



Energy-aware for Tasks Scheduling Using Dynamic Voltage and Frequency Scaling Techniques in Cloud Centers - A Case Study in the Ports and Maritime

Hamed Ghorbani¹ , Behnam Barzegar²  and Mousa Nazari³ 

1. Corresponding author, University College of Rouzbahan, Sari, Iran. Email: hamed.ghorbani67@gmail.com
2. Faculty of Computer Engineering, Islamic Azad University, Babol Branch, Babol, Iran. Email: barzegar@baboliau.ac.ir
3. Faculty of Computer Engineering, University College of Rouzbahan, Sari, Iran. Email: nazari.mousa@gmail.com

Article Info	ABSTRACT
<p>Article type: Research Article</p> <p>Article history: Received 2023 June 9 Received in revised form 2023 July 13 Accepted 2023 July 19 Published online 2023 September 16</p> <p>Keywords: Green Cloud Computing, Resource Management Task Scheduling, Ports and Maritime Organization Data Center, Reduce Energy Consumption.</p>	<p>The issue of energy management is considered as one of the main concerns in cloud computing systems to support the rapid growth of data centers and computing centers. Becomes. Dynamic voltage and frequency scaling strategies focus on reducing energy consumption as well as optimizing efficiency parameters. In this paper, we present an energy-conscious and reliable three-step scheduling algorithm called ERADVFS using the DVFS technique on a cloud center processor with dynamic voltage and frequency scaling capability. The goal of this algorithm is to consume the least amount of energy while ensuring reliability and limited task scheduling. In the first step, we determine the worst execution time of each task in each virtual machine of that type on each frequency level. Then in the next step, we divide the deadline for doing the work between the tasks. Finally, by assigning the DVFS technique, we assign tasks to the best virtual machine. The main idea in selecting a virtual machine intelligently by the DVFS technique is to take advantage of the extended amount of runtime to reduce the frequency of the virtual machine. Finally, to prove the effectiveness of the above algorithm in applications in the real environment, we have performed a series of experiments on a real production process of the Ports and Maritime Organization. The results showed that the above algorithm works better than other methods while maintaining the reliability and time required.</p>

Cite this article: Ghorbani, H., Barzegar, B. & Nazari, M. (2023). Energy-aware for Tasks Scheduling Using Dynamic Voltage and Frequency Scaling Techniques in Cloud Centers - A Case Study in the Ports and Maritime. *Engineering Management and Soft Computing*, 9 (1). 140-161. DOI: <https://doi.org/10.22091/JEMSC.2022.7335.1160>



© The Author(s)
DOI: <https://doi.org/10.22091/JEMSC.2022.7335.1160>

Publisher: University of Qom

مدیریت آگاه به انرژی جهت زمان بندی وظایف با استفاده از تکنیک مقیاس بندی پویای ولتاژ و فرکانس در مراکز داده ابری

مطالعه موردی سازمان بنادر و دریانوردی

حامد قربانی^۱✉، بهنام برزگر^۲، موسی نظری^۳

۱. نویسنده مسئول، کارشناسی ارشد فناوری اطلاعات، دانشکده فنی و مهندسی، موسسه آموزش عالی روزبهان، ساری، ایران. رایانامه: hamed.ghorbani67@gmail.com
۲. استادیار گروه مهندسی کامپیوتر، دانشکده فنی و مهندسی، دانشگاه آزاد اسلامی واحد بابل، بابل، ایران. رایانامه: barzegar@baboliu.ac.ir
۳. استادیار گروه مهندسی کامپیوتر، دانشکده فنی و مهندسی، موسسه آموزش عالی روزبهان، ساری، ایران. رایانامه: nazari.mousa@gmail.com

اطلاعات مقاله	چکیده
نوع مقاله: مقاله پژوهشی	مسئله مدیریت انرژی، به عنوان یکی از نگرانی‌های اصلی در سامانه‌های رایانش ابری برای پشتیبانی از رشد سریع مراکز داده و مراکز محاسباتی محسوب می‌شود. استراتژی‌های مقیاس بندی پویای ولتاژ و فرکانس بر روی کاهش مصرف انرژی و نیز بهینه‌سازی پارامترهای کارایی متمرکز شده‌اند. در این مقاله الگوریتم زمان بندی سه مرحله‌ای آگاه به انرژی و قابلیت اطمینان به نام ERADVFS با استفاده از تکنیک DVFS بر روی پردازنده مراکز ابری با قابلیت مقیاس بندی پویای ولتاژ و فرکانس ارائه می‌دهیم. هدف این الگوریتم مصرف کمترین میزان انرژی و در عین حال تضمین پایایی و محدودیت زمان بندی وظایف است. در مرحله اول بدترین زمان اجرای هر وظیفه در هر ماشین مجازی از آن نوع روی هر سطح فرکانس را تعیین می‌کنیم. سپس در مرحله بعدی مهلت انجام کار بین وظایف را تقسیم می‌کنیم. در مرحله آخر توسط تکنیک DVFS، وظایف را به بهترین ماشین مجازی اختصاص می‌دهیم. ایده اصلی در انتخاب هوشمند ماشین مجازی توسط تکنیک DVFS، سود بردن از مقدار طولانی مدت زمان اجرا، برای کاهش فرکانس ماشین مجازی است. سرانجام جهت اثبات تأثیر الگوریتم فوق در برنامه‌های کاربردی در محیط واقعی، یک سری آزمایش بر روی یک فرآیند تولید شده واقعی سازمان بنادر و دریانوردی انجام داده‌ایم. نتایج نشان داد الگوریتم فوق ضمن حفظ قابلیت اطمینان و زمان مورد نیاز، از سایر روش‌های دیگر بهتر عمل می‌نماید.
تاریخ دریافت: ۱۴۰۲/۰۳/۱۹	
تاریخ بازنگری: ۱۴۰۲/۰۴/۲۲	
تاریخ پذیرش: ۱۴۰۲/۰۴/۲۸	
تاریخ انتشار: ۱۴۰۲/۰۶/۲۵	
کلیدواژه‌ها: رایانش ابری سبز، زمان بندی وظایف مدیریت منابع، مراکز داده سازمان بنادر و دریانوردی، کاهش مصرف انرژی.	

استناد: عاطفی، حمیدرضا و رضایی نور، جلال. (۱۴۰۲). «مدیریت آگاه به انرژی جهت زمان بندی وظایف با استفاده از تکنیک مقیاس بندی پویای ولتاژ و فرکانس در مراکز داده ابری». *مدیریت مهندسی و رایانش نرم، دوره ۹ (۱)*. صص: ۱۶۱-۱۴۰. <https://doi.org/10.22091/JEMSC.2022.7335.1160>



۱) مقدمه

رایانش ابری سبز قصد دارد تا با طراحی الگوریتم‌ها و روش‌هایی برای کاهش مصرف انرژی از منابع به شکلی مؤثر و مقرون‌به‌صرفه، استفاده کند. برای دستیابی به این امر نیاز است که منابع مراکز داده با فن‌های بهره‌وری انرژی مدیریت شوند به طوری که انرژی مصرفی را کاهش دهند. یکی از جنبه‌های مهم رایانش ابری زمان‌بندی تعداد زیادی درخواست‌های وظیفه‌گردش کار است که به وسیله کاربران ارسال می‌شوند و منابع در مراکز داده ابری نه تنها باید وظیفه‌های گردش کار را اجرا کنند بلکه باید با فن‌هایی، مصرف انرژی را نیز مدیریت کنند. هدف زمان‌بندی پیدا کردن یک منبع از مجموع منابع پردازشی است که یک وظیفه برای پردازش به آن نیاز دارد به طوری که بتوان وظایف بیشتری را در زمان تعیین شده توسط کاربر پردازش کرد. زمان‌بندی مناسب از یک طرف موجب کاهش میزان مصرف انرژی مرکز داده، کاهش نقض توافق‌نامه سطح سرویس و همچنین موجب افزایش بهره‌وری منابع می‌گردد. تکنیک مقیاس‌بندی پویای ولتاژ و فرکانس (DVFS)^۱ یک روش جهت ذخیره‌سازی انرژی برای دستگاه‌های کامپیوتری عظیم است که قادر است مصرف انرژی پردازنده‌های دستگاه‌های کامپیوتری را با کاهش فرکانس پایین آورد.

سازمان بنادر و دریانوردی به جهت افزایش قدرت و انعطاف‌پذیری دستگاه‌های محاسباتی برای پاسخگویی به ذینفعان خود نیازمند استفاده از دستگاه‌های جدید محاسباتی است و در این میان رایانش ابری به عنوان مدلی فراتر از یک سیستم ارائه شده که در حال حاضر توانایی پاسخگویی به اکثر درخواست‌ها و نیازمندی‌های مشتریان سازمان بنادر و دریانوردی را دارد.

تاکنون تحقیقات زیادی در رابطه با زمان‌بندی و استفاده از تکنیک DVFS ارائه شده است. این الگوریتم‌ها در منابع (ژانگ لی و لی، ۲۰۱۷) (ژانگ لی و ایکس یو، ۲۰۱۶) قابلیت اطمینان را در نظر نمی‌گیرند.

روش پیشنهادی این مقاله شامل سه مرحله برای نگاشت هوشمندانه وظایف گردش کار برای ماشین‌های مجازی ناهمگن مراکز داده ابری است. در روش پیشنهادی پردازش‌ها را به مناسب‌ترین ماشین مجازی با میزان فرکانس مناسب تخصیص داده، درحالی‌که هم قابلیت اطمینان و هم صحت اتمام وظایف را در محیط ابر تضمین می‌نماید. جهت اثبات تأثیر روش پیشنهادی بر روی فرآیندهای واقعی، یک سری آزمایش انجام پذیرفت. نتایج نشان داد الگوریتم پیشنهادی ERADVFS^۲ ضمن حفظ قابلیت اطمینان و زمان موردنیاز، از روش‌های دیگر بهتر عمل می‌نماید.

ادامه مقاله به شرح زیر است: بخش ۲ یک مرور کلی از کارهای مرتبط فراهم می‌کند. بخش ۳ مدل‌های سیستم، وظیفه، انرژی و قابلیت اطمینان را توضیح می‌دهد. بخش ۴ روش پیشنهادی کاهش مصرف انرژی باهدف قابلیت اطمینان را بیان می‌کند. بخش ۵ ارزیابی روش پیشنهادی را بیان می‌کند و در نهایت در بخش ۶ نتیجه‌گیری و کارهای آینده بحث می‌شود.

۲) پیشینه پژوهش

علت پیدایش رایانش ابری این بوده که اجازه دسترسی به داده‌های مشترک را برای اجرای محاسبات به‌طور مؤثر در منابع

^۱ Dynamic Voltage and Frequency Scaling

^۲ Energy Reliability Availability Dynamic Voltage and Frequency Scaling

ابر فراهم کند (ژی، چن، شیائو، ژو و لی، ۲۰۱۷). در محیط ابر، زمان‌بندی وظیفه برای تخصیص وظایف به منابع مناسب برای به حداکثر رساندن استفاده از منابع، افزایش عملکرد و کیفیت سرویس استفاده می‌شود. (خاکسار، انصاری، صفری، یاری کارین و اجلالی، ۲۰۲۱). با توجه به سخت^۳ بودن زمان‌بندی وظایف در حالت کلی تحقیقات زیادی در این زمینه انجام و روش‌های زیادی برای حل پیچیدگی مسئله انجام شده است (فان، هان و یانگ، ۲۰۱۷). یکی از محبوب‌ترین روش‌ها برای زمان‌بندی وظایف گردش کار، زمان‌بندی لیست است (جندلر، کنول و سازایدز، ۲۰۲۱). الگوریتم‌های زمان‌بندی لیست موجود عبارت‌اند از: مسیر بحرانی پویا^۴، زمان‌بندی سطح پویایی^۵، مسیر بحرانی در پردازنده^۶، اولین دوره پایان^۷ ناهمگن و غیره. به‌عنوان مثال، الگوریتم HEFT که در مرجع (گارگ، ییو، انان داسیوام و بویا، ۲۰۱۱) وجود دارد از معروف‌ترین الگوریتم‌های زمان‌بندی لیست ایستا بر روی پردازنده‌های ناهمگن است. در این روش وظیفه با بالاترین رتبه را انتخاب می‌کند و آن را به یک منبع پردازشی اختصاص می‌دهد که منجر به کاهش زمان اجرای وظیفه می‌شود. نویسندگان در مرجع (فان، هان و یانگ، ۲۰۱۷) دو روش بر اساس اولین دوره پایان ناهمگن و لاس^۸ و گین^۹ را برای گردش کار پیشنهاد دادند که هر دو سعی در بهینه‌سازی زمان و هزینه دارند. DVFS ابزار دستیابی به، صرفه‌جویی در انرژی در تجهیزات سخت‌افزاری است (اسماعیل و فاردون، ۲۰۱۶). تکنیک DVFS بر اساس نیازهای متفاوت، ظرفیت محاسبات برنامه کاربردی را، بر پایه مقیاس‌گذاری فرکانس و ولتاژ پویا، برای دستیابی به، صرفه‌جویی در انرژی تنظیم می‌کند (پنگ، برزگر، یاراحمدی، معتمنی و پیروزمند، ۲۰۲۰). فرکانس پایین می‌تواند مصرف انرژی را کاهش دهد. با این حال، صرفه‌جویی در مصرف انرژی را نمی‌توان فقط با کاهش فرکانس جبران کرد، زیرا برای یک کار خاص، مصرف انرژی تنها زمانی می‌تواند کاهش یابد که ولتاژ و فرکانس به‌طور هم‌زمان کاهش یابند (برزگر، معتمنی و موقر، ۲۰۱۹). الگوریتم‌های بسیاری از روش DVFS استفاده کرده‌اند (اصغری و سهرابی، ۲۰۲۱). در واقع پیاده‌سازی DVFS بستگی به پیش‌بینی موفقیت‌آمیز تعداد و زمان اجرای وظایف پردازشی در میزبان دارد (فاتحی، معتمنی، برزگر و گل‌سرخ‌تبرامیری، ۲۰۲۱). بر مبنای فناوری DVFS، نویسندگان در مرجع (حسن، سالم و سعد، ۲۰۲۰) یک الگوریتم برای کاهش مصرف انرژی وظایف موازی پیشنهاد کردند. اسماعیلیان و همکاران (اسماعیلیون و تاپگولو، ۲۰۲۰) یک الگوریتم زمان‌بندی تکاملی چند هدف را بر اساس تکنیک شبکه عصبی برای بهینه‌سازی قابلیت اطمینان و میک اسپن^{۱۰} و مصرف انرژی پیشنهاد نمودند. در مرجع (ژی، چن، لیو، وی و لی، ۲۰۱۶) الگوریتم MRCRG را برای برنامه‌های موازی بر روی سامانه‌های جداسازی شده ناهمگن پیشنهاد کردند. الگوریتم آن‌ها ابتدا قابلیت اطمینان هر وظیفه را محاسبه می‌کند و سپس کار را به پردازنده با کمترین مصرف منابع اختصاص می‌دهد. بر اساس سامانه‌های جداسازی شده ناهمگن، زای و همکاران در مرجع (ژی، چن، ژیانو، ژو و لی، ۲۰۱۷) الگوریتم ESRG را برای به حداقل رساندن مصرف انرژی با هدف قابلیت اطمینان یک برنامه موازی پیشنهاد دادند. ژانگ و همکاران (ژانگ، لی و لی، ۲۰۱۷) همچنین الگوریتم ژنتیک دو هدف برای بهینه‌سازی کارایی و قابلیت اطمینان انرژی برای برنامه موازی بر روی سامانه‌های محاسباتی ناهمگن را پیشنهاد دادند.

^۲ NP-Complete

^۳ Dynamic Critical Path

^۵ Dynamic Light Scattering

^۶ Critical Path on a Processor

^۷ Heterogeneous Earliest Finish Time

^۸ LOSS

^۹ GAIN

^{۱۰} Makespan

الگوریتم Wu (وو، گو، ژو، وی، لیو و چن، ۲۰۱۸) سه مرحله دارد در مرحله نخست از مدل‌های HEFT برای زمان‌بندی اولیه استفاده می‌شود، که قابلیت اطمینان و مهلت اجرا را فراهم می‌نماید. در مرحله دوم وظایف بدون تغییر سطح فرکانس به ماشین‌های مجازی ارسال می‌شود. در نهایت در مرحله سوم از زمان اضافی بین وظایف برای کاهش بیشتر سطح فرکانس ماشین‌های مجازی جهت کاهش مصرف کلی انرژی و درعین حال حفظ قابلیت اطمینان و زمان مقرر استفاده می‌شود. شبیه به الگوریتم Wu (وو، گو، ژو، وی، لیو و چن، ۲۰۱۸) الگوریتم DEWTS (تنگ، چنگ، لی و خان، ۲۰۱۶) سه مرحله دارد در مرحله اول با استفاده از HEFT وظایف به ماشین‌های مجازی تخصیص می‌دهد و درعین حال الزامات نهایی برآورده می‌شود. در مرحله دوم DEWTS می‌کوشد تا با خاموش کردن ماشین‌های مجازی غیرفعال، تعداد ماشین‌های مجازی مورد استفاده را کاهش دهد تا انرژی مصرفی کلی کاهش یابد. مرحله نهایی مشابه مرحله ۳ الگوریتم Wu هست. الگوریتم Safari (صفری و خراسانی، ۲۰۱۸) در ابتدا مهلت اجرای هدف را بین وظایف توزیع نموده، سپس بر اساس مهلت اجرا به ترتیب صعودی مرتب نموده، در نهایت یک پردازش گر و سطح فرکانس مناسب را انتخاب می‌کند تا مصرف انرژی کاهش یابد.

الگوریتم زودترین زمان پایان ناهمگن HEFT (تاپکوگلو، حریری و وو، ۲۰۰۲) از شناخته‌شده‌ترین الگوریتم‌هاست که وظایف را به ماشین‌های مجازی تخصیص می‌دهد و کمترین میزان میک اسپن را به دست می‌آورد، اما مصرف انرژی را در نظر نمی‌گیرد. توجه داشته باشید که DEWTS، HEFT و Safari از الزام پایایی پشتیبانی نمی‌کنند، بنابراین این رویکردها برای مقایسه منصفانه اصلاح می‌شوند تا از پایایی پشتیبانی کنند.

روش پیشنهادی ERADVFS هدف نهایی جریان کار^{۱۱} را بین وظایف تقسیم می‌نماید. سپس الگوریتم پیشنهادی، فرکانس پردازنده‌های ماشین‌های مجازی به کاررفته در تکنیک DVFS را بدون از دست رفتن وظایف، کاهش می‌دهد. در نتیجه، الگوریتم ERADVFS پردازش‌ها را به مناسب‌ترین ماشین مجازی با میزان فرکانس مناسب تخصیص داده، درحالی که هم قابلیت اطمینان و هم صحت اتمام وظایف را در محیط ابر تضمین می‌نماید. نوآوری در مرحله دوم روش پیشنهادی، تقسیم مهلت هدف، بین همه وظایف است که باعث می‌شود بدترین زمان اجرا برای هر وظیفه طولانی‌تر شود؛ همچنین نوآوری دیگری در مرحله سوم روش پیشنهادی، انتخاب هوشمند ماشین مجازی توسط تکنیک DVFS، سود بردن از مقدار طولانی شده مهلت زمان اجرا برای کاهش فرکانس ماشین مجازی است.

۳) روش‌شناسی پژوهش – مدل‌های پیشنهادی پژوهش

۳-۱) مدل کاربرد

مجموعه وظایف کاربردی گردش کار W است و با استفاده از گراف جهت‌دار بدون دور (DAG^{۱۲}) مدل‌سازی می‌شود. $G\{T, E\}$ نشان‌دهنده DAG است که در آن $T = \{t_1, t_2, \dots, t_n\}$ مجموعه‌ای از n وظیفه وابسته به گردش کار و E مجموعه یال‌های بین وظایف (محدودیت تقدم)^{۱۳} را نشان می‌دهد.

^{۱۱} R Goal

^{۱۲} Directed Acyclic Graph

^{۱۳} Precedence constraint

فرض شده که وظایف انحصاری^{۱۴} بوده و هر وظیفه t_r دارای تعداد میلیون دستورالعمل است که باید اجرا شوند و با i_r نشان داده می‌شود. هر وظیفه t_r دارای مجموعه‌ای از پیشین‌ها^{۱۵} و جانشین‌ها^{۱۶} است که به ترتیب با $\text{pred}(t_r)$ و $\text{succ}(t_r)$ نشان داده می‌شوند. وظیفه ورودی برای گراف جهت‌دار بدون دور (DAG) دارای $\phi = \text{pred}(t_{\text{entry}})$ و وظیفه خروجی دارای $\phi = \text{succ}(t_{\text{exit}})$ است. هر یال $e_{r,s} \in E$ نشان‌دهنده پیوند بین t_r و t_s است که دارای وزن $w_{r,s}$ بوده و مقدار داده‌هایی که باید از وظیفه t_r به t_s منتقل شوند را نشان می‌دهد. هزینه ارتباط (CC) بین دو وظیفه t_r و t_s به ترتیب به vm_i و vm_j اختصاص داده شده و توسط رابطه ۱ نشان داده شده است.

اگر دو وظیفه به ماشین مجازی (vm) یکسان اختصاص داده شوند، آنگاه هزینه ارتباط بین آن‌ها صفر است.

$$CC_{r,s} = \begin{cases} \frac{w_{r,s}}{b_{i,j}} & \text{if } i \neq j \\ 0 & \text{if } i = j \end{cases} \quad (1)$$

۳-۲) مدل بازیابی خطا

هر وظیفه به یک vm با سطح فرکانس مشخص اختصاص داده می‌شود. در نتیجه زمان اجرای وظیفه t_r که به vm_{if_y} اختصاص داده شده با $ET(t_r, vm_{if_1})$ نشان می‌دهیم.

$$ET(t_r, vm_{if_y}) = \frac{i_r}{c_{vmi}} * \frac{vm_{if_{max}}}{vm_{if_y}} \quad (2)$$

در این رابطه i_r تعداد دستورالعمل‌هایی است که باید برای c_{vmi} اجرا شود و برابر حداکثر ظرفیت vm_i در MIPS است، $vm_{if_{max}}$ حداکثر سطح فرکانس برای vm_i و vm_{if_y} سطح فرکانس f_y بوده که می‌تواند برابر با هر مقداری از $f = \{f_1, f_2, \dots, f_k\}$ برای vm_i باشد.

۳-۳) مدل انرژی

این مدل کل انرژی مصرفی را برای n ماشین مجازی ناهمگن، از نظر ریاضی نشان می‌دهد. کل مصرف توان در طول اجرای وظیفه t_r روی پردازنده فیزیکی vm_i در سطح فرکانس f_y ، با $P(t_r, vm_{if_y})$ نشان داده شده که در رابطه (۳) مشاهده می‌گردد. این رابطه شامل مصرف توان پویا و ایستا است که به ترتیب با $P_{dynamic}$ و P_{static} نشان داده می‌شود.

$$P(t_r, vm_{if_y}) = P_{dynamic}(t_r, vm_{if_y}) + P_{static}(t_r, vm_{if_y}) \quad (3)$$

همان‌طور که در رابطه (۴) نشان داده شده توان پویا به سطح فرکانس f_y از vm_i بستگی دارد.

$$P_{dynamic} = P(t_r, vm_{if_y}) = \sigma * (vm_{iv_y})^2 * vm_{if_y} \quad (4)$$

در این رابطه σ ثابت است، vm_{iv_y} ولتاژ تغذیه v_y برای vm_i است که وظیفه t_r به آن تخصیص داده شده است و vm_{if_y} سطح فرکانس f_y برای آن vm_i است که وظیفه t_r به آن تخصیص داده شده است.

¹⁴ Non-preemptive

¹⁵ Predecessor

¹⁶ Successor

بر اساس تحقیقات اخیر (گرواوت، مونتیل، کوستا، کالهریوس، بویا و الکساندرو، ۲۰۱۳)، توان پویا انرژی بیشتری نسبت به توان ایستا مصرف می‌کند. در نتیجه، در این مقاله هدف ما کاهش توان پویا است؛ بنابراین مجموع مصرف توان را برابر با توان پویا که همان انرژی است در نظر می‌گیریم، همان‌طور که در معادلات (۵) و (۶) نشان داده شده است.

$$P(t_r, vm_{ify}) = P_{dynamic}(t_r, vm_{ify}) \quad (5)$$

$$E(t_r, vm_{ify}) = E_{dynamic}(t_r, vm_{ify}) \quad (6)$$

بر اساس تکنیک DVFS مورداستفاده در این مقاله، کاهش سطح فرکانس ماشین مجازی (vm) به کاهش مصرف انرژی منجر خواهد شد. در این مقاله، انرژی مصرف‌شده توسط ماشین‌های مجازی در حالت بیکار^{۱۷} و انرژی تلف‌شده برای تغییر ماشین مجازی از حالتی به حالت دیگر، نادیده گرفته می‌شود.

(۴) رویکرد زمان‌بندی وظیفه

هدف این مقاله یافتن یک الگوریتم زمان‌بندی برای زمان‌بندی وظایف گردش کار T به مجموعه‌ای از ماشین‌های مجازی ناهمگن است. وظایف گردش کار T با گراف جهت‌دار بدون دور (DAG) نشان داده می‌شوند.

از آنجا که گردش کار از مجموعه‌ای از n وظیفه وابسته تشکیل شده است، بنابراین برای تضمین محدودیت تقدم در بین وظایف، هر وظیفه باید زمانی شروع به اجرا کند که اجرای تمامی وظایف قبلی به پایان رسیده باشد؛ بنابراین هر وظیفه دارای ST_{vm_{ify}, t_r} و FT_{vm_{ify}, t_r} است که به ترتیب نشان‌دهنده زمان واقعی شروع و پایان وظیفه t_r اختصاص یافته به vm_i در سطح فرکانس f_y است. برای تضمین محدودیت تقدم وظیفه t_r ، زمان شروع واقعی ST_{vm_{ify}, t_r} بر اساس رابطه (۷) و زمان پایان واقعی FT_{vm_{ify}, t_r} بر اساس رابطه (۸) محاسبه می‌شود.

$$ST_{vm_{ify}, t_r} = \begin{cases} 0 & \text{if } t_r \text{ is an entry task} \\ t_{s \in pred(t_r)}^{max} (FT_{vm_{ify}} + Cc_{r,s}) & \text{other wise} \end{cases} \quad (7)$$

$$FT_{vm_{ify}, t_r} = ST_{vm_{ify}, t_r} + ET_{worst}(t_r, vm_{ify}) \quad (8)$$

بنابراین، همان‌طور که در رابطه (۹) نشان داده شده، زمان کل گردش کار M بازمان پایان آخرین وظیفه در گردش کار برابر است.

$$M = t_{r \in T}^{max}(FT_{vm_{ify}, t_r}) \quad (9)$$

همان‌طور که در رابطه (۱۰) نشان داده شده، پایایی کل گردش کار با R نشان داده می‌شود که با حاصل ضرب تجمعی (ژانگ، لی، لی و لی، ۲۰۱۷) پایایی وظایف پس از نگاهت به ماشین‌های مجازی برابر است.

$$R = \prod_{u=1}^n R(t_u, vm_{ify}) \quad (10)$$

همان‌طور که در رابطه (۱۱) نشان داده شده، کل انرژی مصرف‌شده در یک گردش کار با E نشان داده می‌شود که با مجموع تجمعی انرژی مصرف‌شده پس از نگاهت به ماشین‌های مجازی برابر است.

¹⁷ Idle State

$$E = \sum_{u=1}^n E(t_u, vm_{if_y}) \quad (11)$$

هدف الگوریتم زمان‌بندی، انتخاب مناسب‌ترین vm_i و سطح فرکانس f_y برای تخصیص وظایف به ماشین مجازی در حین تضمین الزام پایایی کلی R_{goal} و محدودیت زمان‌بندی D با کمترین مقدار مصرف انرژی است؛ بنابراین، الگوریتم زمان‌بندی برای یک گردش کار باید:

مجموع انرژی E را به حداقل برساند و درعین حال $R \geq R_{goal}$ و $M \leq D$ را تضمین کند.

۵) رویکرد پیشنهادی زمان‌بندی وظایف

در این مقاله رویکردی باهدف زمان‌بندی وظایف گردش کار به مجموعه‌ای از ماشین‌های ناهمگن در محیط‌های ابری ارائه شده است. هدف این الگوریتم مصرف کمترین میزان انرژی و درعین حال تضمین پایایی و محدودیت زمان‌بندی سیستم ابری است. تمایز رویکرد روش پیشنهادی در تقسیم هدف نهایی جریان کار بین وظایف است.

مجموعه ماشین‌های مجازی ناهمگن را بر اساس نوع ماشین مجازی به چند زیر لیست (لیست فرعی) تقسیم می‌کنیم. همان‌طور که در شکل ۱ نشان داده شده، فرض می‌کنیم گردش کاری متشکل از شش وظیفه و دو نوع ماشین مجازی داشته باشد که هر نوع دارای فهرستی از دو ماشین مجازی است. هر لیست از ماشین‌های مجازی از یک نوع دارای فرکانس، ولتاژ و حداکثر ظرفیت پردازش یکسان است، به این معنا که اگر vm_i از نوع پردازنده ۱ باشد، آنگاه مقدار $ET_{worst}(t_r, vm_{if_y})$ برای تمامی ماشین‌های مجازی از نوع ۱ یکسان است، همان‌طور که در رابطه (۱۲) مشاهده می‌شود.

$$ET_{worst}(t_r, vm_{type1_{f_y}}) = ET_{worst}(t_r, vm_{if_y}) \quad (12)$$

در این رابطه، $ET_{worst}(t_r, vm_{type1_{f_y}})$ بدترین زمان اجرای وظیفه t_r روی هر ماشین مجازی از نوع ۱ در سطح فرکانس f_y است.

برای هر نوع ماشین مجازی یک جدول مراجعه ایجاد می‌کنیم که شامل بدترین زمان اجرای هر وظیفه در هر ماشین مجازی از آن نوع در هر سطح فرکانس است، همان‌طور که در شکل ۱ نشان داده شده است.

شکل ۱ فلوچارت کلی روش پیشنهادی زمان‌بندی وظایف را نشان می‌دهد. زمان‌بندی وظایف از سه مرحله تشکیل شده است: مرحله زمان‌بندی اولیه، مرحله تقسیم‌بندی مهلت و مرحله انتخاب هوشمند ماشین مجازی توسط تکنیک DVFS.

مرحله زمان‌بندی اولیه با محاسبه هر وظیفه $ET_{worst}^{max}(t_r)$ که حداکثر مقدار زمان اجرای هر وظیفه در حداکثر سطح فرکانس ماشین‌های مجازی، گردش کار را برای مرحله تقسیم‌بندی مهلت آماده می‌کند. در صورت وجود یال بین دو وظیفه t_s و t_r ، میانگین هزینه ارتباط $(\overline{C_{r,s}})$ برای هر دو وظیفه را نیز محاسبه می‌کند.

مرحله تقسیم‌بندی مهلت، زمان رسیدن به مهلت هدف را بین وظایف گردش کار تقسیم می‌کند. در نتیجه، بدترین زمان اجرا برای هر وظیفه طولانی‌تر می‌شود تا به زمان اجرای مورد انتظار برای t_r (یعنی، $EET(t_r)$) تبدیل شود. همچنین، به وظیفه t_r یک مقدار مهلت اختصاص داده شده که با $d(t_r)$ نشان داده می‌شود، همان‌طور که در شکل ۱ مشاهده می‌شود.

ما از مهلت هر وظیفه برای افزایش زمان اجرای آن وظیفه تا رسیدن به D (مهلت) برای وظیفه خروج استفاده می‌کنیم و محدودیت تقدم را حفظ می‌کنیم.

انتخاب هوشمند ماشین مجازی توسط تکنیک DVFS، مقادیر مهلت وظایف را به ترتیب صعودی مرتب می‌کند. سپس انتخاب هوشمند ماشین مجازی توسط تکنیک DVFS از جدول مراجعه برای انتخاب هر وظیفه استفاده می‌کند. کل انرژی مصرف‌شده برای اجرای وظیفه t_r در سطح فرکانس انتخاب‌شده برای هر ماشین مجازی از نوع ۱، نزدیک‌ترین سطح فرکانس برای هر نوع پردازنده برای پایان اجرای وظیفه t_r نزدیک به $EET(t_r)$ است که با $E(t_r, vm_{type\ fselected}) \cdot vm_i$ نشان داده می‌شود. توجه داشته باشید که اگر vm_i از نوع پردازنده ۱ باشد، آنگاه مقدار $E(t_r, vm_{i\ fy})$ برای همه ماشین‌های مجازی از نوع ۱ یکسان است، همان‌طور که در رابطه (۱۳) نشان داده شده است.

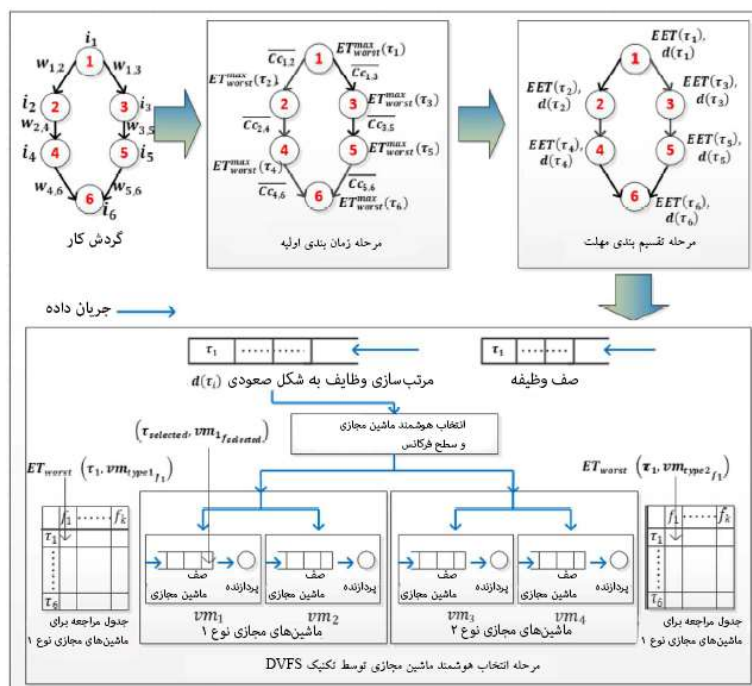
$$E(t_r, vm_{type\ 1\ fy}) = E(t_r, vm_{i\ fy}) \quad (13)$$

سپس، انرژی مصرف‌شده برای اجرای وظیفه t_r در فرکانس انتخاب‌شده برای هر نوع با $E(t_r, vm_{type\ fselected})$ محاسبه می‌شود، همان‌طور که در رابطه (۳) نشان داده شده است. سپس، نوع ماشین‌های مجازی‌ای که انرژی کمتری مصرف می‌کنند را انتخاب می‌کنیم و پس از آن زمان پایان اجرای وظیفه t_r را محاسبه می‌کنیم. در آخر، ماشین دارای زودترین زمان پایان اجرای وظیفه t_r در میان تمامی ماشین‌های مجازی موجود در زیر لیست نوع منتخب را انتخاب می‌کنیم. اگر زمان پایان وظیفه t_r در ماشین مجازی انتخاب‌شده کمتر یا برابر با آخرین مهلت وظیفه t_r باشد، آنگاه وظیفه را با سطح فرکانس انتخاب‌شده به این ماشین مجازی اختصاص می‌دهیم. اگر این‌طور نباشد، ماشین دارای زودترین زمان پایان وظیفه t_r در میان ماشین‌های مجازی موجود در زیر لیست انواع دیگر را انتخاب می‌کنیم.

پس از اختصاص وظایف به ماشین‌های مجازی، R کلی گردش کار را محاسبه می‌کنیم. اگر $R < R_{goal}$ باشد، سطح فرکانس انتخاب‌شده برای همه وظایف یک واحد افزایش می‌یابد و آن‌ها مجدداً به ماشین‌های مجازی نگاشت می‌شوند. سپس مراحل قبلی را تکرار می‌کنیم تا $R \geq R_{goal}$ شود.

۶ روش تجزیه و تحلیل داده‌ها

مرکز داده سازمان بنادر و دریانوردی متشکل از چندین ماشین مجازی از چند پردازنده واقعی ناهمگن است. تعدادی از ماشین‌های مجازی روی پردازنده Intel® Xeon® Silver 4210 CPU @ 2/20Ghz و موارد دیگر روی پردازنده Intel® Xeon(R) CPU E5-2680 v4 @ 2/40GHZ اجرا می‌شوند. هر ماشین مجازی دارای پردازنده فیزیکی از نوع متفاوتی است.



شکل ۱- نمای کلی زمان بندی گردش کار با استفاده از رویکرد زمان بندی وظایف

فرکانس، ولتاژ ورودی و توان مصرفی با استفاده از رابطه (۱۴) (لیو، چن، یینگ، ژانگ، لی، ژیانگ و چی، ۲۰۱۶) محاسبه شده و در جدول ۱ مشخصات پردازنده‌ها لیست شده است.

$$p = CV^2f \tag{۱۴}$$

جدول ۱ مشخصات پردازنده‌ها (گوگا، پارودی، رویو و ترزو، ۲۰۱۸)

Intel(R) Xeon(R) CPU E5-2680 v4 @ 2/40GHz C = 20M			Intel® Xeon® Silver 4210 CPU @ 2/20Ghz C = 13/75		
فرکانس	ولتاژ	توان مصرفی	فرکانس	ولتاژ	توان مصرفی
2/7 GHz	0/60 V	19 W	2/4 GHz	0/60 V	11 W
2/8 GHz	0/81 V	36 W	2/6 GHz	0/71 V	18 W
2/9 GHz	0/90 V	46 W	2/7 GHz	0/79 V	23 W
3 GHz	1/05 V	66 W	2/8 GHz	0/81 V	25 W
3/2 GHz	1/14 V	83 W	2/9 GHz	0/90 V	32 W
3/3 GHz	1/22 V	98 W	3/1 GHz	1/14 V	55 W
3/5 GHz	1/35 V	127/5 W	3/2 GHz	1/5 V	100 W

۷) الگوریتم مرحله زمان بندی اولیه

مرحله زمان بندی اولیه آمادگی برای مرحله تقسیم بندی مهلت را ایجاد می کند. در این مرحله، ابتدا جدول مراجعه محاسبه و تکمیل می شود. جدول مراجعه شامل بدترین زمان اجرای هر وظیفه در هر ماشین مجازی از آن نوع روی هر سطح فرکانس مانند رابطه (۱۲) است. سپس $ET_{worst}^{max}(t_r)$ ، حداکثر مقدار $ET_{worst}(t_r, vm_{type_{fmax}})$ در بین انواع مختلف انتخاب می شود. در این مرحله، میانگین هزینه ارتباط $(\overline{CC_{r,s}})$ بین وظایف t_r و t_s نیز همان طور که در رابطه (۱۵) نشان داده شده، محاسبه می شود.

$$\overline{CC_{r,s}} = \frac{w_{r,s}}{b} \tag{۱۵}$$

در این رابطه \bar{b} میانگین پهنای باند شبکه ارتباطی بین ماشین‌های مجازی است و با رابطه (۱۶) محاسبه می‌شود.

$$\bar{b} = \frac{\sum_{i=1}^m (\sum_{j=1}^m b_{i,j})}{m^2} \quad (16)$$

پس از مرحله زمان‌بندی اولیه، هر وظیفه t_r در گردش کار دارای $ET_{worst}^{max}(t_r)$ است و هزینه یال بین وظایف محاسبه می‌شود.

در این مقاله از نرم‌افزار متلب برای شبیه‌سازی یک مرکز داده ابری استفاده کرده‌ایم. در این آزمایش چندین فرض را برای ساده‌سازی محاسبه بر مبنای مقالات (ژوو و ووی، ۲۰۱۵)، (وو، گو، ژوو، ووی، لیو و چن، ۲۰۱۸) و (دی، روبرت، ویوین، انس لیون، کندو، ونگ و کیلو، ۲۰۱۳) در نظر گرفتیم. ابتدا فرض کردیم که میانگین پهنای باند بین وظایف $\bar{b} = 1 \text{ Gbps}$ است. بدترین تعداد خطاهایی که در طول اجرای وظیفه اتفاق می‌افتد $K_u = 2$ است.

سپس فرض کردیم که سرشار حالت برداری O_{cp} برابر با 0.5 ثانیه است. در مرحله سوم، فرض کردیم که دو خطای گذرای پردازنده‌های فیزیکی واقعی از توزیع پواسون^{۱۸} پیروی می‌کنند.

برای ارزیابی عملکرد الگوریتم، از یک گردش کار واقعی سازمان بنادر و دریانوردی استفاده کردیم. این گردش کار، فرآیند خروج یک شناور از اسکله پهلو دهی شده را مشخص می‌کند که شامل ۸ وظیفه است. شکل ۲ گراف جهت‌دار بدون دور (DAG) فرآیند فوق را نشان می‌دهد که شامل وظایف زیر است.

- وظیفه ۱: دریافت فیلهای پر شده توسط کاربر^{۱۹}
- وظیفه ۲: بررسی وضعیت مالی نماینده کشتیرانی
- وظیفه ۳: بررسی وضعیت توقیف فنی و ایمنی شناور
- وظیفه ۴: بررسی وضعیت پاس شناور
- وظیفه ۵: دریافت مقدارهای^{۲۰} فیلهای پر شده (شامل مقادیر شناور، مقادیر نماینده کشتیرانی، مقادیر محموله و ...)
- وظیفه ۶: ارسال زمان خروج شناور به سایر سامانه‌ها با استفاده از وب‌سرویس^{۲۱}
- وظیفه ۷: ارسال لیست خدمه شناور به سایر سازمان‌های مرتبط
- وظیفه ۸: درج اطلاعات در دیتابیس و به‌روزرسانی داشبورد

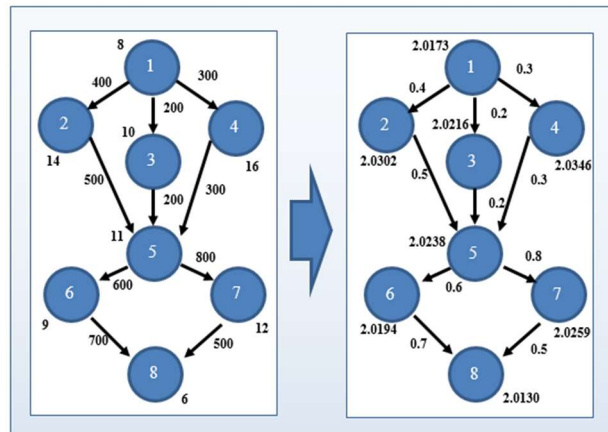
همان‌طور که در جدول ۲ نشان داده‌شده، جدول مراجعه برای نوع ۱ و نوع ۲ پر شده است. سپس حداکثر مقدار $(ET_{worst}(t_r, vm_{type1fmax}))$ را از دو نوع (نوع ۱ و نوع ۲) برای هر وظیفه انتخاب می‌کند، همان‌طور که در جدول ۳ نشان داده‌شده است.

¹⁸ Poisson distribution

¹⁹ Post Method

²⁰ Identity

²¹ Application Programming Interface



شکل ۲- گراف جهت‌دار بدون دور (DAG) هشت وظیفه قبل و بعد از مرحله ۱

جدول ۲ جدول مراجعه ماشین‌های مجازی نوع ۱ و نوع ۲

مراجعه ماشین‌های مجازی نوع ۱							
وظایف	2/7	2/8	2/9	3	3/2	3/3	3/5
t_1	0/0058	0/0056	0/0054	0/0052	0/0049	0/0047	0/0044
t_2	0/0101	0/0097	0/0094	0/0091	0/0085	0/0082	0/0078
t_3	0/0072	0/0069	0/0067	0/0065	0/0061	0/0059	0/0056
t_4	0/0115	0/0111	0/0107	0/0104	0/0097	0/0094	0/0089
t_5	0/0079	0/0076	0/0074	0/0071	0/0067	0/0065	0/0061
t_6	0/0065	0/0063	0/0060	0/0058	0/0055	0/0053	0/0050
t_7	0/0086	0/0083	0/0080	0/0078	0/0073	0/0071	0/0067
t_8	0/0043	0/0042	0/0040	0/0039	0/0036	0/0035	0/0033

مراجعه ماشین‌های مجازی نوع ۲							
وظایف	2/4	2/6	2/7	2/8	2/9	3/1	3/2
t_1	0/0021	0/0019	0/0018	0/0018	0/0017	0/0016	0/0015
t_2	0/0036	0/0033	0/0032	0/0031	0/0030	0/0028	0/0027
t_3	0/0026	0/0024	0/0023	0/0022	0/0021	0/0020	0/0019
t_4	0/0041	0/0038	0/0036	0/0035	0/0034	0/0032	0/0031
t_5	0/0028	0/0026	0/0025	0/0024	0/0023	0/0022	0/0021
t_6	0/0023	0/0021	0/0021	0/0020	0/0019	0/0018	0/0017
t_7	0/0031	0/0028	0/0027	0/0026	0/0025	0/0024	0/0023
t_8	0/0015	0/0014	0/0014	0/0013	0/0013	0/0012	0/0012

جدول ۳ نتیجه حداکثر مقدار وظایف

وظایف	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8
$ET_{worst}(t_r, vm_{type1f_{max}})$	2/0173	2/0302	2/0216	2/0346	2/0238	2/0194	2/0259	2/0130
$ET_{worst}(t_r, vm_{type2f_{max}})$	2/0062	2/0108	2/0077	2/0123	2/0085	2/0069	2/0092	2/0046
$ET_{worst}^{max}(t_r)$	2/0173	2/0302	2/0216	2/0346	2/0238	2/0194	2/0259	2/0130

الگوریتم مرحله زمان‌بندی اولیه

ورودی: مجموعه وظایف گردش کار: $T = \{t_1, t_2, \dots, t_n\}$ ، مجموعه یال‌های بین وظایف E ، مجموعه فرکانس ساعت $f = \{f_1, f_2, \dots, f_g\}$ برای هر پردازنده فیزیکی ماشین مجازی و فهرست‌های فرعی ماشین مجازی بر اساس نوع ماشین‌های مجازی.

خروجی: برگرداندن مجموعه وظیفه گردش کار پس از محاسبه $ET_{worst}^{max}(t_r)$ برای هر وظیفه و $\overline{CC}_{r,s}$ بین وظایف.

```

1. For (r=1 to n) {
2.   For (l=1 to t) {
3.     For (u=1 to k) {
4.       Compute  $ET_{worst}(t_r, vm_{type_{l_{fu}}})$  //۱۳ رابطه
5.       Fill in lookup table VMs from  $type_l$ 
6.     }
7.   }
8. }
9. For (r = 1 to n) {
10.   $ET_{worst}^{max}(t_r) = 0$ 
11.  For (l = 1 to t) {
12.    Select any  $vm$  in  $sub - list_l$  which contains all VMs from type1 .
13.    If  $(ET_{worst}(t_r, vm_{type_{l_{fmax}}}) > ET_{worst}^{max}(t_r))$ 
14.       $ET_{worst}^{max}(t_r) = ET_{worst}(t_r, vm_{type_{l_{fmax}}})$ 
15.    }
16.  }
17. }
18. For (r = 1 to n) {
19.   For (s = 1 to n) {
20.     If edge  $r_{r,s} \in E$ 
21.       Calculate  $\overline{CC}_{r,s}$  //۱۴ رابطه
22.     }
23.   }
24. }

```

۸) الگوریتم مرحله تقسیم بندی مهلت

مرحله تقسیم بندی مهلت، زمان شروع و پایان وظیفه t_r را دوباره تعیین می کند و آن ها را به عنوان زودترین زمان شروع $EST(t_r)$ و دیرترین زمان پایان $LFT(t_r)$ نام گذاری می کند.

$$EST(t_r) = \begin{cases} 0 & \text{if } t_r \text{ an entry task} \\ \max_{t_s \in pred(t_r)} (EST(t_s) + ET_{worst}^{max}(t_s) + \overline{CC}_{r,s}) & \text{other wise} \end{cases} \quad (17)$$

$$LFT(t_r) = \begin{cases} D & \text{if } t_r \text{ an exit task} \\ EST(t_r) + ET_{worst}^{max}(t_r) & \text{other wist} \end{cases} \quad (18)$$

ایده مرحله تقسیم بندی مهلت، تقسیم مهلت هدف (D) بین همه وظایف است که به طولانی شدن $ET_{worst}^{max}(t_r)$ برای هر وظیفه تا رسیدن به $EST(t_r)$ منجر می شود. همچنین، برای حفظ محدودیت تقدم در بین وظایف موجود در DAG، هر وظیفه باید دارای یک $d(t_r)$ باشد. در این مرحله برای محاسبه $EET(t_r)$ و $d(t_r)$ ، زودترین زمان شروع (EST) و دیرترین زمان پایان (LFT) برای همه وظایف گردش کار محاسبه می شود. سپس یک مسیر بحرانی^{۲۲} انتخاب می شود که شامل وظایفی با طولانی ترین مقدار LFT از وظیفه خروج به وظیفه ورود DAG است. پس از آن، فاکتور مقیاس گذاری S را برای مسیر بحرانی (CP) (صفری و خرسند، ۲۰۱۸) محاسبه می شود تا D را به طور عادلانه بین وظایف توزیع کند، همان طور که در رابطه (۱۹) نشان داده شده است.

$$S(CP) = \frac{LFT(t_{exit \in CP}) - EST(t_{entry \in CP}) - length(CP)}{\sum_{t_r \in CP} (ET_{worst}^{max}(t_r))} \quad (19)$$

در این رابطه $LFT(t_{exit \in CP})$ ، دیرترین زمان پایان (LFT) وظیفه خروج در مسیر بحرانی (CP) است و $EST(t_{entry \in CP})$ ، زودترین زمان شروع (EST) وظیفه ورود در مسیر بحرانی (CP) است. $length(CP)$ طول مسیر

بحرانی است که برابر مجموع $ET_{worst}^{max}(t_r)$ برای همه وظایف و $\overline{C_{r,s}}$ بین وظایف موجود در مسیر بحرانی است. مقدار $EET(t_{r \in CP})$ (صفری و خرسند، ۲۰۱۸) برای وظایف موجود در مسیر بحرانی با رابطه (۲۰) محاسبه می‌شود.

$$EET(t_{r \in CP}) = ET_{worst}^{max}(t_r) + (S(CP) * (ET_{worst}^{max}(t_r))) \quad (20)$$

مرحله تقسیم‌بندی مهلت، مقدار زودترین زمان شروع (EST) همه وظایف موجود در مسیر بحرانی (CP) را به‌جای $ET_{worst}^{max}(t_r)$ در رابطه (۱۷) مجدداً با استفاده از $EET(t_s) \in CP$ محاسبه می‌کند. سپس، مهلت هدف برای هر وظیفه $d(t_r \in CP)$ در مسیر بحرانی (CP) را مانند رابطه (۲۱) محاسبه می‌کند. پس از آن، $EET(t_r) \in CP$ را برای وظیفه خروج مسیر بحرانی (CP) به $D - EST(t_r)$ به‌روزرسانی می‌کند. در آخر، مقدار LFT مربوط به وظایف موجود در CP را به $LFT(t_r \in CP) = d(t_r \in CP)$ به‌روزرسانی می‌کند و این وظایف را علامت‌گذاری می‌کند.

$$d(t_{r \in CP}) = \begin{cases} D & \text{if } t_r \text{ is an exit task} \\ EST(t_r) + EET(t_r) & \text{other wise} \end{cases} \quad (21)$$

مرحله تقسیم‌بندی مهلت، با اسکن وظایف بدون علامت $t_{r \notin CP}$ از پایین DAG، $EET(t_r)$ را برای وظایف باقیمانده گردش کار محاسبه می‌کند. در طول اسکن، $LFT(t_{r \notin CP})$ و $EET(t_{r \notin CP})$ را برای وظایف بدون علامت مانند معادلات (۲۲) و (۲۳) به‌روزرسانی کرده و آن‌ها را علامت‌گذاری می‌کند. همین روش تا رسیدن به وظیفه ورودی تکرار می‌شود.

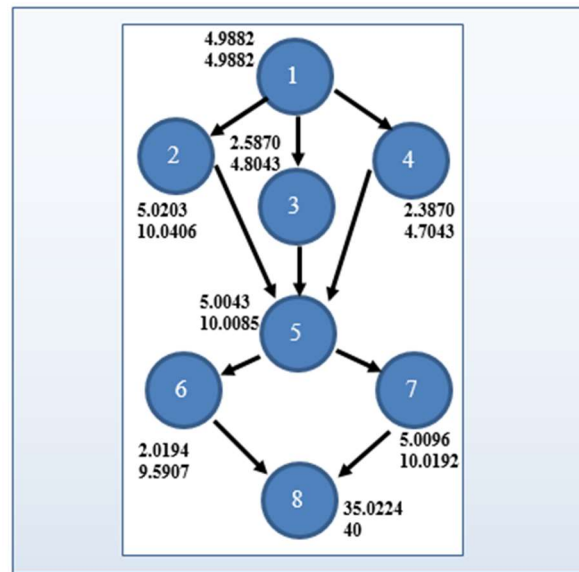
$$LFT(t_{r \notin CP}) = \begin{cases} D & \text{if } t_r \text{ is an exit task} \\ \min(LFT(t_s)_{t_s \in succ(t_r)} - EET(t_s) - \overline{C_{r,s}}) & \text{other wise} \end{cases} \quad (22)$$

$$EET(t_{r \notin CP}) = LFT(t_r) - EST(t_r) \quad (23)$$

جدول ۴ نتایج پایان مرحله تقسیم‌بندی مهلت

وظایف	EST (t_r)	LFT (t_r)	d (t_r)	EET (t_r)	علامت
t_1	0	4/9882	4/9882	4/9882	☑
t_2	5/0203	10/0406	10/0406	5/0203	☑
t_3	2/2173	4/8043	4/8043	2/5870	☑
t_4	2/3173	4/7043	4/7043	2/3870	☑
t_5	5/0043	10/0085	10/0085	5/0043	☑
t_6	7/5713	9/5907	9/5907	2/0194	☑
t_7	5/0096	10/0192	10/0192	5/0096	☑
t_8	4/9776	40	40	35/0224	☑

مهلت وظایف بدون علامت با دیرترین زمان پایان به‌روزرسانی برابر است $(LFT(t_{r \notin CP}) = d(t_{r \notin CP}))$. در آخر، هر وظیفه گردش کار، مدت‌زمان اجرا را به $EET(t_r)$ افزایش می‌دهد و یک مهلت $d(t_r)$ به آن اختصاص داده می‌شود تا محدودیت تقدم را تضمین کند. نتایج پس از پایان تقسیم مهلت در جدول ۴ و شکل ۳ نشان داده شده است.



شکل ۳ گراف جهت‌دار بدون دور از نتایج پایان تقسیم مهلت

الگوریتم مرحله تقسیم‌بندی مهلت

ورودی: مجموعه وظایف گردش کار: $T = \{t_1, t_2, \dots, t_n\}$ پس از مرحله ۱ با $ET_{worst}^{max}(t_r)$ برای هر وظیفه و $\overline{CC}_{r,s}$ بین وظایف و مهلت هدف D خروجی: بازگرداندن مجموعه وظایف گردش کار پس از محاسبه $EET(t_r)$ و $d(t_r)$ برای هر وظیفه.

1. For ($r=1$ to n) {
2. Calculate the $EST(t_r)$ and $LFT(t_r)$ // ۱۷ و ۱۶ رابطه
3. }
4. $CP = \text{null}$
5. $t_r = t_{exit}$
6. Add t_r to CP
7. While ($t_r \neq t_{entry}$) {
8. Add $t_s \in pred(t_r)$ with longest value of $LFT(t_r)$ to CP
9. $t_r = t_s$
10. }
11. Compute scaling factor S for critical path CP // ۱۸ رابطه
12. For all tasks $t_r \in CP$
13. Compute the $EET(t_r \in CP)$ // ۱۹ رابطه
14. }
15. For ($r=1$ to n) {
16. Recompute the $EST(t_r)$ using $EET(t_s \in CP)$ value for tasks in CP rather than $ET_{worst}^{max}(t_s)$ // ۱۶ رابطه
17. If ($t_r \in CP$)
18. Compute $d(t_r \in CP)$ // ۲۰ رابطه
19. Update $LFT(t_r \in CP) = d(t_r \in CP)$
20. If (t_r is an exit task)
21. Update $EET(t_r \in CP) = D - EST(t_r)$
22. }
23. Mark task t_r
24. }
25. }
26. While (exist unmark task $t_r \in CP$) {
27. $t_r \in CP = \text{last unmark task from bottom of DAG}$
28. Update $LFT(t_r \in CP)$ // ۲۱ رابطه
29. $d(t_r \in CP) = LFT(t_r \in CP)$
30. Compute the $EET(t_r \in CP)$ // ۲۲ رابطه
31. $t_r \in CP = \text{next unmark task from bottom of DAG}$
32. }

۹) الگوریتم مرحله انتخاب هوشمند ماشین مجازی توسط تکنیک DVFS

پس از تقسیم منصفانه مهلت هدف بین وظایف در مرحله قبل، وظایف به ترتیب صعودی و بر اساس مهلت $d(t_r)$ مرتب می‌شوند تا اجرای صحیح وظایف تضمین شود.

مرحله انتخاب هوشمند ماشین مجازی توسط تکنیک DVFS تخصیص هوشمندانه وظایف به بهترین ماشین مجازی ای را هدف قرار می‌دهد. ماشینی مجازی انتخاب می‌شود که اجرایی را زود تمام کند و انرژی کمتری مصرف کند. ایده اصلی پشت انتخاب هوشمند ماشین مجازی توسط تکنیک DVFS، سود بردن از مقدار طولانی $EET(t_r)$ برای کاهش فرکانس ماشین مجازی است. در نتیجه، نزدیک‌ترین مقدار $ET_{worst}(t_r, vm_{type_{fc}})$ برای هر نوع ماشین مجازی در جدول مراجعه خود را بررسی می‌کند. سپس، بر اساس $ET_{worst}(t_r, vm_{type_{fselected}})$ انتخاب شده، سطح فرکانس هر نوع را تعیین می‌کند. به این ترتیب، بهترین سطح فرکانس برای هر نوع را انتخاب می‌کند تا اجرای وظیفه قبل از مهلت تعیین شده در یک گام را تضمین کند.

علاوه بر این، $E(t_r, vm_{type_{fselected}})$ برای هر نوع انتخاب شده را مانند رابطه (۱۷) محاسبه می‌کند. سپس، انواع مختلف را به ترتیب صعودی و بر اساس مصرف انرژی $E(t_r, vm_{type_{fselected}})$ مرتب می‌کند. سپس، نوع دارای کمترین $E(t_r, vm_{type_{fselected}})$ را انتخاب می‌کند. در گام بعدی، در زیر لیست ماشین‌های مجازی را برای ماشین مجازی با زودترین زمان پایان جستجو می‌کند. سپس، بررسی می‌کند که آیا این وظیفه به آن ماشین مجازی اختصاص داده شده است یا خیر و فرکانس انتخاب شده $vm_{type_{fselected}}$ قبل از $d(t_r)$ به پایان می‌رسد یا خیر. اگر این طور باشد، وظیفه با فرکانس انتخاب شده را به آن ماشین مجازی اختصاص می‌دهد. اگر این طور نباشد، نوع بعدی ماشین‌های مجازی با مصرف انرژی کمتر را انتخاب کرده و تکرار مراحل قبلی را تا یافتن ماشین مجازی ای ادامه می‌دهد که اجرای وظیفه را قبل از پایان مهلت تمام می‌کند. انتخاب هوشمند ماشین مجازی توسط تکنیک DVFS تلاش می‌کند تا بهترین سطح ماشین مجازی و سطح فرکانس برای اتمام اجرای وظیفه قبل از پایان مهلت تعیین شده را انتخاب کند.

در آخر، مرحله انتخاب ماشین مجازی DVFS هوشمند، برای برآورده کردن الزام پایایی، پس از نگاشت وظایف به ماشین‌های مجازی، گردش کار کلی را محاسبه می‌کند. اگر $R < R_{goal}$ باشد، سطح فرکانس انتخاب شده برای وظایف $vm_{type_{fselected}}$ یک سطح افزایش می‌یابد. سپس، این وظایف را به ماشین‌های مجازی دارای $vm_{type_{fselected+1}}$ تغییر می‌دهد تا ماشین مجازی دارای مصرف انرژی کمتر را انتخاب کند. سپس R را مجدداً محاسبه کرده و بررسی می‌کند که آیا $R < R_{goal}$ هست یا خیر. اگر این طور باشد، سطح فرکانس انتخاب شده برای وظایف را مجدداً یک سطح افزایش می‌دهد و این مراحل را تکرار می‌کند تا $R \geq R_{goal}$ شود.

در این مثال، پایایی $10 - 1/7350e = R_{goal}$ است. وظایف به این شکل مرتب شده‌اند: $\{t_4, t_3, t_1, t_6, t_5, t_7, t_2, t_8\}$ تا آن وظیفه t_i که $EET(t_i)$ آن برابر 4.7043 است انتخاب شود. جدول ۵ نتایج تخصیص را نشان می‌دهد.

الگوریتم انتخاب هوشمند ماشین مجازی توسط تکنیک DVFS

ورودی: مجموعه وظایف گردش کار: $T = \{t_1, t_2, \dots, t_n\}$ پس از مرحله ۲ با $EET(t_r)$ و $d(t_r)$ برای هر وظیفه، جدول مراجعه برای هر نوع ماشین مجازی و پایایی R_{goal} . خروجی: تخصیص وظایف به ماشین‌های مجازی دارای سطح فرکانس انتخاب شده.

1. Order l in ascending order according to $d(t_r)$
2. For $(r = 1$ to $n)$ {
3. Select $vm_{type_{fselected}}$ that have $ET_{worst}(t_r, vm_{type_{fselected}})$ near to $EET(t_r)$ for each type from lookup table
4. Compute $E(t_r, vm_{type_{fselected}})$ for each type
5. Sort sub-lists of types of VMs in an ascending according to $E(t_r, vm_{type_{fselected}})$
6. For $(l=1$ to $t)$ {
7. Select vm from that $sub - list_l$ with the earliest finish time of tasks assigned to it

```

8.      If ((the finish time of the chosen vm's tasks +  $ET_{worst}(t_r, vm_{type_{f_{selected}}}) \leq d(t_r)$ )
9.      Assign task  $t_r$  to vm at frequency  $vm_{type_{f_{selected}}}$ 
10.     Break
11.     }
12.     }
13.   }
14.   Compute overall reliability of workflow R // رابطه ۱۰
15.   Queue = 0
16.   While (R <  $R_{goal}$ ) {
17.     Queue++;
18.     Repeat steps 2 to 13 of smart DVFS virtual machine selection phase algorithm with  $vm_{type_{f_{selected}+q}}$ 
19.     Recompute the overall reliability of workflow R
20.   }

```

جدول ۵ نتیجه تخصیص وظایف به ماشین‌های مجازی

وظایف	ماشین‌های مجازی	فرکانس
t_1	vm_1	2.4
t_2	vm_2	2.4
t_3	vm_3	2.7
t_4	vm_4	2.7
t_5	vm_4	2.7
t_6	vm_4	2.7
t_7	vm_4	2.7
t_8	vm_4	2.7

۱۰ یافته‌های پژوهش – معیارهای ارزیابی

در این بخش، چهار آزمایش مختلف برای ارزیابی عملکرد روش پیشنهادی در مورد چهار الگوریتم مبتنی بر تکنیک DVFS را اجرا کرده‌ایم:

Wu (وو، گوو، یوو، ووی، لیو و چن، ۲۰۱۸)، DEWTS (تنگ، چنگ، لی و خان، ۲۰۱۶)، Safari (صفری و خراسانی، ۲۰۱۸) و HEFT (تاپکوگلو، حریری و وو، ۲۰۰۲). آزمایش اول، مصرف انرژی گردش کار با مقادیر مختلف مهلت هدف (D) و مقدار پایایی ثابت (R_{goal}) را تخمین می‌زند. آزمایش دوم، میزان مصرف انرژی گردش کار با مقادیر مختلف پایایی (R_{goal}) و مقدار مهلت هدف ثابت (D) را تخمین می‌زند. آزمایش سوم، مصرف انرژی گردش کار با تعداد مختلف وظایف و مقدار ثابت مهلت و پایایی هدف را تخمین می‌زند. آزمایش چهارم، میانگین زمان اجرای موردنیاز برای پایان یافتن هر الگوریتم را ارزیابی می‌کند.

در این مقاله از سه معیار برای مقایسه الگوریتم پیشنهادی با سایر روش‌ها استفاده کرده‌ایم. معیار اول، پایایی کلی گردش کار R است که با رابطه (۱۰) محاسبه می‌شود. معیار دوم، کل انرژی مصرف‌شده نرمال است که با رابطه (۲۴) محاسبه می‌شود که نشان‌دهنده مقدار نرمال E نسبت به بالاترین مقدار است. بالاترین مقدار E برای الگوریتم HEFT است چون مصرف انرژی را در نظر نمی‌گیرد.

$$E_{normalized} = \frac{E}{E \text{ for HEFT algorithm}} \quad (24)$$

توجه داشته باشید که $E_{normalized}$ را به جای E محاسبه می‌کنیم تا نتایج دقیق‌تر باشند.

$$Execution \ Time = \frac{\sum_{i=1}^{1000} Execution \ Time_i}{1000} \quad (25)$$

در این رابطه i تعداد کارآموده‌ها و $Execution\ Time_i$ زمان اجرای شماره دنباله i از الگوریتم در حال اجرا است. ما در آزمایش‌های خود چندین فرض را برای ساده‌سازی محاسبه بر مبنای مقالات (وو، گوو، یوو، ووی، لیو و چن، ۲۰۱۸) در نظر گرفتیم. ابتدا فرض کردیم که میانگین پهنای باند بین وظایف $\bar{b} = 1\text{ Gbps}$ است. سپس فرض کردیم که سربار حالت برداری O_{cp} برابر با $0/5$ ثانیه است. در مرحله سوم، فرض کردیم که دو خطای گذرای پردازنده‌های فیزیکی واقعی از توزیع پواسون پیروی می‌کنند. همچنین فرض کردیم که میانگین نرخ ورود خطاهای گذرا روی vm_i در حداکثر میزان فرکانس $f_{max}, \lambda vm_{i_{max}}$ برابر 10^{-6} است و پارامتری که برای اندازه‌گیری حساسیت نرخ ورود خطابه مقیاس گذاری فرکانس ξ_{vm_i} استفاده می‌شود برابر دو است. در آخر، فرض کردیم که حداکثر تعداد خطاهای گذرا دو عدد باشد.

از آنجا که هدف اصلی HEFT دستیابی به کمترین زمان پایان برای زمان‌بندی، بدون در نظر گرفتن مصرف انرژی است و HEFT به بالاترین مقدار پایایی دست پیدا می‌کند. در نتیجه، HEFT را مبنای آزمایش خود در نظر می‌گیریم. بنابراین مقدار مهلت هدف را روی $D = \beta * M_{HEFT}$ تنظیم می‌کنیم که در آن β یک مقدار ثابت است و M_{HEFT} زمان کل پس از اجرای الگوریتم HEFT است. برای اطمینان از اینکه وظایف گردش کار در طول زمان‌بندی از مهلت را رعایت می‌کنند، طول مسیر بحرانی گردش کار را محاسبه می‌کنیم و سپس مطابق آن، کمترین مقدار β را انتخاب می‌کنیم. بنابراین $D = \beta * M_{HEFT}$ باید بزرگتر-مساوی طول مسیر بحرانی باشد.

(۱۱) برآورد گردش کارها با مقادیر مهلت مختلف

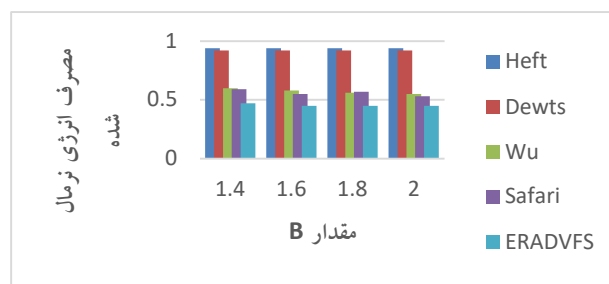
در این بخش، با استفاده از دو وظیفه مختلف عملکرد الگوریتم پیشنهادی را با سایر روش‌ها مقایسه می‌کنیم. در اینجا مصرف انرژی نرمال شده برای گردش کار واقعی با پایایی ثابت و مقادیر مهلت متغیر را محاسبه می‌کنیم. مقدار پایایی ثابت را برابر با $0/99$ تعیین می‌کنیم و مقدار مهلت بر اساس β همان‌طور که در جدول ۶ ارائه شده تغییر می‌کند که در آن $D = \beta * M_{HEFT}$ است.

جدول ۶ مقدار مهلت بر اساس β

گردش کار	تعداد وظایف	Length (CP)	M_{HEFT}	Range of β
W_1	30	196/17	143/65	1/4, 1/6, 1/8, 2/0

همان‌طور که در شکل ۴ نشان داده شده، رویکرد پیشنهادی اعمال شده بر یک گردش کار واقعی، از سایر روش‌ها

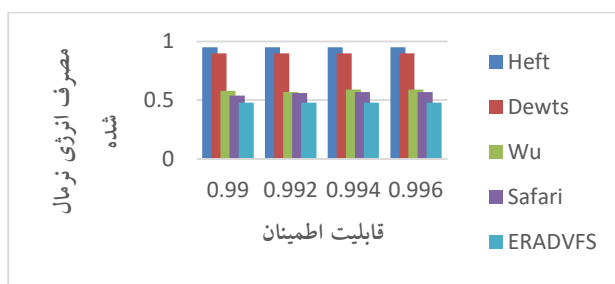
HEFT، DEWTS، Wu و Safari بهتر عمل می‌کند.



شکل ۴ مصرف انرژی نرمال شده با مقادیر مختلف β و مقدار R_{goal} ثابت

۱۲) برآورد گردش کارها با مقادیر پایایی مختلف

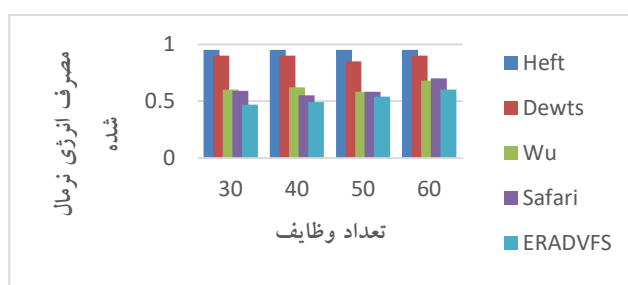
مشابه آزمایش اول، عملکرد الگوریتم پیشنهادی را با سایر روش‌ها مقایسه می‌کنیم اما در این آزمایش، مهلت ثابت و مقادیر پایایی متغیر است. مقدار ثابت β را برابر با $1/4$ تعیین می‌کنیم. مقدار پایایی R_{goal} در محدوده $0/99$ تا $0/996$ متغیر است. شکل ۵، نتایج مصرف انرژی نرمال شده در یک گردش کار واقعی را ارائه می‌دهد.



شکل ۵ مصرف انرژی نرمال شده با مقادیر مختلف R_{goal} و مقادیر β ثابت

۱۳) برآورد گردش کار با شماره وظیفه‌های مختلف

در این بخش عملکرد الگوریتم پیشنهادی در مورد شماره وظیفه را در گردش کار مورد نظر آزمایش می‌کنیم. در این موارد R_{goal} را برابر با $0/990$ و β را برابر $1/4$ تعیین می‌کنیم. تعداد وظایف از ۳۰ تا ۶۰ متغیر است. همان‌طور که در شکل ۶ نشان داده شده، بهبود مصرف انرژی نرمال شده با افزایش تعداد وظایف کاهش می‌یابد. چون وقتی تعداد وظایف افزایش می‌یابد، مقدار زمان بیکاری نیز افزایش پیدا می‌کند و به اختلاف اندک بین $EET(t_r)$ و $ET_{worst}(t_r, vm_{type}_{fc})$ می‌افزاید.



شکل ۶ مصرف انرژی نرمال شده با تعداد وظایف مختلف و R_{goal} و β ثابت

۱۴) برآورد میانگین زمان اجرای گردش کارها

در این بخش، می‌خواهیم عملکرد الگوریتم پیشنهادی را جهت برآورد میانگین زمان اجرای گردش کار با سایر روش‌ها آزمایش کنیم. برای اثبات این امر هر یک از آزمایش‌های قبلی را ۱۰۰۰ بار تکرار می‌کنیم و هر بار زمان لازم برای اجرای هر الگوریتم را محاسبه می‌کنیم. در آخر، میانگین زمان اجرا را مثل رابطه (۲۳) برحسب میلی ثانیه محاسبه می‌کنیم. میانگین زمان اجرا (یعنی، Execution Time) پس از ۱۰۰۰ بار اجرا با پایایی ثابت $0/99 = R_{goal}$ و مقادیر متغیر β در جدول ۷ ارائه شده است. به‌طور مشابه، میانگین زمان اجرا پس از ۱۰۰۰ بار اجرا با پایایی متغیر و مقادیر β برابر با $1/4$ در جدول ۸ ارائه شده است. جدول ۹ میانگین زمان اجرا برای وظایف مختلف با ثابت β برابر با $1/4$ و $R_{goal} = 0/99$ را نشان می‌دهد.

جدول ۷ میانگین زمان اجرا برای گردش کار کوچک با مقادیر مختلف β و مقدار ثابت R_goal

گردش کار	مقدار β	الگوریتم				
		ERADVFS	Safari	Wu	DEWTS	HEFT
W_1	1/4	3/439	10/219	130/710	99/954	2/670
	1/6	3/988	10/610	126/753	93/517	2/670
	1/8	4/465	11/250	128/344	98/452	2/670
	2/0	5/057	12/189	116/019	95/859	2/670

جدول ۸ میانگین زمان اجرای گردش کار کوچک با مقادیر مختلف R_goal و مقادیر ثابت β

گردش کار	مقدار R_goal	الگوریتم				
		ERADVFS	Safari	Wu	DEWTS	HEFT
W_1	0/990	3/439	10/219	130/710	99/954	2/670
	0/992	4/969	11/188	117/140	90/907	2/670
	0/994	5/419	12/719	113/435	97/407	2/670
	0/996	5/472	12/609	122/783	98/125	2/670

جدول ۹ میانگین زمان اجرا برای گردش کارها با تعداد وظایف مختلف و β و R_goal ثابت

گردش کار	تعداد وظایف	الگوریتم				
		ERADVFS	Safari	Wu	DEWTS	HEFT
W_1	30	3/439	10/219	130/710	99/954	2/670
	40	6/625	17/432	309/500	205/400	2/891
	50	7/336	19/394	421/900	274/800	3/032
	60	7/780	22/531	471/309	399/221	3/350

بر اساس جدول ۷، اگرچه HEFT دارای کمترین پیچیدگی زمانی است، اما بیشترین انرژی را مصرف می‌کند. در نتیجه، الگوریتم پیشنهادی از الگوریتم‌های Safari، DEWTS و Wu سریع‌تر و از الگوریتم HEFT کندتر است.

۱۵) نتیجه‌گیری و پیشنهادها

یکی از فن‌های پرفرمداری که امروزه برای کاهش مصرف انرژی در محیط‌های ابر سبز استفاده می‌شود، تکنیک DVFS است؛ اما این تکنیک آسیب‌پذیری سیستم نسبت به خطاهای طول اجرا را افزایش می‌دهد و این امر به کاهش کیفیت خدمات ابری منجر می‌شود؛ بنابراین، چالش این است که الگوریتم زمان‌بندی امیدوارکننده‌ای برای کاهش مصرف انرژی و در عین حال تأمین الزام پایایی سیستم ابری پیدا کنیم. در این مقاله، الگوریتم هوشمند زمان‌بندی آگاه از پایایی و انرژی ERADVFS را پیشنهاد کرده‌ایم. همان‌طور که نشان داده شد، الگوریتم پیشنهادی شامل سه مرحله برای نگاشت هوشمندانه وظایف گردش کار برای ماشین‌های مجازی ناهمگن مراکز داده ابری است و الزامات پایایی و زمان تکمیل با پیچیدگی زمانی پایین‌تر را فراهم می‌کند.

چندین آزمایش روی گردش کار واقعی انجام دادیم تا اثربخشی الگوریتم پیشنهادی در میان سایر روش‌ها را ثابت کنیم. نتایج به‌دست‌آمده از آزمایش‌های مختلف به‌وضوح نشان می‌دهد که الگوریتم پیشنهادی در تمامی موارد بدون

تخطی از مهلت و پایایی لازم در مقایسه با سایر روش‌ها به کمترین میزان مصرف انرژی نیاز دارد. همان‌طور که نشان داده‌شده، الگوریتم پیشنهادی در مقایسه با الگوریتم HEFT بر اساس گردش کار و تعداد وظایف به ترتیب به ۶۷/۴ تا ۳۵/۷ درصد کاهش مصرف انرژی دست پیدا می‌کند. علاوه بر این، الگوریتم پیشنهادی حدود ۳۵ تا ۵۵ درصد کاهش در مصرف انرژی در مقایسه با سایر روش‌ها را به دست می‌آورد.

کارهای آینده و پیشنهادها را می‌توان به دو بخش تقسیم کرد. در بخش اول می‌توان جهت محاسبه مصرف انرژی در زمان‌بندی وظایف، برای اجزای دیگر مانند دیسک، حافظه و شبکه و باهدف کاهش انرژی، پیشنهاد شود. ارائه مدلی جهت محاسبه مصرف انرژی دیگر اجزا می‌تواند به‌عنوان کارهای آتی محسوب شود. در بخش دوم می‌توان در الگوریتم پیشنهادی از فن‌های تکثیر و کلاسترینگ استفاده کرد که می‌تواند باعث صرفه‌جویی بیشتر در مصرف انرژی شود.

منابع

- Asghari, A., & Sohrabi, M. K. (2021). Combined use of coral reefs optimization and multi-agent deep Q-network for energy-aware resource provisioning in cloud data centers using DVFS technique. *Cluster Computing*. <https://doi.org/10.1007/s10586-021-03368-3>
- Barzegar, B., Motameni, H., & Movaghar, A. (2019). EATSDCD: A green energy-aware scheduling algorithm for parallel task-based application using clustering, duplication and DVFS technique in cloud datacenters. *Journal of Intelligent & Fuzzy Systems*, 36(6), 5135–5152. <https://doi.org/10.3233/JIFS-171927>
- Di, S., Robert, Y., Vivien, F., Vivien, F., ENS Lyon and INRIA, F., Profile, V., Kondo, D., Wang, C.-L., & Cappello, F. (2013). Optimization of cloud task processing with checkpoint-restart mechanism. *Proceedings of the International Conference on High Performance Computing*. <https://doi.org/http://dx.doi.org/10.1145/2503210.2503217>
- Fan, M., Han, Q., & Yang, X. (2017). Energy minimization for on-line real-time scheduling with reliability awareness. *Journal of Systems and Software*, 127, 168–176.
- Fatchi, S., Motameni, H., Barzegar, B., & Golsorkhtabaramiri, M. (2020). Energy Aware Multi Objective Algorithm for Task Scheduling on DVFS-Enabled Cloud Datacenters using Fuzzy NSGA-II. *International Journal of Nonlinear Analysis and Applications*. <https://dx.doi.org/10.22075/ijnaa.2020.21625.2283>
- Garg, S. K., Yeo, C. S., Anandasivam, A., & Buyya, R. (2011). Environment-conscious scheduling of HPC applications on distributed cloud-oriented data centers. *Journal of Parallel and Distributed Computing*, 71(6), 732–749.
- Gendler, A., Knoll, E., & Sazeides, Y. (2021). I-DVFS: Instantaneous Frequency Switch during Dynamic Voltage and Frequency Scaling. *IEEE Micro*, 1–1. <https://doi.org/10.1109/MM.2021.3096655>
- Goga, K., Parodi, A., Ruiu, P., & Terzo, O. (2018). Performance Analysis of WRF Simulations in a Public Cloud and HPC Environment (pp. 384–396). <https://doi.org/10.1007/978-3-319-61566-035>
- Guérout, T., Monteil, T., Costa, G. Da, Calheiros, R., Buyya, A., & Alexandru, M. (2013). Energy-aware simulation with DVFS. *Simulation Modelling Practice and Theory*. <https://www.sciencedirect.com/science/article/abs/pii/S1569190X13000786>
- Hassan, H. A., Salem, S. A., & Saad, E. M. (2020). A smart energy and reliability aware scheduling algorithm for workflow execution in DVFS-enabled cloud environment. *Future Generation Computer Systems*, 112, 431–448. <https://doi.org/10.1016/j.future.2020.05.040>
- Ismail, L., & Fardoun, A. (2016). Eats: Energy-aware tasks scheduling in cloud computing systems. *Procedia Computer Science*, 83, 870–877.
- Ismayilov, G., & Topcuoglu, H. R. (2020). Neural network based multi-objective evolutionary algorithm for dynamic workflow scheduling in cloud computing. *Future Generation Computer Systems*, 102, 307–322. <https://doi.org/10.1016/j.future.2019.08.012>
- Liu, D., Chen, X., Ying, Y., Zhang, L., Li, W., Jiang, L., & Che, S. (2016). MnZn power ferrite with high Bs and low core loss. *Ceramics International*, 42(7), 9152–9156. <https://doi.org/10.1016/j.ceramint.2016.03.005>
- Peng, Z., Barzegar, B., Yarahmadi, M., Motameni, H., & Pirouzmand, P. (2020). Energy-Aware Scheduling of Workflow Using a Heuristic Method on Green Cloud. *Scientific Programming*, 2020, 1–14. <https://doi.org/10.1155/2020/8898059>
- Safari, M., & Khorsand, R. (2018). Energy-aware scheduling algorithm for time-constrained workflow tasks in DVFS-enabled cloud environment. *Simulation Modelling Practice and Theory*. <https://doi.org/10.1016/j.simpat.2018.07.006>
- Tang, Z., Cheng, Z., Li, K., & Khan, S. . (2016). An energy-efficient task scheduling algorithm in DVFS-enabled cloud environment. *J. Grid Comput.*
- Topcuoglu, H., Hariri, S., & Wu, M. (2002). Performance-effective and low-complexity task scheduling for heterogeneous computing. *IEEE Transactions on Parallel and Distributed Systems*, 13(3), 260–274.

- Wu, T., Gu, H., Zhou, J., Wei, T., Liu, X., & Chen, M. (2018). Soft error-aware energy-efficient task scheduling for workflow applications in DVFS-enabled cloud. *Journal of Systems Architecture*. <https://doi.org/https://doi.org/10.1016/j.sysarc.2018.03.001>
- Xie, G., Chen, Y., Liu, Y., Wei, Y., Li, R., & Li, K. (2016). Resource consumption cost minimization of reliable parallel applications on heterogeneous embedded systems. *IEEE Transactions on Industrial Informatics*, 13(4), 1629–1640.
- Xie, G., Chen, Y., Xiao, X., Xu, C., Li, R., & Li, K. (2017). Energy-efficient fault-tolerant scheduling of reliable parallel applications on heterogeneous distributed embedded systems. *IEEE Transactions on Sustainable Computing*, 3(3), 167–181.
- Yeganeh-Khaksar, A., Ansari, M., Safari, S., Yari-Karin, S., & Ejlali, A. (2021). Ring-DVFS: Reliability-Aware Reinforcement Learning-Based DVFS for Real-Time Embedded Systems. *IEEE Embedded Systems Letters*, 13(3), 146–149. <https://doi.org/10.1109/LES.2020.3033187>
- Zhang, L., Li, K., Li, C., & Li, K. (2017a). Bi-objective workflow scheduling of the energy consumption and reliability in heterogeneous computing systems. *Inf. Sci. (Ny)*, 379, 241–256.
- Zhang, L., Li, K., Li, C., & Li, K. (2017b). Bi-objective workflow scheduling of the energy consumption and reliability in heterogeneous computing systems. *Information Sciences*, 379, 241–256.
- Zhang, L., Li, K., & Xu, Y. (2016). Joint optimization of energy efficiency and system reliability for precedence constrained tasks in heterogeneous systems. *International Journal of Electrical Power & Energy Systems*.
- Zhou, J., & Wei, T. (2015). Stochastic thermal-aware real-time task scheduling with considerations of soft errors. *Journal of Systems and Software*. <https://doi.org/https://doi.org/10.1016/j.jss.2014.12.009>