علوم رایانش و فناوری اطلاعات
نشریه علمی انجمن کامپیوتر ایران
مجلد ۱۷، شماره ۱، ۱۳۹۸
صفحات ۳۴-۲۸
مقاله عادی

بهبود مصالحه هزینه-کارایی در شبکه‌های تحویل محتوا با استفاده از الگوریتم کلونی زنبورعسل

حمید قاسمی[۱]، مهدی جعفری سیاوشانی[۲*]

[*]نویسنده مسئول، دریافت: ۹۷/۰۲/۱۶، بازنگری: ۹۷/۰۶/۰۹، پذیرش: ۹۷/۱۱/۰۹

[۱]دانش‌آموخته کارشناسی ارشد، مهندسی کامپیوتر ، دانشکده مهندسی کامپیوتر، دانشگاه صنعتی شریف ، تهران، ایران

[۲] استادیار، مهندسی کامپیوتر، دانشکده مهندسی کامپیوتر، دانشگاه صنعتی شریف ، تهران، ایران

چکیده

امروزه یکی از راهکارهای اصلی افزایش کارایی سیستم‌های غیرمتمرکز، استفاده از چندین سرویس‌دهنده و پخش بار میان آن‌ها است. با استفاده از این روش، نه تنها میزان کارایی سیستم افزایش می‌یابد، بلکه دسترس‌پذیری سیستم نیز به طور قابل توجهی افزایش خواهدیافت. در حال حاضر الگوریتم‌ها و روش‌های زیادی به منظور پیاده‌سازی یک پخش‌کننده بار ارائه شده‌است که هریک بر بخشی از نیازمندی‌ها غلبه کرده‌است. نکته‌ای که در میان روش‌های ارائه شده دور از چشم مانده‌است، هزینه انتقالی است که سیستم (یا کاربر انتهایی) به ازای افزایش کارایی متحمل می‌شود. همانطور که در [۱] اشاره شده‌است، همواره مصالحه‌ای[۱] میان هزینه و کارایی سیستم وجود دارد. در این مقاله از الگوریتم کلونی زنبور عسل به منظور پخش بار استفاده می‌کنیم. در الگوریتم ارائه شده علاوه بر توزیع بار، پارامتر هزینه نیز در نظر گرفته‌شده و در انتها نشان می‌دهیم که به منظور کاهش هزینه و کارایی به صورت هم‌زمان، استفاده از این الگوریتم نسبت به الگوریتم‌های ارائه شده در [۱] نتایج بهتری داشته و هم چنین سربار کنترلی کم‌تری به سیستم تحمیل خواهدکرد.

کلمات کلیدی: توزیع بار، الگوریتم زنبورعسل، سرویس‌دهنده‌های غیرمتمرکز، شبکه‌های تحویل محتوا، مصالحه هزینه-کارایی.

۱- مقدمه

در دنیای امروز، استفاده از اینترنت و به اشتراک‌گذاری فایل‌های مختلف در این بستر تبدیل به نیازی روزافزون برای کاربران شده‌است. وب‌سایت‌های بسیار زیادی وجود دارند که از سراسر دنیا کاربران به آن‌ها متصل شده و حتی هر کاربر روزانه چندین بار به آن‌ها نیاز پیدا می‌کند. حجم درخواست‌های ارسال شده به سرویس‌دهنده چنین پایگاهی، به قدری زیاد است که یک سرور به تنهایی امکان پاسخگویی به همه کاربران را نخواهد داشت. به همین دلیل استفاده از شبکه‌های تحویل محتوا به منظور انتقال اطلاعات به کاربران، به امری اجتناب ناپذیر تبدیل شده‌است. با استفاده از شبکه‌های تحویل محتوا، اطلاعات مورد نیاز بر روی سرویس‌دهنده‌های مختلفی در سراسر دنیا قرار داده شده تا درخواست کاربران میان این سرویس‌دهنده‌ها تقسیم شود.

در این میان، برخی مواقع ارسال درخواست‌ها به سرویس دهنده‌ها افزایش یافته و باعث ایجاد مشکلاتی در کیفیت پاسخگویی شبکه تحویل محتوا خواهدشد. به طور کلی هرچه الگوریتم پخش بار استفاده شده در سیستم بهینه‌تر عمل کند، احتمال بوجود آمدن مشکل و تاخیر در پاسخگویی کاهش خواهد یافت. همچنین همانطور که در [۱] بیان شده‌است، مصالحه‌ای میان هزینه و کارایی الگوریتم‌های پخش بار وجود دارد که در نتیجه توجه به هزینه بوجود آمده توسط الگوریتم پخش بار ضروری خواهدبود.

به طور کلی روش‌های فعلی بدون در نظر گرفتن هزینه ایجاد شده، به دنبال افزایش کارایی سیستم هستند. به علاوه، در اکثر این روش‌ها داشتن اطلاعات به‌روز و دقیق از وضعیت هر سرویس‌دهنده، لازمه کارکرد صحیح الگوریتم است که باعث افزایش حجم اطلاعات کنترلی ارسال شده از طریق شبکه خواهدشد. در

بخش شبیه‌سازی خواهیم دید که افزایش حجم اطلاعات کنترلی (اطلاعات انتقال داده‌شده از وضعیت سرویس‌دهنده‌ها) باعث شلوغی شبکه و در نتیجه کاهش کارایی سیستم خواهد شد.

در این مقاله ابتدا روش‌ها و الگوریتم‌های ارائه شده به منظور پخش بار در سیستم را مرور کرده، سپس پارامترهای تاثیرگذار در ارزیابی یک الگوریتم پخش بار را به طور مختصر معرفی می‌کنیم. در بخش ۳ الگوریتم پیشنهادی خود را ارائه نموده و در بخش ۴ نتایج پیاده‌سازی و شبیه‌سازی الگوریتم را تشریح خواهیم کرد.

۲- روش‌های مرسوم در پخش بار

در این مقاله به طور کلی روش‌های پخش بار به دو دسته متمرکز و غیرمتمرکز تقسیم می‌شوند. در سیستم‌های پخش بار متمرکز، همواره یک سیستم مرکزی تصمیم‌گیرنده وجود دارد که کاربران را به سمت سرویس‌دهنده‌های مناسب هدایت می‌کند. اما بر خلاف آن در سیستم‌های غیرمتمرکز، هریک از سرویس‌دهنده‌ها قابلیت هدایت کاربران به سرورهای مناسب را دارند. بنابراین کاربر ابتدا به نزدیک‌ترین سرویس‌دهنده موجود متصل شده و پس از آن به سمت بهترین سرویس‌دهنده هدایت خواهد شد [۳].

انتخاب بهترین سرویس‌دهنده مستلزم داشتن اطلاعات کاملی از وضعیت شبکه خواهد بود که بدست آوردن این اطلاعات باعث ایجاد سربار اضافه در شبکه می‌شود [۴]. به همین دلیل روش‌های پیشنهادی امروزی تلاش می‌کنند تا با داشتن اطلاعات کم‌تر از سرویس دهنده‌ها، بهترین سرویس‌دهنده را انتخاب کنند. در ابتدایی‌ترین این روش‌ها بهترین سرویس‌دهنده با استفاده از دو انتخاب تصادفی و برگزیدن گزینه بهتر مشخص می‌شوند [۵]. نسخه‌های دیگر برگرفته از این روش در شبکه‌های تحویل محتوا، ضمن رعایت اصل سادگی الگوریتم، کارایی بهتری در انتخاب سرویس‌دهنده مناسب داشته‌اند [۶]. در مقاله [۷] نویسندگان با بهره‌گیری از روش چند انتخاب تصادفی تلاش می‌کنند تا با قرار دادن درخواست هر کاربر در چند صف به صورت هم‌زمان، نرخ پاسخ به کاربران را کاهش دهند.

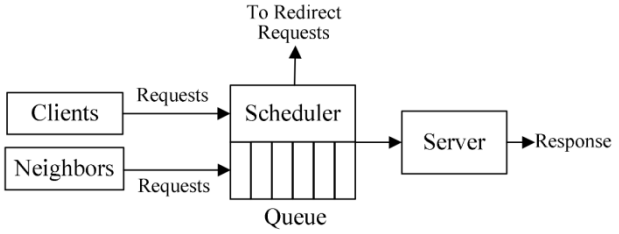
در مقاله [۸] نویسندگان تلاش می‌کنند تا با استفاده از مدل جریان سیال[۲]، بار وارد شده به هر سرویس‌دهنده را میان همسایگان به صورت متناسب پخش کنند. در این روش، به‌روز رسانی وضعیت سرویس‌دهنده‌ها به صورت دوره‌ای انجام می‌گیرد. هرچه بازه زمانی هر دوره کوتاه‌تر باشد، اطلاعات از وضعیت همسایگان دقیق‌تر بوده اما اطلاعات کنترلی ارسال شده در شبکه باعث ایجاد شلوغی بیشتری خواهد شد. بنابراین بدست آوردن دوره به‌روز رسانی مناسب، از چالش های ارزیابی این روش خواهد بود.

در کنار روش های پخش بار در شبکه‌های تحویل محتوا، الگوریتم‌هایی به منظور مدیریت اطلاعات موجود در هر سرویس‌دهنده ارائه شده‌است که به صورت غیرمستقیم به افزایش کارایی شبکه تحویل محتوا کمک خواهدکرد. در این الگوریتم‌ها هدف بر آن است تا فایل‌هایی که بیشتر مورد توجه کاربران بوده و نرخ دسترسی به آن‌ها بیشتر است، در سرویس دهنده‌های بیشتری قرار گیرند[۹]. نحوه تصمیم‌گیری در مورد اضافه کردن یک فایل به سرویس‌دهنده و انتخاب یک فایل به منظور حذف از سرویس‌دهنده، از چالش‌های موجود در این روش‌هاست.

توجه به هزینه بوجود آمده در اثر پخش بار میان سرویس‌دهنده‌ها، نکته‌ای است که در [۱] به آن اشاره شده‌است. با توجه به این مقاله، همواره میان تأخیر پاسخ به کاربران و هزینه تحمیل شده به سیستم مصالحه‌ای وجود دارد. بنابراین توجه به هزینه ایجاد شده در هر یک از روش‌های پخش بار می‌تواند در ارزیابی الگوریتم تاثیرگذار باشد.

در مقاله [۱۰] نویسندگان تلاش می‌کنند تا هزینه تحمیل شده به سیستم را تا حد ممکن کاهش دهند. در نتیجه این کار، با توجه به این که شلوغی سرویس‌دهنده‌ها مورد توجه قرار نمی‌گیرد، روش ارائه شده دارای کارایی مناسب نبوده و نرخ پاسخ به کاربران به میزان قابل توجهی کاهش خواهدیافت.

در مقاله [۱۱] که اخیرا منتشر شده‌است، نویسندگان الگوریتمی ارائه کرده‌اند که بتوانند پخش بار را با توجه به میزان تأخیر دریافت پاسخ به کاربر انتهایی انجام دهند. در واقع نویسندگان مقاله میزان تأخیر اضافه شده به پاسخ کاربران توسط الگوریتم پخش‌بار را به عنوان هزینه در نظر گرفته و تلاش می‌کنند حداکثر هزینه تحمیل شده به کاربر را توسط یک پارامتر حدآستانه[۳] کنترل نمایند. هر کاربر در این روش درخواست خود را به نزدیک‌ترین سرویس‌دهنده موجود ارسال خواهد کرد و سرویس‌دهنده با توجه به وضعیت شلوغی سرویس‌دهنده‌های همسایه و وضعیت خود تصمیم می‌گیرد که به درخواست وارد شده پاسخ دهد و یا آن را به سرویس‌دهنده دیگری بسپارد. میزان شلوغی هر سرویس‌دهنده بر اساس تعداد درخواست‌های موجود در صف آن سرویس‌دهنده مشخص خواهد شد. همانطور که در شکل۱ می‌بینید، ۲ دسته درخواست به سرویس‌دهنده ارسال خواهدشد، درخواست‌های رسیده از طرف کاربران انتهایی و درخواست‌های انتقال داده‌شده از طرف دیگر سرویس‌دهنگان. هر سرویس‌دهنده موظف است به درخواست‌های انتقال داده‌شده پاسخ دهد، بنابراین الگوریتم مسیریابی تنها روی درخواست‌هایی که مستقیما از طرف کاربر انتهایی رسیده‌است اجرا خواهد شد.



شکل ۱- مدل صف در نظر گرفته‌شده برای الگوریتم پخش بار [۱۱].

برای انجام مسیریابی هر سرویس‌دهنده نیاز به اطلاع از وضعیت دیگر سرویس‌دهنده‌های موجود در شبکه دارد، بنابراین در هر بازه زمانی T هر سرویس‌دهنده وضعیت خود را برای دیگران ارسال می‌کند و تا بازه زمانی بعدی از این اطلاعات ارسال‌شده استفاده خواهدشد. به منظور کنترل هزینه انتقال درخواست‌ها، نویسندگان، پارامتر D_T را به عنوان حداکثر فاصله قابل تحمل معرفی کرده‌اند و در صورتی که فاصله ۲ سرویس‌دهنده بیش از این مقدار باشد، انتقال درخواست میان آن‌ها انجام نخواهد شد.

در بخش‌های بعدی از این الگوریتم به منظور انجام مقایسه با راهکار پیشنهادی خود استفاده خواهیم کرد.

۲-۱- الگوریتم زنبور عسل

الگوریتم زنبورعسل[۴] یک الگوریتم بهینه‌سازی است که بر اساس هوش جمعی و با استفاده از شبیه‌سازی رفتار دسته‌های زنبورعسل توسعه یافته است. در نسخه ابتدایی، این الگوریتم ترکیبی از جستجوی محلی و جستجوی تصادفی را انجام می‌دهد [۲].

یک کلونی از زنبورهای عسل می‌توانند در طول فواصل بلند و در جهات مختلف به طور هم‌زمان به برداشت شهد یا گرده از منابع غذایی متعدد پراکنده شوند. بخش کوچکی از این کلونی به طور مداوم محیط زیست را برای پیدا کردن تکه‌های گل جدید جستجو می‌کنند. این زنبورهای دیده بان به طور تصادفی در منطقه اطراف کندو حرکت می‌کنند و به ارزیابی سودآوری منابع غذایی وارد شده می‌پردازند. وقتی آن‌ها به کندو باز می‌گردند، آن دسته از زنبورهایی که منبع غذایی بسیار سودآوری پیدا کرده‌اند به قسمتی از کندو رفته و رقص مخصوصی را انجام می‌دهند. از آن جا که طول رقص متناسب با امتیاز دیده‌بان از منبع غذایی

است، کاوشگرهای بیشتری برای برداشت تکه‌های گل با بهترین امتیاز استخدام می‌شوند.

در الگوریتم ۱ شبه کد استاندارد الگوریتم زنبور عسل را مشاهده می‌کنید. در روال اولیه نصب تعداد n_s زنبورهای عسل دیده‌بان به طور تصادفی در فضای جستجو قرار می‌گیرند و سازگاری محلی که در آن قرار دارند را ارزیابی می‌کنند. در مرحله استخدام، تعداد $n_b \leq n_s$ زنبور که بهترین جواب‌ها را بدست آورده‌اند رقص مخصوص را انجام داده و به نسبت سازگاری نتیجه بدست آورده زنبورهایی را برای انجام جستجوی محلی استخدام می‌کنند. سپس زنبورهای استخدام شده شروع به جستجوی محل مورد نظر کرده و در نهایت بهترین پاسخ محلی را بدست خواهند آورد.

در طول این زمان، بخشی از زنبورهای جستجوگر که منطقه مناسبی نیافته‌اند به صورت تصادفی در مناطق دیگر جستجو انجام داده و در صورتی که پاسخی مناسب‌تر از هریک از n_b های موجود بدست آورند، جایگزین آن می‌شوند. در مقالات دیگر، کارایی این الگوریتم مورد بررسی قرارگرفته است [۱۲]. همچنین نسخه‌های دیگری از این الگوریتم با هدف بهبود سرعت دستیابی به بهترین جواب ارائه شده‌است [۱۳].

الگوریتم ۱ شبه کد الگوریتم زنبورعسل [۱۳].

```
//FP stands for Flower Patch
for i = 1 ··· n_s do
    scout[i] = InitialiseScout()
    flowerPatch[i] = InitialiseFlowerPatch(scout[i])
end for
repeat
    Recruitment()
    for i = 1 ··· n_b do
        FP[i] = LocalSearch(FP[i])
        FP[i] = SiteAbandonment(FP[i])
        FP[i] = NeighbourhoodShrinking(FP[i])
    end for
    for i = n_b ··· n_s do
        FP[i] = GlobalSearch(FP[i])
    end for
until StoppingCondition = TRUE
```

۳- روش پیشنهادی

هدف اصلی ما در این مقاله ارائه الگوریتمی است که علاوه بر تأخیر پاسخدهی کم، هزینه کم‌تری را در شبکه ایجاد کند. همچنین با توجه به این‌که اطلاعات کنترلی ردوبدل شده میان سرویس‌دهنده‌ها تأثیر قابل توجهی بر شلوغی شبکه و در نتیجه کارایی سیستم خواهد داشت، تلاش می‌کنیم تا در الگوریتم ارائه شده سربار کم‌تری را به شبکه تحمیل کنیم.

۳-۱- تعریف مسئله

فرض می‌کنیم شبکه تحویل محتوایی در اختیار داریم که هدف از آن رساندن فایل‌هایی از یک مجموعه شامل N فایل به صورت $W = \{W_1, \dots, W_N\}$ به کاربرانی است که هریک از آن‌ها را درخواست می‌کنند. شبکه مذکور دارای ۳ عنصر اصلی است:

- L سرویس‌دهنده که هریک قابلیت ذخیره‌سازی M فایل را دارند.
- K کاربر که به صورت پیوسته و با توزیع پوآسن مجزا هرکدام با نرخ λ_i درخواست‌های خود را ارسال می‌کنند، $i \in [1:K]$.
- یک شبکه ارتباطی که از طریق آن درخواست کاربران و پاسخ آن‌ها انتقال پیدا می‌کند. هزینه انتقال یک فایل از سرور j به کاربر i معادل $C_{i,j}$ بوده که در نتیجه می‌توانیم ماتریس هزینه $C = [c_{i,j}]_{i=1,j=1}^{i=K,j=L}$ را تشکیل دهیم.

در هر درخواست، احتمال درخواست فایل W_i برابر p_i است. در نتیجه مجموعه $\rho = \{p_1, \dots, p_n\}$ میزان محبوبیت هر فایل را مشخص می‌کند.

فرض کنید زمان زیادی (T) از اجرای پخش بار گذشته باشد، می‌توانیم پارامترهای زیر را تعریف کنیم:

- n_i = تعداد درخواست‌های کاربر iام در این بازه زمانی. این عدد یک متغیر تصادفی با $\mathbb{E}[n_i] = T\lambda_i$ خواهد بود.
- $t_i = (t_{i,1}, \dots, t_{i,n_i})$ ، درحالی که $t_{i,j}$ برابر است با زمان رسیدن jامین درخواست کاربر iام.
- $d_i = (d_{i,1}, \dots, d_{i,n_i})$ ، درحالی که $d_{i,j}$ برابراست با شماره فایل jامین درخواست کاربر i. در این حالت $d_{i,j}$ ها یک دسته متغیر تصادفی i.i.d. با توزیع ρ را تشکیل خواهند داد.

همانطور که در بالا اشاره شد، با هر ورود درخواست در زمان t، الگوریتم پخش بار اجرا شده و با در نظر گرفتن وضعیت بار سرویس‌دهنده‌ها (طول صف انتظار)، q_t، و داشتن ماتریس هزینه ارتباط C، درخواست را به یک سرویس‌دهنده ارسال می‌کند. این عمل تخصیص هر درخواست به یک سرور را با تابع ψ نمایش خواهیم داد:

$$\psi\left((i,j), C, q(t_{i,j})\right) \rightarrow \{1, \dots, L\}. \quad (۱)$$

درحالی که (i,j) jامین درخواست کاربر iام را مشخص کرده و $t_{i,j}$ نمایانگر زمان دریافت این درخواست است.

علاوه بر تابع بالا، ۲ پارامتر زیر را تعریف می‌کنیم:

- $c_i = \left(c_{i,S_{i,1}}, \dots, c_{i,S_{i,n_i}}\right)$ درحالی که $c_{i,S_{i,j}}$ هزینه ارتباط jامین درخواست کاربر iام را مشخص می‌کند. در این‌جا $S_{i,j}$، سرویس‌دهنده مسئول این درخواست را مشخص کرده که با استفاده از رابطه فوق بدست آمده است.

یعنی: $S_{i,j} \triangleq \psi\left((i,j), C, q(t_{i,j})\right)$.

- $\tau_i = (\tau_{i,1}, \dots, \tau_{i,n_i})$، درحالی که $\tau_{i,j}$ برابر است با بازه زمانی از لحظه‌ای که درخواست دریافت‌شده را به سرویس‌دهنده $S_{i,j}$ ارسال کرده تا زمانی که کاربر کلیه پاسخ موردنظر خود را دریافت کند.

با استفاده از تعاریف فوق می‌توانیم دو معیار برای اندازه‌گیری کارایی الگوریتم پخش بار در نظر بگیریم. اولین معیار در نظر گرفته‌شده، هزینه متوسط به ازای هر درخواست است:

$$\bar{C}(\psi) \triangleq \mathbb{E}\left[\frac{1}{\sum_{i=1}^{K} n_i} \sum_{i=1}^{K} \sum_{j=1}^{n_i} c_{i,S_{i,j}}\right], \quad (۲)$$

و همچنین دومین معیار درنظر گرفته‌شده، متوسط زمان انتظار خواهدبود:

$$\bar{D}(\psi) \triangleq \mathbb{E}\left[\frac{1}{\sum_{i=1}^{K} n_i} \sum_{i=1}^{K} \sum_{j=1}^{n_i} \tau_{i,j}\right]. \quad (۳)$$

با استفاده از این دو معیار و با توجه به وجود مصالحه میان آن‌ها تلاش می‌کنیم الگوریتمی ارائه دهیم تا با در نظر گرفتن هر ۲ پارامتر، نتیجه‌ای قابل قبول بدست آوریم.

۳-۲- راهکار ارائه‌شده

همانطور که در بخش ۲-۱ بیان شد، الگوریتم زنبورعسل در حل مسائل بهینه‌سازی پیچیده، تأثیر به‌سزایی دارد. از این الگوریتم در محاسبات ابری و نحوه زمانبندی کارها برای پخش بار استفاده شده‌است [۱۵] [۱۶]. راهکار ارائه شده در این بخش، استفاده از این الگوریتم در پخش بار شبکه‌های تحویل محتوا است به گونه‌ای که بتوانیم دو پارامتر هزینه و تأخیر پاسخدهی را به صورت همزمان مورد توجه قرار دهیم. به همین منظور ابتدا تلاش می‌کنیم مسئله تعریف شده در بخش ۳-۱ را بر الگوریتم زنبورعسل منطبق نماییم.

با توجه به این‌که استفاده از یک پخش‌کننده بار مرکزی باعث ایجاد مشکل نقطه آسیب‌پذیر مرکزی[۵] خواهد شد، پخش بار را به صورت غیرمتمرکز انجام خواهیم داد. به این صورت که هر یک از سرویس‌دهنده‌ها به صورت مجزا الگوریتم را اجرا خواهند کرد. همچنین به منظور جلوگیری از دست‌به‌دست شدن بیش از حد یک درخواست میان سرویس‌دهنده‌های مختلف، هر سرویس‌دهنده موظف است بدون در نظر گرفتن شرایط شبکه، به درخواست انتقال داده شده از سرویس‌دهنده دیگر پاسخ دهد.

در راهکار پیشنهادی از هیچ عامل اضافی به منظور تخمین وضعیت سرویس‌دهنده‌های دیگر استفاده نمی‌کنیم. به همین دلیل هر درخواستی که به سرویس‌دهنده‌های دیگر انتقال می‌یابد به عنوان یک زنبور جستجوگر در نظر گرفته می‌شود. هنگامی که پاسخ هر سرویس‌دهنده به یک درخواست پایان یافت، یک بسته از وضعیت فعلی سرویس‌دهنده به مبدأ انتقال درخواست ارسال می‌شود تا سرویس‌دهنده مبدأ با استفاده از این پاسخ‌ها بتواند اطلاعات خود را از دیگر سرویس‌دهنده‌ها به‌روز کند. همچنین به منظور انجام جستجوی تصادفی در طول اجرای این الگوریتم، بخشی از درخواست‌های دریافت شده توسط هر سرویس‌دهنده به صورت تصادفی به سرویس‌دهنده‌هایی که فایل مورد نظر را در اختیار دارند انتقال داده می‌شوند. در الگوریتم۲ فرم کلی الگوریتم پیشنهادی نمایش داده شده است. پارامتر γ در این الگوریتم، نسبت میزان جستجوی تصادفی و محلی را مشخص می‌کند. بهترین مقدار این پارامتر پس از پیاده‌سازی الگوریتم و اجرای آن در شرایط مختلف و با مقادیر مختلف بدست خواهد آمد.

الگوریتم ۲ شبه کد الگوریتم پخش بار پیشنهادی.

Require: $(i,j), \gamma, q(t_{i,j})$
$\Lambda \leftarrow \{k | k \in [1 \cdots L], d_{i,j} \text{ in } k \text{ cache}\}$
$fittestServer = EvaluateServersStatus(\Lambda)$
$rnd \leftarrow$ uniform random in $[0,1]$
if $rnd < \gamma$ **then**
  $SendReqToRandomServer(\Lambda \setminus \{fittestServer\})$
else
  $LocalSearch(fittestServer)$
end if
$updateServersStatus()$

در قسمت ارزش‌گذاری هر سرویس‌دهنده علاوه بر در نظر گرفتن وضعیت شلوغی سرویس‌دهنده (که رابطه مستقیم با میزان تأخیر پاسخ دارد)، میزان هزینه تحمیلی را نیز به عنوان یک پارامتر تأثیرگذار درنظر خواهیم گرفت. به همین منظور از رابطه زیر برای ارزش‌گذاری هر سرویس‌دهنده استفاده خواهیم کرد:

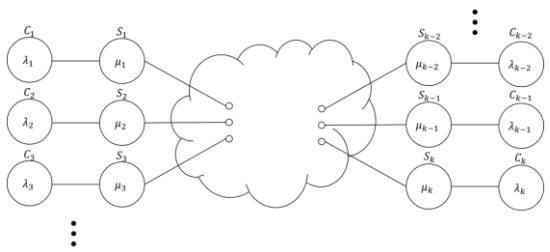
$$\alpha \bar{C}(\psi) + (1-\alpha)\bar{D}(\psi). \quad (۴)$$

پارامتر α عددی در بازه $[0,1]$ بوده که میزان توجه به هریک از پارامترهای هزینه و تأخیر پاسخدهی را مشخص می‌کند. با تغییر این پارامتر می‌توانیم الگوریتم را به گونه‌ای تنظیم کنیم که در نقاط مختلف نمودار هزینه-کارایی کار کند. در این رابطه هرچه مقدار بدست آمده برای یک سرویس‌دهنده کم‌تر باشد، سرویس دهنده مذکور گزینه بهتری برای انتقال درخواست خواهد بود.

۴- ارزیابی کارایی

به منظور ارزیابی الگوریتم ارائه شده از شبیه‌سازی یک شبکه تحویل محتوا استفاده خواهیم کرد. در حال حاضر چند شبیه‌ساز برای شبکه‌های تحویل محتوا ارائه شده‌است [۱۷] [۱۸] که متأسفانه به دلیل محدودیت‌های زیادی که در پیاده‌سازی دارند، هیچ‌یک شبیه‌ساز مرجعی نبوده و هم‌چنین امکان پیاده‌سازی مسئله مطرح شده در این مقاله را نیز ندارند. بنابراین ابتدا با استفاده از شبیه‌ساز شبکه NS-3 [۱۹] یک شبکه تحویل محتوا ایجاد کرده و سپس به ارزیابی الگوریتم ارائه شده خواهیم پرداخت.

شبکه طراحی شده شامل L=25 گره بوده که به صورت یک گراف لاتیس[۶] به به یکدیگر متصل شده‌اند. با توجه به مقایسه‌های انجام شده در [۲۰]، گراف‌های مختلف درنظر گرفته شده، تأثیری بر نتایج شبیه‌سازی نداشته و همواره در همه آن‌ها مصالحه هزینه-کارایی برقرار است. بنابراین استفاده از گراف لاتیس برای شبیه‌سازی تأثیری بر نتیجه بدست آمده نخواهد داشت. هریک از گره‌ها می‌توانند درخواست‌های کاربران نزدیک به خود را دریافت نموده و درخواست را به سرویس‌دهنده مناسبی که فایل مذکور را در خود داشته باشد انتقال دهد. به همین منظور به هر سرویس‌دهنده یک گره کاربر متصل شده‌است (K=25) و هر کاربر درخواست‌های خود را با توزیع پوآسن و نرخ λ_i به سمت سرویس‌دهنده می‌فرستد (شکل۲). در واقع هر گره کاربر بیانگر درخواست‌های تجمیع شده از همه کاربرانی است که در نزدیکی هر سرویس‌دهنده قرار داشته و درخواست‌های آن‌ها در اولین قدم به آن سرویس‌دهنده خاص ارسال می‌شود.



شکل ۲- شبکه طراحی‌شده در شبیه‌ساز

در این شبیه‌سازی، فاصله هر دو سرویس‌دهنده با یکدیگر را به عنوان هزینه انتقال درخواست میان این دو سرویس‌دهنده در نظر گرفته‌ایم. بنابراین با استفاده از این فواصل می‌توانیم ماتریس هزینه C را بدست آوریم. به منظور تحلیل رفتار الگوریتم در حالت شلوغی، شبکه را در حالت زیر بررسی خواهیم کرد:

$$\sum_{i=1}^{K} \lambda_i = 0.9 \sum_{j=1}^{L} \mu_j. \quad (۵)$$

در این رابطه μ_j نرخ پاسخدهی به هر درخواست توسط سرویس‌دهنده‌ها است که با توزیع نمایی انجام می‌شود. در شبیه‌سازی انجام شده، نرخ پاسخدهی به درخواست‌ها را برای همه سرویس‌دهنده‌ها به یک اندازه درنظر گرفته‌ایم (یعنی: $\forall i: \mu_j = \mu$)

به منظور ارزیابی روش پیشنهادی، شبیه‌سازی را در سه حالت زیر نیز انجام داده‌ایم:

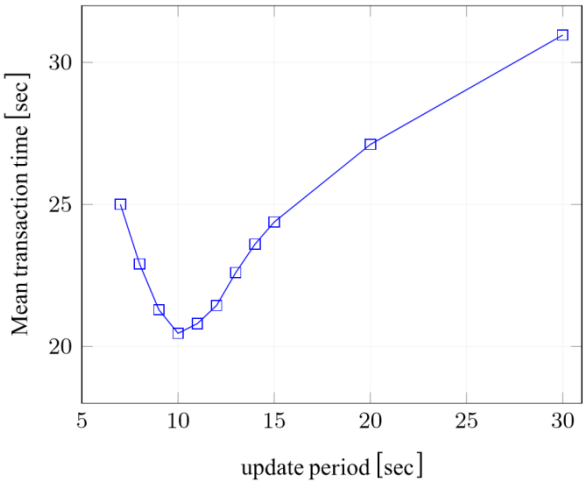
۱- فرض می‌کنیم بدون انتقال اطلاعات کنترلی میان سرویس‌دهنده‌ها، همه آن‌ها از وضعیت دیگر سرویس‌دهنده‌ها آگاه هستند (حالت

ایده‌آل) و برای تصمیم‌گیری در مورد بهترین سرویس‌دهنده از رابطه۴ استفاده می‌کنند.

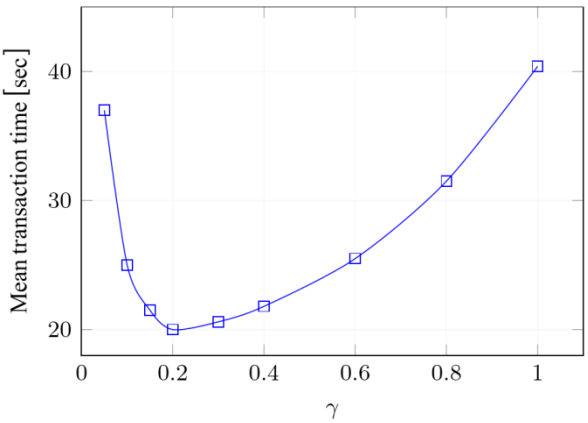
۲- انتقال اطلاعات کنترلی میان سرویس‌دهنده‌ها به صورت دوره‌ای انجام خواهدشد (رایج‌ترین شیوه فعلی) و برای تصمیم‌گیری در مورد بهترین سرویس‌دهنده از رابطه ۴ استفاده می‌شود.

۳- انتقال اطلاعات وضعیت سرویس‌دهنده‌هابه صورت piggyback انجام می‌شود و سرویس‌دهنده انتخاب شده برای پاسخگویی به درخواست از رابطه ۴ بدست خواهد آمد.

در حالت دوم، برای بدست آوردن بهترین جواب ممکن، نیاز به انجام شبیه‌سازی‌هایی داشته تا بتوانیم بهترین بازه زمانی به‌روزرسانی سرویس‌دهنده‌ها را بدست آوریم. در شکل۳ نحوه تغییر میانگین زمان تراکنش[۷] را با افزایش بازه زمانی به‌روزرسانی سرویس‌دهنده‌ها مشاهده می‌کنید. همانطور که مشخص است با کاهش بیش از حد زمان به‌روزرسانی، شبکه دچار ازدحام شده و زمان تراکنش افزایش پیدا می‌کند. بنابراین بهترین بازه زمانی به‌روزرسانی سرویس‌دهنده‌ها حدود ۱۰ ثانیه خواهد بود که در مقایسه از آن استفاده خواهد شد. لازم به ذکر است که این مقدار با توجه به ویژگی‌های هر شبکه متفاوت بوده و برای دیگر شبکه‌ها این مقدار بهینه متفاوت خواهد بود.



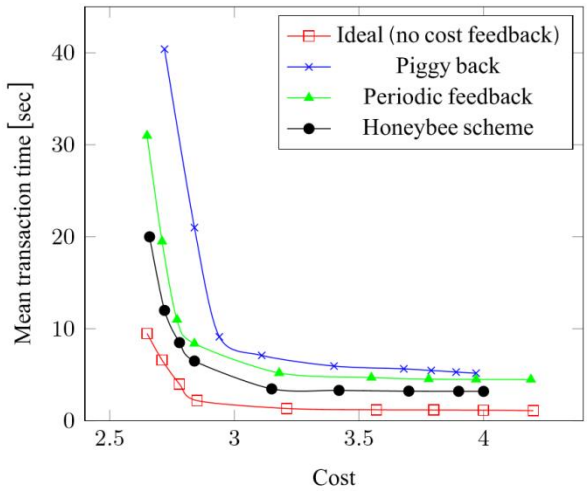
شکل ۳- نمودار تغییر میانگین زمان انجام تراکنش نسبت به بازه زمانی به‌روزرسانی وضعیت سرویس‌دهنده‌ها.



شکل ۴- نمودار تغییر میانگین زمان انجام تراکنش نسبت به پارامتر γ در الگوریتم ارائه شده.

همچنین برای بدست آوردن بهترین مقدار پارامتر γ در الگوریتم زنبورعسل ارائه شده، شبیه‌سازی‌های مختلفی با مقادیر مختلف γ انجام شده‌است که با توجه به نتایج بدست آمده برای این پارامتر مقدار ۰.۲ را در نظر خواهیم گرفت. در شکل۴ نمودار تغییر زمان پاسخگویی به کاربران به ازای γهای مختلف قابل مشاهده است.

در جدول۱ مقادیر پارامترهای درنظر گرفته‌شده برای انجام شبیه‌سازی‌ها ارائه شده‌است. در هربار اجرای شبیه‌سازی، ۴۰۰۰۰ درخواست به کل سیستم ارسال شده و برای بدست آوردن مقدار دقیق، برای هر نقطه ۵ شبیه‌سازی اجرا شده‌است. پس از قرار دادن نقاط مختلف بدست آمده در نمودار و اتصال آن‌ها به یکدیگر، نمودار مصالحه هزینه-زمان تراکنش برای هر الگوریتم بدست آمده است. در شکل۵ نمودارهای بدست آمده برای هر الگوریتم قابل مشاهده است. همانطور که در این شکل مشخص است، مصالحه در الگوریتم زنبورعسل نزدیک به بهترین حالت بوده و می‌توان گفت با استفاده از این روش می‌توان ترکیب هزینه و زمان تراکنش مناسب‌تری از دیگر روش‌های آزمایش شده بدست آورد. هم‌چنین همانطور که گفته شد، در روش پیشنهادی تعداد بسته‌های کنترلی انتقال داده شده در شبکه، بسیار کم بوده و به همین علت ترافیک اضافی در شبکه ایجاد نخواهد کرد. در نمودار شکل۶، مقایسه‌ای از میزان ترافیک اضافی تولید شده توسط هر الگوریتم قابل مشاهده است. نکته قابل ذکر آن است که با افزایش تعداد سرویس‌دهنده‌ها، حجم بسته‌های کنترلی در روش ۲ (به روز رسانی دوره‌ای وضعیت سرویس‌دهنده‌ها) به صورت نمایی افزایش می‌یابد در حالی که در الگوریتم ارائه شده، این عدد به تعداد سرویس‌دهنده‌ها بستگی نداشته و همواره ثابت است.



شکل ۵- نمودار مصالحه میان هزینه و زمان تراکنش در الگوریتم‌های مطرح شده.

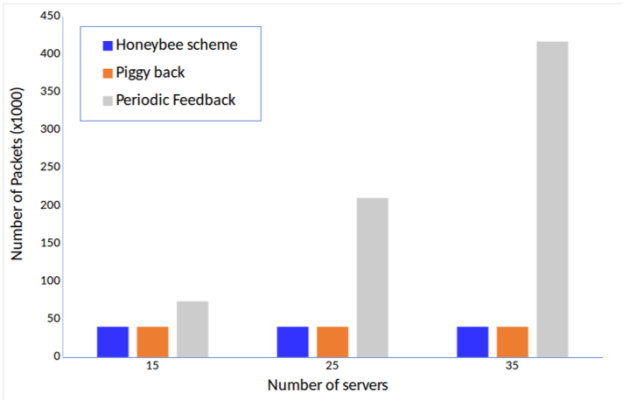
جدول ۱- مقدار پارامترهای درنظر گرفته شده برای انجام شبیه سازی.

نام پارامتر	مقدار پارامتر
تعداد سرویس‌دهنده‌ها	۲۵
تعداد فایل‌ها	۲۵
ظرفیت هر سرویس‌دهنده	۴
گراف ارتباط سرویس‌دهنده‌ها	Lattice
نرخ زمان پاسخ به هر درخواست	۱
سایز داده‌های کنترلی	۱ کیلوبایت
سایز هر فایل	۳ مگابایت

۴-۱- مقایسه الگوریتم پیشنهادی و روش [۱۱]

همانطور که بیان شد، در الگوریتم [۱۱] نویسندگان تلاش کرده‌اند تا با تعیین حداکثر هزینه ارتباط، سیستم پخش‌بار را به گونه‌ای مدیریت کنند که هزینه ارتباط در پخش‌بار از حد مشخص‌شده D_T راتر نرود. پر واضح است که با تغییر مقدار D_T در این روش، می‌توان به مصالحه موجود میان هزینه ارتباط و زمان پاسخ به کاربران انتهایی دست‌یافت. نکته مورد توجه در تعیین D_T آن است

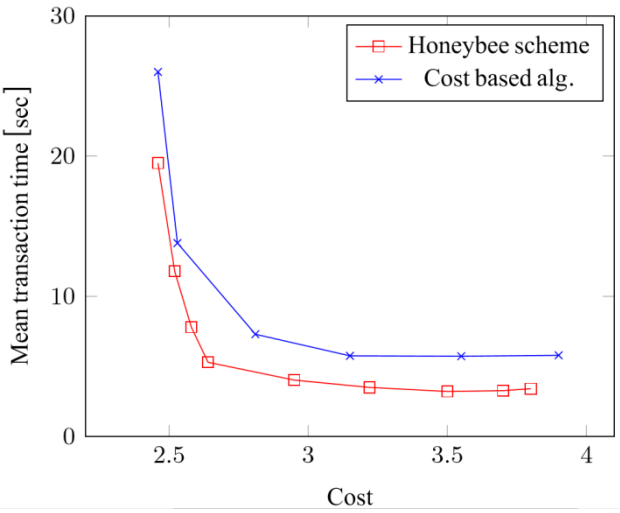
که اگر درخواست به سرویس‌دهنده‌ای برسد که فاصله آن از اولین سرویس‌دهنده شامل فایل درخواستی کاربر، بیشتر از D_T باشد، الگوریتم امکان تصمیم‌گیری برای ارسال درخواست را نخواهدداشت. بنابراین هنگام اجرای شبیه‌سازی‌های مربوط به این الگوریتم اگر شرایط گفته‌شده بوجود آمد، درخواست دریافت‌شده را فارغ از مقدار D_T به نزدیک‌ترین سرویس‌دهنده شامل فایل مورد درخواست، ارسال می‌کنیم. به منظور کم کردن احتمال بوجود آمدن حالت مذکور، در شبیه‌سازی‌های انجام‌شده، تعداد فایل‌های ذخیره‌شده در هر سرویس‌دهنده را افزایش داده‌ایم. در جدول۲ مقدار پارامترهای تعیین‌شده برای انجام شبیه‌سازی مذکور قابل مشاهده است.



شکل ۶- مقایسه بسته‌های کنترلی ارسال شده در شبکه.

جدول ۲ مقدار پارامترهای درنظر گرفته شده برای انجام مقایسه.

نام پارامتر	مقدار پارامتر
تعداد سرویس‌دهنده‌ها	۲۵
تعداد فایل‌ها	۲۵
ظرفیت هر سرویس‌دهنده	۶
گراف ارتباط سرویس‌دهنده‌ها	Square Lattice
نرخ ارسال درخواست‌ها توسط هر کاربر	۰.۹
نرخ زمان پاسخ به هر درخواست	۱
سایز داده‌های کنترلی	۱ کیلوبایت
سایز هر فایل	۳ مگابایت



شکل -۷ مقایسه الگوریتم پیشنهادی و روش ارائه شده در [۱۱]

در شکل۷ نتایج حاصل از انجام شبیه‌سازی و مقایسه دو الگوریتم مذکور قابل مشاهده‌است. همان‌طور که مشخص است با استفاده از روش پیشنهادی می‌توان ترکیب مناسب‌تری از هزینه ارتباط و زمان پاسخ به کاربران پیدا کرد. لازم به ذکر است که در این شکل، نقاط شبیه‌سازی شده برای الگوریتم ارائه‌شده در [۱۱] کم‌تر از الگوریتم پیشنهادی است. دلیل این امر آن است که با توجه به این‌که حداکثر فاصله میان دو سرویس‌دهنده در شرایط شبیه‌سازی شده ۸ است، و با توجه به این‌که قرار دادن مقدار D_T کم، مانند ۱ و ۲ باعث می‌شود در بیشتر مواقع از حد تعیین شده عدول نماییم، الگوریتم مذکور در مجموعه $D_T \in \{3,4,5,6,7,8\}$ شبیه‌سازی شده‌است.

۵- نتیجه گیری و کارهای آینده

در این مقاله تلاش کردیم تا با استفاده از الگوریتم کلونی زنبورعسل و با توجه به وجود مصالحه میان هزینه و تأخیر پاسخ به کاربران الگوریتمی را ارائه کنیم تا علاوه بر پخش بار مناسب میان سرویس‌دهنده‌های مختلف این امکان را به اجرا کننده الگوریتم بدهیم که بتواند با استفاده از تغییر پارامتر α، میزان توجه الگوریتم به هریک از پارامترهای هزینه و کارایی سیستم را مشخص نمایند. هم‌چنین با توجه به بسته‌های کنترلی کمی که در این روش میان سرویس‌دهنده‌های مختلف انتقال داده می‌شود، می‌توان گفت این الگوریتم سربار زیادی را به شبکه تحمیل نمی‌کند تا در نتیجه بتوان به شکل بهینه‌تری از پهنای باند موجود استفاده نمود.

در شبیه سازی‌های انجام شده مشخص شد که استفاده از این الگوریتم کارایی مناسبی دارد، با این حال در این مقاله از الگوریتم پایه کلونی زنبورعسل استفاده شده است که می‌توان در تحقیقات آتی الگوریتم‌های بهبود داده شده بر مبنای کلونی زنبورعسل [۱۳] را مورد بررسی قرار داد. هم‌چنین همانطور که در بخش۲ بیان شد، روش های غیرمستقیمی برای افزایش بهره‌وری یک الگوریتم پخش بار ارائه شده‌است [۹]. ترکیب نمودن این روش‌ها با الگوریتم ارائه شده می‌تواند به ایجاد یک الگوریتم جامع منجر شود که درنتیجه بتوان با سربار محاسباتی و بسته‌های کنترلی کم‌تر نتیجه مناسب‌تری به‌دست آورد.

مراجع

[1] M. Jafari Siavoshani, S. P. Shariatpanahi, H. Ghasemi, and A. Pourmiri, "On communication cost vs. load balancing in Content Delivery Networks," *in Computers and Communications (ISCC)*, pp.651-656, IEEE, 2017.

[2] D. Pham, A. Ghanbarzadeh, E. Koc, S. Otri, S. Rahim, and M. Zaidi, "The bees algorithm-A novel tool for complex optimization," *in Intelligent Production Machines and Systems-2nd I* PROMS Virtual International Conference*, pp.454-459, 2011.

[3] O. Ercetin and L. Tassiulas, "Market-based resource allocation for content delivery in the internet," *IEEE Transactions on Computers*, vol.52, no.12, pp.1573-1585, 2003.

[4] A.M.K. Pathan and R. Buyya, "A taxonomy and survey of content delivery networks," *Grid Computing and Distributed Systems Laboratory, University of Melbourne, Technical Report*, vol.4, 2007.

[5] M. Mitzenmacher, "The power of two choices in randomized load balancing," *IEEE Transactions on Parallel and Distributed Systems*, vol.12, no.10, pp. 1094-1104, 2001.

[6] C.M. Chen, Y. Ling, M. Pang, W. Chen, S. Cai, Y. Suwa, and O. Altintas, "Scalable request routing with next-neighbor load sharing in multi-server environments," *in IEEE Advanced Information Networking and Applications (AINA '05)*, vol.1, pp.441-446, 2005.

[7] K. Grdner, M. Harchol-Balter, A. Scheller-Wolf, M. Veledrnitsky, and S. Zbarsky, "Redundancy-d: The power of d choices for redundancy," *Operations Research*, vol.65, no.4, pp.1078-1094, 2017.

[8] S. Manfredi, F. Oliviero, and S. P. Romano, "A distributed control law for load balancing in content delivery networks," *IEEE/ACM Transactions on Networking (TON)*, vol.21, no.1, pp.55-68, 2013.

[9] D. S. Berger, R. K. Sitaraman, and M. Harchol-Balter, "AdaptSize: Orchestrating the Hot Object Memory Cache in a Content Delivery Network," *in 14th {USENIX} Symposium on Networked Systems Design and Implementation (NSDI 17)*, (Boston MA), pp.483-498, 2017.

[10] M. Adler, R. K. Sitaraman, and H. Venkataramani, "Algorithms for optimizing the bandwidth cost of content delivery," *Computer Networks*, vol.55, no.18, pp.4007-4020, 2011.

[11] Q. Shuai, K. Wang, F. Miao, and L. Jin, "A cost-based distributed algorithm for load balancing in content delivery network," in *IEEE Intelligent Human-Machine Systems and Cybernetics (IHMSC), 19th International Conference*, vol.1, pp.11-15, 2017.

[12] D. T. Pham and M. Castellani, "Benchmarking and comparison of nature-inspired population-based continuous optimisation algorithms," *Soft Computing*, vol.18, no.5, pp.871-903, 2014.

[13] H. R. Nasrinpour, A. M. Bavani, and M. Teshnehlab, "Grouped Bees Algorithm: A Grouped Version of the Bees Algorithm," *Computers*, vol.6, no.1, p.5, 2017.

[14] D. T. Pham, M. Castellani, "The bees algorithm: modelling foraging behaviour to solve continuous optimization problems," Proceedings of *the Institution of Mechanical Engineers*, Part C: Journal of Mechanical Engineering Science, vol.223, no.12, pp.2919-2938, 2009.

[15] D. B. LD and P. V. Krishna, "Honey bee behavior inspired load balancing of tasks in cloud computing environments," *Applied Soft Computing*, vol.13, no.5, pp.2292-2303, 2013.

[16] A. Dave, B. Patel, and G. Bhatt, "Load balancing in cloud computing using optimization techniques: A study," in *IEEE Communication and Electronics Systems (ICCES)*, pp.1-6, 2016.

[17] T. Bostoen, J. Napper, S. Mullender, and Y. Berbers, "A simulator to assess energy-saving techniques in content distribution networks," *in Energy-Efficient Data Centers*, pp.83-98, Springer, 2014.

[18] K. Stamos, G. Pallis, A. Vakali, D. Katsaros, A. Sidiropoulos, and Y. Manolopoulos, "Cdnsim: A simulation tool for content distribution networks," *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, vol.20, no.2, p.10, 2010.

[19] G. F. Riley and T. R. Henderson, "The ns-3 network simulator," *Modeling and tools for network simulation*, pp.15-34, 2010.

[20] M. Jafari Siavoshani, A. Pourmiri, and S. P. Shariatpanahi, "Storage, Communication, and Load Balancing Trade-off in Distributed Cache Networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 29, no. 4, pp. 943-957, 2017.

مهدی جعفری سیاوشانی در حال حاضر استادیار دانشکده کامپیوتر دانشگاه صنعتی شریف است. او مدرک کارشناسی برق و فیزیک خود را از دانشکده برق و فیزیک دانشگاه صنعتی شریف، و مدرک ارشد و دکتری خود را از دانشکده مخابرات و علوم کامپیوتر دانشگاه صنعتی لوزان دریافت کرده است. همچنین او بعد از اتمام مقطع دکترا، مدتی را به عنوان محقق پسادکترا در «موسسه کدگذاری شبکه» در دانشگاه چینی هنگ‌کنگ سپری کرده است. از علایق پژوهشی او می‌توان به یافتن محدودیت‌های بنیادی در سیستم‌ها و شبکه‌های کامپیوتری اشاره کرد. سیستم‌های توزیع‌شده، پردازش داده‌های حجیم و یادگیری ماشین از دیگر علاقمندی‌های پژوهشی او هستند.

آدرس پست الکترونیکی ایشان عبارت است از:

mjafari@sharif.edu

حمید قاسمی مدرک کارشناسی خود را از دانشگاه صنعتی امیرکبیر و کارشناسی ارشد خود را از دانشگاه صنعتی شریف، هر دو در رشته مهندسی کامپیوتر دریافت کرده‌است. از علاقمندی‌های پژوهشی او می‌توان به تحلیل و افزایش کارایی شبکه‌های کامپیوتری، بررسی سیستم‌های توزیع‌شده و رایانش ابری اشاره کرد.

آدرس پست الکترونیکی ایشان عبارت است از:

hghasemi@ce.sharif.edu

---

1 Trade off
2 Fluid Flow Model
3 Threshold
4 Bees algorithm
5 Single Point of Failure
6 Lattice Graph
7 Transaction time

The CSI Journal on Computing Science and Information Technology
Vol. 17, No.1, 2019

Improvement of Cost-Performance Trade-off in Content Delivery Networks using Honeybee Algorithm

Hamid Ghasemi, Mahdi Jafari Siavoshani

Department of Computer Engineering, Sharif University of Technology, Tehran, Iran

Abstract

Today, using multiple servers and balancing the load between them, is one of the leading solutions to improve the performance of services. Using this method will both increase the performance and availability of the whole system dramatically. Currently, there are different algorithms proposed to improve performance and meet some of the system requirements. However, one of the most important parameters that have not been given enough attention is the communication cost added to the system by balancing the load among different servers. As it is mentioned in [1], there is a tradeoff between the communication cost and performance of the system .

In this paper, we are going to use a scheme, inspired by the bee's algorithm, to balance the load among servers while we consider both performance and the communication cost. In the evaluation phase, we show that using the proposed algorithm will help us to achieve better communication cost and performance simultaneously compared to the other existing solutions.

Keywords: Load Balancing, Honeybee Algorithm, Content Delivery Networks, Distributed Service Providers, Cost-Performance Trade-off.