7th International Conference on Web Research
May 19-20, 2021; Tehran, IRAN ICWR2021

A Self-adaptive Resource Management Approach for Cloud Computing Environments

Ahmadreza Sheibanirad[1], Mehrdad Ashtiani[2]*

[1] Master of Computer Engineering, Iran University of Science and Technology, Tehran, Iran
ahmadrezasheibanirad@gmail.com

[2] Assistant Professor, School of Computer Engineering, Iran University of Science and Technology, Tehran, Iran
m_ashtiani@iust.ac.ir

Abstract

The Cloud computing environment is a new computing resource that its users have increased compared to the past due to convenience and quick services. The increased demand for cloud services has confronted the cloud server with the challenges like scheduling data center resources to balance slowdown, reduce makespan, and gives an equal chance of selecting jobs(requests). Job scheduling is a crucial unit in data centers. The hand-crafted heuristic can not automatically adapt to the environment and optimize for a specific workload. As well, the allocation of multi-dimensional resources over time and space. To confront the mentioned challenges, we applied reinforcement learning as a sequential decision-making method that changes its behavior to deal with environmental changes. The proposed scheduler is evaluated through a series of simulation scenarios and real data from Google using the metrics of average response slowdown, the balance point of mean slowdown, and resource utilization of data center. Results show that the proposed scheduler has a significant improvement compared to the automated learning methods Monte Carlo and the actor-critic.

Keywords: Cloud Computing, Datacenter, Job Scheduling, Reinforcement Learning, Soft Actor-Critic.

7th International Conference on Web Research
May 19-20, 2021; Tehran, IRAN ICWR2021

یک رهیافت خودتطبیق‌پذیر مدیریت منابع در محیط‌های رایانش ابری

احمدرضا شیبانی‌راد[۱]، مهرداد آشتیانی[۲*]

[۱] کارشناسی ارشد مهندسی نرم افزار، دانشگاه علم و صنعت ایران، تهران،
ahmadrezasheibanirad@gmail.com

[۲] استادیار، دانشکده مهندسی کامپیوتر، دانشگاه علم و صنعت ایران ، تهران،
m_ashtiani@iust.ac.ir

چکیده

افزایش تقاضا، سرویس دهنده ابری را با چالش زمان‌بندی مناسب منابع مرکز داده برای حفظ تعادل زمان پاسخگویی به کار ها، کاهش این زمان و داشتن شانس تقریبی برابر انتخاب هر یک از کارها روبه رو ساخته است. زمان بندی کارها، یک واحد اصلی در مرکز داده محیط رایانش ابری است. الگوریتم های اکتشافی توانمندی تطبیق پذیری خودکار نسبت به بهینگی و تغییرات محیطی یک بارکاری از پیش تعیین شده را ندارند. از طرف دیگر، فرآیند تخصیص چندگانه منابع در واحد زمان و مکان منجر به ایجاد پیچیدگی بیشتر این فرآیند در محیط رایانش ابری می شود. در جهت توجه و مواجهه با چالش‌های مطرح شده از یادگیری تقویتی به عنوان یک روش تصمیم گیری ترتیبی با امکان تغییر رفتار در مقابل تغییر محیطی استفاده شده است.

رهیافت پیشنهادی، با کارهای پژوهشی DeepRM و DeepScheduler در حوزه زمان‌بندی خودکار و مبتنی بر یادگیری تقویتی با دادگان شبیه سازی از منظر معیار کندی میانگین پاسخ، نقطه تعادلی میانگین پاسخگویی به کارها و کمینه‌سازی بهره وری منابع مرکز داده در الگوی تقاضای نرمال کارها مورد ارزیابی قرار گرفت و مولفه زمان‌بند رهیافت پیشنهادی، عملکرد بهتری نسبت به آن‌ها در مدیریت منابع مرکزداده از خود نشان داد.

کلمات کلیدی

محیط رایانش ابری، مرکز داده، زمان‌بندی و تخصیص کار، یادگیری تقویتی، عملگر-منتقد نرم

۱- مقدمه

یکی از چالش‌های محیط رایانش ابری، مواجهه با بارهای کاری دسته‌ای غیرقابل پیش‌بینی از سمت کاربران(عدم قطعیت در منابع درخواستی کارها و تنوع آن‌ها) به مرکزداده است. به عنوان مثال در وب‌سایت Weibo پس از این‌که یک فرد مشهور چینی رابطه جدید خود را اعلام کرد، با هجوم هواداران این فرد معروف به وب‌سایت مواجهه شد. بنابراین، مرکزداده محیط رایانش ابری باید به گونه‌ای طراحی شوند که برای رسیدگی به رخدادهای غیرمنتظره مثل انفجار تقاضا کاربران، خودتطبیق پذیر باشند. مراکزداده شکل‌دهنده فن‌آوری امروز هستند و اکثر داده‌های موجود در اینترنت را می‌توان در آن‌ها جستوجو نمود. با ظهور اینترنت اشیا تعداد کاربران متصل به اینترنت و خدمت‌دهندگان در مراکزداده افزایش یافته است. همچنین نیاز به پردازش در نزد خدمت‌دهنده برای خدمات مختلف بیش از پیش مردم عادی را به سوی محیط رایانش ابری برده است. بدین ترتیب مدیریت کارآمد منابع مرکزداده که امکان اجرای موازی، سازگار با تغییر و تنوع کارهای دسته‌ای دریافتی را داشته باشد از اهمیت بالایی برخوردار است. در رهیافت مدیریت منابع، زمان‌بندی برای برنامه‌ریزی و نظم‌بخشی به اجرای برنامه‌های کاربردی مختلف نیاز است تا استفاده بهینه از منابع و کارایی زمان اجرا میسر شود. به خاطر وجود تعداد

زیادی از برنامه‌های کاربردی که باید همزمان روی محیط رایانش ابری قرار گیرند، تخصیص آن‌ها به صورت دستی به یک ماموریت تقریبا غیرممکن تبدیل شده است. به منظور توجه به این چالش از فرآیند خودکارسازی برای به حداقل رساندن تلاش‌های دستی مدیریت و زمان‌بندی بارکاری محیط رایانش ابری استفاده شده است. این دسته از فرآیندها، روش‌های طراحی خودکار و ابزارهایی(مثل هوش مصنوعی) را شامل می‌شود که بر روی محیط رایانش ابر مجازی قرار دارند تا تصمیمات بی‌درنگ تخصیص و مدیریت منابع را اتخاذ کنند[1,2].

رهیافت‌های زیادی تاکنون برای مدیریت منابع محیط رایانش ابری با توجه به مدل های تجاری موجود این حوزه ارائه شده است اما در حوزه یادگیری تقویتی کارهای تحقیقاتی کمتری صورت گرفته است. در سال‌های اخیر، یادگیری تقویتی برای حل مسائل تصمیم‌گیری به خصوص زمان‌بندی کارها در مراکز داده‌ای ابری مورد استفاده قرار گرفته‌است. کار تحقیقاتی DeepRM [3]، از یادگیری تقویتی عمیق برای زمان‌بندی کارها (با تعریف حالت‌های محیط عامل یادگیرنده به شکل تصاویر) استفاده می‌کند. در DeepRM+ آقای چن[4] به بهبود DeepRM پرداخته و امکان مدیریت منابع در یک محیط چند خوشه‌ای و چند منبعی را با ارائه توصیف مشابه از محیط مطابق DeepRM محقق ساخته است. آقای چونگ و همکارانش در SCARL [5]، تمرکز بر شرایط عملیاتی پیچیده خوشه با نیازمندی و قابلیت‌های منابع مختلف دارد و در این راستا از جاسازی دقیق[1] و زمان‌بندی ضریب کنش‌ها که به طور موثر وابستگی درونی متغیر زمانی کار و ماشین را با هم در یادگیری تقویتی ترکیب و یک سیاست مقیاس پذیر پایانی را برای زمان‌بندی کارهای مختلف بر روی ماشین‌های ناهمگن فراهم می‌کند. در CuSH آقای دومنیکونی و همکارانش[6]، برای رسیدگی به پیچیدگی زمان‌بندی منابع ناهمگن، چارچوبی ارائه شده است که به کمک عامل یادگیری تقویتی کار بعدی را از صف انتخاب کرده و منابع را به آن تخصیص می‌دهد. آقای لی و همکارش[7]، الگوریتم DeepJS که یک الگوریتم زمان‌بندی کار براساس یادگیری تقویتی عمیق و چارچوب مسئله بسته‌بندی با جعبه[2] است و به طور خودکار به محاسبه تابع برازش می‌پردازد تا زمان تکمیل کارها (حداکثر سازی توان عملیاتی) را به حداقل برساند؛ ارائه کرده است. آقای لیانگ و همکارانش[8]، رهیافت DeepScheduler را ارائه کردند که با رهیافت عملگر-نقاد پیشرفته[3] در یادگیری تقویتی به زمان‌بندی کارها می‌پردازد. رهیافت پیشنهادی متشکل از دو عامل است: عملگر که به یادگیری خودکار سیاست زمان‌بند می‌پردازد و منتقد که خطای پیش‌بینی را کاهش می‌دهد. این رویکرد بر پایه کاهش واریانس تخمین گرادیان و بروزرسانی موثر پارامترها طراحی شده است.

هوانگ و همکارانش[9]، رهیافت RLSK را به عنوان یک زمان‌بند یادگیری تقویتی عمیق برای زمان‌بندی سازگار کارهای دسته‌ای[4] مستقل در میان دسته‌های مختلف محاسبه ابری متحد[5] ارائه کرده است. این زمان‌بند، با محیط سیستم تعامل دارد و خودکار راهبردهای زمان‌بندی را بدون دانش قبلی و دستورالعمل‌های انسانی بر روی محیط چند خوشه‌ای یاد می‌گیرد.

در این پژوهش ما، به دنبال طراحی مولفه زمان بند مبتنی بر یادگیری تقویتی هستیم تا فرآیند ملالت‌آور تکراری مثل زمان‌بندی منابع مرکزداده را با مدیریت خودکار آزمایش کند و در حین ایجاد تغییرات جدید، سیاست‌های هوشمندانه ابتکاری متناسب با اهداف مدنظر برای برخورد با مسئله ارائه کند. زمان‌بند ما به دنبال توجه به خاصیت خودکاری، عدم‌قطعیت، پویایی، تنوع در بارکاری، فراگیری سریع، بدون دخالت انسانی عامل یادگیرنده و بهره‌مندی عامل از انگیزش ذاتی کنجکاوی برای جستجوی بهتر فضای حالت و شناخت محیطی است.

در ادامه در این مقاله در بخش دوم برخی از مفاهیم اولیه مورد نیاز مرور می‌شوند. در بخش سوم رهیافت زمان‌بندی پیشنهادی معرفی می‌شود. جزئیات الگوریتم مولفه زمان‌بندی پیشنهادی در بخش چهارم ارائه شده است. در بخش پنج کارایی الگوریتم مولفه زمان‌بند در مقایسه با دو مولفه مشابه دیگر در حوزه یادگیری تقویتی ارزیابی شده است. در انتها و در بخش ششم نتیجه و کارهای آتی برای توسعه رهیافت پیشنهادی این مقاله ارائه شده است.

۲- مفاهیم اولیه

در این بخش برای فهم بهتر مطالب در ادامه این مقاله برخی مفاهیم اولیه توضیح داده شده اند.

۲-۱- خودتطبیق‌پذیری و ویژگی‌های آن

دسته گسترده‌ای از سیستم‌های خودکار و انطباقی[6] تمایل دارند که با پویایی بدون تداخل انسان مقابله کنند اما این سیستم‌ها لزوماً به عدم قطعیت[7] توجه ندارند. به عبارتی این سیستم‌ها تغییرات را زمانی‌که رخ می‌دهند متوجه و متناسب با آن رفتار می‌کنند. سیستم خودتطبیق‌پذیر از دسته سیستم‌هایی است که پویایی و عدم‌قطعیت را درنظر می‌گیرند. سیستم‌های خودتطبیق‌پذیر شامل دو زیرسیستم، مدیریتی و کنترل‌کننده سازگار است. در این‌جا به طور خاص سیستمی خودتطبیق‌پذیر است که توانایی ارائه رفتار سازگاری با توجه به درک عدم قطعیت محیط و وضعیت خود را داشته باشد. خودتطبیق‌پذیری دارای چهار ویژگی خودپیکربندی[8]، خودبهبودی[9]، خودبهینگی[10]، خودمحافظتی[11] است که هریک هدف خاصی را محقق می‌سازد[10].

۲-۲- یادگیری تقویتی

یادگیری تقویتی، نحوه انجام یک کار و نگاشت حالت‌ها به کنش‌هایی است که پاداش عددی یادگیرنده را بیشینه کند. همان‌طور که در شکل (۱) مشاهده می‌شود؛ در این رویکرد، به عامل یادگیرنده گفته نمی‌شود که چه کنش‌هایی را انجام دهد و عامل است که کشف می‌کند که چه کنش‌هایی منجر به دریافت بیش‌ترین پاداش خواهد شد[11]. شیوه فراگیری هوشمند سیاست[12]

در یادگیری تقویتی بر پایه فضای کنش[13] است. مدل ریاضی این سیستم با یک چهارتایی (S, A, P, R) تعریف می‌شود که در آن به ترتیب S فضای حالت[14]، A فضای کنش، P فضای احتمالی انتقال به وضعیت نامعلوم در محیط ($P:S\times S\times A\rightarrow[0,\infty)$) و R پاداشی که محیط به هر انتقال ($r:S\times A\rightarrow[rmin , rmax]$) نسبت می‌دهد و بیانگر میزان مطلوب بودن یک کنش است. نماد $\rho_\pi (s_t)$ و $\rho_\pi (s_t, a_t)$ به ترتیب بیانگر توابع حالت و حالت – کنش[15] توزیع حاشیه‌ای طی شده[16] عامل و $\pi(s_t, a_t)$ سیاست اتخاذ شده (یک تابع ساده، یک جدول یا شامل محاسبات پیچیده‌ای مانند یک فرآیند جستجو) است. آخرین بخش یک سیستم یادگیری تقویتی مدل محیط است. مدل، رفتار محیط را تقلید می‌کند[12].

۳- معرفی رهیافت زمان‌بندی

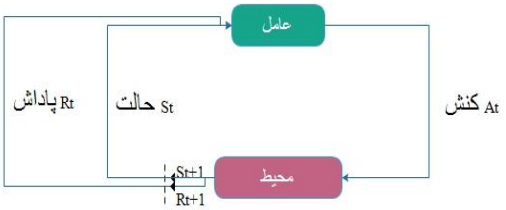
در این بخش به طرح مسأله و توضیح رهیافت خودتطبیق‌پذیر، معماری مولفه زمان‌بند رهیافت، فرمول‌بندی مسئله، اهداف و محدودیت‌ها می‌پردازیم.

۳-۱- رهیافت خودتطبیق‌پذیر پیشنهادی

با توجه به پیچیدگی روزافزون سیستم‌های فن‌آوری اطلاعات نیاز به سازوکاری که سیستم خودش بتواند بدون نیاز به دخالت انسان در برابر تغییرات محیطی واکنش نشان داده و خودکار خود را تطبیق دهد. شرکت آی بی ام[17] در یک بیانیه[18] در رابطه با محاسبات خودکار، یک معماری مرجع تحت عنوان سیستم‌های خودتطبیق‌پذیر ارائه و نام‌گذاری کرد. در یک معماری خودتطبیق‌پذیر لزوماً تمام مولفه‌های خودتطبیق‌پذیر عمل نمی‌کنند بلکه نتیجه رفتار آن‌ها کافی است که خودتطبیق‌پذیر باشد. در رهیافت پیشنهادی ما خودتطبیق‌پذیری با نظارت مداوم بر محیط و درک تغییر خصوصیات محیطی، سعی در تغییر سیاست مولفه خود برای بالابردن کارایی سیستم در مقابل تغییرات محیطی دارد.

۳-۲- معماری مولفه زمان‌بند رهیافت خودتطبیق‌پذیر

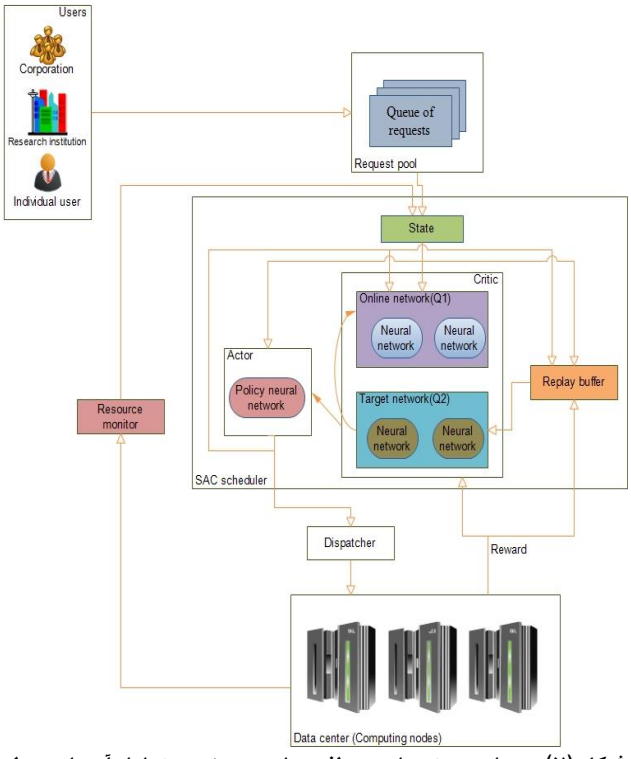
مولفه زمان‌بند پیشنهادی که از الگوریتم عملگر–منتقد نرم در زمان‌بندی خود استفاده می‌کند؛ از دو عامل تعاملی عملگر و منتقد تشکیل شده است. عامل عملگر، وظیفه تنظیم سیاست انتخاب کارهای دریافتی به صف و انباره(زمان‌بندی کارهای دریافتی با توجه به نیازمندی‌های آن‌ها بر روی منابع دردسترس محیط مرکز داده و برآوردن اهداف تابع ارزش‌محور) را برعهده



شکل (۱): شمای کلی الگوی یادگیری تقویتی

دارد. عامل منتقد، وظیفه ارزیابی خودکار توزیع سیاست‌های اعمال شده عملگر را با ارائه تابع وضعیت–کنش برعهده دارد. مولفه زمان‌بند نیازمند آموزش خواهد بود. این آموزش تکراری، متناسب با پاداش حاصل از تابع ارزش‌محور تا رسیدن به همگرایی ادامه می‌یابد. شکل (۲) طرح و شمایی از جزئیات عناصر درونی مولفه زمان‌بند پیشنهادی و نحوه تعامل آن‌ها با یکدیگر، مرکز داده و کارهای دریافتی از سمت کاربران را نشان می‌دهد.

عامل مدیریت منابع در تصمیم‌گیری‌های خود تمایل دارد که منابع موجود در محیط رایانش ابری را به صورت کارآمد و متناسب با نیازمندی‌های کار درخواستی کاربران فراهم کند تا پاسخگویی به تمام کارها در کمترین زمان ممکن میسر شود. بخش عمده اثربخشی این تصمیم‌گیری برعهده زمان‌بند منابع است. این مولفه، در ابتدا کارهای ارسالی از طرف کاربران محیط رایانش ابری را دریافت می‌کند و سپس بر پایه منابع درخواستی هر کاربر، منابع موجود و قابل تخصیص در محیط را به هر یک از درخواست‌ها نگاشت می‌کند تا معیارهای ارزیابی را بهینه کند. مولفه زمان‌بند در هنگام اجرای کارها، با بررسی وضعیت منابع خوشه، صف منابع درخواستی کاربر و انباره، برخط شیوه و سیاست زمان‌بندی دسترسی کارها به منابع را تعیین می‌کند و با ارزیابی شیوه و سیاست خود به اصلاح و بروزرسانی آن می‌پردازد. پس از تخصیص منابع به کار منتخب منابع تا پایان اجرای موفق سرویس درخواستی کاربر قبضه نخواهد شد(انحصاری) و با اتمام اجرای آن، منابع آزاد



شکل (۲): معماری پیشنهادی مولفه زمان‌بند و نحوه تعامل آن با محیط و عناصر درونی

و در اختیار خوشه منابع مرکز داده قرار می‌گیرد. مولفه زمان‌بند خودکار پیشنهادی در فرآیند زمان‌بندی در جهتی گام برمی‌دارد که همواره منابع را مشغول نگه ندارد و در صورت نبود درخواست جدید منتظر درخواست(بارکاری) جدید بماند. الگوریتم هوشمند انتخابی برای زمان‌بندی منابع مرکز داده با پیروی از چرخه فرآیند اجرایی مولفه زمان‌بند در تلاش است تا خودکار و بدون دخالت انسانی اهداف تعیین‌شده تابع ارزش‌محور را محقق سازد.

۳-۳- فرمول‌بندی مسئله

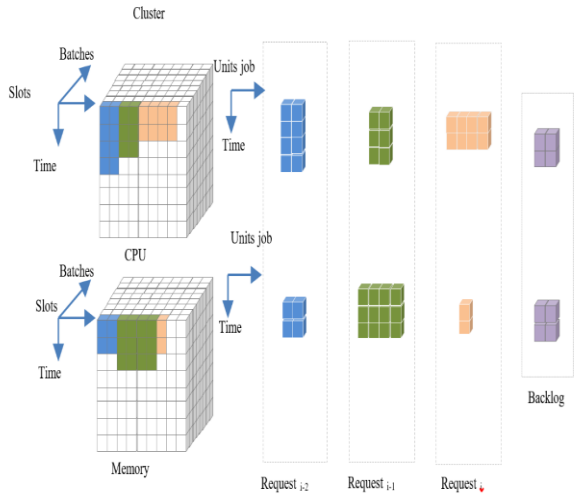
مدل ریاضی زمان‌بند کار که شامل وضعیت مرکز داده و صفوف درخواست و انباره، کنش مربوط به انتخاب یک درخواست از صف درخواست‌ها یا انباره، تابع ارزش‌محور و ضریب تخفیف را متناسب با الگوریتم عملگر-منتقد نرم و ذکر جزئیات ارائه می‌کنیم. اجزای این مدل ریاضی که متناسب به شرح زیر است:

۱). حالت ($s_t \in S$): s_t بیانگر وضعیت منابع مرکز داده با توجه به وضعیت خوشه، صف درخواست سرویس‌های کاربران در زمان t(کارهای موجود در صف) و انباره(کارهایی که امکان قرارگیری در صف پاسخگویی زمان‌بند را ندارند) است. مجموعه S یک مجموعه متناهی است. وضعیت s یک سه تایی است که به ترتیب شامل وضعیت منابع تخصیص‌یافته و منابع دردسترس مرکز داده، منابع درخواستی توسط کاربران که در صف قرار دارند و وضعیت درخواست‌های معطل در صف انباره که به هنگام زمان‌بندی برخط از آن‌ها بهره می‌بریم؛ است. زمان‌بند تنها درخواست‌های موجود در صف را زمان‌بندی می‌کند. شکل (۳) مثالی از وضعیت s را در یک گام زمانی نشان می‌دهد:

(۲). کنش ($a_t \in A$): یک کنش همانند a_t بیانگر شیوه انتخاب یک درخواست از صف انتظار درخواست‌های سرویس کاربران برای تخصیص منابع مرکز داده در زمان t است. اگر N درخواست سرویس در صف وجود داشته باشند آن‌گاه کنش منتخب از رابطه $a_t = \{a_t\}_1^N$ به دست می‌آید (۱ در این رابطه بیانگر تکرار کنش توسط عامل کنش‌گر است). در رابطه a_t فضای کنش شامل N+1 عضو است که در آن N انتخاب یکی از درخواست‌های سرویس کاربران و عضو آخر عدم انتخاب هیچ یک از کارها را نشان می‌دهد ($a_t = i \ (\forall i \in \{1..N\})$).

(۳). ضریب تخفیف[۱۹] (۷): ضریب تخفیفی در بازه [0,1] است که درجه تاثیر پاداش آنی و پاداش آتی را روی تابع پاداش نشان می‌دهد. این پارامتر ثابت که به صورت دستی انتخاب می‌شود هر چه دارای مقدار عددی کمتری باشد، اهمیت پاداش آتی که از تخمین تابع وضعیت-کنش آتی به‌دست می‌آید را کمتر نشان می‌دهد.

(۳). تابع پاداش($r \in R = S \times A \to (-\infty, 0)\}$): بازخورد عملگر در تعامل با مرکز داده و کارها منجر به دریافت پاسخ از محیط و عامل اجرایی محیط می‌شود. عملگر الگوریتم پیشنهادی به دنبال بیشینه کردن تابع ارزش محور (تابع پاداش و بی‌نظمی) است. تابع پاداش در زمان t با رابطه



شکل(۳): ورودی حالت مولفه زمان‌بند شامل وضعیت مرکز داده، صفوف درخواست و انباره در زمان t

$r_t = -\frac{1}{T_j}$ بیانگر زمان اجرای درخواست سرویس کاربر j است و T_j بیانگر مقدار جریمه تاخیر در پردازش کارهای موجود در حالت جاری یا جریمه نگهداشتن کارها در وضعیت جدید مرکز داده و یا جریمه ازدست رفتن درخواست به دلیل پر بودن صف) محاسبه می‌شود. در طی یک دور آموزش به دلیل وجود تکرارهای آموزشی تمام پاداش‌ها به دست‌آمده با یکدیگر تجمیع و میانگین آن‌ها به عامل منتقد مولفه زمان‌بند ارائه می‌شود.

(۴). تابع حالت-کنش($Q(s_t, a_t)$): ارزیابی کارایی سیاست‌های مختلف اعمالی از طرف عملگر بر روی مرکز داده با کمک تابع وضعیت-کنش که معادل مقدار متوسط(امید ریاضی) پاداش دریافتی از مرکز داده در طی تکرارهای متوالی است؛ سنجیده می‌شود. این تابع به کمک عامل منتقد تخمین زده می‌شود تا به کمک آن در تکرارهای بعدی عامل منتقد براساس آن پیش‌بینی سیاست خود را اصلاح کند.

۳-۴- اهداف مدیریت منابع پیشنهادی

در رهیافت پیشنهادی هدف اصلی طراحی زمان‌بند منابع است که به ما توانمندی خودکاری، توجه به عدم قطعیت، پویایی و تنوع در بارکاری، فراگیری سریع و بدون دخالت انسانی عامل یادگیرنده و جستجوی بهتر فضای مرکزداده دهد. همچنین در گام بعدی، با سیاست فراگیری‌شده با نگاشت کارها به منابع مرکزداده میانگین کندی پاسخ را کاهش و بهره‌وری منابع را برای تمام کارهای دریافتی افزایش دهد.

۳-۵- محدودیت‌های زمان‌بندی

رهیافت مطرح‌شده در این پژوهش برای حل مسئله زمان‌بندی خودکار منابع در مرکز داده دارای معایب و محدودیت‌های زیر است:

- سربار محاسباتی و زمانی فرآیند همگرا شدن عامل یادگیرنده
- امکان بروز over fit در نتایج
- عدم پوشش ناهمگونی منابع مرکز داده
- انحصاری بودن زمان‌بندی

٤- الگوریتم زمان‌بندی رهیافت پیشنهادی

در این بخش جزئیات الگوریتم زمان‌بندی پیشنهادی را شرح می‌دهیم.

٤-١- الگوریتم عملگر- منتقد نرم[۲۰]

این الگوریتم، یک الگوریتم یادگیری تقویتی بدون مدل عمیق است که برای کارهای مرتبط با تصمیم‌گیری و مدیریت استفاده می‌شود. سیاست استفاده شده در این الگوریتم، خط‌مشی غیربرخط[۲۱] است. به بیان دقیق‌تر، در این الگوریتم، ارزیابی یا بهبود سیاست عامل هوشمند از سیاست‌های قبلی انجام‌شده بر روی محیط (آخرین سیاست اعمالی) پیروی نمی‌کند بلکه به تمام سیاست‌هایی که در تکرارهای گذشته رخداده است توجه می‌کند که منجر به آموزش پایدار عامل می‌شود. الگوریتم عملگر-منتقد نرم، همانند سایر شیوه‌های یادگیری تقویتی، به دنبال بیشینه کردن پاداشی است که عامل یادگیرنده بر اثر تعامل با محیط به دست می‌آورد[13]. از الگوریتم ذکرشده در محیط رایانش ابری، به منظور بیشینه کردن تابع هدف فرآیند مدیریت منابع (تخصیص بهینه منابع با توجه به نیازمندی‌های دریافتی از کاربران است) استفاده شده است. دیگر ویژگی این الگوریتم، توجه به بی‌نظمی به معنای غیر قابل پیش‌بینی بودن احتمال وقوع یک رخداد است. این ویژگی در الگوریتم پیشنهادی زمان‌بند، امکان پاسخگویی و مدیریت بهتر مولفه زمان‌بند نسبت به تنوع درخواست خدمت‌های دریافتی کاربران که می‌توانند بسته به شرایط مختلف غیرقابل پیش‌بینی باشند را فراهم می‌کند.

٤-٢- شیوه بهینه‌سازی الگوریتم

در الگوریتم عملگر-منتقد نرم به شکل عمومی‌تر، عامل یادگیرنده علاوه بر بیشینه کردن پاداش خود در تعامل با محیط به دنبال بیشینه کردن بی‌نظمی در سیاست اعمالی خود در محیط است. بی‌نظمی، یک کمیت پایه‌ای در نظریه اطلاعات[۲۲] است. این کمیت، یک معیار عددی از میزان اطلاعات (میزان تصادفی بودن) یک متغیر تصادفی است. به بیان دقیق‌تر، بی‌نظمی یک متغیر تصادفی، مقدار متوسط (امید ریاضی) میزان اطلاعات حاصل از مشاهده آن است. (به عبارت ساده‌تر، هرچه بی‌نظمی یک متغیر تصادفی بیشتر باشد ابهام ما نسبت به متغیر تصادفی بیشتر است. لذا با مشاهده نتیجه قطعی آن اطلاعات بیشتری به دست می‌آید. اگر X یک متغیر تصادفی با تابع احتمال P(x) باشد آن‌گاه بی‌نظمی به صورت فرمول (۱) تعریف می‌شود:

$$H(X) = -\sum_{i=1}^{n} P(x_i)\log_b P(x_i) = \mathbb{E}[-\log_b P(X)] \quad (۱)$$

*شایان ذکر است که متداول‌ترین مقادیر برای b، ۲، e و ۱۰هستند.

وجود بی‌نظمی، عامل هوشمند را به جستجوی بیشتر تشویق می‌کند تا امکان انتخاب کنش‌هایی با احتمال یکسان یا کنش‌هایی که نتایجی نزدیک به حالت بهینه دارند؛ انتخاب شوند تا عامل محدود به تعدادی کنش تکراری محدود نباشد. بی‌نظمی متغیر تصادفی X به شرط متغیر تصادفی S را مطابق فرمول(۲) تعریف شده است:

$$H(X|S) = -\sum_{i,j} P(x_i, y_i)\, log \frac{P(x_i, y_i)}{P(y_i)} \quad (۲)$$

دیگر بخش مهم در نظریه اطلاعات، مفهوم اطلاعات متقابل[۲۳] (میزان شباهت بین توزیع مشترک H(X|Y) است. این فرمول احتمالی به صورت غیر مستقیم امکان دستیابی به اطلاعات متغیر Y با داشتن اطلاعات متغیر تصادفی X را فراهم می‌کند[14]. فرمول (۳) اطلاعات متقابل متغیر تصادفی X و Y را نشان می‌دهد:

$$I(X;Y) = H(X) - H(X|Y) \quad (۳)$$

اطلاعات متقابل شرطی به ما امکان اندازه‌گیری اطلاعات موجود در یک متغیر نسبت به متغیر تصادفی دیگر به شرط داشتن مقدار متغیر تصادفی سوم را فراهم می‌کند. اطلاعات متقابل شرطی به شیوه‌های مختلف نوشته می‌شود که فرمول(۴) به آن‌ها اشاره شده است:

$$I(X;Y|S) = H(X|S) - H(X|Y,S) = H(Y|S) - H(Y|X,S) \quad (۴)$$

با توجه به آنچه در مورد شیوه بیشینه‌سازی مقدار متوسط پاداش و بی نظمی مطرح شد؛ سیاست اعمالی الگوریتم عملگر-منتقد نرم بر روی وضعیت‌های محیط$\rho_\pi(s_t)$ را مطابق فرمول (۵) است:

$$J(\pi) = \sum_{t=0}^{T} \mathbb{E}_{(s_t,a_t)\sim\rho_\pi}[r_{(s_t,a_t)} + \alpha H(\pi(.|s_t))] \quad (۵)$$

در فرمول بالا، $J(\pi)$ بیانگر میزان کارایی سیاست π در بازه صفر تا T، T طول فرآیند اجرایی عامل هوشمند، $r_{(s_t,a_t)}$ پاداش مربوط به انتخاب

کنش a در وضعیت s و زمان t است، $H(\pi(.|s_t))$ بی‌نظمی تمام کنش‌های احتمالی ممکن در وضعیت s و زمان t که از معادله ۱ به دست می‌آید و α پارامتر دما[۲۴] اهمیت نسبی بی‌نظمی در برابر پاداش دریافتی را نشان می‌دهد.

٤-٣- سیاست تکراری

در فرآیند تصمیم‌گیری مارکوفی (MDP) کلاسیک، به رهیافت استانداردی که به دنبال یافتن سیاستی است که متوسط پاداش کاسته‌شده تجمیعی[۲۵] برای هر وضعیت محیط بیشینه کند؛ تکرار سیاست[۲۶] گویند. تکرار سیاست یک برنامه تکراری دو مرحله‌ای است که میان ارزیابی سیاست[۲۷] و بهبود سیاست[۲۸] متناوب نوسان می‌کند. در مرحله ارزیابی یک سیاست، هدف نهایی دستیابی به تابع هدفی بهینه‌ای است که نتایج مدنظر طراح عامل هوشمند را برآورده سازد. بدین ترتیب به طور تکراری از معادله بهینه بلمن[۲۹] به شکل فرمول (۶) استفاده می‌شود:

$$[T_\pi V] = \mathbb{E}_{a\sim\pi(.|S)}[r(S,a) + \gamma \mathbb{E}_{S|\acute{S},a}[V(\acute{S})] \quad (۶)$$

این معادله، با توجه به مقدار اولیه تابع ارزش V تضمین می‌کند که مقدار بهینه تابع ارزش V^* را با توجه به سیاست بهینه π^* به دست آورد. سیاست تکراری الگوریتم عملگر–منتقد نرم، یک تعمیم از رهیافت استاندارد معادله ۵ است که بی‌نظمی را در کنار پاداش استاندارد به عنوان سیاست تکراری خود درنظر می‌گیرد. بدین ترتیب معادله بهینه بلمن به صورت معادله (۷) تعمیم می‌یابد:

$$T^\pi Q(s_t,a_t) \triangleq r_{(s_t,a_t)} + \gamma \mathbb{E}_{\acute{a}\sim\pi}[Q(s_{t+1},\acute{a}) - \log\pi(\acute{a}|s_t)] \quad (۷)$$

در مرحله بهبود سیاست، سیاست الگوریتم با اعمال توزیع سافت–مکس[۳۰] بر تابع Q بروزرسانی می‌شود. این انتخاب منجر به بیشینه شدن پاداش عامل و تضمین بهبود نتایج حاصل از سیاست (مقدار سافت[۳۱]) خواهد شد. فرمول‌های (۸) و (۹) نحوه کمینه کردن بی‌نظمی نسبی (سیاست الگوریتم) را نشان می‌دهد[15]:

$$\pi_{new} = argmin_{\pi\epsilon\Pi}^{J_\pi(\varphi)} \quad (۸)$$

$$J_\pi(\varphi) = \mathbb{E}_{s_t\sim D}[D_{KL}(\pi_\varphi(.|s_t))||\frac{\exp(\frac{1}{a}Q_\theta(s,.)}{Z_\theta(s_t)}] \quad (۹)$$

با حذف پارامتر ثابت Z و تبدیل انتگرال به امید ریاضی (تبدیل فضای پیوسته الگوریتم به گسسته) معادله بالا به فرمول (۱۰) کاهش می‌یابد:

$$J_\pi(\varphi) = \mathbb{E}_{s_t\sim D}[\mathbb{E}_{a_t\sim\pi_\varphi}[\alpha\log(\pi_\varphi(a_t|s_t) - Q_\theta(s_t,a_t)]] \quad (۱۰)$$

و اگر سیاست اعمالی بر عامل یادگیرنده توزیع رسته‌ای[۳۲] باشد فرمول(۱۱) نتیجه آن خواهد بود:

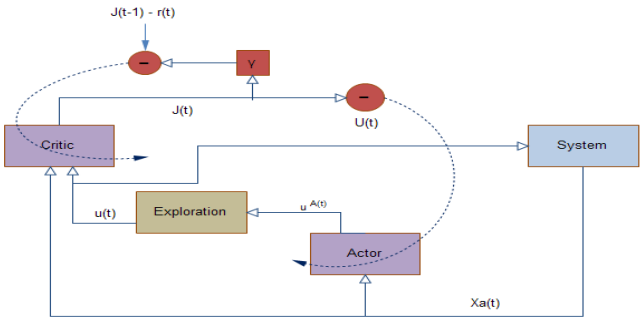
$$J_\pi(\varphi) = \mathbb{E}_{s_t\sim D}[\pi_\varphi(.|s_t)^T[\alpha\log(\pi_\varphi(a_t|s_t) - Q_\theta(s_t,a_t)]] \quad (۱۱)$$

٤-٤- شمای کلی الگوریتم عملگر- منتقد نرم

در شکل (۴)، x(t) بیانگر وضعیت جاری محیط، r(t) پاداش آنی که توسط سیستم، $u^A(t)$ خروجی تقریبی کنش بهینه که با دریافت Xa(t) توسط عملگر، J(t) تقریبی از تمام پاداش های تولیدی از لحظه صفر تا T است که توسط منتقد تولید می‌شود و U (t) هدف نهایی عامل هوشمند است. چارچوب رهیافت الگوریتم عملگر– منتقد نرم شامل ۳ مولفه اصلی است: (۱) عملگر، (۲) نقاد و (۳) سیستم. عملگر، کنش‌های عامل هوشمند کنترلی را تولید می‌کند. سیستم، مسئول پاسخگویی به کنش‌های کنترلی دریافتی از عامل و انتقال از وضعیت جاری به وضعیت آتی را برعهده دارد. منتقد نیز کنش‌های کنترلی تولیدشده عملگر را ارزیابی می‌کند.

٥- ارزیابی و تحلیل کارایی

طراحی و پیاده‌سازی رهیافت پیشنهادی بر پایه DeepRM و زبان پایتون[۳۳] نسخه ۳ آماده‌سازی شده است. عوامل یادگیری تقویتی که برای یادگیری، اصلاح و پیش‌بینی حالت بعدی محیط به‌کار گرفته شده همگی در چارچوب کتابخانه‌های نامپای[۳۴] و پایتورچ مدل‌سازی شده‌اند. فرآیند آموزش، با ویژگی سخت‌افزاری: ۱) هشت هسته مجازی پردازنده مرکزی ۲) شانزده گیگابایت حافظه رم و ۳) سی گیگابایت حافظه دیسک سخت و ویژگی‌های نرم‌افزاری مرتبط به کامپایل و اجرا برنامه تولیدی شامل: ۱) سیستم عامل ویندوز۱۰ ۲) کامپایلر پایچارم[۳۵] مرکز داده دانشگاه علم و صنعت ایران انجام شده است.



شکل(٤): شمای کلی الگوریتم عملگر- منتقد نرم

۵-۱- آزمایش شبیه‌سازی

عملکرد مولفه زمان‌بند خودکار، در بخش شبیه‌سازی برپایه کارهای مرتبط و سنتی در دو الگو مولفه پیشنهادی براساس پیش آموزش عامل یادگیرنده در طول بازه شبیه‌سازی ۵۰ تایی به تعداد ۶۰ بار و با یک دور تکرار با داده‌های از پیش مشاهده نشده برای روش‌های یادگیری تقویتی اجرا شده است.

۵-۱-۱- الگوی تغییر تقاضای منابع نرمال

در این الگو، میزان منابع درخواستی کاربر(میزان پردازنده مرکزی و حافظه) مستقل از هم و متناسب با زمان درخواستی استفاده از منابع در بازه مفروضات اولیه آزمایش(در بخش بعد جزئیات آن بیان شده است) به صورت تصادفی و متناسب با توزیع نرمال انتخاب شده‌اند. در شکل (۵) نمونه این الگو نشان داده شده است.

۵-۲- مقایسه و ارزیابی کارایی

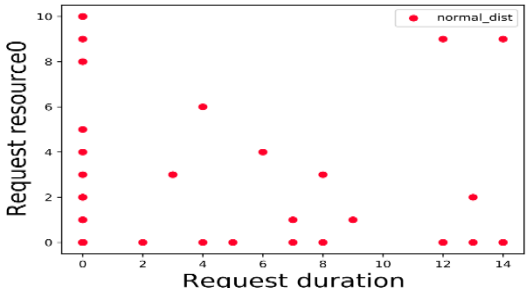
عملکرد مولفه زمان‌بند پیشنهادی ما با DeepRM و DeepScheduler مطابق با الگوهای بخش ۱-۵ مقایسه شده است.. برای ارزیابی کارایی از معیار بیشینه و میانگین کندی درخواست[۳۶] که کندی درخواست مطابق فرمول (۱۲) تعریف می‌شود و در آن c_i زمان تکمیل درخواست و T_i مدت زمان درخواست سرویس است استفاده شده است. همچنین به‌منظور مقایسه کارایی سیاست مولفه زمان‌بند، میانگین بهره‌وری منابع و درجه تعادل کندی پاسخ (از تفاضل کمینه و بیشنه کندی پاسخ در یک در طول بازه شبیه‌سازی) مقایسه شده‌اند.

$$s_i = \frac{t_i^f - t_i^a}{t_i^e} = \frac{c_i}{T_i} \qquad (۱۲)$$

همان‌طور که جدول (۱) مشاهده می‌شود زمان‌بند پیشنهادی در مقایسه با مولفه‌های مشابه خودکار بادگیری با اختلاف زیاد از منظر تمام مولفه‌های ارزیابی بهبود داشته است. همچنین عدم بهبود کندی میانگین پاسخ کارهای مقایسه‌شده به خاطر توقف و نگهداشتن برخی از کارها در صف درخواستی مرکز داده است که خود گواهی از بیشتر شدن گرسنگی در کارهای بزرگ و عدم توجه به ایجاد تعادل در پاسخگویی به درخواست‌ها و بهره‌وری پایین مرکز داده است. رهیافت پیشنهادی به خوبی و بدون نیاز به تنظیم پارامترهای یادگیری سیاست تخصیص و زمان‌بندی کارها فراگرفته است و این خود دلیلی دیگر بر ویژگی الگوریتم عملگر-منتقد نرم است که با بهره از سیاست خاموش‌سازی نمونه‌گیری کارآمدتری نسبت به سایر مولفه‌ها داشته باشد.

جدول (۱): نتایج ارزیابی دو مولفه مشابه با رهیافت پیشنهادی

مولفه‌های زمان‌بند	بهره‌وری CPU	بهره‌وری حافظه	درجه تعادل کندی پاسخ	بیشینه کندی پاسخ	میانگین کندی پاسخ
DeepRM	۰/۰۷۸۹۲۰	۰/۰۷۲۸۷	۲۵/۸۷۱۸۹	۲۵۶	۹/۸۵۶۲۵
DeepScheduler	۰/۰۳۶۷۸۰	۰/۳۵۴۵۸	۵۸/۲۸۸۱۱	۱۰۰۵	۱۷/۲۲۴۷
رهیافت پیشنهادی	۰/۱۶۱۰۱۶	۰/۱۶۴۴۴۶	۸/۲۳۱۴۴	۶۳	۷/۵۳۲۰۸



شکل(۵): نمونه‌ای از الگوی نرمال تقاضای منابع کارها

۶- نتیجه

در این پژوهش در ابتدا مزایا و معایب هر یک از پژوهش‌های مرتبط با رهیافت ارائه‌شده را مورد بررسی قرار داده است. سپس رهیافت پیشنهادی، در جهت توسعه و پیشرفت کارهای تحقیقاتی مرتبط امروزی تلاش شده است مولفه زمان‌بند خودکار مبتنی بر یادگیری تقویتی(به کمک انگیزش ذاتی، ساختار الگوریتم عملگر-منتقد نرم) طراحی شود که با آن زمان‌بندی بهتری نسبت به شرایط متنوع و غیر قطعی کارها در مرکز داده از خود نشان دهد. در رهیافت پیشنهادی، الگوریتم منتخب یک الگوریتم یادگیری تقویتی عمیق است که با بیشینه‌سازی بی‌نظمی به جستجو بیشتر در محیط(حاوی اختلالات محیطی) می‌پردازد تا نمونه‌های کارآمدتر از محیط به‌دست آورد. نوآوری و دستاوردهای این پژوهش به شرح زیر است:

(۱). مولفه زمان‌بند نسبت به سایر پژوهش‌ها معیارهای ارزیابی بیشینه میانگین کندی پاسخ و نقطه تعادل در پاسخ‌دهی به کارها را کاهش و بهره‌وری را افزایش داده است.

(۲). اثربخشی رهیافت پیشنهادی نسبت به رهیافت‌های خودکارسازی مرتبط بدون نیاز به تنظیم‌های دقیق چندباره پارامترهای یادگیری نشان داده شد و دلیل اصلی آن، نمونه‌برداری کارآمد و بی‌نظمی در انتخاب‌های مولفه زمان‌بند است.

برای توسعه و بهبود آتی رهیافت ارائه شده می‌توان اجرای رهیافت با الگوی‌های دیگر و متنوع تقاضای منبع، استفاده از دادگان خوشه‌های محیط‌های رایانش ابری، مدل‌سازی مسئله مولفه زمان‌بند با شاخص‌های ارزیابی دیگر همچون هزینه، زمان‌بندی بر مبنای ناهمگونی و احتمال خرابی

منابع مرکزداده و بهره‌مندی از سایر رهیافت‌های مدیریت بهینگی منابع همچون نظریه بازی‌ها تمرکز داشت..

مراجع

[1] M. Xu and R. Buyya, "*Brownout Approach for Adaptive Management of Resources and Applications in Cloud Computing Systems*", *ACM Computing Surveys*, vol. 52, no. 1, pp. 1-27, 2019. Available: 10.1145/3234151.

[2] G. Rjoub, J. Bentahar, O. Abdel Wahab and A. Saleh Bataineh, "*Deep and reinforcement learning for automated task scheduling in large- scale cloud computing systems*", *Concurrency and Computation: Practice and Experience*, 2020. Available: 10.1002/cpe.5919.

[3] H. Mao, M. Alizadeh, I. Menache and S. Kandula, "*Resource Management with Deep Reinforcement Learning*", in Proceedings of the *15th ACM Workshop on Hot Topics in Networks*, Atlanta GA, USA, November-2016, pp. 50–56.

[4] W. Chen, Y. Xu, and X. Wu, "*Deep Reinforcement Learning for Multi-Resource Multi-Machine Job Scheduling*", arXiv preprint arXiv:1711.07440, Nov. 2017.

[5] M. Cheong, H. Lee, I. Yeom and H. Woo, "*SCARL: Attentive Reinforcement Learning-Based Scheduling in a Multi-Resource Heterogeneous Cluster*", *IEEE Access*, vol.7,pp.153432-153444,2019.Available: 10.1109/access.2019.2948150.

[6] G. Domeniconi, E. Lee, V. Venkataswamy and S. Dola, "*CuSH: Cognitive Scheduler for Heterogeneous High Performance Computing System*", in Proceedings of *DRL4KDD 19: Workshop on Deep Reinforcement Learning for Knowledge Discovery (DRL4KDD)*, Alaska, USA, 2019.

[7] F. Li and B. Hu, "*DeepJS: Job Scheduling Based on Deep Reinforcement Learning in Cloud Data Center*", in Proceedings of *the 2019 4th International Conference on Big Data and Computing, Guangzhou*, China, May-2019, pp. 48–53.

[8] S. Liang, Z. Yang, F. Jin and Y. Chen, "*Data Centers Job Scheduling with Deep Reinforcement Learning*", in Proceedings of *24th Pacific-Asia Conference on Knowledge Discovery and Data Mining*, Singapore, Singapore, May-2020, pp. 906-917.

[9] J. Huang, C. Xiao and W. Wu, "*RLSK: A Job Scheduler for Federated Kubernetes Clusters based on Reinforcement Learning*", in Proceedings of *2020 IEEE International Conference on Cloud Engineering (IC2E)*, Sydney, Australia, Australia, April-2020.

[10] T. Chen, R. Bahsoon and X. Yao, "*A Survey and Taxonomy of Self-Aware and Self-Adaptive Cloud Autoscaling Systems*", *ACM Computing Surveys*, vol. 51, no. 3, pp. 1-40, 2018. Available: 10.1145/3190507.

[11] R. Sutton, F. Bach and A. Barto, *Reinforcement Learning*, 2nd ed. *Massachusetts: MIT Press Ltd*, 2018.

[12] X. Dutreilh, S. Kirgizov, O. Melekhova, J. Malenfant, N. Rivierre and I. Truck, "*Using Reinforcement Learning for Autonomic Resource Allocation in Clouds: towards a fully automated workflow*", in Proceedings of *the 7th International Conference on Autonomic and Autonomous Systems (ICAS'2011)*, Venice, Italy, May-2011, pp. 67-74.

[13] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, *"Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor"*, arXiv preprint arXiv: 1801.01290, Aug 2018.

[14] A. Aubret, L. Matignon, and S. Hassas, *"A survey on intrinsic motivation in reinforcement learning"*, arXiv preprint arXiv:1908.06976, Nov 2019.

[15] P. Christodoulou, "*Soft Actor-Critic for Discrete Action Settings*", arXiv preprint arXiv: 1910.07207, Oct. 2019.

زیرنویس‌ها

۱ Attentive embedding
۲ Bin packing
۳ Advance Actor Critic
٤ Batch
٥ Federated
٦ Adaptive
۷ Uncertainty
۸ Self-configuration
۹ Self-healing
۱۰ Self-optimization
۱۱ Self-protection
۱۲ Policy
۱۳ Action
۱٤ State
۱٥ State-action
۱٦ Marginal of trajectory distribution
۱۷ IBM
۱۸ Manifest
۱۹ Discount factor
۲۰ soft-actor-critic
۲۱ Offline
۲۲ Information theory
۲۳ Mutual information
۲٤ Temperature parameter
۲٥ Expected cumulative discounted
۲٦ Policy iteration
۲۷ Policy evaluation
۲۸ Policy improvement
۲۹ Bellman
۳۰ Softmax
۳۱ Soft value
۳۲ Categorical distribution
۳۳ Python
۳٤ Numpy
۳٥ PyCharm

Average slow down [۲۶]