یازدهمین کنگره ملی سراسری
فناوریهای نوین در حوزه توسعه پایدار ایران
11th National Congress of
the New Technologies in Sustainable Development of Iran
senaconf.ir

بررسی عملکرد الگوریتم‌های تعادل بار بر اساس پارامترهای کیفی

امیررضا شریفی[1]

[1] دانشجوی کارشناسی ارشد شبکه های کامپیوتری، amirreza.sharifi93@gmail.com

چکیده

در یک محیط پردازش توزیع‌شده، مانند Grid یا محیط پردازش ابری وظایف ممکن است به شکل کاملاً تصادفی به گره‌های مختلف اختصاص داده شوند. همین امر می‌تواند موجب توزیع به‌صورت نامتوازن بار پردازشی بین گره‌ها شده و برخی گره‌ها را به‌شدت درگیر کند درحالی‌که برخی دیگر نیز بی‌کار بمانند. لذا برای جلوگیری از این امر الگوریتم‌ها و روش‌های تعادل بار طراحی شدند. در خلال سال‌های گذشته الگوریتم‌های مختلفی ابداع‌شده‌اند. در این مقاله سعی داریم شما را با این الگوریتم‌ها آشنا کرده و آن‌ها را با یکدیگر مقایسه کنیم.

واژه‌های کلیدی

الگوریتم ها، توازن بار پویا، گره ها، پارامترهای کیفی توازن بار، سیستم های توزیع شده

۱- مقدمه

یکی از بزرگترین مشکلات در شبکه‌های توزیع‌شده بزرگ مسئله توزیع بار به شکلی کارآمد است. به‌طوری‌که سیستمی پایدار با کمترین زمان پاسخگوئی و بیشترین استفاده از منابع را داشته باشیم.

ممکن است دریک سیستم چندین پردازش معطل درصف داشته باشیم درحالی‌که درهمان زمان پردازنده‌های بی‌کار داشته باشیم. این اتفاق به دلیل ناکارآمدی الگوریتم انتخاب‌شده برای سیستم رخ می‌دهد. پس باید انواع الگوریتم‌های توازن بار را بشناسیم و با توجه به محیط کاری مناسب‌ترین مورد را انتخاب کنیم. در این مقاله قصد داریم شما را با مفهوم توازن بار و انواع الگوریتم‌های موجود آشنا کرده و درنهایت بین آنها مقایسه‌ای به عمل‌آوریم.

۲- توازن بار چیست؟

Lood balance یا توازن بار عملی است که طی آن درخواست‌های واردشده دریک سیستم توزیع‌شده به‌صورت عادلانه و میزان بندی شده بین منابع سیستم تقسیم می‌شوند؛ به‌طوری‌که میزان انتظار درخواست‌ها کمتر شود.

استفاده از الگوریتم‌های توازن بار موجب می‌شود تا از سیستم و منابع موجود در سیستم بهترین استفاده را ببریم.

۳- وظایف الگوریتم توازن بار:

همانطورکه در تعریف نیز اشاره شد، الگوریتم توازن بار تلاش می‌کند که بار را به‌صورت عادلانه بین پردازنده‌ها تقسیم کند و مانع از آن شود که یک پردازنده مشغول شده درحالی‌که پردازنده دیگری بی‌کار مانده.

عمل توازن بار در سیستم‌های توزیع‌شده عاملی مهم وحیاتی درافزایش کارائی دارد.

۴- انواع الگوریتم‌های توازن بار:

به‌صورت کلی می‌توان این الگوریتم‌ها را به دو نوع تقسیم کرد: الگوریتم‌های ایستا و الگوریتم‌های پویا؛ که در ادامه این الگوریتم‌ها را بررسی می‌کنیم.

۴-۱- *الگوریتم‌های static (ایستا):*

در این‌گونه الگوریتم‌ها تصمیم‌گیری درباره پردازش و تخصیص آن به یک پردازنده در زمان کامپایل کردن پردازش صورت می‌گیرد. اطلاعات هر پردازش اعم از منابع موردنیاز و زمان اجرا نیز در همان زمان مشخص می‌شوند. در این‌گونه الگوریتم‌ها مفهومی بنام انتقال پردازش نداریم. چراکه تمام پردازش از ابتدا تا انتها درهمان پردازنده‌ای اجرا می‌شود که به آن اختصاص‌یافته؛ بنابراین این پردازش‌ها ذاتاً از نوع non-preemptive هستند. هدف اصلی این الگوریتم‌ها به حداقل رساندن زمان اجرای پردازش و ارتباطات بین گرهاست. مشکل اساسی این الگوریتم‌ها در آن است که یک پردازش تمام دوره حیات خود را در یک پردازنده مشخص می‌گذراند؛ که این امر می‌تواند موجب عدم توازن بار در بین پردازنده‌ها شود.

۴-۱-۱- *انواع الگوریتم‌های static (ایستا):*

به‌صورت کلی چهار نوع الگوریتم رایج ایستا داریم Robin Round، Randomized، Central Manager و Threshold که در زیر توضیح داده‌شده‌اند:

۴-۱-۱-۱- الگوریتم *Round Robin*:

در این الگوریتم پردازنده اصلی پردازش‌ها را به‌صورت مساوی و مرتب بین پردازنده‌های فرعی تقسیم می‌کند. تمام پردازش‌ها به ترتیب چرخشی بین پردازنده‌ها تقسیم می‌شوند و وقتی به آخرین پردازنده رسید دوباره از پردازنده اول شروع می‌کند. این روش را روش RR می‌گویند. مزیت این الگوریتم در آن است که پردازنده‌ها به‌صورت مستقل و جدا از هم کارهای محول شده به خود را انجام می‌دهند و نیازی به ارتباط با یکدیگر
ندارند، همین امر تا حد زیادی ترافیک ارتباطی را کم می‌کند.

مشکل این الگوریتم زمانی عیان می‌شود که پردازش‌های زمان اجرای متفاوتی داشته باشند. در این حالت مشاهده می‌شود که پردازنده‌هایی که پردازش‌های طولانی را گرفته‌اند به‌شدت مشغول‌اند و پردازنده‌هایی که پردازش‌های کوچک را گرفته‌اند کاملاً بی‌کار مانده‌اند. پس با توجه به این نقص، این الگوریتم تنها برای محیط هائی مناسب است که پردازش‌های هم‌اندازه‌ای داشته باشند. برای مثال وب‌سرویس‌ها از این الگوریتم استفاده می‌کنند چراکه درخواست‌های HTTP اغلب هم‌اندازه و مشابه‌اند.

۴-۱-۱-۲- الگوریتم *Rondomized*:

این الگوریتم یکی از پردازنده‌ها را به‌صورت تصادفی از میان پردازنده‌های آماده‌به‌کار انتخاب کرده و پردازش را به آن می‌دهد. در برخی کاربردها این الگوریتم بهترین کارائی را از خود نشان می‌دهد.

۴-۱-۱-۳- الگوریتم *Central Manager*:

در این الگوریتم پردازنده‌ای انتخاب می‌شود که کمترین بار را داشته باشد هر بار که پردازش جدیدی وارد سیستم می‌شود پردازنده مرکزی اطلاعات تمام پردازنده‌ها را می‌خواند و پردازش را به پردازنده‌ای که کمترین بار را دارد ارسال می‌کند. این الگوریتم نیاز به ارتباط بین پردازنده‌ای زیادی دارد زیرا هر بار که پردازش جدیدی وارد می‌شود اطلاعات تمام پردازنده‌ها به پردازنده اصلی ارسال می‌شود.

۴-۱-۱-۴- الگوریتم *Threshold*:

این روش از سطح‌بندی خاصی استفاده می‌کند تا میزان بار هر پردازنده را تخمین بزند در این الگوریتم هر پردازنده ازنظر میزان بار می‌تواند دریکی از حالات زیر باشد.

underloaded : Load < t_under

Medium : t_under ≤ load ≤t_upper

Overloaded:load< t_upper

در فرمول‌های بالا load میزان بار فعلی پردازنده است. همچنین t_under و t_upper به ترتیب حد پایین و حد بالای انتخاب‌شده برای بارمی باشد. هر گره درون خود یک جدول دارد که وضعیت بار فعلی گره‌های دیگر در آن ثبت‌شده. با تغییر وضعیت هر پردازنده آن پردازنده یک پیام به بقیه‌ی پردازنده‌ها ارسال می‌کند تا وضعیت آن را در جدول خود بروز کند. هنگامی‌که یک پردازش جدید وارد یک پردازنده شود، اگر آن پردازنده در حالت overloaded نباشد، همان پردازنده پردازش را انتخاب کرده و انجام می‌دهد. در غیراینصورت پردازش به یک پردازنده که در حالت underloaded قرار دارد داده می‌شود، و اگر هیچ پردازنده‌ای در این حالت نباشد پردازش، پس از مدتی انتظار توسط همان پردازشگر اجرا می‌شود. این الگوریتم ارتباط بین پردازنده‌ای کلی دارد و اغلب پردازش‌ها به‌صورت محلی اجرا می‌شوند. درنتیجه انتقال پردازش‌ها بین پردازشگرها کم شده و کارائی سیستم بالا می‌رود. عیب این الگوریتم در

آن است که هنگامی‌که تمام پردازنده‌ها در حالتoverloaded باشند ممکن است یک پردازنده صدها پردازش را بگیرد؛ ولی پردازنده دیگر تنها چند پردازش داشته باشد. همین امر موجب افزایش زمان اجرای پردازش می‌شود.

۴-۲- *الگوریتم‌های Dynamic (پویا) :*

در این‌گونه الگوریتم‌ها برخلاف الگوریتم‌های ایستا بار پردازشی در حین اجرای پردازش‌ها از پردازنده‌ای با بار بیشتر به پردازش با بار کمتر منتقل می‌شود. این کار باهدف بهبود کارائی سیستم انجام می‌شود.

در سیستم‌های توزیع‌شده توازن بار می‌تواند به دو صورت انجام شود: توزیع‌شده و غیر توزیعی. در حالت توزیع‌شده الگوریتم توسط تمام گره‌ها اجرا می‌شود و درواقع عمل توازن بار بین تمام گره‌ها به‌صورت مشترک اجرا می‌شود. ارتباط بین دو گره نیز در این الگوریتم‌ها به دو شکل cooperativeو non-cooperative است. در حالتcooperative گره‌ها با یکدیگر کار می‌کنند تابه یک هدف مشترک دست یابند. برای مثال بهبود زمان پاسخگوئی کل سیستم و... اما در روش non-cooperative هر گره برای اهداف محلی و شخصی خود، کار می‌کنند. مثلاً بهبود زمان پاسخگوئی یک پردازش محلی. الگوریتم‌های پویا معمولاً پیام‌های بیشتری نسبت به الگوریتم ایستا تولید می‌کنند؛ زیرا هر گره در سیستم نیاز به ارتباط و تعامل با بقیه گره‌ها دارد. مزیت این امر در آن است که اگر یک یا چند گره به هر علتی از کار بیافتند، سیستم قادر به ادامه حیات خواهد بود. در الگوریتم‌های غیر توزیع‌شده یک یا گروهی از گره‌ها وظیفه تعادل بار را به عهده‌دارند. این الگوریتم‌ها به‌صورت کلی دو نوع هستند: مرکزی: در نوع مرکزی الگوریتم توازن بار تنها روی یک گره اجرا می‌شود وهمان گره وظیفه توازن بار دیگر گره‌ها را دارد. بقیه گره‌ها نیز تنها با همین گره (گره مرکزی) ارتباط برقرار می‌کنند. شبه توزیع‌شده: در این روش گره‌های مختلف سیستم به کلاسترهای مختلف تقسیم‌بندی می‌شوند و در هر گره عمل توازن بار به روش مرکزی انجام می‌شود. در هر کلاستر، گره مرکزی بر اساس یک سری پارامتر انتخاب‌شده و وظیفه توازن بار را به انجام می‌رساند. انتقال پیام درروش متمرکز بسیار کمتر از روش شبه توزیع‌شده است زیرا گره‌ها تنها لازم است با یک گره مرکزی که برای تمام سیستم مشخص‌شده ارتباط برقرار کنند. بااین‌حال روش متمرکز به دلیل استفاده از تنها یک گره برای انجام عمل توازن بار ممکن است تحت‌فشار محاسباتی زیادی قرارگرفته و قادر نباشد برای تمام سیستم و درخواست‌ها پاسخگو باشد. از طرفی اگر این گره به هر دلیل دچار نقص شود، تمام سیستم از کار خواهد افتاد. به همین دلیل الگوریتم‌های مرکزی بیشتر مناسب شبکه‌های کوچک‌اند.

۴-۲-۱- *استراتژی‌های بکار رفته در الگوریتم‌های تعادل بار پویا:*

استراتژی‌ها و سیاست‌های مختلفی در الگوریتم‌های تعادل بار پویا وجود دارند که قسمتی از آنها را در این بخش توضیح داده‌ایم:

۴-۲-۲- *استراتژی انتقال:*

قسمتی از الگوریتم که تصمیم به انتقال پردازش‌ها از یک گره سنگین شده می‌گیرد را استراتژی انتقال گویند درواقع این استراتژی تشخیص می‌دهد که چه وقتی نیاز به کمتر کردن بار یک پردازنده داریم. این کار در اغلب الگوریتم‌ها بر اساس تکنیک threshold انجام می‌شود.

۴-۲-۳- *استراتژی انتخاب:*

این استراتژی بعد از استراتژی انتقال صورت می‌گیرد. به‌محض اینکه استراتژی انتقال تصمیم به انتقال پردازش از یک گره را گرفت آنگاه باید انتخاب کنیم که کدام پردازش باید برای انتقال انتخاب شود. ساده‌ترین و محبوب‌ترین روش انتخاب آخرین پردازش واردشده است. این کار سبک‌ترین راه‌حل موجود است.

۴-۲-۴- *استراتژی مکان :*

قسمتی از الگوریتم که گره مقصد را برای پردازش انتخاب‌شده مشخص می‌کند استراتژی مکان نام دارد. این استراتژی ابتدا چک

می‌کند که آیا گره مقصد منابع موردنیاز برای پردازش موردنظر را دارد یا خیر.

۴-۲-۵- *استراتژی اطلاعات :*

قسمتی از الگوریتم است که وظیفه جمع‌آوری اطلاعات درباره یک گره را دارد. استراتژی زمان مناسب برای جمع‌آوری اطلاعات از پردازنده‌ها را تعیین می‌کند. این کار می‌تواند به سه شکل دوره‌ای، مبتنی بر درخواست و مبتنی بر تغییر وضعیت گره‌ها انجام شود.

۴-۳- *انواع الگوریتم‌های توازن بار پویا :*

۴-۳-۱- *الگوریتم‌های صف مرکزی:*

این نوع الگوریتم‌ها بر اساس اصل توزیع پویا عمل می‌کنند. این الگوریتم‌ها فعالیت‌های جدید و ناتمام را در یک صف حلقوی در گره مرکزی نگاه می‌دارند. پردازش‌های جدید در انتهای صف قرار می‌گیرند و درصورتی‌که گره ای بی‌کار شود یک پردازش را در صورت وجود، از ابتدای صف بیرون کشیده و اجرا می‌کند. درغیراینصورت آن گره درصف درخواست‌کننده‌ها قرار می‌گیرد تا اینکه یک پردازش وارد صف پردازش‌ها شده و به آن تعلق گیرد. گره زمانی درخواست پردازش جدید می‌کند که میزان بار آن از حد معینی کمتر شود. در این صورت گره یک درخواست به مدیر بار مرکزی می‌فرستد. و مدیر نیز در صورت وجود پردازش درصف پردازش‌هایکی را از ابتدای صف‌کشیده و به آن گره می‌دهد تا آن را اجرا کند.

۴-۳-۲- *الگوریتم صف محلی:*

مشخصه اصلی این الگوریتم انتقال کاملاً پویای پردازش‌ها است. به این صورت که پردازش‌ها ابتدا که پردازنده مرکزی منتقل می‌شوند؛ چون در حالت اولیه تمام پردازنده‌های سیستم در حالت under-loaded قرار دارند، پردازنده مرکزی پردازش‌ها را به‌صورت مساوی بین پردازنده‌ها تقسیم می‌کند. و وقتی یک گره در حالت under-loaded قرار می‌گیرد تلاش می‌کند که از بقیه پردازنده‌های مشغول‌تر پردازش را گرفته و اجرا کند. این گره به‌صورت تصادفی پیام هائی را شامل تعداد پردازش‌های آماده بکار خود، به گره‌های دیگر ارسال می‌کند. وقتی یک گره این پیام را دریافت می‌کند، ابتدا تعداد پردازش‌های خود را با تعداد پردازش‌های درخواست‌کننده مقایسه می‌کند، اگر پردازش‌های او بیشتر باشند تعدادی از پردازش‌ها را به درخواست‌کننده ارسال می‌کند تا آنها را اجرا کند. بدین‌صورت تعادل بار در بین گره‌های سیستم ایجاد می‌شود.

۵- پارامترهای کیفی ارزیابی الگوریتم‌های توازن بار:

برای آنکه بتوانیم مقایسه‌ای بین الگوریتم‌های مختلف توازن بار داشته باشیم، ابتدا باید پارامترهای مؤثر در کیفیت آنها را بدانیم در این بخش این پارامترها را بررسی خواهیم کرد:

۵-۱- *قابلیت رد بار بیش‌ازحد:*

هنگامی‌که سیستم دیگر قادر به پذیرش بار بیشتر نیست باید بار درخواستی را رد کند و به بهترین شکل بار موجود را مدیریت کند.
قابلیت اعتماد:

این عامل قابلیت اعتماد یک الگوریتم را در هنگام بروز خرابی در سیستم اندازه می‌گیرد. الگوریتم‌های static معمولاً قابلیت اطمینان کمتری دارند، زیرا در هنگام بروز خطا در یک ماشین، پردازش قابلیت انتقال به ماشین دیگر را ندارد. اما برخلاف آنها الگوریتم‌های پویا به دلیل قابلیت انتقال پردازش به ماشین‌های دیگر قابل اعتمادتراند.

۴-۳-۳- *قابلیت سازگاری:*

از این پارامتر استفاده می‌کنیم تا قابلیت سازگاری الگوریتم با تغییرات مختلف در سیستم را بسنجیم. الگوریتم‌های ایستا الگوریتم‌های سازگاری نیستند، چراکه با تغیر محیط این الگوریتم‌ها دیگر قادر به ادامه کار نخواهند بود. برای مثال، تغیر تعداد پردازنده‌ها موجب ازکارافتادن این‌گونه الگوریتم‌ها خواهد شد. اما در مقابل، الگوریتم‌های پویا همانطور که از نامشان مشخص است، کاملاً خود را با شرایط جدید وقف می‌دهند و با تغییر محیط هیچ اختلالی در کار ایجاد نخواهد شد.

۴-۳-۴- *قابلیت پایداری:*

پایداری عبارت است از تأخیر در انتقال اطلاعات بین پردازنده‌ها. الگوریتم‌های ایستا الگوریتم‌هایی کاملاً پایدار هستند. چراکه هیچ اطلاعاتی بین پردازنده‌ها رد و بدل نمی‌شود.

۴-۳-۵- *قابلیت پیش‌بینی:*

این پارامتر قابلیت پیش‌بینی عملکرد یک الگوریتم را دریک زمان مشخص، اندازه می‌گیرد. قابلیت پیش‌بینی الگوریتم‌های ایستا بسیار بالاست زیرا اغلب موارد مربوط به پردازش مانند زمان اجرا و پیش‌نیازهای آن و... در زمان ترجمه پردازش و پیش از اجرای آن مشخص می‌شوند. اما در الگوریتم‌های پویا به دلیل اینکه بسیاری از پارامترهای یک پردازش درزمان اجرا مشخص‌شده و تغییر می‌کنند، پیش‌بینی مشکل تراست.

۴-۳-۶- *قابلیت تعامل (cooperative):*

این پارامتر مشخص می‌کند که آیا پردازنده‌ها برای تصمیم‌گیری در مورد مکان پردازش با یکدیگر اطلاعاتی را ردوبدل می‌کنند یا خیر. آنچه که این پارامتر مشخص می‌کند وسعت استقلالی است که هر پردازنده در استفاده از منابع خود دارد در شرایط cooperative تمام پردازنده‌ها مسئول‌اند که وظیفه شخصی خود را انجام دهند اما درنهایت تمام پردازنده‌ها باهم همکاری می‌کنند تا تمام سیستم کارائی بهتری پیدا کند. اما در مورد الگوریتم‌های non-cooperative پردازنده‌ها به‌صورت موجودیت‌های مستقل از هم کار می‌کنند و صرف‌نظر از بقیه سیستم وظیفه خود را انجام می‌رسانند.

۴-۳-۷- *قابلیت خطاپذیری:*

این پارامتر هنگام وقوع خطا به الگوریتم امکان ادامه عمل را می‌دهد. هنگام وقوع خطا الگوریتم به میزان وسعت وجدی بودن آن خطا دچار نقصان کارائی می‌شود. در برخی الگوریتم‌های ایستا حتی یک خطای کوچک‌تر نیز می‌تواند موجب ازکارافتادن تمام سیستم شود.

۴-۳-۸- *استفاده از منابع:*

دریک سیستم توزیع‌شده ممکن است تعداد زیادی پردازش به‌صورت ناگهانی وارد سیستم شوند. در این حالت الگوریتم باید بتواند پردازش‌ها را به پردازنده‌های با بار کمتر بدهد و به بهترین نحو از منابع سیستم استفاده کند. الگوریتم‌های ایستا در این زمینه ضعیف‌تر عمل می‌کنند چراکه این الگوریتم‌ها تنها تلاش می‌کنند که پردازش‌ها را به نحوی تقسیم کنند که سیستم کمترین زمان پاسخ را داشته باشد، و به این نکته توجه نکرده‌اند که ممکن است یک پردازنده، پردازش خود را به اتمام رسانده و بی‌کار بماند. اما در مقابل الگوریتم‌های پویا از منابع نسبتاً بهتر استفاده می‌کنند چراکه به این واقعیت تأکیددارند که بار باید به‌صورت مساوی بین منابع تقسیم‌شده و هیچ منبعی بی‌کار نماند.

۴-۳-۹- قابلیت (preemptive)

الگوریتم‌هایی که پردازش خود را از یک پردازنده به پردازنده دیگر انتقال می‌دهندpreemptive هستند و الگوریتم‌های که پردازش را انحصاراً در یک پردازنده اجرا می‌کنندnon- preemptive هستند. الگوریتم‌های static طبیعتاً همگی non- preemptive هستند زیرا پردازش‌ها را تنها در یک پردازنده اجرا می‌کنند، و الگوریتم‌های پویا نیز می‌توانندpreemptive یا non- preemptive باشند.

۴-۳-۱۰- زمان پاسخ:

زمان پاسخ عبارت است از زمانی که طول می‌کشد تا یک پردازش وارد سیستم شود و به آن پاسخ دهیم. الگوریتم‌های staticمعمولاً زمان پاسخ کمتری دارند.همانطورکه در قسمت‌های قبل نیز گفته‌ایم این الگوریتم‌ها بر اتمام سریعتر کارها تأکیددارند اما الگوریتم‌های Dynamic بر استفاده بهینه از منابع تأکید می‌کنند.

۴-۳-۱۱- زمان انتظار:

جمع بازه‌های زمانی است که یک پردازش درصف انتظار می‌ماند.

۴-۳-۱۲- زمان انجام کار (Throughput)

فاصله زمانی بین گرفتن یک پردازش تا زمان اتمام آن را زمان انجام پردازش گویند.

۴-۳-۱۳- Processor thrashing

این حالت زمانی اتفاق می‌افتد که اغلب پردازنده‌های سیستم بیشتر وقت خود را برای انتقال پردازش‌ها از یک پردازنده به پردازنده دیگر تلف کنند. بدون اینکه هیچ کار مفیدی انجام دهند. این حالت تنها در الگوریتم‌های Dynamic پدید می‌آیند زیرا در الگوریتم‌های Static پردازش‌ها تنها در یک پردازنده انجام‌شده و جابجا نمی‌شوند.

Parameters	Round Robin	Random	Local queue	Central Queue	Central Manager	Threshold
Nature	Static	Static	Dynamic	Dynamic	Static	Static
Overload Rejection	No	No	Yes	Yes	No	No
Reliability	Less	Less	More	More	Less	Less
Adaptability	Less	Less	More	More	Less	Less
Stability	Large	Large	Small	Small	Large	Large
Predictability	More	More	Less	Less	More	More
Forecasting Agency	More	More	Less	Less	More	More
Cooperative	No	No	Yes	Yes	Yes	Yes
Fault Tolerant	No	No	Yes	Yes	Yes	No
Resource Utilization	Less	Less	More	Less	Less	Less
Process Migration	No	No	Yes	No	No	No
Preemptiveness	Non-preemptive	Non-preemptive	Preemptive and Non-preemptive	Preemptive and Non-preemptive	Non-preemptive	Non-preemptive
Response Time	Less	Less	More	More	Less	Less
Waiting Time	More	More	Less	Less	More	More
Turnaround Time	Less	Less	More	More	Less	Less
Execution System	Decentralized	Decentralized	Decentralized	Centralized	Centralized	Decentralized
Throughput	Low	Low	High	High	Low	Low
Processor Thrashing	No	No	Yes	Yes	No	No

۶- نتیجه‌گیری

هدف از ارائه این مقاله مقایسه الگوریتم‌های مختلف، بر اساس پارامترهای کیفی بوده است. مقایسه انجام‌شده نشان می‌دهد که الگوریتم‌های ایستا پایدارتر از الگوریتم‌های پویا هستند. اما در عوض الگوریتم‌های پویا ازنظر قابلیت اعتماد، قابلیت سازگاری، قابلیت همکاری، خطاپذیری، استفاده بهینه از منابع و... عملکرد بهتری از خود نشان می‌دهند. این امر برتری الگوریتم‌های پویا را نسبت به ایستا در بسیاری از محیط‌های شبکه‌ای نشان می‌دهد.

در جدول زیر انواع الگوریتم‌های توضیح داده شده در این مقاله را نسبت به پارامترهای مختلف مقایسه کردیم.

شکل ۱: مقایسه الگوریتم‌های مختلف براساس پارامترهای کیفی[۱]

مراجع

[1] Abhijit A. Rajguru, S.S. Apte, "A Comparative Performance Analysis of Load Balancing Algorithms in Distributed System using Qualitative Parameters" *International Journal of Recent Technology and Engineering*, Vol. 1, Issue. 3, August 2012.

[2] Rupam Mukhopadhyay, Dibyajyoti Ghosh, Nandini Mukherjee, "A Study on the Application of Existing Load Balancing Algorithms for Large, Dynamic, Heterogeneous Distributed Systems" *recent advances in software engineering, parallel and distributed systems*, Vol. 1, No. 1, 2012.
[3] Zenon Chaczko, Venkatesh Mahadevan, Shahrzad Aslanzadeh, Christopher Mcdermid, "Availability and Load Balancing in Cloud Computing" *International Conference on Computer and Software Modeling*, vol.14, 2011.
[4] M. Nikravan and M. H. Kashani, "A Genetic Algorithm for Process Scheduling in Distributed Operating Systems Considering Load balancing", Proceedings 21st European Conference on Modelling and Simulation (ECMS), 2007.
[5] Hendra Rahmawan, Yudi Satria Gondokaryono, "The Simulation of Static Load Balancing Algorithms", 2009 International Conference on Electrical Engineering and Informatics, Malaysia.
[6] Sandeep Sharma, Sarabjit Singh, and Meenakshi Sharma, "Performance Analysis of Load Balancing Algorithms", academy of science, engineering and technology, issue 38, February 2008, pp. 269-272.
S. Malik, "Dynamic Load Balancing in a Network of Workstation", 95.515 Research Report, 19 November,2000.[8] Ali M. Alakeel, A Guide to Dynamic Load Balancing in Distributed Computer Systems, IJCSNSInternational Journal of Computer Science and Network Security, VOL.10 No.6, June 2010.