بکارگیری رقابت استعماری در بهینه‌سازی پرس‌وجو در پایگاه داده‌های رابطه‌ای

محمدرضا شیخ صادقی و فاطمه سعادت جو و علی محمد اسمعیلی زینی

آزمایشگاه کامپیوتر، گروه ارشد کامپیوتر، دانشکده فنی مهندسی، دانشگاه علم و هنر

shsadeghi@stu.sau.ac.ir و saadatjou@sau.ac.ir و esmailizaini@gmail.com

چکیده

موضوع بهینه‌سازی پرس‌وجو یک فرایند حساس به هزینه است و با توجه به تعداد جداول مرتبط در پرس‌وجو، تعداد جایگشت‌های جداول در پیوند به‌صورت نمایی رشد می‌کند که هزینه اجرای آن‌ها باهم متفاوت است. کار بهینه‌ساز پرس‌وجو انتخاب بهترین جایگشت با کمترین زمان اجرا است. بر این اساس، الگوریتم‌های مختلف تاکنون برای حل این مشکل ارائه شده است. در این مقاله به دنبال طراحی یک بهینه‌ساز پرس‌وجو هستیم و برای کاهش زمان اجرای یک پرس‌وجو از الگوریتم رقابت استعماری استفاده شده است. با استفاده از این الگوریتم زمان یافتن بهینه‌ترین طرح اجرایی ازنظر زمان در پرس‌وجو پایگاه داده رابطه‌ای نسبت به الگوریتم‌های PSO و ژنتیک در پایگاه داده رابطه‌ای کاهش یافته است.

۱ مقدمه

هدف اولیه بهینه‌ساز پرس‌وجوهای رابطه‌ای کاهش هزینه‌های عملگر پیوند است. در هر پرس‌وجو از چندین رابطه استفاده می‌شود که در نهایت پس از اجرا جواب پرس‌وجو در یک رابطه قرار می‌گیرد. بهینه‌سازی پرس‌وجو فعالیتی است که در آن از یک استراتژی کارآمد برای انجام پرس‌وجو استفاده می‌شود [۱، ۳، ۴، ۶، ۸].

الگوریتم‌های قطعی، الگوریتم‌هایی هستند که کل فضای حالت را جستجو می‌کنند. اگر بخواهیم این فضا کاهش یابد لازم است از الگوریتم‌های مکاشفه‌ای[۱] استفاده شود [۸]. مهم‌ترین ایراد الگوریتم‌های قطعی این بود که با افزایش تعداد روابط، فضای حالت جستجو به‌صورت نمایی رشد کرده و ازنظر مصرف حافظه و پردازنده به مشکل برمی‌خورد. با توجه به طبیعت الگوریتم‌های تکاملی و اینکه در اکثر مواقع مقاوم و کاراتر می‌باشند و با در نظر گرفتن کارهای صورت گرفته در این زمینه، مناسب‌ترین گزینه برای حل مسئله بهینه‌سازی پرس‌وجو، استفاده از الگوریتم‌های تکاملی می‌باشد. در [۴] مسئله بهینه‌سازی ترتیب پیوندها با استفاده از الگوریتم ژنتیک حل شده است. از ویژگی‌های دیگر این الگوریتم، قابلیت بکارگیری آن در معماری موازی می‌باشد. استفاده از الگوریتم تعمیم‌یافته بهینه‌سازی گسسته ذرات با رویکرد مقایسه چهار معیار زمان اجرای الگوریتم، زمان اجرای طرح، خطا و انحراف معیار در [۳] بیان شده است که نتایج بهتری نسبت به الگوریتم بکار رفته در [۴] دارد. در [۱] یک الگوریتم ترکیبی برای حل مسئله ترتیب عملگرهای پیوند در پرس‌وجوهای پایگاه داده‌ی رابطه‌ای پیشنهاد گردیده است. در آن از دو روش الگوریتم‌های ژنتیکی و آتاماتاهای یادگیر به‌طور همزمان برای جستجو در فضای حالت استفاده شده است. دستاورد آن مقاله این است که استفاده همزمان از آتاماتاهای یادگیر و الگوریتم‌های ژنتیکی در فرایند جستجو، سرعت رسیدن به جواب را افزایش داده و از به دام افتادن الگوریتم در مینیمم‌های محلی جلوگیری نموده است. در [۲] برای بهینه‌سازی پرس‌وجو ابتدا گراف متناظر با روابط پیوند داده‌شده طراحی می‌شود. سپس گراف را به چندین درخت تجزیه نموده و درخت‌ها را به‌صورت توزیع‌شده تحت الگوریتم ژنتیک و تپه نوردی بهینه نموده است.

در این مقاله با بکارگیری الگوریتم رقابت استعماری[۲] سعی در کاهش هزینه اجرای پرس‌وجوهای پیوند نموده‌ایم. به‌این‌ترتیب که هزینه طرح‌های اجرایی تولید شده توسط الگوریتم را ارزیابی نموده و طرحی با کمترین هزینه بر اساس ساختار الگوریتم استفاده شده به‌عنوان جواب نهایی انتخاب می‌شود. نتایج حاصل از بکارگیری الگوریتم رقابت استعماری روی یک پایگاه داده استاندارد AdventureWorks نشان داد که نسبت به سایر الگوریتم‌های مشابه دارای هزینه عملیات CPU و عملیات I/O کمتری می‌باشد.

ساختار مقاله به فرم زیر می‌باشد. در بخش دوم بهینه‌ساز پرس‌وجو بیان شده است. بیان مسئله استفاده شده در این مقاله در بخش سوم شرح داده‌شده است و در بخش چهارم نتیجه‌گیری و مقایسه آمده است.

۲ بهینه‌ساز پرس‌وجو

از بین عملگرهای رابطه‌ای موجود، پردازش و بهینه‌سازی عملگر پیوند[۳] که به‌وسیله نماد ∞ نمایش داده می‌شود، جزو پرهزینه‌ترین عملگرها ازنظر زمان اجرا است. ازآنجاکه عملگر پیوند دارای خاصیت انجمنی[۴] و جابجایی[۵] است، تعداد طرح‌های اجرایی موجود برای پاسخ‌دهی به یک پرس‌وجو با افزایش تعداد پیوندهای بین روابط، به‌صورت نمایی رشد می‌کند. گرچه تمامی طرح‌های اجرایی موجود برای پاسخ به یک پرس‌وجوی مشخص، دارای خروجی یکسان می‌باشد اما ازآنجا که تعداد سطرهای روابط میانی ایجادشده یکسان نیستند، طرح‌های اجرایی به وجود آمده دارای هزینه متفاوتی خواهند بود؛ بنابراین، انتخاب ترتیب مناسب برای اجرای عمل پیوند در هزینه کلی، تأثیرگذار است [۵].

در یك استراتژي بهينه‌سازي بر مبناي هزينه، چند طرح اجرا براي يك پرس‌وجو مشخص توليد و يك هزينه ازنظر زمان اجرا براي هر طرح محاسبه مي‌شود. بهينه‌ساز پرس‌وجو، طرحي كه داراي كمترين هزينه باشد را انتخاب مي‌كند كه يكي از مراحل اساسی در پردازش پرس‌وجو است [۴].

۳ بیان مسئله

در این مقاله هدف یافتن بهترین طرح اجرایی برای n رابطه است که باهم پیوند داده‌شده‌اند. فرض بر این است که روابط به‌صورت $R_1 \infty R_2 \infty \ldots \infty R_n$ باهم پیوند دارند. هر طرح اجرایی می‌تواند جایگشتی از روابط فوق باشد. برای مثال $R_3 \infty R_1 \infty R_2 \infty \ldots \infty R_n$ یک طرح اجرایی و $R_1 \infty R_3 \infty R_2 \infty \ldots \infty R_n$ نیز یک طرح اجرایی دیگر قلمداد می‌شود. هدف از حل مسئله بهینه‌ساز پرس‌وجو یافتن یک طرح اجرایی از بین n! طرح اجرایی موجود می‌باشد که دارای کمترین زمان اجرا باشد. تعداد سطرهای حاصل از پیوند نهایی در همه طرح‌ها یکسان است ولی تعداد سطرهای حاصل از پیوند دو جدول باهم یکسان نیست و همین مسئله باعث تغییر در زمان اجرای طرح خواهد شد.

جهت بكارگيري الگوريتم رقابت استعماري تعدادي كشور به‌صورت تصادفي ايجاد مي‌شود. هر كشور تركيبي از قرارگيري روابط پايگاه داده و يا به‌عبارت ديگر يک طرح اجرايي مي‌باشد. تعدادی از کشورهای اولیه که دارای تابع هدف کمتری نسبت به بقیه هستند را به‌عنوان استعمارگر و مابقی کشورها را به‌صورت مساوی به‌عنوان مستعمره انتخاب می‌کنیم. تابع هدف، هزینه اجرای هر دو پیوند متوالی که شامل زمان عملیات CPU و عملیات I/O است را محاسبه و باهم جمع می‌کند. تابع هدف به فرم زیر است: $F = \min \sum_{i=1}^{n-1} C_i$ که در آن C_i مجموع هزینه عملیات CPU و عملیات I/O برای پیوند $R_i \infty R_{i+1}$ می‌باشد. تابع جذب بکارگرفته شده جهت تولید کشورهای جدید از روی کشورهای موجود در ادامه آورده شده است. برای این کار با توجه به دو کشور والد، یک جدول همسایگی ایجاد می‌شود. طریقه ایجاد آن با یک مثال نشان داده شده است. فرض کنید پنج رابطه R_1, R_2, R_3, R_4, R_5 وجود دارد و $[R_1 R_2 R_3 R_4 R_5]$ نشان دهنده کشور مستعمره و $[R_1 R_2 R_5 R_3 R_4]$ نشان دهنده کشور استعمارگر است. جدول ۱ بیانگر همسایه‌های هر گره در دو کشور می‌باشد.

جدول ۱: همسایه‌های هر گره

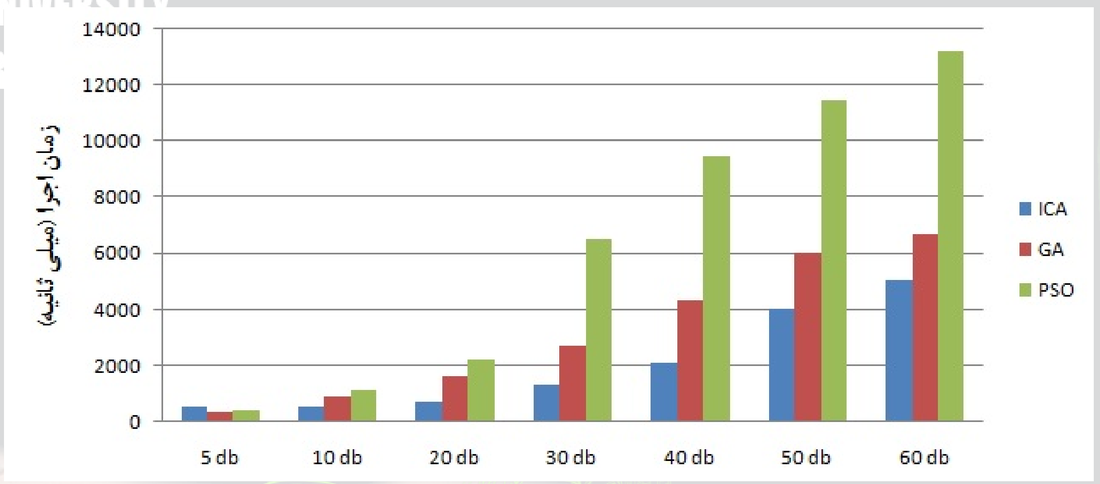
گره	گره همسایه
R_1	R_2, R_5, R_4
R_2	R_1, R_3, R_5
R_3	R_2, R_4, R_5
R_4	R_3, R_5, R_1
R_5	R_4, R_1, R_2, R_3

الگوریتم اولین گره از کشور مستعمره را انتخاب می‌کند (در اینجا R_1) با توجه به جدول ۱ همسایه‌های ملاقات شده گره R_1 در مجموعه $S = \{R_2, R_5, R_4\}$ قرار می‌گیرد. حال اگر C_{ij} نشان دهنده فاصله بین دو گره i و j یا به‌عبارت دیگر هزینه الحاق بین آن دو رابطه باشد، آنگاه گره‌های موجود در S با احتمال زیر مورد ملاقات قرار می‌گیرد که در آن i گره اصلی و j گره موجود در S است.

$$V_j = \frac{1/c_{ij}}{\sum_{j \in S}(1/c_{ij})} \qquad \forall j \in S \qquad (1)$$

فرض کنید با توجه به فرمول (۱) بیشترین احتمال مربوط به گره R_2 باشد پس گره جدید به‌صورت $[R_1 R_2 \_ \_ \_]$ خواهد شد. یک‌بار دیگر همسایه‌های R_2 از جدول ۱ بررسی می‌شود. ازآنجاکه قبلاً R_1 انتخاب شده است. لذا احتمال تنها دو همسایه R_3 و R_5 از فرمول (۱) به دست می‌آید و گره بعدی انتخاب می‌شود. این کار تا یافتن تمام گره‌های کشور جدید ادامه می‌یابد. با این کار عمل جذب برای تمام کشورها انجام می‌شود.

در اینجا مسئله بهینه ساز پرس‌وجو را برای پیوند جداول متفاوت توسط الگوریتم رقابت استعماری حل نموده‌ایم. سپس جهت مقایسه نتایج، از الگوریتم‌های GA و PSO روی همان مسئله استفاده شده است. نتایج حاصل از اجرای این الگوریتم‌ها با تعداد روابط متفاوت در شکل ۱ آورده شده است.



شکل ۱: زمان اجرای هر الگوریتم

همان‌طور که از شکل ۱ پیداست، هر چه تعداد جداول کمتر باشد الگوریتم ژنتیک نسبت به الگوریتم رقابت استعماری و PSO بهتر عمل می‌کند. با افزایش تعداد جداول در پیوند زمان اجرای الگوریتم رقابت استعماری نسبت به سایر الگوریتم‌ها بهتر خواهد شد.

۴ نتایج

یک نمونه جدول به صورت زیر است: در این مقاله از الگوریتم رقابت استعماری برای حل مسئله بهینه‌سازی ترتیب اجرای عملگرهای پیوند در پرس‌وجوهای پایگاه داده رابطه‌ای استفاده‌شده است. این الگوریتم از هزینه‌ي يک کشور با ارزيابي تابع f در متغيرهاي $(R_1, R_2, R_3, \ldots, R_n)$ برای جستجو در فضای حالات مسئله استفاده می‌نماید. هر یک از این متغییرها در مسئله بهینه‌ساز پرس‌وجو شامل یک رابطه در مسئله می‌باشد و ترتیب قرارگیری آن‌ها دقیقاً بیانگر ترتیب اجرای پیوند رابطه‌ها خواهد بود. از طرفی هزینه هر کشور برابر با مجموع هزینه عملیات CPU و عملیات I/O جهت اجرای پیوند بین متغییرها (روابط) خواهد بود. با استفاده از این الگوریتم یک دسته اولیه‌ای از این روابط تولید می‌شود و دسته‌بندی آن‌ها در قالب امپراتوری‌ها و مستعمره‌ها انجام می‌شود. با اعمال سیاست جذب از طرف استعمارگران بر روي مستعمرات و همچنين احتمال انقلاب در امپراتوری‌ها به جستجوي بهترين كشور (رابطه) پرداخته می‌شود که همان جواب مسئله بهینه‌ساز پرس‌وجو و یا به‌عبارت‌دیگر بهترین ترتیب اجرای پیوند رابطه‌ها در پرس‌وجو می‌باشد. نتایج آزمایش‌ها، برتری الگوریتم را نسبت به روش‌های مبتنی بر الگوریتم دیگر نشان می‌دهد.

مراجع

[۱] اصغری، کیوان، صفری ممقانی، علی، محمودی، فریبرز، میبدی، محمدرضا، *بهینه‌سازی اجرای پرس‌وجوها در پایگاه داده‌های رابطه‌ای با الگوریتم تکاملی ترکیبی*، مجله کامپیوتر و روباتیک ۱، ۱۳۸۷.

[۲] پارسا، سعید، ایزدخواه، حبیب و حسین زاده، امیر، *ترکیب الگوریتم ژنتیك با الگوریتم تپه نوردي در ماشین با ساختار موازي براي بهينه‌سازی پرس‌وجوهای بزرگ درگراف پرسش*، سومین کنفرانس فناوری اطلاعات و دانش، مشهد، ۱۳۸۶.

[۳] محمودی، فریبرز، زاهدی انارکی، مرتضی و زاهدی انارکی، امیرحسین، *بهینه‌سازی اجرای پرس‌وجو در پایگاه داده رابطه‌ای با استفاده از الگوریتم اکتشافی ذرات*، شانزدهمین کنفرانس ملی سالانه انجمن کامپیوتر ایران، تهران، ۱۳۸۹.

[4] K. Bennet, M. C. Ferris, Y. E. Ioannidis, *A genetic algorithm for database query optimization*, In Proc. Of the Fourth Intl. Conf. On Genetic Algorithms, San Diego, USA (1991), pp. 400–407.

[5] T. Ibaraki, T. Kameda, *On the optimal nesting order for computing n-relational joins*, Transactions on Database Systems, Vol. 9, ACM, 1984, pp. 482–502.

[6] A. A. Kolaei, M. Ahmadzadeh, *The Optimization of Running Queries in Relational Databases Using ANT-Colony Algorithm*, International Journal of Database Management Systems, Vol. 5, 2013.

[7] R. Lanzelotte, P. Valduriez, M. Zait, *On the effectiveness of optimization search strategies for parallel execution spaces*, The International Journal on Very Large Data Bases, Vol. 9, 1993, pp. 493–504

[8] M. Steinbrunn, G. Moerkotte, A. Kemper, *Heuristic and randomized optimization for the join ordering problem*,The International Journal on Very Large Data Bases, Vol. 6, 1997, pp. 191–208.

[۱] Heuristic
[۲] Imperialist Competitive Algorithm (ICA)
[۳] Join
[۴] Associative
[۵] Commutative