌باشد. مابه‌ازای آن در مدل تحلیلی، زمانی که حداقل یک توکن در مکان Powered Off PMs وجود داشته باشد، گذار Powering on فعال می‌شود و می‌تواند شلیک کند؛ با شلیک این گذار، یک توکن از مکان Powered Off PMs کسر و L توکن به مکان VM Pool اضافه می‌گردد. به ترتیب عکس، گذار Powering off رویداد خاموش کردن یک ماشین فیزیکی بی‌کار را در کلاستر مدل می‌نماید. بنابراین، زمانی که حداقل L توکن در مکان VM Pool وجود داشته باشند، گذار Powering off فعال می‌شود و می‌تواند شلیک کند؛ با شلیک این گذار، L توکن از مکان VM Pool کسر و یک توکن به مکان Powered Off PMs اضافه می‌گردد. با سازوکار ارائه شده در مدل پیشنهادی برای کلاستر ابری، امکان تعریف رده‌های متفاوتی از ماشین‌های فیزیکی که روشن شدن آن‌ها منجر به ایجاد تعداد متفاوتی از ماشین‌های مجازی آزاد باشد، وجود دارد.




**شکل ۲: مدل تحلیلی ارائه شده برای یک کلاستر ابری**

## ۵-۲- مدل پیشنهادی در سطح مرکز داده ابری

با استفاده از مدل ارائه شده برای یک کلاستر ابری، مدل تحلیلی شکل ۳، برای یک مرکز داده ابری که چندین کلاستر همگن را شامل می‌شود، ارائه شده است. در این مدل، گره VM Requests یک گره منبع است که ارسال درخواست‌های IaaS کاربران را مدل می‌نماید. این درخواست‌ها را می‌توان از یک یا بیشتر رده درخواست مختلف تشکیل داد؛ به گونه‌ای که هر رده، آهنگ ورود متفاوتی داشته باشند. درخواست‌های تولید شده توسط گره منبع VM Requests وارد سیستم صف Cloud Queue که معادل با صف ابر یا CLQ است می‌شوند. این صف با ظرفیت مشخص تعریف شده است؛ بنابراین چنانچه مجموع تعداد درخواستهای در حال رسیدگی و در حال انتظار در آن، از ظرفیت صف فراتر رود، درخواست‌های جدید رسیده به این سیستم بلاک خواهند شد. گره بعدی در این مسیر، گره Load Balancer است که از نوع مسیریاب تعریف شده است. در گره از نوع مسیریاب می‌توان الگوریتم توازن بار در مرکز داده ابری را پیاده‌سازی نمود. تعریف تعدادی از الگوریتم‌های موازنه بار در بخش ۳-۵ خواهد آمد. بسته به الگوریتم پیاده‌سازی شده در این گره و شرایط حین اجرا، درخواست کاربر به یکی از کلاسترهای موجود در مرکز داده گسیل می‌گردد. مدل تحلیلی یک کلاستر ابری در زیربخش قبلی توضیح داده شده است.




**شکل ۳: مدل تحلیلی یکنوای ارائه شده برای مرکز داده ابری**

## ۵-۳- الگوریتم‌های موازنه بار

مولفه توزیع‌کننده بار مرکز داده ابری، از الگوریتم‌ها و سیاست‌های مختلفی برای موازنه بار میان کلاسترهای موجود در مرکز داده می‌تواند استفاده نماید. در ادامه تعدادی از این سیاست‌ها و الگوریتم‌ها معرفی می‌شوند:

- ✓ الگوریتم تصادفی (RN): در این الگوریتم، یک کلاستر بصورت کاملا تصادفی به عنوان مقصد انتخاب شده و درخواست به آن ارسال می‌شود.
- ✓ الگوریتم روندروبین (RR): در این الگوریتم، درخواست‌ها به صورت نوبه‌ای به کلاسترهای متصل به متوازن‌کننده بار ارسال می‌شوند. هدف از این الگوریتم، توزیع متوازن بار میان کلاسترها صرف نظر از نوع درخواست‌ها می‌باشد.
- ✓ الگوریتم احتمالی (PR): در این الگوریتم، برای هر یک از رده‌ها یا انواع درخواست‌های ورودی می‌توان مقادیر احتمال مشخصی را برای گسیل به هر یک از کلاسترهای مرکز داده تعریف کرد.
- ✓ الگوریتم کمترین بهره‌وری (LU): در این الگوریتم، درخواست به کلاستری که کمترین میزان بهره‌وری را دارد،

گسیل می‌شود. هدف از این الگوریتم، بهبود بهره‌وری کلاسترهای کمتر مشغول می‌باشد.

- ✓ الگوریتم توان k (PK): در این الگوریتم، k کلاستر بصورت تصادفی از میان کلاسترهای موجود انتخاب می‌شوند. آنگاه، درخواست به آن کلاستری از میان این k کلاستر ارسال می‌گردد که کمترین طول صف ورودی را داشته باشد.

### ۵-۴- معیارهای کارایی

در این بخش، معیارهای کارایی مختلفی که از مدل‌های ارائه شده در بخش ۵-۱ و ۵-۲ قابل تعریف و ارزیابی است، معرفی می‌گردد. این معیارها را می‌توان بصورت تحلیلی با تعریف نرخ‌های پاداش مختلف به هر علامت‌گذاری[۱۲] ممکن از مدل سیستم و آنگاه محاسبه میزان پاداش مورد انتظار در حالت پایدار محاسبه نمود. شبیه‌سازی گسسته‌پیشامد سیستم، روش دیگری برای تعیین مقادیر این معیارها محسوب می‌شود. تعدادی از این معیارها در سطح کلاسترهای زیرین، و تعدادی دیگر در سطح کل مرکز داده و متوازن‌کننده بار آن قابل تعریف است. در ادامه، این معیارها معرفی می‌شود:

- ✓ $E[VM_{avail}]$: این معیار، متوسط تعداد ماشین‌های مجازی آزاد در مخزن منابع مجازی سیستم ابر را نشان می‌دهد و با استفاده از فرمول (۱) محاسبه می‌گردد:

$$E[VM_{avail}] = \sum_{i=1}^{N_{cl}} E[VM_{avail}(i)] \qquad (۱)$$

که در آن $E[VM_{avail}(i)]$ بیانگر تعداد ماشین‌های مجازی آزاد در مخزن منابع مجازی کلاستر $i$ ام می‌باشد. بعلاوه، $N_{cl}$ تعداد کلاسترهای سیستم ابری را نشان می‌دهد.

- ✓ $E[PM_{off}]$: این معیار، متوسط تعداد کل ماشین‌های فیزیکی خاموش در سیستم ابر را نشان می‌دهد و از فرمول (۲) محاسبه می‌گردد:

$$E[PM_{off}] = \sum_{i=1}^{N_{cl}} E[PM_{off}(i)] \qquad (۲)$$

که در آن $E[PM_{off}(i)]$ بیانگر متوسط تعداد ماشین‌های فیزیکی خاموش در کلاستر $i$ ام می‌باشد.

- ✓ $E[VM_{exe}^{rt}]$: این معیار، متوسط تعداد ماشین‌های مجازی در حال اجرای درخواست‌های از نوع $rt$ در سیستم ابر را نشان می‌دهد و از فرمول (۳) محاسبه می‌گردد:

$$E[VM_{exe}^{rt}] = \sum_{i=1}^{N_{cl}} E[VM_{exe}^{rt}(i)] \qquad (۳)$$

که در آن $E[VM_{exe}^{rt}(i)]$ بیانگر متوسط تعداد ماشین‌های مجازی در حال اجرا در کلاستر $i$ ام هستند که درخواست‌های از نوع $rt$ را اجرا می‌نمایند.

- ✓ $E[NQ_{clq}^{rt}]$: این معیار نشان‌دهنده متوسط تعداد درخواست‌های منتظر از نوع $rt$ در صف سراسری ابر یا CLQ می‌باشد.
- ✓ $E[NQ_{crq}^{rt}]$: این معیار، نشان‌دهنده متوسط تعداد درخواست‌های منتظر از نوع $rt$ در صف‌های CRQ کلاسترها می‌باشد. این معیار از رابطه (۴) قابل محاسبه خواهد بود:

$$E[NQ_{crq}^{rt}] = \frac{\sum_{i=1}^{N_{cl}} E[NQ_{crq}^{rt}(i)]}{N_{cl}} \qquad (۴)$$

که در آن $E[NQ_{crq}^{rt}(i)]$ متوسط تعداد درخواستهای در حال انتظار از نوع $rt$ در صف کلاستر $i$ ام است.

- ✓ $E[X_{clq}]$: این معیار، نشان‌دهنده متوسط گذردهی یا توان عملیاتی صف سراسری CLQ می‌باشد. به بیان دقیق‌تر، این معیار تعداد متوسط درخواست‌های گذر کرده از این صف در واحد زمان را نشان می‌دهد.
- ✓ $E[X_{crq}]$: این معیار، نشان‌دهنده متوسط گذردهی سرور صف‌های CRQ کلاسترها می‌باشد. این معیار از رابطه (۵) قابل محاسبه خواهد بود:

$$E[X_{crq}] = \frac{\sum_{i=1}^{N_{cl}} E[X_{crq}(i)]}{N_{cl}} \qquad (۵)$$

که در آن $E[X_{crq}(i)]$ گذردهی متوسط صف کلاستر $i$ ام است.

- ✓ $E[TQ_{clq}]$: این معیار، نشان‌دهنده متوسط زمان انتظار درخواست‌های ورودی در صف سراسری CLQ می‌باشد.
- ✓ $E[TQ_{crq}]$: این معیار، نشان‌دهنده متوسط زمان انتظار درخواست‌ها در صف‌های CRQ کلاسترها می‌باشد. این معیار از فرمول (۶) قابل محاسبه خواهد بود:

$$E[TQ_{crq}] = \frac{\sum_{i=1}^{N_{cl}} E[TQ_{crq}(i)]}{N_{cl}} \qquad (۶)$$

که در آن $E[TQ_{crq}(i)]$ متوسط زمان انتظار درخواست‌های منتظر در صف کلاستر $i$ ام است.

**جدول ۱: پیکربندی مرکز داده ابری مورد مطالعه**

| پارامتر | توصیف | مقدار یا بازه |
|---|---|---|
| $\lambda_{in}^{s}$ | نرخ ورود درخواست‌های کوچک | [15-150] درخواست در ساعت |
| $\lambda_{in}^{b}$ | نرخ ورود درخواست‌های بزرگ | [10-100] درخواست در ساعت |
| $\mu_{lb}$ | نرخ رسیدگی به درخواست‌ها در LB | ۱۰۰۰ درخواست در ساعت |
| $\mu_{crm}$ | نرخ رسیدگی به درخواست‌ها در CRM | ۱۰۰ درخواست در ساعت |
| $1/\mu_{vp}$ | متوسط زمان آماده‌سازی ماشین‌های مجازی | ۲ دقیقه |
| $1/\mu_{ex}^{s}$ | متوسط زمان اجرای هر ماشین مجازی برای درخواست‌های کوچک | ۶۰ دقیقه |
| $1/\mu_{ex}^{b}$ | متوسط زمان اجرای هر ماشین مجازی برای درخواست‌های بزرگ | ۱۲۰ دقیقه |
| $1/\lambda_{on}$ | متوسط زمان روشن کردن ماشین‌های فیزیکی | ۳ دقیقه |
| $1/\lambda_{off}$ | متوسط زمان خاموش کردن ماشین‌های فیزیکی | ۱ دقیقه |
| $N_{req}^{s}$ | تعداد ماشین‌های مجازی برای اجرای درخواست‌های کوچک | ۱ |
| $N_{req}^{b}$ | تعداد ماشین‌های مجازی برای اجرای درخواست‌های بزرگ | ۳ |
| $N_{cl}$ | تعداد کلاسترها | ۳ |
| $N_{pm_i}$ | تعداد ماشین‌های فیزیکی در هر کلاستر | ۶۰ |
| $L$ | ضریب تسهیم | ۸ |
| $Q_{clq}$ | ظرفیت صف CLQ | ۵۰ |
| $Q_{crq}$ | ظرفیت صف CRQ | ۵ |

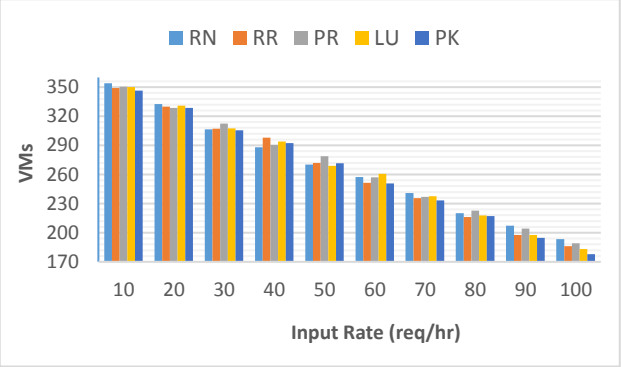
## ۶- نتایج ارزیابی کارایی

در این بخش، الگوریتم‌های توازن بار معرفی شده برای مدل پیشنهادی بر اساس معیارهای کارایی تعریف شده در بخش ۴-۵ با یکدیگر مقایسه می‌شوند. در اینجا فرض می‌شود که مرکز داده شامل سه کلاستر ابری است. برای مدل‌سازی و اجرای مدل شبیه‌سازی مرکز داده ابری مورد مطالعه، از نرم‌افزار JSimGraph از مجموعه ابزار JMT در محدوده وسیعی از مقادیر ورودی استفاده شده است. برای صرفه‌جویی در فضا، در اینجا به بخشی از خروجی حاصل از این ارزیابی اشاره خواهد شد. بسیاری از پارامترهای ورودی مدل پیشنهادی، بر اساس مقادیر به‌کار رفته در مطالعات پیشین بوده است [۴, ۲۶-۲۴]. در این مطالعه فرض شده است که درخواست‌های کاربران از دو نوع کوچک و بزرگ می‌باشد. درخواست‌های کوچک، به یک ماشین مجازی برای اجرای فرامین کاربر نیاز دارند؛ درحالیکه لازمه تکمیل درخواست‌های بزرگ، تخصیص و اجرای سه ماشین مجازی است. در همه کلاسترها فرض می‌شود که در ابتدای کار، همه ماشین‌های فیزیکی خاموش هستند و بنابراین، هیچ ماشین مجازی آزاد یا آماده‌ای وجود ندارد. مقادیر به کار رفته در پیکربندی و اجرای مدل ارائه شده، به صورت خلاصه در جدول ۱ آمده است.

پیاده‌سازی الگوریتم احتمالی یا PR، به این شکل در نظر گرفته شده است که درخواست‌های کوچک با احتمال ۰.۱، ۰.۳، و ۰.۶ به ترتیب به کلاسترهای یک، دو، و سه ارسال می‌شوند. همچنین درخواست‌های بزرگ با احتمال ۰.۶، ۰.۳، و ۰.۱ به ترتیب به کلاسترهای یک، دو، و سه گسیل می‌گردند. جهت پیاده‌سازی الگوریتم توان k یا PK، مقدار k برابر دو فرض شده است. بنابراین، در زمان دریافت هر درخواست از ورودی، ابتدا دو کلاستر از میان سه کلاستر موجود بصورت تصادفی انتخاب می‌گردند؛ آنگاه، از میان این دو کلاستر، درخواست به کلاستری ارسال می‌گردد که طول صف ورودی آن کوتاه‌تر باشد.

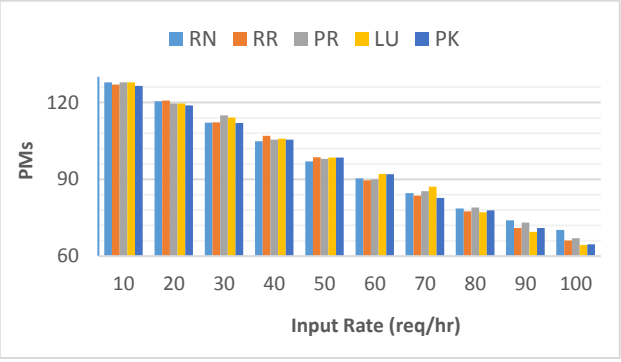
در محاسبه مقادیر خروجیِ معیارهای مطلوب کارایی، بازه اطمینان برابر ۹۹ درصد و حداکثر درصد خطای نسبی برابر ۳ درصد لحاظ شده است. همه نمودارهای ارائه شده به عنوان خروجی ارزیابی کارایی بر مبنای نرخ ورود درخواست‌های بزرگ، یعنی از ۱۰ تا ۱۰۰ درخواست در ساعت رسم شده‌اند. در همین بازه، نرخ ورود درخواست‌های کوچک، از ۱۵ تا ۱۵۰ درخواست در ساعت متغیر بوده است.

شکل ۴ مقادیر متوسط تعداد ماشین‌های مجازی آزاد در مخزن منابع مجازی سیستم ابر، $E[VM_{avail}]$ ، را برای الگوریتم‌های مختلف توزیع بار کاری نشان می‌دهد. تعداد این ماشین‌های مجازی از ۳۵۴ ماشین در الگوریتم RN و نرخ ورود ۱۰ درخواست در ساعت تا ۱۷۷ ماشین در الگوریتم PK و نرخ ورود ۱۰۰ درخواست تغییر می‌نماید. به نظر می‌رسد که با افزایش نرخ ورود درخواست‌ها، ماشین‌های مجازی بیشتری در حال اجرا خواهند بود و بنابراین، تعداد ماشین‌های مجازی آزاد کاهش می‌یابد. همانطور که در شکل ملاحظه می‌شود، در مقادیر کمینه و بیشینه نرخ ورود، تعداد ماشین‌های مجازی آزاد در الگوریتم RN بیش از چهار الگوریتم دیگر است؛ در حالیکه در مقادیر میانی نرخ ورود (۳۰ تا ۶۰ درخواست در ساعت)، الگوریتم‌های PR، RR و LU عملکرد بهتری از خود نشان می‌دهند.



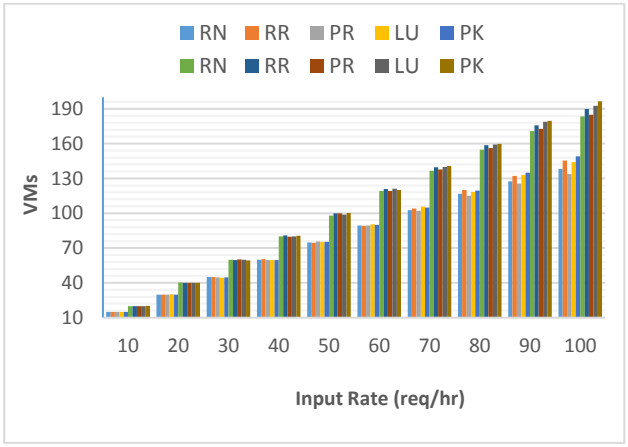

**شکل ۴: تعداد ماشین‌های مجازی آزاد در مخزن منابع مجازی سیستم ابر**

شکل ۵ مقادیر متوسط تعداد ماشین‌های فیزیکی خاموش در سیستم ابر، $E[PM_{off}]$، را برای الگوریتم‌های مختلف توزیع بار کاری نشان می‌دهد. همانطور که در شکل ملاحظه می‌شود، تعداد این ماشین‌های مجازی از ۱۲۸ ماشین در الگوریتم‌های RN و PR و نرخ ورود ۱۰ درخواست در ساعت تا ۶۴ ماشین در الگوریتم LU و PK و نرخ ورود ۱۰۰ درخواست متغییر است. بنابراین در مقدار کمینه نرخ ورود درخواست‌ها به سیستم، حدود ۷۱ درصد از ماشین‌های فیزیکی در الگوریتم‌های مورد اشاره خاموش خواهند بود. در حالت بیشینه نرخ ورودِ درخواست‌ها نیز کمینه درصد ماشین‌های خاموش، حدود ۳۵ درصد از کل ماشین‌های فیزیکی موجود در سیستم ابر می‌باشد. این کاهش در درصد ماشین‌های خاموش به این صورت قابل توجیه است که با افزایش نرخ ورود درخواست‌ها، ماشین‌های مجازی بیشتری در حال اجرا خواهند بود و در نتیجه، تعداد ماشین‌های فیزیکی روشن، افزایش و تعداد ماشین‌های فیزیکی خاموش، کاهش می‌یابد.




**شکل ۵: تعداد ماشین‌های فیزیکی خاموش در سیستم ابر**

در شکل ۶ مقادیر متوسط تعداد ماشین‌های مجازی در حال اجرا در کلاسترهای سیستم ابر، $E[VM_{exe}^{rt}]$، برای الگوریتم‌های مختلف توزیع بار کاری و درخواست‌های نوع کوچک و بزرگ نشان داده شده است. به گونه‌ای که تعداد ماشین‌های مجازی اجرا کننده درخواست‌های کوچک یا همان $E[VM_{exe}^{s}]$ با رنگ‌های سطر بالایی علائم و اختصارات این شکل نمایش داده شده است. همانطور که مشخص است، کمینه این تعداد در پایین‌ترین نرخ ورود بوده و برای تمامی الگوریتم‌های مورد بررسی تقریبا یکسان و برابر با حدود ۱۵ ماشین مجازی می‌باشد. بیشینه این تعداد هم در بالاترین نرخ ورود درخواست‌ها اتفاق می‌افتد که از حداقل ۱۳۳ برای الگوریتم PR تا حداکثر ۱۴۹ برای الگوریتم PK متغیر می‌باشد.

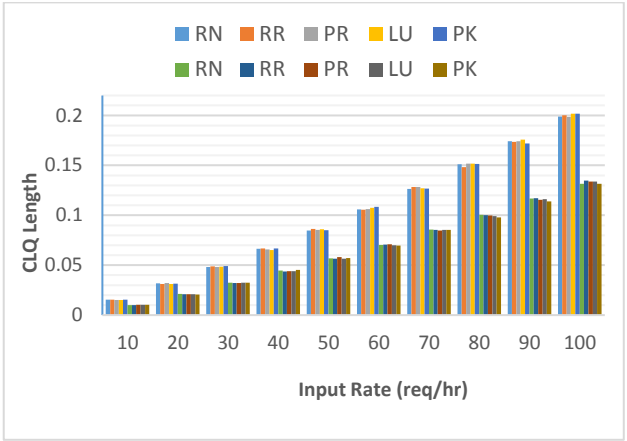



**شکل ۶: تعداد ماشین‌های مجازی اجراکننده درخواست‌های کوچک و بزرگ در سیستم ابر**

از سوی دیگر، تعداد ماشین‌های مجازی اجرا کننده درخواست‌های بزرگ یا همان $E[VM_{exe}^{b}]$ با رنگ‌های سطر پایینی علائم و اختصارات شکل ۶ نمایش داده شده است. با وجود اینکه نرخ ورود درخواست‌های بزرگ کمتر از نرخ ورود درخواست‌های کوچک فرض شده است، اما از آنجا که تعداد ماشین‌های مجازی لازم برای اجرا درخواست‌های بزرگ، سه برابر این تعداد برای ماشین‌های مجازی کوچک می‌باشد، مشاهده می‌شود که تعداد ماشین‌های مجازی در حال اجرای درخواست‌های بزرگ بیشتر است. با بررسی مقادیر نمایش داده شده در این شکل، درمی‌یابیم که کمینه این تعداد در نرخ ورود ۱۰ درخواست بزرگ در ساعت برای تمامی الگوریتم‌های مورد بررسی تقریبا یکسان و برابر با حدود ۲۰ ماشین مجازی می‌باشد. بیشینه تعداد ماشین‌های مجازی در حال اجرا هم در نرخ ورود ۱۰۰ درخواست بزرگ در ساعت مشاهده می‌شود که از حداقل ۱۸۳ برای الگوریتم RN تا حداکثر ۱۹۷ برای الگوریتم PK متغیر می‌باشد.

در شکل ۷ متوسط تعداد درخواست‌های منتظر در صف سراسری ابر، $E[NQ_{clq}^{rt}]$، برای الگوریتم‌های مختلف توزیع بار کاری نمایش داده شده است. تعداد درخواست‌های کوچک منتظر در این صف یا همان $E[NQ_{clq}^{s}]$ با رنگ‌های سطر بالایی علائم و اختصارات این

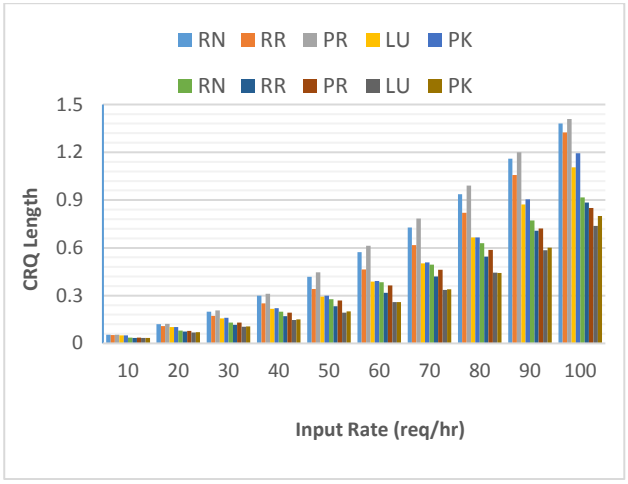
شکل نشان داده شده است. کمینه این تعداد در پایین‌ترین نرخ ورود رخ می‌دهد و برای تمامی الگوریتم‌های مورد بررسی تقریبا برابر ۰.۰۱۵ درخواست می‌باشد. بیشینه این تعداد هم در بالاترین نرخ ورود درخواست‌ها اتفاق می‌افتد که از حداقل ۰.۱۹۸ درخواست برای الگوریتم PR تا حداکثر ۰.۲۰۲ درخواست برای الگوریتم‌های LU و PK متغیر می‌باشد.




**شکل ۷: تعداد درخواست‌های منتظر در صف CLQ ابر به تفکیک درخواست‌های کوچک و بزرگ**

تعداد درخواست‌های بزرگ منتظر در صف CLQ یا همان $E[NQ^{b}_{clq}]$ با رنگ‌های سطر پایینی علائم و اختصارات شکل ۷ نشان داده شده است. همانطور که مشاهده می‌شود، تعداد درخواست‌های بزرگ حاضر در صف CLQ از تعداد درخواست‌های کوچک حاضر در این صف کمتر است. علت این امر، نرخ پایین‌تر ورود درخواست‌های بزرگ نسبت به درخواست‌های کوچک در سیستم ابر مورد مطالعه است. در این حالت، کمینه تعداد درخواست‌های منتظر بزرگ در نرخ ورود ۱۰ درخواست در ساعت رخ می‌دهد و برای تمامی الگوریتم‌های مورد بررسی تقریبا برابر ۰.۰۱۰ درخواست می‌باشد. بیشینه این تعداد هم در نرخ ورود ۱۰۰ درخواست در ساعت اتفاق می‌افتد که از حداقل ۰.۱۳۱ درخواست برای الگوریتم RN و PK تا حداکثر ۰.۱۳۵ درخواست برای الگوریتم RR متغیر است.

در شکل ۸ متوسط طول صف‌های کلاسترها، $E[NQ^{rt}_{crq}]$، برای الگوریتم‌های مختلف توزیع بار کاری نمایش داده شده است. میانگین تعداد درخواست‌های کوچک منتظر در این صف‌ها یا همان $E[NQ^{s}_{crq}]$ با رنگ‌های سطر بالایی علائم و اختصارات این شکل نشان داده شده است. کمینه این تعداد در پایین‌ترین نرخ ورود مربوط به الگوریتم‌های LU و PK و تقریبا برابر ۰.۰۵۰ درخواست می‌باشد. بیشینه این تعداد هم در بالاترین نرخ ورود درخواست‌ها اتفاق می‌افتد که مربوط به الگوریتم PR با ۱.۴۱۰ درخواست می‌باشد.

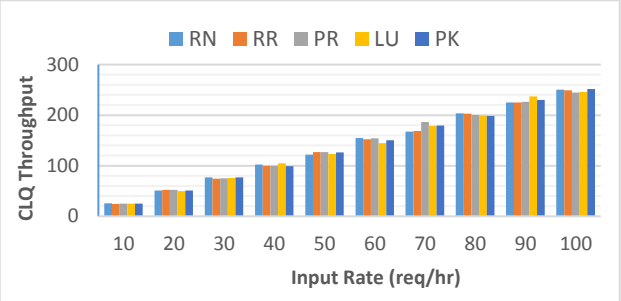



**شکل ۸: تعداد درخواست‌های منتظر در صف‌های CRQ ابر به تفکیک درخواست‌های کوچک و بزرگ**

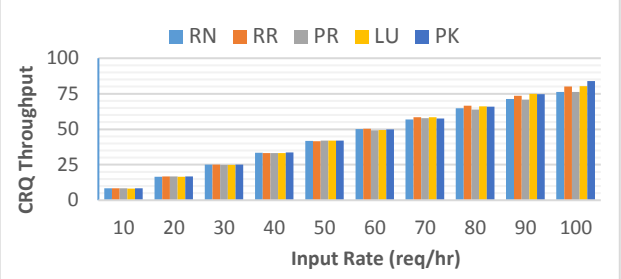
متوسط تعداد درخواست‌های بزرگ منتظر در صف‌های CRQ یا همان $E[NQ^{b}_{crq}]$ با رنگ‌های سطر پایینی علائم و اختصارات شکل ۸ نشان داده شده است. همانطور که مشاهده می‌شود، تعداد درخواست‌های بزرگ حاضر در صف‌های کلاسترها از تعداد درخواست‌های کوچک حاضر در این صف کمتر است. علت این امر، نرخ پایین‌تر ورود درخواست‌های بزرگ نسبت به درخواست‌های کوچک در سیستم ابر مورد مطالعه است. ضمن اینکه، درخواست‌های کوچک به این دلیل که نسبت به درخواست‌های بزرگ به تعداد کمتری ماشین مجازی نیاز دارند، در رقابت با درخواست‌های بزرگ، سریعتر ماشین‌های مجازی درخواستی خود را دریافت نموده و صف CRQ را ترک می‌نمایند. در این حالت، کمینه تعداد درخواست‌های منتظر بزرگ در نرخ ورود ۱۰ درخواست بزرگ در ساعت رخ می‌دهد که حداقل آن برابر ۰.۰۳۳ درخواست و مربوط به الگوریتم‌های LU و PK می‌باشد. بیشینه این تعداد هم در نرخ ورود ۱۰۰ درخواست بزرگ در ساعت اتفاق می‌افتد که از حداقل ۰.۷۳۷ درخواست برای الگوریتم LU تا حداکثر ۰.۹۱۶ درخواست برای الگوریتم RN متغیر است.

در شکل‌های ۹ و ۱۰ متوسط توان عملیاتی صف سراسری CLQ و صف‌های CRQ کلاسترها نشان داده شده است. همانطور که از شکل ۹ قابل مشاهده است، توان عملیاتی صف CLQ، $E[X_{clq}]$، از ۲۴.۷ در کمینه نرخ ورود درخواست‌ها برای الگوریتم RR تا ۲۵۱.۵ در بیشینه نرخ ورود درخواست‌ها برای الگوریتم PK تغییر می‌نماید. مطابق شکل ۱۰، میانگین توان عملیاتی صف کلاسترها، $E[X_{crq}]$، در کمینه نرخ ورود برای همه الگوریتم‌های توزیع بار

تقریبا مشابه و حدود ۸.۴ می‌باشد؛ با افزایش نرخ ورود، توان عملیاتی این صف‌ها هم افزایش می‌یابد تا در نرخ ورود ۱۰۰ درخواست بزرگ در ساعت به مقدار بیشینه خود برای همه الگوریتم‌ها می‌رسد. حداقل این مقدار بیشینه مربوط به الگوریتم RN با مقدار ۷۶.۲ و حداکثر این مقدار مربوط به الگوریتم PK با مقدار ۸۴.۰ است. به این ترتیب، در بالاترین نرخ ورود، الگوریتم PK بیشترین توان عملیاتی CLQ و CRQها را در میان الگوریتم‌های مورد مطالعه دارا است.



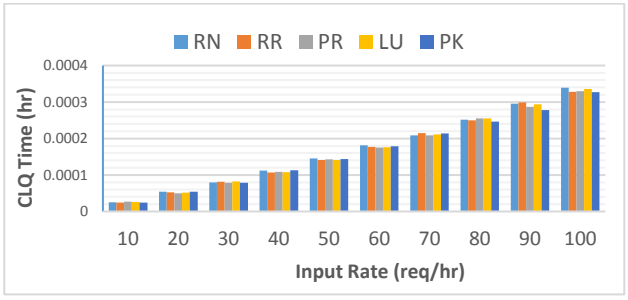

**شکل ۹: توان عملیاتی صف CLQ ابر**



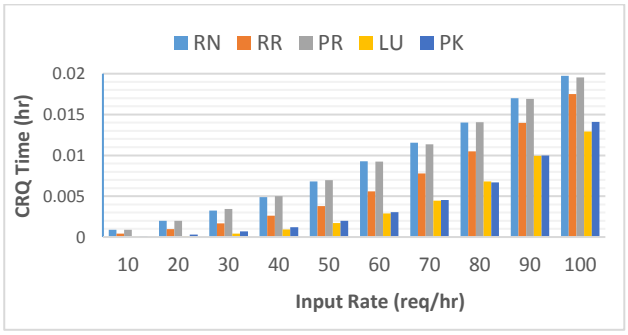

**شکل ۱۰: توان عملیاتی صف‌های CRQ کلاسترها**

در شکل‌های ۱۱ و ۱۲، متوسط زمان انتظار درخواست‌های ورودی در صف سراسری CLQ و صف‌های CRQ کلاسترها نشان داده شده است. همانطور که از شکل ۱۱ قابل مشاهده است، زمان انتظار صف CLQ، $E[TQ_{clq}]$، برای الگوریتم‌های مختلف در نرخ‌های ورود پایین، تقریبا مشابه یکدیگر می‌باشد. اما با افزایش نرخ ورود، این زمان انتظار به تدریج میان الگوریتم‌های مختلف توزیع بار متمایز می‌شود؛ به گونه‌ای که در بیشینه نرخ ورود، الگوریتم PK و پس از آن RR اندکی بهتر از سایر الگوریتم‌ها رفتار می‌نمایند و کمترین زمان انتظار در صف را ارائه می‌دهند.

مطابق شکل ۱۲ ملاحظه می‌گردد که میانگین زمان انتظار در صف‌های CRQ، $E[TQ_{crq}]$، در الگوریتم‌های مختلف توزیع بار کاری، به ازای همه نرخ‌های ورود متفاوت است. به گونه‌ای که الگوریتم‌های RN و PR بدترین عملکرد (یعنی بیشترین زمان انتظار صف) را در کل بازه نرخ ورود ارائه می‌دهند؛ در حالیکه الگوریتم LU و پس از آن PK کمترین زمان انتظار را به درخواست‌های حاضر در صف کلاسترها تحمیل می‌نمایند.




**شکل ۱۱: زمان انتظار درخواست‌ها در صف CLQ ابر**




**شکل ۱۲: زمان انتظار درخواست‌ها در صف‌های CRQ کلاسترها**

## ۷- نتیجه‌گیری

در این مقاله، مدل ترکیبی مبتنی بر نظریه صف و شبکه‌های پتری برای مدل‌سازی یک سیستم ابر زیرساخت-به-عنوان-سرویس ارائه شد. مدل پیشنهادی، بصورت سلسله‌مراتبی ارائه گردید. به این معنی که در گام نخست، یک مدل برای کلاستری از مرکز داده ابری و در گام دوم، مدل دیگری بر مبنای مدل اول برای کل مرکز داده ابری مطرح شد. مزیت استفاده از روش پیشنهادی در این مطالعه در این است که از یک سو، مدل ترکیبی، مزایای دو فرمالیسم مذکور را با یکدیگر ترکیب نموده و دست طراح مرکز ابری را برای گنجاندن قابلیت‌ها و جزئیات بیشتر در مدل می‌گشاید، و از سوی دیگر، ارائه مدل بصورت سلسله‌مراتبی، راهکاری برای غلبه بر پیچیدگی‌های مدل‌سازی جلوی پای طراح قرار می‌دهد.

بر مبنای مدل منعطف پیشنهادی، الگوریتم‌های مختلفی برای موازنه بار کاری ورودی میان کلاسترهای مرکز ابری پیشنهاد شد. سپس کارایی الگوریتم‌های معرفی شده بر اساس چند معیار مختلف و متداول ارزیابی کارایی سیستم‌های صف مقایسه گردید. نتایج عددیِ حاصل از ارزیابی کارایی نشان داد که هیچ یک از الگوریتم‌های معرفی شده، برتری قطعی نسبت به سایر الگوریتم‌ها

ندارند؛ بعنوان مثال، در حالیکه الگوریتم LU در معیارهای متوسط تعداد درخواست‌های منتظر در صف‌های CRQ و متوسط زمان انتظار در این صف‌ها بهتر از سایر الگوریتم‌ها عمل می‌نماید، الگوریتم PK بهترین کارایی را در معیارهای متوسط زمان انتظار در صف CLQ، متوسط توان عملیاتی صف‌های CRQ و متوسط تعداد VMهای در حال اجرای درخواست‌های کاربران از خود نشان می‌دهد. به هر روی، تحلیل انجام شده در این مقاله این امکان را به فراهم‌کننده ابری می‌دهد که بنا به شرایط مختلف و بسته به اینکه چه معیاری از منظر فراهم‌کننده یا مشتری از اهمیت بیشتری برخوردار است، الگوریتم خاصی را طراحی نماید، از بین الگوریتم‌های قبلا طراحی و پیاده‌سازی شده برای موازنه برگزیند، یا بین الگوریتم‌های مختلف سوئیچ نماید.

در گام بعدی، مدل‌سازی و ارزیابی کارایی و اتکاپذیری مفاهیم جدیدتری از رایانش ابری نظیر ارکستراسیون کانتینرها و رایانش بدونِ سرور مد نظر خواهد بود. همچنین مهاجرت کانتینرها در رایانش مه از زمینه‌های دیگری است که در حال مدل‌سازی و ارزیابی کارایی آن هستیم. بعلاوه، طراحی و مدل‌سازی الگوریتم‌های موازنه بار با استفاده از مدل‌های ارائه شده در این مقاله، که معیارهای مورد درخواست فراهم‌کنندگان – نظیر کاهش مصرف توان- یا مشتریان - نظیر کاهش نقض سرویس- را بهینه نمایند، از دیگر اهداف پژوهشی آینده خواهد بود.

## مراجع


[1] Y. Han, J. Chan, T. Alpcan, and C. Leckie, "Using virtual machine allocation policies to defend against co-resident attacks in cloud computing," *IEEE Transactions on Dependable and Secure Computing*, vol. 14, no. 1, pp. 95-108, 2015.

[2] S. Fakhrolmobasheri, E. Ataie, and A. Movaghar, "Modeling and evaluation of power-aware software rejuvenation in cloud systems," *Algorithms*, vol. 11, no. 10, p. 160, 2018.

[3] K. RahimiZadeh, M. AnaLoui, P. Kabiri, and B. Javadi, "Workload-Aware Placement of Multi-Tier Applications in Virtualized Datacenters," *The Computer Journal*, vol. 60, no. 2, pp. 210-239, 2017.

[4] E. Ataie, R. Entezari-Maleki, S. E. Etesami, B. Egger, D. Ardagna, and A. Movaghar, "Power-aware performance analysis of self-adaptive resource management in IaaS clouds," *Future Generation Computer Systems*, vol. 86, pp. 134-144, 2018.

[5] K. Bilal, S. U. R. Malik, S. U. Khan, and A. Y. Zomaya, "Trends and challenges in cloud datacenters," *IEEE cloud computing*, vol. 1, no. 1, pp. 10-20, 2014.

[6] M. Faraji Shoyari, E. Ataie, R. Entezari-Maleki, and A. Movaghar, "Availability modeling in redundant OpenStack private clouds," *Software: Practice and Experience*, vol. 51, no. 6, pp. 1218-1241, 2021.

[7] A. N. Asadi, M. A. Azgomi, and R. Entezari-Maleki, "Analytical evaluation of resource allocation algorithms and process migration methods in virtualized systems," *Sustainable Computing: Informatics and Systems*, vol. 25, p. 100370, 2020.

[8] S. Sebastio, R. Ghosh, and T. Mukherjee, "An availability analysis approach for deployment configurations of containers," *IEEE Transactions on Services Computing*, vol. 14, no. 1, pp. 16-29, 2018.

[9] N. Mahmoudi and H. Khazaei, "Performance Modeling of Metric-Based Serverless Computing Platforms," *arXiv preprint arXiv:2202.11247*, 2022.

[10] A. Shahidinejad, "Elasticity Management in Cloud Computing Using Colored Petri Net," *TABRIZ JOURNAL OF ELECTRICAL ENGINEERING*, vol. 50, no. 3, pp. 1261-1272, 2020.

[11] E. Ataie, A. Evangelinou, E. Gianniti, and D. Ardagna, "A Hybrid Machine Learning Approach for Performance Modeling of Cloud-Based Big Data Applications," *The Computer Journal*, 2021.

[12] R. Ghosh, F. Longo, F. Frattini, S. Russo, and K. S. Trivedi, "Scalable analytics for IaaS cloud availability," *IEEE Transactions on Cloud Computing*, vol. 2, no. 1, pp. 57-70, 2014.

[13] E. Ataie, R. Entezari-Maleki, L. Rashidi, K. S. Trivedi, D. Ardagna, and A. Movaghar, "Hierarchical stochastic models for performance, availability, and power consumption analysis of IaaS clouds," *IEEE Transactions on Cloud Computing*, vol. 7, no. 4, pp. 1039-1056, 2017.

[14] K. RahimiZadeh, M. AnaLoui, and P. Kabiri, "Multi-Tier Applications Placement in Virtualized Datacenter," *Journal of Soft Computing and Information Technology*, vol. 5, no. 3, pp. 1-15, 2016.

[15] D. P. Mahato and R. S. Singh, "Load balanced scheduling and reliability modeling of grid transaction processing system using colored Petri nets," *ISA transactions*, vol. 84, pp. 225-236, 2019.

[16] K. RahimiZadeh and A. Dehghani, "Design and evaluation of a joint profit and interference-aware VMs consolidation in IaaS cloud datacenter," *Cluster Computing*, vol. 24, no. 4, pp. 3249-3275, 2021.

[17] R. B. Cooper, "Queueing theory," in *Proceedings of the ACM'81 conference*, 1981, pp. 119-122.

[18] W. H. Sanders and J. F. Meyer, "Stochastic activity networks: formal definitions and concepts⋆," in *School organized by the European Educational Forum*, 2000: Springer, pp. 315-343.

[19] M. Ajmone Marsan, G. Conte, and G. Balbo, "A class of generalized stochastic Petri nets for the performance evaluation of multiprocessor systems," *ACM Transactions on Computer Systems (TOCS)*, vol. 2, no. 2, pp. 93-122, 1984.

[20] M. Bertoli, G. Casale, and G. Serazzi, "JMT: performance engineering tools for system modeling," *ACM SIGMETRICS Performance Evaluation Review*, vol. 36, no. 4, pp. 10-15, 2009.

[21] M. Sedaghat, F. Hernández-Rodriguez, and E. Elmroth, "Decentralized cloud datacenter reconsolidation through emergent and topology-aware behavior," *Future Generation Computer Systems*, vol. 56, pp. 51-63, 2016.

[22] E. Ataie, R. Entezari-Maleki, S. E. Etesami, B. Egger, L. Sousa, and A. Movaghar, "Modeling and evaluation of dispatching policies in IaaS cloud data centers using SANs," *Sustainable Computing: Informatics and Systems*, vol. 33, p. 100617, 2022.

[23] J. Zhang, X. Wang, H. Huang, and S. Chen, "Clustering based virtual machines placement in distributed cloud computing," *Future Generation Computer Systems*, vol. 66, pp. 1-10, 2017.

[24] D. Bruneo, A. Lhoas, F. Longo, and A. Puliafito, "Modeling and evaluation of energy policies in green clouds," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 11, pp. 3052-3065, 2014.

[25] H. Khazaei, J. Mišić, and V. B. Mišić, "Performance of an IaaS cloud with live migration of virtual machines," in *2013 IEEE Global Communications Conference (GLOBECOM)*, 2013: IEEE, pp. 2289-2293.

[26] H. Khazaei, J. Misic, and V. B. Misic, "Performance analysis of cloud computing centers using m/g/m/m+ r queuing systems," *IEEE Transactions on parallel and distributed systems*, vol. 23, no. 5, pp. 936-943, 2011.

**پاورقی‌ها:**

[1] Infrastructure as a Service
[2] Virtual Machine
[3] Physical Machine
[4] Service Level Agreements
[5] Reliability Block Diagram
[6] Continuous Time Markov Chains
[7] Serverless
[8] Stochastic Reward Networks
[9] Folding
[10] Fixed-point
[11] Multiplexing Factor
[12] Marking