دومین کنفرانس بین‌المللی ترکیبیات، رمزنگاری و محاسبات
I4C 2017
Homepage: http://i4c.iust.ac.ir
دانشگاه علم و صنعت ایران

ارائه یک الگوریتم جدید جهت ایجاد تعادل بار روی ماشین‌های مجازی در رایانش ابری

Propose A New Algorithm for Load Balancing on Virtual Machines in Cloud Computing

زهرا عیسوندی[۱]، نیما جعفرزاده[۲]

۱- کارشناسی ارشد، دانشگاه آزاد اسلامی، واحد تهران جنوب، گروه کامپیوتر، تهران، ایران
Isvandi_info@yahoo.com

۲- عضو هیئت علمی دانشگاه آزاد اسلامی، واحد تهران جنوب، گروه کامپیوتر، تهران، ایران
jafarzadenima@gmail.com

چکیده

در حال حاضر مؤلفه‌ی تعادل بار[۱] یک چالش اصلی در معماری رایانش ابری می‌باشد. تعادل بار کمک می‌کند تا حجم کار پویا بین گره‌های مختلف توزیع شود تا اطمینان حاصل شود که هیچ گره‌ای دچار بار بیش از حد[۲] نشود. این موضوع را می‌توان به عنوان مشکل بهینه‌سازی در نظر گرفت. یک متعادل‌کننده بار مناسب باید به بهبود همزمان پارامترهای: کاهش زمان اتمام آخرین کار در میان منابع، کم کردن مجموع هزینه پرداختی از سوی کاربران جهت اجاره منابع، توانایی مقیاس‌پذیری، اجتناب از گلوگاه‌ها و افزایش میزان بهره‌وری منجر گردد. به‌علاوه کارائی سیستم را ارتقا دهد. اگرچه الگوریتم‌های تعادل بار مختلفی طراحی شده‌اند که با انتخاب ماشین‌های مجازی درست، در درخواست تخصیص کارآمد می‌باشند ولی با توجه به اهمیت فرآیند تعادل بار در رایانش ابری، هدف این تحقیق بررسی این فرآیند و ارائه یک الگوریتم جدید جهت ایجاد تعادل بار در رایانش ابری با به حداقل رساندن زمان تکمیل آخرین کار[۳] و به حداکثر رساندن توان عملیاتی[۴] می‌باشد. نتایج حاصل از شبیه‌سازی، در نرم‌افزار متلب پیاده‌سازی شد و نشان داد روش پیشنهادی جدید که مبتنی بر ترکیب «الگوریتم بهینه‌سازی ازدحام ذرات[۵] و استراتژی جهش الگوریتم ژنتیک[۶]» می‌باشد، از کارائی بالایی برخوردار است و می‌تواند به تعادل بار خوبی در محیط رایانش ابری مقیاس بزرگ نسبت به الگوریتم‌های تعادل بار قبلی برسد. مقایسه‌ی نتایج در قالب نمودار و مشروح ارائه گردیده است.

کلمات کلیدی[۷]: رایانش ابری، تعادل بار، ماشین‌های مجازی، الگوریتم‌های تکاملی

[1] Load Balancing
[2] Overloaded
[3] Makespan
[4] Speedup
[5] Particle Swarm Optimization (PSO)
[6] Genetic Algorithm (GA)
[7] Key phrases: Cloud Computing, Load Balancing, Virtual Machines, Evolutionary Algorithms

۱- مقدمه

رایانش ابری یکی از جدیدترین تحولات در فن‌آوری اطلاعات است و با مرور زمان فراگیر می‌شود. کارایی یک سیستم کامپیوتری به چندین فاکتور وابسته است که یکی از آن‌ها متوازن‌سازی درخواست‌ها است. این مکانیزم کاملا به میزان کاری که در یک دوره‌ی زمانی خاص به سیستم محول شده، وابسته است. بنابراین سیستم، می‌بایست براساس اصول اولویت کارها مدیریت شود. محاسبات ابری مدلی کامپیوتری است که تلاش می‌کند دسترسی کاربران را براساس نوع تقاضاهایی که از منابع اطلاعاتی و محاسباتی دارند آسان کند[۱۱]. این مدل سعی دارد با کم‌ترین نیاز به منابع نیروی انسانی و کاهش هزینه‌ها و افزایش سرعت دسترسی اطلاعات جوابگوی نیاز کاربران باشد[۱۹]. همیشه یک راه‌حل توزیعی مورد نیاز می‌باشد، زیرا همواره نگهداری یک یا چند سرور بی‌کار و غیرفعال تنها برای انجام برخی از خواسته‌های مورد نیاز، امکان‌پذیر نمی‌باشد و یا این‌که به‌صرفه نیست. تعادل بار یک تکنیک بهینه‌سازی محسوب می‌شود. زیرا توان عملیاتی و بهره‌وری را بالا می‌برد، سرعت بازیابی و سرعت پاسخ‌گویی را کاهش داده و از رخ دادن سرریز در سیستم جلوگیری می‌کند[۴]. تعدادی از الگوریتم‌های متوازن‌سازی درخواست‌ها برای بهبود و بهینه‌سازی عملکرد ابر وجود دارند. طبیعت هر الگوریتم می‌تواند پویا یا ثابت باشد به‌علاوه الگوریتم‌هایی وجود دارند که در عین سادگی، قابلیت کارایی بهتری تحت شرایط خاصی دارند. برای مدیریت مناسب منابع فراهم کننده سرویس، به تعادل بار نیازمندیم که به ارائه دهنده‌ی سرویس پیشنهاد می‌شود. پردازش ابر یک معماری سرویس محور است که ارتباط آن از طریق اینترنت محقق می‌شود یکی از مهم‌ترین هدف‌های سرویس‌دهنده فراهم آوردن حداکثر منابع خروجی است که این مهم از طریق تعبیه کردن الگوریتم‌های متعادل‌سازی درخواست‌ها به دست می‌آید.

یک الگوریتم تعادل بار کارآمد نشان می‌دهد که هر گره در سیستم می‌تواند کارکرد مشابهی با یک گره دیگر داشته باشد. مدیریت الگوریتم تعادل بار به معنی مدیریت شرایطی است که در آن یک منطقه ابری برای خدمات بدون کاربرد استفاده می‌شود. از این روی، کل زمان واکنش قابل دسترس، بهبود یافته و در عین حال زمان واکنش دسترسی بهبود می‌یابد و یک منبع کارآمد را در اختیار می‌گذارد. عدم قطعیت با شرایط ابری متغیر است[۲۶] و لذا یکی از چالش‌های مهم تعادل بار است، زیرا تعیین میزان تقاضا در محیط ابری به سادگی میسر نیست.

رهیافت‌های بسیاری برای حل مسئله‌ی تخصیص وظایف در محیط‌های محاسباتی ارائه شده‌اند که می‌توانند به چهار گروه اصلی تقسیم شوند: تئوری گراف، برنامه‌ریزی خطی، جستجوی فضای حالت و رهیافت‌های اکتشافی از قبیل، تبرید شبیه‌سازی شده، جستجوی ممنوعه، الگوریتم ژنتیک، هوش ازدحامی[۵].

هدف از ارائه الگوریتم پیشنهادی، به دست آوردن یک الگوی مناسب جهت نگاشت کار به ماشین مجازی می‌باشد که بتواند بدون افزایش توان مصرفی، زمان پاسخ و زمان تکمیل آخرین کار را به حداقل برساند. در این پژوهش، برای حل مسأله تعادل بار از ترکیب دو الگوریتم ژنتیک و تجمع ذرات استفاده شده است. مشکل اصلی الگوریتم بهینه‌سازی ازدحام ذرات گیر افتادن ذرات در بهینه محلی و به وجود آمدن همگرایی زودرس می‌باشد. مشکل اصلی الگوریتم ژنتیک فاقد حافظه است. در نتیجه، جهت بهبود همگرایی و تعادل بارکاری روی ماشین‌های مجازی در محیط رایانش ابری از عملگر جهش الگوریتم ژنتیک در الگوریتم بهینه‌سازی ازدحام ذرات استفاده شده است.

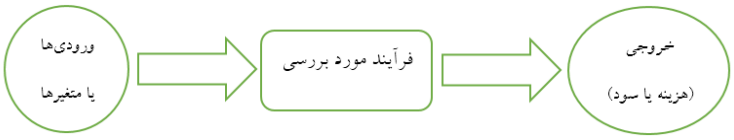
ادامه این مقاله بدین صورت است که در بخش دوم به توصیف بهینه‌سازی و معرفی الگوریتم بهینه‌سازی ازدحام ذرات و الگوریتم ژنتیک پرداخته شده؛ در بخش سوم به بررسی روش پیشنهادی خود در حوزه‌ی تخصیص منابع و ایجاد تعادل بار مناسب در محیط رایانش ابری پرداخته شده است. در واقع با استفاده از دو الگوریتم «بهینه‌سازی ازدحام ذرات و ژنتیک» سعی شده است که نتیجه بهتری را برای مسأله تعادل بار در بستر ابر به‌دست آورد. در بخش چهارم به بیان نتایج حاصل از شبیه‌سازی پرداخته شده و در نهایت بخش پنجم به نتیجه‌گیری مقاله پرداخته شده است.

۲- بهینه‌سازی

تعادل بار یک تکنیک بهینه‌سازی محسوب می‌شود. زیرا توان عملیاتی و بهره‌وری را بالا می‌برد، سرعت بازیابی و سرعت پاسخ‌گویی را کاهش داده و از رخ دادن سرریز در سیستم جلوگیری می‌کند[۴]. بهینه‌سازی فرآیندی است که برای بهتر کردن چیزی دنبال می‌شود. فکر، ایده و یا طرحی که به وسیله یک دانشمند یا یک مهندس مطرح می‌شود، طی روال بهینه‌سازی بهتر می‌شود[۱]. در هنگام بهینه‌سازی، شرایط اولیه با روش‌های مختلف مورد بررسی قرار می‌گیرد و اطلاعات به دست آمده، برای بهبود بخشیدن به یک فکر یا روش مورد استفاده قرار می‌گیرند. بهینه‌سازی ابزاری ریاضی است که برای یافتن پاسخ بسیاری از پرسش‌ها در خصوص چگونگی راه‌حل مسایل مختلف به کار می‌رود[۳،۲].

در بهینه‌سازی از یافتن بهترین جواب برای یک مسأله صحبت به میان می‌آید. لفظ بهترین به طور ضمنی بیان می‌کند که بیش از یک جواب برای مسأله‌ی مورد نظر وجود دارد که البته دارای ارزش یکسانی نیستند. تعریف بهترین جواب، به مسأله مورد بررسی، روش حل و همچنین میزان خطای مجاز وابسته است. بنابراین نحوه فرمول‌بندی مسأله نیز بر چگونگی تعریف بهترین جواب تأثیر مستقیم دارد. برخی از مسایل جواب‌های مشخصی دارند؛ بهترین بازیکن یک رشته‌ی ورزشی، طولانی‌ترین روز سال و پاسخ یک معادله‌ی دیفرانسیل معمولی درجه اول از مثال‌هایی هستند که می‌توان از آن‌ها به عنوان مسایل ساده نام برد. در مقابل، برخی از مسایل دارای جواب‌های بیشینه[۱] یا کمینه[۲]ی متعددی هستند که به نام نقاط بهینه یا اکسترمم[۳] شناخته می‌شوند، و به احتمال بهترین جواب یک مفهوم نسبی خواهد بود. بهترین اثر هنری، زیباترین منظره و گوش‌نوازترین قطعه‌ی موسیقی از مثال‌هایی هستند که می‌توان برای این‌گونه مسایل بیان کرد [۹،۸،۷،۶،۳،۲].

بهینه‌سازی، تغییر دادن ورودی‌ها و خصوصیات یک دستگاه، فرآیند ریاضی و یا آزمایش تجربی است به نحوی که بهترین خروجی یا نتیجه به دست بیاید شکل ۱. ورودی‌ها متغیرهای فرآیند یا تابع مورد بررسی هستند که به نام‌های تابع هدف[۴]، تابع هزینه[۵]، و یا تابع برازندگی[۶] نامیده می‌شود. خروجی نیز به صورت هزینه، سود و یا برازندگی تعریف می‌شود[۹،۸،۷،۶،۳،۲]. در این نوشتار نیز، مطابق با بسیاری از نوشتارهای مرتبط با موضوع، تمام مسایل بهینه‌سازی به صورت کمینه‌سازی مقدار یک تابع هزینه در نظر گرفته شده‌اند. به راحتی می‌توان نشان داد که هر نوع مسأله بهینه‌سازی را می‌توان در قالب یک مسأله کمینه‌سازی تعریف نمود.



شکل ۱- فرآیند یا تابعی که بهینه‌سازی می‌شود. در بهینه‌سازی ورودی‌ها یا متغیرها به نحوی تغییر داده می‌شوند که خروجی مطلوب به‌دست آید[۱].

[1] maximum
[2] minimum
[3] extremum
[4] objective function
[5] cost function
[6] fitness function

۲-۱- الگوریتم بهینه‌سازی ازدحام ذرات

هوش جمعی خاصیتی است سیستماتیک که در این سیستم، عامل‌ها[1] به طور محلی با هم همکاری می‌نمایند و رفتار جمعی تمام عامل‌ها باعث یک همگرایی در نقطه‌ای نزدیک به جواب بهینه سراسری می‌شود. نقطه قوت این الگوریتم عدم نیاز به یک کنترل سراسری می‌باشد. هر ذره[2](عامل) در این الگوریتم‌ها خودمختاری نسبی دارد که می‌تواند در سراسر فضای جواب‌ها حرکت کند و می‌بایست با سایر ذرات(عامل‌ها) همکاری داشته باشد. یکی از الگوریتم مشهور هوش جمعی بهینه‌سازی ازدحام ذرات می‌باشد که به طور گسترده در مسائل بهینه‌سازی برای پیدا کردن بهینه سراسری مورد استفاده قرار می‌گیرد.

روش بهینه‌سازی ازدحام ذرات یک روش سراسری کمینه‌سازی است که با استفاده از آن می‌توان با مسائلی که جواب آن‌ها یک نقطه یا سطح در فضای n بعدی می‌باشد، برخورد نمود. در این‌چنین فضایی، فرضیاتی مطرح می‌شود و یک سرعت ابتدایی به آن‌ها اختصاص داده می‌شود، همچنین کانال‌های ارتباطی بین ذرات در نظر گرفته می‌شود. سپس این ذرات در فضای پاسخ حرکت می‌کنند، و نتایج حاصله برمبنای یک «ملاک شایستگی» پس از هر بازه‌ی زمانی محاسبه می‌شود. با گذشت زمان، ذرات به سمت ذراتی که دارای ملاک شایستگی بالاتری هستند و در گروه ارتباطی یکسانی قرار دارند، شتاب می‌گیرند. علی‌رغم این‌که هر روش در محدوده‌ای از مسائل به خوبی کار می‌کند، این روش در حل مسائل بهینه‌سازی پیوسته موفقیت بسیاری از خود نشان داده است.

الگوریتم بهینه‌سازی ازدحام ذرات برای اولین بار در سال ۱۹۹۵ توسط راسِل اِبرهارت و جیمز کِندی به‌عنوان یک روش جستجوی غیرقطعی برای بهینه‌سازی تابعی مطرح گشت این الگوریتم از حرکت دسته جمعی پرندگانی که به دنبال غذا می‌باشند الهام گرفته شده است. گروهی از پرندگان در فضایی به صورت تصادفی دنبال غذا می‌گردند. تنها یک تکه غذا در فضای مورد بحث وجود دارد. هیچ یک از پرندگان محل غذا را نمی‌دانند. یکی از بهترین استراتژی‌ها می‌تواند دنبال کردن پرنده‌ای باشد که کم‌ترین فاصله را تا غذا داشته باشد. این استراتژی در واقع جان‌مایه الگوریتم است. هر راه‌حل که به آن یک ذره گفته می‌شود، در الگوریتم معادل یک پرنده در الگوی حرکت جمعی پرندگان می‌باشد. هر ذره یک مقدار شایستگی دارد که توسط یک تابع شایستگی محاسبه می‌شود. در واقع هر چقدر ذره در فضای جستجو به هدف (غذا در مدل حرکت پرندگان) نزدیک‌تر باشد، شایستگی بیشتری دارد. همچنین هر ذره دارای یک سرعت است که هدایت حرکت ذره را بر عهده دارد.

هر ذره با دنبال کردن ذرات بهینه در حالت فعلی، به حرکت خود در فضای مسئله ادامه می‌دهد. آغاز الگوریتم بهینه‌سازی ذرات به این شکل است که گروهی از ذرات به صورت تصادفی به وجود می‌آیند و با به‌روز کردن نسل‌ها سعی در یافتن راه‌حل بهینه می‌نمایند. در هر گام، هر ذره با استفاده از دو بهترین مقدار به‌روز می‌شود. اولین مورد، بهترین موقعیتی است که تاکنون ذره موفق به رسیدن به آن شده است که با نام pbest[3] موقعیت مذکور شناخته و نگهداری شده و توسط الگوریتم مورد استفاده قرار می‌گیرد و دومین مقدار بهترین موقعیتی است که تاکنون توسط جمعیت ذرات به دست آمده است. این موقعیت با gbest[4] نمایش داده می‌شود. در الگوریتم بهینه‌سازی ذرات هر ذره در فضای جستجو با بردار $x_i = (x_{i1}, x_{i2}, \dots, x_{iD})$ نمایش داده می‌شود که D ابعاد مسئله را نشان می‌دهد. همچنین این ذره دارای سرعت v

[1] Agents
[2] Particle
[3] Personal best
[4] Global best

است که با بردار $v_i = (v_{i1}, v_{i2}, \dots, v_{iD})$ نشان داده می‌شود. در هر تکرار پس از یافتن بهترین مقادیر (pbest و gbest)، بردار سرعت و مکان هر ذره با استفاده از معادلات (۱) و (۲) به‌روز می‌شود.

$$v = v \times \omega + C_1 \times r_1 \times (pbest - x) + C_2 \times r_2 \times (gbest - x) \quad (1)$$
$$x = x + v \quad (2)$$

این ذره دارای سرعت v است که با بردار $v_i = (v_{i1}, v_{i2}, \dots, v_{iD})$ نشان داده می‌شود. در هر تکرار پس از یافتن بهترین مقادیر (pbest و gbest)، بردار سرعت و مکان هر ذره با استفاده از معادلات (۱) و (۲) به‌روز می‌شود.

که در معادله (۱) سرعت آینده ذره به‌دست می‌آید. این معادله از سه عنصر اصلی تشکیل شده است: ۱. وزن اینرسی؛ یک وزن ورودی است. این وزن در واقع درصدی از سرعت قبلی ذره را در محاسبه سرعت جدید تأثیر می‌دهد. هرچه این مقدار بیشتر باشد جستجوی عمومی افزایش می‌یابد و هرچه این وزن کمتر باشد میزان جستجوی محلی افزایش می‌یابد. ۲. مؤلفه‌ی شناختی pbest. ۳. مؤلفه‌ی جمعی gbest؛ تفاوت این دو مؤلفه در سایز همسایگی است که برای هر ذره در نظر گرفته می‌شود. همچنین، C_1 و C_2 ثابت‌ها می‌باشند($C_1, C_2 > 0$) که C_1 برای بهترین تجربه شخصی در نظر گرفته شده و C_2 برای بهترین تجربه عمومی در نظر گرفته شده است. r_1 و r_2 اعدادی تصادفی هستند که به صورت نرمال در بازه $[0,1]$ تولید می‌شوند. در معادله (۲) وضعیت فعلی ذره با سرعت آینده جمع می‌شود و موقعیت آینده ذره به‌دست می‌آید.

روش بهینه‌سازی اجتماع ذرات ذاتا دارای سرعت همگرایی بالایی می‌باشد. از آنجایی که تمام ذرات در فضای جستجو به سمت بهترین موقعیتی که در طول مسیر حرکتشان یافته‌اند، شتاب می‌گیرند، در همان تکرارهای اولیه الگوریتم، ذراتی که در ابتدای کار در فضای جستجو پراکنده بوده‌اند، به سرعت به همدیگر و به نقطه بهینه‌ای که توسط الگوریتم یافته شده نزدیک می‌شوند و در نتیجه از سرعت و پراکندگی آن‌ها به شدت کاسته می‌شود. در مسائل بهینه‌سازی با بعد بالا و پیچیده، معمولا سرعت همگرایی بالا باعث همگرایی زودرس می‌شود که نتیجه آن این است که الگوریتم در یک نقطه بهینه محلی متوقف می‌شود. در چنین مواردی لازم است الگوریتم وقت بیشتری را صرف جستجوی نقطه بهینه کند. برای غلبه بر این مشکل می‌توان پراکندگی ذرات در فضای جستجو را کنترل کرد و اگر ذرات خیلی به‌هم نزدیک شدند، شروع به پراکنده کردن آن‌ها کرد. در این تحقیق از عملکرد جهش ژنتیکی برای رسیدن به هدف مورد نظر استفاده شده است.

۲-۲- الگوریتم ژنتیک

الگوریتم ژنتیک برای یافتن راه‌حل تقریبی برای بهینه‌سازی و مسایل جستجو استفاده می‌شود. یک روش مبتنی بر تکرار است. یک روش بهینه‌سازی الهام گرفته از طبیعت جاندار است که می‌توان در طبقه‌بندی‌ها از آن به عنوان یک روش عددی، جستجوی مستقیم و تصادفی یاد کرد. برای بهینه‌سازی، جستجو و یادگیری ماشین مورد استفاده قرار می‌گیرد. این الگوریتم با رشته‌های بیتی کار می‌کند که هرکدام از این رشته‌ها کل مجموعه متغیرها را نشان می‌دهد، برای راهنمایی جهت جستجو، انتخاب تصادفی انجام می‌دهد. در الگوریتم ژنتیک روش‌های جستجو براساس مکانیزم انتخاب و ژنتیک طبیعی عمل می‌نمایند. در هر تکرار چند نقطه از فضای جستجو را در نظر می‌گیرد. بنابراین شانس این‌که به یک ماکزیمم محلی همگرا شود کاهش می‌یابد و فقط نیاز به اطلاعاتی در مورد کیفیت حل‌های ایجاد شده به وسیله هر مجموعه از متغیرها دارد. پس قابل انعطاف‌تر است. الگوریتم مذکور از قوانین احتمالی پیروی می‌کند نه از قوانین قطعی. فضای جواب را به طور همه‌جانبه جستجو می‌کند بنابراین امکان کمتری برای همگرایی به یک نقطه بهینه محل وجود خواهد داشت.

الگوریتم ژنتیک در هر تکرار هر یک از رشته‌های موجود در جمعیت رشته‌ها، رمزگشایی شده و مقدار تابع هدف برای آن به‌دست می‌آید. براساس مقادیر به دست آمده تابع هدف در جمعیت رشته‌ها، به هر رشته یک عدد برازندگی نسبت داده می‌شود. این عدد برازندگی احتمال انتخاب را برای هر رشته تعیین خواهد کرد. براساس این احتمال انتخاب مجموعه‌ای از رشته‌ها انتخاب شده و با اعمال عمگرهای تکاملی ژنتیکی روی آن‌ها رشته‌های جدید جایگزین رشته‌هایی از جمعیت اولیه می‌شوند تا تعداد جمعیت رشته‌ها در تکرارهای محاسباتی مختلف ثابت باشد. بدین طریق در هر مرحله تکرار، یک نسل جدید تولید می‌شود که با توجه به اصلاحاتی که در آن صورت پذیرفته است روبه سوی تکامل خواهد داشت[۲۵،۲۱،۱۹،۱۵،۱۱] .

۳- الگوریتم پیشنهادی

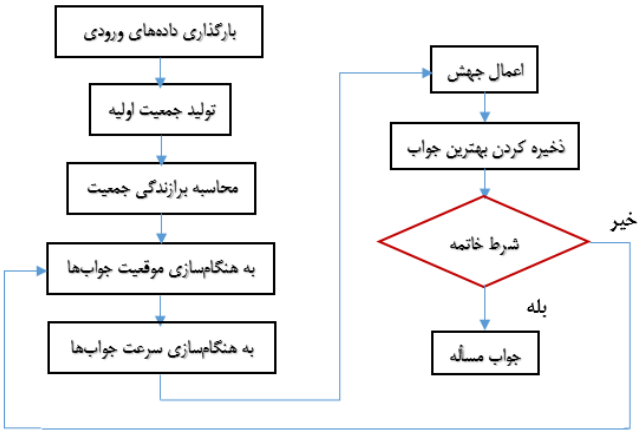
یکی از مهم‌ترین قسمت‌های یک سرور رایانش ابری که می‌توان آن را بهبود داد و بهبود آن تأثیر بسیاری بر افزایش راندمان و افزایش کیفیت سرویس‌ها دارد، الگوریتم و روش تخصیص منابع به درخواست‌ها می‌باشد. ساختار و معماری سرور رایانش ابری به این صورت است که هر سرور به چند میزبان[1] تقسیم شده و هر میزبان نیز از چند ماشین مجازی تشکیل شده است. نحوه تخصیص منابع به این صورت است که پس از دسته‌بندی و آماده کردن درخواست‌ها برمبنای یک الگوریتم مشخص کارها[2] را به ماشین مجازی مشخص شده توسط الگوریتم می‌فرستند و در واقع منابعی که در آن ماشین مجازی قرار دارد را به آن کار اختصاص می‌دهند. همچنین این نکته را نیز باید در نظر گرفت که هر ماشین مجازی با توجه به توان و منابع خود می‌تواند چند کار را به صورت همزمان میزبانی کرده و انجام دهد. وقتی کاربر یک سرویس خاص را از سرور ابر درخواست می‌کند مراحل ارائه آن سرویس توسط سرور بررسی شده و به چندین کار وابسته یا آن سرویس توسط سرور بررسی شده غیروابسته تجزیه می‌شود و برای انجام آماده می‌شوند. در این تحقیق یک الگوریتم برای مدیریت منابع و تخصیص آن‌ها به کارها در درخواست‌ها یا به عبارتی یک الگوریتم برای تخصیص ماشین‌های مجازی به وظایف[3] ارائه شده که در واقع یک مدل بهینه از الگوریتم ازدحام ذرات می باشند. در واقع هدف نهایی این راه‌حل پیشنهادی به دست آوردن یک الگوی مناسب جهت نگاشت کار به ماشین مجازی می‌باشد که بتواند بدون افزایش توان مصرفی زمان پاسخ و زمان تکمیل آخرین کار را به حداقل برساند.

زمان‌بندی و مدیریت منابع در محیط‌های توزیع شده مانند سرورهای رایانش ابری به پارامترهای زیادی بستگی دارد، زیرا در این سیستم‌ها، منابع و درخواست‌های متفاوت، پویا و گاها وابسته به‌هم وجود دارند و همچنین حجم‌کاری سیستم نسبت به سایر محیط‌ها بیشتر می‌باشد. با توجه به این‌که مهم‌ترین پارامترها در سرورهای رایانش ابری پارامترهایی هستند که بر زمان تکمیل درخواست‌ها و توزیع متناسب بار مبتنی می‌باشند، ما در این تحقیق روی روشی کار می‌کنیم که بتواند پارامترهای زمان پاسخ[4] و زمان تکمیل آخرین کار[5] که در واقع نشان‌دهنده‌ی توزیع متناسب بار می‌باشد را نسبت به الگوریتم‌های ارائه شده قبلی کاهش دهد. برای انجام این کار ما از یک الگوریتم ترکیبی مبتنی بر الگوریتم ازدحام ذرات و استراتژی جهش الگوریتم ژنتیک استفاده می‌کنیم.

[1] Host
[2] Tasks
[3] tasks
[4] Response Time
[5] Makespan

ایده اصلی الگوریتم پیشنهادی به این صورت است که؛ در یک مرحله جواب‌های الگوریتم بهینه‌سازی ازدحام ذرات برای جهش کردن جواب‌های الگوریتم ژنتیک از جواب‌های بهینه محلی به کار برده می‌شود. ژنتیک در هر مرحله یک جهش دارد. برای این‌که این جهش‌ها به گونه‌ای انجام شوند که ما به جواب مطلوب برسیم از جواب‌های الگوریتم بهینه‌سازی ازدحام ذرات استفاده کرده‌ایم و در مرحله دیگر جواب‌های الگوریتم ژنتیک برای وسیع کردن دامنه جستجوی الگوریتم بهینه‌سازی ازدحام ذرات به کار برده می‌شود. در این روش پیشنهادی که ترکیب دو الگوریتم بهینه‌سازی ازدحام ذرات و ژنتیک می‌باشد به تعدادی مشخص مراحل آن‌ها تکرار می‌شود تا به نتیجه بهینه برسد. با این کار از همگرایی زودرس الگوریتم بهینه‌سازی ازدحام ذرات جلوگیری شده و سبب می‌شود که تمام فضای مسئله به صورت کامل‌تر جستجو شود. این کار دقت الگوریتم بهینه‌سازی ازدحام ذرات و بهینگی آن را نیز افزایش خواهد داد. روند کار بدین صورت است که دو الگوریتم؛ بهینه‌سازی ازدحام ذرات و ژنتیک با هم همزمان و به طور موازی اجرا می‌شوند و در حین اجرا شدن با هم در حال تبادل اطلاعات هستند، به طوری‌که هر کدام از این الگوریتم‌ها در هر زمانی که به جواب بهینه‌تری برسد آن جواب را در اختیار الگوریتم دیگر قرار می‌دهد و این الگوریتم جواب به دست آمده را به کار می‌گیرد و به اجرا ادامه می‌دهد. حال ممکن است این الگوریتم با به کار بردن این نتیجه در ادامه به جواب بهینه‌تری برسد، در این هنگام این الگوریتم جواب بهینه به دست آمده را در اختیار دیگری قرار می‌دهد، همین‌طور به تبادل اطلاعات با یکدیگر می‌پردازند تا به نتیجه مطلوب برسند. برای مثال در روش پیشنهادی ما در حین اجرا، الگوریتم بهینه‌سازی ازدحام ذرات جواب بهینه‌تری را می‌دهد. در این لحظه این الگوریتم جواب بهینه خود را در اختیار الگوریتم ژنتیک قرار می‌دهد و الگوریتم ژنتیک این جواب را گرفته و به کار خود ادامه می‌دهد. حال ممکن است در ادامه اجرای الگوریتم ژنتیک (با به کار بردن جواب‌های الگوریتم بهینه‌سازی ازدحام ذرات) به جواب‌های بهینه‌تری دست پیدا کند پس باید این جواب‌های بهینه را در اختیار الگوریتم بهینه‌سازی ازدحام ذرات قرار دهد. به همین صورت این دو الگوریتم با یکدیگر به تبادل نتایج می‌پردازند تا به نتیجه بهینه برسند. جواب‌های بهینه براساس معیارهای از پیش تعیین شده به دست می‌آید، الگوریتم بهینه‌سازی ازدحام ذرات و ژنتیک هم مانند الگوریتم‌های تکاملی دیگر یک تابع برازندگی دارند. در این الگوریتم‌ها یک تابع برازندگی براساس یک سری معیارها تعریف می‌شود که این تابع برازندگی جواب بهینه الگوریتم را مشخص می‌کند.

در شکل ۲ فلوچارت روش پیشنهادی برای زمان‌بندی در ابر آورده شده است. تفاوت اصلی الگوریتم بهینه‌سازی ذرات استاندارد با الگوریتم بهینه‌سازی ذرات بهبودیافته در عملگر جهش آن است. در الگوریتم بهینه‌سازی ذرات بهبودیافته در هر تکرار، تعدادی از اعضای جمعیت به تصادف انتخاب شده و عملگر جهش بر روی این اعضا انجام می‌پذیرد. همان‌طور که در این شکل دیده می‌شود نقطه تفاوت الگوریتم پیشنهادی با الگوریتم بهینه‌سازی ذرات استاندارد در این عملگر جهش است. در واقع با استفاده از این عملگر امکان ایجاد تنوع در جمعیت و فرار از بهینه محلی در الگوریتم بهینه‌سازی ذرات ایجاد شده است.



شکل ۲- فلوچارت روش پیشنهادی

شبه کد روش پیشنهادی در جدول ۱ آورده شده است.

جدول ۱- شبه کد الگوریتم پیشنهادی

```
Algorithm2. Pseudocode of PSO & GA
Begin Algorithm
  For each particle
    Initialize particle
  END
  Do
    For each particle
      Calculate fitness value
      If the fitness value is better than the best fitness value (pbest) in history
        set current value as the new pbest
    End

    Choose the particle with the best fitness value of all the particles as the gbest
    For each particle
      Calculate particle velocity according equation (1)
      Update particle position according equation (2)
      pbest particles are change their positions by mutation operator of GA
    End
  While maximum iterations or minimum error criteria is not attained
End Algorithm
```

۳-۱- فرضیات

برای پیاده‌سازی این الگوریتم یک‌سری فرضیات در نظر گرفته شده است که در جداول زیر آورده شده است. شرایط کارها در جدول ۲ آورده شده‌اند:

جدول ۲- مشخصات کارها

مهلت زمانی[1]	نوع کار	زمان مورد نیاز برای تکمیل	منبع مورد نیاز
ندارند	مستقل	۰٫۵-۱۰۰	پردازنده

[1] deadline

همان‌طور که در جدول ۱ مشخص شده است نوع کارها پردازش بوده و به‌هم وابسته نیستند، همچنین شرایط ماشین‌های مجازی در جدول ۳ آورده شده است.

جدول ۳- مشخصات ماشین های مجازی

قابلیت میزبانی بیش از یک کار(task)	تعداد ماشین‌مجازی	ظرفیت ماشین‌مجازی	نوع منبع ماشین‌مجازی
دارد	۱۰-۵۰	۱۰-۱۰۰	پردازنده

۳-۲- نحوه محاسبه تابع برازندگی

برای تشخیص این‌که یک راه‌حل، تا چه اندازه می‌تواند پاسخ مناسبی ارائه دهد می‌بایست ارزش آن توسط یک تابع برازندگی سنجیده شود، در این پژوهش کاهش زمان makespan به‌عنوان تابع برازندگی در نظر گرفته شده است:

fitness = minimize(cost.makespan) (3)

۴- نتایج شبیه‌سازی و تحلیل آن‌ها

الگوریتم‌های تکاملی ماهیتی تصادفی دارند و رسیدن به جواب نهایی به مقدار زیادی به جواب‌های اولیه وابسته است.

۴-۱- تغییر در پارامترهای اولیه الگوریتم پیشنهادی

از آنجا که الگوریتم‌های تکاملی ماهیتی تصادفی دارند و جواب نهایی آن‌ها به پارامترهای اولیه الگوریتم وابسته است بنابراین در این بخش با تغییر پارامترهای اولیه میزان زمان تکمیل آخرین کار محاسبه خواهد شد.

۴-۱-۱- تغییر در تعداد تکرارهای برنامه

در این بخش تعداد تکرارهای برنامه تغییر خواهد کرد اما سایر پارامترهای برنامه ثابت نگه داشته می‌شوند که در جدول ۴ مقادیر این پارامترها آورده شده است.

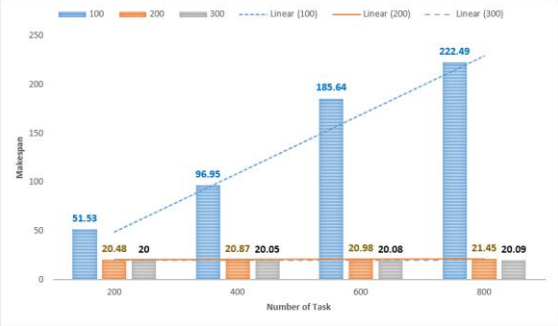
جدول ۴- پارامترهای الگوریتم پیشنهادی

جمعیت اولیه	۱۰۰
W	۱
C1	۲
C2	۲
تکرارهای برنامه	متغیر

جدول ۵- نتایج برنامه برای تعداد تکرارهای مختلف

تعداد وظایف / تعداد تکرار	۲۰۰	۴۰۰	۶۰۰	۸۰۰
۱۰۰	۵۱,۵۳	۹۶,۹۵	۱۸۵,۶۴	۲۲۲,۴۹
۲۰۰	۲۰,۴۸	۲۰,۸۷	۲۰,۹۸	۲۱,۴۵
۳۰۰	۲۰	۲۰,۰۵	۲۰,۰۸	۲۰,۰۹

همان‌طور که در جدول بالا مشاهده می‌شود با تغییر در تکرار برنامه الگوریتم جواب‌های بهتری را ارائه می‌دهد در نمودار ۱ مقایسه بین تکرارهای مختلف آورده شده است.



نمودار ۱- نتایج برنامه برای تکرارهای مختلف

۴-۱-۲- تغییر در جمعیت اولیه

در این بخش جمعیت اولیه تغییر خواهد کرد اما سایر پارامترهای برنامه ثابت نگه داشته می‌شوند که در جدول ۶ مقادیر این پارامترها آورده شده است.

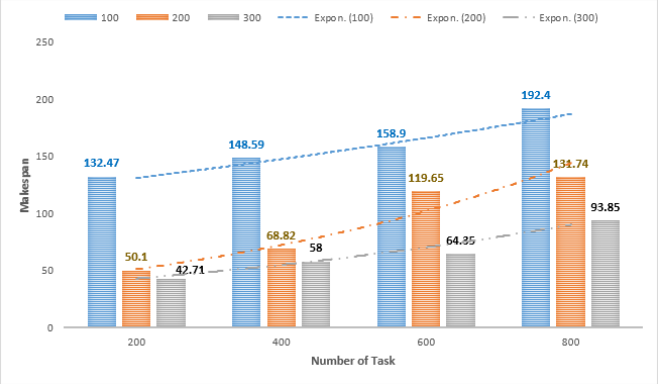
جدول ۶- پارامترهای الگوریتم پیشنهادی

متغیر	جمعیت اولیه
۱	W
۲	C1
۲	C2
۱۰۰	تکرارهای برنامه

جدول ۷- نتایج برنامه برای جمعیت‌های مختلف

تعداد وظایف / جمعیت اولیه	۲۰۰	۴۰۰	۶۰۰	۸۰۰
۱۰۰	۱۳۲,۴۷	۱۴۸,۵۹	۱۵۸,۹۰	۱۹۲,۴۰
۲۰۰	۵۰,۱۰	۶۸,۸۲	۱۱۹,۶۵	۱۳۱,۷۴
۳۰۰	۴۲,۷۱	۵۸	۶۴,۳۵	۹۳,۸۵

همان‌طور که در جدول بالا مشاهده می‌شود با تغییر در جمعیت اولیه، الگوریتم جواب‌های بهتری را ارائه می‌دهد اما این جواب‌ها در مقایسه با تغییر دادن تعداد تکرارهای برنامه به مراتب جواب‌های بدتری را به خروجی می‌برد، در نمودار ۲ مقایسه بین جمعیت‌های مختلف آورده شده است.



نمودار ۲- نتایج برنامه برای جمعیت‌های مختلف

۴-۱-۳- تغییر در مقادیر C1 و C2:

در این بخش مقادیر C1 و C2 تغییر خواهند کرد اما سایر پارامترهای برنامه ثابت نگه داشته می‌شوند که در جدول ۸ مقادیر این پارامترها آورده شده است.

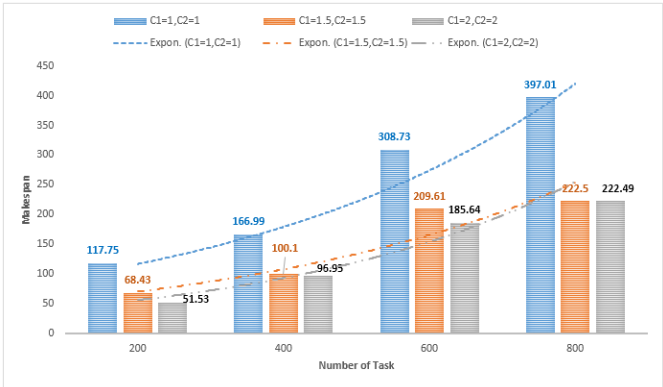
جدول ۸- پارامترهای الگوریتم پیشنهادی

جمعیت اولیه	۱۰۰
W	۱
C1	متغیر
C2	متغیر
تکرارهای برنامه	۱۰۰

جدول ۹- نتایج برنامه برای مقادیر مختلف C1 و C2

تعداد وظایف / جمعیت اولیه	۲۰۰	۴۰۰	۶۰۰	۸۰۰
C1=1&C2=1	۱۱۷,۷۵	۱۶۶,۹۹	۳۰۸,۷۳	۳۹۷,۰۱
C1=1.5&C2=1.5	۶۸,۴۳	۱۰۰,۱۰	۲۰۹,۶۱	۲۲۲,۵۰
C1=2&C2=2	۵۱,۵۳	۹۶,۹۵	۱۸۵,۶۴	۲۲۲,۴۹

همان‌طور که در جدول بالا مشاهده می‌شود با تغییر در مقادیر C1 و C2 الگوریتم جواب‌های مختلفی را ارائه می‌دهد که برای مقادیر C1=2 و C2=2 الگوریتم پیشنهادی جواب‌های بهتری را به خروجی می‌برد، در نمودار ۳ مقایسه بین مقادیر مختلف C1 و C2 آورده شده است.



نمودار ۳- نتایج برنامه برای مقادیر مختلف C1 و C2

۴-۱-۴- نتیجه گیری این بخش:

همان‌طور که از جداول و شکل‌های بالا مشخص است الگوریتم پیشنهادی به میزان بسیار زیادی به پارامترهای اولیه وابسته است که از میان پارامترهای اولیه الگوریتم میزان تکرار الگوریتم به مراتب نتایج بهتری را به خروجی می‌برد، با توجه به جدول ۵ اگر میزان تکرارها را ۳۰۰ در نظر بگیرم الگوریتم تقریبا همگرا شده و تغییر در تعداد کارها هیچ تأثیری بر میزان همگرایی نخواهد داشت بنابراین می‌توان نتیجه گرفت که روش پیشنهادی با تعداد تکرار ۳۰۰ بهترین جواب که همان حداقل میزان makespan است را به ما می‌دهد، در ادامه این الگوریتم با الگوریتم‌های دیگر در زمینه زمان‌بندی کارها مقایسه خواهد شد.

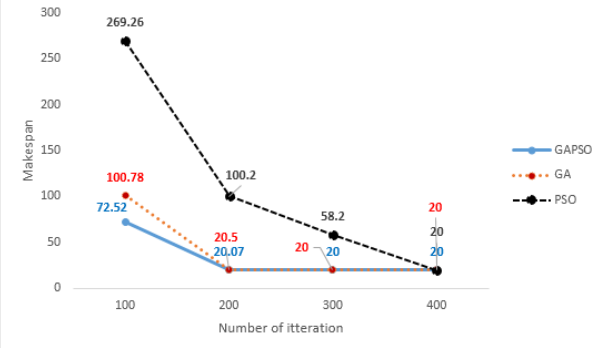
۴-۲- مقایسه روش پیشنهادی با الگوریتم‌های دیگر

در این بخش روش پیشنهادی با الگوریتم‌های دیگر در زمینه زمان‌بندی وظایف در محاسبات ابری آورده خواهد شد.

۴-۲-۱- مقایسه زمان تکمیل آخرین کار

در این بخش نتایج مقایسه زمان تکمیل آخرین کار الگوریتم پیشنهادی با الگوریتم‌های ژنتیک و ازدحام ذرات استاندارد برای تکرارهای مختلف آورده خواهد شد. لازم به ذکر است که در این بخش آزمایش در دو مرحله اجرا شده است:

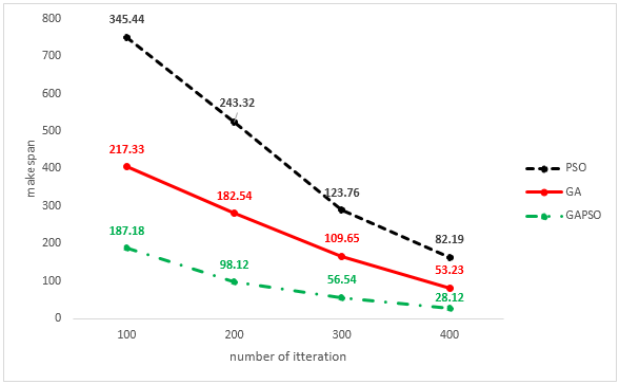
- حالت اول: تعداد وظایف ۱۰۰ وظیفه است. در این حالت نتایج آزمایش نشان داد که الگوریتم پیشنهادی و االگوریتم ژنتیک هر دو در ۳۰۰ امین تکرار با هم برابرند اما زمان پاسخ در روش پیشنهادی کمتر است. نتیجه این آزمایش در نمودار ۴ آورده شده است.



نمودار ۴- مقایسه روش پیشنهادی با الگوریتم‌های مختلف (برای ۱۰۰ وظیفه)

دومین کنفرانس بین‌المللی ترکیبیات، رمزنگاری و محاسبات
Homepage: http://i4c.iust.ac.ir

- حالت دوم: تعداد وظایف ۱۰۰۰ وظیفه است. نتایج آزمایش نشان داد که روش پیشنهادی نسبت به بقیه روش‌های مقایسه شده هم در تعداد تکرار کم‌تر جواب بهینه به دست می‌آورد و هم زمان پاسخ کم‌تر شده است. نتیجه این آزمایش در نمودار ۵ آورده شده است.



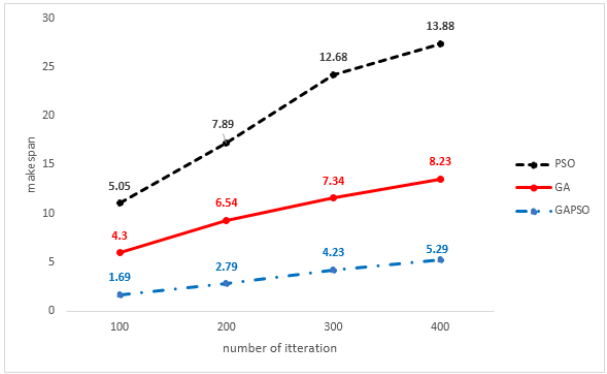
نمودار ۵- مقایسه روش پیشنهادی با الگوریتم‌های مختلف (برای ۱۰۰۰ وظیفه)

با استفاده از نتایج آزمایش می‌توان نتیجه گرفت که روش پیشنهادی زمانی که وظایف زیاد باشد در یافتن نتیجه بهینه بهتر عمل می‌کند.

۲-۲- مقایسه زمان اجرای برنامه

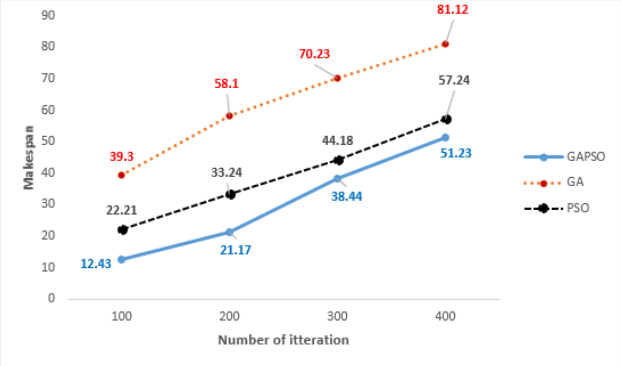
در این بخش نتایج مقایسه زمان اجرای الگوریتم پیشنهادی با الگوریتم‌های ژنتیک و ازدحام ذرات استاندارد برای تکرارهای مختلف آورده خواهد شد. لازم به ذکر است که در این بخش نیز آزمایش در دو مرحله اجرا شده است:

- حالت اول: در این حالت تعداد وظایف ۱۰۰ در نظر گرفته شده است. همان‌طور که از نمودار ۶ مشخص است الگوریتم پیشنهادی به مراتب از سایر الگوریتم‌ها نتایج بهتری را نشان می‌دهد.



نمودار ۶- مقایسه زمان روش پیشنهادی با الگوریتم‌های مختلف (برای ۱۰۰وظیفه)

- حالت دوم: در این حالت تعداد وظایف ۱۰۰۰ در نظر گرفته شده است. همان‌طور که از نمودار ۷ مشخص است الگوریتم پیشنهادی به مراتب از سایر الگوریتم‌ها نتایج بهتری را نشان می‌دهد.



نمودار ۷- مقایسه زمان روش پیشنهادی با الگوریتم‌های مختلف (برای ۱۰۰۰وظیفه)

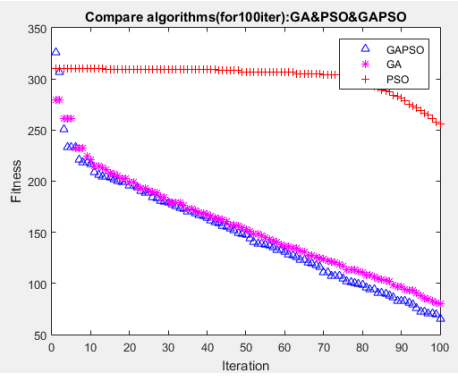
با استفاده از نتایج آزمایش می‌توان نتیجه گرفت که زمان اجرای الگوریتم پیشنهادی نسبت به الگوریتم‌های مورد مقایسه در هر دو حالت کم‌تر می‌باشد.

۴-۲-۳- نمودار همگرایی

در این بخش نمودار همگرایی روش پیشنهادی در مقایسه با الگوریتم‌های دیگر با تعداد تکرارهای مختلف آورده خواهد شد.

۴-۲-۳-۱- تعداد تکرارها ۱۰۰

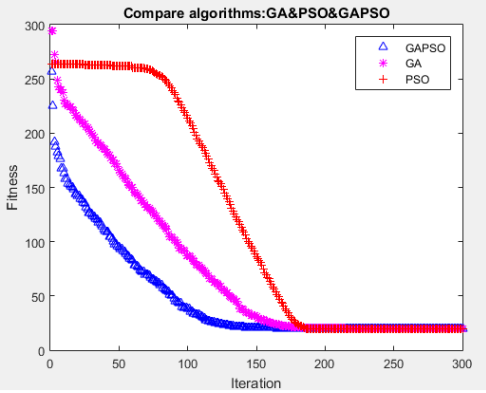
همان‌طور که از نمودار ۸ مشخص است روش پیشنهادی با تعداد تکرار ۱۰۰، روند همگرایی بسیار بهتری نسبت به دو الگوریتم دیگر دارد.



نمودار ۸- مقایسه روند همگرایی روش پیشنهادی با دو الگوریتم GA و PSO

۴-۲-۳-۲- تعداد تکرارها ۳۰۰

همان‌طور که از نمودار ۹ مشخص است در الگوریتم ژنتیک تقریبا با ۱۷۰بار تکرار، زمان پاسخ به ۲۰ رسیده و الگوریتم همگرا شده است. الگوریتم بهینه‌سازی ازدحام ذرات بدترین جواب را داده است. درحالی‌که در روش پیشنهادی تقریبا با ۱۵۰ بار تکرار، زمان پاسخ به ۲۰ رسیده که بهترین جواب بهینه با تعداد تکرار کم‌تر حاصل شده است.



نمودار ۹- مقایسه روند همگرایی روش پیشنهادی با دو الگوریتم GA و PSO

۴-۲-۴- نتیجه‌گیری این بخش:

همان‌طور که از نمودارها و جداول این بخش مشخص است هرچه زمان پاسخ کوته‌تر باشد جواب بهینه‌تر خواهد بود و در نتیجه تعادل بار کاری بهتری روی ماشین‌های مجازی در محیط رایانش ابری برقرار خواهد شد. در الگوریتم پیشنهادی زمان اجرا، زمان تکمیل آخرین کار و همچنین روند همگرایی بسیار بهتری نسبت به الگوریتم‌های دیگر حاصل شده است.

۴-۳- نتایج شبیه‌سازی

نتایج شبیه‌سازی نشان داد که الگوریتم پیشنهادی به میزان بسیار زیادی به پارامترهای اولیه وابسته است که از میان پارامترهای اولیه الگوریتم، میزان تکرار الگوریتم به مراتب نتایج بهتری را به خروجی می‌برد. در واقع روش پیشنهادی با تعداد تکرار ۳۰۰، بهترین جواب که همان حداقل میزان makespan است را به ما می‌دهد. کاهش makespan در بی‌کار نماندن ماشین‌های مجازی و برقراری تعادل بارکاری بیشتر در محیط ابر مؤثر است. همچنین الگوریتم پیشنهادی با الگوریتم‌های دیگر در زمینه زمان‌بندی کارها مقایسه شد که نتایج نشان داد الگوریتم پیشنهادی زمان اجرا، زمان تکمیل

آخرین کار و روند همگرایی بسیار بهتری نسبت به الگوریتم‌های دیگر دارد. این بدان معنی است که روش پیشنهادی از دو روش دیگر مؤثرتر و دارای کارایی بالاتری است.

۵- نتیجه‌گیری

با توجه به گستردگی و پویایی مراکز داده رایانش ابری و این‌که این مراکز با بارکاری بسیار متنوع و متغیری روبه‌رو هستند، روش‌هایی که برای استقرار ماشین‌های مجازی به‌کار می‌رود باید مقیاس‌پذیر، پویا و سریع باشد و بتواند با چنین بارکاری به صورت بهینه رفتار کند و با افزایش حجم و تنوع درخواست‌ها بازدهی آن افت ننماید. در این تحقیق، عملگرهای ژنتیک و بهینه‌سازی ازدحام ذرات مورد تحلیل قرار گرفت و یک روش جدید با استفاده از ترکیب الگوریتم بهینه‌سازی ازدحام ذرات و الگوریتم ژنتیک برای حل مسأله تعادل بار روی ماشین‌های مجازی در محیط رایانش ابری ارائه شده است. هدف از به‌کارگیری الگوریتم PSOGA بهبود تعادل بار روی ماشین‌های مجازی در محیط رایانش ابری بوده است. نتایج تجربی نشان داد الگوریتم پیشنهادی در حل مسأله تعادل بارکاری در محیط رایانش ابری از کارایی بالایی برخوردار است که نه تنها سرعت همگرایی PSO را افزایش داد، بلکه همیشه از افتادن در دام بهینگی محلی اجتناب می‌کرد و توانسته یک توازن بار بین معیارهای کاربران به‌وجود آورد. برای مقایسه از دو روش GA و PSO استفاده گردیده است. نتایج ارزیابی نشان داد که روش پیشنهادی در اثربخشی و بهره‌وری از سایر روش‌ها بهتر عمل می‌کند. پیشنهاد می‌شود جهت عملکرد بهتر تعادل بار در رایانش ابری، از الگوریتم‌هایی استفاده شود که به ظرفیت و بار منابع توجه کنند تا بتواند با بارکاری متنوع و متغیر به صورت بهینه رفتار کند و با افزایش حجم و تنوع درخواست‌ها بازدهی آن افت ننماید.

تقدیر و تشکر

در اینجا از آقایان دکتر نیما جعفرزاده و دکتر مقداد میرابی که در این تحقیق بنده را یاری کردند، تشکر و قدردانی می‌نمایم.

مراجع

منابع فارسی

[۱]. میرمهدی سید اسفهلان، "کاربرد الگوریتم های تکاملی برای تحلیل مسائل آنتن و ماکروویو"، سمینار کارشناسی ارشد، دانشگاه علم و صنعت،دانشکده مهندسی برق،گروه مهندسی مخابرات، ۱۳۸۷.

[۲]. باقر زارعی، "پردازش تکاملی"، جزوه آموزشی، دانشگاه آزاد اسلامی واحد شبستر،دانشکده کامپیوتر، ۱۳۸۷.

[۳]. ناصر لطفی، "زمان‌بندی گراف وظایف برای پردازش موازی مبتنی بر پردازش تکاملی"، پایان نامه کارشناسی ارشد، دانشگاه آزاد اسلامی واحد نجف آباد،دانشکده کامپیوتر، ۱۳۸۴.

[۴]. نوید فرخی، "مدل‌سازی و شبیه‌سازی رایانش ابری"، انتشارات؛ بابل: دانش‌بنیان، علوم رایانه، ۱۳۹۵، ۲۳۲ص

منابع انگلیسی

[5]. Abdul Razaque, Nikhileshwara Reddy Vennapusa, Nisargkumar Soni, Guna Sree Janapati, khilesh Reddy Vangala, “Task scheduling in Cloud computing”. Long Island Systems, Applications and Technology Conference (LISAT), 2016 IEEE.
[6]. Yavuz Eren, İbrahim B. Küçükdemiral, İlker Üstoğlu. “Introduction to Optimization”. Optimization in Renewable Energy Systems, 2017, Pages 27-74.
[7]. Jesper Christensen, Christophe Bastien, “Introduction to General Optimization Principles and Methods”. Nonlinear Optimization of Vehicle Safety Structures, 2016, Pages 107-168
[8]. S. B. L. Vandenberghe, "Convex Optimization", Cambridge University Press, 2004.
[9]. Jasbir Singh Arora, “Introduction to Design Optimization”. Introduction to Optimum Design (Fourth edition), 2017, Pages 3-18
[10]. Anthony T. Velte, Toby J. Velte and Robert Elsenpeter, "Cloud Computing: A Practical Approach". McGraw-Hill Companies, pp. 91-110, 2010.
[11]. Clavister. (2007), "Security in the Cloud," http:/www.it-wire.nu/, posted at: 2007/01/15, Accessed, 2011.
[12]. Hai Zhong, Kun Tao, Xuejie Zhang, "An Approach to Optimized Resource Scheduling Algorithm for Open-Source Cloud Systems". In IEEE transactions on evolutionary computation, 2010, 8(3), 256-279.
[13]. Padhy R. P, Rao G. P. (2011). Load Balancing in cloud computing Systems. Bachelor Thesis .
[14]. Rajwinder Kaur and Pawan Luthra, (2014), ” Load Balancing in Cloud Computing”, Proc. of Int. Conf. on Recent Trends in Information, Telecommunication and Computing, ITC
[15]. Caldas, L. G, “A Generative Design System for Low-Energy Architecture Design,” Departamento de Engenharia Civil e Arquitecture, Instituto Superior Tecnico, Lisboa, Portugal.
[16]. Sahu, Yatendra and Pateriya, RK, “Cloud Computing Overview with Load Balancing Techniques”,International Journal of Computer Applications,2013,vol.65, Sahu2013
[17]. Chaudhari, Anand and Kapadia, Anushka, “ Load Balancing Algorithm for Azure Virtualization with Specialized VM”, 2013,algorithms,vol 1,pages 2, Chaudhari
[18]. Nayandeep Sran,Navdeep Kaur , “Comparative Analysis of Existing Load Balancing Techniques in Cloud Computing ”,vol 2,jan 2013
[19]. Grossman R. L, Y.Gu, M.Sabala, W.Zhang. (2009), "Compute and Storage Clouds Using Wide Area Hight Performance Networks," Future Generation computer Systems, Volume 25, Issue 2 , February.
[20]. Chaczko, Zenon and Mahadevan, Venkatesh and Aslanzadeh, Shahrzad and Mcdermid, Christopher, “Availability and load balancing in cloud computing”, International Conference on Computer and Software Modeling, Singapore, chaczko2011availability

[21]. Shariat Panahi, M, "An Introduction to Genetic Algorithms," Department of Mechanical Engineering, University of Alberta, 1995.
[22]. R K Jena, "Multi objective Task Scheduling in Cloud Environment Using Nested PSO Framework". Procedia Computer Science, Volume 57, 2015, Pages 1219-1227
[23]. Amandeep Verma, Sakshi Kaushal, "A hybrid multi-objective Particle Swarm Optimization for scientific workflow scheduling". Parallel Computing, Volume 62, February 2017, Pages 1-19
[24]. Reena Pan war, Prof. Dr. Bhawna Mallick, (2015), "Load Balancing in Cloud Computing Using Dynamic Load Management Algorithm".
[25]. P.Zech. (2011), "Risk-Based Security Testing in Cloud Computing Environments" , Fourth IEEE International Conference on Software Testing, Berlin, March, PP, 411-414.
[26]. Foram F Kherani, Prof. Jignesh Vania, "Load Balancing in cloud computing", International Journal of Engineering Development and Research, Volume 2, Issue 1 \ISSN: 2321-9939.
[27]. Sidhu, A, K. and Kinger, S. (2013). Analysis of Load Balancing Techniques in Cloud Computing. International Journal of Computers and Technology.
[28]. kansal.N.J and CHANA.I cloud load balancing techniques : Astep towards Green computing , IJCSI international journal of Computer Science Issues , vol. 9 Issue 1 , No 1, January 2012.
[29]. G. Jones, "Genetic and Evolutionary Algorithms", University of Sheffield, UK
[30]. J. H Holland, 'Adaptation in Natural and Artificial Systems', MT Press, Cambridge, MA, 1992.
[31]. D. E. Goldberg, 'Genetic Algorithms in Search Optimization and Machine Learning', Addison-Wesley, Reading, MA, 1989.