DOR :20.1001.1.2538161.1401.7.4.3.4

تاریخ دریافت: ۱۴۰۱/۰۸/۱۲
تاریخ پذیرش:۱۴۰۱/۱۰/۱۸

نوع مقاله: پژوهشی

ارائه رویکردی برای استفادهٔ بهینه از منابع با آگاهی از زمان پردازش درخواست‌ها در محیط‌های رایانش ابری

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

شفق رستگاری
دانشکده مهندسی کامپیوتر ـ دانشگاه علم و صنعت ایران ـ تهران ـ ایران
پست الکترونیکی: shafaghrastegari@gmail.com

چکیده

رایانش ابری، یک مدل محاسباتی مبتنی بر شبکه است که به منظور عرضه، مصرف و ارائه خدماتی نظیر زیرساخت، نرم‌افزار و منابع رایانشی از طریق شبکه ایجاد شده‌است. مسئله زمان‌بندی مجموعه کارها در این سیستم‌ها به مسئله‌ای مهم و پیچیده تبدیل شده‌است و حل این مسئله می‌تواند عملکرد و تعامل گره‌ها در این سیستم توزیع شده را بهبود ببخشد. الگوریتم‌های زمان‌بندی، با در نظر گرفتن کیفیت خدمات اقدام به تخصیص کارها به منابع می‌کنند و هدف از زمان‌بندی در این سیستم‌ها به حداکثر رساندن بازدهی سیستم با اختصاص دادن کارهای صحیح به ماشین‌های صحیح، به حداقل رساندن زمان اجرا و به حداکثر رساندن استفاده از منابع است. هدف از انجام این پژوهش ارائه رویکردی برای استفاده بهینه از منابع است. از آنجایی که در کارهای پیشین در خصوص زمان‌بندی وظایف، دسته‌بندی وظایف و منابع در برخی موارد توزیع دقیق آن‌ها را بر روی منابع تضمین نمی‌کنند، در این مقاله پس از دسته‌بندی وظایف و منابع به کمک برخی ویژگی‌های آن‌ها و با کمک یک شبکه عصبی به دنبال یافتن بهترین ماشین برای وظیفه انتخاب شده در سیستم می‌گردیم. لایه‌های موجود در شبکه عصبی و مرحله‌های یادگیری و در نهایت استفاده از مدل آموزش دیده به ما در انتخاب منبع مناسب موجود برای وظیفه انتخاب شده کمک می‌کند و این موضوع می‌تواند بازدهی سیستم را بهبود ببخشد. نتایج حاصل از ارزیابی راه‌حل ارائه شده حاکی از زمان اتمام وظایف کوتاه‌تر الگوریتم پیشنهادی نسبت به الگوریتم‌های تصادفی، اولین برازش و Tetris است به‌طوری که میانگین زمان اتمام وظایف حداقل ۱۰ واحد زمانی نسبت به الگوریتم اولین برازش و Tetris و همچنین تقریبا ۱ واحد زمانی از الگوریتم انتخاب تصادفی بهتر است.

واژه‌های کلیدی: زمان‌بندی وظایف، رایانش ابری، خوشه‌بندی، یادگیری تقویتی عمیق.

* نویسندهٔ مسئول

۱- مقدمه

رایانش ابری با فراهم آوردن انعطاف‌پذیری و مقیاس‌پذیری به ابزاری پراستفاده برای سازمان‌ها تبدیل شده است .[۱]رایانش ابری یک سیستم توزیع شده در مقیاس بزرگ است که مجموعه‌ای از منابع محاسباتی را از طریق اینترنت به مصرف کنندگان خود ارائه می دهد. آن‌ها خدمات و منابع را بر اساس پرداخت به ازای استفاده در هر زمان و از هر مکان به کاربران ارائه می‌دهند. ارائه‌دهندگان این سرویس از طریق فناوری مجازی‌سازی [۲]، یک کارساز فیزیکی را به‌وسیلهٔ استقرار یک لایه انتزاع در بالای منابع سخت‌افزاری یا سیستم‌عامل به چندین محیط ایزوله شده تقسیم می‌کنند. با این کار می‌توان چندین ماشین مجازی را بر روی یک ماشین فیزیکی اجرا کرد[۳]. از آن جا که ارائه دهندگان ابر باید به بسیاری از مصرف کنندگان در سیستم رایانش ابری خدمت کنند، زمان‌بندی موضوعی اصلی در استقرار سیستم رایانش ابری به منظور کاهش زمان اجرا و هزینه و در نتیجه به حداکثر رساندن استفاده از منابع است. از این رو، زمان‌بندی و تخصیص منابع و وظایف از مشکلات حیاتی در رایانش ابری است که تحقیقات زیادی روی آن انجام شده است. [٤] زمان‌بندی وظایف یکی از مهم‌ترین راه‌ها برای افزایش توان عملیاتی و استفاده از منابع است. منابع در دو سطح می‌تواند در سیستم‌های ابری زمان‌بندی شوند. یکی در سطح میزبان و دوم در سطح ماشین‌های مجازی. در سطح ماشین‌های مجازی وظایف به ماشین مناسب اختصاص داده می‌شود و در سطح میزبان، ماشین‌های مجازی را به سخت‌افزارهای فیزیکی اختصاص می‌دهند. بنابراین، زمان‌بندی وظایف به اختصاص وظایف به ماشین مجازی مناسب گفته می‌شود[٦][٥]. در فرآیند زمان‌بندی، وظایف به یک ماشین مجازی اختصاص داده می‌شوند.[۷] از این‌رو، زمان‌بندی کار، استفاده از منابع را بهینه می‌کند که این امر منجر به افزایش توان عملیاتی کل سیستم می‌شود و در عین حال کیفیت بالای خدمات را برای کاربران تضمین می‌کند.[۸] به همین خاطر، در این پژوهش به دنبال به‌دست آوردن راه حل بهینه‌ای برای زمان‌بندی وظایف هستیم که توسط آن استفاده بهینه از منابع در سیستم‌های رایانش ابری را باعث شویم. نوآوری اصلی این مقاله نسبت به روش‌های استفاده شده در این حوزه ارائه راه‌حلی برای زمان‌بندی کاراتر وظایف به وسیله استفاده از الگوریتمی با زمان اتمام کار کمتر است که این امر با کمک خوشه‌بندی وظایف و منابع از طریق در نظر گرفتن ویژگی‌هایشان و اختصاص آن‌ها به یکدیگر به گونه‌ای که قابلیت اطمینان بیشتر و توزیع دقیق‌تر وظایف به منابع را نسبت به کارهای گذشته تضمین کند محقق خواهد شد. همان‌طور که از نتایج به‌دست آمده مشخص است در این سیستم ارائه شده، شاخص میانگین زمان اتمام وظایف حداقل ۱۰ واحد زمانی نسبت به الگوریتم اولین برازش و Tetris و همچنین تقریبا ۱ واحد زمانی از الگوریتم انتخاب تصادفی بهتر عمل کرده است. انگیزه اصلی از این کار افزایش کیفیت خدمات به کاربران از جانب ارائه دهنده خدمات ابری است به‌طوری که نتایج حاصل از الگوریتم نشان دهنده دقت بیشتر در اختصاص دادن وظایف به منابع باشد. در ادامه این مقاله در بخش ۲ پژوهش‌های مرتبط انجام شده را مرور می‌کنیم. در بخش ۳ روش پیشنهادی برای به‌دست آوردن زمانبندی بهینه را شرح می‌دهیم. در بخش ٤ به آزمایش و ارزیابی روش پیشنهادی می‌پردازیم. در انتها در بخش ٥ به نتیجه‌گیری از پژوهش انجام شده و ارائه پیشنهاد برای کارهای آینده پرداخته شده‌است.

۱. کارهای مرتبط

پژوهش‌های انجام شده برای استفاده با بهره‌وری بالاتر از منابع سیستم را می‌توان بر حسب ««الگوریتم استفاده شده برای حل مسئله» دسته‌بندی کرد که هر کدام با در نظر گرفتن پارامترهای متفاوتی به حل مسئله پرداخته‌اند. در این بخش به چند روش متداول استفاده شده در این حوزه می‌پردازیم.

۱-۱- الگوریتم تتریس

در الگوریتم تتریس(tetris) با پیدا کردن وظیفه‌ای که در آن حاصل ضرب داخلی ویژگی‌های وظایف و ویژگی‌های ماشین مجازی مقدار بیشینه داشته باشد را به‌عنوان خروجی در نظر می‌گیرد[۹]. به‌طور مثال در شکل (۱) که بیانگر این الگوریتم است با در نظر گرفتن توان پردازشی و حافظه موقت به‌عنوان ویژگی‌های وظایف و ماشین‌های مجازی عمل زمان‌بندی انجام می‌شود.

۱-۲- الگوریتم شارما و همکاران

این مقاله[۱۰] با درنظر گرفتن اهمیت کیفیت خدمات و توافقنامه سطح خدمات اقدام به مطرح کردن راه‌حل خود می‌کند. نویسندگان بیان می‌کنند که به علت تعداد وظایف بسیار زیادی که در هر لحظه وارد سیستم می‌شوند این مسئله در دسته مسائل پیچیده قرار می‌گیرد و با این حال پاسخ به این مسئله باعث افزایش بازدهی منابع و تقویت خروجی سیستم گردیده که خود منجر به افزایش رضایت کاربران خواهد شد.[۱۱]

سیستم ارائه شده توسط شارما و همکاران با در نظر گرفتن ویژگی‌های ماشین‌های مجازی و وظایف عمل می‌کند. در واقع بر اساس امتیاز داده شده به وظایف با در نظر گرفتن اولویت و زمان اجرایی وظایف و همچنین با در نظر گرفتن برخی ویژگی‌های ماشین‌های مجازی از الگوریتم K-means برای دسته‌بندی وظایف و ماشین‌های مجازی استفاده می‌کند.

همان‌طور که در شکل (۲) مشخص است در این سیستم پیشنهادی ابتدا تمام وظایفی که باید زمان‌بندی شوند و همچنین اطلاعات ماشین‌های مجازی در دسترس توسط Datacenter Broker جمع‌آوری می‌شود. در مرحله بعد پس از جمع‌آوری اطلاعات و نیازمندی‌های وظایف، مقدار پارامتر امتیاز بر اساس طول زمان پردازش و اولویت برای آن‌ها محاسبه می‌شود. سپس مانند مرحله گذشته ولی این بار برای ماشین‌های مجازی ظرفیت محاسباتی آنان بر اساس قدرت محاسباتی و میزان منابع در دسترس

```
1: function TETRIS
2:     scores = []
3:     pairs = []
4:     for machine do
5:         for task do
6:             if machine.accomodate(task) then
7:                 pairs.append([machine, task])
8:                 scores.append((machine.cpu · task.cpu) + (machine.memory · task.memory))
9:             end if
10:        end for
11:    end for
12:    index = argmax(scores)
13:    return pairs[index]
14: end function
```

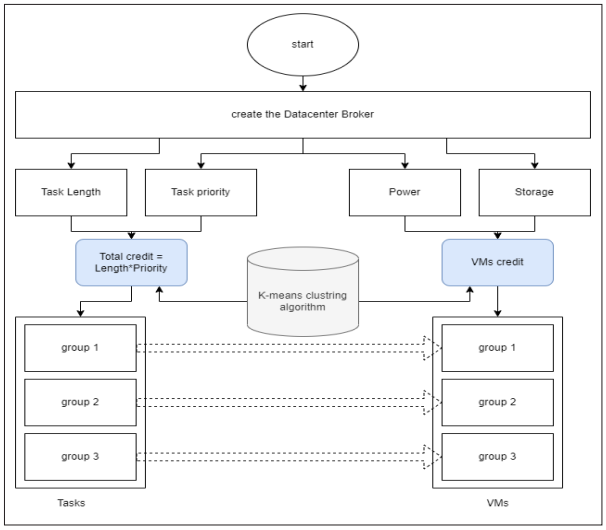
شکل ۱: الگوریتم Tetris

آن‌ها محاسبه می‌شود. برای اختصاص بهینه وظایف به منابع مناسب، دسته‌بندی وظایف و منابع به کمک الگوریتم K-means بر اساس امتیازهای محاسبه شده انجام می‌شود. دسته وظایف به‌دسته منابع طوری اختصاص یافته است که میزان خطا کاهش یافته است و نتیجه آن در بهینه سازی زمان پردازش کمک می‌کند.

۱-۳- الگوریتم دژآباد و همکاران

در این مقاله ابتدا به شرح یکی از مشکلات اساسی زمان‌بندی و قیمت‌گذاری پرداخته شده است که ارائه‌دهندگان سعی می‌کنند به صورت پویا قیمت هر نوع نمونه را تغییر دهند تا حداکثر درآمد را به‌دست آورند. بنابراین، بهتر است برای تعیین قیمت در آینده، منابع وعده داده شده و حجم کاری آینده را در نظر بگیریم و بدین‌صورت می‌توان از منابع موجود در سیستم حداکثر استفاده و درآمد را به‌دست آورد [۱۲].

بدین منظور، ابتدا برای بار کاری موجود در سیستم، پروفایلی براساس منابع استفاده شده ایجاد می‌شود و سپس تقاضاهای آینده پیش‌بینی می‌شود. برای دسته‌بندی وظایف در سیستم از دسته‌بندی سلسه مراتبی و با درنظر گرفتن CPU، حافظه و طول مدت زمان کار استفاده می‌شود که به کمک آن بار کاری موجود به سه دستهٔ تقاضای کم، متوسط و زیاد دسته‌بندی خواهند شد. سپس، این الگوریتم را بر روی داده‌های محک [۱۳] اعمال کرده و الگوی



شکل ۲: روندنما الگوریتم پیشنهادی توسط شارما و همکاران.

ورودی برای هرنوع درخواســت را به‌دست آورده است. این سیستم بینشی به ارائه‌دهندگان می‌دهد تا با استفاده از آن بتوانند تقاضاها را پیش‌بینی کنند و منابع به روشی بهتر به کاربران اختصاص دهند.

۱-٤- الگوریتم سناج و همکاران

در این مقاله ابتــدا به ارائه تکنیکی برای بهبود عملکرد الگوریتــم با گــردش نوبت برای افزایش کارایی سیســتم بدون تاثیر بر عملکرد مدل کلاسیک آن پرداخته شده و در نهایت از طریق کلادیسم نتایج حاصل از این دو الگوریتم با یکدیگر مقایسه شده است. نتایج این پژوهش نشان می‌دهد که کل زمان انتظار برای وظایف در الگوریتم جدید نســبت به مدل سنتی آن در شرایط یکسان به حداقل می‌رسد[١٤].

۱-۵- الگوریتم Vijayalakshmi و همکاران

این مقاله بیان می‌کند که الگوریتم‌های زمان‌بندی ایستا زودترین زمان پایان کار وظیفه را برای به حداقل رساندن زمان کل اجرا در نظر می‌گیرند. بنابراین، به ارائه روشــی برای تخصیص کارآمد منابع با امتیاز می‌پردازند. عملکرد کلی الگوریتم پیشنهاد شده در دو مرحله صورت می‌گیرد. در مرحله اول، کارها بر اساس رتبه صعودی اولویت‌بندی می‌شــوند و در مرحله دوم، یک امتیاز نرمال شــده برای ماشــین‌های مجازی با در نظر گرفتــن دو معیار احتمال در دســترس بودن و زودترین زمان پایان کار محاســبه می‌شــود. در نهایت کارها بر روی یک ماشــین مجازی با کمترین امتیاز در بین تمام ماشین‌های مجازی برنامه‌ریزی می‌شــوند. بنابراین، می‌توان گفت تخصیص ماشــین‌های مجازی به وظایف بر اساس امتیازی است که به هر ماشین مجازی با در نظر گرفتن معیارهای متعدد داده می‌شــود. نتایج نشــان می‌دهد که الگوریتم مطرح شده در مقایسه با الگوریتم‌هــای موجود که فقط زودترین زمان پایان کار را برای تخصیص در نظر می‌گیرند، عملکرد بهتری با قابلیت اطمینان بیشتری دارد[١٥].

۱-۶- الگوریتم Jiechao Gao و همکاران

ایــن مقاله با بیان این‌که پیش‌بینــی دقیق حجم کار در مدیریت منابع ابری بسیار مهم است به ارائه روشی برای پیش‌بینی حجم کاری در سیســتم پرداخته است. بنابراین، روش پیش‌بینی m-gap را برای انجام پیش‌بینی حجم کار در زمان معینی قبل از نقطه زمانی پیش‌بینی‌شده پیشنهاد شده است تا زمان کافی برای زمان‌بندی کار بر اساس حجم کار پیش‌بینی‌شده باقی بماند. در ادامه، یک روش پیش‌بینی بار کاری مبتنی بر خوشــه‌بندی به منظور برخورداری از دقت بالاتر در پیش‌بینی به کار گرفته شــده اســت. در این روش وظایف با الگوهای بار کاری مشــابه را خوشه‌بندی می‌شــود و بدین ترتیب یک مدل پیش‌بینی حجم کار برای هر خوشه ایجاد می‌شود و از مدل مربوطه برای پیش‌بینی حجــم کاری آتی یک کار می‌توان اســتفاده کرد. در نهایت این روش در مقایسه با روش‌های پیش‌بینی سنتی از دقت پیش‌بینی بالاتری برخوردار است[١٦].

۱-۷- الگوریتم Muthusamy و همکاران

در ایــن مقاله با بیان آن که تعادل بار فرآیندی اســت برای توزیع وظایف ورودی به منابع در دسترس در مراکز داده ابــری بیان می‌کند کــه تعادل بار مناســب منجر به، کاهش زمان محاسبه سیستم و بهبود نرخ استفاده از منابع می‌شود. بنابراین، با استفاد از خوشه‌بندی K-Means و با

در نظر گرفتن طول کار و ظرفیت ماشـــین مجازی اقدام به ارائه چارچوب زمان‌بندی کار مبتنی بر خوشه‌بندی می‌کند. وظایف بر اســـاس طـــول آن‌ها و ماشـــین‌های مجازی بر اساس ظرفیت پردازش آن‌ها خوشه‌بندی می‌شوند. پس از خوشه‌بندی، هر وظیفه در هر خوشه به یک ماشین مجازی در دســـته ماشـــین‌های مجازی اختصاص می‌یابد. نتایج حاصل از این پژوهش نشـــان می‌دهد که روش پیشنهادی نتایج بهتری از نظر زمان اجرا وظایف، Makespan نسبت به الگوریتم Min-Min معمولی و الگوریتم‌های اکتشـــافی اخیر دارد [۱۷].

۱-۸- الگوریتم وادهانکر و همکاران

در این مقاله[۱۸] با در نظر گرفتن دو شـــاخص طول و زمان پایان وظایف به این کار پرداخته شده‌است. به‌طوری که بعد از محاســـبه این مقادیر برای وظایف در سیستم با درنظر گرفتن اختلاف آن‌ها با میانگین این مقادیر اقدام به اختصاص آن‌ها به ماشین مجازی مناسب می‌کند.

۱-۹- الگوریتم شهیدی‌نژاد و همکاران

در ایـــن مقاله[۱۹] بـــا در نظر گرفتن توافقنامه ســـطح خدمات که شـــامل عوامـــل کیفیت خدمات هســـتند به حل مســـئله زمان‌بندی می‌پردازند. عواملی که در این مقاله به آن‌ها توجه شده‌اســـت در دسترس بودن، قابلیت اطمینان، محدودیـــت زمانـــی پاســـخ و محدودیت هزینه اســـت. در روش پیشـــنهادی این مقالـــه در ابتدا پس از وارد شـــدن درخواســـت‌های کاربران به سیستم، این درخواستها وارد مرحله پیش‌پردازش می‌شوند. سپس وارد مرحله دسته‌بندی بارکاری شـــده که بر اســـاس الگوریتم رقابت استعماری (Imperialist Competitive Algorithm) دسته‌بندی می‌شوند. هماننـــد دیگر الگوریتم‌هـــای تکاملی، ایـــن الگوریتم، نیز با تعدادی جمعیـــت اولیه تصادفی که هر کـــدام از آن‌ها یک «کشـــور» نامیده می‌شـــوند؛ شروع می‌شـــود. تعدادی از بهترین عناصر جمعیت (معادل نخبه‌ها در الگوریتم ژنتیک) به‌عنوان استعمارگر انتخاب می‌شوند. باقی‌مانده جمعیت نیز به‌عنوان مستعمره، در نظر گرفته می‌شوند. استعمارگران بسته به قدرتشان، این مستعمرات را با یک روند خاص به سمت خود می‌کشـــند. قدرت کل هر امپراطوری، به هر دو بخش تشـــکیل‌دهنده آن یعنی کشور استعمارگر (به‌عنوان هسته مرکزی) و مستعمرات آن، بستگی دارد. با شکل‌گیری امپراطوری‌های اولیه، رقابت استعماری میان آن‌ها شروع می‌شود. هر امپراطوری‌ای که نتواند در رقابت استعماری، موفـــق عمل کرده و بر قدرت خـــود بیفزاید (و یا حداقل از کاهش نفوذش جلوگیری کند)، از صحنه رقابت استعماری، حذف خواهد شد. بنابراین بقای یک امپراطوری، وابسته به قدرت آن در جذب مستعمرات امپراطوری‌های رقیب، و به ســـیطره درآوردن آن‌ها خواهد بود. در نتیجه، در جریان رقابت‌های استعماری، به تدریج بر قدرت امپراطوری‌های بزرگ‌تر افزوده شـــده و امپراطوری‌های ضعیف‌تر، حذف خواهند شـــد. امپراطوری‌ها بـــرای افزایش قـــدرت خود، مجبور خواهند شـــد تا مســـتعمرات خود را نیز پیشرفت دهند. در جریان رقابت‌های امپریالیســـتی، امپراطوری‌های ضعیف به تدریج ســـقوط کرده و مستعمراتشان به‌دست امپراطوری‌های قوی‌تر می‌افتد. در الگوریتم پیشنهاد شده، یک امپراطوری زمانی حذف شـــده در نظر گرفته می‌شود که مســـتعمرات خود را از دست داده باشد و این روند بین وظایف موجود در سیستم تا رسیدن به تعداد خوشه‌های مورد نظر ادامه می‌یابد. در جدول (۱) روش‌ها بر اســـاس معیارهای زیر دسته‌بندی شده‌اند:

۱. در نظـــر گرفتن ویژگی‌های منابع به‌عنوان پارامتری در الگوریتم ارائه شده.

۲. در نظر گرفتن ویژگی‌های وظایف به‌عنوان پارامتری در الگوریتم ارائه شده.

۳. دســـته‌بندی وظایف و منابع بر اســـاس پارامترهای تعریف شده در مسئله.

٤. ارائه روشـــی برای سازگار شدن سیستم با منابع و وظایف موجود در سیستم.

جدول ۱: مقایسه روش‌های ارائه شده مرتبط با مطالعه جاری

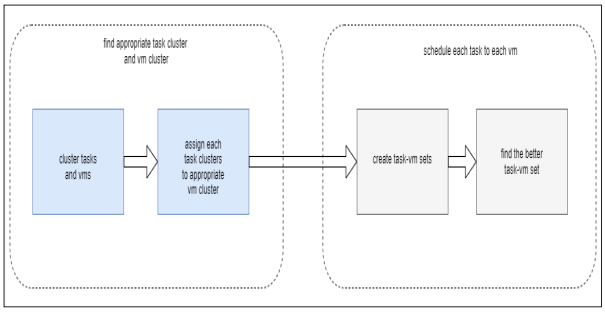
الگوریتم	ویژگی‌های منابع	ویژگی‌های وظایف	دسته‌بندی وظایف و منابع	سازگاری با اطلاعات ورودی
شارما و همکاران	✓	✓	✓	X
دژآباد و همکاران	✓	X	X	✓
سناج و همکاران	X	X	X	X
Vijayalakshmi و همکاران	✓	✓	X	X
Jiechao Gao و همکاران	X	✓	فقط وظایف	✓
Muthusamy و همکاران	✓	✓	✓	X
وادهانکر و همکاران	X	✓	X	X
شهیدی‌نژاد و همکاران	X	✓	X	X
پیشنهادی	✓	✓	✓	✓

۲- روش پیشنهادی

همان‌طور که در شکل (۳) مشخص است روش پیشنهادی از دو بخش اصلی تشکیل شده‌است که شامل خوشه‌بندی وظایف و ماشین‌های مجازی و سپس ایجاد جفت‌های ماشین_ وظیفه و سپس انتخاب بهترین جفت به‌عنوان راهبرد زمان‌بندی ارائه شده‌است. در ادامه به توضیح هر یک از بخش‌ها خواهیم پرداخت.

۲-۱- خوشه‌بندی

همان‌طور که پیش‌تر گفته شده ما قصد داریم در این پروژه ابتدا وظایف و ماشین‌های مجازی را خوشه‌بندی کنیم. بنابراین، وظایفی که به سیستم وارد شده‌اند را بر اساس پارامترهای Memory, CPU, Processing time خوشه‌بندی می‌کنیم تا با این کار وظایفی که توان پردازشی بالایی احتیاج دارند در کنار وظایفی که توان پردازشی پایینی احتیاج دارند قرار گیرند تا بتوان از گرسنگی در سیستم جلوگیری کرد. در مقابل همین روند خوشه‌بندی را برای ماشین‌های مجازی نیز تکرار می‌کنیم به‌طوری که آن‌ها را بر اساس پارامترهای Memory, CPU, Disk در خوشه‌هایی قرار می‌دهیم تا آن‌ها را از نظر نوع منبع جدا کنیم.



شکل ۳: معماری کلی روش پیشنهادی

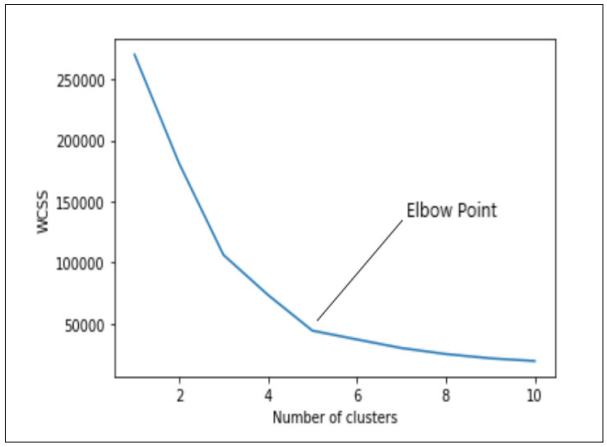
۲-۲- محاسبه تعداد خوشه‌ها

محاسبه تعداد خوشه‌ها در سیستم به صورت پویا انجام می‌گیرد. بنابراین، ابتدا تعداد خوشه‌های مورد نیاز برای دسته‌بندی وظایف را استخراج و سپس تعداد خوشه‌های مورد نیاز برای دسته‌بندی ماشین‌های مجازی را نیز محاسبه می‌کنیم. این محاسبه توسط الگوریتم Elbow Method انجام می‌گیرد[۲۰]. بدین‌صورت که مجموع فواصل درون خوشه‌ای داده‌ها را به‌عنوان تابعی از تعداد خوشه‌ها در نظر گرفته می‌شود. به این ترتیب تعداد خوشه‌ها به نحوی انتخاب می‌شوند که افزودن یک خوشه دیگر، بهبودی در سیستم ایجاد نکند. پس از آن برای آن که باید تعداد مساوی خوشه از وظایف و ماشین‌های مجازی داشت به علت آن که بتوان خوشه‌های ایجاد شده از آن‌ها را با یکدیگر جفت کرد بیشترین مقدار از بین تعداد خوشه‌های ایجاد شده برای هرکدام را به‌عنوان تعداد خوشه نهایی انتخاب می‌کنیم. همان‌طور که در شکل (٤) مشخص است ٥ خوشه برای این سیستم بهترین تعداد خوشه بوده زیرا بعد از آن مدل بهتری به‌دست نمی‌آید.

پس از خوشه‌بندی وظایف و ماشین‌های مجازی باید خوشه‌های ایجاد شده را از وظایف به خوشه مناسب از ماشین‌های مجازی اختصاص داد. برای این کار مقدار هزینه زیر را برای هر خوشه از وظایف محاسبه می‌کنیم.

$$cost = \alpha(x') + \beta(y') + \gamma(z') \quad (1)$$

در رابطه (۱)مقدار α بیانگر ضریب اثرگذاری مقدار پارامتر x' و مقدار β بیانگر ضریب اثرگذاری مقدار پارامتر y' است و همچنین مقدار γ بیانگر ضریب



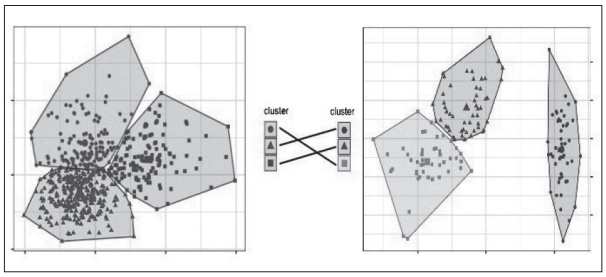
شکل ۴: مثالی از elbow method [۲۱]

اثرگذاری مقدار پارامتر z' است. این مقادیر مقدار میانگین Memory, CPU, Processing Time خوشه است که با استفاده از α و β و γ می‌توان میزان تاثیر این پارامترها را در محاسبه هزینه آن دسته تغییر داد. با محاسبه مقدار هزینه برای هر دسته از وظایف و ماشین‌های مجازی به مقادیری خواهیم رسید که ابتدا باید آن‌ها را به ترتیب صعودی مرتب می‌کنیم. سپس به ترتیب صعودی آن‌ها را به هم نگاشت می‌دهیم. به این صورت که خوشه وظایف با بیشترین مقدار هزینه به خوشه ماشین‌های مجازی با بیشترین مقدار هزینه نگاشت می‌شود. شکل (۵) بیانگر فرآیند نگاشت خوشه‌های وظایف و منابع است. در اصل خوشه‌بندی به علت در نظر گرفتن سه پارامتر در وظایف و منابع در محیط سه‌بعدی انجام می‌شود ولی به منظور رعایت سادگی شکل به صورت دوبعدی تصویر شده است.

پس از طی کردن نگاشت خوشه‌های ایجاد شده به یکدیگر باید از الگوریتمی برای اختصاص دادن هر وظیفه به یک ماشین استفاده کنیم.

۲-۳- نگاشت وظایف به منابع

در این قسمت قصد داریم که از روش یادگیری تقویتی عمیق برای این منظور استفاده کنیم. همان‌طور که پیش‌تر گفته شد در الگوریتم‌های یادگیری تقویتی در لحظه t عامل در محیط اقدامی انجام می‌دهد که این اقدام باعث انتقال او از حالت S_t به S_{t+1} می‌شود. همچنین با توجه به عملکردش به او پاداش r_t داده می‌شود. در الگوریتم‌های یادگیری تقویتی



شکل ۵: نگاشت خوشه‌بندی وظایف و منابع

عمیق محاسبات مرکزی این الگوریتم توسط یک شبکه عصبی مدیریت می‌شود. در یادگیری تقویتی خط مشی (policy gradient)مورد استفاده برای تصمیم‌گیری تحت عنوان توزیع احتمال $\pi(a|s)$ تعریف می‌شود. در بسیاری از مسائل عملی ممکن است تعداد جفت‌های (حالت, عمل) بسیار زیاد باشند و امکان ذخیره‌سازی توزیع احتمال به صورت جدولی برای تک تک جفت‌ها وجود نداشته باشد. برای حل کردن این مشکل از تخمین استفاده می‌شود. تابع تخمین تعداد مشخصی پارامتر θ که قابل تغییر هستند دارد. بنابراین، تابع احتمال به صورت $\pi_\theta(a|s)$ نمایش داده می‌شود. تخمین‌زننده‌ها در اشکال مختلفی وجود دارند. اخیراً شبکه‌های عصبی عمیق در قالب تخمین‌زننده در بسیاری از مسائل یادگیری تقویتی با مقیاس بزرگ موفق عمل کرده‌اند و از آن‌ها به‌عنوان تخمین‌زننده در روش پیشنهادی استفاده می‌شود. همان‌طور که گفته شد ما در اینجا در قسمت شبکه عصبی از روش شیب خط مشی استفاده می‌کنیم. که به مدل کردن و بهینه کردن سیاست استفاده شده در سیستم کمک می‌کند. در نتیجه، پاداش تجمعی مورد انتظار از رابطه ۲ به‌دست می‌آید.

$$J(\theta) = E_{\tau \sim p_\theta(\tau)}[r(\tau)] \qquad (۲)$$

که در آن:

$$p_\theta(\tau) = p(s_1, a_1, \dots, s_T, a_T)$$
$$= p(s_1)\pi_\theta(a_1|s_1)\prod_{t=2}^{T} p(s_t|s_{t-1}, a_{t-1})\pi_\theta) \qquad (۳)$$

$$r(\tau) = r(s_1, a_1, \dots, s_T, a_T)$$
$$= \sum_{t=1}^{T} r_t = \sum_{t=1}^{T} r(s_t, a_t) \qquad (۴)$$

$$r(\tau) = r(s_1, a_1, \dots, s_T, a_T) = \sum_{t=1}^{T} r_t = \sum_{t=1}^{T} r(s_t, a_t) \quad (۵)$$

و برای اعمال گرادیان کاهشی گرادیان $J(\theta)$ را به شکل زیر محاسبه می‌کنیم:

$$\nabla_\theta J(\theta) = \nabla_\theta \int p_\theta(\tau) r(\tau) d(\tau) = \int p_\theta(\tau) \nabla_\theta \log p_\theta(\tau) r(\tau) d\tau \quad (۶)$$

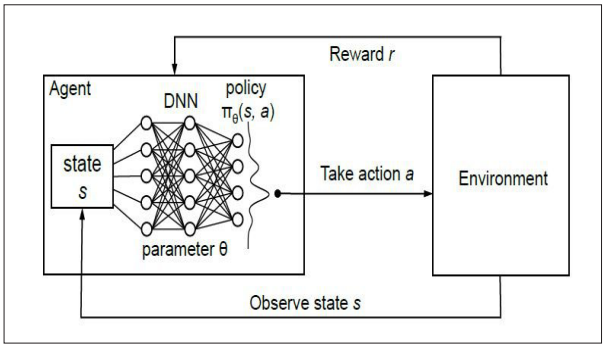
برای این کار ابتدا لازم است وظایفی که ویژگی‌های مورد نیاز منابعی که خوشه‌اش به آن اختصاص داده‌شده دارد را بیابیم. برای این کار به مجموعه‌ای از ماشین-وظیفه احتیاج داریم. هر وظیفه دارای مشخصاتی است که پیش‌تر گفته شد. در نتیجه، زمان پردازش T_i و بردار ویژگیهای وظیفه i-ام به صورت $r_i = r_{i,1} + \cdots + r_{i,d}$ و همچنین بردار ویژگی‌های ماشین i-ام به صورت $a_j = a_{j,1} + \cdots + a_{j,d}$ تعریف میشود. وظیفه i-ام تنها در صورتی که شرط زیر برقرار باشد می‌تواند روی ماشین j-ام اجرا شود:

$$a_{j,k} \geq r_{i,k} \; \forall k \in [1, d] \quad (۷)$$

بنابراین (i, j) جفت ماشین _ وظیفه مناسب خواهد بود. عامل در طول کار خود به دنبال پیدا کردن جفت ماشین_ وظیفه‌ای است که مقدار بزرگ‌ترین برازش را از بین تمامی جفتهای موجود دارد.

۲-٤- یادگیری تقویتی عمیق

یادگیری تقویتی زیرمجموعه‌ای از یادگیری ماشین است که در آن به یک عامل این امکان را می‌دهد تا از تعامل با محیط و از طریق آزمون و خطا یاد بگیرید. عامل، بازخوردهایی از محیط می‌گیرد و تجربه‌هایی از محیط کسب می‌کند که همه به یادگیری‌اش کمک می‌کنند[۲۲]. در یادگیری تقویتی عمیق، این روش یادگیری با یادگیری عمیق ترکیب می‌شوند. در یادگیری تقویتی عمیق ورودی‌ها می‌توانند بسیار بزرگ باشند به‌طوری که با الگوریتم یادگیری تقویتی قابل حل نیستند.[۲۳] به همین علت در این



شکل ۶: یادگیری تقویتی عمیق[۲۵]

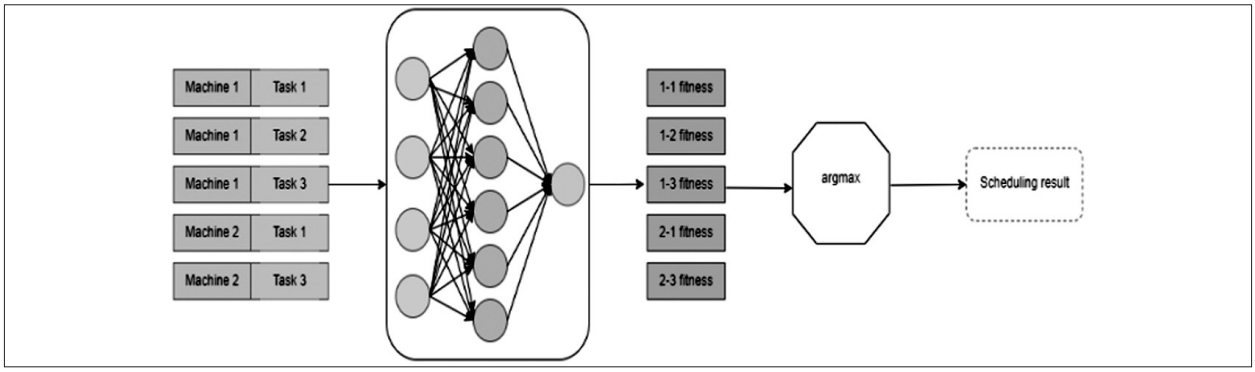
روش از شبکه‌های عصبی برای نگاشت حالت‌ها به مقادیر استفاده می‌شود. چراکه استفاده از جدول برای ذخیره، فهرست‌بندی و به‌روزرسانی همهٔ حالت‌های ممکن و مقادیر آن‌ها مشکل است. بنابراین، می‌توانیم یک شبکه عصبی را بر روی نمونه‌هایی از حالت یا فضای عمل آموزش دهیم تا پیش‌بینی کنند چقدر آن‌ها ارزشمند هستند.[۲٤] معماری کلی این شبکه‌ها در شکل (٦) مشخص است.

همان‌طور که در شکل (۷) مشخص است پس از یافتن جفت ماشین-وظیفه مناسب برای پیدا کردن بهترین جفت از شبکه عصبی کمک می‌گیریم که خروجی آن یک عدد است که میزان برازش آن بوده و بزرگ‌ترین آن‌ها به‌عنوان نتیجه الگوریتم استفاده می‌شود.

فضای حالت: فضای حالت مسئله اشاره به لیست تمامی جفت ماشین-وظیفه ها دارد که در طول زمانبندی مقدار آن کم و زیاد می‌شود.

فضای عمل: در یک لحظه خاص می‌توان گفت N وظیفه و M ماشین وجود دارد که عامل ممکن است هرکدام از وظایف را به هرکدام از ماشین‌ها بدهد. به‌طور کلی انتخاب کردن جفت ماشین _ وظیفه از بین لیست آن‌ها و اختصاص آن‌ها به هم کاری است که در سیستم انجام می‌شود. بنابراین، در هر لحظه تصمیمات زمان‌بندی صورت می‌گیرد تا لیست ماشین _ وظیفه خالی شود. پس از آن عامل متوقف شده و در مرحله بعدی زمان‌بندی مجدد فعال می‌شود. در الگوریتم زیر فرآیند کلی این راه‌حل ارائه شده مطرح شده‌است.

پاداش: به منظور این‌که عامل به‌طور موثر یاد بگیرد و



شکل ۱: مدل شبکه به کار برده شده

```
while True do
    pairList ← getPairList()
    while pairList ≠ ∅ do
        pair ← selectBestPair(pairList)
        machine ← pair.machine
        task ← pair.task
        schedule task to machine
        pairList ← getPairList()
    end while
    sleep for one timeout
end while
```

شکل ۲: فرآیند زمان‌بندی عامل

عملکرد سیستم را بهتر کند از پاداش به‌عنوان عامل هدایت فرآیند یادگیری اســتفاده می‌کنیم. ما در اینجا برای کاهش makespan در سیستم، به اندازه ۱- به‌عنوان پاداش در هر مرحلــه زمانی تا زمانی که همه کارها تکمیل گردد انتخاب می‌کنیم. بنابراین، عامل تلاش می‌کند تا پاداش را به حداکثر برســاند و با این کار به کاهــش makespan وظایف کمک می‌کند. بنابراین به ازای وظایفی که تموم نشــده باشند در انتهای عمل، عامل پاداش خود را می‌گیرد که از رابطه زیر تعریف می‌شود.

$$reward = -(\sum \frac{1}{T_i}) \quad (۸)$$

به صورتی که T_i زمان پردازش تسک‌ها است.

عامل: در این مقاله از یک شــبکه عصبی کاملاً متصل به‌عنوان تصمیم‌گیرنده عامل استفاده می‌کنیم. همان‌طور که گفتیم حالت محیط با یک لیست جفت ماشین_وظیفه نشان داده می‌شود و همین لیست به‌عنوان ورودی شبکه عصبی قرار دارد.

مدل یادگیری برازندگی: در مراحل مختلفی شــبکه عصبی ایجاد شــده را آموزش می‌دهیم. در هر مرحله تعداد مشخصی وظیفه وجود دارند که عامل باید آن‌ها را زمان‌بندی کند. زمانی که تمام وظایف موجود زمابندی شــدند یک مرحله به اتمام می‌رســد. پس از آن توسط عامــل وضعیت حالت سیســتم، عمل و پاداش توســط عامل ارزیابی می‌شود. در شکل (۹) الگوریتم این مرحله وجود دارد که L بیانگر تعداد تصمیماتی اســت که عامل گرفته است.

۳- ارزیابی روش پیشنهادی

ارزیابی روش پیشنهادی به شرح زیر است:

۳-۱- محیط آزمایش

برای ارزیابی و پیاده‌ســازی روش ارائه شــده در این پژوهــش از زبــان برنامه‌نویســی پایتون تحت سیســتم عامل لینوکس استفاده شده‌است. همچنین از simpy برای شبیه‌ســازی محیط اســتفاده گردیده اســت. سخت‌افزار استفاده شده عبارت است از:

۱. پردازنده core i7.
۲. ۱۶ گیگ حافظه اصلی.
۳. ۳ ترابایت دیسک.

۳-۲- معیارهای ارزیابی

برای بررســی عملکرد و میزان کارایی الگوریتم ارائه

```
α ← 0001
job_sequence ← generate_job_sequence()
for eachiteration do
    for i ← 1, N do
        τ_i ← [s_1^i, a_1^i, r_1^i, ··· , s_{L_i}^i, a_{L_i}^i, r_{L_i}^i]
        for t ← 1, L_i do
            q_t^i ← Σ_t^{L_i} r_t^i
        end for
    end for
    L ← max(L_1, ··· , L_N)
    for t ← 1, L do
        b_t ← 1/N Σ_{i=1}^N q_t^i
    end for
    for i ← 1, N do
        Δθ ← 0
        for t ← 1, L_i do
            advantage_t^i ← q_t^i − b_t
            Δθ ← Δθ + ∇_θ log π_θ(a_t^i|s_t^i) advantage_t^i
        end for
        θ ← θ + αΔθ
    end for
end for
```

شکل ۳: الگوریتم آموزش

شـده لازم اسـت تا معیارهایی برای ارزیابی آن داشـته باشـیم که این معیارها هم در روش پیشـنهادی و هم در روش‌هـای دیگری که قرار اسـت با عملکـرد الگوریتم ما مقایسه شـوند مورد استفاده هسـتند. ابتدا بهتر است به تعریف این معیارها بپردازیم.

معیار پاداش در سیستم: در هر اپیزود به ازای منفی حاصـل جمع مقدار زمـان وظایف باقیمانده در سیسـتم عددی به‌عنوان پاداش به سیستم داده می‌شود.

معیار ضرر در سیستم: این متغیر میزان خطا در هر بار اجرای شبکه عصبی را به نشان می‌دهد. برای محاسبه آن در سیسـتم از sparse categorical cross entropy استفاده شده که با محاسبه شباهت خروجی شبکه عصبی با نتایج به‌دست آمده از شبیه‌ساز است.

معیار زمـان تکمیل کار: در سیسـتم متغیری به نام makespan وجود دارد که به تفاوت زمانی بین شـروع و پایان یک دنباله از کارها یا وظایف می‌گویند. بنابراین، برای انجام به موقع وظایف لازم است این متغیر کاهش پیدا کند.

معیار میانگین زمان اتمام وظایف: در سیستم متغیری بـه نـام Average Completion Time تعریف کرده‌ایم که میانگیـن زمان اتمـام وظایف را در سیسـتم به ما اطلاع می‌دهـد. به این صورت که اختلاف زمان شـروع و زمان پایان وظیفـه را اندازه‌گیری می‌کنیم و آن‌ها را با هم جمع می‌کنیم و بر تعـداد وظایف موجود و اندازه‌گیری شـده تقسیم می‌کنیم. در نتیجه، کاهش این زمان به بهبود عملکرد سیستم کمک می‌کند. همچنین می‌توان با تاثیر معیار طول مدت اجـرای آن وظیفه معیـار جدیدی به نـام Average Slowdown تعریف کرد که باز هم کاهش آن نشـان‌دهنده عملکرد بهتر سیستم است.

۳-۳- تعداد خوشه‌های مناسب

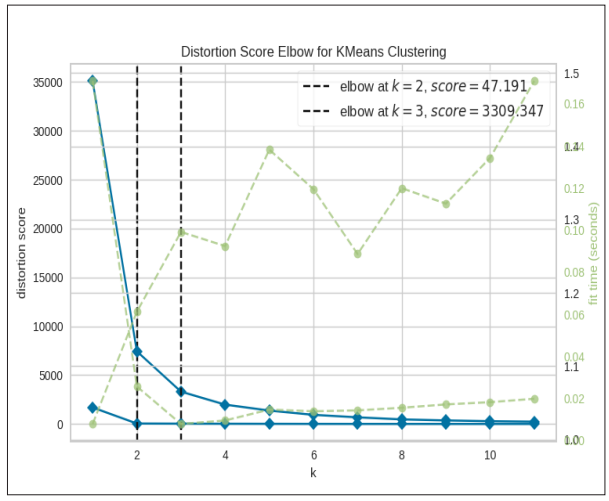
از آن جایـی کـه باید تعـداد خوشـه‌های وظایف و ماشـین‌های مجازی یکسان باشـد به منظور جفت کردن آن‌ها با یکدیگر حداکثر تعداد خوشه‌های به‌دست آمده برای هرکدام را به‌عنوان تعداد نهایی انتخاب می‌کنیم. وظایف و ماشـین‌ها در این سیستم طبق روشی که در بخش گذشته گفته شد به ۳ خوشه تقسیم می‌شوند.

۳-۴- نتایج آموزش مدل

همان‌طور که در ابتدای بخش اشاره شد برای بررسی عملکرد سیسـتم معیارهایی را معرفی کردیم که در ادامه به بررسـی ایـن معیارها در سیسـتم می‌پردازیم. پس از خوشه‌بندی وظایف و اختصاص خوشه‌های ایجاد شده به یکدیگر هر کدام از خوشه‌ها به صورت موازی در سیستم اجرا خواهند شد. در این بخش به بررسی اطلاعات حاصل از آموزش مدل می‌پردازیم. تمام نتایج موجود پس از ۵۰۰ تکرار است.

پاداش: همان‌طور که بـالا در توضیحات این پارامتر گفته شـد به ازای مجموع زمان‌های باقیمانده عددی منفی در سیستم لحاظ می‌شود بنابراین مدل در هر لحظه تلاش می‌کند پاداش بیشتری بگیرد که باتوجه به شکل (۱۱) مقدار آن حالت صعودی داشته است.

میانگین زمان اتمام وظایف: همان‌طور که در شـکل



شکل ۴: تعداد خوشه‌های مورد نیاز

(۱۲) مشخص است پس از ۵۰۰ تکرار مقدار میانگین زمان اتمام وظایف به‌طور کلی کاهش داشته است.

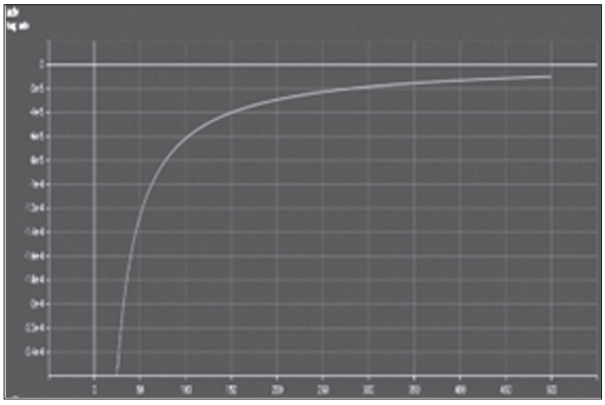
و به‌طور مشـــابه مقدار Average slowdown به شکل کلی روند نزولی داشته است.

ضرر: باتوجه به شکل (۱۳) با گذر زمان نتایج پیش‌بینی شبکه عصبی با نتیجه شبیه‌ساز نزدیک‌تر می‌شود به‌طوری کـــه در مرحله آخر یادگیری، ضـــرر ۶۱,۰۶ درصد کاهش یافته است.

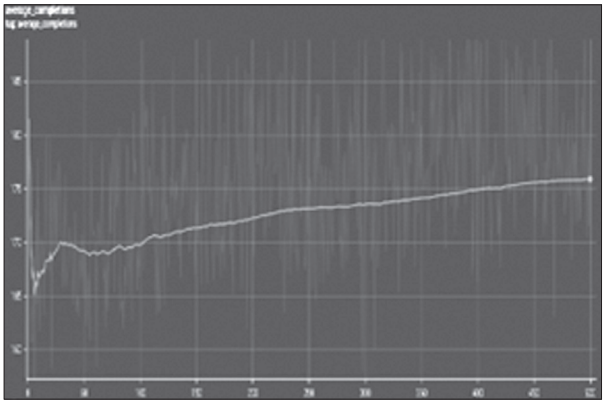
Makespan: همان‌طور که گفته شد برای عملکرد بهتر سیستم لازم است این متغیر در طول اجرا کاهش یابد. در شـــکل (۱۵) روند کلی این متغیر مشخص است که در طول بـــازه ۵۰۰ تکرار با وجود نوســـان در طول بازه مقدار این متغیر به میزان قابل توجهی کاهش یافته است.

۴- نتیجه‌گیری و کارهای آینده

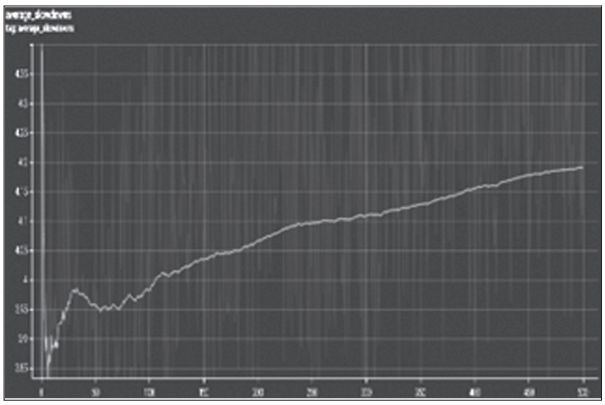
در سیســـتم ارائه شـــده برخلاف برخی راهکارها در گذشـــته با در نظر گرفتن ویژگی‌های ماشین‌های مجازی و همچنیـــن وظایف به حل مســـئله زمان‌بندی پرداختیم تا بتوانیم از جنبه‌های مختلف به آن‌ها در مســـئله نگاه کنیم و بتوانیم راه‌حل موثرتری ارائه دهیم. سپس برای استفاده بهینه از منابع در دســـترس، وظایف و ماشین‌های مجازی را با استفاده از این ویژگی‌ها دسته‌بندی می‌کنیم و سپس با کمک یک شبکه عصبی به آموزش مدلی به منظور زمانبندی



شکل ۵: تغییرات پاداش در بازه ۵۰۰ تکرار

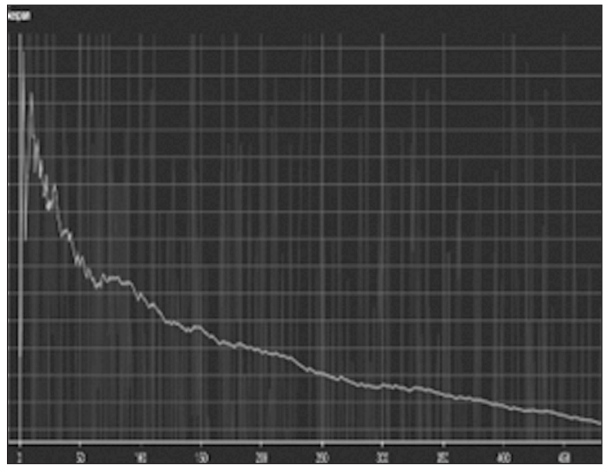


شکل ۶: تغییرات میانگین زمان اتمام وظایف در بازه ۵۰۰ تکرار



شکل ۸: تغییرات ضرر در بازه ۵۰۰ تکرار

وظایف پرداختیم و راه‌حل ارائه شده توانست نتایجی مثبتی را نشـــان دهد. برای بررسی دقیق‌تر خروجی‌های حاصل شده به بررسی پارامتر میانگین زمان اتمام وظایف که در بخش گذشته آن را تعریف کردیم می‌پردازیم. همان‌طور که گفته شد هر چه این پارامتر مقدار کمتری داشته باشد نشان دهنده عملکرد بهتر آن الگوریتم اســـت بنابراین همان‌طور که در شـــکل (۱۶) مشخص اســـت اعداد به‌دست آمده در خروجی سیســـتم حاکی از عملکرد بهتر الگوریتم پیشنهاد



شکل ۹: تغییرات makespan در بازه ۵۰۰ تکرار

شـــده نســـبت به الگوریتم‌های تصادفی، اولیـــن برازش و Tetris است. در محور عمودی این نمودار میزان این متغیر در هر ٤ حالت بیان شـــده است که الگوریتم پیشنهادی از بقیه کمتر می‌باشد که نشان دهنده عملکرد بهتر است.

طبق بررســـی عملکرد سیستم طراحی شده در بعضی موارد ممکن اســـت برخی خوشـــه‌ها وظیفه‌ای برای انجام دادن نداشـــته باشند. پس بهتر اســـت برای عملکرد بهتر، خوشـــه‌بندی به صورت پویا انجام شـــود تا بارکاری بین آن‌ها تقســـیم شـــود. با وجود این‌که احتمال دارد این کار هزینه اضافه برای سیســـتم داشـــته باشـــد ولی جابجایی وظایف بین خوشـــه‌ها در برخی مـــوارد به خروجی بهتر می‌انجامد

مراجع

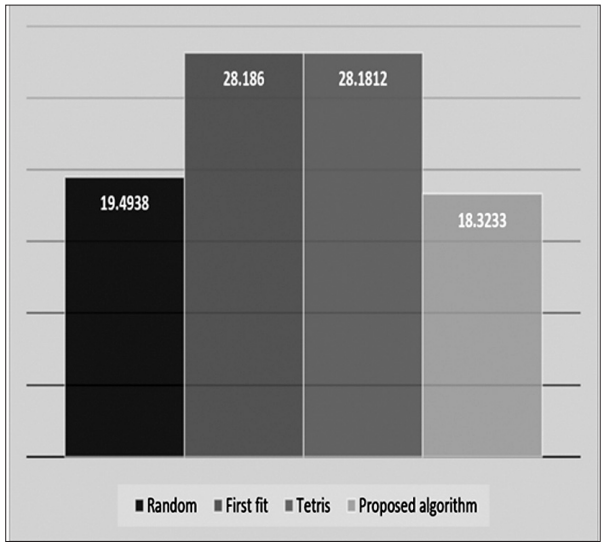
[1] J. Li, J. Li, D. Xie, and Z. Cai, "Secure auditing and deduplicating data in cloud," IEEE Transactions on Computers, vol. 65, no. 8, pp. 2386–2396, 2016.

[2] M. N. Sadiku, S. M. Musa, and O. D. Momoh, "Cloud computing: Opportunities and challenges," IEEE Potentials, vol. 33, no. 1, pp. 34–36, 2014.

[3] L. Liu and Z. Qiu, "A survey on virtual machine scheduling in cloud computing," in Proceedings of the 2016 2nd IEEE International Conference on Computer and Communications (ICCC), 2016, 14-17 Oct, Chengdu, China, pp. 2717–2721.

[4] N. Almezeini and A. Hafez, "Review on scheduling in cloud computing," IJCSNS International Journal of Computer Science and Network Security, vol. 12. no.8, 2018.

[5] E. Pacini, C. Mateos, and C. G. Garino, "Multi-objective swarm intelligence schedulers for online scientific clouds," Computing, vol. 98, no. 5, pp. 495–522, 2016.



شکل ۱۰: مقایسه الگوریتم پیشنهادی و سه الگوریتم ابتدایی دیگر

[6] R. Annette, A. Banu, and S. Raghunathan, "A taxonomy and survey of scheduling algorithms in cloud: Based on task dependency," International Journal of Computer Applications, vol. 82, no.12, pp. 20–26, 11 2013.

[7] P. V. P. Vivek Manglani, Abhilasha Jain, "Task scheduling in cloud computing," International Journal of Advanced Research in Computer Science, vol.15, no. 45, 2017.

[8] D. P. Chandrashekar, "Robust and fault-tolerant scheduling for scientific workflows in cloud computing environments," Submitted in total fulfilment of the requirements of the degree of Doctor of Philosophy, Department of Computing and Information Systems, The University Of Melbourne, 2015.

[9] R. Grandl, G. Ananthanarayanan, S. Kandula, S. Rao, and A. Akella, "Multi-resource packing for cluster schedulers," SIGCOMM Comput. Commun. Rev., vol. 44, no. 4, p. 455–466, Aug 2014. [Online].

[10] V. Sharmaa and M. Balab, "A scheduling strategy for cloud computing with improved processing time," International Journal of Grid and Distributed Computing, vol. 12, no. 2, pp. 54–62, 2019.

[11] M. Bourguiba, K. Haddadou, I. E. Korbi, and G. Pujolle, "Improving network i/o virtualization for cloud computing," IEEE Transactions on Parallel and Distributed Systems, vol. 25, no. 3, pp. 673–681, 2014.

[12] N. Dezhabad, S. Ganti and G. Shoja, "Cloud Workload Characterization and Profiling for Resource Allocation," in Proceedings of the 2019 IEEE 8th International Conference on Cloud Networking (CloudNet), Coimbra, Portugal, 4-6 Nov, 2019, pp. 1-4.

[13] Cluster data collected from production clusters in Alibaba for cluster management research: Available at: https://github.com/alibaba/clusterdata.

[14] M. S. Sanaj and P. M. Joe Prathap, "An Enhanced Round Robin (ERR) algorithm for Effective and Efficient Task Scheduling in cloud environment," in Proceedings of the 2020 Advanced Computing and Communication Technologies for High Performance Applications (ACCTHPA), Kerala, India,

[20] Scikit. (nd) Elbow Method. [Accessed: 22 9 2022]. [Online]. Available: https://www.scikit-yb.org/en/latest/api/cluster/elbow.html
[21] B. Saji. (2022) In-depth intuition of k-means clustering algorithm in machine learning. [Accessed:22 9 2022]. [Online]. Available: https://www.analyticsvidhya.com/blog/2021/01/ in-depth-intuition-of-k-means-clustering-algorithm-in-machine-learning/
[22] Y. Li, "Deep reinforcement learning: An overview," arXiv preprint arXiv:1701.07274, 2017.
[23] V. François-Lavet, P. Henderson, R. Islam, M. G. Bellemare, J. Pineau et al., "An introduction to deep reinforcement learning," Foundations and Trends in Machine Learning, vol. 11, no. 3-4, pp. 219–354, 2018.
[24] W. Chen, Y. Xu, and X. Wu, "Deep reinforcement learning for multi-resource multi-machine job scheduling," arXiv preprint arXiv:1711.07440, 2017.
[25] Deep reinforcement learning: Value functions, DQN, actor-critic method, back-propagation through stochastic functions. [Accessed:25 9 2022]. [Online].
Available: https://medium.com/@vishnuvijayanpv/ deep-reinforcement-learning-value-functions-dqn-actor-critic-method-backpropagation-through-83a277d8c38d.
2-4 July, 2020, pp. 107-110.
[15] V. A. Lepakshi and C. S. R. Prashanth, "Efficient Resource Allocation with Score for Reliable Task Scheduling in Cloud Computing Systems," in Proceedings of the 2020 2nd International Conference on Innovative Mechanisms for Industry Applications (ICIMIA), Sagar, India, 5-7 March, 2020, pp. 6-12.
[16] J. Gao, H. Wang and H. Shen, "Machine Learning Based Workload Prediction in Cloud Computing," in Proceedings of the 29th International Conference on Computer Communications and Networks (ICCCN), 2020, 3-6 August, Hawaii, USA, pp. 1-9.
[17] Muthusamy, G. & Chandran, S.R. (2021). Cluster-based task scheduling using K-means clustering for load balancing in cloud datacenters. Journal of Internet Technology. 22. 121-130.
[18] A. Wadhonkar and D. Theng, "A task scheduling algorithm based on task length and deadline in cloud computing," International Journal of Scientific & Engineering Research, vol.12, no.34, 2016.
[19] M. Shahidinejad, Ghobaei-Arani, "Resource provisioning using workload clustering in cloud computing environment: a hybrid approach," Cluster Computing, vol.24, pp.319-342, Springer, 2020.