



## بهینه‌سازی اجرای پرس‌وجوها در پایگاه داده‌های رابطه‌ای با الگوریتم تکاملی ترکیبی

کیوان اصغری<sup>۱</sup>، علی صفری‌مقانی<sup>۱</sup>، فربرز محمودی<sup>۱\*</sup>، محمدرضا میبیدی<sup>۲</sup>

<sup>۱</sup> دانشکده مهندسی برق، رایانه و فن‌آوری اطلاعات دانشگاه آزاد اسلامی قزوین

<sup>۲</sup> دانشکده مهندسی کامپیوتر دانشگاه صنعتی امیر کبیر

دریافت مهر ۱۳۸۶، تجدید نظر بهمن ۱۳۸۶، پذیرش فروردین ۱۳۸۷

### چکیده

بهینه‌سازی پرس‌وجوهای پایگاه‌داده‌ای، یکی از مسائل تحقیقاتی مشکل است. تکنیک‌های جستجوی جامع مانند: برنامه‌نویسی پویا برای پرس‌وجوهای با تعداد روابط کم، مناسب هستند ولی با افزایش تعداد روابط موجود در پرس‌وجو، بدلیل نیاز به مصرف زیاد حافظه و پردازش، استفاده از این روش‌ها مناسب نخواهند بود، بنابراین مجبوریم از روش‌های تصادفی و تکاملی استفاده کنیم. استفاده از روش‌های تکاملی بدلیل کارایی و قدرتمندی آنها، تبدیل به ناحیه تحقیقاتی مناسبی در زمینه بهینه‌سازی پرس‌وجو گردیده است. در این مقاله یک الگوریتم تکاملی ترکیبی برای حل مساله بهینه‌سازی ترتیب اجرای عملگرهای پیوند در پرس‌وجوهای پایگاه داده‌ای پیشنهاد شده است. این الگوریتم از دو روش الگوریتم‌های ژنتیکی و آتاماتاهای یادگیر بطور همزمان برای جستجو در فضای حالات مسئله استفاده می‌نماید. در این مقاله، نشان داده شده است که با استفاده همزمان از آتاماتاهای یادگیر و الگوریتم‌های ژنتیکی در فرایند جستجو، سرعت رسیدن به جواب افزایش پیدا کرده و از بدام افتادن الگوریتم در مینیمم‌های محلی جلوگیری می‌شود. نتایج آزمایش‌ها، برتری الگوریتم ترکیبی را نسبت به روش‌های مبتنی بر الگوریتم ژنتیکی و آتاماتای یادگیر نشان می‌دهد.

**واژه‌های کلیدی:** بهینه‌سازی پرس‌وجو، عملگر پیوند، آتاماتای یادگیر، الگوریتم ژنتیکی

### ۱. مقدمه

تکنیک‌های جستجوی جامع، از لحاظ حافظه و زمان پرهزینه خواهند بود. پرس‌وجوهای با تعداد پیوند زیاد در سیستم‌های جدید مانند سیستم‌های مدیریت پایگاه داده استنتاجی<sup>۱</sup>، سیستم‌های خبره<sup>۲</sup>، سیستم‌های مدیریت پایگاه داده‌های مهندسی

مدل داده‌ای رابطه‌ای توسط کاد [۱]، معرفی شده است و در سالهای اخیر، سیستم‌های پایگاه‌داده رابطه‌ای بعنوان استاندارد در انواع کاربردهای علمی و تجاری، شناخته می‌شوند. کار بر روی عملگر پیوند، بدلیل هزینه‌های ارزیابی بالای این عملگر، هدف اولیه بهینه‌سازی پرس‌وجوی رابطه‌ای است. اگر پرس‌وجوها در حالت محاوره‌ای باشند، شامل تعداد کمی رابطه خواهد بود که بهینه‌سازی این عبارات را می‌توان بوسیله یک جستجوی جامع انجام داد. اما در صورتی که تعداد روابط بیش از ۵ یا ۶ رابطه باشد،

\* نویسنده مکاتبه‌کننده: mahmoudi@itrc.ac.ir, Tel (281)3675782

1 Deductive

2 System Expert

System-R، در اکثر مواقع دارای هزینه کمتری می‌باشد. از ویژگی‌های دیگر این الگوریتم، قابلیت بکارگیری آن در معماری موازی می‌باشد. از دیگر کارهای صورت گرفته مبتنی بر الگوریتم ژنتیکی، روش آرایه شده توسط استین‌بران و همکارانش بوده است [۷] که روشهای کدگذاری و عملگرهای ژنتیکی متفاوتی را بکار گرفته‌اند. نمونه دیگر از الگوریتم‌های تکاملی بکار گرفته شده برای مسئله بهینه‌سازی پیوندها، روش برنامه‌نویسی ژنتیکی می‌باشد که توسط استیلگر و اسپیلیوپولو [۱۶] مطرح شده است. بهینه‌ساز ژنتیکی CGO نیز توسط مولرو و همکارانش در [۱۷] آرایه شده است. در این مقاله یک الگوریتم تکاملی ترکیبی برای حل مساله بهینه‌سازی ترتیب اجرای عملگرهای پیوند در پرس‌وجوهای پایگاه داده‌ای پیشنهاد کرده‌ایم. این الگوریتم از دو روش الگوریتم‌های ژنتیکی و آتاماتاهای یادگیر بطور همزمان برای جستجو در فضای حالات مسئله استفاده می‌نماید. نشان داده شده است که با استفاده همزمان از آتاماتاهای یادگیر و الگوریتم‌های ژنتیکی در فرایند جستجو، سرعت رسیدن به جواب افزایش پیدا کرده و از بدام افتادن الگوریتم در مینیمم‌های محلی جلوگیری می‌شود. نتایج آزمایش‌ها، برتری الگوریتم ترکیبی را نسبت به روش‌های مبتنی بر الگوریتم ژنتیکی و آتاماتای یادگیر نشان می‌دهد.

ادامه مقاله بدین صورت سازماندهی شده است که بخش دوم، مساله ترتیب عملگرهای پیوند در پرس‌وجوهای پیوندی را تعریف می‌کند. توضیح مختصری درباره آتاماتاهای یادگیر و الگوریتم‌های ژنتیکی در بخش ۳ آمده است. در بخش ۴، الگوریتم ترکیبی پیشنهادی در این مقاله شرح داده می‌شود و در بخش ۵، نتایج آزمایش‌ها آرایه می‌گردد. بخش ۶ نیز شامل نتیجه‌گیری می‌باشد.

## ۲. تعریف مسئله

بهینه‌سازی پرس‌وجو، فعالیتی است که طی آن، یک طرح کارا برای اجرای پرس‌وجو (qep)<sup>۳</sup>، تولید می‌گردد و یکی از مراحل اساسی در پردازش پرس‌وجو می‌باشد. در این مرحله، سیستم مدیریت پایگاه داده از بین تعدادی طرح اجرا، بهترین طرح را بر می‌گزیند. بگونه‌ای که اجرای پرس‌و-

(CAD/CAM)، سیستم‌های پشتیبانی تصمیم<sup>۴</sup>، داده‌کاوی<sup>۵</sup> و سیستم‌های مدیریت پایگاه‌داده‌های علمی و ... دیده می‌شوند. به هر حال سیستم‌های مدیریت پایگاه‌داده<sup>۶</sup>، نیازمند بکارگیری تکنیک‌های بهینه‌سازی پرس‌وجو با هزینه کم برای مقابله با چنین پرس‌وجوهای پیچیده‌ای هستند. دسته‌ای از الگوریتم‌های جستجوی ترتیب مناسب برای اجرای عملگرهای پیوند، الگوریتم‌های قطعی می‌باشند که کل فضای حالات را بصورت کامل جستجو می‌کنند و بعضاً با بکارگیری روشهای مکاشفه‌ای<sup>۷</sup>، این فضا را کاهش می‌دهند [۷]. یکی از این الگوریتم‌ها، روش برنامه‌نویسی پویاست<sup>۸</sup> که اولین بار برای بهینه‌سازی ترتیب پیوند در System-R توسط سلینگر و همکارانش مطرح گردید [۹، ۲۰]. مهمترین ایراد الگوریتم این است که با افزایش تعداد روابط موجود در پرس‌وجو، نیازمند مصرف زیاد حافظه و پردازنده می‌باشد. از دیگر الگوریتم‌های قطعی می‌توان به الگوریتم حداقل قابلیت انتخاب<sup>۹</sup> [۷]، الگوریتم KBZ [۱۰] و الگوریتم AB [۱۱] اشاره کرد. جهت نشان دادن ناتوانی و ضعف الگوریتم‌های قطعی در مقابله با پرس‌وجوهای بزرگ، الگوریتم‌های دیگری به نام الگوریتم‌های تصادفی معرفی شده‌اند. الگوریتم‌های مطرح شده در این زمینه، الگوریتم بهبود مکرر<sup>۱۰</sup> [۵، ۶، ۱۲]، الگوریتم نرم کردن شبیه‌سازی شده<sup>۱۱</sup> [۵، ۱۲، ۱۳، ۱۴]، بهینه‌سازی دو مرحله‌ای<sup>۱۱</sup> [۱۲]، نرم کردن شبیه‌سازی شده گردشی<sup>۱۲</sup> [۸] و نمونه‌گیری تصادفی<sup>۱۳</sup> [۱۵] می‌باشد.

با توجه به طبیعت الگوریتم‌های تکاملی و اینکه در اکثر مواقع مقاوم و کارا تر می‌باشند و با در نظر گرفتن کارهای صورت گرفته در این زمینه، مناسب‌ترین گزینه برای حل این مسئله، استفاده از الگوریتم‌های تکاملی می‌باشد. اولین کار انجام شده بر روی مسئله بهینه‌سازی ترتیب پیوندها با استفاده از الگوریتم ژنتیکی، توسط بنت و همکارانش آرایه گردید [۳]. در حالت کلی الگوریتم بکار رفته توسط آنها در مقایسه با الگوریتم برنامه‌نویسی پویای بکار رفته برای

3 Decision Support system- DSS

4 Data Mining

5 Database Management System-DBMS

6 Heuristic

7 Dynamic Programming

8 Minimum Selectivity

9 Iterative Improvement -II

10 Simulated Annealing

11 Two -Phase OPTimization

12 Toured Simulated Annealing

13 Random Sampeling

14 Query Execution Plan