تاریخ دریافت مقاله: ۹۵/۱۲/۱۰
تاریخ پذیرش مقاله: ۹۶/۰۲/۱۸

ارائه روش نوین جهت کشف آگاهانه منابع در سیستم گرید بر اساس الگوریتم بهینه‌سازی ازدحام ذرات

سحر کاوش
دانشجوی کارشناسی ارشد گروه مهندسی برق و کامپیوتر- واحد ماهشهر- دانشگاه آزاد اسلامی-ماهشهر- ایران
پست الکترونیکی: S.kavosh@ymail.com

مرجان عبدیزدان *
استادیار گروه مهندسی برق و کامپیوتر- واحد ماهشهر- دانشگاه آزاد اسلامی-ماهشهر- ایران
پست الکترونیکی: abdeyazdan87@yahoo.com

چکیده

امروزه تمایل زیادی به بررسی و بهبود روش‌های کشف و دسترسی به منابع در سیستم گرید دیده می‌شود. با توسعه محیط‌های شبکه و افزایش تعداد منابع و توزیع جغرافیایی، پیدا کردن الگوریتمی که بتواند در زمان کوتاه و با ترافیک کم، منابع مورد نیاز کاربر را کشف کند، امری مهم می‌باشد. در این مقاله، با استفاده از مدل کشف آگاهانه منبع و روش غیرمتمرکز نشان داده می‌شود که می‌توان منبع بهینه را با کمترین گره‌های ملاقاتی و در کمترین زمان ممکن برای ذخیره جستجو کرده و آن را اخذ کرد. الگوریتم پیشنهادی پس از شبیه‌سازی با استفاده از الگوریتم بهینه‌سازی ازدحام ذرات بهینه‌سازی شده است تا بتواند کوتاه‌ترین مسیر را شناسایی کند و سپس با دو الگوریتم BITMAP و FRDT که از روش‌های سلسله مراتبی کشف منابع می‌باشند، مقایسه شده است. نتایج نشان می‌دهد که تعداد گره‌های ملاقات شده در الگوریتم پیشنهادی از الگوریتم‌های دیگر کمتر است. همچنین نتایج حاصل از ارزیابی نشان می‌دهد که روش پیشنهادی در مقایسه با روش‌های قبلی از سرعت و کارایی بالاتری برخوردار است و در نتیجه ترافیک روش پیشنهادی نسبت به الگوریتم روش FRDT، ۱۲٪ و نسبت به الگوریتم روش BITMAP، ۳۰٪ کاهش یافته است.

کلید واژه‌ها: الگوریتم بهینه‌سازی ازدحام گروهی ذرات، سیستم توزیع شده، کشف آگاهانه منابع، محاسبات گرید، مدل غیرمتمرکز

مقدمه

گرید در سال ۱۹۹۸ توسط یان فوستر و کارل کِسِلمن با الهام از شبکه‌های توزیع برق، معرفی شد. آن‌ها در سال ۲۰۰۱ و به کمک توک، گرید را این‌گونه تعریف کردند (حل مسئله و به اشتراک‌گذاری منابع به صورت هماهنگ در سازمان‌های مجازی چندنهادی و پویا) و بعد اضافه کردند که: (به اشتراک‌گذاری منابع که ما از آن صحبت

* نویسندهٔ مسئول

می‌کنیم، مبادله فایل نیست، بلکه منظور دسترسی مستقیم به کامپیوترهـــا، نرم‌افزارها، داده‌ها و دیگر منابع اســـت.) مرجع این به اشـــتراک‌گذاری تحت کنتـــرل کامل صاحب منبع اســـت. مجموعه‌ای از افراد و یا شرکت‌ها که این نوع به اشـــتراک‌گذاری را داشته باشـــند، سازمان‌های مجازی خوانده می‌شـــوند[۲۱]. سپس فوســـتر، گرید را سیستمی تعریف می‌کند که سه ویژگی زیر را داشته باشد:[۴]

الف) مجموعه‌ای از منابع هماهنگ که دارای یک مدیریت مرکزی نیستند.

ب) از واســـط‌ها و پروتکل‌هـــای اســـتاندارد باز و عام منظوره، بهره می‌برند.

پ) کیفیت ســـرویس‌دهی که کل سیستم ارائه می‌دهد، بزرگ‌تر از کیفیت سرویس هرکدام از منابع است.

جستجوی منابع در بسیاری از سیستم‌های توزیع شده براساس نام می‌باشد.

سیســـتم‌هایی مانند Gnutella [۹]، DNS [۱۵]، Tapestry [۱۲] و Chord [۱۱]، از نـــام منابع (فایل‌ها) برای جســـتجو، اســـتفاده می‌کنند. به‌علت این‌که سیستم گرید محیطی پویا می‌باشـــد، اســـتفاده از نام‌های منحصر به فـــرد به‌عنوان شناسه سراسری منابع، روشی ناکارآمد می‌باشد. راه‌حل این مشکل، اســـتفاده از مشخصات منابع برای جستجوی آن‌ها می‌باشـــد. در این روش‌ها درخواســـت‌های جستجو از مجموعـــه‌ای از صفـــات و مقادیر، مانند نام و نســـخه سیستم‌عامل و بار پردازنده تشکیل می‌شوند.

روش پیشنهاد شده در این مقاله بر اساس ارائه روش نامتمرکز کشـــف منبع به صورت سلسله مراتبی در شبکه گرید می‌باشد. یک سازوکار کشف منبع باید بتواند بهترین منبع درخواســـت شده را کشـــف کند. در روش پیشنهادی سعی شده است درختی با کمترین زمان و کمترین ملاقات گره‌ها، منبع را کشـــف کند و محیط را برای کشـــف منبع بعدی، همیشه به‌روز نگه دارد.

بخش ۲ شـــامل تحقیقات انجام شده جهت کشف منابع در شبکه است. بخش ۳ در مورد الگوریتم‌های فرا ابتکاری و الگوریتم ازدحـــام گروهی ذرات می‌باشـــد. در بخش ۴ روش پیشنهادی مطرح شده است. در این بخش، ایده جدید برای دسترســـی به منابع در سیستم توزیع شده گرید به روش آگاهانه و سلســـله مراتبی با اســـتفاده از الگوریتم ازدحام گروهی ذرات به جهت بهینه شـــدن برای کشـــف منابع تشریح گردیده است. بخش ۵ شامل نتایج و ارزیابی این پژوهش می‌باشـــد. در این بخش، روش پیشـــنهادی با اســـتفاده از نرم‌افزار متلب شبیه‌ســـازی می‌شود و نتایج حاصل مورد ارزیابی و مقایســـه قرار داده می‌شـــود. در بخش ۶ نتیجه‌گیری نهایی تحقیق مطرح شده است.

۱- کارهای انجام شده

در [۱۶] برای جســـتجو و دسترســـی بـــه منابع روش MMO ارائه‌شده اســـت. این روش بر اساس مدل نظیر به نظیر اســـت که توزیع‌شده اســـت. مدل نظیر به نظیر برای جســـتجوی منابع از مشـــخصات منابع استفاده می‌کند. از ویژگی‌های خوب شـــبکه‌های نظیر به نظیر، مقیاس‌پذیری و تحمل‌پذیری خطاست اما میزان ترافیک در این مدل زیاد اســـت و باعث کاهش کارایی می‌شود. در [۱۷] و [۱۸] مدل super peer بررســـی شده اســـت که در این مدل، مدیریت منابع در هر ســـازمانی به‌صـــورت سلســـله مراتبی و یا مرکزی انجام می‌شـــود. سپس از هر ســـازمان، یک گروه به‌عنوان super-peer انتخاب می‌شـــود و تمامی super-peer ها به‌صورت نظیر به نظیر به یکدیگر متصل می‌شوند. در [۱۳] روشـــی برای کشـــف منابع در گرید بر اساس روش اَبَر زمان‌بندی و با استفاده از الگوریتم بهینه‌سازی کلونی مورچه ارائه شـــده است که برای شـــبکه‌ای با محیط پویا، مقیاس‌پذیـــر و بدون کنترل‌کننده جهانی اســـت. در [۲۵] و [۲۶] روش سیل‌آسا برای دسترسی به منابع بررسی‌شده است. ارسال درخواست کاربران و کشف منابع با استفاده از الگوریتم‌های آبشاری انجام می‌شود. در روش سیل‌آسا، پیـــام به تمام اعضای شـــبکه پخش می‌شـــود و هر عضو شبکه، پیام جستجو را به همسایگان متصل به خود ارسال

می‌کند. روش بیان شده در [۱]، نوعی معماری درختی برای کشـف منابع ارائه داده است که در آن از بیت نگاشت‌های مختلف استفاده می‌شود و روش MMO ارائه‌شده در [۱۶] را بهبود داده اسـت. مدیران بر اسـاس سـاختار سلسله مراتبی سـازمان‌دهی شـده‌اند و کشـف منابع به‌صورت آگاهانه انجام می‌شـود. در این معماری، همه ویژگی‌های منابع به بیت نگاشـت تبدیل می‌شـود و درخواست کاربر به مدیران منابع فرسـتاده می‌شود. برای جستجوی منبع، تعداد گره‌هایی که درخواست‌ها به آن‌ها ارسال‌شده است درروش بیت نگاشـت کمتر از روش MMO است. در [۱۹] از مدل سلسـله مراتبی برای مدیریت و دسترسی به منابع در گرید اسـتفاده شده اسـت. در این روش تمامی یال‌ها کدگذاری شده است.

اگر یک گره n فرزند داشـته باشـد تعداد بیت‌های این کد برابر[logn] می‌باشـد و فرزندان ازسمت چپ به ترتیب از صفر و به صورت دودویی شـماره‌گذاری می‌شـوند. هر گره دو رکورد بـه نام‌های «Path Bitmap» و «Counter Bitmap» دارد. رکورد Counter Bitmap تعداد منابع موجود در گره‌های فرزند را مشخص می‌کند. منابع محلی در Path Bitmap ذخیره می‌شود. طول Path Bitmap دو برابر تعداد خصوصیات منبع است. مدل مطرح‌شده در [۲۲] به بررسی انواع مدل‌های کشـف منابع در سیسـتم‌های توزیع‌شده پرداختـه اسـت و مدل‌های مدیریت و کشـف منابع را به پنج دسـته تقسیم نموده است که شامل مدل مرکزی، مدل توزیع‌شـده، مدل نظیر به نظیر، مدل سلسله مراتبی و مدل مبتنی بر عامل اسـت. در [۷] یک روش برای کشف منابع در گرید پیشـنهاد شده است که در آن از الگوریتم درخت دودویی بردار فاصله برای کشف منابع استفاده شده است. همچنین در [۲۳]، [۲۴] و [۲۵] به کشف منابع بر اساس مدل نظیر به نظیر اشاره شـده است. معماری که در [۲۰] بیان شـده است، از روش سلسـله مراتبی و ترکیب آن با مدل نظیر به نظیر بر اساس حد آسـتانه برای کشف منابع در گرید استفاده شـده است که مدیران هر خوشه به‌صورت نظیر به نظیر با همدیگر در تماس هستند. در [۶] از اتوماتای یادگیر به همراه الگوریتم بهینه‌سـازی ازدحام ذرات برای کشف منابع در گرید استفاده شده است.

در [۸] یک مدل کشـف شـبکه بر اساس وب‌سرویس مطرح‌شـده اسـت. این مـدل ترکیبی از نظیر بـه نظیر و فنّاوری وب‌سـرویس اسـت که با ترکیبی از قاب متمرکز و توزیع‌شـده ارائه‌شده است. در [۲۹] روشی برای کشف منابع ارائه‌شـده اسـت که برای سیستم‌های توزیع‌شده و با مقیاس بزرگ اسـت و تا حدودی مشـکل کشف منابع را با اسـتفاده از الگوریتم کلونی مورچه‌ها برطرف کرده اسـت. اما به دلیل این‌که روش پیشـنهادی دارای شناخت کمی از محیط اطراف خود اسـت عملیات کشـف به‌خوبی صورت نمی‌گیرد. همچنین قابلیت مقیاس‌پذیری در روش مذکور دشـوار است. در [۳۰] روشـی برای کشف منابع در محیط گرید بر اسـاس رفتار شبکه‌های نظیر به نظیر و رفتار طبیعی کلونی مورچگان ارائه‌شده است که الگوریتم کلونی مورچگان طرح پیشنهادی را مقیاس‌پذیر نموده و در صورت بروز مشکل، توانایی خودسازمان‌دهی دارند. طرح پیشنهادی ترافیک شبکه را در زمان جستجو کاهش می‌دهد و نیـاز به کنترل‌کنندهٔ مرکزی ندارد. در این روش مورچه عامل اطلاعات تعداد منابع را شناسایی نمی‌کند و شناخت کافـی از محیط اطراف خود ندارد و درنتیجه جسـتجوی هوشمند و هدفمندی ندارد.

۲- الگوریتم‌های فرا ابتکاری

الگوریتم‌هـای فرا ابتـکاری نوعـی از الگوریتم‌های تصادفی هستند که برای یافتن پاسخ بهینه به کار می‌روند. از الگوریتم‌های شـناخته شده فرا ابتکاری بر پایه جمعیت می‌تـوان الگوریتم‌هـای تکاملـی[۳]، (الگوریتـم ژنتیک، برنامه‌ریزی ژنتیک،...)، بهینه‌سـازی کلونی مورچگان [۲]، کلونـی زنبورهـا [۲۸]، روش بهینه‌سـازی ازدحام ذرات گروهی، الگوریتم ریشه-پاجوش و الگوریتم چکه آب‌های هوشـمند را نام برد. از الگوریتم‌های متداول فرا ابتکاری

مبتنی بر یک جواب می‌توان الگوریتم جستجوی ممنوعه [۵] و الگوریتم تبرید شبیه‌سازی شده [۱۰] را نام برد. طراحی و پیاده سازی الگوریتم‌های فرا ابتکاری دارای سه مرحلهٔ متوالی است که هر کدام از آن‌ها دارای گام‌های مختلفی هستند. در هر گام فعالیت‌هایی باید انجام شود تا آن گام کامل شود. مرحلهٔ اول آماده‌سازی است که در آن باید شناخت دقیقی از مسئله‌ای که می‌خواهیم حل کنیم به‌دست آوریم، و اهداف طراحی الگوریتم فرا ابتکاری برای آن باید با توجه به روش‌های حل موجود برای این مسئله به‌طور واضح و شفاف مشخص شود. مرحلهٔ بعدی، ساخت نام دارد. مهم‌ترین اهداف این مرحله انتخاب راهبرد حل، تعریف معیارهای اندازه گیری عملکرد، و طراحی الگوریتم برای راهبرد حل انتخابی می‌باشد. آخرین مرحله پیاده‌سازی است که در آن پیاده‌سازی الگوریتم طراحی شده در مرحلهٔ قبل، شامل تنظیم پارامترها، تحلیل عملکرد و در نهایت تدوین و تهیه گزارش نتایج باید انجام شود [۲۷].

۲-۱ الگوریتم بهینه‌سازی ازدحام ذرات

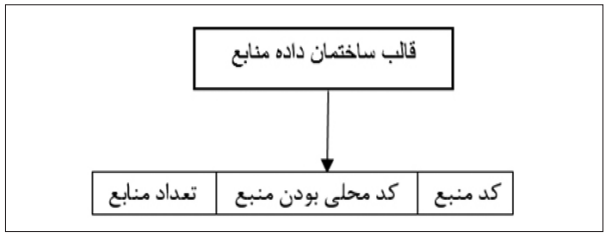
بهینه‌سازی ازدحام گروهی ذرات روشی جهت کشف فضای یک مسئلهٔ ارائه شده برای تنظیمات یا پارامترهای لازم جهت به حداکثر رساندن یک هدف مشخص است. این روش برای اولین بار توسط کندی و ابرهات شرح داده شد. این شبیه‌سازی در سال ۱۹۹۵ انجام شده و آن ها را به سمت شبیه‌سازی رفتار پرندگان برای یافتن دانه رهنمون کرد. این کار تحت تاثیر دو دانشمند دیگر هپنر و گرناندر بود که در سال ۱۹۹۰ برای شبیه‌سازی رفتار پرندگان به صورت یک سیستم غیرخطی انجام شد و این کار منجر به طرح ریزی الگوریتم بهینه‌سازی ازدحام گروهی ذرات جهت بهینه‌سازی شد.

الگوریتم بهینه‌سازی ازدحام گروهی ذرات، راهکاری برای خروج از این بهینه محلی ارائه نمی‌دهد و این بزرگ‌ترین مشکل بهینه‌سازی ازدحام گروهی ذرات استاندارد است که سبب می‌شود در حل مسائل چند قله‌ای، مخصوصا با فضای حالت بزرگ ناتوان باشد. یکی از اعمال صورت گرفته برای مقابله با مشکل بهینه‌های محلی در الگوریتم حرکت دسته جمعی ذرات، استفاده از جهش است [۲۱].

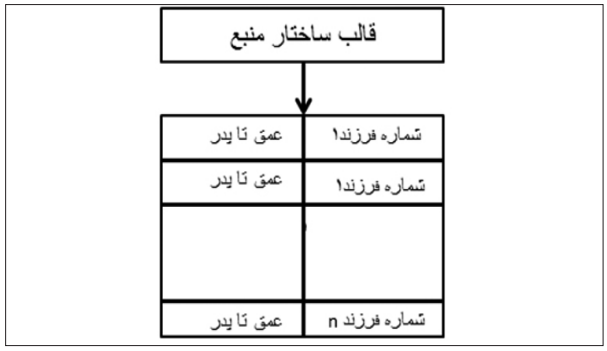
بسیاری از مسائل بهینه‌سازی دنیای واقعی که در آن هدف اصلی بهینه‌سازی می‌باشد و بهینه محلی در آن بر اساس زمان تغییر می‌کند، با استفاده از الگوریتم بهینه‌سازی ازدحام ذرات به خوبی انجام می‌شود[۱۴].

۳- روش پیشنهادی

در روش پیشنهادی سعی شده است درختی با کمترین زمان و کمترین ملاقات گره‌ها، منبع را کشف کند و محیط را برای کشف منبع بعدی، همیشه به‌روز نگه دارد. در روش پیشنهادی که از این پس به‌عنوان NORD_TREE از آن نام خواهیم برد، بر خلاف روش‌های قبلی درختی، که عملیات کشف را به صورت محلی انجام می‌دادند و در آن حالت لزوما بهترین منبع کشف نمی‌شد، سعی شده است بهترین منبع را، با ترافیک کمتر و سریع‌تر از روش‌های قبلی برای کاربران کشف کند. ساختمان داده‌ای که در هر گره برگ، در آخرین سطح، برای ذخیره‌سازی اطلاعات منابع استفاده می‌شود، در شکل ۱، نمایش داده شده است؛ به نحوی که اولین ستون همیشه «کد منبع»، ستون دوم «محلی بودن منبع» و ستون سوم «تعداد هر منبع» را نشان می‌دهد. اگر منبع به صورت محلی در این گره وجود داشته باشد «۰» و اگر در فرزندانش موجود باشد «۱» در این حوزه قرار می‌گیرد. برای نمایش اطلاعات سایر گره‌های فرزندان در شکل ۲، یک لیست پیوندی به گره پدر اضافه می‌شود که اطلاعات فرزندان را در آن لیست نمایش می‌دهد. این لیست پیوندی فقط برای فرزندان هر پدر است؛ یعنی این لیست در صورتی به جدول شکل ۱ اضافه می‌شود که کد محلی بودن فرزند آن حتما «۱» باشد و این بدان معناست که پدر آن گره منبع مورد نظر را به صورت محلی در دسترس ندارد و می‌تواند آن را از گره فرزند خود بگیرد و به جدول



شکل۱: جدول قالب ساختمان داده استفاده شده در کشف منبع بهینه برای گره برگ در اخرین سطح



شکل۲: جدول قالب ساختمان داده استفاده شده در کشف منبع بهینه برای گره‌های فرزند در سطوح بالاتر

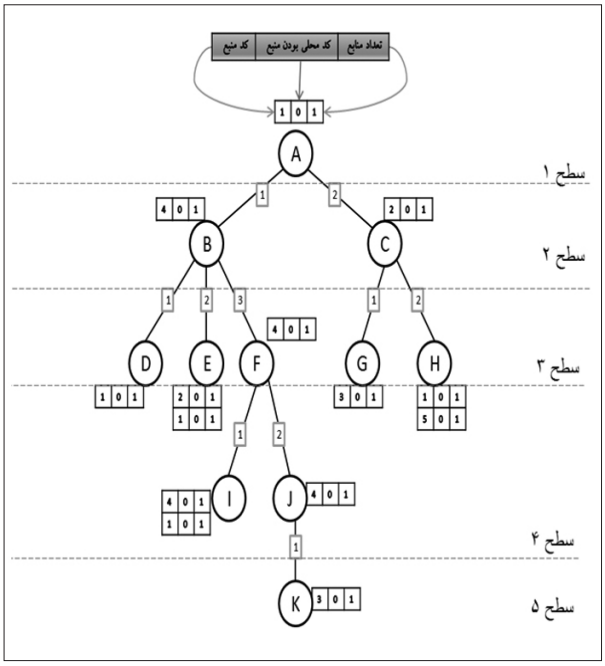
اضافه کند، به‌طوری که اطلاعات موجود در این لیســت به شــکل «شماره فرزند» و «عمق آن تا گره پدر» می‌باشد. به ترتیب که از پایین جدول به سمت بالا پیمایش انجام شود، اطلاعات تمام منابع در این جدول وارد می‌شــود تا نهایتا به گره ریشه برســد که جدول گره ریشه شامل اطلاعات تمامی منابع موجود می‌باشــد. با انجام این کار، اطلاعات کامل‌تــری در مورد شناســایی و کشــف منابع در محیط گرید نســبت به حالات قبلی کشف منبع یافت خواهد شد و در نتیجه زمان جســتجو کاهش می‌یابد و بهترین منبع در دسترس شناسایی خواهد شد.

۳-۱- کشف منبع بهینه در روش پیشنهادی

(NORD_TREE) از دو مرحله تشــکیل شــده اســت، به‌طوری که مرحله اول شامل جمع‌آوری اطلاعات و مرحله دوم شامل کشف منابع می‌باشد.

۳-۱-۱- مرحله اول :جمع‌آوری اطلاعات

این مرحله شامل دو بخش است. بر اساس شکل ۳ در بخــش اول تمام گره‌ها، اگر منبعــی را در خود به صورت محلی داشته باشند باید نوع منبع را در جدول خود ذخیره کننــد. در این بخش پیوندها به صورت صعودی از چپ به

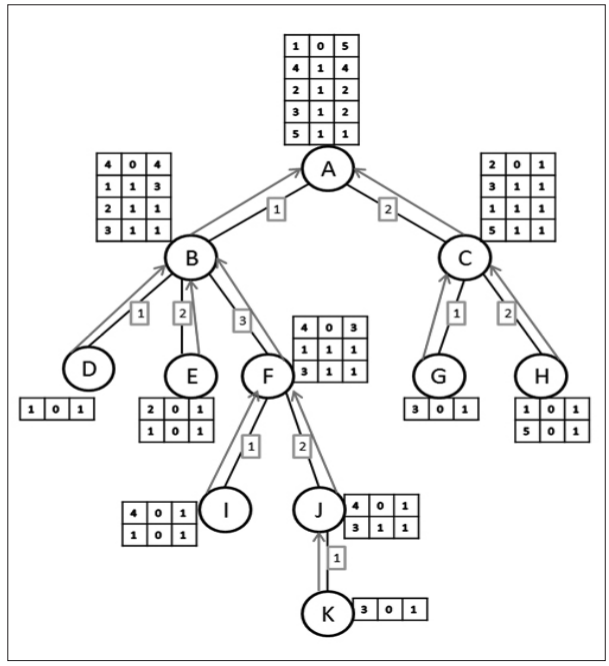


شکل ۳: بخش اول جمع‌آوری اطلاعات در روش NORD_TREE

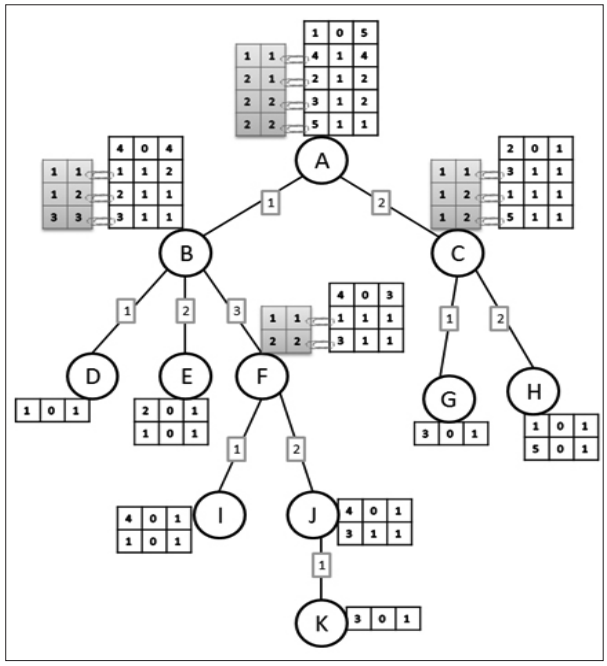
راست شماره‌دار می‌شوند و هر پیوند یک شماره منحصر به فرد دارد. این شــماره‌ها برای شناســایی هر گره مورد استفاده قرار می‌گیرد.

بر اساس شــکل ۴ در بخش دوم، تمام گره‌ها اطلاعات خــود را به پدر خود انتقــال می‌دهند و این انتقال اطلاعات تا وقتی به ریشــه برســد ادامه پیدا می‌کند. اگر منبع مورد جســتجو در یک گره به صورت محلی موجود باشــد در ستون دوم سطر جدولش «۰» قرار می‌گیرد. هر گره پدر در واقع به‌عنوان یک شاخص کارساز در نظر گرفته می‌شود. اگر منبع درخواســتی در پدر باشد آنگاه آن منبع به فرزند تخصیص داده می‌شــود در غیر این‌صورت گره پدر خود نقش یک فرزند را گرفته و از گره پدر بالاتر خود درخواست منبع را می‌کند. این عمل آنقدر تکرار می‌شود تا نزدیک‌ترین منبع پیدا شود و تخصیص صورت گیرد. در بدترین حالت درخواست برای کشف منبع تا ریشه پیش می‌رود و ریشه، اطلاعات تمام گره‌ها را دارد.

با توجه به شــکل۵ اگر همین منبــع در فرزندانش هم موجود بــود به ســطرهای جدول اضافه نمی‌شــود فقط اطلاعــات آن به لیســت پیوندی متصل به جــدول اضافه می‌شــود. اگر منبعی درخــودش موجود نباشــد ولی در



شکل ۴: بخش دوم جمع‌آوری اطلاعات در روش NORD_TREE

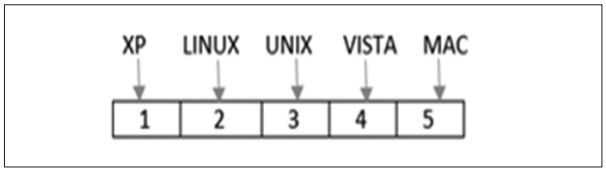


شکل۵: اضافه کردن منابع موجود در فرزندان از طریق لیست پیوندی و محاسبه فاصله آن‌ها تا گره پدر

فرزندانش موجود باشـــد، در ستون دوم سطر منبع مورد نظـــرش «۱» می‌گیـــرد. هر خانه این لیســـت پیوندی از دو قسمت تشکیل شـــده اســـت. خانه اول «عمق منبع تا پدر» و خانه دوم «شـــماره فرزند» می‌باشـــد، عمـــق گره تا پدر به صورت صعودی مرتب شـــده اســـت. نکته مهمی که در مرحله جمع‌آوری اطلاعات مورد توجه می‌باشـــد این است که هـــر دو بخش ۱ و ۲ از مرحله جمـــع‌آوری اطلاعات به صورت همزمان انجام می‌شود. مرحله جمع‌آوری اطلاعات یک بار در طول عمر شبکه انجام می‌شود و در مراحل بعد فقط به‌روزرسانی می‌شـــود که همین عمل مانع از کاهش سرعت کشف در شبکه می‌شود.

۳-۱-۲- مرحله دوم: کشـــف و به‌روزرسانی منبع در روش پیشنهادی

این گام نیز از دو بخش تشکیل شده است. در بخش اول باید یک منبع مناســـب برای کاربر یا همان گره درخواست کننده منبع، پیدا شود. در خواست کشف از هر نقطه از شبکه می‌تواند به شبکه وارد شود. پس در هر شرایط باید روش کشف منبع بتواند منبع بهینه را که همان نزدیک‌ترین منبع به درخواست کاربر می‌باشد کشف کند. اساس کار کشف منبع بهینه در روش‌های درختی قبلی به این صورت بود که کشف منبع ابتدا اولویت را به فرزندان یک گره می‌داد، یعنی تا وقتی گرهی، منبع درخواستی را بین فرزندان خود داشت به سراغ منابع پدر خود نمی‌رفت. این کار شاید خوب باشد ولی در بعضی مواقع خوب جواب نمی‌دهد، مثلا فرض کنید یک گره در عمق ۱۰۰ یک منبع را از یک نوع در بین فرزندان خـــود دارد، در حالی که آن گره، همان منبع را از همان نوع به‌صورت محلی داشته باشد و یا این‌که در گره پدر خود و با فاصله کمتر آن منبع وجود داشته باشد، دراین صورت اگـــر بخواهد از منبع موجود در عمـــق پایین‌تر فرزند خود برای اخذ منبع اقدام کند ضرر کرده است چون ممکن است همـــان منبع در مکانی نزدیک‌تر در گره‌های بالاتر و یا نزد پدرش موجود باشد. در روش‌های کشف محلی، منبعی که در عمق ۱۰۰ هست کشف می‌شود ولی ما با یک روش بهتر منبع را کشف می‌کنیم. بدین شکل که هر درخواست قبل از کشـــف و طی کردن مسیر کشف ابتدا با گره پدر مشورت می‌کنـــد و در صورت به صرفه بودن عملیات کشـــف این کار را انجام می‌دهد. با این کار می‌توان بیشتر اوقات، منبع بهینه را برای کاربر کشف کرد چرا که پدر هر گره اطلاعات



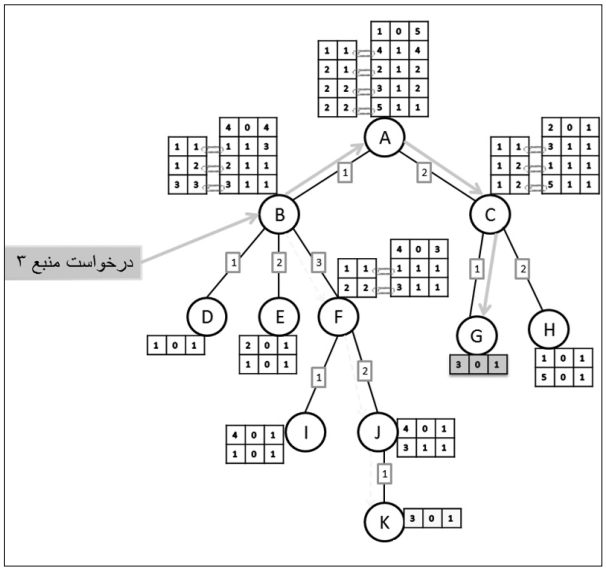
شکل ۶: قالب کدگذاری منابع در NORD_TREE

کاملـــی در مورد فرزنـــدان خود دارد و همین امر ســـبب می‌شود که مشـــورت کردن با گره پدر اغلب موفقیت‌آمیز باشد و این موفقیت آمیز بودن از آن جهت است که وقتی با روش‌های قبلی آن را مقایسه می‌کنیم متوجه می‌شویم که روش‌های قبلی لزوما منبع بهینه را برای ما پیدا نمی‌کردند، اما با پیاده سازی این روش نزدیک‌ترین منبع کشف خواهد شد. طبق شـــکل ۶ از قالب زیر برای کدگذاری و شناسایی هر منبع استفاده می‌کنیم.

همان‌طور که درشـــکل ۷ نشـــان داده شـــده است، یک درخواست از نوع سیســـتم عامل یونیکس به گره B وارد می‌شـــود این منبع در بین فرزنـــدان B و در گره K وجود دارد ولی یک منبع از این نوع هم در گره C قابل دســـترس اســـت با این تفاوت که نشـــانی C در گره A قرار دارد. اگر درخواســـت بخواهد منبع موجود در گره K را کشـــف کند ضرر کرده اســـت چون این منبع لزوما بهترین منبع نیست در حالی‌که منبع موجود در گره C به گره B نزدیک‌تر است.

روش‌های قبلی درختی کشف مبنع در این حالت، گره K را کشف می‌کردند که منبع بهینه نبود، ولی روش NORD_ TREE با یک مقایســـه در پدر که همان مشورت با پدر هم هست منبع بهینه را که نزدیک‌ترین منبع است کشف می‌کند که همان گره C می‌باشـــد چون از گره B تا گره K ده گام فاصله اســـت (ســـطح چهارم)، ولی از گره B تا گره C دو گام فاصله اســـت (سطح دوم)، پس گره C منبع بهینه برای درخواست در گره B می‌باشد تا منبع را از آن گره ذخیره کند.

زمانی که درخواســـت کشف منبع از گرهی غیر ریشه صادر شـــود، بخش دوم کار خود را شـــروع می‌کند. این مرحله طبق شـــکل ۸ خبر ذخیـــره از گرهی که منبع در آن وجود دارد را به ســـمت ریشه می‌فرســـتد به‌طوری که با

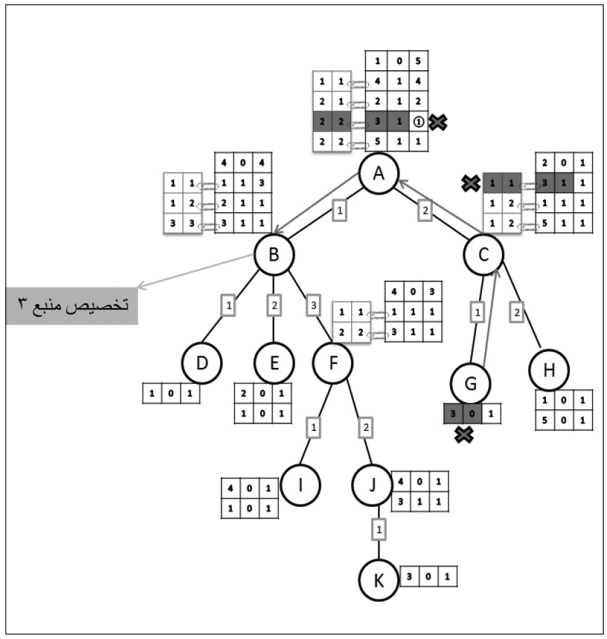


شکل ۷: بخش کشف در روش NORD_TREE

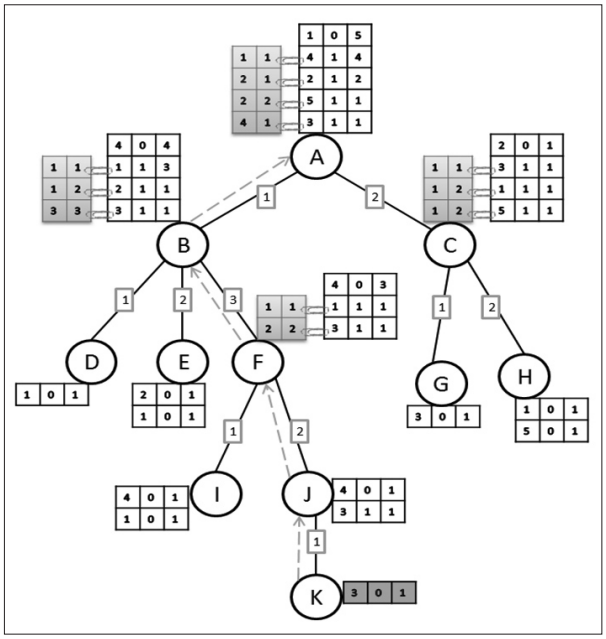
بخش کشف از گرهی که در آن درخواست کشف به سمت برگ‌های درختی آن می رود، همزمان است.

در بخش دوم از مرحله به‌روزرســـانی منبع، به خاطر این‌که بعد از کشـــف مشخص اســـت که کدام گره منبع را دارد همزمان خبر کشف آن گره به ریشه فرستاده می‌شود تا اطلاعات از پدر آن منبع تا ریشـــه حذف شود. وقتی یک گره درخواســـت کشف منبع را صادر می‌کند اگر منبع بین فرزندان آن گره وجود داشت گره مورد نظر یک درخواست به ســـمت پایین برای کشف منبع مورد نظر و همزمان یک درخواســـت پاکسازی به سمت ریشـــه می‌فرستد. در این بخش درخواست پاکسازی در مسیر به سمت ریشه می‌رود و اسم منبعی که قرار است کشف شود را از جدول مربوط به آن منبع در هر گره در مسیر خود حذف می‌کند. در این مرحله بخش کشف و پاکسازی با یکدیگر، همپوشانی دارند و همزمان انجام می‌شوند.

طبق شـــکل ۹ بعد از آن‌که منبع درخواســـتی کشـــف و تخصیص داده شد از مقدار شـــمارنده آن یک واحد کاسته می‌شود و اگر شمارنده به صفر برسد نشان دهنده این است که از آن نوع منبع دیگر در شبکه وجود ندارد و از عمل جست وجو جلوگیری می‌شود. در غیر این‌صورت، اگر شمارنده آن نوع منبع عددی غیر از صفرباشد، آنگاه جست و جو مجدداً



شکل ۸: بخش تخصیص و به‌روزرسانی در روش NORD_TREE



شکل۹: به‌روزرسانی نهایی NORD_TREE

از پایین فقط برای همان نوع منبع انجام می‌شود و منبع جدید جایگزین منبع قبلی می‌شود و سطرهایی از جداول که آن منبع را داشتند مقدارشان به‌روز می‌شود و به‌روزرسانی نهایی برای آن منبع بعد از تخصیص نشان داده می‌شود.

۳-۲- بهینه‌سازی روش پیشنهادی با الگوریتم ازدحام ذرات

اساسی‌ترین بخش کشف منابع تعداد گام‌هایی است که تا رسیدن به منبع مورد نظر باید طی شود. هدف اصلی این است که با ترکیب روش پیشنهادی با الگوریتم بهینه‌سازی ازدحام ذرات تعداد گام‌های کشف منبع را کاهش داده و منبع بهینه را شناسایی کنیم. برای هر گره باید یک تابع برازندگی در نظر گرفت که گره‌ها بر اساس ملاک شایستگی که دارند انتخاب شوند. هدف اصلی این است که با افزایش تعداد درخواست منبع و بررسی تعداد منابع قابل دسترس، مجموع تقاضاها کاهش یابد. به همین دلیل در این مقاله، هزینه همان اولویت گام‌ها در نظر گرفته شده است که برای هر گره از معادله (۱) محاسبه می‌شود:

$$ZP=1/((ni*nr))*abs(sum(RDEM(:))) \qquad (۱)$$

که در آن ZP موقعیت هر ذره است، ni تعداد درخواست منابع و nr تعداد منابع قابل دسترس می‌باشد و (sum(RDEM(:))) ماتریس مجموع تقاضا برای منبع مورد نظر است. برای تمام گره‌ها هزینه محاسبه می‌شود و هر گره که هزینه کمتری داشته باشد انتخاب می‌شود. به‌طور مثال اگر مشتری وارد شبکه شد و تقاضای منبعی را داشت و این منبع در دسترس نبود، آنگاه الگوریتم ازدحام ذرات مشخص می‌کند که اولویت با کدام گره بعدی باشد که قرار است به آن مراجعه شود.

۳-۳- الگوریتم ازدحام گروهی ذرات

تعداد ذرات یا همان جمعیت اولیه ۱۰ در نظر گرفته می‌شود و در ابتدا این جمعیت به صورت تصادفی با توزیع یکنواخت به هر ذره تخصیص داده می‌شود و تعداد گام‌ها برای ذرات بهینه شده محاسبه می‌شود. حال باید میزان هزینه هر یک از ذرات را تعیین کنیم. طبق معادله (۲) هزینه هر ذره به شکل زیر محاسبه می‌شود:

$$fit=sum([U.dis])+ZP*(sum(RDEM(:)) \qquad (۲)$$

که در آن کل هزینه برابر کل تعداد گام‌هایی است که محاسبه می‌شود. sum([U.dis]) ماتریس مجموع فاصله تا گره مورد نظر می‌باشد، ZP موقعیت هر ذره (گره) است و (sum(RDEM(:)) ماتریس مجموع تقاضا برای منبع مورد نظر می‌باشد.

پس از محاسبه مقدار هزینه هر ذره، مقدار سرعت هر ذره و موقعیت هر ذره با اســتفاده از معادلات (۳) و (۴) به روز می‌شود:

$$par(i).v=W*par(i).v+\cdots C1*rand(size.x).*(bpar(i).x-par(i).x)+ C2*rand(size.x).*(gpar(i).x-par(i).x) \quad (۳)$$

$$par(i).x = par(i).x + par(i).v \quad (۴)$$

کــه در آن Par(i).v ســرعت ذره i ام، W اهمیت تجربه شخصی(وزن اینرســی) و مقدار آن ۱ در نظر گرفته شده است، C1 اهمیت بهترین تجربه شخصی و مقدار آن ۲ است، C2 اهمیت بهترین تجربه کل گروه و مقدار آن ۲ می‌باشــد. برای تک تک ذرات (گره‌ها) این عمل انجام می‌شود، سپس هزینه محاســبه می‌شود و اگر شــرایط مورد نظر حاصل شد حلقه پایان می‌یابد. (شرط پایان رسیدن به تعداد تکرار ۱۰۰ می‌باشــد). در غیر این‌صورت عملیات ادامه می‌یابد و سرعت‌های جدید جایگزین می‌شود.

پس از پایــان یک دور از حلقه اصلــی مقدار W تغییر خواهد کرد و مقدار آن از معادله (۵) محاسبه می‌شود:

$$w=w*w\_RF \quad (۵)$$

که در اینجا W_RF همان ضریب کاهنده W می‌باشد و مقدار آن ۰/۹۷ در نظر گرفته شده است و در هر بار تکرار حلقه در انتها مقدار W تغییر می‌کند.

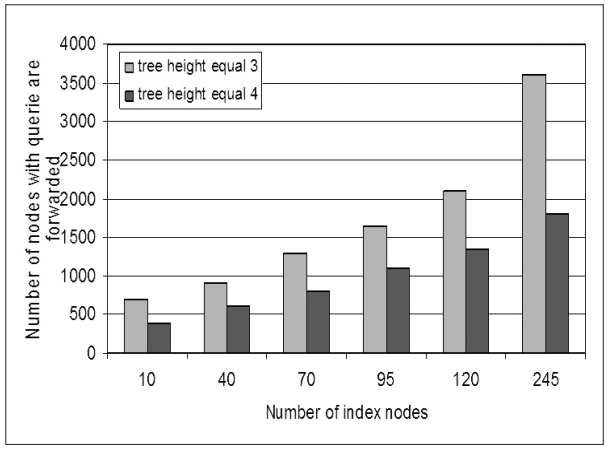
۴- پیاده‌سازی

شبیه‌ســازی در محیط متلب انجام شده است. در این بخش، ما روش پیشــنهادی خود را با روش بیان شده در [۱] و [۱۹] مقایســه کرده‌ایم. در آزمــون آزمایش اول، ما ارتفاع درخت را مانند روش‌های گفته شــده در [۱] و [۱۹]، ۳ در نظــر می‌گیریــم و تاثیر تعداد مختلف شــاخص‌های کارســاز (تعداد کارسازهایی که در مسیر جستجو توسط هر گره ملاقات می‌شــود) برای درخت روش پیشــنهادی را مشــاهده می‌کنیم. در این آزمایش فرض شده است که ۲۰۰ گره در درخت کشــف منبع وجــود دارد. ما منابع را بــه صورت تصادفی بــه هر گره اختصــاص می دهیم و درخواست‌ها را از طریق مسیر درخت ارسال کرده و تعداد گره‌هایی که در هر روش ملاقات می‌شوند را با هم مقایسه می‌کنیم. در آزمون آزمایشــی دوم مــا ارتفاع درخت‌ها را ۳ و ۴ در نظــر می‌گیریم و برای ۱۰۰ درخواســت نتایج را با استفاده از روش پیشــنهادی با هم مقایسه می‌کنیم. در آزمون آزمایشــی ســوم، تعداد کلی گره‌هایی که در طول عملیات کشف و به روزرسانی ملاقات می‌شوند را مشاهده می‌کنیم و با روش پیشــنهاد شــده در [۱] و [۱۹] مقایسه می‌کنیم. در شبیه‌ســازی بعدی تعداد واحد زمانی صرف شده برای ۳۰۰ درخواست کشف منبع را محاسبه کرده‌ایم و بــا روش‌هــای قبلی مقایســه نموده‌ایــم. همان‌طور که می‌دانیم برای محاســبه ترافیک می‌توانیم تعداد پیوندهایی را که برای کشــف منابع یا به‌روزرسانی اشغال می‌شوند محاسبه کنیم. اگر تعداد پیوندهای ملاقات شده برای کشف منابع یا به‌روزرسانی نسبت به روش‌های دیگر کمتر باشد می‌توان گفت ترافیک نسبت به روش‌های دیگر کاهش یافته اســت. در آخرین شبیه‌سازی مشاهده می‌کنیم که ترافیک ناشــی از گره‌های موجود در شــبکه نسبت به روش‌های دیگر کاهش یافته است. در این آزمون فرض شده است که ۱۰۰۰ درخواســت مختلف برای گره‌های موجود در شبکه وجود دارد که با روش‌های گذشته مقایسه می‌شود.

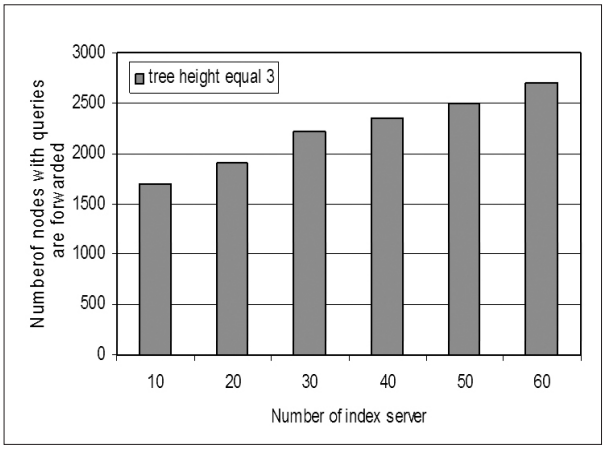
٤-۱- تنظیمات شبیه‌سازی

- تأخیر شبکه در نظر گرفته نمی‌شود.
- بــرای هر گره، درخواســت‌ها به‌صــورت تصادفی صادر می‌شود.
- عمق درخت دسترســی منابع، مانند روش‌های بیان شده در [۱] و [۱۹] برابر ۳ و ۴ است.
- برای هر پیام درخواستی، کاربر تنها یک عدد از منبع موردنظر را درخواست می‌کند.
- درخت کشف منبع ما یک درخت n تایی کامل است. به این معنی که هر گره غیر برگ باید دقیقاً n تا فرزند داشــته باشد.

در آزمــون آزمایش اول، ما ارتفاع درخت را ۳ در نظر



شکل ۱۰: مشاهده مقادیر مختلف شاخص‌های کارساز برای ۱۰۰درخواست



شکل۱۱: تعداد گره‌هایی که ۱۰۰ درخواست به آن‌ها فرستاده می‌شود برای درخت هایی با ارتفاع ۳ و۴

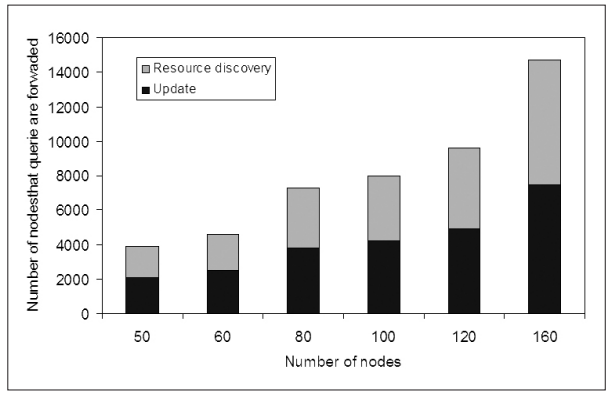
می‌گیریم و اثرات تعداد مختلف شاخص‌های کارساز برای درخت روش پیشنهادی را مشاهده می‌کنیم. در این آزمایش ۲۰۰ گره در درخت کشف منبع وجود دارد و ما این آزمایش را با ۱۰۰ درخواست انجام می‌دهیم. ما منابع را به‌صورت تصادفی به هر گره اختصاص می‌دهیم و درخواست‌ها را از طریق مسیر درخت ارسال کرده و تعداد گره‌هایی که در هر روش ملاقات می‌شوند را باهم مقایسه می‌کنیم. همان‌طور که در شکل ۱۰ می‌بینیم هنگامی‌که تعداد شاخص‌های کارساز افزایش می‌یابد، درخواست‌های کاربران به گره‌های بیشتری فرستاده می‌شود. دلیل این امر این است که ارتفاع درخت فقط ۳ در نظر گرفته‌شده است. ریشه در سطح صفر است. بقیه در سطح یک و برگ‌ها در سطح دو هستند. به‌طور مثال زمانی که برای درختی با ارتفاع ۳، تعداد سرویس‌دهنده‌های شاخص در درخت ۱۰ باشد، آنگاه ۱۰ گره در سطح یک و ۱۹۰ گره در سطح دو (۹ گره برای هر فرزند در سطح ۱) وجود خواهد داشت. درنتیجه می‌توان گفت هزینه مالی که هر شاخص کارساز باید پرداخت کند شامل یک حافظه بسیار بزرگ و یک پردازنده سریع است که بتواند تعداد فرزندان بیشتری را پشتیبانی کند.

در آزمون آزمایشی دوم ما ارتفاع درخت‌ها را ۳ و ۴ در نظر می‌گیریم و برای ۱۰۰ درخواست نتایج را با استفاده از روش پیشنهادی باهم مقایسه می‌کنیم. همان‌گونه که در شکل ۱۱ مشاهده می‌کنیم، متوجه می‌شویم که درخت با ارتفاع ۴ اجرای بهتری نسبت به درخت با ارتفاع ۳ دارد، به این دلیل که با همان تعداد از گره‌ها، یک درخت بلندتر لاغرتر است، به بیان ساده‌تر می‌توان این‌گونه گفت که هر گره، فرزند کمتری برای درخواست‌هایی که به آن‌ها فرستاده می‌شود دارد.

در آزمایش بعدی تعداد کل گره که در طول کشف و به‌روزرسانی ملاقات می‌شوند برای ۳۰۰ درخواست با استفاده از الگوریتم روش پیشنهادی مشاهده می‌کنیم. همان‌گونه که در شکل ۱۲ نشان داده‌شده است به علت این‌که عملیات کشف و به‌روزرسانی با همدیگر همپوشانی دارند درنتیجه تعداد گره‌های ملاقات شده در هر دو مرحله کشف و به‌روزرسانی از یک مقدار تقریباً یکسانی برخوردار است.

کارایی الگوریتم روش پیشنهادی را برای ۳۰۰ درخواست کشف منبع با ۲۰ بار تکرار آزمایش در جدول ۱ نشان می‌دهیم و همان‌گونه که مشاهده می‌شود، الگوریتم بهینه‌سازی ازدحام ذرات ازنظر کارایی سرعت فوق‌العاده بالایی دارد.

در شکل۱۳ با مقادیر مختلف شاخص‌های کارساز برای درختی با ارتفاع ۴ و ۳۰۰ درخواست با استفاده از روش پیشنهادی و مقایسه آن با روش‌های دیگر متوجه می‌شویم که روش پیشنهادی با استفاده از الگویتم بهینه‌سازی



شکل ۱۲: تعداد پیوندهایی که درخواست کشف و به‌روزرسانی منبع به آن‌ها فرستاده می‌شود

ازدحام ذرات تعداد میانگین شـــاخص‌های کارساز کمتری را در مسیر خود ملاقات می‌کند و نسبت به روش‌های دیگر کارایی بهتری دارد.

در روش‌هـــای پیشـــین به دلیل این‌که هـــر گره دو بار ملاقات می‌شود، یک بار برای کشف منبع و بار دیگر برای انجام عمل به‌روزرســـانی، در نتیجه تعداد گره‌های ملاقات شـــده در شـــبکه زیاد می‌شـــود ولی در روش پیشنهادی به دلیل این‌که عملیات کشـــف و به‌روزرســـانی با همدیگر همپوشـــانی دارند و هر گره تنها یکبار ملاقات می‌شـــود، تعداد گره‌های ملاقات شده، کمتر می‌شود.

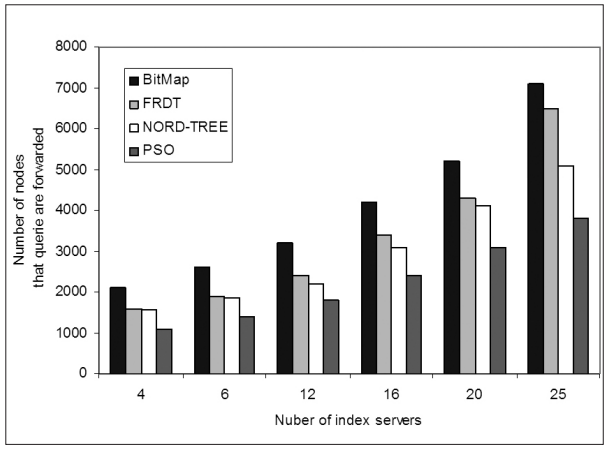
مـــا روش پیشـــنهادی خـــود را با [۱] و [۱۹] مقایســـه کرده‌ایـــم و نشـــان داده‌ایم کـــه روش پیشـــنهادی ما در محیط‌هـــای مختلف به مراتب کارایی بالاتـــری دارد، دلیل آن هم این اســـت که الگوریتم‌های دیگر، تمام منابع تطبیقی کـــه نیازهای کاربر را بـــرآورده می‌کند پیـــدا خواهد کرد بنابراین تعداد گره‌هایی که درخواست ها به آن‌ها فرستاده می‌شود از روش ما بیشتر اســـت. ابتدا فرض کرده‌ایم که ۳۰۰ درخواســـت توسط کاربران تولید می‌شود و نوع این درخواست ها تکی می‌باشد، یعنی در این آزمایش هر گره درخواست کننده منبع، فقط یک منبع نیاز دارد. در شکل۱۴ مقایسه روش پیشنهادی با روش‌های قبلی بر اساس تعداد گره‌های مختلف نشان داده شده است. در این شبیه‌سازی، تعداد گره‌های موجود در محیط را متغیر در نظر گرفته ایم تا کارایی روش‌ها در شـــرایط مختلف سنجیده شود. قابل

جدول۱: کارایی الگوریتم روش پیشنهادی با ۲۰ بار تکرار آزمایش

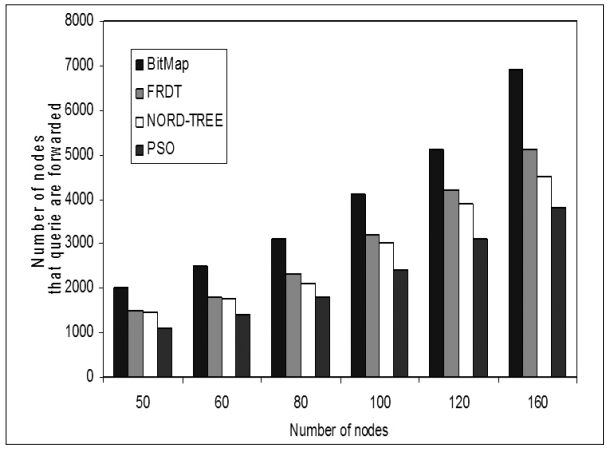
شماره آزمایش	تکرار	میانگین	بهترین هزینه	زمان/ثانیه
۱	۵۱	۳۵	۳۲	۱,۱۰۵۸
۲	۵۳	۳۵	۳۸,۸	۰,۵۷۵۱
۳	۵۱	۲۶	۳۷,۸	۰,۵۸۴۳
۴	۵۷	۲۶	۳۷,۲	۰,۵۹۸۲
۵	۵۱	۲۶	۲۹,۸	۰,۵۷۰۵
۶	۶۴	۲۶	۲۷,۸	۰,۷۶۹۴۴
۷	۵۱	۲۶	۳۱,۴	۰,۵۶۹۱۶
۸	۷۸	۲۶	۲۱,۴	۰,۸۹۸۴۲
۹	۸۲	۲۶	۲۹,۶	۰,۹۰۰۴۸
۱۰	۶۷	۳۵	۴۸,۴	۰,۷۰۲۳۲
۱۱	۵۱	۵۴	۵۹,۶	۰,۵۱۸۴۹
۱۲	۵۳	۴۵	۴۵	۰,۵۵۴۰۷
۱۳	۵۵	۲۶	۳۵,۲	۰,۵۷۸۶۵
۱۴	۵۱	۴۵	۴۵	۰,۵۴۴۴۶
۱۵	۶۶	۲۶	۳۵,۲	۰,۷۷۱۱۳
۱۶	۵۱	۲۶	۴۵	۰,۵۷۰۹۲
۱۷	۵۲	۳۵	۳۳,۴	۰,۵۶۷۱۵
۱۸	۶۴	۳۵	۳۱,۶	۰,۶۵۸۵۹
۱۹	۶۲	۳۵	۳۵	۰,۶۳۹۹۹
۲۰	۵۲	۲۶	۴۰,۶	۰,۵۴۴۱۶

ذکر است که در این آزمایش درخت به ندرت گسترش پیدا می‌کند و بزرگ می‌شود تا نتایج بررسی شود.

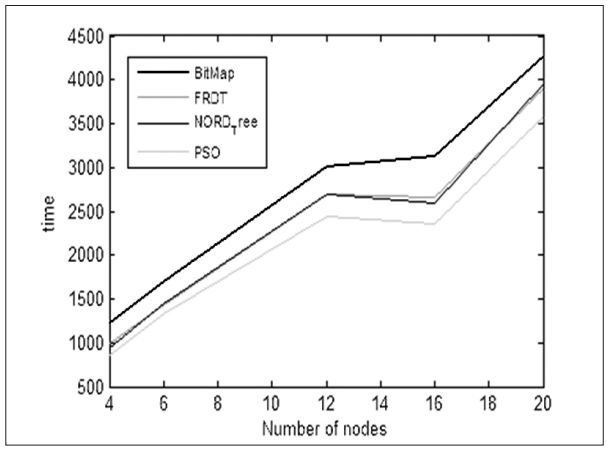
در شبیه‌سازی بعدی روش پیشنهادی(NORD-TREE) را با روش‌های گذشته مقایسه می‌کنیم و همان‌گونه که در شـــکل ۱۵ نشان داده شده است مشاهده می‌کنیم که روش ما نسبت به روش‌های گذشته تعداد واحد زمانی کمتری را برای ۳۰۰ درخواست کشف منبع سپری می‌کند و همچنین با زیاد شـــدن تعداد گره‌های محیط و بزرگ شـــدن شبکه، بر خلاف روش‌های گذشـــته، در روش پیشـــنهادی، تعداد واحدهای زمانی صرف شـــده برای کشـــف منبع افزایش نمی‌یابـــد. همان‌طور که مشـــخص اســـت، هر روشـــی که تعداد واحدهای زمانی صرف شـــده برای کشـــف منبع در آن کمتر باشد، ســـرعت بیشتری برای کشف دارد. واضح است که سرعت کشف روش پیشنهادی به دلیل حذف عمل به‌روزرسانی که عملی بسیار وقت‌گیر بود، افزایش می‌یابد. همچنین به دلیل این‌که در روش پیشنهادی، لیست پیوندی



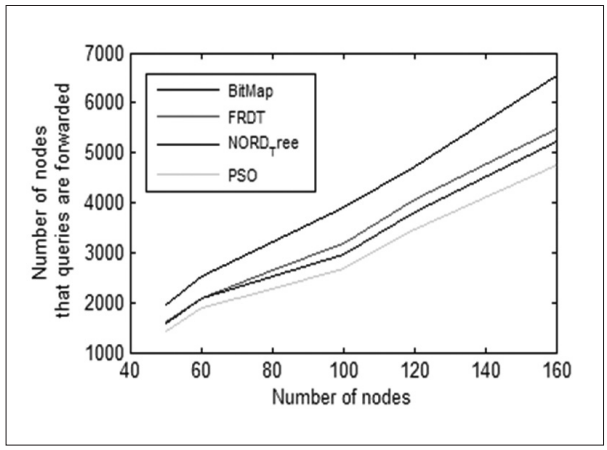
شکل ۱۳: مقایسه روش پیشنهادی و روش‌های مختلف برای ۳۰۰ درخواست با مقادیر مختلف شاخص‌های کارساز



شکل ۱۴: مقایسه روش پیشنهادی و روش‌های مختلف برای ۳۰۰ درخواست



شکل ۱۵: تعداد واحد زمانی صرف شده برای ۳۰۰ درخواست



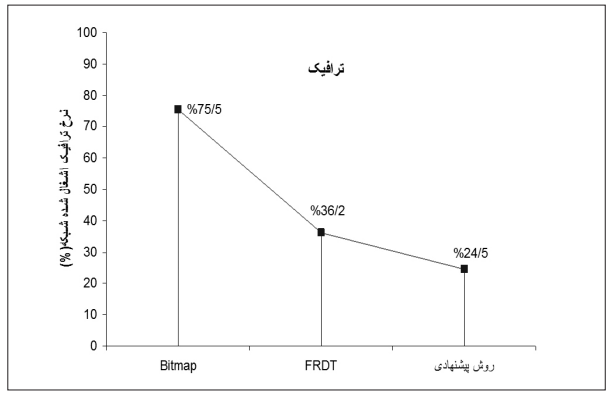
شکل ۱۶: میزان ترافیک ناشی از ۱۰۰۰ درخواست کشف منبع

بر اســـاس عمق مرتب می‌شـــود، نزدیک‌ترین منبع کشف می‌شـــود. به همین دلیل سرعت کشف بالا می رود و دیگر نیازی نیست مسیر طولانی برای کشف منبع طی کند، ولی در روش‌های پیشین بدین شکل عمل می‌کند که در صورت موجود بودن منبع در یکی از مســـیرها، عمل کشف در آن مسیر انجام می‌شود و دیگر بررسی نمی‌کند که آیا مسیری نزدیک‌تر هم وجود دارد یا خیر، و این عمل باعث می‌شود که سرعت کشـــف، کاهش یابد زیرا بهترین منبع که همان نزدیک ترین منبع است کشف نخواهد شد.

در روش [۱] محدودیتی برای پیشروی پیام دسترسی منابع تا گره‌هـــای برگ وجود نـــدارد، بنابراین تعداد پیام جستجوی بیشتری تولید می‌شود و ترافیک بیشتری ایجاد می‌گردد و کارایی و ســـرعت کاهش می‌یابد. اما در روش پیشنهادی به دلیل آن‌که مشخص است که گره بعدی برای ذخیره شدن کدام گره است در نتیجه جستجو موفقیت آمیز می‌باشد و گره‌های دیگر اشغال نمی‌شوند.

در آزمایش آخر با افزایش تعداد گره‌های درخت کشف منابع و به‌روزرســـانی، ترافیک روش پیشـــنهادی را برای گره‌های مختلف با ۱۰۰۰ درخواســـت نشـــان داده‌ایم و با روش‌های مبتنی در [۱] و [۱۹] مقایســـه کرده‌ایم. در شکل ۱۶ مشـــاهده می‌کنیم که ترافیک ناشی از روش پیشنهادی نسبت به سایر روش‌ها کمتر است.

همچنین میزان نرخ کاهش ترافیک برای ۱۰۰۰ درخواست را در شکل۱۷ نشان داده‌ایم و همان‌گونه که مشاهده می‌شود روش پیشنهادی ترافیک را تقریباً تا ۳۰٪ نسبت به روش [۱] و ۱۲٪ نسبت به روش [۱۹] کاهش داده است.



شکل۱۷: نرخ کاهش ترافیک روش پیشنهادی نسبت به روش‌های پیشین

۵- نتیجه‌گیری

روش پیشنهادی بر کاهش تعداد گره‌های ملاقات شده در اثر دسترسی منابع و افزایش سرعت جستجو، تمرکز دارد. روش پیشنهاد شده در این تحقیق مبتنی بر مدل سلسله مراتبی و کشف آگاهانه با استفاده از الگوریتم بهینه‌سازی ازدحام ذرات می‌باشد. هر گره که دارای فرزندی می‌باشد اطلاعات تمام فرزندان زیرین سطح خود را دارد. در نتیجه هنگامی که درخواستی وارد می‌شود هر گره با گره بالایی خود که نقش پدر را ایفا می‌کند مشورت می‌کند و در صورت لزوم به جای این‌که به سطوح پایین‌تر مراجعه کند و منبع را اخذ کند به سطوح بالاتر می رود و منبع نزدیک‌تر را می‌گیرد. در نتیجه تعداد مدیران منبع کمتری ملاقات می‌شود و سرعت پردازش درخواست کاربر وسرعت دسترسی منابع افزایش می‌یابد.

سپاسگزاری

با تشکر از دانشگاه آزاد اسلامی واحد ماهشهر جهت حمایت و پشتیبانی از این تحقیق.

مراجع

1- Chang RS, Hu M, 2010. A resource discovery tree using bitmap for grids. Journal of Elsevier Future Generation Computer Systems, 24:29-37.

2- Dorigo M, Stützle T, 2004. Ant colony optimization. Volume 165 Issue 2, July 2005. Pages 261 - 264. Elsevier Science Publishers Ltd..... Ann. Oper. Res. 131 (1-4) (2004) 373-395.

3- Eiben A.E, Smith A.E, 2003, Introduction to Evolutionary Computing, Springer, Natural Computing Series 1st edition, ISBN: 3-540-40184-9. Corr.

4- Foster I, Kesselman C (1997) Globus: a metacomputing infrastructure toolkit. Int J High Perform Comput Appl 11(2):115–128.

5- Glover F. and Laguna, M. 1983. Tabu Search, Kluwer Academic Publishers, 1997simulated annealing, Science, Vol. 220, No. 4598, pp. 671–680.

6- Hasanzadeh M, Meybodi MR. January 2015. Distributed optimization Grid resource discovery, Volume 71, Issue 1, pp 87-120.

7- Hashemseresht S.E, Pourhaji kazem A.A. June 2012. RD-VBT: Resource Distance Vector Binary Tree Algorithm for Resource Discovery in Grid ,International Journal of Artificial Intelligence (IJ-AI), Vol. 1, No. 2, pp. 45~52.

8- Jianhu Zhu, Fang Cheng. The Grid Resource Discovery Model based on RWSRF, International Conference on Information Sciences, Machinery, Materials and Energy (ICISMME 2015).

9- Kalogeraki V, Gunopulos D, Zeinalipour-Yazti D. 2002. A local search mechanism for peer-to-peer networks.Proceedings of 11th ACM International Conference on Information and Knowledge Management. Virginia, USA, pp: 300-307.

10- Kirkpatrick S, Gelatt C. D., and Vecchi M. P. 1983. Optimization by simulated annealing, Science, Vol. 220, No. 4598, pp. 671–680.

11- Kovacs J, Araujo A, Boychenko S, Keller M, Brinkmann A. 2013. Monitoring unicore jobs executed on desktop grid resources. Proceeding of IEEE 35th International Conference on MIPRO. Dec, PP:265-270.

12- Leavitt N. 2012. Big iron moves toward exascale computing. Journal of IEEE Computer Society, 45:14-17.

13- Liu L., Yang Y., Lian L., Shi W. 2006. "Ant Colony Optimization for SuperScheduling in Computational Grid, Services Computing, APSCC '06. IEEE AsiaPacific Conference, pp.539-545.

14- M. Kamosi, A. B. Hashemi, M. R. Meybodi. 2010. "A hibernating multi-swarm optimization algorithm for dynamic environme nts", in 2010 Second World Congress on Nature and Biologically Inspired Computing (NaBIC) , pp. 363-369.

15- Marowka A. 2011. Back to thin-core massively parallel processors. Journal of IEEE Computer Society, 44:49-54.

16- Marzolla M, Mordacchini M, Orlando S. 2005. Resource discovery in a dynamic grid environment. Proceeding of IEEE 6th International Conference on Database and Expert Systems Applications. PP:356-360.

17- Mastroianni C, Talia D, Verta O. 2005. A super-peer model for building resource discovery services in grids: design and simulation analysis. Journal of Advances in Grid Computing, 132-143.

18- Mastroianni C, Talia D, Verta O. 2008. Designing an information system for grids:comparing hierarchical, decentralized P2P and super-peer models. Journal of Elsevier Parallel Computing, 34: 593-611.

19- Mohammad Khanli L, Kargar S. 2011. FRDT: footprint resource discovery tree for grids.Journal of Elsevier Future

Mobile Computing, 3:60-68.
25-Tanenbaum A, Vansteen M. 2007. Distributed systems. 2th end, Amsterdam, Amsterdam University Publishing, pp:21-22.
26-Trunfio P, Talia D, Papadakis H, Fragopoulou P, Mordacchini M, Pennanen M. 2007. Peer-to-Peer resource discovery in Grids: Models and systems. Journal of Elsevier Future Generation Computer Systems, 23:864-878.
27- Yaghini M , Akhavan R. 2010. DIMMA: A Design and Implementation Methodology for Metaheuristic Algorithms - A Perspective from Software Development, International Journal of Applied Metaheuristic Computing, Vol.1, No.4, pp. 57-74.
28- Yonezawa Y, Kikuchi T. 1996. Ecological algorithm for optimal ordering used by collective honey bee behavior, In 7th International Symposium on Micro Machine and Human.
۲۹ - زجاجـی, غزاله؛ فاطمه مافی و احمد خادم‌زاده، ۱۳۹۰، کشـف منابع در شـبکه‌های گرید، دومین کنفرانس ملی محاسـبات نرم و فن‌آوری اطلاعات، ماهشهر، دانشگاه آزاد اسلامی واحد ماهشهر، http://www.civilica.com/Paper-NCSCIT02-NCSCIT02_176.htm
Generation Computer Systems, 27: 148-156.
20- MollaMotalebi M, Maghami R, and Ismail A.S. 2014. THRD: Threshold-based hierarchical resource discovery for Grid environments, International Journal of Computing, Springer Vienna.
21- Pandey, S., Wu, L., Guru, S., Buyya, R. 2011. "A particle swarm optimization-based heuristic for scheduling workflow applications in cloud computing environments".
22- Rahman M, Ranjan R, Buyya R, Benatallah B. 2011. A taxonomy and survey on autonomic management of applications in grid computing environments. Journal of Concurrency and Computation: Practice and Experience , 23: 1990-2019.
23- Sadashiv N, Kumar D. 2013. Cluster, grid and cloud computing: A detailed comparison.Proceeding of IEEE the 6th International Conference on Computer Science and Education,Singapore, August 3-5, PP:447-482.
24- Schulz S, Blochinger W, Poths M. 2009. A network substrate for peer-to-peer grid computing beyond embarrassingly parallel applications. Journal of IEEE Communications and